UNIVERSITAT POLITÈCNICA DE CATALUNYA

DOCTORAL THESIS

# Large-scale tree-based unfitted finite elements for metal additive manufacturing


*Author:*
Eric NEIVA

*Supervisors:*
Prof. Santiago BADIA
Prof. Michele CHIUMENTI


*A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy*

*in the*

Doctorat en Enginyeria Civil

Escola Tècnica Superior d'Enginyeria de Camins, Canals i Ports de Barcelona

Barcelona, September 2020

*To Cristina, Cel and Joca*

"All men by nature desire to know."

Aristotle in *Metaphysics*

# *Abstract*

This thesis addresses large-scale numerical simulations of partial differential equations posed on evolving geometries. Our target application is the simulation of metal additive manufacturing (or 3D printing) with powder-bed fusion methods, such as Selective Laser Melting (SLM), Direct Metal Laser Sintering (DMLS) or Electron-Beam Melting (EBM). The simulation of metal additive manufacturing processes is a remarkable computational challenge, because processes are characterised by multiple scales in space and time and multiple complex physics that occur in intricate three-dimensional growing-in-time geometries. Only the synergy of advanced numerical algorithms and high-performance scientific computing tools can fully resolve, in the short run, the simulation needs in the area.

The main goal of this Thesis is to design a a novel highly-scalable numerical framework with multi-resolution capability in arbitrarily complex evolving geometries. To this end, the framework is built by bringing together three computational tools: (1) parallel mesh generation and adaptation with forest-of-trees meshes, (2) robust unfitted finite element methods and (3) parallel finite element modelling of the geometry evolution in time. Our numerical research is driven by several limitations and open questions in the state-of-the-art of the three aforementioned areas, which are vital to achieve our main objective. All our developments are deployed with high-end distributed-memory implementations in the large-scale open-source software project FEMPAR. In considering our target application, (4) temporal and spatial model reduction strategies for thermal finite element models are investigated. They are coupled to our new large-scale computational framework to simplify optimisation of the manufacturing process.

The contributions of this Thesis span the four ingredients above. Current understanding of (1) is substantially improved with rigorous proofs of the computational benefits of the 2:1 k-balance (ease of parallel implementation and high-scalability) and the minimum requirements a parallel tree-based mesh must fulfil to yield correct parallel finite element solvers atop them. Concerning (2), a robust, optimal and scalable formulation of the aggregated unfitted finite element method is proposed on parallel tree-based meshes for elliptic problems with unfitted external contour or unfitted interfaces. To the author's best knowledge, this marks the first time techniques (1) and (2) are brought together. After enhancing (1)+(2) with a novel parallel approach for (3), the resulting framework is capable of mitigating a major performance bottleneck in large-scale simulations of metal additive manufacturing processes by powder-bed fusion: scalable adaptive (re)meshing in arbitrarily complex geometries that grow in time. Along the development of this Thesis, our application problem (4) is investigated in two joint collaborations with the Monash Centre for Additive Manufacturing and Monash University in Melbourne, Australia. The first contribution is an experimentally-supported thorough numerical assessment of time-lumping methods, the second one is a novel experimentally-validated formulation of a new physics-based thermal contact model, accounting for thermal inertia and suitable for model localisation, the so-called virtual domain approximation.

By efficiently exploiting high-performance computing resources, our new computational framework enables large-scale finite element analysis of metal additive manufacturing processes, with increased fidelity of predictions and dramatical reductions of computing times. It can also be combined with the proposed model reductions for fast thermal optimisation of the manufacturing process. These tools open the path to accelerate the understanding of the process-to-performance link and digital product design and certification in metal additive manufacturing, two milestones that are crucial to exploit the technology for mass-production.

# *Resum*

Aquesta tesi tracta la simulació a gran escala d'equacions en derivades parcials sobre geometries variables. L'aplicació principal és la simulació de procesos de fabricació additiva (o impressió 3D) amb metalls i per mètodes de fusió de llit de pols, com ara Selective Laser Melting (SLM), Direct Metal Laser Sintering (DMLS) o Electron-Beam Melting (EBM). La simulació d'aquests processos és un repte computacional excepcional, perquè els processos estan caracteritzats per múltiples escales espaitemporals i múltiples físiques que tenen lloc sobre geometries tridimensionals complicades que creixen en el temps. La sinèrgia entre algorismes numèrics avançats i eines de computació científica d'alt rendiment és la única via per resoldre completament i a curt termini les necessitats en simulació d'aquesta àrea.

El principal objectiu d'aquesta tesi és dissenyar un nou marc numèric escalable de simulació amb capacitat de multiresolució en geometries complexes i variables. El nou marc es construeix unint tres eines computacionals: (1) mallat paral·lel i adaptatiu amb malles de boscs d'arbres, (2) mètodes d'elements finits immersos robustos i (3) modelització en paral·lel amb elements finits de geometries que creixen en el temps. Algunes limitacions i problemes oberts en l'estat de l'art, que són claus per aconseguir el nostre objectiu, guien la nostra recerca. Tots els desenvolupaments s'implementen en arquitectures de memòria distribuïda amb el programari d'accés obert FEMPAR. Quant al problema d'aplicació, (4) s'investiguen models reduïts en espai i temps per models tèrmics del procés. Aquests models reduïts s'acoplen al nostre marc computacional per simplificar l'optimització del procés.

Les contribucions d'aquesta tesi abasten els quatre punts de dalt. L'estat de l'art de (1) es millora substancialment amb proves riguroses dels beneficis computacionals del 2:1 balancejat (fàcil paral·lelització i alta escalabilitat), així com dels requisits mínims que aquest tipus de mallat han de complir per garantir que els espais d'elements finits que s'hi defineixin estiguin ben posats. Quant a (2), s'ha formulat un mètode robust, òptim i escalable per agregació per problemes el·líptics amb contorn o interfases immerses. Després d'augmentar (1)+(2) amb un nova estratègia paral·lela per (3), el marc de simulació resultant mitiga de manera efectiva el principal coll d'ampolla en la simulació de processos de fabricació additiva en llits de pols de metall: adaptivitat i remallat escalable en geometries complexes que creixen en el temps. Durant el desenvolupament de la tesi, es col·labora amb el Monash Centre for Additive Manufacturing i la Universitat de Monash de Melbourne, Austràlia, per investigar el problema d'aplicació. En primer lloc, es fa una anàlisi experimental i numèrica exhaustiva dels mètodes d'aggregació temporal. En segon lloc, es proposa i valida experimentalment una nova formulació de contacte tèrmic que té en compte la inèrcia tèrmica i és adequat per a localitzar el model, l'anomenada aproximació per dominis virtuals.

Mitjançant l'ús eficient de recursos computacionals d'alt rendiment, el nostre nou marc computacional fa possible l'anàlisi d'elements finits a gran escala dels processos de fabricació additiva amb metalls, amb augment de la fidelitat de les prediccions i reduccions significatives de temps de computació. Així mateix, es pot combinar amb els models reduïts que es proposen per l'optimització tèrmica del procés de fabricació. Aquestes eines contribueixen a accelerar la comprensió del lligam procés-rendiment i la digitalització del disseny i certificació de productes en fabricació additiva per metalls, dues fites crucials per explotar la tecnologia en producció en massa.

# *Acknowledgements*

I would like to gratefully acknowledge the guidance, support and encouragement of my two advisors, Santiago Badia and Michele Chiumenti. Thank you very much for your patience, availability and for pushing me forward through these years. I extend my gratitude to Alberto and Francesc. It has been the greatest pleasure and privilege to work together with and learn from such brilliant and commited research professionals. I have learned so many lessons from you four. I hope they have made me grow up a little bit as a researcher.

I have had the luck to carry out my thesis in an outstanding research group. Thank you Javier, Oriol, Alba, Marc, Jesús, Pere Antoni, Víctor, Hieu, Àlex, Manuel, Jerrad and Daniel for making up for a very pleasant, supportive and productive working environment. I am very grateful to Miguel, Emilio, Joan, Zhuoer and Gabriele, who I had also the pleasure to collaborate with. I also thank Txema and Louis for showing appreciation and interest for my work.

I would like to thank all my mentors in my early years, whose support and valuable lessons I carry with me everyday: Mari T, Ascensión S, Joan Manel B, Sílvia P, Joan G, Maria Alba M. Thank you Antonio R-F, Marina G and Xavier A for leading me to study the topic of my thesis. I would also like to acknowledge all my English teachers and the programming lessons from Salvador R. Thank you, Sílvia A, Antoni M and all UPC and AGAUR staff, who have kindly helped me get through all the administrative procedures.

I would also like to express my gratitude to the public education system and I beg everybody to preserve it at any cost. I also acknowledge the support received from the Generalitat de Catalunya through a FI fellowship (2017 FI-B-00219) and the CAxMan Project (680448) funded under the Horizon 2020 EU framework programme.

I want to thank Maria, Pontus and Eva for their warm welcome into Cal Monjo. Many thanks to the Ajuntament de la Vall d'En Bas for setting up such a wonderful and unique coworking space. I also want to thank all the people that helped me in Melbourne and contributed to make me feel comfortable from day one at Monash: John, Óscar, Abbas, Angelika, Vincent and Ricardo. I am very grateful to my friends in Australia and the happy times I spent with them: Richard, Ryan, Matthew, Steph, Archie, Qing and Johnson. Richard, thank you very much for your kind hospitality and selfless support.

I want to thank all my friends from Uni and Lleida: Ana L, Albert, Blanca, Cesc, Irene, Leire, Lidia, Lleó, Marc, Marina, Montse, Pepe, Salo, Tomás, Ana M, Andreu, Ferran, Gemma, Israel, Ivan, Jenni, Mar, Martí, Daniel, Guille, Manel, Raül, Santi, Anuar, Jessi, Johnny, Miri, Noa and my beloved godson Hugo. I have had so much (often crazy) fun and memorable moments with you. Without you around, my life would be so dull and forgettable...

I am very grateful to Aurora, Joan, Irene, Càndida, Núria, Emili and all of my husband's relatives, who I have met during my PhD studies and have kindly received me as part of their family since the very beginning. Thank you, Eva (and family), Jordi, Padrinho and Avó for always being there and checking up on me every now and then. To my dearest Irene: it's about time we visit Taiwan. To my dearest Lluïsa (and family): "un plaer, com sempre". I love you both. Bro, I could not be more proud of you and I cannot wait to see what becomes of that tiny little ill-tempered toddler. Thanks for putting up with me. Thank you, Dad, for being my very first teacher. Thank you, Mom, for teaching me perseverance and resilience. Thank you both for your enormous sacrifice. To my dear Carles-Hug, your critical thinking and methodical ways have been my main source of inspiration throughout my thesis. Above all, you have made me a very happy man, like no other. Thank you!

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# List of Abbreviations

| | |
|---|---|
| **AM** | **A**dditive **M**anufacturing |
| **PBF** | **P**owder-**B**ed **F**usion |
| **HPC** | **H**igh-**P**erformance **C**omputing |
| **AMR** | **A**daptive **M**esh **R**efinement |
| **FE** | **F**inite **E**lement |
| **PDE** | **P**artial **D**ifferential **E**quation |
| **SFC** | **S**pace-**F**illing **C**urve |
| **FEM** | **F**inite-**E**lement **M**ethod |
| **MCAM** | **M**onash **C**entre for **A**dditive **M**anufacturing |
| **MU** | **M**onash **U**niversity |
| **TM** | **T**etrahedral **M**orton |
| **DOF** | **D**egree **O**f **F**reedom |
| **VEF** | **V**ertex, **E**dge and **F**ace |
| **AMG** | **A**lgebraic **M**ultigrid |
| **OO** | **O**bject-**O**riented |
| **DG** | **D**iscontinuous **G**alerkin |
| **CG** | **C**ontinuous **G**alerkin |
| **MPI** | **M**essage **P**assing **I**nterface |
| **SLM** | **S**elective **L**aser **M**elting |
| **CAD** | **C**omputer-**A**ided **D**esign |
| **DMLS** | **D**irect **M**etal **L**aser **S**intering |
| **EBM** | **E**lectron **B**eam **M**elting |
| **HAZ** | **H**eat **A**ffected **Z**one |
| **HAV** | **H**eat **A**ffected **V**olume |
| **HTC** | **H**eat **T**ransfer **C**oefficient |
| **DD** | **D**omain **D**ecomposition |

| | |
|---|---|
| **BDDC** | **B**alancing **D**omain **D**ecomposition by **C**onstraints |
| **HF** | **H**igh **F**idelity |
| **PP** | **P**art-**P**late |
| **P** | **P**art-**O**nly |
| **VDA** | **V**irtual **D**omain **A**pproximation |
| **EBM** | **E**lement-**B**irth **M**ethod |
| **HST** | **H**yperplane **S**eparation **T**heorem |
| **PCG** | **P**reconditioned **C**onjugate **G**radient |
| **BVP** | **B**oundary **V**alue **P**roblem |
| **CGM** | **C**onjugate **G**radient **M**ethod |
| **LB** | **L**i and **B**ettess |
| **OB** | **O**ñate and **B**ugeda |

# Chapter 1

# Introduction

## 1.1 Motivation

Additive manufacturing (AM), also known as 3D Printing, is emerging as a game-changer manufacturing technology. AM refers to a myriad of manufacturing processes that create three-dimensional objects, by means of joining or solidifying material under computer control, typically layer by layer. Many industrial sectors, including the aerospace, defence, dental or biomedical, are putting vast efforts into leveraging the potential of AM to produce objects with unprecedented shapes and enhanced properties in short time-to-market [45].

However, widespread industrial adoption of AM technologies, especially for metals, cannot be long-term sustained without a software ecosystem equipped with fast and predictive computer simulation tools. Nowadays, the biggest hurdles are rather poor understanding of the physical mechanisms, governing the manufacturing process, and lack of control of the countless factors affecting final quality and performance [192]. To move from prototypes and demonstrators to real industrial use and exploitation, one needs to document and certify the quality of the outcomes of AM processes, such as product strength, surface quality, material behaviour and shape constraints. It comes as no surprise that current practice to meet these requirements is mostly restricted to time-consuming and expensive trial-and-error physical experimentation, involving the production of hundreds of copies of the final product. By contrast, accurate computer-aided simulations could enormously help overcoming these bottlenecks. On the one hand, it could speed up the understanding of the process-structure-property-performance link to unlock the full potential of AM applications. On the other hand, it could enable to shift to a digital design and certification paradigm, in which engineers would be able to certify *before* fabrication, leading to faster and cheaper product design in AM.

This thesis centres upon the modelling and numerical simulation for metal AM processes by powder-bed fusion (PBF) methods (see graphical description in Figure 6.1). Experience gained in modelling conventional manufacturing processes, such as casting or welding, has been successfully reinterpreted to generate the first meaningful numerical AM-PBF models [8, 33, 109, 127, 159]. Nonetheless, compared to traditional

processes, mathematical modelling of AM-PBF processes poses a wholly new and exceptional computational challenge. Indeed, it involves dealing with multiple space-time scales and multiple physics, e.g. coupled thermomechanics at the component [cm-mm,h], fluid dynamics at the melt pool [µm,ms] and solidification of the microstructure [µm,h]. On top of this, an industrial-ready numerical framework must handle arbitrarily complex and growing-in-time three-dimensional geometries, which can even have thin features, such as lattice structures. It follows that the simulation of AM-PBF processes is extremely complex and dramatically expensive, due to the level of refinement required at the small scales to capture the deposition zone, the very large number of time steps required to complete a full simulation and the difficulty of producing body-fitted meshes tracking the growth of the domain at all times.

To illustrate the computational challenge, let us consider a brute force direct numerical FE simulation of a whole AM process on structured meshes. A laser metal deposition simulation would easily involve a 100mm $\times$ 100mm $\times$ 100mm build size, a 50µm resolution mesh and layer thickness, a 50µs time step and 50h of laser scanning time. Using uniform space and time grids, it would lead to $4 \cdot 10^6$ FEs per layer, $8 \cdot 10^9$ FEs in total and $3.6 \cdot 10^6$ time steps. Clearly, such brute-force simulations are out of reach now and will still be in the future, even with an efficient exploitation of the forthcoming exascale supercomputers. Only the synergy of *novel* (a) geometrical treatment of evolving geometries, (b) advanced multiscale and multiphysics numerical algorithms with adaptive meshes and (c) highly scalable implementations can provide short-term answers to AM simulation needs. These tools will clear the path for efficient and robust fully-resolved AM simulations in a time scale compatible with the time-to-market of AM. Eventually, the combination of advanced numerical algorithms and high-performance computing (HPC) resources will enable the use of AM at the mass-production level.

## 1.2 Thesis objectives

The main objective of this thesis is to *design, for the first time, an abstract numerical framework, suitable to complex evolving geometries, through high-performance computing tools*. Our driving application is the part-scale heat transfer analysis of metal AM processes by powder-bed fusion. We aim to show that, by efficiently exploiting HPC resources, our new computational tools enable large-scale FE analysis of AM-PBF processes. In particular, they are able to provide increased fidelity of predictions and dramatical reductions of CPU times.

Scalable multiresolution capability in arbitrarily complex evolving geometries is the key attribute for the sought-after numerical framework. To achieve this, our methodology is grounded on three main building blocks (1) parallel mesh generation and adaptation with forest-of-trees meshes, (2) robust unfitted FE methods and (3) parallel FE

modelling of partial differential equations (PDEs) posed on growing geometries. After reviewing the state-of-the-art in techniques (1-3), we have identified several gaps in knowledge, that are crucial to meet the goal of this thesis, but remain to be filled or have not yet been dealt with in proper depth. As a result, they drive our numerical research and lead to several secondary objectives, see **O1-4** below. Furthermore, in the application to AM-PBF, we investigate existing and novel model reduction strategies, that can be easily coupled to our HPC framework, to enable large-scale optimisation of AM-PBF processes, see **O5-6**. All these developments (except in **O5**) are deployed with high-end distributed-memory implementations **O7** in the open-source project `FEMPAR`. In the next paragraphs, we briefly describe all the open questions that we address in this thesis. We refer to subsequent chapters for further motivation and literature review. For improved clarity, we summarise all the objectives in Table 1.1.

Let us start with the first ingredient (1) of the framework. Tree-based adaptive mesh refinement (AMR) endowed with space-filling curves (SFCs) has arisen in recent years as a petascale-capable method for parallel mesh generation and adaptation [93]. SFCs enable efficient data storage and traversal of the adaptive mesh, fast computation of hierarchy and neighbourhood relations among mesh cells, and scalable partitioning and dynamic load balancing. State-of-the-art in this technique has already proven excellent suitability to efficient large-scale FE approximation of PDE problems, characterised by multiple physical scales [7, 40, 148, 162, 174, 188]. However, current literature clearly fails to explain when and why parallel algorithms and data structures required to support generic *conforming* FE discretisations atop tree-based adaptive meshes are mathematically correct. For this reason, our first goal is to:

> **Objective 1 (O1)**
> Address *with mathematical rigour* the requirements parallel tree-based meshes must fulfil to lead to correct parallel generic finite element solvers atop them.

While research on parallel tree-based adaptive FE methods is achieving maturity, applications in arbitrarily complex geometries have barely received attention to date. Here, we bring ingredient (2) into action. Usage of *body-fitted* meshes (i.e. those whose faces conform to the domain boundary) is a rather inconvenient choice in large-scale parallel computations, since there are currently no scalable methods to generate and partition large unstructured meshes. On the other hand, *unfitted* (or *embedded* or *immersed*) FE methods, in place of requiring body-fitted meshes, embed the domain of analysis in geometrically simple background grids, e.g. uniform or adaptive Cartesian grids, which are by far more efficiently generated. Hence, unfitted methods blend naturally well with parallel adaptive tree-based meshes. Together, they could bypass the prohibitive requirement of generating adaptive body-fitted meshes in large-scale applications, such as metal AM. Despite the potential, this line of work has been widely neglected. According to this, considering the recently developed *aggregated* unfitted

finite-element method (FEM), known as agFEM [24], we propose to bridge these techniques for the first time:

> **Objective 2 (O2)**
> Formulate and analyse the (h-adaptive) aggregated finite element method on parallel tree-based meshes.

For the previous objective, our target applications are elliptic PDE problems with *unfitted contour* or *unfitted interfaces*. We observe that the agFE method is not hindered by the classical *small cut cell problem*, appearing when the intersection of a background cell with the physical domain is arbitrarily small, leading to severe ill-conditioning issues [62]. The formulation is equipped with well-behaved numerical properties that do not depend on cut location, such as stability, condition number bounds, optimal convergence and continuity with respect to data. However, until now, only unfitted contour elliptic or Stokes problems have been studied, even if multiphase problems are ubiquitous in large-scale applications. For instance, in our application context, the thermal analysis of a solid part being printed in a powder-bed is clearly a two-phase problem. Therefore, it remains to see if the same sound properties of agFEM hold for unfitted interfaces:

> **Objective 3 (O3)**
> Formulate and analyse the (h-adaptive) aggregated finite element method on $n$-interface elliptic problems.

The last ingredient (3) is critical to unlock large-scale AM-PBF applications. Considering the usual approach of discretising in space and integrating in time, the growth of the domain, following the addition of material during printing, has been typically modelled by attaching new elements into the computational mesh [136]. Until now these so-called *activation* mechanisms have solely been formulated in serial computer environments and distributed-memory extensions are not available. In fact, parallel performance and scalability have been generally neglected in AM simulations, in spite of their importance. The body of literature lacks HPC tools targeting simulation of AM processes and exploiting parallelism at all simulation stages. For this reason, we propose to:

> **Objective 4 (O4)**
> Formulate a *fully* parallel framework to deal with PDE problems posed on evolving geometries, grounded on generic unfitted FEs on tree-based adaptive meshes.

The main outcome of **O1-4** is a novel multiscale and multiphase parallel numerical framework that addresses a major performance bottleneck in large-scale simulations of AM-PBF: adaptive (re)meshing in complex geometries that grow in time. Along the work to achieve **O1-4**, we have the occasion to *simultaneously* collaborate (twice) with a group of material scientists, with long expertise in AM technologies and based at the Monash Centre for Additive Manufacturing (MCAM) and Monash University (MU) in

Melbourne, Australia. We aim to exploit together our novel framework (at different stages of maturity) considering part-scale heat transfer analyses of AM-PBF processes as target application. In particular, we join our efforts to push forward efficient and reliable reduced modelling strategies, in order to make thermal AM-PBF models more accessible and affordable to experimentalists and engineers.

Many authors have proposed temporal and spatial model reductions in thermal AM-PBF part-scale models [123]. The resulting models strike good balance between computational efficiency and accuracy. In general, they are not suitable for high-fidelity applications (e.g. as input for microstructural analyses), but can be used to globally optimise the process (e.g. tune parameters or scan path) in reasonable computational times. They can also be combined with HPC tools to provide answers in much shorter times. In spite of this, these strategies have been subject to insufficient numerical assessment, sensitivity analysis and contrast with physical experiments; especially, at printing sizes that approach the limits of current machines. According to this, in our first work with MCAM-MU, we address the most widely used temporal reduction strategy in AM, namely, merging or lumping scan passes or layers [128]:

> **Objective 5 (O5)**
> *Experimentally-supported* numerical assessment of (scan pass or layer) lumping methods in *industrial-scale* thermal FE models of metal AM by powder-bed fusion.

In **O5**, we intend to focus in accuracy, computational cost and utility to help engineers selecting the best lumping strategy for their simulation needs. In our second collaboration with MCAM-MU, we turn our attention to the fact that AM-PBF processes are very localised, e.g. high heat fluxes concentrate in the last printed layers. Hence, the thermal history predicted by a full model of the printing system, i.e. including the loose unfused powder and the building platform, does not significantly vary in front of *faster* spatially-reduced models that only consider relevant subregions of the system, e.g. only the printed object. However, spatial model reductions by, e.g. exclusion of the loose powder-bed from the computational domain, have been restricted so far to *naive* approaches that neglect the thermal inertia at the excluded region, such as prescribing a constant heat loss through the powder-bed. To address this issue, we aim to leverage a state-of-the-art technique in modelling of casting solidification, the so-called virtual mould approach [61, 94, 141], to propose a new rationale to localise thermal AM-PBF models:

> **Objective 6 (O6)**
> Formulate and *experimentally* validate a new *physics-based* thermal contact model, accounting for thermal inertia and suitable for localising AM-PBF models.

The last objective is transversal to **O1-6** and cornerstone to provide short-term solutions for AM-PBF:

**Objective 7 (O7)**
Highly-scalable implementation of the novel framework in high-end large-scale FE codes, exploiting state-of-the-art parallel adaptive tree-based mesh engines and scalable iterative linear solvers.

To meet **O7** we implement the numerical framework of **O1-4** in FEMPAR (Finite Elements Multiphysics PARallel code), an open source scientific code under GPU/GPL license. The FEMPAR project started in 2011 and is lead by Santiago Badia, Alberto F. Martín and Javier Príncipe. FEMPAR is a parallel hybrid OpenMP/MPI object-oriented framework for the massively parallel finite element simulation of multiphysics problems governed by PDEs (e.g. solid mechanics, fluid mechanics or electromagnetism) [19]. The code has been proven to have excellent weak scalability properties up to half million cores and two million MPI tasks on one of the largest supercomputers in Europe (JUQUEEN) [18] and efficiently runs on supercomputers worldwide (CURIE; HERMIT; HPC-FF; MARENOSTRUM; MINOTAURO; HELIOS). Therefore, it provides the ideal computer environment to implement and test the code developments of this thesis.

## 1.3 Document structure

In the first chapter of this thesis, we introduce and motivate the object of our research and describe the goals of this Thesis. Chapters 2-7 contain the main contributions of this study. Each one of the chapters has a one-to-one correspondence to the list of publications in the next section. The chapters are self-contained, preserve the structure of the associated paper and can be read independently. However, we have tried to keep notation as homogeneous as possible.

Chapter 2 is devoted to formally derive and prove the correctness of the algorithms and data structures in a parallel, distributed-memory, generic finite element framework that supports $h$-adaptivity on computational domains represented as forest-of-trees. We motivate the work and review the state-of-the-art in Section 2.1. In Section 2.2, we introduce tree-based AMR with SFCs. In Section 2.3, we present the outer mesh layer in our two-layered mesh approach. In Section 2.4, we discuss the construction of conforming FE spaces in parallel and linear system assembly. In Section 2.5, we assess the parallel strong scalability and performance of the proposed algorithms and data structures as implemented in FEMPAR for the Poisson and Maxwell PDEs. Finally, we draw conclusions in Section 2.6.

In Chapter 3, we describe a novel adaptive unfitted finite element scheme that combines the aggregated finite element method with parallel adaptive mesh refinement. After the introduction in Section 3.1, we present, in Section 3.2, a possible way to construct conforming AgFE spaces on top of non-conforming (adaptive) meshes, including its distributed-memory extension. We consider the Poisson equation as model problem in Section 3.3 and prove well-posedness of the associated agFE method in Section 3.4.

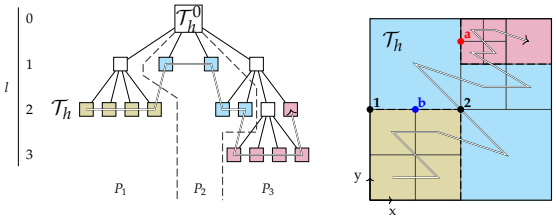
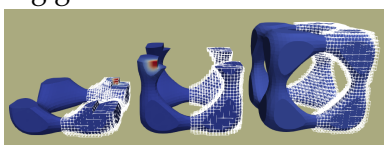
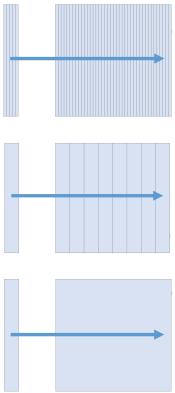
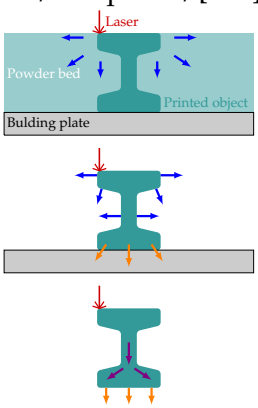
<table>
<tr><td>Motivation<br>in metal AM</td><td>1. Speed up control of process-to-performance link<br>2. Faster and cheaper product design and certification</td></tr>
<tr><td>Main goal</td><td>A novel highly-scalable numerical framework<br>tailored to PDEs posed on evolving geometries</td></tr>
<tr><td>achieved through</td><td>• O1/Chapter 2/[16] Mathematically-sound<br>generic FE framework on tree-based meshes<br>  <br>• h-adaptive aggregated FE methods for immersed<br>boundaries O2/Chapter 3/[15] interfaces O3/Chapter 4/[142]<br>  <br>• O4/Chapter 5/[143] Parallel FE modelling<br>of PDEs in growing geometries<br>  </td></tr>
<tr><td>Code development</td><td>O7 Highly-scalable implementation<br>in the large-scale FE code <code>FEMPAR</code></td></tr>
<tr><td>Target application</td><td>Part-scale heat transfer analysis of AM-PBF processes, focus in<br>numerical and experimental study of model reduction strategies<br>Scan pass or layer lumping O5/Chapter 6/[143]<br>  <br>Physics-based localisation O6/Chapter 7/[144]<br>  </td></tr>
</table>

Table 1.1: Overview of thesis motivation and objectives.

In the numerical tests of Section 3.5, we consider the model problem on several complex geometries and *hp*-FEM standard benchmarks. We draw the main conclusions of our chapter in Section 3.6.

Chapter 4 is devoted to design a fully robust and highly-scalable, *h*-adaptive aggregated unfitted finite element method for large-scale interface elliptic problems. After motivation and literature review in Section 4.1, we introduce the embedded interface geometry setup and define AgFE spaces for embedded *n*-interfaces in Section 4.2. Afterwards, we restrict ourselves to the approximation of single interface linear elasticity problems, see Section 4.3. In the numerical tests of Section 4.4, we consider both the linear elasticity and Poisson equations as model problems on several complex geometries and several *hp*-FEM standard benchmarks. Finally, we report the main conclusions and contributions of the chapter in Section 4.5.

In Chapter 5, we introduce an innovative parallel, fully-distributed finite element framework for growing geometries and its application to metal additive manufacturing. After introducing the topic in Section 5.1, we move on to briefly overview hierarchical AMR with octree meshes in Section 5.2. Then, we consider geometry growth modelling in Section 5.3. In Section 5.4 we consider an application to the heat transfer analysis of metal AM processes by powder-bed fusion. We describe next the main software abstractions supporting our implementation in Section 5.5 and the numerical study is carried out in Section 5.6. We draw the main conclusions of this work in Section 5.7.

Chapter 6 deals with model reduction strategies in AM by powder-bed fusion by scan-pass lumping. After motivation and literature review in Section 6.1, the formulation of the heat transfer problem is detailed in Section 6.2. The FE *activation* technique used to simulate the metal deposition is explained in Section 6.3. Section 6.4 describes the experimental setting at the MCAM. The calibration of the numerical model and the evaluation of the different model reduction approaches is addressed in Section 6.5. Finally, Section 6.6 presents the conclusions of this work.

Chapter 7 is devoted to model reduction by physics-based localisation in AM by powder-bed fusion. After the introduction in Section 7.1, we describe the heat transfer formulation for metal AM processes in Section 7.2. We introduce next the new *physics-based* rationale for domain reduction in Section 7.3. We address a simple verification example in Section 7.4.1 and experimental validation against physical experiments in Section 7.4.2. We draw main conclusions of this Chapter in Section 7.5.

To conclude this document, we lay out global conclusions and summarise the main contributions of this Thesis in Chapter 8. Furthermore, we list open lines of research to pursue, based on the developments in this thesis.

## 1.4 Research publications

The developments of this thesis have led to several publications, some of them already available in international peer-reviewed journals. Each publication directly corresponds to a different chapter in the body of the thesis:

**Chapter 2**

[16] S. BADIA, A. F. MARTÍN, EN AND F. VERDUGO, *A generic finite element framework on parallel tree-based adaptive meshes*, SIAM Journal on Scientific Computing, in press.

**Chapter 3**

[15] S. BADIA, A. F. MARTÍN, EN AND F. VERDUGO, *The aggregated unfitted finite element method on parallel tree-based adaptive meshes*, Submitted.

**Chapter 4**

[142] EN AND S. BADIA, *Robust and scalable h-adaptive aggregated unfitted finite elements for interface elliptic problems*, Submitted.

**Chapter 5**

[143] EN, S. BADIA, A. F. MARTÍN AND M. CHIUMENTI, *A scalable parallel finite element framework for growing geometries. Application to metal additive manufacturing*, International Journal for Numerical Methods in Engineering, 119 (2019), pp. 1098-1125.

**Chapter 6**

[55] M. CHIUMENTI, EN, E. SALSI, M. CERVERA, S. BADIA, J. MOYA, Z. CHEN, C. LEE AND C. DAVIES, *Numerical modelling and experimental validation in Selective Laser Melting*, Additive Manufacturing, 18 Supplement C (2017), pp. 171-185.

**Chapter 7**

[144] EN, M. CHIUMENTI, M. CERVERA, E. SALSI, G. PISCOPO, S. BADIA, A. F. MARTÍN, Z. CHEN, C. LEE AND C. DAVIES, *Numerical modelling of heat transfer and experimental validation in Powder-Bed Fusion with the Virtual Domain Approximation*, Finite Elements in Analysis and Design, 168 (2020), p. 103343.

## 1.5 Conference talks

In addition, the author has presented the contents of this thesis, *as presenting speaker*, in the following international conferences.

2020 EN AND S. BADIA, *An h-adaptive unfitted finite element method for interface elliptic boundary value problems*, I Monash WS on Numerical Differential Equations and Applications. Melbourne, Australia.

2019 EN, S. BADIA, M. CHIUMENTI, A. F. MARTÍN AND F. VERDUGO, *FEMPAR-AM: A parallel FE framework for the simulation of powder-bed metal additive manufacturing processes*, II International Conference on Simulation for Additive Manufacturing. Pavia, Italy.

2019 EN, S. BADIA, M. CHIUMENTI, A. F. MARTÍN AND F. VERDUGO, *FEMPAR-AM: Leveraging unfitted finite elements, hierarchical octree meshes and balancing domain decomposition by constraints for digital design and certification in 3D printing with metals*, IX International Congress on Industrial and Applied Mathematics. Valencia, Spain.

2018 EN, S. BADIA, M. CHIUMENTI, A. F. MARTÍN AND F. VERDUGO, *FEMPAR-AM: A parallel Finite-Element Framework for the simulation of Metal Additive Manufacturing*, Additive Manufacturing Benchmarks 2018. Washington, USA.

2017 EN, S. BADIA, M. CHIUMENTI, A. F. MARTÍN AND F. VERDUGO, *Parallel finite-element analysis of heat transfer in AM processes by metal deposition*, I International Conference on Simulation for Additive Manufacturing. Munich, Germany.

2017 EN, S. BADIA, M. CHIUMENTI AND A. F. MARTÍN, *A parallel finite-element framework for the heat transfer analysis of metal additive manufacturing*, XIV International Conference on Computational Plasticity. Barcelona, Spain.

## 1.6 Research stays

During the course of the doctoral studies, the author carried out a six-month research stay at Monash University, under the supervision of Prof. Santiago Badia. The work done during the stay lead to Chapters 3 and 4.

# Chapter 2

# A generic finite element framework on parallel tree-based adaptive meshes

*The contents of this chapter correspond to the research publication*

[16] S. BADIA, A. F. MARTÍN, EN AND F. VERDUGO, *A generic finite element framework on parallel tree-based adaptive meshes*, SIAM Journal on Scientific Computing, in press.

In this chapter we formally derive and prove the correctness of the algorithms and data structures in a parallel, distributed-memory, *generic* finite element framework that supports *h*-adaptivity on computational domains represented as forest-of-trees. The framework is grounded on a rich representation of the adaptive mesh suitable for generic finite elements that is built on top of a low-level, light-weight forest-of-trees data structure handled by a specialised, highly parallel adaptive meshing engine, for which we have identified the requirements it must fulfil to be coupled into our framework. Atop this two-layered mesh representation, we build the rest of data structures required for the numerical integration and assembly of the discrete system of linear equations. We consider algorithms that are suitable for both subassembled and fully-assembled distributed data layouts of linear system matrices. The proposed framework has been implemented within the `FEMPAR` scientific software library, using `p4est` as a practical forest-of-octrees demonstrator. A strong scaling study of this implementation when applied to Poisson and Maxwell problems reveals remarkable scalability up to 32.2K CPU cores and 482.2M degrees of freedom. Besides, a comparative performance study of `FEMPAR` and the state-of-the-art `deal.II` finite element software shows at least comparative performance, and at most factor 2-3 improvements in the *h*-adaptive approximation of a Poisson problem with first- and second-order Lagrangian finite elements, respectively.

## 2.1 Introduction

Research over the past few years has led to a new generation of petascale-capable algorithms and software for fast AMR using adaptive tree-based meshes endowed with SFCs [93]. SFCs are exploited for efficient data storage and traversal of the adaptive mesh, fast computation of hierarchy and neighbourship relations among the mesh cells, and scalable partitioning and dynamic load balancing. For hexahedral meshes, the state-of-the-art in tree-based AMR with SFCs is available at the `p4est` software [43, 99]. It provides parallel forest-of-octrees (i.e. a data structure which results from patching together multiple adaptive octrees) grounded on the standard $1{:}2^d$ isotropic refinement rule and the Morton SFC index [43]. We refer to [179] (and references therein) for works on single-octree handling, and to [43] for their extension to a forest-of-octrees. However, tree-based AMR with SFCs can be generalised to other cell topologies as well. The authors of [41] present an approach for simplicial adaptive meshes grounded on Bey's red-refinement rule, and the so-called tetrahedral Morton (TM) SFC index [41]. Holke's dissertation [93] goes even further by providing abstract, extensible forest-of-trees manipulation algorithms, i.e. cell-topology- and dimension-independent. By means of implementing a set of low-level local-to-cell operations, these algorithms may be extended to work with general polytopes (e.g. lines, simplices, bricks, prisms, pyramids, etc.). These ideas are being implemented in the on-going, open source software effort `t8code` [93].

The exploitation of single-octree or, more generally, forest-of-octrees in parallel adaptive FE solvers (see, e.g. [188]) has been shown to be cornerstone in several large-scale application problems (see, e.g. [147, 162]). Forest-of-trees meshes provide multi-resolution capability by local adaptation. Indeed, this capability makes them perfectly suitable for the efficient approximation of multi-scale PDEs. They are also highly appealing for PDEs posed on complex geometries in combination with unfitted (a.k.a. embedded boundary) FE methods [20, 24]. However, multi-resolution comes at a price. Forest-of-trees meshes are, in the most general case, *non-conforming*, i.e. they contain the so-called *hanging Vertices, Edges, and Faces (VEFs)*. These occur at the interface of neighbouring cells with different refinement levels. Mesh non-conformity introduces additional implementation complexity specially in the case of conforming FE formulations. DOFs sitting on *hanging* VEFs cannot have an arbitrary value, as this would result in violating the trace continuity requirements for conformity across interfaces shared by a coarse cell and its finer cell neighbours.

The set up (during FE space construction) and application (during FE assembly) of hanging DOFs constraints is well-established knowledge; see e.g. [157] and [170], resp. In a parallel distributed-memory environment, these steps are much more involved. In order to scale FE simulations to large core counts, the adaptive mesh must be partitioned among the parallel tasks such that each of these only holds a fraction of the global mesh cells. In particular, an overlapping partition of the mesh cells is

required where the processor-local portion is extended with a set of geometrically adjacent (neighbouring) off-processor cells, i.e. the so-called *ghost* cells. The standard practice is to constrain the data structures to keep a single layer of ghost cells, thus minimizing the impact of these extra cells on memory scalability. Under this constraint, if a set of suitable conditions are not satisfied, then the hanging DOF constraint dependencies may expand beyond the single layer of ghost cells, thus leading to an incorrect parallel FE solver. To make things worse, *the current literature clearly fails to explain when and why the parallel algorithms and data structures required to support generic conforming FE discretisations atop tree-based adaptive meshes are correct; see, e.g. [27, Section 3.3.4].*

The main contribution of the chapter is to address *in a rigorous way* the development of algorithms and data structures for parallel adaptive FE analysis on tree-based meshes endowed with SFCs. These are part of a generic FE simulation framework *that is designed to be general enough to support a range of cell topologies, recursive refinement rules for the generation of adaptive trees endowed with SFCs, and various types of conforming FEs.* Starting with a set of (reasonable) assumptions on the basic building blocks that drive the generation of the forest-of-trees, namely, a recursive refinement rule and a SFC index [93], we infer results based on mathematical propositions and proofs, yielding the (correctness of the) parallel algorithms in our framework.

The framework follows a *two-layered meshing approach*, namely an inner, light-weight layer encoding the forest-of-trees, handled by an external specialised meshing engine, and an outer representation of the adaptive mesh suitable for the implementation of generic adaptive FE spaces. The approach of reconstructing a rich mesh data structure to support generic FEs from a specialised forest-of-trees meshing engine is not new in itself. The excellent work in [27] discusses the particular approach followed by the state-of-the-art `deal.II` FE software library [28] in order to support generic adaptive FE spaces atop parallel forest-of-octrees meshes. However, *both the outer layer mesh representation itself, and the approach followed in this chapter to reconstruct the outer layer from the inner one are new*. In particular, [27] follows the so-called match-tree-recursive approach, while the one here is based on local neighbourhood information across the cell boundaries in the adapted mesh.

Atop this two-layered mesh representation, we design the rest of algorithms and data structures in the framework towards the final assembly of the discrete system of linear equations. In order to be able to leverage non-overlapping domain decomposition solvers, preconditioner [71], the algorithms in the framework are designed assuming that one deals with a subassembled data layout for the linear algebra data structures on the interface among subdomains.[1] In any case, by means of an extra final step, which is also discussed in the chapter, one can construct a global DOF numbering across the whole domain, and thus support fully-assembled global matrices and preconditioners, such as parallel algebraic multigrid (AMG), in the same framework. We

[1] We refer the reader to [17, 18] for the expected performance and scalability of the domain decomposition solver within `FEMPAR`.

can in particular leverage parallel distributed-memory linear algebra software packages, such as, e.g. `PETSc` [26], or `TRILINOS` [91].

An implementation of the framework is available at `FEMPAR` [19], an open source object-oriented (OO) Fortran200X scientific software package for the HPC simulation of complex multiphysics problems governed by PDEs at large scales. `FEMPAR` uses `p4est` as its specialised forest-of-octrees meshing engine, although any other engine able to fulfil a set of requirements described in the chapter could be plugged in the algorithms of the framework as well. A strong scaling study of this implementation when applied to Poisson and Maxwell problems reveals remarkable scalability up to 32.2K CPU cores and 482.2M degrees of freedom. Besides, a comparison of `FEMPAR` performance with that of `deal.II`, reveals at least competitive performance, and at most factor 2-3 improvements on a massively parallel supercomputer.

This chapter is structured as follows. In Section 2.2, we overview tree-based AMR with SFCs. In Section 2.3, we present the outer mesh layer in our two-layered mesh approach. In Section 2.4, we discuss the construction of conforming FE spaces *in parallel*, and that of sub- and fully-assembled linear systems. The main contributions of the chapter are concentrated in these two latter sections. When appropriate, differences among our approach and the one in [27] are highlighted for context. In Section 2.5, we assess the parallel strong scalability and performance of the proposed algorithms and data structures as implemented in `FEMPAR` for the Poisson and Maxwell PDEs on the MN-IV petascale supercomputer, and compare them with their counterparts in `deal.II` for the former problem. Finally, we draw conclusions in Section 2.6.

## 2.2 Overview of tree-based AMR endowed with SFCs

We describe the so-called tree-based AMR with SFCs approach for scalable mesh generation and partitioning. The idea was originally proposed in [43] for the particular case of octrees, and extended to general trees in [93]. Let $\Omega \subset \mathbb{R}^d$ be an open bounded domain in which our PDE problem is posed. AMR with SFCs can be seen as a two-level decomposition of $\Omega$, referred to as macro and micro level, resp.[2] In the macro level, we assume that there is a partition $\mathcal{C}$ of $\Omega$ into cells $T \in \mathcal{C}$ such that each of these cells can be expressed as a differentiable homeomorphism $\Phi_T$ over a set of admissible reference polytopes [19]. The mesh $\mathcal{C}$ is referred to as the coarse mesh and it is assumed to be a *conforming mesh* (see Section 2.2.1). In the micro level, each of the cells of $\mathcal{C}$ becomes the root of an adaptive tree. This two-tier adaptive structure is referred to as *forest-of-trees*. It represents a locally refined mesh $\mathcal{T}$. The process that generates it is essentially characterised by defining two complementary ingredients, namely a (recursive) refinement rule (see Section 2.2.2) and an SFC index (see Section 2.2.3). We note that the ideas presented in this section are not new. However, we cover them to an extent such that we can precisely state a set of assumptions on the aforementioned two ingredients, and

[2] We note that this two-level construction is for geometrical purposes only.

define the notation that we require to give mathematical rigour to the derivation of the algorithms in our framework.

### 2.2.1 Polytopes and coarse mesh

Reference polytopes for FE analysis are usually cubes or tetrahedra (and their extensions to arbitrary dimensions), but prisms and pyramids can also be used. A reference polytope spans an open domain. We consider a partition of its boundary into *disjoint* polytopes of lower dimension (i.e. vertices, edges without endpoints, etc.), denoted as $n$-faces, with $0 \leq n < d$ being their topological dimension; the $d$-face is the polytope itself. Furthermore, the $n$-faces on the boundary of an $m$-face, $n < m$, of $\hat{P}$ are also $n$-faces of $\hat{P}$. For example, with $d = 3$, the boundary is composed of VEFs, representing the 0, 1, and 2-faces of the cell, resp. Hereafter, we may abuse notation by using the acronym VEF for any $n$-face with $n \in [0, d)$. The set of VEFs of $\hat{P}$ is represented with $\mathcal{F}_{\hat{P}}$.

Cells in the physical domain $T \in \mathcal{T}$ are determined by $\Phi_T$ and the reference polytope $\hat{P}$. The map $\Phi_T$ applied to every $t \in \mathcal{F}_{\hat{P}}$ generates the set of *local* VEFs of $T$, denoted as $\mathcal{F}_T$. A local VEF can be understood as a tuple of the cell $T$ it belongs to and the geometrical domain it represents, which we denote as $[f]$. Global VEFs only represent the geometrical domain. Thus, given a local VEF $f$, its corresponding global VEF is $[f]$. Formally, local VEFs are glued together into global VEFs as dictated by an equivalence relation $\sim$ such that $f \sim g$ iff $[f] = [g]$. We represent with $\mathcal{F}$ the set of *global* VEFs of $\mathcal{T}$, i.e. the quotient space of all equivalence classes (global VEFs) resulting from the application of $\sim$ to the elements of $\bigcup_{T \in \mathcal{T}} \mathcal{F}_T$. $[\cdot]$ is the so-called local-to-global VEF map.

As mentioned at the beginning of Section 2.2, in AMR with SFCs, the coarse mesh $\mathcal{C}$ is assumed to be conforming. A formal definition of this property is as follows.

**Definition 2.2.1** (Conforming mesh)**.** *A mesh $\mathcal{T}$ is conforming if, for any two cells $T, T' \in \mathcal{T}$ with $\overline{T} \cap \overline{T'} \neq \emptyset$, there exist $f \in \mathcal{F}_T$, $f' \in \mathcal{F}_{T'}$ such that $[\bar{f}] = [\overline{f'}] = \overline{T} \cap \overline{T'}$.*

### 2.2.2 Refinement rule

Let us consider a cell $T \in \mathcal{C}$ in the coarse mesh of $\Omega$. A *refinement rule* prescribes how one may subdivide the coarse cell $T$ into finer cells that cover the same region as $T$. $T$ is referred to as the *parent* cell, and the latter ones, (its) *children* cells. The refinement rule is usually defined at the polytope and then mapped to the physical space using $\Phi_T$. The reverse application of the refinement rule, i.e. the replacement of the children cells by its parent, is referred to as coarsening. Mesh generation in this context is a hierarchical process based on the recursive application of refinement and coarsening. At each level in the hierarchy, some cells are marked for refinement, and some other for coarsening. A cell marked for refinement is replaced by its children cells following the refinement rule. On the other hand, if all children cells of a given parent are marked

for coarsening, they are collapsed into the parent cell. As a result of this process for all cells in $\mathcal{C}$, we obtain a *forest* of tree-like refinement structures, with a tree rooted at every $T \in \mathcal{C}$. We make the following assumption on the refinement rule.

**Assumption 2.2.2** (Admissible refinement rule)**.** *The cell refinement rule over a cell $T$ provides a conforming mesh $\mathcal{R}_T$ of $T$ (into children cells) and an orientation to the n-faces of $\mathcal{R}_T$. The restriction of $\mathcal{R}_T$ to any n-face $f \in \mathcal{F}_T$ with $n > 0$ is a non-trivial partition of $f$, i.e. these n-faces are subdivided. Besides, when recursively applying the refinement rule to a cell, the resulting mesh is non-degenerate: there is a constant $\rho > 0$ independent of the number of levels of refinement such that, for any cell $T$ in the resulting mesh, there is a ball of diameter $\rho\,\mathrm{diam}(T)$ contained in $T$, where $\mathrm{diam}(T)$ denotes the diameter of $T$.*

We need Assumption 2.2.2 in order to be able to mathematically prove the correctness of the algorithms in our framework. In practice, this assumption is not restrictive, since it holds for those refinement rules and SFCs which are at the heart of the state-of-the-art software in tree-based adaptive meshes endowed with SFCs [43, 93], i.e. uniform refinement rules of hexahedra, tetrahedra, prisms, and pyramids (and 2D counterparts).

We define the following relation between the VEFs of $\mathcal{R}_T$, represented with $\mathcal{F}_{\mathcal{R}_T} \doteq \bigcup_{T_c \in \mathcal{R}_T} \mathcal{F}_{T_c}$, and the $n$-faces in $\mathcal{F}_T \cup \{T\}$.

**Definition 2.2.3** (Owner of refined VEF)**.** *For every VEF $f \in \mathcal{F}_{\mathcal{R}_T}$, we define its owner n-face $O_f$ as the unique n-face in $\mathcal{F}_T \cup \{T\}$ that contains it, i.e. $f \subset O_f$.*

By Definition 2.2.3, $O_f$ has topological dimension greater than or equal to the one of $f$.

A particular adaptive mesh $\mathcal{T}$ of $\Omega$ to be used for FE discretisation is defined as the union of all leaf cells (i.e. cells with no children) in the forest. We denote by $|\mathcal{T}|$ the number of cells in $\mathcal{T}$. For every cell $T \in \mathcal{T}$, we can define $\ell(T)$ as the *level of refinement* of $T$ in the forest, with $\ell(T) = 0$ if $T \in \mathcal{C}$ and $\ell(T) > 0$ otherwise. If, for any two cells $T, T' \in \mathcal{T}$, we have that $\ell(T) > \ell(T')$, then $T$ is *finer* than $T'$, and $T'$ is *coarser* than $T$.

### 2.2.3 The SFC index

Let us assume that we bound the maximum level of refinement for any forest-of-trees that can be built by means of the hierarchical process described in Section 2.2.2; this is the case in practice since available memory is limited. Let us denote with $n > 0$ such maximum level of refinement and with $\mathcal{S}_n$ the refinement tree that results from $n$ recursive applications of the refinement rule to all leaves. $\mathcal{S}_n$ includes the set of all cells with a maximum refinement level of $n$ that can be potentially constructed from $\mathcal{C}$ by means of AMR. With this notation, we can readily introduce the concept of *SFC index*.[3] The maximum level subscript is omitted in the definition.

[3]The definition in the sequel has been adapted from [93], where a novel approach to the theory of discrete SFCs suitable for tree-based AMR is presented.

**Definition 2.2.4.** *An SFC index $\mathcal{I}$ on $\mathcal{S}$ is formally defined as a map $\mathcal{I} : \mathcal{S} \to \mathbb{N}_0$ that fulfills the following three properties. First, the map $\mathcal{I} \times \ell : \mathcal{S} \to \mathbb{N}_0 \times \mathbb{N}_0$ is injective, and thus the pair composed by the SFC index and the refinement level of a cell uniquely identifies it among all cells in $\mathcal{S}$. Second, for any $T, T' \in \mathcal{S}$, where $T'$ is a descendant of $T$, then $\mathcal{I}(T) \leq \mathcal{I}(T')$. Additionally, given a $T''$ that is not a descendant of $T$ and $\mathcal{I}(T) < \mathcal{I}(T'')$, it holds that $\mathcal{I}(T') < \mathcal{I}(T'')$.*

A key requirement from the computational viewpoint is to choose the $\mathcal{I}$ mapping such that functions that locally operate on a cell (or small set of cells) take *constant time*, independently of the level of refinement of the cell. Local cell functions include computing the SFC index of a cell, determining the SFC index of its parent, children, and neighbours with the same refinement level, or computing vertex coordinates. Besides, the exploitation of the SFC index for implementing such operations has to result in significant memory savings with respect to unstructured meshing, in which a list of neighbours has to be stored, as well as the coordinates of all vertices in the mesh, for each cell. An SFC index that fulfils Definition 2.2.4, and the aforementioned requirement, is highly suited for efficient data storage and traversal of the adaptive mesh, fast computation of hierarchy and neighbourship relations among the mesh cells, and scalable partitioning and dynamic load balancing [93].[4]

### 2.2.4 Non-conformity and cell neighbours

Tree-based meshes provide multi-resolution capability by local adaptation, i.e. cells in $T \in \mathcal{T}$ might have different value for $\ell(T)$. However, these meshes are (potentially) *non-conforming*, i.e. they contain the so-called *hanging VEFs*. These occur at the interface of neighbouring cells with different refinement levels.

In order to provide support to FE applications, tree-based AMR with SFCs engines also require to keep track of neighbouring relationships between cells in $\mathcal{T}$ (apart from hierarchical relationships). One possible way of describing these is by means of the notion of *cell neighbours across local n-faces*, which we define in the sequel.

**Definition 2.2.5** (Neighbours of a cell across local $n$-faces)**.** *The neighbours of $T \in \mathcal{T}$ across its n-face $f \in \mathcal{F}_T$ are classified into the sets $\mathcal{T}_{T,f}$, $\mathcal{T}^+_{T,f}$, and $\mathcal{T}^-_{T,f}$, referred to as conformal, higher-level, and lower-level neighbours, resp. They are composed of those cells $T' \in \mathcal{T} \setminus T$ that contain $f' \in \mathcal{F}_{T'}$ such that: (a) $[f] \odot [f']$, with $\odot$ being $=$, $\supseteq$, and $\subsetneq$ for $\mathcal{T}_{T,f}$, $\mathcal{T}^+_{T,f}$, and $\mathcal{T}^-_{T,f}$, resp.; (b) $[\overline{f}] \cap [\overline{f'}] = \overline{T} \cap \overline{T'}$.*

The connectivity information underlying Definition 2.2.5 is required by our framework in order to (re)build a mesh data structure suitable for the construction of generic conforming FEs; see Section 2.3. We expect any forest-of-trees mesh engine to be able

[4] Dynamic load balancing is the ability of an adaptive mesh to be re-distributed in the presence of an unacceptable amount of load imbalance, e.g. the one generated by means of AMR in a highly localised region of $\Omega$.

to provide this sort of relationships, as these are indeed internally determined as a requirement of some of the algorithms provided by the engine; see, e.g. `Balance` or `Ghost` in Section 2.2.6.

With the cell neighbours across $n$-faces, one can compute the local-to-global VEF map $[\cdot]$; see Algorithm 1 for more details. Given $f \in \mathcal{F}_T$ and $F = [f]$, we define $\mathcal{T}_{T,F} \doteq \mathcal{T}_{T,f}$ (analogously for $\mathcal{T}^+_{T,F}$ and $\mathcal{T}^-_{T,F}$). We denote by $[\mathcal{F}_T] \doteq \{[f] : f \in \mathcal{F}_T\}$ the set of global VEFs of $T$. We next introduce a proposition on $\mathcal{T}_{T,F}$, $\mathcal{T}^+_{T,F}$, and $\mathcal{T}^-_{T,F}$. In order to prove this proposition (and some of those in Section 2.3), we need the following assumption on the recursive refinement procedure.

**Assumption 2.2.6** (Uniform refinement conformity)**.** *The mesh composed of all leaves of the refinement tree $\mathcal{S}_\ell$ obtained after an arbitrary level $\ell$ of uniform refinements of $\mathcal{C}$ is also conforming.*

The uniform refinement conformity in Assumption 2.2.6 requires a refinement rule consistent among cells in the coarse mesh. We note that this assumption is already fulfilled by the standard uniform refinement rules mentioned after Assumption 2.2.2.

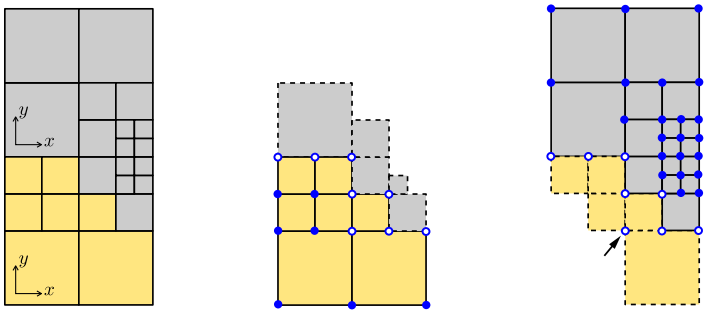
**Proposition 2.2.7.** *Given a cell $T \in \mathcal{T}$, and an $n$-face $F \in [\mathcal{F}_T]$ such that $n > 0$, then $\mathcal{T}_{T,F}$ can only be composed of neighbour cells at the same refinement level as $T$. On the other hand, $\mathcal{T}^-_{T,F}$ (resp., $\mathcal{T}^+_{T,F}$) can only be composed of neighbour cells with lower (resp., higher) refinement level than that of $T$.*

*Proof.* By Definition 2.2.5, for any $T' \in \mathcal{T}_{T,F}$, it holds $F \in [\mathcal{F}_{T'}]$ and $\overline{F} \subset \overline{T} \cap \overline{T'}$. If $\ell(T) > \ell(T')$ were true, then there would be an ancestor $T''$ of $T$ at the same refinement level of $T'$. By Assumption 2.2.6, the mesh composed of all leaves of the refinement tree $\mathcal{S}_{\ell(T')}$ is a conforming mesh. Thus, by Definition 2.2.1, there exists an $n$-face $G \in [\mathcal{F}_{T'}] \cap [\mathcal{F}_{T''}]$ such that $\overline{G} = \overline{T'} \cap \overline{T''}$. Since $\overline{F} \subset \overline{T} \cap \overline{T'}$, $\overline{G} = \overline{T'} \cap \overline{T''}$, $T \subset T''$, and the refinement rule subdivides all cell $n$-faces with $n > 0$ (see Assumption 2.2.2), then $F \subsetneq G$. As a result, $T'$ has two *different* VEFs $F, G \in [\mathcal{F}_{T'}]$ such that $F \cap G \neq \emptyset$. This is a contradiction, since the VEFs of a cell are disjoint by definition. We proceed analogously for $\ell(T) < \ell(T')$. It proves the result for $\mathcal{T}_{T,F}$. The result for $\mathcal{T}^-_{T,F}$ and $\mathcal{T}^+_{T,F}$ can be proved in a similar way. □

However, for $n = 0$, $\mathcal{T}_{T,F}$ might be composed of neighbours at the same, higher or lower-level of refinement than $T$. This can be readily observed, e.g. in Figure 2.1(a), for any corner that meets at the boundary of cells with different levels of refinement.

### 2.2.5 Balanced forest-of-trees

For FE applications, mesh non-conformity makes harder the construction of conforming FE spaces, and the subsequent steps in the simulation; see Section 2.4. However, common practice in order to significantly alleviate this extra complexity consists of




(a) $\mathcal{T}$. (b) $\mathcal{T}^1$. (c) $\mathcal{T}^2$.

Figure 2.1: 2:1 0-balanced forest-of-quadtrees mesh with two quadtrees (i.e. $|\mathcal{C}| = 2$) distributed (non-uniformly) among two processors (i.e. $P = 2$) with 0-ghost cells, 1:4 refinement and the Morton SFC index [43]. Cells in $\mathcal{T}_{\mathrm{L}}^p$ are depicted with continuous boundary lines, while those in the 0-ghost layer $\mathcal{T}_{\mathrm{G}}^p$ with dashed ones. All vertices in $\mathcal{F}_{\mathrm{L}}^p$ are depicted with circles. In particular, vertices in $\mathcal{F}_{\mathrm{I}}^p$ are depicted with blue circles, whereas vertices in $\mathcal{F}_{\Gamma}^p$ are depicted with unfilled blue circles. The mesh vertex in $\mathcal{T}^2$ pointed by the arrow is in $\mathcal{F}_{\mathrm{L}}^p$ as it fulfils Definition 2.3.11 (d).

enforcing the so-called 2:1 balance constraint (a.k.a. balance or mesh regularity condition). A general definition of the 2:1 balance condition, parameterized by an integer $0 \leq k < d$, is as follows.

**Definition 2.2.8.** *A forest-of-trees mesh $\mathcal{T}$ is 2:1 k-balanced if and only if, for any cell $T \in \mathcal{T}$, and n-face $F \in [\mathcal{F}_T]$, with $n \in [k, d)$, there is no neighbour $T'$ of $T$ across $F$ (see Definition 2.2.5) such that $|\ell(T') - \ell(T)| > 1$.*

In other words, geometrically neighbouring cells may differ at most by a single level of refinement, with the notion of cell neighbourhood depending on the value of $k$. Figure 2.1(a) illustrates a forest-of-quadtrees, with two quadtrees (i.e. $|\mathcal{C}| = 2$), which is 2:1 0-balanced, i.e. balance across corners and facets. We note that, as a consequence of Proposition 2.2.7, and Definition 2.2.8, the $\mathcal{T}_{T,f}^{+}$, and $\mathcal{T}_{T,f}^{-}$ sets for local $n$-faces $f$ with $n \geq \max(k, 1)$, are composed of cells $T'$ such that $\ell(T') = \ell(T) + 1$ and $\ell(T') = \ell(T) - 1$, resp. On the other hand, if $n = k = 0$, then $\mathcal{T}_{T,f}$ may be composed of neighbours at the same, one unit higher, or one unit lower level of refinement than $T$; see Figure 2.1(a).

In Section 2.4, it will become clear why this constraint extremely simplifies the construction of conforming FE spaces, especially in a distributed-memory context.

### 2.2.6 Forest-of-trees handler operations

In practice, parallel tree-based AMR using SFCs is a specialised feature that numerical applications typically outsource to an external adaptive meshing engine. In the interface among these, a set of core application-level operations have been identified as cornerstone in order to fully realize this functionality in numerical applications. These are briefly outlined in the sequel. (a) `New`: creates a new uniformly refined forest, up to a user-provided level, from a data structure describing the connectivity of mesh cells in

$\mathcal{C}$; (b) `Adapt`: refines and coarsens the current forest according to a user-provided criterion; (c) `Partition`: redistributes the forest leaves among processors for dynamic load balancing; (d) `Balance`: ensures the 2:1 $k$-balance condition among neighbouring cells by local refinement where required; (e) `Ghost`: creates a data structure that contains the so called $s$-ghost cell set, i.e. a layer of off-processor leaf cells that are neighbours of cells in the current processor (see Section 2.3); (f) `Iterate`: given a 2:1 $k$-balanced forest, provides a mechanism to iterate over its leaf cells, and *a subset of* the inter-cell interfaces (i.e. $n$-faces such that $n \geq k$), while letting applications get local neighbourhood information of each interface visited, including both conforming and non-conforming cell interfaces. We refer to [43, 93, 99] and references therein for a comprehensive presentation of the algorithms behind these operations, implementation details, and cost and communication volume analysis, among others.

## 2.3 A general FE-suitable distributed adaptive mesh representation

In this section we present a distributed adaptive mesh representation that supports generic FE spaces built atop. This data structure implements the outer mesh layer in our two-layered framework; see Section 2.1. In Section 2.3.1 and 2.3.2, we formally define the sort of data it handles (e.g. adjacencies among cells and global VEFs). The exposition is supplemented with a set of propositions that are required in order to prove the correctness of the algorithms presented in Section 2.3.3 and Section 2.4. The concepts underlying our adaptive mesh data structure are not tailored to a particular tree-based AMR with SFCs technology or cell topology.

### 2.3.1 Cells and global VEFs adjacencies

Our adaptive mesh representation considers three different kinds of adjacency relations among cells and global VEFs. First, given a cell $T \in \mathcal{T}$, $[\mathcal{F}_T]$ keeps track of the global VEFs (equivalence classes) corresponding to the local VEFs of $T$. The remaining two relations are neighbourship relations. Given a global VEF $F \in \mathcal{F}$, the set of cells around the VEF is defined as $\mathcal{T}_F \doteq \{T \in \mathcal{T} : F \in [\mathcal{F}_T]\}$. For our purposes, the definition of the *coarser cells around a global VEF* is essential.

**Definition 2.3.1** (Coarser cells around a global VEF)**.** *The set of coarser cells around a VEF $F \in \mathcal{F}$ is represented with $\tilde{\mathcal{T}}_F$ and is composed by the cells $T \in \mathcal{T}$ such that: (a) $F \notin [\mathcal{F}_T]$; (b) $F \subset \bar{T}$.*

The following proposition holds for the $\mathcal{T}_F$ and $\tilde{\mathcal{T}}_F$ sets.

**Proposition 2.3.2.** *For any forest-of-trees $\mathcal{T}$, $F \in \mathcal{F}$, $T \in \mathcal{T}_F$ and $T' \in \tilde{\mathcal{T}}_F$, it holds $\ell(T) > \ell(T')$.*

*Proof.* Cell levels must be different due to the conformity in Assumption 2.2.6. Let us assume that $\ell(T) < \ell(T')$. We can consider the recursive refinement of $T$ till reaching $\ell(T')$. In this process, by the definition of the refinement rule, any $n$-face of $T$ with $n > 0$ is being partitioned into a set of VEFs at level $\ell(T')$. By the conformity of the uniformly refined tree at $\ell(T')$, any VEF in $\mathcal{F}_{T'}$ in touch with $\overline{T}$ is a strict subset of a VEF in $\mathcal{F}_T$ or a vertex in $\mathcal{F}_T$. Thus, the situation in the statement is not possible and $\ell(T) > \ell(T')$ must hold. □

The $\mathcal{T}_F$ and $\tilde{\mathcal{T}}_F$ sets can be readily obtained from the neighbours of cells across local $n$-faces, as stated in the following two propositions. For conciseness, we define the sets

$$\mathcal{S}_{T,F} \doteq \bigcup_{G \in [\mathcal{F}_T] : F \subset \overline{G}} \mathcal{T}_{T,G}, \quad \mathcal{S}^-_{T,F} \doteq \bigcup_{G \in [\mathcal{F}_T] : F \subset \overline{G}} \mathcal{T}^-_{T,G},$$

$$\mathcal{S}^+_{T,F} \doteq \left\{ T' \in \left( \bigcup_{G \in [\mathcal{F}_T] : F \subset \overline{G}} \mathcal{T}^+_{T,G} \right) : F \in [\mathcal{F}_{T'}] \right\}.$$

**Proposition 2.3.3.** *Given a cell $T \in \mathcal{T}$ and $F \in [\mathcal{F}_T]$, then we have that:*

$$\mathcal{T}_F = \{T\} \cup \mathcal{S}_{T,F} \cup \mathcal{S}^+_{T,F} \cup \{T' \in \mathcal{S}^-_{T,F} : F \in [\mathcal{F}_{T'}]\}, \tag{2.1}$$

$$\tilde{\mathcal{T}}_F = \{T' \in \mathcal{S}^-_{T,F} : F \notin [\mathcal{F}_{T'}]\}. \tag{2.2}$$

*Proof.* Let us represent the right-hand side of (2.1) (resp., (2.2)) with $\mathcal{S}_F$ (resp., $\tilde{\mathcal{S}}_F$). We want to prove that $\mathcal{T}_F = \mathcal{S}_F$ and $\tilde{\mathcal{T}}_F = \tilde{\mathcal{S}}_F$. First, we note that $\mathcal{T}_F \cup \tilde{\mathcal{T}}_F$ is the set of all cells $T' \in \mathcal{T}$ such that $F \subset \overline{T'}$. On the other hand, $\mathcal{S}_F \cup \tilde{\mathcal{S}}_F = \{T\} \cup \mathcal{S}_{T,F} \cup \mathcal{S}^+_{T,F} \cup \mathcal{S}^-_{T,F}$ spans the same set of cells, which can be proved using Definition 2.2.5. Thus, $\mathcal{T}_F \cup \tilde{\mathcal{T}}_F = \mathcal{S}_F \cup \tilde{\mathcal{S}}_F$. On the other hand, one can check that $\mathcal{S}_F \subset \mathcal{T}_F$, since: (a) $T \in \mathcal{T}_F$ by hypothesis, (b) $\mathcal{S}_{T,F} \subset \mathcal{T}_F$ by Definition 2.2.5 and (c) $\mathcal{S}^+_{T,F}$ and the fourth set in the definition of $\mathcal{S}_F$ are subsets of $\mathcal{T}_F$ by definition. One can also readily check that $\tilde{\mathcal{S}}_F \cap \mathcal{T}_F = \emptyset$ by its definition. Combining these results, we prove that $\mathcal{T}_F = \mathcal{S}_F$ and $\tilde{\mathcal{T}}_F = \tilde{\mathcal{S}}_F$. □

**Proposition 2.3.4.** *Given an $n$-face $F \in \mathcal{F}$, with $n > 0$, and a cell $T \in \mathcal{T}$ such that $F \in [\mathcal{F}_T]$, then we have that: (a) $\mathcal{T}_F = \mathcal{S}_{T,F} \cup \{T\}$; and (b) $\tilde{\mathcal{T}}_F = \mathcal{S}^-_{T,F}$.*

*Proof.* Let us prove that for $n$-faces $F$ with $n > 0$, it holds: (1) $\mathcal{S}^+_{T,F} = \emptyset$ and (2) $\{T' \in \mathcal{S}^-_{T,F} : F \in [\mathcal{F}_{T'}]\} = \emptyset$ and $\{T' \in \mathcal{S}^-_{T,F} : F \notin [\mathcal{F}_{T'}]\} = \mathcal{S}^-_{T,F}$. We prove (1) using the same argument as in Proposition 2.3.2; a cell $T' \in \mathcal{S}^+_{T,F}$ belongs to $\mathcal{T}^+_{T,G}$ for $G \in [\mathcal{F}_T]$. It holds $\ell(T') > \ell(T)$ by Proposition 2.2.7, since $G$ is an $n$-face with $n > 0$. On the other hand, $F \notin [\mathcal{F}_T] \cap [\mathcal{F}_{T'}]$ by using the fact that $n$-faces with $n > 0$ are partitioned with refinement. Thus, $\mathcal{S}^+_{T,F}$ is empty. The same arguments can be readily used to prove (2). Combining these results and Proposition 2.3.3, we end the proof. □

The set $\tilde{\mathcal{T}}_F$ can be used to classify $\mathcal{F}$ into *regular* and *hanging* VEFs.

**Definition 2.3.5** (Regular and hanging VEFs)**.** *A VEF is regular if $\tilde{\mathcal{T}}_F = \emptyset$ and hanging otherwise.*

We denote as $\mathcal{F}_{\mathrm{R}}$ and $\mathcal{F}_{\mathrm{H}}$ the set of regular and hanging VEFs, resp. Clearly, $\{\mathcal{F}_{\mathrm{R}}, \mathcal{F}_{\mathrm{H}}\}$ is a partition of $\mathcal{F}$.

**Proposition 2.3.6.** *Let us consider a 2:1 k-balanced mesh $\mathcal{T}$ and an n-face $F \in \mathcal{F}_{\mathrm{H}}$ with $n \geq k$. For any $T \in \mathcal{T}$ with a local VEF $f \in \mathcal{F}_T$ satisfying $[f] = F$, there exists $T' \in \mathcal{T}$ and $g \in \mathcal{F}_{T'}$ such that $[g] = [O_f]$. Thus, one can define the owner VEF of $F$ as $O_F \doteq [g]$ and $F \subsetneq O_F$. If $F$ is a vertex, it also holds for a 2:1 1-balanced mesh.*

*Proof.* Due to Proposition 2.2.7 and the 2:1 $k$-balance, since the VEF $F$ is hanging, it is in touch with a cell $T'$ at level $\ell(T) - 1$. By definition, $O_f \in \mathcal{F}_{T_p}$ where $T_p$ is the parent cell of $T$. By conformity of the uniformly refined tree at level $\ell(T) - 1$, there exists a $g \in \mathcal{F}_{T'}$ such that $[O_f] = [g]$. Thus, one can define $O_F$ as $[g] \in \mathcal{F}$. If $F \in \mathcal{F}_{\mathrm{H}}$ is a vertex, there is a $T' \in \mathcal{T}$ such that $F \subset \overline{T'}$ but $T \notin [\mathcal{F}_{T'}]$. For any $T \in \mathcal{T}$, $f \in \mathcal{F}_T$ such that $[f] = F$, we consider the ancestor $T_p$ of $T$ at level $\ell(T')$. By conformity of the uniformly refined tree, we can pick $g \in \mathcal{F}_{T_p}$ and $g' \in \mathcal{F}_{T'}$ such that $\overline{T_p} \cap \overline{T'} = \bar{g} = \overline{g'}$. Clearly, $f \subsetneq g'$ (otherwise $f \in \mathcal{F}_{T'}$) and thus, $g$ and $g'$ are $n$-faces with $n > 0$. By the 2:1 1-balance, $\ell(T') = \ell(T_p) = \ell(T) - 1$. As a result, $O_f \in \mathcal{F}_{T_p}$ (i.e. $T_p$ is the parent cell), $O_f$ is a VEF of $g$ (by the polytope definition), and there exists an $h \in \mathcal{F}_{T'}$ such that $[h] = [O_f]$. Therefore, $[O_f] \in \mathcal{F}$ and we can define $O_F \doteq [O_f]$. □

**Proposition 2.3.7.** *Given a 2:1 k-balanced mesh $\mathcal{T}$ and $F \in \mathcal{F}_{\mathrm{H}}$ such that its owner VEF $O_F$ is an n-face with $n \geq k$, then the sets $\tilde{\mathcal{T}}_F$ and $\mathcal{T}_{O_F}$ are identical (or, equivalently, $O_F \in \mathcal{F}$).*

*Proof.* It is a direct consequence of the construction of $O_F$ in Proposition 2.3.6. □

The following *key property* holds for hanging VEFs.

**Proposition 2.3.8.** *Given a 2:1 k-balanced forest-of-trees mesh $\mathcal{T}$ and $F \in \mathcal{F}_{\mathrm{H}}$ such that $O_F$ is an n-face with $n \geq k$, it holds that $O_F \in \mathcal{F}_{\mathrm{R}}$. Besides, any n-face $E$ of $\overline{O_F}$ with $n \geq k$ is also regular, i.e. $E \in \mathcal{F}_{\mathrm{R}}$.*

*Proof.* If $F \in \mathcal{F}_{\mathrm{H}}$, there exists a $T \in \mathcal{T}_F$ and a $T' \in \tilde{\mathcal{T}}_F$. If $O_F \in \mathcal{F}_{\mathrm{H}}$, it would be in touch with a cell in level $\ell(T') - 1$ due to Proposition 2.3.2. It contradicts the 2:1 $k$-balanced assumption. Thus, $O_F \in \mathcal{F}_{\mathrm{R}}$. Finally, an $n$-face $E$ of $O_F$ belongs to $\mathcal{F}_{T'}$; the $n$-faces of an $n$-face of $T$ are $n$-faces of $T$ by construction of a polytope. Thus, it is easy to check that $E$ is in touch with $T'$ and a sibling cell of $T$. As above, if $E$ would be hanging, it would violate the 2:1 $k$-balance. □

As a consequence of Proposition 2.3.8, one enjoys the following two cornerstone benefits: (a) *only* single-level (a.k.a. direct) hanging node constraints are required when constructing global conforming FE spaces; (b) in a distributed-memory context, all such constraints can be resolved locally with a single layer of ghost cells. This will be revisited in Section 2.4. A stronger result is proved below for 2:1 1-balanced forest-of-trees.

**Proposition 2.3.9.** *Given a 2:1 1-balanced forest-of-trees mesh $\mathcal{T}$ and $F \in \mathcal{F}_{\mathrm{H}}$, then $O_F \in \mathcal{F}_{\mathrm{R}}$. Besides, any n-face $E$ of $\overline{O_F}$ is also regular, i.e. $E \in \mathcal{F}_{\mathrm{R}}$.*

*Proof.* To prove the first result, we must show that $O_V \in \mathcal{F}_{\mathrm{R}}$ for any vertex $V \in \mathcal{F}_{\mathrm{H}}$. The result for $O_V$ being an n-face with $n > 0$ has been proved in Proposition 2.3.8. If $O_V$ would be a vertex, $V = O_V$ and thus $O_V \in \mathcal{F}_{\mathrm{H}}$. Thus, $V$ would belong to a cell $T$ and its parent cell in $\ell(T) - 1$ would be in touch with a cell $T'$ in $\ell(T) - 2$ or lower in $\tilde{\mathcal{T}}_V$. Thus, there would be an $n$-face $G \in [\mathcal{F}_{T'}]$ such that $V \subsetneq G$, thus $n > 0$. Thus, $G$ would violate the 2:1 1-balance property. It proves the first result and also shows that hanging vertices are always 2:1 balanced. Using this fact, we can prove the second result as in Proposition 2.3.8. □

Proposition 2.3.9 implies that, for those global conforming FE spaces for which vertices carry out DOFs (e.g. those constructed from Lagrangian FEs), 2:1 0-balanced trees are not required to have (a) and (b) above, only 2:1 1-balance is required. Provided this is acceptable for the problem at hand from a numerical point of view, 2:1 1-balance may lead to computational savings, as enforcing it requires less balance refinement than 0-balance in general. While the result itself is known [98], up to the authors' knowledge, it has not been stated formally nor mathematically proven.

### 2.3.2 Distributed-memory context

With the definitions associated to the global adaptive mesh $\mathcal{T}$ so far, we can readily discuss *its representation in a parallel distributed-memory context*. All cells $T \in \mathcal{T}$ are assigned *a processor owner* $p = 1, \ldots, P$, with $P$ being the number of processors involved in the parallel computation. Given this cell-to-processor ownership mapping, the local portion of the global mesh $\mathcal{T}$ that processor $p$ stores locally, referred to as $\mathcal{T}^p \subset \mathcal{T}$, is defined as the union of two *disjoint* sets of cells, i.e. $\mathcal{T}^p \doteq \mathcal{T}^p_{\mathrm{L}} \cup \mathcal{T}^p_{\mathrm{G}}$, $\mathcal{T}^p_{\mathrm{L}} \cap \mathcal{T}^p_{\mathrm{G}} = \emptyset$, with $\mathcal{T}^p_{\mathrm{L}}$ being the *local* cells set and $\mathcal{T}^p_{\mathrm{G}}$ the so-called *ghost* cells set. The former includes those cells that processor $p$ owns. By construction, the set $\{\mathcal{T}^p_{\mathrm{L}}\}$, for $p = 1, \ldots, P$, is a partition of $\mathcal{T}$. The formal definition of the *ghost* cells set is also parameterized by a parameter $s$, with $0 \leq s < d$.

**Definition 2.3.10.** *The s-ghost cell set includes all cells $T \in \mathcal{T} \setminus \mathcal{T}^p_{\mathrm{L}}$ (i.e. off-processor cells) that are neighbours of cells in $\mathcal{T}^p_{\mathrm{L}}$ across n-faces with $n \geq s$ (see Definition 2.2.5).*

The sets $\mathcal{T}^p_{\mathrm{L}}$ and $\mathcal{T}^p_{\mathrm{G}}$ with $s = 0$, for $p = 1, 2$, are illustrated in Figure 2.1(b), 2.1(c), resp., for the forest-of-quadtrees in Figure 2.1(a). We note that a given cell in $\mathcal{T}$ belongs to only one $\mathcal{T}^p_{\mathrm{L}}$ set. It, however, belongs to either none, one, or more $\mathcal{T}^p_{\mathrm{G}}$ sets.

Since a given processor only stores a local portion of the global mesh, i.e. $\mathcal{T}^p$, it can only store a subset of the global VEFs set $\mathcal{F}$, denoted as $\mathcal{F}^p \subset \mathcal{F}$, and referred to as the proc-local VEFs set. $\mathcal{F}^p$ is generated by gluing together the local VEFs that lie on the boundary of (the local and ghost) cells in $\mathcal{T}^p$. The sets $\mathcal{T}_{T,f}$, $\mathcal{T}^-_{T,f}$, and $\mathcal{T}^+_{T,f}$ are also restricted to cells in $\mathcal{T}^p$. We denote their restriction as $\mathcal{T}^p_{T,f} \doteq \mathcal{T}_{T,f} \cap \mathcal{T}^p$, $\mathcal{T}^{p,+}_{T,f} \doteq$

$\mathcal{T}^p \cap \mathcal{T}^+_{T,f}$, and $\mathcal{T}^{p,-}_{T,f} \doteq \mathcal{T}^p \cap \mathcal{T}^-_{T,f}$, resp., for all $T \in \mathcal{T}^p$. *It turns out that, for all cells $T \in \mathcal{T}^p_{\mathrm{L}}$, and local $n$-faces $f \in \mathcal{F}_T$ such that $n \geq s$, the processor-local and global sets are equivalent by Definition 2.3.10.* Likewise, $\mathcal{T}_F$ and $\tilde{\mathcal{T}}_F$ are restricted to the cells in $\mathcal{T}^p$. For a given proc-local VEF $F \in \mathcal{F}^p$, we denote their restriction as $\mathcal{T}^p_F \doteq \mathcal{T}_F \cap \mathcal{T}^p$ and $\tilde{\mathcal{T}}^p_F \doteq \tilde{\mathcal{T}}_F \cap \mathcal{T}^p$, resp. By the equivalences in Propositions 2.3.3 and 2.3.4, we have that $\mathcal{T}^p_F = \mathcal{T}_F$ and $\tilde{\mathcal{T}}^p_F = \tilde{\mathcal{T}}_F$ for those $n$-faces $F \in \mathcal{F}^p$ such that $n \geq s$, and there is at least one local cell in $\mathcal{T}^p_F$. As shown along the chapter, this equivalence has key implications.

The set of proc-local VEFs $\mathcal{F}^p$, as in the case of cells, can be split into two disjoint sets of VEFs, i.e. $\mathcal{F}^p \doteq \mathcal{F}^p_{\mathrm{L}} \cup \mathcal{F}^p_{\mathrm{G}}$. This separation into the $\mathcal{F}^p_{\mathrm{L}}$ and $\mathcal{F}^p_{\mathrm{G}}$ sets is, however, of different nature compared to that of the cells.

**Definition 2.3.11** ($\mathcal{F}^p_{\mathrm{L}}$ and $\mathcal{F}^p_{\mathrm{G}}$ sets)**.** *A VEF $F \in \mathcal{F}^p$ is in $\mathcal{F}^p_{\mathrm{L}}$ if it fulfils one of the four following conditions: (a) at least one of the cells in $\mathcal{T}^p_F$ is local, i.e. $\mathcal{T}^p_F \cap \mathcal{T}^p_{\mathrm{L}} \neq \emptyset$; (b) all cells in $\mathcal{T}^p_F$ are ghost but at least one cell in $\tilde{\mathcal{T}}^p_F$ is local, i.e. $\tilde{\mathcal{T}}^p_F \cap \mathcal{T}^p_{\mathrm{L}} \neq \emptyset$; (c) all cells in $\mathcal{T}^p_F$ are ghost and there exists $F' \in \mathcal{F}^p_{\mathrm{L}}$ such that$F = O_{F'}$; (d) all cells in $\mathcal{T}^p_F$ are ghost and $F$ lies at the boundary of a VEF that fulfils (c). It is in $\mathcal{F}^p_{\mathrm{G}}$ otherwise.*

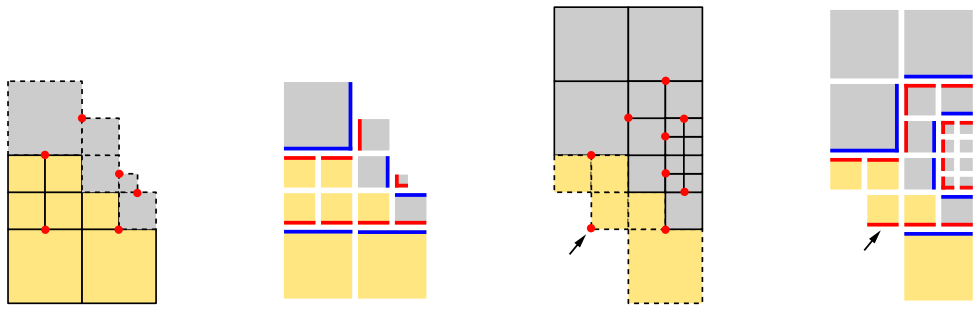
Thus, a global VEF $F \in \mathcal{F}$ might be in $\mathcal{F}^p_{\mathrm{L}}$ at multiple processors, while a local cell cannot. We will refer to this kind of VEFs as *interface* VEFs, while we will use the term *interior* to refer to those VEFs which are in $\mathcal{F}^p_{\mathrm{L}}$ only at a single processor. We denote the former and latter sets as $\mathcal{F}^p_{\Gamma}$ and $\mathcal{F}^p_{\mathrm{I}}$, resp. Clearly, $\{\mathcal{F}^p_{\mathrm{I}}, \mathcal{F}^p_{\Gamma}\}$ is a partition of $\mathcal{F}^p_{\mathrm{L}}$. Those vertices in the sets $\mathcal{F}^p_{\mathrm{L}}$, $\mathcal{F}^p_{\mathrm{I}}$, and $\mathcal{F}^p_{\Gamma}$ for the forest-of-quadtrees in Figure 2.1(a) distributed among two processors, are illustrated in Figure 2.1(b), 2.1(c) for $p = 1, 2$, resp. It remains to define a classification of VEFs of $\mathcal{T}^p$ into sets of regular and hanging VEFs suitable in a distributed context, i.e. that only requires processor-local information. With this aim, we define the concept of proc-regular and proc-hanging VEFs.

**Definition 2.3.12** (Proc-regular and proc-hanging VEFs)**.** *A VEF $F \in \mathcal{F}^p$ is proc-regular if $\tilde{\mathcal{T}}^p_F = \emptyset$, and proc-hanging otherwise.*

The set of proc-regular and proc-hanging VEFs are denoted as $\mathcal{F}^p_{\mathrm{R}}$ and $\mathcal{F}^p_{\mathrm{H}}$, resp. We stress that, for VEFs in $\mathcal{F}^p_{\mathrm{G}}$, this definition of proc-regular (resp., proc-hanging) VEFs is not equivalent to regular (resp., hanging) VEFs. In Figures 2.2(a)-2.2(d), we illustrate the $\mathcal{F}^p_{\mathrm{R}}$ and $\mathcal{F}^p_{\mathrm{H}}$ sets, with $p = 1, 2$, resp., for the forest-of-quadtrees in Figure 2.1(a). The ghost vertex and ghost face pointed by an arrow in Figure 2.2(c) and 2.2(d) are such that they belong to $\mathcal{F}_{\mathrm{H}}$ in $\mathcal{T}$ but to $\mathcal{F}^p_{\mathrm{R}}$ in $\mathcal{T}^p$. Fortunately, the algorithms that run on top of the mesh *only* require these definitions to be equivalent for $n$-faces in $\mathcal{F}^p_{\mathrm{L}}$, where the values of $n$ are again determined by the FE space at hand. This invariant holds, as stated in the next proposition.

**Proposition 2.3.13.** *For a 2:1 $k$-balanced forest-of-trees mesh $\mathcal{T}$ with $k$-ghost cells, then $\mathcal{F}^p_{\mathrm{R}} \cap \mathcal{F}^p_{\mathrm{L}} = \mathcal{F}_{\mathrm{R}} \cap \mathcal{F}^p_{\mathrm{L}}$ and $\mathcal{F}^p_{\mathrm{H}} \cap \mathcal{F}^p_{\mathrm{L}} = \mathcal{F}_{\mathrm{H}} \cap \mathcal{F}^p_{\mathrm{L}}$ for $n$-faces $F$ with $n \geq k$. This statement also holds for 0-faces when $k = 1$ (i.e. 2:1 1-balance and 1-ghost cell set).*

*Proof.* Let us first consider the case $F \in \mathcal{F}_{\mathrm{R}}^p \cap \mathcal{F}_{\mathrm{L}}^p$. As $F \in \mathcal{F}_{\mathrm{R}}^p$, then $\tilde{\mathcal{T}}_F^p = \emptyset$. Thus, $F \in \mathcal{F}_{\mathrm{L}}^p$ because of (a), (c), or (d) in Definition 2.3.11 (i.e. it cannot be in $\mathcal{F}_{\mathrm{L}}^p$ due to (b)). Let us assume that $F \in \mathcal{F}_{\mathrm{L}}^p$ because of (a) in Definition 2.3.11, i.e. at least one of the cells in $\mathcal{T}_F^p$ is local. For the first proposition statement, as $F$ fulfils (a), then $\mathcal{T}_F = \mathcal{T}_F^p$ and $\tilde{\mathcal{T}}_F = \tilde{\mathcal{T}}_F^p = \emptyset$ for those $n$-faces $F$ for which $n \geq k$ (see discussion above where the $\mathcal{T}_F^p$ and $\tilde{\mathcal{T}}_F^p$ sets are defined). For the second proposition statement, i.e. 0-faces $F$ and $k = 1$, we have that $\tilde{\mathcal{T}}_F = \tilde{\mathcal{T}}_F^p = \emptyset$ due to Proposition 2.3.3. Thus, $F \in \mathcal{F}_{\mathrm{R}}$ in both cases. If $F \in \mathcal{F}_{\mathrm{R}}^p$ is in $\mathcal{F}_{\mathrm{L}}^p$ because it holds (c) or (d) in Definition 2.3.11, $F \in \mathcal{F}_{\mathrm{R}}$ due to Proposition 2.3.8 in the case of the first proposition statement, and due to Proposition 2.3.9 in the case of the second. On the other hand, if $F \in \mathcal{F}_{\mathrm{R}} \cap \mathcal{F}_{\mathrm{L}}^p$ it is clearly in $\mathcal{F}_{\mathrm{R}}^p \cap \mathcal{F}_{\mathrm{L}}^p$, thus $\mathcal{F}_{\mathrm{R}}^p \cap \mathcal{F}_{\mathrm{L}}^p = \mathcal{F}_{\mathrm{R}} \cap \mathcal{F}_{\mathrm{L}}^p$. Using the fact that both $\{\mathcal{F}_{\mathrm{R}}^p \cap \mathcal{F}_{\mathrm{L}}^p, \mathcal{F}_{\mathrm{H}}^p \cap \mathcal{F}_{\mathrm{L}}^p, \}$ and $\{\mathcal{F}_{\mathrm{R}} \cap \mathcal{F}_{\mathrm{L}}^p, \mathcal{F}_{\mathrm{H}} \cap \mathcal{F}_{\mathrm{L}}^p\}$ are partitions of $\mathcal{F}_{\mathrm{L}}^p$, we readily have that $\mathcal{F}_{\mathrm{H}}^p \cap \mathcal{F}_{\mathrm{L}}^p = \mathcal{F}_{\mathrm{H}} \cap \mathcal{F}_{\mathrm{L}}^p$. □



(a) (b) (c) (d)

Figure 2.2: Hanging vertices (red circles) and faces (red lines) in $\mathcal{T}^1$ ((a) and (b), resp.) and $\mathcal{T}^2$ ((c) and (d), resp.). Those regular faces that are the owner VEF of a hanging VEF are depicted in blue. The hanging VEFs pointed out by the arrows in $\mathcal{T}^2$ are two examples of ghost VEFs that, while being hanging in $\mathcal{T}$, cannot be identified as such by processor $p = 2$, i.e. they are proc-regular.

### 2.3.3 Construction of a FE-suitable distributed adaptive mesh data structure

Algorithm 1 shows a simplified pseudocode of the process in charge of building the outer mesh layer in our framework. In a distributed-memory context, each processor $p$ executes its own instance of Algorithm 1, *with no communication at all involved among processors*. The algorithm assumes the global forest-of-trees mesh $\mathcal{T}$ to be 2:1 $k$-balanced, with $k$ being a user-level parameter to be set up according to Proposition 2.4.1; see Section 2.4.3. The input of Algorithm 1 are the proc-local variants of the cell neighbours across local $n$-faces sets presented in Section 2.2.4. From this input, Algorithm 1 glues together the local VEFs lying at the boundary of the mesh cells in $\mathcal{T}^p$ that satisfy the condition in Line 4, thus generating: (a) $[\mathcal{F}_T]$, for $T \in \mathcal{T}^p$; (b) the proc-local $\mathcal{F}^p$, $\mathcal{F}_{\mathrm{H}}^p$ and $\mathcal{F}_{\mathrm{R}}^p$ VEFs sets; (c) the proc-local $\mathcal{T}_F^p$ and $\tilde{\mathcal{T}}_F^p$ sets; (d) $O_F$ for all $F \in \mathcal{F}_{\mathrm{H}}^p$. Although omitted from Algorithm 1 in order to keep the presentation short, the actual algorithm also builds the $\mathcal{F}_{\mathrm{L}}^p$, $\mathcal{F}_{\mathrm{G}}^p$, $\mathcal{F}_{\mathrm{I}}^p$, and $\mathcal{F}_{\Gamma}^p$ sets (see Section 2.3.2).

The correctness of Algorithm 1 is mathematically supported by the propositions in Section 2.3.1 and 2.3.2. First, due to Proposition 2.3.3 and 2.3.4, Algorithm 1 is able to obtain $\mathcal{T}_F^p$ from the proc-local variants of the cell neighbours across local $n$-faces sets; see Lines 8-28. Second, Lines 29-34 are well-defined due to Proposition 2.3.7, 2.3.8, and 2.3.9. These lines are in charge of generating the $\tilde{\mathcal{T}}_F^p$ set for a hanging VEF that fulfils the condition in Line 4. To this end, Line 31 triggers the generation of the $\mathcal{T}_F^p$ and $\tilde{\mathcal{T}}_F^p$ sets for its owner VEF. By Proposition 2.3.7, we know that the owner VEF exists and due to Proposition 2.3.8 and 2.3.9, that the depth of the recursive call in Line 31 is always one, as the owner VEF of a hanging VEF is always a regular VEF (i.e. $\tilde{\mathcal{T}}_{O_F}^p = \emptyset$). Besides, due to Proposition 2.3.7, Line 32 is correct. Third, thanks to Proposition 2.3.13, it is safe to compute $\mathcal{F}_{\mathrm{R}}^p$ and $\mathcal{F}_{\mathrm{H}}^p$. This can be done at each processor without any inter-processor communication. The only VEFs for which this locally computed mesh information does not coincide with the global one are not required in practice for the parallel generation of global FE spaces and their numerical integration (see Section 2.4).

**Remark 2.3.14.** *Algorithm 1, at several points, has to determine the local VEF $\hat{f}$ of a neighbour cell $\hat{T}$ of $T$ across $g$ that either satisfies $\hat{f} \sim f$ or $\hat{f} \sim O_f$, with $g \in \mathcal{F}_T$ such that $f \subset \bar{g}$. This becomes a requirement that the forest-of-trees inner layer meshing engine has to be able to fulfil.*

**Remark 2.3.15.** *Algorithm 1 follows a fundamentally different approach from its counterpart in `deal.II` [27, Figure 1], and imposes a different set of requirements to the tree-based AMR mesh engine. This latter algorithm matches recursively the forest-of-octrees within the `deal.II` mesh data structure and the one in `p4est`. In order to do so, it starts from the root octants of the forest, and using a set of queries to `p4est`, reconstructs the local part of the mesh as by-product of a full hierarchical AMR process in which the mesh is incrementally transformed by successive refinement and coarsening steps, until the leaves in the former match those in the latter. At each step, the aforementioned set of queries lets `deal.II` determine which cells in the current mesh have to be refined and coarsened in the path towards the final match. While this approach takes into account the prior state of the forest-of-octrees within the `deal.II` mesh data structure, and thus does not need to actually perform changes in the data structure if no changes have been performed in its `p4est` counterpart, it still has to go over a full hierarchical AMR process to determine whether there is a match or not. Besides, when significant changes have been performed in `p4est`, it turns to be a quite complex, harder-to-update data structure (i.e. cell hierarchy, n-faces hierarchy, etc.), than the one introduced in this chapter. In the experiments in Section 2.5, the implementation of the latter data structure in `FEMPAR` turns out to be up to twice faster than the one in `deal.II`, even if Algorithm 1 (as now conceived) does not try to exploit the previous state of the mesh data structure. We believe that this performance improvement is algorithmic in nature, although we could not fully discard it to be caused by technical differences in the HPC implementation of both libraries.*

**Algorithm 1:** Construction of FE-suitable adaptive mesh from cell neighbours across $n$-faces.

1 $\mathcal{F}_{\mathrm{R}}^{p} \leftarrow \emptyset$; $\mathcal{F}_{\mathrm{H}}^{p} \leftarrow \emptyset$
2 **for** $T \in \mathcal{T}^{p}$ **do** /* loop over all cells in the local portion of processor $p$ */
3  **for** $f \in \mathcal{F}_T$ **do** /* loop over current cell local VEFs */
4  **if** ($f$ is a 0-face and $k \leq 1$) or ($f$ is an $n$-face with $n \geq k$) **then**
5  **if** $[f]$ not created yet **then** /* current VEF first visited from any cell */
6  $[f] \leftarrow F \leftarrow$ `new_proc_local_vef` (); /* create new VEF equivalence class */
7  $\mathcal{T}_F^p \leftarrow \{T\}$; $\tilde{\mathcal{T}}_F^p \leftarrow \emptyset$ /* initialize the $\mathcal{T}_F^p$ and $\tilde{\mathcal{T}}_F^p$ sets */
8  **for** $g \in \mathcal{F}_T : f \subset \bar{g}$ **do** /* Generate $\mathcal{T}_F^p$ (Lines 8-28) */
9  **if** $\mathcal{T}_{T,g}^{p} \neq \emptyset$ **then**
10  **for** $\hat{T} \in \mathcal{T}_{T,g}^{p}$ **do** /* loop over conformal neighbours of $T$ across $g$ */
11  $[\hat{f}] \leftarrow F$, with $\hat{f} \in \mathcal{F}_{\hat{T}} : \hat{f} \sim f$; $\mathcal{T}_F^p \leftarrow \mathcal{T}_F^p \cup \{\hat{T}\}$
12  **end**
13  **end**
14  **if** $\mathcal{T}_{T,g}^{p,-} \neq \emptyset$ **then**
15  **if** $f$ is a 0-face **then**
16  **for** $\hat{T} \in \mathcal{T}_{T,g}^{p,-} : \exists\, \hat{f} \in \mathcal{F}_{\hat{T}}$ satisfying $\hat{f} \sim f$ **do**
17  $[\hat{f}] \leftarrow F$; $\mathcal{T}_F^p \leftarrow \mathcal{T}_F^p \cup \{\hat{T}\}$
18  **end**
19  **end**
20  **end**
21  **if** $\mathcal{T}_{T,g}^{p,+} \neq \emptyset$ **then**
22  **if** $f$ is a 0-face **then**
23  **for** $\hat{T} \in \mathcal{T}_{T,g}^{p,+} : \exists\, \hat{f} \in \mathcal{F}_{\hat{T}}$ satisfying $\hat{f} \sim f$ **do**
24  $[\hat{f}] \leftarrow F$; $\mathcal{T}_F^p \leftarrow \mathcal{T}_F^p \cup \{\hat{T}\}$
25  **end**
26  **end**
27  **end**
28  **end**
29  **if** $\{T' \in \mathcal{S}_{T,f}^{-} : \nexists f' \in \mathcal{F}_{T'}, f' \sim f\} \neq \emptyset$ **then** /* is $f$ hanging? Generate $\tilde{\mathcal{T}}_F^p$ (Lines 29-34) */
30  Let $\hat{T}$ be an arbitrary neighbour cell in $\{T' \in \mathcal{S}_{T,f}^{-} : \nexists f' \in \mathcal{F}_{T'}, f' \sim f\}$ and $\hat{f} \in \mathcal{F}_{\hat{T}}$ such that $\hat{f} \sim O_f$
31  Execute lines 5-36 replacing $f \equiv \hat{f}$ and $T \equiv \hat{T}$
32  $\tilde{\mathcal{T}}_F^p \leftarrow \mathcal{T}_{O_F}^p$
33  $O_F \leftarrow [O_f]$
34  **end**
35  $(\tilde{\mathcal{T}}_F^p = \emptyset)$ ? $\mathcal{F}_{\mathrm{R}}^{p} \leftarrow \mathcal{F}_{\mathrm{R}}^{p} \cup \{F\} : \mathcal{F}_{\mathrm{H}}^{p} \leftarrow \mathcal{F}_{\mathrm{H}}^{p} \cup \{F\}$ /* "X ? Y : Z" denotes the conditional ternary operator */
36  **end**
37  **end**
38  **end**
39 **end**
40 $\mathcal{F}^{p} \leftarrow \mathcal{F}_{\mathrm{R}}^{p} \cup \mathcal{F}_{\mathrm{H}}^{p}$

**Remark 2.3.16** (Implementation remark)**.** *Algorithm 1 can be implemented for forest-of-octrees using* `p4est` *as the inner layer mesh engine. In particular, this library offers the so-called* `pXest_mesh_t` *data structure, with* `X`=4, 8 *for* $d = 2, 3$*, resp., which provides neighbours across vertices, edges, and faces.* `pXest_mesh_t` *is constructed by means of the so-called universal mesh topology iterators, introduced in [99]; see also* `Iterate` *in Section 2.2.6. As a consequence,* `pXest_mesh_t` *only provides neighbours for cells in* $\mathcal{T}_{\mathrm{L}}^{p}$*. However, Algorithm 1 requires these for cells in* $\mathcal{T}_{\mathrm{G}}^{p}$ *as well. To this end, we developed a nearest neighbours exchange communication stage to complete* `pXest_mesh_t` *prior to the actual execution of Algorithm 1. Apart from this, it turns out that, in* `p4est` *v2.2 (latest stable release at the date of writing), the* `p8est_mesh_t` *data structure does not provide the following required information: (a) cell neighbours across edges sets; (b) vertices hanging on a coarse edge across which there are vertex neighbours satisfying Definition 2.2.5. Fortunately, we could obtain them using edge topology iterators [99].*

## 2.4 Handling conforming FE spatial discretisations on non-conforming meshes

### 2.4.1 Problem statement and its conforming FE spatial discretisation

We aim at solving a PDE problem in $\Omega$, supplemented with appropriate boundary conditions on $\partial\Omega$. In order to end up with a computable version of this problem, the FE method requires finite-dimensional spaces of functions $\mathcal{V}_h$ with some approximability properties. A particular type of FE methods, referred to *conforming FE methods*, require $\mathcal{V}_h$ to be *a conforming FE space*, i.e. a subspace of its infinite-dimensional counterpart $\mathcal{V}$, i.e. $\mathcal{V}_h \subset \mathcal{V}$.

In practice, FE spaces are made of functions which are piece-wise polynomial (i.e. smooth) on each cell. Thus, the conformity requirement translates into a trace continuity requirement on the interface among the mesh cells. When the mesh is conforming, the trace continuity requirement for conformity can be achieved combining two different ingredients, that are defined *abstractly* as follows. First, we define the local space of functions $\mathcal{Q}(T)$ unisolvent with respect to a set of local DOFs (functionals) $\Sigma_T$ for every cell $T \in \mathcal{T}$.[5] Second, with these cell-wise spaces, the global FE space $\mathcal{V}_h$ is determined by an equivalence class to glue together local DOFs and create the set of global DOFs $\Sigma$. The equivalent class (global DOF) of a local DOF $\alpha$ is represented with $[\alpha]$; $[\cdot]$ is the so-called local-to-global DOF map.

The equivalence relation and the local FE spaces complement each other strategically such that the continuity of global DOF values resulting from gluing together local DOFs, implies the required trace continuity of $\mathcal{V}_h$. Examples of grad-, curl- and div-conforming finite-dimensional spaces are those which are built from Lagrangian, Nédélec (a.k.a. edge) and Raviart-Thomas (a.k.a. face) FEs by imposing trace continuity,

[5] Here, we assume the same polynomial space in all cells.

tangent and normal component trace continuity, resp. Underlying their construction, there exists the notion of global (local) VEF *owner* of a global (local) DOF, i.e. the global (local) VEFs of the triangulation *may carry out* global (local) DOFs. For example, for first-order Lagrangian FEs spaces, only the 0-faces (i.e. vertices) carry out (own) DOFs, while for first-order Nédélec ones, only the 1-faces do. We use $\Sigma_T^f$ to refer to the set of local DOFs owned by $f \in \mathcal{F}_T$, and $\Sigma_F \subset \Sigma$ denotes the set of all equivalence classes (i.e. global DOFs) resulting from the application of the equivalence relation to all cell-local DOFs $\alpha \in \Sigma_T^f$, for all $K \in \mathcal{T}_F$ such that $F = [f]$. For conciseness, we do not cover here how the abstract ingredients are defined for each specific conforming FE space at hand. A comprehensive exposition can be found at Section [19, Section 3].

### 2.4.2 Generation of proc-local DOFs

In our distributed-memory framework, each processor restricts itself to the generation of those equivalence classes (i.e. global DOFs) of $\Sigma$ corresponding to $n$-faces (i.e. VEFs and cells) in $\mathcal{T}^p$. Given $F \in \mathcal{F}^p$, we denote by $\Sigma_F^p$ the set of all equivalence classes on $F$ at processor $p$. We define $\Sigma^p \doteq \bigcup_{F \in \{\mathcal{F}^p \cup \mathcal{T}^p\}} \Sigma_F^p$, with $\Sigma^p \subset \Sigma$, and refer to it as the proc-local DOFs set. We also define:

$$\Sigma_{\mathrm{R}}^p \doteq \cup_{F \in \{\mathcal{F}_{\mathrm{R}}^p \cup \mathcal{T}^p\}} \Sigma_F^p, \quad \Sigma_{\mathrm{H}}^p \doteq \cup_{F \in \mathcal{F}_H^p} \Sigma_F^p, \quad \Sigma_{\mathrm{L}}^p \doteq \cup_{F \in \{\mathcal{F}_{\mathrm{L}}^p \cup \mathcal{T}_{\mathrm{L}}^p\}} \Sigma_F^p,$$
$$\Sigma_{\mathrm{G}}^p \doteq \cup_{F \in \{\mathcal{F}_{\mathrm{G}}^p \cup \mathcal{T}_{\mathrm{G}}^p\}} \Sigma_F^p, \quad \Sigma_{\mathrm{I}}^p \doteq \cup_{F \in \{\mathcal{F}_{\mathrm{I}}^p \cup \mathcal{T}_{\mathrm{L}}^p\}} \Sigma_F^p, \quad \Sigma_{\Gamma}^p \doteq \cup_{F \in \{\mathcal{F}_{\Gamma}^p\}} \Sigma_F^p.$$

Assuming that one only requires to build a distributed subassembled linear system (see Section 2.4.3), the need for a local-to-global index map $[\cdot]$ can be by-passed by combining a local-to-proc-local index map, denoted hereafter as $[\cdot]_p$, and Remark 2.4.4. In other words, local DOFs $\alpha$ in $\Sigma_T$, for $T \in \mathcal{T}^p$, are labelled with a proc-local identifier $[\alpha]_p$ in the range $\{1, \ldots, |\Sigma^p|\}$. Thus, one can prevent dealing with 64-bit integers (i.e. global DOF identifiers across the whole domain) and use 32-bit ones instead (i.e. processor-local DOF identifiers within each subdomain). This reduces memory consumption and bandwidth demands. Furthermore, using proc-local DOF identifiers, to determine whether a given proc-local DOF in $\Sigma^p$ is in a given set, say $\Sigma_{\mathrm{H}}^p$, can be implemented *in constant time*, whereas with global DOF identifiers, one has to pay the cost of a search in a global index set.

### 2.4.3 Hanging DOF constraints. Which are strictly required locally? When/why can they be resolved locally?

Conformity on the interface of cells at different refinement levels must be enforced explicitly by introducing additional linear algebraic constraints in $\mathcal{V}_h$, i.e. the so-called hanging DOF constraints. The structure and set up of such constraints is well established knowledge; see, e.g. [157] and [148] for grad- and curl-conforming FE spaces, resp.

*In a distributed-memory computing environment, however, it is (much) more intricate.* In general, distributed-memory computers are most efficiently exploited if one maximizes local work while minimizing inter-processor communication. To this end, in this context, we aim to build without communication a local subassembled portion of the global linear system coefficient matrix (and right-hand-side vector). This local portion is of size $\Sigma_{\mathrm{R}}^p \cap \Sigma_{\mathrm{L}}^p$, and is built by means of a FE assembly process restricted to the cells $T \in \mathcal{T}_{\mathrm{L}}^p$ (i.e. local cells).[6] It follows that only the constraints on hanging DOFs touched by local cells have to be resolved in order to build the subassembled portion. In order to be able to set up and apply such constraints locally, without communication, we have to ensure that there is room in $\mathcal{T}^p$ for any of the free DOFs whose values constraint the one of the hanging DOF touched by local cells. In other words, we have to guarantee that the hanging DOF constraints dependencies do not expand beyond the single layer of ghost cells. Proposition 2.4.1 precisely answers under which conditions this is guaranteed. We observe that the current literature clearly lacks mathematical rigour to address when and why these constraints can be resolved locally; see, e.g. [27, Section 3.3.4].

Let us denote with $D(\mathcal{V}_h)$ the dimension of the $n$-face of lowest dimension that has a non-empty set of owned DOFs. For instance, it is 0 for Lagrangian FEs (vertices own DOFs), 1 for edge FEs (edges own DOFs, vertices do not), 2 for face FEs (faces own DOFs, vertices/edges do not), and $d$ (the space dimension) for discontinuous Galerkin (DG) (only cells own DOFs since no inter-cell continuity must be enforced). We have the following result.

**Proposition 2.4.1.** *Let us consider a distributed 2:1 k-balanced forest-of-trees mesh $\mathcal{T}$ with k-ghost cells. If* $\max(1, D(\mathcal{V}_h)) \geq k$*, all hanging DOF constraints are* direct*, i.e. they only depend on regular DOFs. Furthermore, all constraints of DOFs owned by n-faces in $\mathcal{F}_{\mathrm{H}} \cap \mathcal{F}_{\mathrm{L}}^p$ only depend on DOFs owned by $\mathcal{F}_{\mathrm{R}} \cap \mathcal{F}_{\mathrm{L}}^p$, and thus can be solved in parallel with processor-local information.*

*Proof.* The fact that the constraints are direct is an immediate consequence of how the constraints are defined [157], Proposition 2.3.8, and (for $D(\mathcal{V}_h) = 0$) Proposition 2.3.9. On the other hand, for DOFs owned by VEFs $F \in \mathcal{F}_{\mathrm{H}} \cap \mathcal{F}_{\mathrm{L}}^p$, then $O_F$, and any of its boundary VEFs are in $\mathcal{F}_{\mathrm{L}}^p$, due to Definition 2.3.11, i.e. such DOFs are only constrained by DOFs owned by $\mathcal{F}_{\mathrm{R}} \cap \mathcal{F}_{\mathrm{L}}^p$. Since $k$-ghost cells are locally accessible, then $O_F$, and any of its boundary VEFs are locally accessible, and thus the constraints on DOFs owned by such VEFs $F$ can be computed locally. □

In other words, as by-product of the journey to Proposition 2.4.1, we can prove the correctness of the parallel algorithms in our framework. If the conditions of Proposition 2.4.1 are not fulfilled, one may have an incorrect parallel FE solver if one does

[6] We note that this assembly process has to deal with the application of hanging DOF constraints, following, e.g. the transformation approach in [170]. In a nutshell, this approach carries out the constraint application during FE assembly, in order to directly build the so-called constrained (a.k.a. condensed) linear system as the cell-local matrices and force vectors are built and assembled.

not add additional iterative communication steps [47]. This is illustrated in Figure 2.3. Even more than that, $k = \max(1, D(\mathcal{V}_h))$ is the largest possible value for which the parallel FE solver is still guaranteed to be correct, i.e. the value of $k$ that minimizes $|\mathcal{T}|$. As assessed in Section 2.5, and depending on the particular refinement pattern at hand, choosing the largest possible $k$ can lead to a noticeably reduction in the number of mesh cells and computational times (compared to the most conservative value $k = 0$).

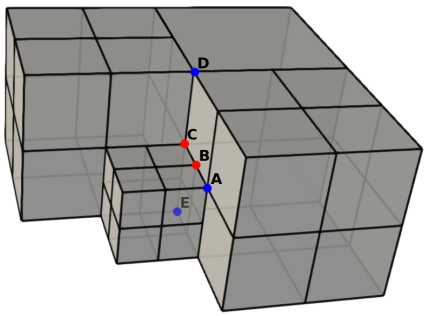



Figure 2.3: 2-balanced forest-of-octrees with 4 octrees (i.e. $|\mathcal{C}| = 4$) and a global conforming FE space grounded on trilinear FEs, i.e. $k = 2$ and $D(\mathcal{V}_h) = 0$. For visualization purposes, some of the cells of the mesh are hidden. All these cells are such that $\ell(T) = 1$. In this particular scenario, the conditions of Prop. 2.4.1 are not fulfilled, as $\max(1, D(\mathcal{V}_h)) = 1 \ngeq k = 2$. The regular and hanging DOFs of interest are marked with blue and red circles, resp. The hanging DOF labelled as $B$ is constrained by $A$ and $C$, which is in turn constrained by $D$ and $E$. Assuming a one-to-one mapping among cells and processors, the processor that owns the cell that contains the edge connecting $A$ and $B$ cannot locally compute the linear constraint associated with $B$, because the cell that contains the edge connecting $D$ and $E$ is not a ghost cell in this processor. We note that in this case the parallel code does not actually crash, but builds a FE space $\mathcal{V}_h$ which is non-conforming, thus, leading to an incorrect parallel FE solver. In order to solve this problem, the root of the octree located in the upper, right corner must be refined so that the forest becomes 1-balanced, and the conditions of Proposition 2.4.1 are fulfilled.

### 2.4.4 Subdomain-wise assembly of proc-regular interface DOF values

In a distributed-memory context, non-overlapping domain decomposition solvers need, in the iterative solution process of $\boldsymbol{A}\boldsymbol{u} = \boldsymbol{b}$, to assemble, subdomain-wise, the (sub-assembled) values corresponding to proc-regular interface DOFs (i.e. DOFs in the $\Sigma_{\mathrm{R}}^p \cap \Sigma_{\Gamma}^p$ set).[7] This operation can be stated as follows. Given a distributed *subassembled* (i.e. partially-summed) vector $\boldsymbol{x}$, i.e. a vector such that entries of $\boldsymbol{x}^p$ corresponding to proc-regular interface DOFs hold partial contributions to the corresponding entries in

[7] We note that hanging DOF constraints are eliminated from the system during FE assembly, i.e. we do not allocate an equation/unknown for DOFs in the $\Sigma_{\mathrm{H}}^p \cap \Sigma_{\mathrm{L}}^p$ set.

$\boldsymbol{x}$, *transform* $\boldsymbol{x}$ *such that it becomes fully-assembled*, i.e. entries of $\boldsymbol{x}^p$ corresponding to proc-regular interface DOFs contain the (i.e. fully summed) value of the corresponding entries in $\boldsymbol{x}$.

We define the set $\mathcal{S}^{\text{proc}}(g) \doteq \{q : g \in \Sigma_{\text{R}}^q \cap \Sigma_{\Gamma}^q\}$ for any $g \in \Sigma_{\text{R}}^p \cap \Sigma_{\Gamma}^p$. For each of these DOFs, we assign a *processor owner* among the set of processors in $\mathcal{S}^{\text{proc}}(g)$ (using an algorithm presented later in this section). The owner processor is denoted by $\mathcal{O}^{\text{proc}}(g) \in \mathcal{S}^{\text{proc}}(g)$. The rest of processors in $\mathcal{S}^{\text{proc}}(g)$ become non-owner processors. As required for the exposition, we also define equivalently $\mathcal{S}^{\text{proc}}(F)$ and $\mathcal{O}^{\text{proc}}(F)$ for any VEF $F \in \mathcal{F}_{\Gamma}^p \cap \mathcal{F}_{\text{R}}^p$. Thus, we can write $\mathcal{S}^{\text{proc}}(g) \doteq \mathcal{S}^{\text{proc}}(F)$ (resp. $\mathcal{O}^{\text{proc}}(g) \doteq \mathcal{O}^{\text{proc}}(F)$), for any $g \in \Sigma_F^p$. In any case, the algorithms in this section do not compute $\mathcal{S}^{\text{proc}}(F)$ and $\mathcal{O}^{\text{proc}}(F)$, but $\mathcal{O}^{\text{proc}}(g)$ and $\mathcal{S}^{\text{proc}}(g)$ directly. This may lead to computational savings, as not all VEFs necessarily own DOFs. (This obviously depends on the FE at hand.) The subdomain-wise assembly operation is split into two communication stages. In the first stage, referred to as **S1**, processor owners receive the corresponding partially-summed contributions from non-owner processors, and reduce-sum the partial sums received into fully-assembled values (at processor owners). In the second stage, referred to as **S2**, processor owners send fully-summed values resulting from the first stage to non-owner processors, so that all processors end up with fully-assembled proc-regular interface DOF values. In the sequel, we sketch the process that sets up the communication patterns underlying **S1** and **S2**.

Each processor needs to figure out the following information in order to set up the **S1** communication pattern. *On the receive side*, $p$ has to determine the set of processor identifiers from which it receives data. Let us denote this set by $\mathcal{S}_{\text{rcv}}^p$. Besides, $p$ needs to figure out, for each processor $q \in \mathcal{S}_{\text{rcv}}^p$, the DOFs whose values have to be accumulated to the (partially-summed) values received from $q$. Let us denote this set by $\Sigma_{\text{rcv}}^{p \leftarrow q}$, for all $q \in \mathcal{S}_{\text{rcv}}^p$. Thus, $|\Sigma_{\text{rcv}}^{p \leftarrow q}|$ is the amount of data items that $p$ receives from $q$. *On the send side*, each processor $p$ has to determine $\mathcal{S}_{\text{snd}}^p$ and $\Sigma_{\text{snd}}^{p \rightarrow q}$, for each $q \in \mathcal{S}_{\text{snd}}^p$, i.e. the set of processor identifiers to which $p$ sends data, and the DOFs whose values have to be sent to each processor $q \in \mathcal{S}_{\text{snd}}^p$, resp. Likewise, $|\Sigma_{\text{snd}}^{p \rightarrow q}|$ is the amount of vector entries that $p$ sends to $q$. On the other hand, the communication pattern for **S2** can be very easily determined from the one of **S1** by swapping send-side sets and receive-side sets.

Algorithm 2 sketches the process in charge of generating the **S1** communication pattern. The inputs of the algorithm are $\mathcal{O}^{\text{proc}}(g)$, for $g \in \Sigma_{\text{R}}^p \cap \Sigma_{\Gamma}^p$, and $\mathcal{S}^{\text{proc}}(g)$ for $g \in \Sigma_{\text{R}}^p \cap \Sigma_{\Gamma}^p$ such that $\mathcal{O}^{\text{proc}}(g) = p$. (Note that the algorithm does not actually require $\mathcal{S}^{\text{proc}}(g)$ if $\mathcal{O}^{\text{proc}}(g) \neq p$.) From these inputs, the algorithm generates the $\mathcal{S}_{\text{rcv}}^p$, $\Sigma_{\text{rcv}}^{p \leftarrow q}$, $\mathcal{S}_{\text{snd}}^p$, and $\Sigma_{\text{snd}}^{p \rightarrow q}$ sets.

It remains to discuss how $\mathcal{O}^{\text{proc}}(g)$ and $\mathcal{S}^{\text{proc}}(g)$ are generated. Algorithm 3 presents a sketch of the process in charge of determining $\mathcal{O}^{\text{proc}}(g)$, for $g \in \Sigma_{\text{R}}^p \cap \Sigma_{\Gamma}^p$, and $\mathcal{S}^{\text{proc}}(g)$ for $g \in \Sigma_{\text{R}}^p \cap \Sigma_{\Gamma}^p$ such that $\mathcal{O}^{\text{proc}}(g) = p$ (i.e. the input of Algorithm 2). Algorithm 3 is split into two stages. On the one hand, a local stage spanning Lines 1-13, in which each

**Algorithm 2:** Determine $\mathcal{S}^p_{\text{rcv}}$, $\Sigma^{p\leftarrow q}_{\text{rcv}}$, $\mathcal{S}^p_{\text{snd}}$ and $\Sigma^{p\to q}_{\text{snd}}$.

```
1  for F ∈ F_Γ^p do                                          /* loop over interface VEFs */
2      if F ∈ F_R^p then                                     /* current VEF is proc-regular */
3          for g ∈ Σ_F^p do
4              q ← O^proc(g)
5              if p = q then                                 /* i am the owner of g */
6                  for q ∈ S^proc(g) \ {p} do
7                      S^p_rcv ← S^p_rcv ∪ {q}; Σ^{p←q}_rcv ← Σ^{p←q}_rcv ∪ {g}
8                  end
9              else                                          /* a remote processor q is the owner of g */
10                 S^p_snd ← S^p_snd ∪ {q}; Σ^{p→q}_snd ← Σ^{p→q}_snd ∪ {g}
11             end
12         end
13     end
14 end
```

processor $p$ builds as much as it can of $\mathcal{O}^{\text{proc}}(\cdot)$ and $\mathcal{S}^{\text{proc}}(\cdot)$ only using local information. The local stage of Algorithm 3 also walks through proc-hanging interface VEFs; see Line 1 and Lines 8-12. This is justified by the fact that, due to the hanging DOFs constraints, all $g \in \Sigma^p_{\mathcal{O}_F}$ become local in all processors that own cells in $\mathcal{T}^p_F$. On the other hand, Algorithm 3 features a communication stage spanning Lines 14-30, in which the processors complete the partially constructed version of $\mathcal{O}^{\text{proc}}(\cdot)$ and $\mathcal{S}^{\text{proc}}(\cdot)$ during the first stage. The communication stage is composed by pack (Lines 15-20), nearest-neighbour exchange (Line 21), and unpack (Lines 22-30) steps. In the rest of the section we mathematically prove why this algorithm is correct.

Let us define the owner processor of a DOF $g \in \Sigma^p_F$ for $F \in \mathcal{F}^p_\Gamma \cap \mathcal{F}^p_R$ as $\mathcal{O}^{\text{proc}}(g) \doteq \max_{T\in\mathcal{T}_F} \mathcal{O}^{\text{proc}}(T)$, with $\mathcal{O}^{\text{proc}}(T)$ denoting the processor owner of $T \in \mathcal{T}$; see Section 2.3.2. The following result holds.

**Proposition 2.4.2.** *Let us consider a distributed forest-of-trees mesh $\mathcal{T}$ with s-ghost cells, where $D(\mathcal{V}_h) \geq s$. For any processor $p$, the owner processor of $g \in \Sigma^p_F$, for $F \in \mathcal{F}^p_\Gamma \cap \mathcal{F}^p_R$, can be locally computed as $\mathcal{O}^{\text{proc}}(g) = \max_{T\in\mathcal{T}^p_F} \mathcal{O}^{\text{proc}}(T)$ if $\mathcal{T}^p_F \cap \mathcal{T}^p_L \neq \emptyset$ (see Line 6). Otherwise (i.e. $\mathcal{T}^p_F \cap \mathcal{T}^p_L = \emptyset$), any neighbour processor that owns a ghost cell in $\mathcal{T}^p_F$ can compute $\mathcal{O}^{\text{proc}}(g)$ (and, thus, $p$ can fetch this information from any of these neighbours).*

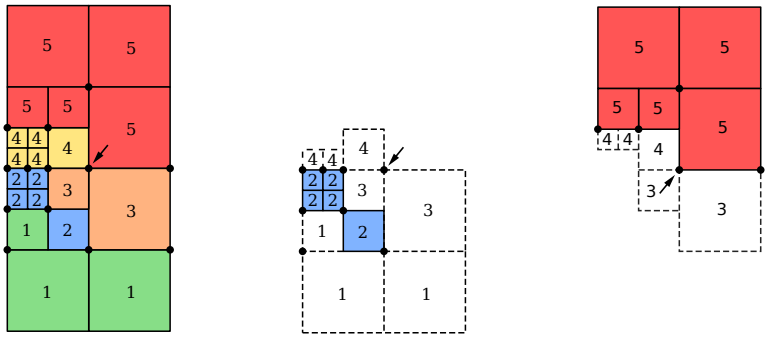
*Proof.* If $D(\mathcal{V}_h) \geq s$, then for $g \in \Sigma^p_R \cap \Sigma^p_\Gamma$ such that $g \in \Sigma^p_F$ with $\mathcal{T}^p_F \cap \mathcal{T}^p_L \neq \emptyset$, then $\mathcal{T}^p_F = \mathcal{T}_F$ (see Section 2.3.2). Thus, all processors in $\mathcal{S}^{\text{proc}}(g)$ such that $g$ is surrounded by at least one local cell can compute $\mathcal{O}^{\text{proc}}(g)$ locally. The last part of the proposition is straightforward. □

As a result of this proposition, the communication stage in Algorithm 3 (Lines 14-30) is guaranteed to obtain $\mathcal{O}^{\text{proc}}(g)$ on processors such that $\mathcal{T}^p_F \cap \mathcal{T}^p_L = \emptyset$ as well.

On the other hand, with regard to $\mathcal{S}^{\text{proc}}(g)$, we note that a processor that is owner of a proc-regular interface DOF $g$, i.e. $\mathcal{O}^{\text{proc}}(g) = p$, may not able to determine the full $\mathcal{S}^{\text{proc}}(g)$ set solely using local information (i.e. local cells plus a layer of $s$-ghost cells). This scenario is illustrated in Figure 2.4(c) for the DOF at processor $p = 5$ pointed by

**Algorithm 3:** Determine $\mathcal{O}^{\text{proc}}(\cdot)$ and $\mathcal{S}^{\text{proc}}(\cdot)$ for proc-regular interface DOFs.

1 **for** $F \in \mathcal{F}_\Gamma^p$ **do** /* loop over interface VEFs */
2 $\quad \mathcal{W} \leftarrow \bigcup_{T \in \mathcal{T}_F^p} \mathcal{O}^{\text{proc}}(T)$
3 $\quad$ **if** $F \in \mathcal{F}_R^p$ **then** /* current VEF is proc-regular */
4 $\quad\quad$ **for** $g \in \Sigma_F^p$ **do**
5 $\quad\quad\quad \mathcal{S}^{\text{proc}}(g) \leftarrow \mathcal{S}^{\text{proc}}(g) \cup \mathcal{W}$
6 $\quad\quad\quad$ **if** $\mathcal{T}_F^p \cap \mathcal{T}_L^p \neq \emptyset$ **then** $\mathcal{O}^{\text{proc}}(g) \leftarrow \max_{T \in \mathcal{T}_F^p} \mathcal{O}^{\text{proc}}(T)$ /* at least one cell surrounding $F$ is local */
7 $\quad\quad$ **end**
8 $\quad$ **else** /* current VEF is proc-hanging */
9 $\quad\quad$ **if** $\Sigma_F^p \neq \emptyset$ **then**
10 $\quad\quad\quad$ **for** $g \in \Sigma_{\overline{O_F}}^p$ **do** $\mathcal{S}^{\text{proc}}(g) \leftarrow \mathcal{S}^{\text{proc}}(g) \cup \mathcal{W}$ /* loop over DOFs owned by $\overline{O_F}$ */
11 $\quad\quad$ **end**
12 $\quad$ **end**
13 **end**
14 Allocate communication buffers $\mathcal{O}_{\text{buf}}(\cdot,\cdot)$ and $\mathcal{S}_{\text{buf}}(\cdot,\cdot)$ to size $|\mathcal{T}^p| \times |\Sigma_T|$
15 **for** $F \in \mathcal{F}_R^p \cap \mathcal{F}_\Gamma^p$ **do** /* pack local data into communication buffers (Lines 15-20) */
16 $\quad$ **for** $T \in \mathcal{T}_F^p$ such that $T \in \mathcal{T}_L^p$ **do** /* loop over local cells around $F$ */
17 $\quad\quad$ Find $f \in \mathcal{F}_T$ such that $[f] = F$
18 $\quad\quad$ **for** $\alpha \in \Sigma_T^f$ **do** $\mathcal{X}_{\text{buf}}(K,\alpha) \leftarrow \mathcal{X}^{\text{proc}}([\alpha]_p)$, with $\mathcal{X} = \mathcal{O}, \mathcal{S}$
19 $\quad$ **end**
20 **end**
21 Fetch ghost cell data of $\mathcal{O}_{\text{buf}}(\cdot,\cdot)$ and $\mathcal{S}_{\text{buf}}(\cdot,\cdot)$ from remote processors via nearest-neighbours exchange
22 **for** $T \in \mathcal{T}_G^p$ **do** /* unpack data from communication buffers (Lines 22-30) */
23 $\quad$ **for** $f \in \mathcal{F}_T$ such that $[F] \in \mathcal{F}_L^p \cap \mathcal{F}_R^p$ **do**
24 $\quad\quad$ **if** $\mathcal{T}_F^p \cap \mathcal{T}_L^p \neq \emptyset$ **then**
25 $\quad\quad\quad$ **for** $\alpha \in \Sigma_T^f$ **do** $\mathcal{S}^{\text{proc}}([\alpha]_p) \leftarrow \mathcal{S}^{\text{proc}}([\alpha]_p) \cup \mathcal{S}_{\text{buf}}^p(K,\alpha)$ /* augment $\mathcal{S}^{\text{proc}}(\cdot)$ with neighbours data */
26 $\quad\quad$ **else**
27 $\quad\quad\quad$ **for** $\alpha \in \Sigma_T^f$ **do** $\mathcal{O}^{\text{proc}}([\alpha]_p) \leftarrow \mathcal{O}_{\text{buf}}^p(K,\alpha)$ /* remote neighbours determine $\mathcal{O}^{\text{proc}}(\cdot)$ */
28 $\quad\quad$ **end**
29 $\quad$ **end**
30 **end**




(a) $\Sigma \cap \bigcup_p \Sigma^p_{\mathrm{R}} \cap \Sigma^p_{\mathrm{L}}$. (b) $\Sigma^2_R \cap \Sigma^2_\Gamma$. (c) $\Sigma^5_R \cap \Sigma^5_\Gamma$.

Figure 2.4: Proc-regular interface DOFs (black circles) of a global conforming FE space grounded on bilinear Lagrangian FEs on top of a 2:1 0-balanced forest-of-quadtrees mesh with two quadtrees distributed (non-uniformly) among $P = 5$ processors with 0-ghost cells. The DOF pointed out by an arrow in Figure 2.4(c) is such that processor $p = 5$, even being its owner, is not able to determine the full $\mathcal{S}^{\mathrm{proc}}(g)$ set solely using local information. Fortunately, as stated by Prop. 2.4.3, this set can be obtained combining local information and a single nearest neighbour communication.

an arrow. Using only local information, $\mathcal{S}_5(g) = \{3,4,5\}$. However, processor 2 is an element of $\mathcal{S}_5(g)$ as well. The cell owned by processor 2 that would let processor 5 complete the $\mathcal{S}_5(g)$ set is not in the ghost cells set of processor 5. However, processor 5 retrieves such missing local information fetching data from processor 3 in the processor 5 ghost cells which are owned by processor 3, in particular in Line 25 of Algorithm 3. The approach followed by Algorithm 3 in order to build $\mathcal{S}^{\mathrm{proc}}(g)$ is correct as proved in the following proposition.

**Proposition 2.4.3.** *Let us consider a conforming FE space $\mathcal{V}_h$ and a distributed 2:1 k-balanced forest-of-trees mesh $\mathcal{T}$ with k-ghost cells, where* $\max(1, D(\mathcal{V}_h)) \geq k$. *Then, for any $g \in \Sigma^p_{\mathrm{R}} \cap \Sigma^p_\Gamma$ such that $\mathcal{O}^{\mathrm{proc}}(g) = p$, the full $\mathcal{S}^{\mathrm{proc}}(g)$ set can be computed combining local information and a single nearest-neighbour communication.*

*Proof.* Given a processor $q \in \mathcal{S}^{\mathrm{proc}}(g)$, by definition, the VEF $F$ that owns $g$ must be in $\mathcal{F}^q_{\mathrm{L}}$ because of (a), (c), or (d) in Definition 2.3.11. Using the definition of $k$-ghost cells in Definition 2.3.10, when $F$ satisfies (a) or (c) at processor $q$, the fact that $q$ is in $\mathcal{S}^{\mathrm{proc}}(g)$ can be locally determined at $\mathcal{O}^{\mathrm{proc}}(g)$. When (d) holds for some VEF $J$ such that $F \subset \bar{J}$ (where $J$ is regular by Prop. 2.3.9), it is easy to check that $\mathcal{O}^{\mathrm{proc}}(J)$ (which cannot be $q$, as $J$ is only surrounded by ghost cells in $q$) is neighbour of $\mathcal{O}^{\mathrm{proc}}(g)$ and can determine that $q \in \mathcal{S}^{\mathrm{proc}}(g)$. Thus, $q$ can be fetched at $\mathcal{O}^{\mathrm{proc}}(g)$ with a single nearest neighbour communication. □

**Remark 2.4.4** (Implementation remark)**.** *In order to keep the presentation short, we have omitted from Algorithms 2 and 3 that DOFs in $\Sigma^{p\to q}_{\mathrm{snd}}$ and $\Sigma^{q\leftarrow p}_{\mathrm{rcv}}$, for any pair of neighbours $p$ and $q$, have to be glued together at both sides of the interface, as a final step to complete the equivalence classes corresponding to DOFs at the subdomain interfaces. We achieve this*

*is in practice as follows. Algorithm 3 generates, in situ, a global DOF identifier for all DOFs* $g \in \Sigma_{\mathrm{R}}^p \cap \Sigma_{\Gamma}^p$. *The algorithm that generates such identifiers follows the same pattern to the one that determines* $\mathcal{O}^{\mathrm{proc}}(g)$, *and thus is mathematically supported by Proposition 2.4.2 as well. In particular, if a proc-regular interface DOF is surrounded by at least one local cell, a global DOF identifier can be generated solely using processor-local information, while if not, it can be fetched on ghost cells via a nearest-neighbour exchange. Then, using such global DOF identifiers, Algorithm 2 arranges the DOFs in* $\Sigma_{\mathrm{snd}}^{p\to q}$ *and* $\Sigma_{\mathrm{rcv}}^{q\leftarrow p}$ *in increasing global DOF numbering within each set, such that they are consistently paired at both sides of the interface among p and q. The global DOF identifiers are then discarded, and not used elsewhere.*

### 2.4.5 Setting up parallel distributed fully-assembled linear systems

The vast majority of parallel linear algebra packages (e.g. `PETSc` [26], or `TRILINOS` [91]) use a data distribution model in which the global linear system is partitioned by rows, i.e. each processor *owns* a non-overlapping subset of *fully-assembled* global rows of $\boldsymbol{A}$ and $\boldsymbol{b}$, resp. Besides, the data structures are *globally addressable*, i.e. a given processor may contribute to a global matrix row (or a global vector entry) which is not necessarily owned by it. Such packages can be leveraged by means of a distributed algorithm able to generate the equivalence classes in $\Sigma_{\mathrm{R}}$ with the local-to-global index map $[\cdot]$ built such that the DOFs owned by the first processor are the ones to be numbered first, immediately followed by those of the second, and so on.[8]

The distributed algorithm presented in [27, Section 3.1] (available in `deal.II`) follows a cell-wise approach that exploits the overlapped mesh partition to directly generate such index map. Here, we instead follow a DOF-wise approach which exploits the locally generated local-to-proc-local index map $[\cdot]_p$ (see Section 2.4.2) and Remark 2.4.4 to assign a global DOF identifier to DOFs in $\Sigma_{\mathrm{L}}^p \cap \Sigma_{\mathrm{R}}^p$, for all $p$. We define $\Sigma_{\mathrm{O}}^p \doteq \{g \in \Sigma_{\mathrm{L}}^p \cap \Sigma_{\mathrm{R}}^p : \mathcal{O}^{\mathrm{proc}}(g) = p\}$, and refer to it as the set of DOFs owned by processor $p$. This algorithm encompasses four main steps: (a) each processor locally computes $|\Sigma_{\mathrm{O}}^p|$; (b) an exclusive prefix reduction (i.e. `MPI_Exscan`) of $|\Sigma_{\mathrm{O}}^p|$, for all $p$, computes a starting offset global DOF identifier on each processor; (c) each processor locally assigns a global, consecutive DOF identifier to owned DOFs starting from the global offset identifier computed in (b); (d) each processor fetches from remote neighbours the global DOF identifiers for those DOFs in $\Sigma_{\mathrm{L}}^p \cap \Sigma_{\mathrm{R}}^p$ which are not owned by it. This operation involves a DOF-wise nearest neighbour communication with $\mathcal{S}_{\mathrm{rcv}}^p$, $\Sigma_{\mathrm{rcv}}^{p\leftarrow q}$ as send side, and $\mathcal{S}_{\mathrm{snd}}^p$, $\Sigma_{\mathrm{snd}}^{p\to q}$ as receive side (i.e. owners send, non-owners receive).

[8] We note that to have consecutive global row identifiers in each processor is a constraint of `PETSc` [26], but not actually `TRILINOS` [91].

## 2.5 Numerical experiments

### 2.5.1 Experimental environment

The numerical experiments are run at the Marenostrum-IV (MN-IV) supercomputer, hosted by BSC. This petascale machine is equipped with 3,456 compute nodes interconnected together with the Intel OPA HPC network. Each node has 2x Intel Xeon Platinum 8160 multi-core CPUs (Skylake), with 24 cores each (i.e. 48 cores per node) and 96 GBytes of RAM.

With respect to the software, we used `FEMPAR` v1.0.0 [19], linked against `p4est` v2.2 [43, 99] as its forest-of-octrees manipulation engine. In its current status, up to the authors' knowledge, `t8code` [93] only provides facet-oriented variants of the main operations outlined in Section 2.2.6 (e.g. `Balance` is restricted to $d-1$-balance), and thus cannot be readily used for the implementation of generic FEs. For this reason, in this chapter we restrict to `p4est` as a practical demonstrator. Besides, we used `deal.II` v9.0.0 [28], linked against `p4est` v2.2, and `PETSc` v3.9.0 [26] for distributed-memory linear algebra data structures and solvers. These software were compiled with Intel v18.0.1 compilers using system recommended optimization flags and linked against the Intel message-passing interface (MPI) Library (v2018.1.163) for message-passing and the BLAS/LAPACK and PARDISO available on the Intel MKL library for optimised dense linear algebra kernels and sparse direct solvers, resp. All floating-point operations were performed in IEEE double precision.

### 2.5.2 Poisson problem

We evaluate the performance and strong scalability of the different stages in our framework as implemented in `FEMPAR`, and compare them against those in `deal.II` for the numerical solution of the Poisson problem. We consider a 3D cube domain $\Omega = [0,1]^3$ with homogeneous Dirichlet boundary conditions (strongly) imposed over the entire domain boundary. The problem is discretised using a global Lagrangian FE space $\mathcal{V}_h \subset H_0^1(\Omega)$. The weak formulation of this problem reads: find $u_h \in \mathcal{V}_h$ such that

$$(\boldsymbol{\nabla} u_h, \boldsymbol{\nabla} v_h) = (f, v_h), \qquad \forall v_h \in \mathcal{V}_h.$$

The right-hand-size $f(x,y,z)$ is a piece-wise function defined as:

$$f(x,y,z) = \begin{cases} 1 & z > \frac{1}{2} + \frac{1}{4}\,\sin(4\pi x)\,\sin(4\pi y) \\ -1 & z \leq \frac{1}{2} + \frac{1}{4}\,\sin(4\pi x)\,\sin(4\pi y) \end{cases}.$$

The discontinuity in $f$ leads to a solution that is non-smooth along a sinusoidal surface through the domain, so that very localised AMR is required in order to reduce the error in that area, while keeping the computational requirements reasonably low.

The experiment is designed as follows. We start with a coarse mesh that is obtained after uniformly refining the root octant of the octree up to four times. This mesh has 16 cells (hexahedra) per coordinate direction, and 4096 cells in total. Then, the Poisson problem is solved on a hierarchy of meshes where the mesh at a given step is obtained from the previous one by means of a user-level mesh handling primitive in our framework named `Refine_and_coarsen`. This primitive adapts the forest-of-trees mesh based on cell flags set by the user, while transferring data that the user might have attached to the mesh objects (i.e. cells and VEFs) to the new mesh objects generated after mesh adaptation. `Refine_and_coarsen` invokes `Adapt`, `Balance`, and `Ghost`; see Section 2.2.6. With load balancing in mind, the mesh is dynamically redistributed at each step by means of the `Redistribute` mesh handling primitive right after each call to `Refine_and_coarsen`. This primitive dynamically balances the number of cells in each processor. The data that the user might have attached to the mesh objects (i.e. cells and VEFs) is also migrated among processors. `Redistribute` invokes `Partition`, and `Ghost`; see Section 2.2.6. We note that both `Refine_and_coarsen` and `Redistribute` also have to: (a) set up the data structures providing cell neighbours across $n$-faces; (b) call Algorithm 1. The process underlying (a) is implemented by means of invocations to `Iterate` (Section 2.2.6).

For a given (fixed) number of AMR steps, we measure the elapsed time (i.e. wall clock time) spent in each of the stages in the simulation of process, *aggregated across all steps*, and evaluate at which rate they are reduced with increasing number of CPU cores (i.e. strong scalability test). The number of AMR steps is adjusted such that a "sufficiently large" problem size at the last step is obtained for the CPU core range on which we run the strong scaling test. The decision of which cells to be refined or coarsened is performed as follows. Given the solution of the linear system, each processor computes independently of each other an error indicator for cells $T \in \mathcal{T}_{\mathrm{L}}^{p}$ using the a-posteriori error estimator proposed by Kelly et al. in [105]. Then, given user-defined refinement and coarsening fractions, denoted by $\alpha_r$ and $\alpha_c$, resp., we find the thresholds $\theta_r$ and $\theta_c$ such that the *total* number of cells with error indicator larger (resp., smaller) than $\theta_r$ (resp., $\theta_c$) is (approximately) a fraction $\alpha_r$ (resp., $\alpha_c$) of the *total* number of cells. To this end, we use the iterative algorithm proposed in [27, Figure 5]. We set $\alpha_r = 0.15$ and $\alpha_c = 0.03$, so that, the number of cells is at least doubled at each AMR step (assuming that no cells can be coarsened). The actual number of cells at each mesh in the hierarchy, however, also depends on the algorithm within `p4est` that 2:1 balances the forest-of-octrees, as it may need to apply more refinement in order to keep this constraint [43].

In the sequel, we will focus on evaluating the following stages, which are labelled as:

- MESH: Includes the calls to the `Refine_and_coarsen` and `Partition` primitives.
- FE SPACE SUB-ASSEMBLY: Includes the generation of proc-local DOFs (Section

2.4.2), Algorithm 3, and the computation of hanging DOFs constraints for all proc-local DOFs $g \in \Sigma_{\mathrm{H}}^p \cap \Sigma_{\mathrm{L}}^p$.

- FE SPACE FULL-ASSEMBLY: Includes FE SPACE SUB-ASSEMBLY and the algorithm outlined in Section 2.4.5.

- ASSEMBLY SUB-ASSEMBLY: Includes the computation of the cell matrix and force for all local cells, the treatment of hanging DOF constraints according to [170], and the incremental assembly and final compression of the local-to-subdomain coefficient matrix data structure; see [19, Section 11.1] for more details on this last step.

- ASSEMBLY FULL-ASSEMBLY: Includes ASSEMBLY SUB-ASSEMBLY and the setup of the fully-assembled linear system (using the data structures in `PETSc`). This latter step is split into three substeps: (A) the sparsity pattern of locally owned rows is completed by means of a nearest neighbour exchange; (B) the full sparsity pattern of locally owned rows computed in (A) is used to set up the fully-assembled matrix data structure; (C) the subassembled non-zero matrix entries and vector entries are injected, on a row-by-row fashion, into the global, fully-assembled data structures. Then, a final inter-processor assembly stage involves the transfer of partial contributions to local entries but actually stored in (i.e. owned by) remote processors. This communication phase is resolved by the external linear algebra package.

- ERROR ESTIMATOR: Includes the computation of the error estimators for all local cells, the computation of $\theta_r$ and $\theta_c$ thresholds, and to flag the cells for the `Refine_and_coarsen` primitive.

- TOTAL SUB-ASSEMBLY: The aggregation of MESH, FE SPACE SUB-ASSEMBLY, ASSEMBLY SUB-ASSEMBLY, and ERROR ESTIMATOR.

- TOTAL FULL-ASSEMBLY: The aggregation of MESH, FE SPACE FULL-ASSEMBLY, ASSEMBLY FULL-ASSEMBLY, and ERROR ESTIMATOR.

In this section, we also compare the performance and scalability of the algorithms and data structures within `FEMPAR` with those in `deal.II`. In particular, we adapted the `deal.II step-40` tutorial programme, [9] in order to solve problem (2.3), and grouped the different steps in this programme into stages according to the classification above. There are, though, still several significant worth-noting differences among the following stages:

1. The MESH stage in `deal.II` does not actually call Algorithm 1, but a fundamentally different algorithm; see Remark 2.3.15.

[9] Documentation available at `https://www.dealii.org/9.0.0/doxygen/deal.II/step_40.html`.

2. The FE SPACE FULL-ASSEMBLY stage in `deal.II` does not use the steps in FE SPACE SUB-ASSEMBLY, but the algorithm in [27, Section 3.1]. Besides, it also sets up the so-called *locally owned* and *locally relevant* index sets of global DOF identifiers. These data structures are required to manipulate globally addressable, fully-assembled linear algebra matrices and vectors in `deal.II`. We refer to [27, Section 3] for additional details.

3. The ASSEMBLY FULL-ASSEMBLY stage in `deal.II`, as its counterpart in `FEMPAR`, comprises three different steps, with step (B) in `deal.II` being equivalent to its `FEMPAR` counterpart. Step (A) first determines, *by means of symbolic assembly*, the location of non-zero elements in the local subassembled matrix and then, a communication stage, equivalent to the one in `FEMPAR` above, allows all processors to complete the non-zero pattern of locally owned rows; in step (C) the contribution of local cells are computed and *directly* assembled into the global linear system data structures, including the transfer of entries locally computed but stored in remote processors. We refer to [27, Section 4] for additional details.

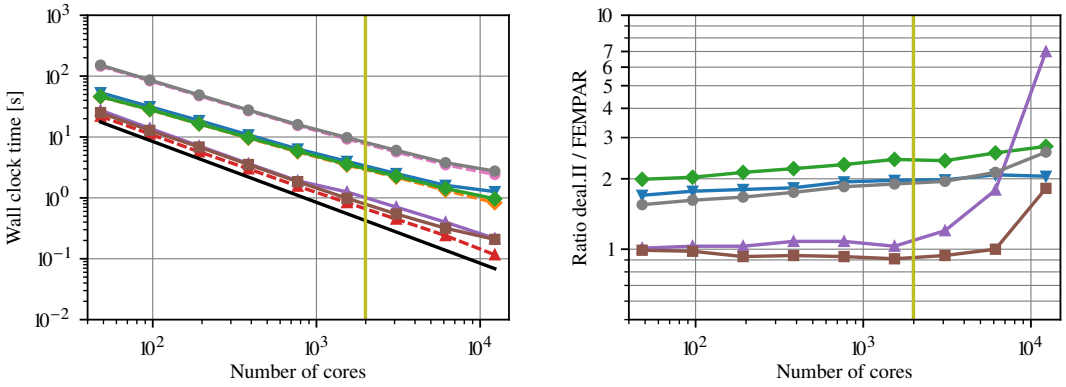
We do not report the computational time spent in the linear system solve stage, but focus on the algorithms presented in this chapter instead.

We note that, for the problem at hand, $D(\mathcal{V}_h) = 0$. Thus, due to Proposition 2.4.1, and 2.4.2, we may use $k = 1$ and $s = 0$. However, we use $k = 0$ and $s = 0$ to have a fair comparison against `deal.II`.[10] The experiments with $k = 1$ and $s = 0$ reveal a *negligible* reduction in the number of adaptive mesh cells in the case of the experiments reported in Figure 2.5 (less than 0.02% for the mesh in the last AMR step), but *a much more noticeable one* for the ones in Figure 2.6 (up to 12.1% for the mesh in the last AMR step).

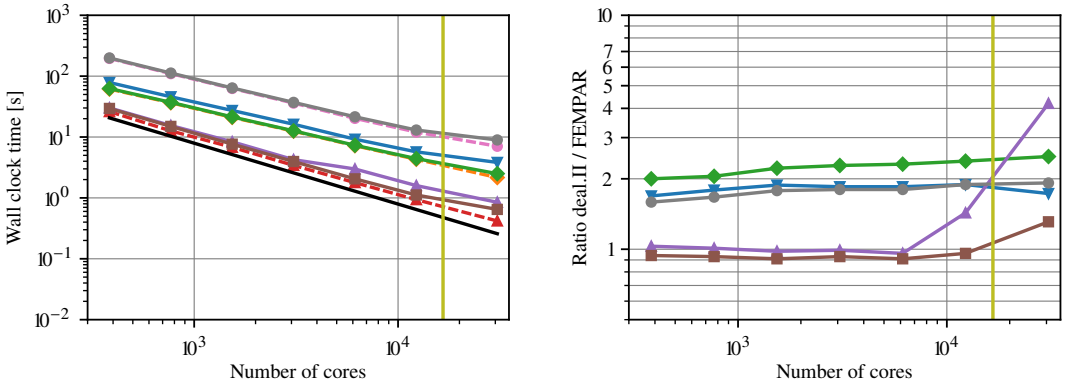
Figure 2.5 (left) shows the strong scalability of the algorithms and data structures in `FEMPAR` for problem (2.3) discretised with trilinear ($\mathcal{Q}_1(T)$) Lagrangian FEs. On the other hand, Figure 2.5 (right) shows the ratio, $R$, among the computational times spent in the stages in `deal.II` and their counterparts in `FEMPAR`. If $R$ is larger than one, then the corresponding stage in `FEMPAR` is faster than the one in `deal.II` by a factor $R$. For readability purposes, Figure 2.5 (left) also provides the ideal strong scaling slope (solid black line). The more parallel a given strong scaling curve is to the ideal slope the more strongly scalable the corresponding stage is. Two different global problem sizes suitable for the $[48, 12228]$ and $[384, 30672]$ CPU cores range were tested. In particular, the results in Figure 2.5(a) correspond to a problem in which we perform 13 AMR steps, resulting into a 49.8 MDOFs problem size at the last step, while in Figure 2.5(b), we

[10] `deal.II` always instructs `p4est` to build 2:1 0-balanced meshes, i.e. $k$ is not a parameter the user can play around with. See line 2873 of `https://www.dealii.org/9.0.1/doxygen/deal.II/distributed_2tria_8cc_source.html` for more details.

report the ones for 16 AMR steps, and a 415.5 MDOFs problem size.[11] *These DOF counts, and the ones in the rest of the chapter, include both regular and hanging DOFs.* The average load per core at the last step ranges, in Figure 2.5(a), from 1.04M DOFs/core to 4.06K DOFs/core, and, in Figure 2.5(b), from 1.08M DOFs/core to 13.53K DOFs/core. For readability purposes, we also plot a vertical line corresponding to the 25K DOFs/core regime. Beyond that point, one can barely expect AMG linear solvers to scale.




(a) 13 AMR steps, 49.8 MDOFs, up to 12.2K cores.




(b) 16 AMR steps, 415.5 MDOFs, up to 30.7K cores.

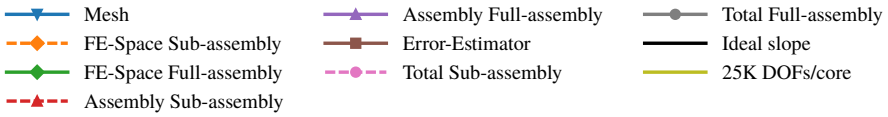



Figure 2.5: Strong scaling test results on MN-IV for `FEMPAR` (left) and ratio `deal.II` versus `FEMPAR` (right). Poisson problem with $\mathcal{Q}_1(T)$ Lagrangian FEs.

Figure 2.5(a) and 2.5(b) (left) reveal high strong scalability for the ASSEMBLY SUB-ASSEMBLY and ERROR ESTIMATOR stages in the full CPU cores range tested. For example, the parallel efficiency for the ASSEMBLY SUB-ASSEMBLY stage only decays to 74% and 78% with the largest number of cores tested in Figure 2.5(a) and Figure 2.5(b),

[11] We stress that this is by no means a parallel scaling limit of the algorithms proposed, but the largest number of cores we can exploit on MN-IV provided the access constraints that we have to this supercomputer. We expect the algorithms proposed to be able to scale up to hundreds of thousands of cores in the solution of hundred of billions DOFs problem sizes.

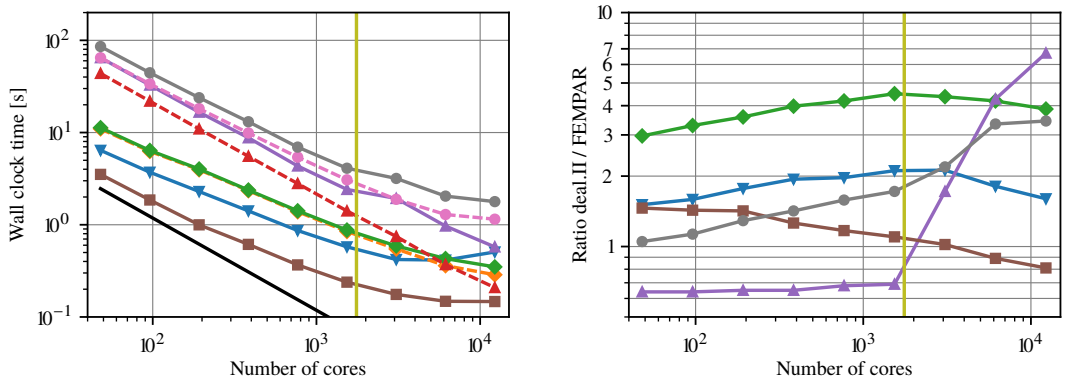
resp. An intermediate degree of strong scalability is observed for the ASSEMBLY FULL-ASSEMBLY stage, that decays to 50% and 44% in Figure 2.5(a) and Figure 2.5(b), resp., due to the communication overhead involved in stages (A) and (C) outlined above. Also worth noting is the extra overhead of ASSEMBLY FULL-ASSEMBLY compared to ASSEMBLY SUB-ASSEMBLY, which ranges from 25% for 48 cores to 85% for 12.2K cores in Figure 2.5(a), and from 15% for 384 to 100% for 30.7K cores in Figure 2.5(a). This is caused by the extra stages (A)-(C) involved in the setup of fully-assembled linear systems. While less strong scalability is observed for the MESH, FE SPACE SUB-ASSEMBLY, and FE SPACE FULL-ASSEMBLY stages, the computational time reduction with the number of cores is still very significant for these stages, with the parallel efficiency decaying to 16%, 21%, and 19%, in Figure 2.5(a), and 26%, 36%, and 23%, resp., in Figure 2.5(b). This is not surprising as these three stages, and specially MESH, involves significantly more communication volume (relative to local work) than, e.g. ASSEMBLY SUB-ASSEMBLY.

In Figure 2.5(a) and 2.5(b) (right), we can observe that the algorithms and data structures in `FEMPAR` are either competitive (ERROR ESTIMATOR and ASSEMBLY FULL-ASSEMBLY stages) or *up to 2-3 times faster* than the ones in `deal.II` (MESH, FE SPACE FULL-ASSEMBLY). The ratio in the MESH stage is, e.g. as large as 2.08x for the largest number of cores in Figure 2.5(a), and increases at a moderate pace from 1.70x to 2.08x from 48 to 12.2K cores. These results evidence what it seems a more efficient coupling of the data structures in `FEMPAR` and those of `p4est`, although we could not completely discard whether the performance improvement is caused by differences among low-level technical details in the HPC implementation of both codes and the underlying hardware. Even larger ratios for the FE SPACE FULL-ASSEMBLY stage are observed in Figure 2.5(a) and 2.5(b), as large as 2.75x, and 2.49x, resp., for the largest number of cores evaluated. The extra communication volume and overhead associated to handling a global numbering of DOFs suitable for fully-assembled operators in `deal.II` (e.g. searches on index sets of global identifiers, 64-bit DOF identifiers) is the most contributing factor to this trend.

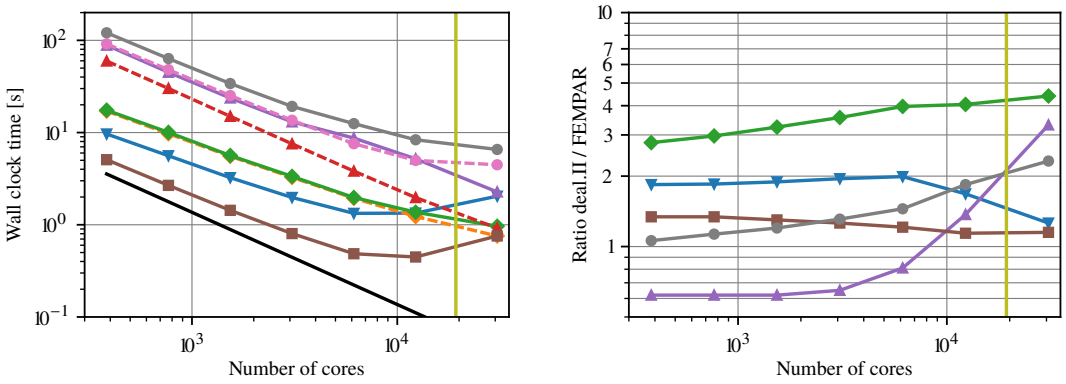
If we focus on the ratios for the ASSEMBLY FULL-ASSEMBLY stage in Figure 2.5(a) and 2.5(b), one can observe a sudden parallel scaling drop of `deal.II` compared to `FEMPAR`. We confirmed that the source of this difference is concentrated in steps (A) and (C) of ASSEMBLY FULL-ASSEMBLY in `deal.II`. We could not, however, determine the actual cause of the difference. In any case, we consider this issue to be of relative importance provided that this effect arises at a regime in which one has a relatively low load per core. Indeed, this was actually not observed in [27] (at least as evidently). This might be caused by the fact that experiments in [27] considered strong scaling results up to larger loads per core to those considered here.

Overall, *the balance achieved among all these stages ends up with* TOTAL FULL-ASSEMBLY *in* `FEMPAR` *being up to 2.60x faster in Figure 2.5(a), and 1.92x faster in Figure 2.5(b) compared to* TOTAL FULL-ASSEMBLY *in* `deal.II` *(for the largest number of cores tested).*

In order to showcase the suitability of the proposed $h$-adaptive pipeline for higher than linear-order FEs, Figure 2.6 shows its strong scalability for problem (2.3) discretised with triquadratic ($\mathcal{Q}_2(T)$) Lagrangian FEs. The average load per core at the last step ranges, in Figure 2.6(a), from 0.92M DOFs/core to 3.57K DOFs/core, and, in Figure 2.6(b), from 1.25M DOFs/core to 15.72K DOFs/core. These loads per core are similar to those in Figure 2.5. However, for a given number of cores, the load per core in terms of number of mesh cells at the last step, is smaller in Figure 2.6, compared to the one in Figure 2.5. The results in Figure 2.6(a) correspond to a problem in which 9 AMR steps are performed (4 steps less than with trilinear FEs), resulting into a 43.9 M DOFs problem size at the last step, while in Figure 2.6(b), we have 12 AMR steps (3 steps less), and a 482.2 MDOFs problem size.




(a) 9 AMR steps, $\mathcal{Q}_2(T)$ FEs, 43.9 MDOFs, up to 12.2K cores.




(b) 12 AMR steps, $\mathcal{Q}_2(T)$ FEs, 482.2 MDOFs, up to 30.7K cores.

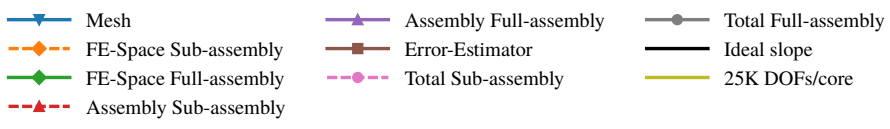



Figure 2.6: Strong scaling test results on MN-IV for `FEMPAR` (left) and ratio `deal.II` versus `FEMPAR` (right). Poisson problem with $\mathcal{Q}_2(T)$ Lagrangian FEs.

All stages in Figure 2.6 are less strongly scalable than their counterparts in Figure 2.5, except for the ASSEMBLY SUB-ASSEMBLY stage, which is as scalable as in Figure 2.5, and, by far, the most strongly scalable stage among those in Figure 2.6. In

any case, the computational time reduction with the number of cores is still significant for the rest of stages, except for the MESH and ERROR ESTIMATOR stages and the two largest core counts, for which any benefit of parallelism is totally absorbed by parallel overheads. Also worth noting in Figure 2.6 is that ASSEMBLY SUB-ASSEMBLY and ASSEMBLY FULL-ASSEMBLY concentrate a higher percentage of TOTAL SUB-ASSEMBLY and TOTAL FULL-ASSEMBLY, resp. (compared to Figure 2.5). This hides the higher drop of parallel efficiency that is observed for the MESH and ERROR ESTIMATOR stages.

If we compare the performance and scalability of `FEMPAR` and `deal.II` stages in Figure 2.6(a) and 2.6(b) (right), rather different conclusions can be extracted depending on the stage. The MESH stage in `FEMPAR` is faster than the one in `deal.II` for the whole CPU core range evaluated. However, while it scales better within a first range of CPU cores (e.g. in the [48, 6.1K] range in Figure 2.6(a)), a drop of the ratio `deal.II` versus `FEMPAR` is observed for this stage in Figure 2.6(a) and Figure 2.6(b), with the drop in the latter being more significant than the one of the former. On the contrary, ASSEMBLY FULL-ASSEMBLY stage in `deal.II` is 1.6x faster than its counterpart in `FEMPAR` within a first range of CPU cores in which ASSEMBLY FULL-ASSEMBLY is dominated by local work; this reveals that more effort should be invested in the optimization of stages (A)-(C) for $\mathcal{Q}_2(T)$ FEs, for which, at present, `FEMPAR` only provides a naive, reference implementation. However, provided the number of CPU cores is sufficiently large, the parallel efficiency of the ASSEMBLY FULL-ASSEMBLY stage in `deal.II` starts dropping significantly compared to the one in `FEMPAR`, up to an extent that large `deal.II` versus `FEMPAR` ratios are observed.

The FE SPACE FULL-ASSEMBLY stage in `FEMPAR` is faster and scales better than the one `deal.II` for the whole CPU core range evaluated. Indeed, larger ratios for the FE SPACE FULL-ASSEMBLY stage (and higher ratio increase with the number processors) are observed in Figure 2.6 compared to those in Figure 2.5. For example, the ratio in the FE SPACE FULL-ASSEMBLY stage is as large as 4.4x for the largest number of cores in Figure 2.6(b), and increases from 2.78x to 4.4x from 384 to 30.7K cores due to the extra communication volume and overhead associated to handling a global numbering of DOFs suitable for fully-assembled operators in `deal.II`. The ERROR ESTIMATOR stage is faster in `FEMPAR` within the whole CPU cores range in Figure 2.6(b), and the same applies to Figure 2.6(a) except for the two largest number of cores. However, the drop in parallel efficiency observed in the left part of Figure 2.5(b) and Figure 2.6(b) is accompanied with a drop of the `FEMPAR` to `deal.II` ratio, moderate in the right part of Figure 2.6(b) (in particular, a 14% ratio drop), but much more apparent in the right part of Figure 2.5(b) (a 42% ratio drop). After thorough inspection with the help of performance analysis tools, we could confirm so far that the drop is not caused by the algorithm which determines the $\theta_r$ and $\theta_c$ thresholds, but in the actual numerical computations performed locally within each processor during this stage.

Overall, *the balance achieved among all these stages ends up with* TOTAL FULL-ASSEMBLY *in* `FEMPAR` *being up to 3.4x faster in Figure 2.5(a) and 2.3x faster in Figure 2.5(b) compared to*

TOTAL FULL-ASSEMBLY *in* `deal.II` *(for the largest number of cores tested).*

### 2.5.3 Maxwell problem

In this section we showcase the application of our framework to the numerical solution of the Maxwell equations. We consider a 3D L-shaped domain $\Omega = [-1,1]^3 \setminus [-1,0]^3$ with non-homogeneous Dirichlet boundary conditions (strongly) imposed over the entire domain boundary. The problem is discretised using a global space $\mathcal{V}_h$ that conforms to $H_0(\mathbf{curl},\Omega) \doteq \{\mathbf{u} \in H(\mathbf{curl},\Omega) : \mathbf{n} \times \mathbf{u} = \mathbf{0} \text{ on } \partial\Omega_D\}$. Such $\mathcal{V}_h$ is built from local Nédélec (a.k.a. edge) FEs of first kind. (We refer to [148] for a comprehensive coverage on the general software implementation of this sort of FEs within `FEMPAR`.) The problem reads: find $\boldsymbol{u}_h^0 \in \mathcal{V}_h$ such that

$$(\boldsymbol{\nabla} \times \boldsymbol{u}_h^0, \boldsymbol{\nabla} \times \boldsymbol{v}_h) + (\boldsymbol{u}_h^0, \boldsymbol{v}_h) = (\boldsymbol{f}, \boldsymbol{v}_h) - (\boldsymbol{\nabla} \times E\boldsymbol{u}_{\mathrm{D}}, \boldsymbol{\nabla} \times \boldsymbol{v}_h) + (E\boldsymbol{u}_{\mathrm{D}}, \boldsymbol{v}_h), \qquad \forall \boldsymbol{v}_h \in \mathcal{V}_h,$$

where $\boldsymbol{u}_{\mathrm{D}}$ is the Dirichlet data to be imposed, and $E\boldsymbol{u}_{\mathrm{D}}$ an arbitrary extension, i.e. $E\boldsymbol{u}_{\mathrm{D}} = \boldsymbol{u}_{\mathrm{D}}$ on $\partial\Omega_{\mathrm{D}}$. The magnetic solution field is given by $\boldsymbol{u}_h \doteq \boldsymbol{u}_h^0 + E\boldsymbol{u}_{\mathrm{D}}$. The input data $\boldsymbol{u}_{\mathrm{D}}$ and $\boldsymbol{f}$ are set up such that the (analytical) solution of this problem becomes $\boldsymbol{u} = \boldsymbol{\nabla}\left(r^{\frac{2}{3}} \sin\left(\frac{2t}{3}\right)\right)$, with $t = \arccos\left(\frac{xyz}{r}\right)$, $(x,y,z)$ the Cartesian coordinates of any point within $\Omega$, and $r$ is the corresponding radius in 3D polar coordinates. This analytical solution has a singular behaviour near the origin and $\boldsymbol{u} \notin \boldsymbol{H}^1(\Omega)$. We note that, for this problem, a forest-of-octrees with three adaptive octrees is sufficient in order to geometrically discretise $\Omega$. Besides, as it has known analytical solution, we do not actually use a-posteriori error estimators, but the true error among the analytical and FE solution measured in the $L_2$-norm instead. For this sort of problem, up to the authors' knowledge, there is no `deal.II` step tutorial available at the public domain, so that we could not drive the comparison of `FEMPAR` against `deal.II`. In any case, `deal.II` supports edge *h*-adaptive FEs as well.

Figure 2.7 (left) shows the strong scalability of `FEMPAR` for problem (2.4) discretised with first-order Edge FEs. A single global problem size suitable for the $[48, 12228]$ CPU cores range was tested, in particular, the one corresponding to 11 AMR steps, which already leads to a problem with 115.6 MDOFs at the last step. The averaged load per core at the last step ranges from 2.41M DOFs/core (48 processors) to 9.40K DOFs/core (12.2K processors). The computational times in Figure 2.7 (left) were obtained with $k = 0$ and $s = 0$. However, we note that, as $D(\mathcal{V}_h) = 1$, we can use, due to Proposition 2.4.1 and 2.4.2, $k = 1$ and $s = 1$. A reduction of 4.5% in the number of cells of the mesh in the last AMR step was observed. Besides, Figure 2.7 (right) evaluates the ratio of the computing times obtained with $k = 0$ and $s = 0$ versus the ones obtained with $k = 1$ and $s = 1$.

Figure 2.7 (left) confirms, as with the Poisson problem discretised with Lagrangian FEs, remarkable strong scalability for all stages, and significant computational time reductions for TOTAL SUB-ASSEMBLY in the whole range of CPU cores analysed, despite

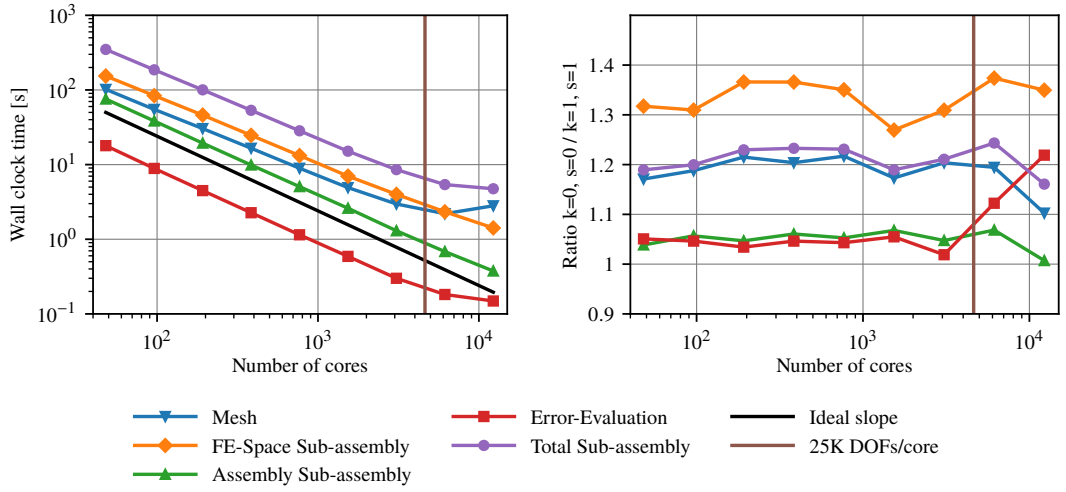



Figure 2.7: FEMPAR strong scaling test results on MN-IV (left) and ratio among computing times with 0-balance and 0-ghost cells versus 1-balance and 1-ghost cells (right). Maxwell problem with first order ($\mathcal{ND}_1(T)$) Edge FEs.

the more general structure and complexity of implementation underlying $H$-curl conforming FE spaces. On the other hand, the computational savings in Figure 2.7 (right), which are enabled by Proposition 2.4.1 and 2.4.2, are such that TOTAL SUB-ASSEMBLY becomes approximately 1.2 times faster with $k = 1$ and $s = 1$.

## 2.6 Conclusions

In this chapter we have presented the building blocks, i.e. data structures and algorithms, of a highly scalable parallel *generic FE* simulation framework that supports AMR via forest-of-trees endowed with SFCs. In particular, we have mathematically proven the correctness of the algorithms for scalable mesh handling, construction of generic global conforming FE spaces on top of non-conforming meshes using hanging DOF constraints, and the parallel subdomain-wise sub-assembly and full-assembly of the discrete linear system of algebraic equations. Central to the framework is a generic FE-suitable adaptive mesh representation that is grounded on concepts that are not tailored to a particular cell topology, SFC, or number of space dimensions. All that the framework requires from the forest-of-trees layer meshing engine to be able to build such mesh data structure is a description of the local neighbourhood across the cell boundaries in the adapted mesh. Along the way, we have (a) mathematically justified which are the crucial benefits (i.e. ease of implementation and parallelisation, high scalability) that one enjoys from enforcing the 2:1 $k$-balance constraint, (b) identified what a conforming FE formulation must fulfil such that it can be implemented in parallel given the constraint that each processor only has access to the off-processor cells in the $s$-ghost cell set, and (c) and determined the largest possible value of $k$ and $s$ that lets one still implement such FE formulation. The software implementation of these algorithms

is available at FEMPAR [19], an open source OO Fortran200X scientific software package for the HPC simulation of complex multiphysics problems governed by PDEs at large scales.

Besides, we have carried out a comprehensive strong scaling study of FEMPAR on a petascale supercomputer when applied to both Poisson and Maxwell PDE problems, for problems in which the usage of AMR is highly beneficial for computational efficiency. The study reveals remarkable scalability of the framework up to several tens of thousands of CPU cores in the solution of problems with several hundreds of millions of degrees of freedom. Besides, we have also compared performance and strong scalability of FEMPAR against the deal.II FE library. This comparison reveals that the algorithms proposed here are at least as fast deal.II, and, in many cases, *2-3 times faster*.

The usage of parallel scalable AMR is still rather limited by most scientific computing practitioners and researchers, mostly because of the underlying complexity behind developing such tool. We expect this chapter to be successful in convincing the reader that the design and development of the algorithms and data structures behind this tool is affordable and of reasonable complexity. With this in mind, while still using a high-level approach when presenting the algorithms, we introduced sufficient discussion and associated details to be helpful in this regard. Besides, wherever applicable, and in contrast to the other related work in the literature, the algorithms are supplemented with mathematical propositions and proofs that support their correctness.

# Chapter 3

# The aggregated unfitted finite element method on parallel tree-based meshes

*The contents of this chapter correspond to research the publication*

[15] S. BADIA, A. F. MARTÍN, EN AND F. VERDUGO, *The aggregated unfitted finite element method on parallel tree-based adaptive meshes*, Submitted.

In this chapter, we present an adaptive unfitted finite element scheme that combines the aggregated finite element method with parallel adaptive mesh refinement. We introduce a novel scalable distributed-memory implementation of the resulting scheme on locally-adapted Cartesian forest-of-trees meshes. We propose a two-step algorithm to construct the finite element space at hand that carefully mixes aggregation constraints of problematic degrees of freedom, which get rid of the small cut cell problem, and standard hanging degree of freedom constraints, which ensure trace continuity on non-conforming meshes. Following this approach, we derive a finite element space that can be expressed as the original one plus well-defined linear constraints. Moreover, it requires minimum parallelization effort, using standard functionality available in existing large-scale finite element codes. Numerical experiments demonstrate its optimal mesh adaptation capability, robustness to cut location and parallel efficiency, on classical Poisson $hp$-adaptivity benchmarks. Our work opens the path to functional and geometrical error-driven dynamic mesh adaptation with the aggregated finite element method in large-scale realistic scenarios. Likewise, it can offer guidance for bridging other scalable unfitted methods and parallel adaptive mesh refinement.

## 3.1 Introduction

AMR using adaptive tree-based meshes is attracting growing interest in large-scale simulations of physical problems modelled with PDEs. Research over the past few years has demonstrated that tree-based AMR enables efficient data storage and mesh traversal, fast computation of mesh hierarchy and cell adjacency and extremely scalable partitioning and dynamic load balancing. Although several cell topologies have

been studied [41, 93], attention has centred around quadrilateral (2D) or hexahedral (3D) adaptive meshes endowed with standard isotropic 1:4 (2D) and 1:8 (3D) refinement rules. They form tree structures that are commonly known as quadtrees [80] or octrees [133] or *forest-of-quadtrees* or *-octrees*, when the former are patched together. There is ample literature concerning single-octree meshes [178, 179, 184] and extensions to forest-of-octrees [43, 99]. State-of-the art in these techniques is available at the open source parallel forest-of-octrees meshing engine `p4est` [43].

In the context of parallel adaptive FE solvers, forest-of-trees have been an essential component in many large-scale application problems [40, 143, 148, 162]. As they provide multi-resolution by local mesh adaptation, they are convenient, among others, in the following three scenarios: (1) a *priori* mesh refinement, when the boundary value problem (BVP) exhibits local features that must be captured with high resolution, but are known in advance, see e.g. [143, 148]; (2) a *posteriori* mesh refinement, driven by error estimators [6], for solutions of BVPs whose local features are not known or spatially evolve over time [162]; and (3) to control geometric approximation errors of static or moving boundaries and interfaces, in combination with unfitted FE methods [39].

In spite of their scalable multi-resolution capability, practical integration of forest-of-trees in large-scale FE codes is hindered by the fact that, in general, they are non-conforming meshes. In particular, they contain the widely known *hanging* VEFs, occurring at the interface of neighbouring cells with different refinement levels. Mesh non-conformity increases implementation complexity of FE methods, especially, when they are conforming. In this case, DOFs lying on *hanging* VEFs cannot have an arbitrary value, they must be constrained to guarantee trace continuity across cell interfaces. Set up (during FE space construction) and application (during FE assembly) of hanging DOF constraints have been thoroughly studied [157, 170]. Several large-scale FE software packages also provide state-of-the-art treatment of hanging DOFs [16, 27]. They accommodate to standard practice of constraining the processor-local portion of the mesh to the cells the processor owns and a single layer of adjacent off-processor cells, the so-called ghost cells; it is well-established that hanging DOF constraints do not expand beyond a single layer of ghost cells, see e.g. Chapter 2 for comprehensive and rigorous demonstration.

While research is mature on generic parallel tree-based adaptive FE methods, enabling applications in arbitrarily complex geometries has been vastly overlooked. Usage of *body-fitted* meshes (i.e. those whose faces conform to the domain boundary) is not a choice in large-scale parallel computations, due to the bottleneck in generating and partitioning large unstructured meshes. On the other hand, unfitted (also known as embedded or immersed) FE methods blend exceptionally well with adaptive tree-based meshes. However, to the authors' best knowledge, this line of research has been barely explored. The main advantage of unfitted methods is that, instead of requiring body-fitted meshes, they embed the domain of interest in a geometrically simple

background grid (usually a uniform or an adaptive Cartesian grid), which can be generated much more efficiently. Unfortunately, unfitted FE methods also suffer from well-known drawbacks, above all, the so-called *small cut cell problem*. The intersection of a background cell with the physical domain can be arbitrarily small, with unbounded aspect ratios. This leads to severely ill-conditioned systems of algebraic linear equations, if no specific strategy alleviates this issue [62].

Many different unfitted methodologies have emerged that cope with the small cut cell problem (see, e.g. the cutFEM method [35], the Finite Cell Method [167], the AgFEM method [24], and some variants of the XFEM method [176]). They have also been useful for many multi-phase and multi-physics applications with moving interfaces (e.g. fracture mechanics [177], fluid–structure interaction [132], free surface flows [166]), in applications with varying domain topologies (e.g. shape or topology optimization [36], or in applications where the geometry is not described by CAD data (e.g. medical simulations based on CT-scan images [145]). However, fewer works have addressed scalable parallel unfitted methods, which are essential for realistic large-scale applications. Notable exceptions are the works in [23, 100], that design tailored preconditioners for unfitted methods. Recent parallelization strategies [187] have taken a different path, by considering enhanced FE formulations that lead to well-conditioned system matrices, regardless of cut location. As a result, they are amenable to resolution with state-of-the-art large-scale iterative linear solvers such as AMG, [164, 186], for which there are highly-scalable parallel implementations in renowned scientific computing packages such as `TRILINOS` [91] or `PETSc` [25]. This approach yields superior scalability, e.g. in [187], a distributed-memory implementation of the aggregated FEM, referred to as AgFEM, scales up to 16K cores and up to nearly 300M DOFs, on the Poisson equation in complex 3D domains, discretised with uniform meshes.

This chapter aims to fill the gap between parallel adaptive tree-based meshing and robust and scalable unfitted FE techniques. We limit the scope of our chapter to AgFEM [24], although other enhanced unfitted formulations, such as the CutFEM method [35], could also be considered. AgFEM is based on aggregating cells on the boundary to remove basis functions associated with badly cut cells and, thus, eliminate ill-conditioning issues. The formulation enjoys good numerical properties, such as stability, condition number bounds, optimal convergence, and continuity with respect to data; detailed mathematical analysis of the method is included in [24] for elliptic problems and in [20] for the Stokes equation. Conversely, cell aggregation locally increases the characteristic size of the resulting aggregated mesh, which has an impact on the constant (not order) in the convergence of the method, even though such constant has experimentally been observed to be similar to the one of the non-aggregated FEM [24]. In this chapter, we demonstrate that AgFEM is also amenable to parallel tree-based meshes and optimal error-driven $h$-adaptivity in practical large-scale FE applications. We refer to the resulting method as $h$-AgFEM. Furthermore, since $h$-AgFEM is capable of adding mesh resolution wherever it is needed, it is not hindered by the local

accuracy issue mentioned above.

The outline of this chapter is as follows. We detail first, in Section 3.2, a possible way to construct conforming AgFE spaces on top of non-conforming (adaptive) meshes. The main challenge is to combine the linear constraints arising from both hanging and problematic DOFs. We propose a two-tier approach that generates first the hanging DOF constraints and then modifies them with the AgFEM constraints. We show that this technique yields unified linear constraints that have no circular dependencies. Furthermore, distributed-memory extension of the method can be implemented using common functionality of large-scale FE software packages. In our case, we have implemented the method in the large-scale FE software package `FEMPAR` [19], which exploits the highly-scalable forest-of-tree mesh engine `p4est`. We consider the Poisson equation as model problem in Section 3.3 and prove well-posedness of the associated agFE method in Section 3.4. In particular, we see that AgFE spaces on non-conforming meshes retain the good numerical properties ensured on uniform meshes. In the numerical tests of Section 3.5, we consider the model problem on several complex geometries and *hp*-FEM standard benchmarks. We demonstrate similar accuracy and optimal convergence as with standard body-fitted *h*-FEM and consistent robustness and scalability, using *out-of-the-box* AMG solvers from the `PETSc` project. We draw the main conclusions of our chapter in Section 3.6.

## 3.2 Aggregated FE spaces on non-conforming adaptive meshes

Our goal is to define conforming, continuous Galerkin (CG), AgFE spaces on top of non-conforming adaptive meshes. In this section, we introduce notation and concepts necessary to construct such spaces. We start with a typical level-set immersed boundary setup on a non-conforming mesh in Section 3.2.1; for scalability reasons, we restrict ourselves to the particular case of (non-conforming) forest-of-trees meshes. We continue with the description of the cell aggregation scheme in Section 3.2.2, which is the cornerstone of AgFEM. As stated in Section 3.1, our two-level strategy to construct AgFE spaces is (1) generation of DOF constraints enforcing conformity on hanging VEFs, followed by (2) generation of DOF aggregation constraints, judiciously combined with the previous ones. To mirror our approach in this text, we define first standard conforming Lagrangian FE spaces in Section 3.2.3, then we lay out aggregated counterparts in Section 3.2.4. At first, we look at the sequential version of these spaces; distributed-memory extension is covered in Section 3.2.5.

### 3.2.1 Embedded boundary setup

Let $\Omega \subset \mathbb{R}^d$ be an open bounded polygonal domain, with $d \in \{2,3\}$ the number of spatial dimensions, in which our PDE problem is posed. As usual, in the context of embedded boundary methods, let $\Omega^{\text{art}}$ be an *artificial* or *background* domain with a simple shape that includes the *physical* one, i.e. $\Omega \subset \Omega^{\text{art}}$ (see Figure 3.1(a)). We

assume that $\Omega^{\text{art}}$ can be easily meshed using, e.g. Cartesian grids or unstructured $d$-simplexes. Let $\mathcal{T}_h$ represent a partition of $\Omega^{\text{art}}$ into cells, with $h_T$ the characteristic size of a cell $T \in \mathcal{T}_h$ and $h \doteq \max_{T\in\mathcal{T}_h} h_T$. Any $T \in \mathcal{T}_h$ is the image of a differentiable homeomorphism $\Phi_T$ over a set of admissible open reference $d$-polytopes [19], such as $d$-simplexes or $d$-cubes. Let $\mathcal{F}_T$ denote the *disjoint* $d-1$-skeleton of $T \in \mathcal{T}_h$, e.g. $\mathcal{F}_T$ is composed of vertices, edges and faces for $d = 3$. Hereafter, we abuse terminology and refer to $\mathcal{F}_T$ as the set of VEFs of $T \in \mathcal{T}_h$. We assume that $\mathcal{T}_h$ is *non-conforming*. In particular, we allow that

**Assumption 3.2.1.** *For any two cells $T, T' \in \mathcal{T}_h$, satisfying $\overline{T} \cap \overline{T'} \neq \emptyset$, there exists $f \in \mathcal{F}_T$ and $f' \in \mathcal{F}_{T'}$ such that: (1) $\overline{f} = \overline{f'} = \overline{T} \cap \overline{T'}$; or (2) $\overline{f} = \overline{T} \cap \overline{T'}$ and $f \subsetneq f'$, or vice versa.*

In other words, any pair of intersecting VEFs are either identical or one is a proper subset of the other. We notice that meshes satisfying (1) everywhere are *conforming*. On the other hand, a *hanging* VEF is any VEF $f$ satisfying $\overline{f} = \overline{T} \cap \overline{T'}$ in (2), while $f'$ is referred to as the *owner* VEF of $f$, see definition in Proposition 2.3.7. Typical examples of hanging VEFs in, e.g. 2D, are cell vertices that lie in the middle of an edge of a coarser cell, see Figure 3.2.

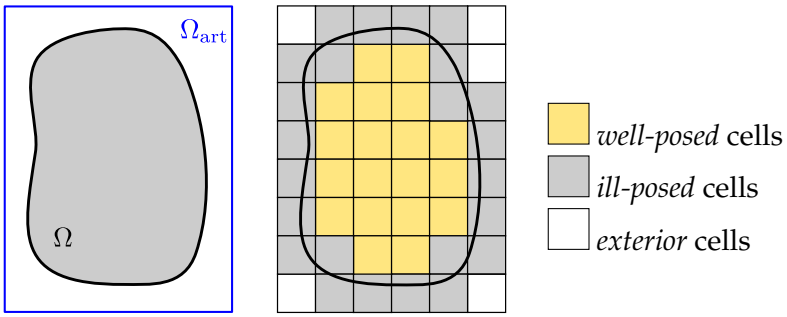



Figure 3.1: Embedded boundary setup. We introduce a *simple* artificial domain $\Omega^{\text{art}}$, that includes the physical one $\Omega$, and a partition of the background mesh $\mathcal{T}_h$ into *well-posed*, *ill-posed* and *exterior*. Note that we use $\eta_0 = 1$, i.e. well-posed iff interior and ill-posed iff cut.

As outlined in Section 3.1, we restrict ourselves to the family of (non-conforming) *forest-of-trees* meshes. This kind of meshes are derived from recursive application of standard isotropic 1:$2^d$ refinement rules on a (possibly unstructured) initial coarse mesh. By construction, they satisfy Assumption 3.2.1. We choose forest-of-trees, because they are a well-established approach for parallel scalable adaptive mesh generation and partitioning [43]; in particular, we aim to exploit the highly-scalable parallel FE framework that supports $h$-adaptivity on forest-of-trees of Chapter 2.

For FE applications, mesh non-conformity hardens the construction of conforming FE spaces and the subsequent steps in the simulation. For the sake of alleviating this extra complexity, we follow common practice [27, 43] of enforcing the *2:1-balance* or *1-irregularity* condition, that prescribes, at most, 2:1 size relations between neighbouring

cells, see Figure 3.2. A general definition of this condition depends on the dimension of the geometrical entity across two neighbouring cells, see Definition 2.2.8. In our context, for simplicity, we adopt the criterion that 2:1 size relations must hold for any two geometrically neighbouring cells (i.e. across a vertex, edge or face), the so-called 2:1 0-balance.[1] 2:1 balance ensures that hanging DOF constraints are *single-level* or *direct*, i.e. hanging DOFs are not constrained by other hanging DOFs, see Proposition 2.3.8 Furthermore, in a distributed-memory environment, any hanging DOF constraint can be locally applied, as each subdomain holds a single layer of ghost cells, see Proposition 2.4.1.

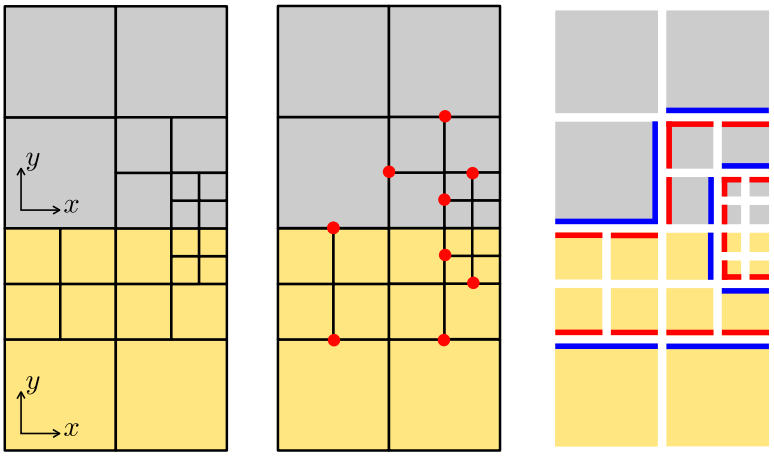



Figure 3.2: 2:1 0-balanced forest-of-quadtrees mesh with 1:4 refinement. There are two quadtrees, depicted in yellow and grey. Hanging VEFs are marked in red: vertices in (a), while edges in (b). All of these VEFs are proper subsets of the (owner) edges marked in blue in (c).

Although the exposition from Sections 3.2.3 to 3.2.5 assumes the mesh is a 2:1 balanced forest-of-trees mesh (with isotropic refinements), all concepts introduced there can be generalised to other families of non-conforming meshes. However, in order to accommodate conforming FE spaces, they must fulfil two necessary, but not sufficient, conditions: (1) all hanging-to-owner VEF relations meet Assumption 3.2.1 and (2) any (chain of) constraints defined on the mesh ends with unconstrained DOFs, i.e. the mesh does not produce cyclic hanging DOF constraint dependencies. Apart from forest-of-trees, other meshing approaches, e.g. anisotropic *solvable* meshes in [47], fulfil (1) and (2). It is not in our scope to fully characterise all possible families of non-conforming meshes that can be considered in this chapter.

We introduce now the immersed boundary setting on top of the artificial domain $\Omega^{\text{art}}$. For the sake of simplicity and without loss of generality, the boundary of the physical domain $\partial\Omega$ is represented by the zero level-set of a known scalar function $\varphi^{\text{ls}}$, namely $\partial\Omega \doteq \{\boldsymbol{x} \in \mathbb{R}^d : \varphi^{\text{ls}}(\boldsymbol{x}) = 0\}$. The problem geometry could be described

[1] We assume 0-balance, because we are interested in Lagrangian FEs. This choice leads to a correct (parallel) FE solver, although weaker 1-balance would also be enough, see Proposition 2.3.9.

by other means, e.g. from 3D CAD data, by providing techniques to compute the intersection between cell edges and surfaces (see, e.g. [129]). In any case, the following exposition does not depend on the way geometry is handled.

Let now the physical domain be defined as the set of points where the level-set function is negative, namely $\Omega \doteq \{\boldsymbol{x} \in \mathbb{R}^d : \varphi^{\text{ls}}(\boldsymbol{x}) < 0\}$. For any cell $T \in \mathcal{T}_h$, let us also define the quantity $\eta_T \doteq \text{meas}_d(T \cap \Omega^i)/\text{meas}_d(T)$, where $\text{meas}_d(\cdot)$ denotes the $d$-dimensional measure, and a user-defined parameter $\eta_0 \in (0, 1]$. In order to isolate badly cut cells, we classify cells of $\mathcal{T}_h$ in terms of $\eta_T$ and $\eta_0$. A cell $T \in \mathcal{T}_h$ is: (1) *well-posed*, if $\eta_T \geq \eta_0$; (2) *ill-posed*, if $\eta_0 > \eta_T > 0$; or (3) *exterior*, if $\eta_T = 0$, i.e. $T \cap \Omega = \emptyset$. We remark that, for $\eta_0 = 1$, well-posed cells coincide with interior cells $T \subset \Omega$, whereas ill-posed ones are cut, see, e.g. [24, Figure 1(b)]. In the general case, $\eta_0 \neq 1$, well-posed cells can also be cut cells with a large enough portion inside the physical domain; the distinction between interior and cut cells is no longer relevant. The set of well-posed (resp. ill-posed and exterior) cells is represented with $\mathcal{T}_h^{\text{W}}$ and its union $\Omega_{\text{W}} = \bigcup_{T \in \mathcal{T}_h^{\text{W}}} \overline{T} \subset \Omega$ (resp. $(\mathcal{T}_h^{\text{I}}, \Omega_{\text{I}})$ and $(\mathcal{T}_h^{\text{E}}, \Omega_{\text{E}})$). We also have that $\{\mathcal{T}_h^{\text{W}}, \mathcal{T}_h^{\text{I}}, \mathcal{T}_h^{\text{E}}\}$ is a partition of $\mathcal{T}_h$. We let $\mathcal{T}_h^{\text{act}} \doteq \mathcal{T}_h^{\text{W}} \cup \mathcal{T}_h^{\text{I}}$ and $\Omega^{\text{act}} \doteq \Omega_{\text{W}} \cup \Omega_{\text{I}}$ denote the so-called *active* triangulation and domain.

We note that, for general expressions of the immersed boundary $\Gamma$, it could happen that a $T \in u_h$ is such that $T \cap \Omega$ is a set of connected domains that are disconnected with each other. In this case, we consider each of these disconnected components as a separate cut cell. Thus, we redefine the mesh $u_h$, replicating these cut cells for each disconnected part, and use the procedure above verbatim. The reason for this is to assure that aggregates are always connected domains and, e.g. the Deny-Lions lemma can be used to prove approximability properties. Alternatively, one could assume that $\Gamma$ has bounded curvature and the mesh is *fine enough* (see, e.g. [88]).

### 3.2.2 Cell aggregation

As later shown in Section 3.2.4, AgFE spaces are grounded on a cell aggregation map that assigns a well-posed cell to any ill-posed cell. We refer to this map as the *root cell map* $R : \mathcal{T}_h \to \mathcal{T}_h^{\text{W}}$; it takes any cell $T \in \mathcal{T}_h$ and returns a cell $R(T) \in \mathcal{T}_h^{\text{W}}$, referred to as the *root* cell. In order to define this map, we consider a partition of $\mathcal{T}_h$, denoted by $\mathcal{T}_h^{\text{ag}}$, into non-overlapping cell aggregates. Each aggregate $A_T$ is a connected set, composed of several ill-posed cells and *only* one well-posed root cell $T$. Aggregates forming $\mathcal{T}_h^{\text{ag}}$ are built with Algorithm 3.2.2; although we detail below the sequential version, the distributed one follows from considering the extra steps described in [187]:

**Algorithm 3.2.2** (Cell aggregation scheme)**.**

1. *Mark* well-posed *cells as touched and set* $R(T) = T$ *for any* $T \in \mathcal{T}_h^{\text{W}}$*. Leave remaining ones untouched.*

2. *For each untouched cell $T$, if there is at least one touched cell connected to it through a facet $F$ such that $F \cap \Omega \neq \emptyset$, aggregate the cell to the touched one $T^*$ that fulfils Definition 3.2.3, i.e. set $R(T) = R(T^*)$. If more than one touched cell meets the definition, pick one arbitrarily, e.g. the one with smaller global id.*

3. *Mark as touched all the cells aggregated in (2).*

4. *Repeat (2) and (3) until all cells are aggregated.*

We recall that facets refer to edges in 2D or faces in 3D. For non-conforming meshes, facet connections comprise those among cells of same or different size. Moreover, step (2) of Algorithm 3.2.2 relies on a rule for choosing among multiple candidate root cells. Our criterion seeks to minimise the bounding box size of the aggregate, i.e. its characteristic length, to improve accuracy of the AgFEM:

**Definition 3.2.3** (Closest root cell criterion)**.** *Given an untouched cell $T \in \mathcal{T}_h$ and the set of aggregating candidates*

$$\mathcal{L}(T) = \{T' \in \mathcal{T}_h : T' \text{ touched } \& \ \exists \text{ facet } F \in \mathcal{F}_T \text{ or } F \in \mathcal{F}_{T'} \text{ with } \bar{F} = \bar{T} \cap \overline{T'},\ F \cap \Omega \neq \emptyset\},$$

*that is, $\mathcal{L}$ is the set of touched cells connected to $T$ through a conforming or hanging facet $F$. The* closest *aggregating candidate $T^*$ satisfies*

$$\tilde{d}(T, T^*) = \min_{T' \in \mathcal{L}(T)} \tilde{d}(T, T')$$

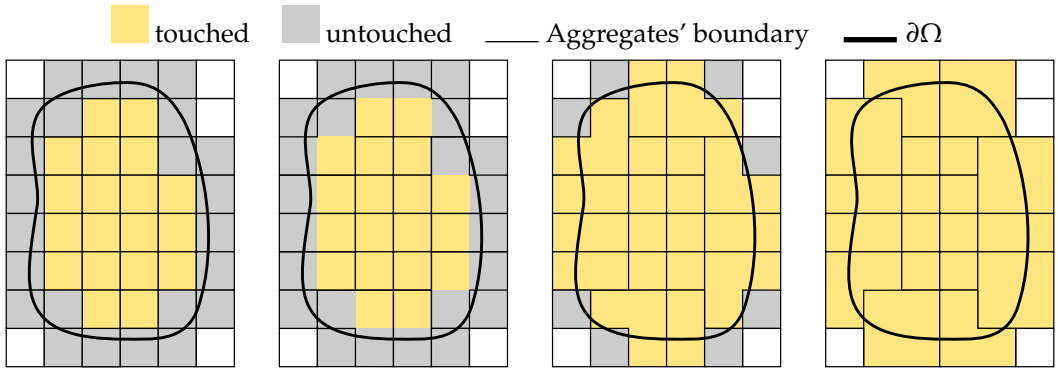
*with*

$$\tilde{d}(T, T') \doteq \frac{\max_{\gamma \in \mathcal{F}^0_{R(T')}, \delta \in \mathcal{F}^0_T} \|\boldsymbol{x}^\gamma - \boldsymbol{x}^\delta\|_\infty}{\max_{\gamma, \gamma' \in \mathcal{F}^0_{R(T')}} \|\boldsymbol{x}^\gamma - \boldsymbol{x}^{\gamma'}\|_\infty},$$

*for any $T' \in \mathcal{L}(T)$, where $\mathcal{F}^0_T$ denotes the set of vertices of $T$, $R(T)$ the root of $T$, $\boldsymbol{x}^\square$ the coordinates of vertex $\square$ and $\|\cdot\|_\infty$ denotes the infinity norm.*

In the first step of Algorithm 3.2.2, only well-posed cells are touched; their root cells are assigned to be themselves. In the second step, an ill-posed cell, when facet-connected to a touched cell, is aggregated to the closest available root cell, in the sense of Definition 3.2.3. Alternate criteria can be prescribed, e.g. in terms of the Euclidean distance between cell barycentres [24]. The output of the Algorithm 3.2.2 is the root cell map $R$.

Figure 3.3 shows an illustration of each step in Algorithm 3.2.2. The black thin lines represent the boundaries of the aggregates. Note that from step 1 to step 2, some of the lines between adjacent cells are removed, meaning that the two adjacent cells have been merged into the same aggregate. Aggregation schemes can be easily applied to arbitrary spatial dimensions. We also observe that, by construction of the cell aggregation scheme, maximum aggregate size is bounded above by a constant times the maximum cell size in the mesh [24].




(a) Step 1. (b) Step 2. (c) Step 3. (d) Step 4.

Figure 3.3: Illustration of the cell aggregation scheme defined in Algorithm 3.2.2 for $\eta_0 = 1$, i.e. well-posed iff interior and ill-posed iff cut, and a Cartesian grid.

### 3.2.3 Standard Lagrangian conforming finite element spaces

Our aim now is to define generic *conforming* FE spaces on top of tree-based meshes, referred to as *standard* or *std*. We define first a typical Lagrangian FE space that is conforming on body-fitted meshes, referred to as $\mathcal{V}_h^{\text{ncf}}$. Upon noting that $\mathcal{V}_h^{\text{ncf}}$ yields discontinuous approximations on hanging VEFs, we introduce a conforming subset of $\mathcal{V}_h^{\text{ncf}}$, namely $\mathcal{V}_h^{\text{std}}$, by imposing a set of linear constraints on hanging DOFs, such that global trace continuity is recovered.

We aim at solving a PDE problem in the physical domain $\Omega$, subject to boundary conditions on $\partial\Omega$. We assume Dirichlet conditions on $\Gamma_{\text{D}} \subset \partial\Omega$. At first, we consider that $\mathcal{T}_h$ is conforming and introduce a FE space, associated with $\mathcal{T}_h$, that takes the form

$$\mathcal{V}_h^{\text{ncf}} \doteq \{v \in \mathcal{C}^0(\Omega) \,:\, v|_T \in \mathcal{V}(T) \text{ for any } T \in \mathcal{T}_h\}.$$

It corresponds to the customary FE space defined in body-fitted conforming meshes. For unfitted meshes, it is not obvious to impose Dirichlet conditions in the approximation space in a strong manner. In consequence, we will assume *weak* imposition of Dirichlet boundary conditions on $\Gamma_{\text{D}}$.

We denote by $\mathcal{V}(T)$ a vector space of functions defined on $T \in \mathcal{T}_h$. For $d$-simplex meshes, we define the local space $\mathcal{V}(T) \doteq \mathcal{P}_q(T)$, i.e. the space of polynomials of order less or equal to $q$ in the variables $x_1, \ldots, x_d$. For $d$-cubes, we define $\mathcal{V}(T) \doteq \mathcal{Q}_q(T)$, i.e. the space of polynomials that are of degree less or equal to $q$ with respect to each variable in $x_1, \ldots, x_d$. We assume that all cells in $\mathcal{T}_h$ have local spaces $\mathcal{V}(T)$ of same order $q$. To simplify notation, we define the elemental functional spaces $\mathcal{V}(T)$ in the physical cell $T \subset \Omega^{\text{act}}$ (even though our computer implementation relies on reference parametric spaces, as usual).

For the sake of simplicity and without loss of generality, we limit ourselves to scalar-valued Lagrangian FEs. Extension to vector-valued or tensor-valued Lagrangian FEs is straightforward; it suffices to apply the same approach component by component.

Extension to other types of FEs (e.g. Nédélec elements [148]) is also possible; it only requires to work with the specific local DOFs of the cell, instead of the Lagrangian nodal values. According to this, the basis for $\mathcal{V}(T)$ is the Lagrangian basis (of order $q$) on $T$. We denote by $\Sigma_T$ the set of Lagrangian nodes of order $q$ of cell $T$, i.e. the set of local DOFs in $T$, and $\Sigma_f$ the set of local DOFs in $f \in \mathcal{F}_T$. There is a one-to-one mapping between nodes $\sigma \in \Sigma_T$ and shape functions $\phi_T^\sigma(\boldsymbol{x})$; it holds $\phi_T^\sigma(\boldsymbol{x}^{\sigma'}) = \delta_{\sigma\sigma'}$, where $\boldsymbol{x}^{\sigma'}$ are the space coordinates of node $\sigma'$ and $\delta$ is the Kronecker delta.

Let now $\sigma(T,\sigma') \in \Sigma$, with $\sigma' \in \Sigma_T$ and $T \in \mathcal{T}_h$, denote a local-to-global DOF map, obtained by gluing together DOFs that lie in the same geometrical position. The set of global DOFs in $\mathcal{T}_h$ is referred to as $\Sigma$. We will often abuse notation and use $\sigma$ to refer to both global and local views of the same DOF. It is common knowledge that $\sigma$ yields $\mathcal{C}^0$-continuous FE approximations on conforming meshes endowed with $\mathcal{V}_h^{\text{ncf}}$. However, if we let now $\mathcal{T}_h$ be non-conforming, FE functions defined in this way are generally discontinuous across hanging VEFs. Therefore, the resulting FE space is non-conforming (i.e. it is not a subspace of its infinite-dimensional counterpart) and, thus, not suitable for CG methods. To recover global continuous FE approximations, values of DOFs lying *only* on hanging VEFs cannot be arbitrary, they must be linearly constrained. In practice, this means to restrict $\mathcal{V}_h^{\text{ncf}}$ into a conforming FE subspace $\mathcal{V}_h^{\text{std}}$.

In order to introduce $\mathcal{V}_h^{\text{std}}$, let $\Sigma \doteq \{\Sigma^{\text{F}}, \Sigma^{\text{H}}\}$ denote a partition into *free* and *hanging* DOFs; the latter refers to the subset of global DOFs lying *only* on hanging VEFs, i.e. the inverse of the local-to-global DOF map is a set of local DOFs on top of hanging VEFs, only. We let now $\mathcal{M}_\sigma^{\text{H}}$ denote the subset of DOFs constraining $\sigma \in \Sigma^{\text{H}}$, referred to as the set of *master* DOFs of $\sigma$. Recalling Assumption 3.2.1 (2), we observe that, given $\sigma \in \Sigma^{\text{H}}$, lying on a hanging VEF $f$ of a cell $T$, its constraining DOFs are located in the closure of their owner VEF $\overline{f'}$ of a coarser cell $T'$ (Proposition 2.3.8). It follows that $\mathcal{M}_\sigma^{\text{H}} = \Sigma_{\overline{f'}}$. As $\mathcal{T}_h$ meets the 2:1 balance condition (cf. Section 3.2.1), constraining DOFs of hanging DOFs are free DOFs (Proposition 2.4.1), i.e.

$$\sigma \in \Sigma^{\text{H}} \Rightarrow \mathcal{M}_\sigma^{\text{H}} \subset \Sigma^{\text{F}}. \tag{3.1}$$

We will again invoke this property in Section 3.2.4. Setup and resolution of hanging DOF constraints for Lagrangian FE spaces is well-established knowledge [157] and, for conciseness, not reproduced here.

Given $v_h = \sum_{T\in\mathcal{T}_h}\sum_{\sigma\in\Sigma_T} v_h^\sigma \phi_T^\sigma \in \mathcal{V}_h^{\text{ncf}}$, we let

$$\mathcal{V}_h^{\text{std}} \doteq \{v_h \in \mathcal{V}_h^{\text{ncf}} : v_h^\sigma = \sum_{\sigma'\in\mathcal{M}_\sigma^{\text{H}}} C_{\sigma\sigma'}^{\text{H}} v_h^{\sigma'} \text{ for any } \sigma \in \Sigma^{\text{H}}\}, \tag{3.2}$$

where $C_{\sigma\sigma'}^{\text{H}} = \phi^{\sigma'}(\boldsymbol{x}^\sigma)$ and $\phi^{\sigma'}$ is the global shape function of $\mathcal{V}_h^{\text{ncf}}$ associated with $\sigma'$. Note that $C_{\sigma\sigma'}^{\text{H}} \neq 0$, by definition of $\sigma \in \Sigma^{\text{H}}$ and $\mathcal{M}_\sigma^{\text{H}}$. We observe that $\mathcal{V}_h^{\text{std}} \subset \mathcal{V}_h^{\text{ncf}}$ is conforming, in particular, $v_h \in \mathcal{C}^0(\Omega)$.

### 3.2.4 Aggregated Lagrangian finite element spaces

The space $\mathcal{V}_h^{\text{std}}$, introduced in Section 3.2.3, is conforming, but leads to arbitrarily ill-conditioned systems of linear algebraic equations, unless an extra technique is used to remedy it. This is the main motivation to introduce AgFE spaces (see, e.g. [20, 24]). The main idea is to remove from $\mathcal{V}_h^{\text{std}}$ problematic DOFs, associated with small cut cells, by constraining them as a linear combination of DOFs with local support in a well-posed cell. This operation leads to linear systems, whose condition number is not affected by small cuts [24]. In this section, we derive an AgFE space from $\mathcal{V}_h^{\text{std}}$. The key task, following our two-tier approach, is to combine the new linear constraints, arising from ill-posed DOF removal, with those already restricting $\mathcal{V}_h^{\text{std}}$, to enforce conformity. The resulting space must be well-defined, e.g. it must not have cycling constraint dependencies.

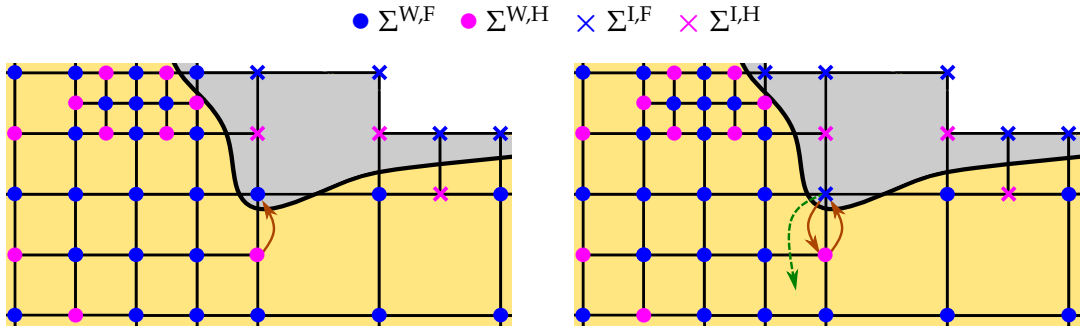
We start introducing sets of DOFs of the form $\Sigma^{\text{X,Y}}$, where $\text{X} \in \{\text{W}, \text{I}\}$ refers to well-posed or ill-posed and $\text{Y} \in \{\text{F}, \text{H}\}$ refers to free or hanging. We refer to Figure 3.4 for an illustration of this classification. Let $\Sigma^{\text{W,H}} \subset \Sigma^{\text{H}}$ denote the set of hanging DOFs that are located in $\mathcal{T}_h^{\text{W}}$, i.e. they are a local DOF of (at least one) well-posed cell. Let now $\Sigma^{\text{W,F}} \subset \Sigma^{\text{F}}$ be defined as follows.

**Definition 3.2.4.** *Given $\sigma \in \Sigma^{\text{F}}$, then $\sigma \in \Sigma^{\text{W,F}}$ is a* well-posed free DOF*, if it meets one of the following: (1) $\sigma$ is located in $\mathcal{T}_h^{\text{W}}$ or (2) $\sigma$ does not meet condition (1), but $\sigma \in \mathcal{M}_{\sigma'}^{\text{H}}$ for some $\sigma' \in \Sigma^{\text{W,H}}$, i.e. $\sigma$ is outside $\mathcal{T}_h^{\text{W}}$, but constrains a well-posed hanging DOF.*

This definition of well-posed free DOFs is backed by the numerical analysis in Appendix 3.4. There, we prove (cut-independent) well-posedness of the linear system arising from the AgFE method on the Poisson problem defined in (3.8). Equivalently, $\Sigma^{\text{W,F}}$ is formed by free DOFs that have local support on well-posed cells. We observe that it includes free DOFs surrounded by ill-posed cells that constrain well-posed hanging DOFs, see Figure 3.4(a). If we let $\Sigma^{\text{I,Y}} \doteq \Sigma^{\text{Y}} \setminus \Sigma^{\text{W,Y}}$, then it becomes clear that $\{\Sigma^{\text{W,F}}, \Sigma^{\text{W,H}}, \Sigma^{\text{I,F}}, \Sigma^{\text{I,H}}\}$ is a partition of $\Sigma$.

In contrast to free DOFs in $\Sigma^{\text{W,F}}$, any $\sigma \in \Sigma^{\text{I,F}}$ is liable to have arbitrarily small local support and, following the AgFEM rationale, must be constrained by DOFs in $\Sigma^{\text{W,F}}$. It follows that, in the AgFE space, free DOFs are reduced to free well-posed DOFs, i.e. $\Sigma^{\text{W,F}}$, whereas constrained DOFs are $\Sigma^{\text{C}} \doteq \{\Sigma^{\text{W,H}}, \Sigma^{\text{I,F}}, \Sigma^{\text{I,H}}\}$. Our goal is to show that any $\sigma \in \Sigma^{\text{C}}$ can be resolved with *direct* constraints, i.e. linear constraints of the same form as those in (3.2), in terms of well-posed free DOFs, only. For this purpose, we go over each subset of $\Sigma^{\text{C}}$ and characterise the subsets of $\Sigma^{\text{W,F}}$ constraining them, as well as the coefficients of the linear constraints. We also argue that the resulting constraint dependency graph, drawn in Figure 3.5, has no cyclic constraint dependencies. The discussion leads to the definition of an aggregated FE space $\mathcal{V}_h^{\text{ag}}$ that is a subspace of $\mathcal{V}_h^{\text{std}}$ with the same structure, i.e. restricted with linear constraints.

According to this, given $\sigma \in \Sigma^{\text{C}}$,




(a) Correct partition of $\Sigma$. The arrow points at a well-posed free DOF, surrounded by ill-posed cells, constraining a well-posed hanging DOF. Hence, it is well-posed due to Definition 3.2.4 (2).

(b) Incorrect partition of $\Sigma$. If the free DOF, pointed in Figure 3.4(a), is marked as ill-posed, it could be aggregated to the cell pointed by the dashed arrow, leading to a circular constraint dependency.

Figure 3.4: Classification of $\Sigma$ into $\{\Sigma^{W,F}, \Sigma^{W,H}, \Sigma^{I,F}, \Sigma^{I,H}\}$ on a portion of a mesh where $\eta_0 = 1$, i.e. well/ill-posed cell iff interior/cut cell. The physical domain is the yellow region and exterior cells are not shown. By marking DOFs, that meet Definition 3.2.4 (2), as well-posed (Figure 3.4(a)), we circumvent any possible circular constraint dependencies (Figure 3.4(b)).

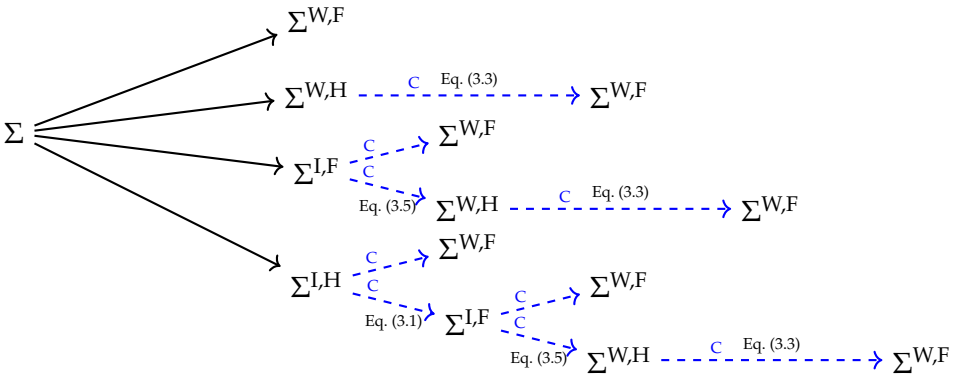



Figure 3.5: Constraint dependency graph of the AgFE space $\mathcal{V}_h^{ag}$. The set of global DOFs $\Sigma$ is partitioned into $\{\Sigma^{W,F}, \Sigma^{W,H}, \Sigma^{I,F}, \Sigma^{I,H}\}$. Subsets $\Sigma^{W,H}$, $\Sigma^{I,F}$ and $\Sigma^{I,H}$ are all constrained by $\Sigma^{W,F}$ with a dependency graph represented by dashed blue edges marked with a C. Dashed blue edges link a constrained subset with the subsets where its masters belong to. We observe that the graph has no cycles.

1. if $\sigma \in \Sigma^{W,H}$, then, by (3.1) and Definition 3.2.4 (2), master DOFs of $\sigma$ are necessarily contained in the set of well-posed free DOFs, i.e.

$$\sigma \in \Sigma^{W,H} \Rightarrow \mathcal{M}_\sigma^H \subset \Sigma^{W,F}. \tag{3.3}$$

   Therefore, linear constraints of $\sigma \in \Sigma^{W,H}$ remain unchanged in the new AgFE space.

2. If $\sigma \in \Sigma^{I,F}$, then we assume that we have composed the root cell map $R : \mathcal{T}_h \to \mathcal{T}_h^W$, introduced in Section 3.2.2, with a map between ill-posed free DOFs $\Sigma^{I,F}$ and ill-posed cells $\mathcal{T}_h^I$. In other words, we assign first each ill-posed free DOF to one

of its surrounding ill-posed cells. The chosen cell is then mapped onto a well-posed cell via $R$. Thus, the outcome of this composition is a map $K : \Sigma^{\mathrm{I,F}} \to \mathcal{T}_h^{\mathrm{W}}$, that assigns an ill-posed free DOF to a well-posed cell; see formal definitions in, e.g. [24, 187]. Given $\sigma \in \Sigma^{\mathrm{I,F}}$, let us denote by $\mathcal{M}_\sigma^{\mathrm{AA}}$ the subset of $\tilde{\sigma} \in \Sigma_{K(\sigma)}$, such that $\phi^{\tilde{\sigma}}_{K(\sigma)}(\boldsymbol{x}^\sigma) \neq 0$. We refer to $\mathcal{M}_\sigma^{\mathrm{AA}}$ as the set of "direct" AgFEM master DOFs of $\sigma \in \Sigma^{\mathrm{I,F}}$. As usual in AgFE methods, given $v_h \in \mathcal{V}_h^{\mathrm{std}}$ and $\sigma \in \Sigma^{\mathrm{I,F}}$, we enforce the constraint

$$v_h^\sigma = \sum_{\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{AA}}} C^{\mathrm{AA}}_{\sigma\tilde{\sigma}} v_h^{\tilde{\sigma}}, \quad \text{with } C^{\mathrm{AA}}_{\sigma\tilde{\sigma}} \doteq \phi^{\tilde{\sigma}}_{K(\sigma)}(\boldsymbol{x}^\sigma), \tag{3.4}$$

that is, we linearly extrapolate the nodal value of an ill-posed DOF with the values at the local DOFs of its root cell. In general, $\mathcal{M}_\sigma^{\mathrm{AA}}$ is composed of both free and hanging DOFs, i.e. some DOFs in the root cell can be hanging; the latter are not master DOFs, in the strict sense, and we need to remove them, i.e. rewrite (3.4) in terms of well-posed free DOFs, only. For that purpose, we introduce the partition $\mathcal{M}_\sigma^{\mathrm{AA}} = \{\mathcal{M}_\sigma^{\mathrm{AF}}, \mathcal{M}_\sigma^{\mathrm{AH}}\}$, with $\mathcal{M}_\sigma^{\mathrm{AF}} \doteq \mathcal{M}_\sigma^{\mathrm{AA}} \cap \Sigma^{\mathrm{F}}$, $\mathcal{M}_\sigma^{\mathrm{AH}} \doteq \mathcal{M}_\sigma^{\mathrm{AA}} \cap \Sigma^{\mathrm{H}}$. Since the image of $K$ is in $\mathcal{T}_h^{\mathrm{W}}$, it is clear that $\mathcal{M}_\sigma^{\mathrm{AF}} \subset \Sigma^{\mathrm{W,F}}$ and $\mathcal{M}_\sigma^{\mathrm{AH}} \subset \Sigma^{\mathrm{W,H}}$. We also have that

$$\sigma \in \Sigma^{\mathrm{I,F}} \Rightarrow \mathcal{M}_\sigma^{\mathrm{AA}} \subset \Sigma^{\mathrm{W,F}} \cup \Sigma^{\mathrm{W,H}}. \tag{3.5}$$

Recalling the first case, i.e. $\sigma \in \Sigma^{\mathrm{W,H}}$, the set of DOFs that are masters of $\mathcal{M}_\sigma^{\mathrm{AH}}$ is given by $\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}$ and, by (3.3), it is included in $\Sigma^{\mathrm{W,F}}$. If the previous property didn't hold, then $\Sigma^{\mathrm{I,F}} \cap \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}\right) \neq \emptyset$ and it could be possible that $\sigma \in \bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}$, i.e. $\sigma$ could (circularly) constrain itself, as in the situation depicted in Figure 3.4(b).

Hence, the "true" set of master DOFs of $\sigma \in \Sigma^{\mathrm{I,F}}$ is $\mathcal{M}_\sigma^{\mathrm{A}} \doteq \mathcal{M}_\sigma^{\mathrm{AF}} \cup \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}\right)$; note that the two set members of $\mathcal{M}_\sigma^{\mathrm{A}}$ are not necessarily disjoint, but $\mathcal{M}_\sigma^{\mathrm{A}} \subset \Sigma^{\mathrm{W,F}}$. Besides, recalling that hanging DOFs are constrained by free DOFs on top of VEFs of coarser neighbour cells, $\mathcal{M}_\sigma^{\mathrm{A}}$ are composed of DOFs located in root cells and (neighbouring) coarser cells around them. This property will be relevant again in Section 3.2.5.

After cancelling hanging DOFs, we can derive an analogous expression to (3.4), in terms of well-posed free DOFs only. The value of the AgFEM constraint, for $\sigma \in \Sigma^{\mathrm{I,F}}$ and $\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{A}}$, is

$$C^{\mathrm{A}}_{\sigma\tilde{\sigma}} \doteq \begin{cases} C^{\mathrm{AA}}_{\sigma\tilde{\sigma}} & \text{if } \tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{AF}}, \text{ only} \\ C^{\mathrm{AA}}_{\sigma\tilde{\sigma}} + \sum_{\left(\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}} \text{ s.t. } \tilde{\sigma} \in \mathcal{M}_{\sigma'}^{\mathrm{H}}\right)} C^{\mathrm{AA}}_{\sigma\sigma'} C^{\mathrm{H}}_{\sigma'\tilde{\sigma}} & \text{if } \tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{AF}} \cap \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}\right) \\ \sum_{\left(\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}} \text{ s.t. } \tilde{\sigma} \in \mathcal{M}_{\sigma'}^{\mathrm{H}}\right)} C^{\mathrm{AA}}_{\sigma\sigma'} C^{\mathrm{H}}_{\sigma'\tilde{\sigma}} & \text{if } \tilde{\sigma} \in \bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}, \text{ only.} \end{cases} \tag{3.6}$$

We refer to Figure 3.6 for an illustration of the three types of $\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{A}}$ in (3.6).

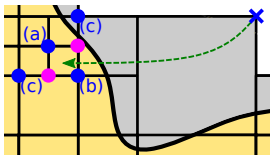



Figure 3.6: Close-up of Figure 3.4(a). Assuming that the top right ill-posed DOF is mapped to the well-posed cell pointed by the dashed arrow, we mark with letters and classify all DOFs $\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{A}}$, as they are distinguished in (3.6). In this sense, (a) shows $\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{AF}}$, only; (b) shows $\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{AF}} \cap \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}\right)$; (c) shows $\tilde{\sigma} \in \bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{AH}}} \mathcal{M}_{\sigma'}^{\mathrm{H}}$, only. Note that DOFs (c) are only in neighbour coarser cells.

3. If $\underline{\sigma \in \Sigma^{\mathrm{I,H}}}$, then $\sigma$ cannot be constrained as in the previous case, i.e. hanging DOF constraints have to be imposed first, to preserve conformity. According to this, $\sigma$ can be constrained by either well-posed or ill-posed free DOFs, i.e. $\mathcal{M}_\sigma^{\mathrm{H}} \subset \Sigma^{\mathrm{W,F}} \cup \Sigma^{\mathrm{I,F}}$; this is an immediate consequence of (3.1). If we consider now a partition of $\mathcal{M}_\sigma^{\mathrm{H}}$ into well-posed and ill-posed master DOFs and use case $\underline{\sigma \in \Sigma^{\mathrm{I,F}}}$ to remove ill-posed master DOFs, we deduce that

$$\mathcal{M}_\sigma^{\mathrm{H}} = \left(\mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{W,F}}\right) \cup \left(\mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{I,F}}\right) = \left(\mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{W,F}}\right) \cup \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{I,F}}} \mathcal{M}_{\sigma'}^{\mathrm{A}}\right) \subset \Sigma^{\mathrm{W,F}},$$

i.e. we can compute the constraints in terms of well-posed free DOFs only; again the two sets in the right-hand side are not necessarily disjoint. After cancelling the AgFEM constraints of $\sigma' \in \mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{I,F}}$, the constraint coefficient for $\sigma \in \Sigma^{\mathrm{I,H}}$ and $\sigma' \in \mathcal{M}_\sigma^{\mathrm{H}}$ becomes

$$C_{\sigma\sigma'}^{\mathrm{HA}} \doteq \begin{cases} C_{\sigma\sigma'}^{\mathrm{H}} & \text{if } \sigma' \in \left(\mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{W,F}}\right), \text{ only} \\ C_{\sigma\sigma'}^{\mathrm{H}} + \sum_{\left(\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{A}} \text{ s.t. } \sigma' \in \mathcal{M}_{\tilde{\sigma}}^{\mathrm{H}}\right)} C_{\sigma\tilde{\sigma}}^{\mathrm{A}} C_{\tilde{\sigma}\sigma'}^{\mathrm{H}} & \text{if } \left(\mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{W,F}}\right) \cap \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{I,F}}} \mathcal{M}_{\sigma'}^{\mathrm{A}}\right) \\ \sum_{\left(\tilde{\sigma} \in \mathcal{M}_\sigma^{\mathrm{A}} \text{ s.t. } \sigma' \in \mathcal{M}_{\tilde{\sigma}}^{\mathrm{H}}\right)} C_{\sigma\tilde{\sigma}}^{\mathrm{A}} C_{\tilde{\sigma}\sigma'}^{\mathrm{H}} & \text{otherwise.} \end{cases}$$

The last step to derive the AgFE space is to gather the previous cases, combining hanging and aggregation DOF constraints, into a unified form equivalent to (3.4). Given $\sigma \in \Sigma^{\mathrm{C}}$, the set of master DOFs is

$$\mathcal{M}_\sigma \doteq \begin{cases} \mathcal{M}_\sigma^{\mathrm{H}} & \text{if } \sigma \in \Sigma^{\mathrm{W,H}} \\ \mathcal{M}_\sigma^{\mathrm{A}} & \text{if } \sigma \in \Sigma^{\mathrm{I,F}} \\ \left(\mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{W,F}}\right) \cup \left(\bigcup_{\sigma' \in \mathcal{M}_\sigma^{\mathrm{H}} \cap \Sigma^{\mathrm{I,F}}} \mathcal{M}_{\sigma'}^{\mathrm{A}}\right) & \text{if } \sigma \in \Sigma^{\mathrm{I,H}}. \end{cases}$$

By definition, $\mathcal{M}_\sigma \subset \Sigma^{\mathrm{W,F}}$, for all $\sigma \in \Sigma^{\mathrm{C}}$, i.e. all constraints can be solved by free well-posed DOFs and, thus, there are no cyclic constraint dependencies; see also the

constraint dependency graph represented in Figure 3.5. On the other hand, the constraint coefficient for $\sigma \in \Sigma^{\mathrm{C}}$ and $\sigma' \in \mathcal{M}_\sigma$ is

$$C_{\sigma\sigma'} \doteq \begin{cases} \mathrm{C}^{\mathrm{H}}_{\sigma\sigma'} & \text{if } \sigma \in \Sigma^{\mathrm{W,H}} \\ \mathrm{C}^{\mathrm{A}}_{\sigma\sigma'} & \text{if } \sigma \in \Sigma^{\mathrm{I,F}} \\ \mathrm{C}^{\mathrm{HA}}_{\sigma\sigma'} & \text{if } \sigma \in \Sigma^{\mathrm{I,H}}. \end{cases}$$

With these notations, the (sequential) *aggregated* or *ag.* FE space can be readily defined as

$$\mathcal{V}_h^{\mathrm{ag}} \doteq \{v_h \in \mathcal{V}_h^{\mathrm{ncf}} : v_h^\sigma = \sum_{\sigma' \in \mathcal{M}_\sigma} C_{\sigma\sigma'} v_h^{\sigma'} \text{ for any } \sigma \in \Sigma^{\mathrm{C}}\}. \tag{3.7}$$

It is clear that $\mathcal{V}_h^{\mathrm{ag}} \subset \mathcal{V}_h^{\mathrm{std}} \subset \mathcal{V}_h^{\mathrm{ncf}}$. For the sake of brevity, further aspects, such as the definition of the resulting shape basis functions or finite element assembly operations are not covered. In the end, constraints supplementing $\mathcal{V}_h^{\mathrm{ag}}$ are of multipoint linear type, in the same way as those of $\mathcal{V}_h^{\mathrm{std}}$; they have been extensively covered in the literature, see e.g. [170, 187]. With regards to the implementation, we remark that the set up of $\mathcal{V}_h^{\mathrm{ag}}$ can also potentially reuse data structures and methods devoted to the construction of $\mathcal{V}_h^{\mathrm{std}}$ or, more generally, any other FE space endowed with linear algebraic constraints. To conclude, we stress out the fact that $\mathcal{V}_h^{\mathrm{ag}}$ retains good numerical properties proved for uniform meshes in [24]. We demonstrate this in Appendix 3.4.

### 3.2.5 Distributed-memory extension

After defining AgFEM in a serial context, we discuss its extension to a domain decomposition (DD) setup for implementation in a distributed-memory computing network. We start by setting up the partition of the mesh into subdomains: Let $\mathcal{S}$ be a partition of $\Omega^{\mathrm{art}}$ into subdomains obtained by the union of cells in the background mesh $\mathcal{T}_h$, i.e. for each cell $T \in \mathcal{T}_h$, there is a subdomain $S \in \mathcal{S}$ such that $T \subset S$ (see Figure 3.7(a)). We denote by $\mathcal{T}_h^{\mathrm{L}(S)}$ the set of *local* cells in subdomain $S \in \mathcal{S}$; naturally, $\{\mathcal{T}_h^{\mathrm{L}(S)}\}_{S\in\mathcal{S}}$ forms a partition of $\mathcal{T}_h$. We assume that $\mathcal{S}$ is easy to generate. This is a reasonable assumption in our embedded boundary context, where $\Omega^{\mathrm{art}}$ can be easily meshed with e.g. tree-based Cartesian grids, which are amenable to load-balanced partitions grounded on space-filling curves [43].

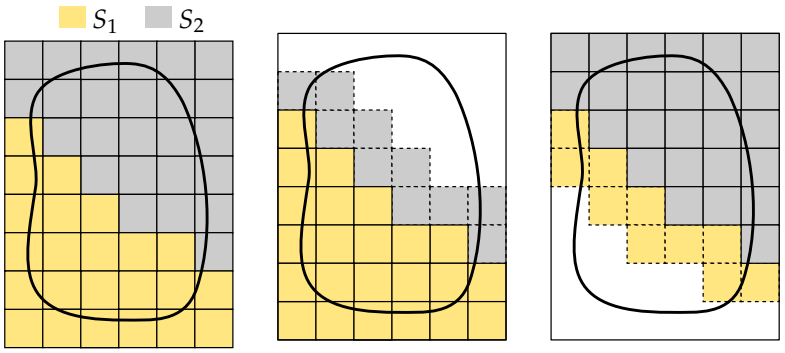
In a parallel, distributed-memory environment, each subdomain $S$ is mapped to a processor. Thus, each processor holds in memory a portion $\mathcal{T}_h^{\mathrm{L}(S)}$ of the global mesh $\mathcal{T}_h$. Naturally, local FE integration in $S$ is restricted to cells in $\mathcal{T}_h^{\mathrm{L}(S)}$. However, to correctly perform the parallel FE analysis, the processor-local portion of $\mathcal{T}_h$ is usually extended with adjacent off-processor cells, a.k.a. *ghost* cells. Ghost cells are essential to generate the global DOF numbering, in particular, to glue together DOFs in processors that represent the same global DOF. For constrained spaces, they are also needed to *locally* solve DOF constraints that expand beyond $\mathcal{T}_h^{\mathrm{L}(S)}$. Standard practice in large-scale FE codes is to constrain the ghost cell set to a single layer of ghost cells. Here, we refer to

them as the *true* ghosts, given by $\mathcal{T}_h^{\mathrm{TG}(S)} \doteq \{T \in \mathcal{T}_h \setminus \mathcal{T}_h^{\mathrm{L}(S)} : \overline{T} \cap \overline{S} \neq \emptyset\}$. This layer suffices to glue together global DOFs among processors for non-constrained spaces, but it is not necessarily enough for constrained ones, see Chapter 2.

Hereafter, all quantities refer to a given subdomain $S$ and we drop the subindex $S$, unless needed for clarity. We assume that our initial distributed-memory setting considers processors holding $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$ locally, as shown in Figure 3.7, and that we have built the distributed $\mathcal{V}_h^{\mathrm{std}}$. Recalling mesh assumptions in Section 3.2.1, we remark that the $S$-subdomain restriction of $\mathcal{V}_h^{\mathrm{std}}$ can be correctly built with processor-local info in $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$, by virtue of Proposition 2.4.1. Since we will invoke again this result, we reinterpret it here for our purposes.

**Proposition 3.2.5.** *Let $\mathcal{T}_h$ be a 2:1 0-balanced forest-of-trees mesh, then all constraint dependencies of hanging DOFs in $\mathcal{T}_h^{\mathrm{L}}$ do not expand beyond $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$, i.e. all hanging DOF constraints in $\mathcal{T}_h^{\mathrm{L}}$ can be resolved in $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$.*

We note that the key property that leads to Proposition 3.2.5 is that all coarser cells around $\mathcal{T}_h^{\mathrm{L}}$ are in $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$.



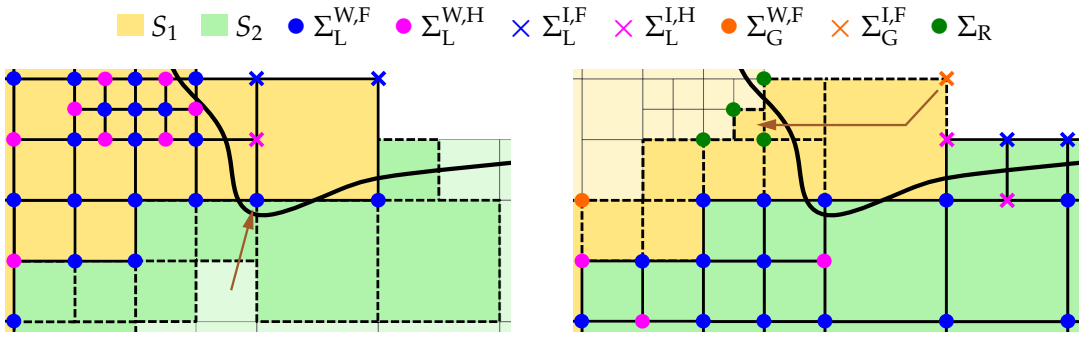

(a) $\mathcal{S} = \{S_1, S_2\}$. (b) $\mathcal{T}_h^{\mathrm{L}(S_1)}$ and $\mathcal{T}_h^{\mathrm{TG}(S_1)}$. (c) $\mathcal{T}_h^{\mathrm{L}(S_2)}$ and $\mathcal{T}_h^{\mathrm{TG}(S_2)}$.

Figure 3.7: Typical domain decomposition setup illustrated for two subdomains in a Cartesian grid. (a) Partition into subdomains. (b–c) Local and *true* ghost cells for each subdomain. Local cells are represented with solid lines, whereas ghost cells are represented with dashed ones.

In our context, we aim to resolve all combined hanging and aggregation DOF constraints with processor-local information only. This allows us to *locally* define the $S$-subdomain AgFE space, i.e. the restriction of $\mathcal{V}_h^{\mathrm{ag}}$ into $S$, leading to the distributed version of $\mathcal{V}_h^{\mathrm{ag}}$. To this end, we use $\Sigma_{\mathrm{Z}}^{\mathrm{X,Y}}$ to denote sets of DOFs where $\mathrm{X} \in \{\mathrm{W}, \mathrm{I}\}$ (well- or ill-posed) and $\mathrm{Y} \in \{\mathrm{F}, \mathrm{H}\}$ (free or hanging), as in Section 3.2.4, while $\mathrm{Z} \in \{\mathrm{L}, \mathrm{G}, \mathrm{R}\}$ will be clear shortly. When $X$ is omitted, we consider both well- and ill-posed DOFs. When $Y$ is also omitted, we consider any kind of DOFs, regardless of being well- or ill-posed, free or hanging.

Let us first denote the set of local DOFs by $\Sigma_{\mathrm{L}} \doteq \bigcup_{T \in \mathcal{T}_h^{\mathrm{L}}} \Sigma_T$. As in the sequential version of AgFEM, we have that $\{\Sigma_{\mathrm{L}}^{\mathrm{W,F}}, \Sigma_{\mathrm{L}}^{\mathrm{W,H}}, \Sigma_{\mathrm{L}}^{\mathrm{I,F}}, \Sigma_{\mathrm{L}}^{\mathrm{I,H}}\}$ forms a partition of $\Sigma_{\mathrm{L}}$, see Figure 3.8. Obviously, any $\sigma \in \Sigma_{\mathrm{L}}^{\mathrm{X,Y}}$ is locally available. Likewise, $\Sigma^{\mathrm{C}} \doteq \{\Sigma_{\mathrm{L}}^{\mathrm{W,H}}, \Sigma_{\mathrm{L}}^{\mathrm{I,F}}, \Sigma_{\mathrm{L}}^{\mathrm{I,H}}\}$

is the subset of local constrained DOFs. Given $\sigma \in \Sigma^{\mathrm{C}}$ we seek to resolve its constraint dependency locally, in the scope of the processor. For this purpose, we identify, in the next paragraphs, two sets of master DOFs that expand beyond $\mathcal{T}_h^{\mathrm{L}}$, represented in Figure 3.8(b). We argue alongside that, while local hanging DOFs can be locally resolved in $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{G}}$, thanks to Proposition 3.2.5, the processor must be supplemented with an extra ghost cell set, namely $\mathcal{T}_h^{\mathrm{RG}}$, to resolve local ill-posed DOFs. If all DOFs constraining $\sigma \in \Sigma^{\mathrm{C}}$ are locally available, then the processor can cancel all constraints, reusing without communication the strategy of Section 3.2.4. This operation leads to the $S$-subdomain AgFE space we are looking for.




(a) Partial view of $\mathcal{T}_h^{S_1}$.

(b) Partial view of $\mathcal{T}_h^{S_2}$.

Figure 3.8: Classification of DOFs in a hypothetical distributed-memory setting of Figure 3.4. Note that light-shaded cells are not in $\mathcal{T}_h^{S_i}$, $i = 1, 2$. Arrow at $\mathcal{T}_h^{S_1}$ points at a free DOF, whose well-posed status can only be known by nearest-neighbour exchange; indeed, it has local support in a well-posed $S_2$-cell that is not in $\mathcal{T}_h^{S_1}$. On another note, arrow at $\mathcal{T}_h^{S_2}$ maps an ill-posed free DOF to its associated root cell, via the $K_S$ map. All its constraining DOFs, after cancelling root hanging DOFs, are marked in green.

The first set of master DOFs beyond $\mathcal{T}_h^{\mathrm{L}}$ is $\Sigma_{\mathrm{G}} \doteq \left(\bigcup_{\sigma \in \Sigma_{\mathrm{L}}^{\mathrm{H}}} \mathcal{M}_\sigma^{\mathrm{H}}\right) \setminus \Sigma_{\mathrm{L}}$, defined as the set of *ghost* master DOFs that constrain (well- or ill-posed) hanging DOFs. Since we have single-level hanging DOF constraints, cf. (3.1), then $\Sigma_{\mathrm{G}}^{\mathrm{H}} = \emptyset$ and $\Sigma_{\mathrm{G}} \equiv \Sigma_{\mathrm{G}}^{\mathrm{F}}$. In contrast, $\Sigma_{\mathrm{G}}$ can contain either well- or ill-posed DOFs. Apart from that, any $\sigma \in \Sigma_{\mathrm{G}}^{\mathrm{F}}$ is located in $\mathcal{T}_h^{\mathrm{TG}}$, due to Proposition 3.2.5. Thus, it is locally available.

We bring now attention to the fact that, in order to locally determine whether a DOF of $V^{\mathrm{std}}$ is well- or ill-posed, the processor needs to have in its scope the full list of cells where the DOF has local support. This is clearly not necessarily available when $\sigma \in \Sigma_{\mathrm{G}}$, but there can even exist $\sigma \in \Sigma_{\mathrm{L}}$ that miss part of this information, see Figure 3.8(a). However, due to Proposition 3.2.5, all DOFs of $V^{\mathrm{std}}$ with local support in $\mathcal{T}_h^{\mathrm{L}}$ are located in $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$. As a result, a DOF that has local support in a well-posed cell of a given subdomain $S$, either via (1) or (2) of Definition 3.2.4, will be necessarily detected by the corresponding processor. It follows that, there is always at least one neighbour processor, due to Proposition 3.2.5, that is able to correctly identify the DOF as well-posed with processor-local information. Hence, any DOF, that lacks local access to the aforementioned list of cells, can determine, with standard nearest-neighbour

communication, whether it has local support on a well-posed cell or not.

Recalling Section 3.2.4, to compute AgFEM constraints, processors need to define the distributed version of the $K$ map, namely $K_S$, for ill-posed free DOFs in $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{G}}$, i.e. $\Sigma_{\mathrm{L}}^{\mathrm{I,F}}$ and $\Sigma_{\mathrm{G}}^{\mathrm{I,F}}$. $K_S$ is obtained via composition of the $S$-subdomain root cell map $R_S$ with a map that assigns ill-posed cells to ill-posed free DOFs; $K_S$ maps ill-posed free DOFs to (well-posed) root cells. We refer to [187] for details on how to construct $R_S$; it leads to the same aggregates as the ones obtained with the sequential method. The only changes in the cell aggregation scheme of [187], with respect to our context, are those described in Section 3.2.2 and the fact that we also need to define $R_S$ for cells in $\mathcal{T}_h^{\mathrm{TG}}$, such that we can apply $K_S$ to any $\sigma \in \Sigma_{\mathrm{G}}^{\mathrm{I,F}}$, see Figure 3.8(b).

We recall that, given $\sigma \in \Sigma_{\mathrm{L\cup G}}^{\mathrm{I,F}}$, where $\Sigma_{\mathrm{L\cup G}}^{\mathrm{I,F}}$ stands for $\Sigma_{\mathrm{L}}^{\mathrm{I,F}} \cup \Sigma_{\mathrm{G}}^{\mathrm{I,F}}$, $\mathcal{M}_\sigma^{\mathrm{A}}$ is the set of (well-posed) master DOFs by aggregation, *after* removing hanging DOF constraints in root cells. Hence, $\mathcal{M}_\sigma^{\mathrm{A}}$ is composed of (well-posed) free DOFs in root cells and (well-posed) free DOFs at their neighbouring coarser cells. We observe now that the image of $K_S$, i.e. $\operatorname{Im} K_S$, gathers all (well-posed) root cells relevant to $S$. Moreover, we let $\mathcal{T}_h^{\mathrm{CA}}$ denote all coarser cells around cells in $\operatorname{Im} K_S$. It follows that master DOFs of $\Sigma_{\mathrm{L\cup G}}^{\mathrm{I,F}}$ are located in $\operatorname{Im} K_S \cup \mathcal{T}_h^{\mathrm{CA}}$. However, in general, we cannot guarantee that $(\operatorname{Im} K_S \cup \mathcal{T}_h^{\mathrm{CA}}) \subset (\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}) \cap \mathcal{T}_h^{\mathrm{W}}$. In other words, there can be DOFs in $\Sigma_{\mathrm{L\cup G}}^{\mathrm{I,F}}$ that are mapped to a well-posed cell and/or depend on a coarser cell beyond $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$, as in Figure 3.9. These cells must be available in the processor, otherwise we cannot locally compute all aggregation DOF constraints.

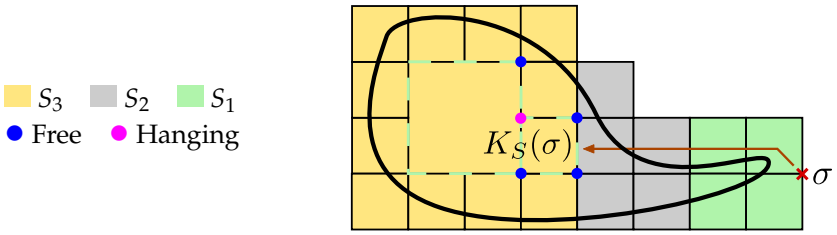



Figure 3.9: A case where an ill-posed DOF $\sigma$ is constrained by a root cell $K_S(\sigma)$ in a remote subdomain. Besides, $K_S(\sigma)$ touches a coarser cell. The value at $\sigma$ is constrained by the nodal values of cell $K_S(\sigma)$, but one of them is a hanging DOF. As a result, to fully resolve the constraint, it is necessary to send to $S_1$ data from the root cell and its coarser neighbour; both cells are in subdomain $S_3$ and marked with dashed green contour. We observe that $S_1$ and $S_3$ are not nearest-neighbour subdomains.

According to this, the second set of master DOFs beyond $\mathcal{T}_h^{\mathrm{L}}$ is $\Sigma_{\mathrm{R}} \doteq \left(\bigcup_{\sigma\in\Sigma_{\mathrm{L\cup G}}^{\mathrm{I,F}}} \mathcal{M}_\sigma^{\mathrm{A}}\right) \setminus \left(\Sigma_{\mathrm{L}}^{\mathrm{W,F}} \cup \Sigma_{\mathrm{G}}^{\mathrm{W,F}}\right)$, defined as the set of *remote* DOFs constraining ill-posed free DOFs. We notice that $\Sigma_{\mathrm{R}}$ isolates all free constraining DOFs of ill-posed DOFs that are not in $\Sigma_{\mathrm{L}} \cup \Sigma_{\mathrm{G}}$ and, potentially, are located beyond $\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{TG}}$. By definition, $\Sigma_{\mathrm{R}} \equiv \Sigma_{\mathrm{R}}^{\mathrm{W,F}}$. Let now $\mathcal{T}_h^{\mathrm{RG}} \doteq (\operatorname{Im} K_S \cup \mathcal{T}_h^{\mathrm{CA}}) \setminus (\mathcal{T}_h^{\mathrm{L}} \cup \mathcal{T}_h^{\mathrm{G}})$ be the set of *remote* ghost cells. Clearly, if the ghost cell layer is extended with $\mathcal{T}_h^{\mathrm{RG}}$, then all DOFs in $\Sigma_{\mathrm{R}}$ are locally available. We note that intermediate hanging master DOFs on root cells, i.e. $\mathcal{M}_\sigma^{\mathrm{AH}}$ necessary to

compute $\mathcal{M}_{\sigma}^{\mathrm{A}}$, are also locally available. Algorithms in charge of importing data associated with cells in $\mathcal{T}_h^{\mathrm{RG}}$ are covered in [187] for uniform meshes. They are grounded on the so-called parallel direct and inverse path reconstruction schemes, which only need nearest-neighbour communication patterns. For non-conforming meshes, it suffices to modify them such that they account for non-conforming adjacency in the path reconstruction and also import missing coarser cells around roots into the processor. We stress the fact that, as shown in [187], this approach does not involve any mesh reconfiguration and repartition, e.g. it keeps the space-filling curve partition, which is essential for performance purposes. It also has little impact on overall parallel performance and scalability and it can be easily implemented in distributed-memory FE codes.

In conclusion, if a processor owns $\mathcal{T}_h^{\mathrm{L}}$ and is extended with (disjoint) off-processor cells $\mathcal{T}_h^{\mathrm{TG}}$ and $\mathcal{T}_h^{\mathrm{RG}}$, then we can proc-locally identify all constraining DOFs that expand beyond $\mathcal{T}_h^{\mathrm{L}}$, with $\Sigma_{\mathrm{G}}$ for hanging DOFs and $\Sigma_{\mathrm{R}}$ for ill-posed free DOFs. It follows that we can resolve all hanging and aggregation constraints of DOFs in $\mathcal{T}_h^{\mathrm{L}}$ *with processor-local information*, thus achieving the very desirable parallel algorithm design property of maximizing local work, while minimising inter-processor communication. Another relevant outcome is that we can accommodate to $\Sigma_{\mathrm{L}}$ the same constraint dependency graph of Figure 3.5; all subsequent steps, to derive expressions for (unified aggregation and hanging) sets of master DOFs and constraining coefficients, follow exactly the sequential counterpart in Section 3.2.4, using the corresponding subdomain definitions. It leads to the definition of the distributed version of $\mathcal{V}_h^{\mathrm{ag}}$ and will not be reproduced here to keep the presentation short.

## 3.3 Approximation of elliptic problems

For the sake of simplicity, we consider the Poisson equation with non-homogeneous Dirichlet boundary conditions in the numerical experiments. After scaling with the diffusion term, the equation reads: *find* $u \in H^1(\Omega)$ *such that*

$$-\Delta u = f, \quad \text{in } \Omega, \qquad u = g, \quad \text{on } \Gamma_{\mathrm{D}} \doteq \partial\Omega, \tag{3.8}$$

where $f \in H^{-1}(\Omega)$ is the source term and $g \in H^{1/2}(\partial\Omega)$ is the prescribed value on the Dirichlet boundary. In the numerical tests, we study both $\mathcal{V}_h^{\mathrm{std}}$ and $\mathcal{V}_h^{\mathrm{ag}}$, see Sections 3.2.3 and 3.2.4, as possible choices of $\mathcal{V}_h^{\mathrm{x}}$. As stated in Section 3.2.3, we consider weak imposition of boundary conditions, since unfitted methods do not easily accommodate prescribed values in a strong sense. As usual in the embedded boundary community, we resort to Nitsche's method to circumvent this problem [24, 35, 167]. We observe that this approach provides a consistent numerical scheme with optimal convergence rates (even for high-order FEs). According to this, we approximate (3.8) with

the variational formulation: *find* $u_h \in \mathcal{V}_h^{\mathrm{x}}$ *such that* $\mathrm{a}(u_h, v_h) = \mathrm{b}(v_h)$ *for all* $v_h \in \mathcal{V}_h^{\mathrm{x}}$, with

$$\begin{aligned} \mathrm{a}(u_h, v_h) &\doteq \int_\Omega \nabla u_h \cdot \nabla v_h \, \mathrm{d}\Omega + \int_{\partial\Omega} \left(\tau u_h v_h - u_h \left(\boldsymbol{n} \cdot \nabla v_h\right) - v_h \left(\boldsymbol{n} \cdot \nabla u_h\right)\right) \, \mathrm{d}\Gamma, \quad \text{and} \\ \mathrm{b}(v_h) &\doteq \int_\Omega v_h f \, \mathrm{d}\Omega + \int_{\partial\Omega} \left(\tau v_h g - \left(\boldsymbol{n} \cdot \nabla v_h\right) g\right) \, \mathrm{d}\Gamma, \end{aligned} \tag{3.9}$$

with $\boldsymbol{n}$ being the outward unit normal on $\partial\Omega$. We note that forms $\mathrm{a}(\cdot,\cdot)$ and $\mathrm{b}(\cdot)$ include the usual terms, resulting from the integration by parts of (3.8), plus additional terms associated with the weak imposition of Dirichlet boundary conditions with Nitsche's method [29, 146]. We prove well-posedness of Problem (3.9) in Section 3.4.2, considering $\mathcal{V}_h^{\mathrm{ag}}$ as the discretisation space.

Coefficient $\tau > 0$ denotes a mesh-dependent parameter that has to be large enough to ensure coercivity of $\mathrm{a}(\cdot,\cdot)$. It is prescribed with the same rationale given in [187, Section 4.2]. For $\mathcal{V}_h^{\mathrm{ag}}$, we have that $\tau = \beta^{\mathrm{ag}} h_T^{-1}$ for all $T \in \mathcal{T}_h^{\mathrm{I}}$, where $h_T$ is the cell characteristic size and $\beta^{\mathrm{ag}}$ is a user-defined constant parameter. Numerical experiments take $\beta^{\mathrm{ag}} = 25.0$; this value is enough for having a well-posed problem in all cases considered in Section 3.5.4. When using $\mathcal{V}_h^{\mathrm{std}}$, the value in a generic ill-posed cell takes the form

$$\tau = \beta^{\mathrm{std}} \lambda_T^{\max}, \text{ for all } T \in \mathcal{T}_h^{\mathrm{I}}, \tag{3.10}$$

where $\beta^{\mathrm{std}} = 2.0$ and $\lambda_T^{\max}$ is the maximum eigenvalue of the generalised eigenvalue problem: *find* $\mu_T \in \mathcal{V}_h^{\mathrm{std}}|_T$ *and* $\lambda_T \in \mathbb{R}$ *such that*

$$\int_{T\cap\Omega} \nabla\mu_T \cdot \nabla\xi_T \, \mathrm{d}\Omega = \lambda_T \int_{T\cap\partial\Omega} (\nabla\mu_T \cdot \boldsymbol{n})(\nabla\xi_T \cdot \boldsymbol{n}) \, \mathrm{d}\Gamma, \text{ for all } \xi_T \in \mathcal{V}_h^{\mathrm{std}}|_T, \text{ for all } T \in \mathcal{T}_h^{\mathrm{I}}.$$

We notice that $\tau$ computed as in (3.10) can be arbitrarily large, as the measure of the cut $\Omega \cap T$, $T \in \mathcal{T}_h^{\mathrm{I}}$, tends to zero. This means that, in contrast with $\mathcal{V}_h^{\mathrm{ag}}$, strongly ill-conditioned systems of linear equations may arise with $\mathcal{V}_h^{\mathrm{std}}$, depending on the position of the cut.

## 3.4 Numerical analysis

In this section, we prove that both the condition number of (a) the mass matrix associated to the AgFE space defined in (3.7) and (b) the linear system arising from (3.9) are bounded. The bounds do not depend on the cut location (but they do depend on the well-posedness threshold $\eta_0$). We use the notation $A \lesssim B$ (resp. $A \gtrsim B$) to represent $A \leq CB$ (resp. $A \geq CB$) for a positive constant $C > 0$ independent of the interface-mesh intersection or the mesh cells sizes.

### 3.4.1 Mass matrix condition number

In order to bound the condition number of the mass matrix, we seek to show the equivalence, for functions in $\mathcal{V}_h^{\mathrm{ag}}$, between the $L^2(\Omega)$-norm and the Euclidean norm of well-posed free DOFs. We devote the next paragraphs to introduce necessary definitions and preliminary results. Given $u_h \in \mathcal{V}_h^{\mathrm{ag}}$, let us denote the nodal vector of well-posed free DOFs by $\boldsymbol{u}$. For a given $T \in \mathcal{T}_h$ and VEF $f$, the cell- or VEF-wise coordinate vector is represented with $\boldsymbol{u}_T$ or $\boldsymbol{u}_f$ and its characteristic sizes by $h_T$ or $h_f$. First, we rely on the maximum and minimum eigenvalues of the local mass matrix in the physical cell $T$ or any of its VEFs $f \in \mathcal{F}_T$:

$$\lambda_{\min} h_X^{d_X} \|\boldsymbol{u}_X\|_2^2 \leq \|u_h\|_{L^2(X)}^2 \leq \lambda_{\max} h_X^{d_X} \|\boldsymbol{u}_X\|_2^2, \quad \text{for } u_h \in \mathcal{V}(T), \tag{3.11}$$

with $X = T$ or $X = f \in \mathcal{F}_T$ and $\|\cdot\|_2$ denoting the Euclidean norm. The values of $\lambda_{\min}$, $\lambda_{\max} > 0$ only depend on the order of the FE space and can be computed for different orders on $n$-cubes or $n$-simplices [76]. By combining (3.11) for $T$ and one of its VEFs, we deduce the bound

$$\|u_h\|_{L^2(T)}^2 \gtrsim h_f^{d-d_f} \|u_h\|_{L^2(f)}^2 > 0, \quad \text{for } u_h \in \mathcal{V}(T),\ f \in \mathcal{F}_T. \tag{3.12}$$

We observe that (3.12) can be applied to any $T \in \mathcal{T}_h^{\mathrm{W}}$ and corresponding VEFs, because we are integrating on the whole objects. If we consider integration on the cut portion of the cell $\Omega \cap T$, (3.11) also holds, up to a positive constant that depends on the well-posedness threshold $\eta_0$. This is a consequence of the following result.

**Lemma 3.4.1.** *Given a well-posed cell $T \in \mathcal{T}_h^{\mathrm{W}}$ and $u_h \in \mathcal{V}(T)$, there exists $C(\eta_0) > 0$, dependent on the well-posedness threshold $\eta_0$, such that $\|u_h\|_{L^2(\Omega\cap T)}^2 \geq C(\eta_0)\|u_h\|_{L^2(T)}^2$.*

*Proof.* Since we consider a well-posedness threshold $0 < \eta_0 \leq 1$, any cell $T \in \mathcal{T}_h^{\mathrm{W}}$ can be either (i) (full) interior or (ii) cut. For case (i), the bound trivially holds. For case (ii), given any polynomial defined in the cell, in particular, any shape function, we must have that $\int_{\Omega\cap T} p(x)^2 \geq C(\eta_0) \int_T p(x)^2 > 0$, for a bounded, strictly positive, constant $C(\eta_0)$ that depends on $\eta_0$. If this were not the case, then we would have that $p(x)$ vanishes in $\Omega \cap T$, with $\mathrm{meas}_d(\Omega \cap T) \neq 0$. As $p(x)$ is a polynomial, the only possibility is that $p \equiv 0$ in $T$. Hence, the bound also holds for case (ii). □

**Remark 3.4.2.** *We observe that we generally do not have an analogous bound to that of Lemma 3.4.1 for $f \in \mathcal{F}_T$, with $T \in \mathcal{T}_h^{\mathrm{W}}$, because $\mathrm{meas}_{d_f}(f \cap \Omega)$ can be arbitrarily small.*

Now, we consider the partition $\Sigma^{\mathrm{W,F}} \doteq \{\Sigma_{\mathrm{int}}^{\mathrm{W,F}}, \Sigma_{\mathrm{ext}}^{\mathrm{W,F}}\}$, where $\Sigma_{\mathrm{int}}^{\mathrm{W,F}}$ groups DOFs that satisfy Definition 3.2.4 (1) and $\Sigma_{\mathrm{ext}}^{\mathrm{W,F}}$ those that satisfy Definition 3.2.4 (2). We prove next two lemmas that, along with Lemma 3.4.1, allow one to compute a lower bound of the $L^2(\Omega)$-norm of functions in $\mathcal{V}_h^{\mathrm{ag}}$ by the Euclidean norm of DOFs in $\Sigma_{\mathrm{ext}}^{\mathrm{W,F}}$. In the first one, we show that for any $\sigma \in \Sigma_{\mathrm{ext}}^{\mathrm{W,F}}$, located atop a coarse VEF $f_{\mathrm{C}}$, we can find a hanging VEF $f_{\mathrm{H}}$ *of a well-posed cell*, with the same dimension of and owned by $f_{\mathrm{C}}$.

**Lemma 3.4.3.** *Given $\sigma \in \Sigma^{\mathrm{W,F}}_{\mathrm{ext}}$, there exists $\sigma' \in \Sigma^{\mathrm{W,H}}$, such that $\sigma \in \mathcal{M}^{\mathrm{H}}_{\sigma'}$, and there exist VEFs $f_{\mathrm{C}} \in T$ and $f_{\mathrm{H}} \in T'$, with $T \in \mathcal{T}^{\mathrm{I}}_h$ and $T' \in \mathcal{T}^{\mathrm{W}}_h$, such that $\sigma \in \Sigma_{\overline{f_{\mathrm{C}}}}$, $\sigma' \in \Sigma_{\overline{f_{\mathrm{H}}}}$ and* $\dim(f_{\mathrm{C}}) = \dim(f_{\mathrm{H}})$.

*Proof.* We use Figure 3.10 to illustrate the proof. Given $\sigma \in \Sigma^{\mathrm{W,F}}_{\mathrm{ext}}$, by Definition 3.2.4 (2), there exists $\sigma' \in \Sigma^{\mathrm{W,H}}$, such that $\sigma \in \mathcal{M}^{\mathrm{H}}_{\sigma'}$. Note that $\sigma$ and $\sigma'$ are related by a nontrivial constraint, by definition of $\mathcal{M}^{\mathrm{H}}_{\sigma'}$ (see Section 3.2.3). In addition, by recalling how hanging DOFs and its constraining DOFs are related (see Proposition 2.3.9), there exist $f_{\mathrm{C}} \in \mathcal{F}_T$, with $T \in \mathcal{T}^{\mathrm{I}}_h$, and $f_{\mathrm{H}} \in \mathcal{F}_{T'}$, with $T' \in \mathcal{T}^{\mathrm{W}}_h$, such that $f_{\mathrm{C}}$ is the owner VEF of $f_{\mathrm{H}}$, $\sigma \in \Sigma_{\overline{f_{\mathrm{C}}}}$ and $\sigma' \in \Sigma_{\overline{f_{\mathrm{H}}}}$. If $\dim(f_{\mathrm{H}}) = \dim(f_{\mathrm{C}})$, the result follows immediately. Otherwise, let us denote by $\mathcal{R}_T$ the mesh resulting from applying the 1:$2^d$ isotropic refinement rule once to $T$. $T$ and $T'$ only differ by one level of refinement, by the 2:1 balance assumption, and $\mathcal{R}_T \cup T'$ forms a conforming mesh, by the construction of the refinement rule. It follows that $f_{\mathrm{H}}$ is one of the VEFs on the boundary of $\mathcal{R}_T$. As $f_{\mathrm{C}}$ is the only VEF of $T$ that contains $f_{\mathrm{H}}$ and the refinement rule implies a nontrivial partition of all VEFs of $T$, there exists $f'_{\mathrm{H}} \subsetneq f_{\mathrm{C}}$, such that $\dim(f'_{\mathrm{H}}) = \dim(f_{\mathrm{C}})$, $f_{\mathrm{H}} \subsetneq \overline{f'_{\mathrm{H}}}$ and $f'_{\mathrm{H}} \in \mathcal{F}_{T'}$. Clearly, $f'_{\mathrm{H}}$ is also a hanging VEF of $\mathcal{T}_h$, with $f_{\mathrm{C}}$ as owner and $\sigma' \in \Sigma_{\overline{f'_{\mathrm{H}}}}$. □

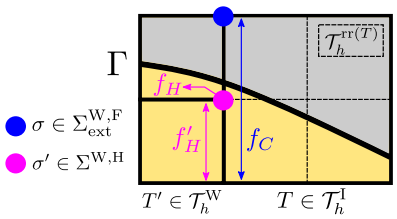



Figure 3.10: A 2D example to illustrate the proof of Lemma 3.4.3. .

The fact that $f_{\mathrm{C}}$ and $f_{\mathrm{H}}$ in Lemma 3.4.3 have the same dimension is key to prove the following bound.

**Lemma 3.4.4.** *Given $\sigma \in \Sigma^{\mathrm{W,F}}_{\mathrm{ext}}$, atop a VEF $f_{\mathrm{C}} \in \mathcal{F}_T$, with $T \in \mathcal{T}^{\mathrm{I}}_h$, we have the bound*

$$\|u_h\|^2_{L^2(f_{\mathrm{H}})} \gtrsim h^{d_{f_{\mathrm{C}}}}_{f_{\mathrm{C}}} \|\boldsymbol{u}_{f_{\mathrm{C}}}\|^2_2, \quad \text{for } u_h \in \mathcal{V}^{\mathrm{ag}}_h,$$

*where $f_{\mathrm{H}}$ is a hanging VEF, owned by $f_{\mathrm{C}}$, $f_{\mathrm{H}} \in \mathcal{F}_T$, $T \in \mathcal{T}^{\mathrm{W}}_h$, such that* $\dim(f_{\mathrm{C}}) = \dim(f_{\mathrm{H}})$.

*Proof.* First we see that, by Lemma 3.4.3, we can find $f_{\mathrm{H}}$ satisfying the hypotheses. As seen in (3.2) of Section 3.2.3, we have that hanging DOF linear constraints, defined for DOFs in $\overline{f_{\mathrm{H}}}$, lead to the relation $\boldsymbol{u}_{f_{\mathrm{H}}} = \mathbf{C}\boldsymbol{u}_{f_{\mathrm{C}}}$, where the coefficients of $\mathbf{C}$ are given by $\phi^{\sigma'}(\boldsymbol{x}^{\sigma})$ with $\sigma \in \Sigma_{\overline{f_{\mathrm{H}}}}$ and $\sigma' \in \Sigma_{\overline{f_{\mathrm{C}}}}$. Coefficients $\phi^{\sigma'}(\boldsymbol{x}^{\sigma})$ of $\mathbf{C}$ can be computed in a reference cell $\hat{T}$, by generating the mesh $\mathcal{R}_T$ and evaluating the shape functions of $\hat{T}$ at its nodes. Since shape functions are pointwise bounded, $\left|\phi^{\sigma'}(\boldsymbol{x}^{\sigma})\right|$ is bounded above, independently of mesh size and cuts. Using the above relation, we have that

$$\|u_h\|^2_{L^2(f_{\mathrm{H}})} = \boldsymbol{u}^T_{f_{\mathrm{H}}} \mathbf{M}_{f_{\mathrm{H}}} \boldsymbol{u}_{f_{\mathrm{H}}} = \boldsymbol{u}^T_{f_{\mathrm{C}}} \mathbf{C}^T \mathbf{M}_{f_{\mathrm{H}}} \mathbf{C} \boldsymbol{u}_{f_{\mathrm{C}}} = \lambda\, \boldsymbol{u}^T_{f_{\mathrm{C}}} \mathbf{M}_{f_{\mathrm{C}}} \boldsymbol{u}_{f_{\mathrm{C}}},$$

where $\mathbf{M}_f$ denotes the local FE mass matrix on VEF $f$ and, in the last equality, we consider the generalized eigenvalue problem $\mathbf{C}^T\mathbf{M}_{f_\mathrm{H}}\mathbf{C}\boldsymbol{u}_{f_\mathrm{C}} = \lambda\ \mathbf{M}_{f_\mathrm{C}}\boldsymbol{u}_{f_\mathrm{C}}$. Since $\mathbf{C}^T\mathbf{M}_{f_\mathrm{H}}\mathbf{C}$, $\mathbf{M}_{f_\mathrm{C}}$ are symmetric and $\mathbf{M}_{f_\mathrm{C}}$ is positive definite (due to (3.11)), the eigenvalues of the above problem are real. Moreover, the same argument in the proof of Lemma 3.4.1 ensures that, if a polynomial vanishes in $f_\mathrm{H}$, it must also vanish in $f_\mathrm{C}$. Therefore, we have that the smallest eigenvalue must be strictly positive, i.e. $\lambda_{\min} > 0$. It suffices to combine this result with (3.11) applied on $\mathbf{M}_{f_\mathrm{C}}$ to see that $\|u_h\|^2_{L^2(f_\mathrm{H})} \gtrsim \lambda_{\min}\, h_{f_\mathrm{C}}^{d_{f_\mathrm{C}}} \boldsymbol{u}_{f_\mathrm{C}}^T \boldsymbol{u}_{f_\mathrm{C}} > 0$. □

We need now some auxiliary definitions: Given $\sigma \in \Sigma^{\mathrm{W,F}}$, we let $\mathcal{S}_\sigma \doteq \{\sigma' \in \Sigma^\mathrm{C} : \sigma \in \mathcal{M}_{\sigma'}\}$ denote the set of DOFs constrained by $\sigma$ (either by mesh nonconformity or aggregation), the global shape function associated to $\sigma$, after solving constraints, is given by $\tilde{\phi}_\sigma \doteq \phi_\sigma + \sum_{\sigma'\in\mathcal{S}_\sigma} C_{\sigma'\sigma}\phi_{\sigma'}$ and we let $\mathcal{T}_h^\sigma \doteq \{T \in \mathcal{T}_h : \mathrm{supp}(\tilde{\phi}_\sigma) \cap T \neq 0\}$ denote the set of cells where $\tilde{\phi}_\sigma$ has local support. We observe that

$$0 < C_{\min} \leq |C_{\sigma'\sigma}| \leq C_{\max}, \tag{3.13}$$

where the bounds are independent of the size of the physical cell $h_T$ or cut location; this result has already been argued in Lemma 3.4.4 for hanging DOF constraints and [24, Lemma 5.1] for aggregation DOF constraints. Apart from that, we let $h_\sigma \doteq \max_{T\in\mathcal{T}_h^\sigma} h_T$. We note that the $h_T$ in the definition of $h_\sigma$ differ by a bounded value, that depends on the 2:1 0-balance restriction and the maximum aggregation distance, i.e. $h_\sigma = C(T)h_T$, for any $T \in \mathcal{T}_h^\sigma$.

We are now in position to show the sought-after equivalence between the $L^2$ norm of functions in $\mathcal{V}_h^{\mathrm{ag}}$ and the Euclidean norm of its nodal values.

**Proposition 3.4.5.** *Given $u_h \in \mathcal{V}_h^{\mathrm{ag}}$, the following bound holds:*

$$\|\boldsymbol{u}\|_\sigma^2 \lesssim \|u_h\|^2_{L^2(\Omega)} \lesssim \|\boldsymbol{u}\|_\sigma^2,$$

*for $\|\boldsymbol{u}\|_\sigma^2 \doteq \sum_{\sigma\in\Sigma^{\mathrm{W,F}}} h_\sigma^d u_\sigma^2$, with $u_\sigma$ the nodal value of $\sigma \in \Sigma^{\mathrm{W,F}}$.*

*Proof.* The upper bound straightforwardly follows from considering triangular inequality repeatedly and the fact that $|C_{\sigma'\sigma}|$ is bounded above (see (3.13)). For the lower bound, we use the results above. First, we see that, by Lemma 3.4.1,

$$\|u_h\|^2_{L^2(\Omega)} \geq \|u_h\|^2_{L^2(\Omega^\mathrm{W})} = \sum_{T\in\mathcal{T}_h^\mathrm{W}} \|u_h\|^2_{L^2(\Omega\cap T)} \gtrsim \sum_{T\in\mathcal{T}_h^\mathrm{W}} \|u_h\|^2_{L^2(T)}.$$

Then, by (3.11), we have the bound for $\Sigma^{\mathrm{W,F}}_{\mathrm{int}}$, that is,

$$\sum_{T\in\mathcal{T}_h^\mathrm{W}} \|u_h\|^2_{L^2(T)} \gtrsim \sum_{T\in\mathcal{T}_h^\mathrm{W}} h_T^d \|\boldsymbol{u}_T\|_2^2 \geq \sum_{\sigma\in\Sigma^{\mathrm{W,F}}_{\mathrm{int}}} h_\sigma^d u_\sigma^2.$$

On the other hand, we let $\mathcal{F}_{\mathrm{C}}$ denote all the set of VEFs $f_{\mathrm{C}}$, that contain at least one DOF of $\Sigma_{\mathrm{ext}}^{\mathrm{W,F}}$. Using Lemma 3.4.3, we pick for each $f_{\mathrm{C}} \in \mathcal{F}_{\mathrm{C}}$ a hanging VEF $f_{\mathrm{H}}$ of the same dimension of $f_{\mathrm{C}}$, with $f_{\mathrm{H}}$ touching a well-posed cell. We denote the set of all $f_{\mathrm{H}}$ by $\mathcal{F}_{\mathrm{H}}$. By (3.12) and Lemma 3.4.4, we obtain a bound for $\Sigma_{\mathrm{ext}}^{\mathrm{W,F}}$:

$$\sum_{T\in\mathcal{T}_h^{\mathrm{W}}} \|u_h\|^2_{L^2(T)} \gtrsim \sum_{f_{\mathrm{H}}\in\mathcal{F}_{\mathrm{H}}} h_{f_{\mathrm{H}}}^{d-d_{f_{\mathrm{H}}}} \|u_h\|^2_{L^2(f_H)} \gtrsim \sum_{f_{\mathrm{C}}\in\mathcal{F}_{\mathrm{C}}} h_{f_{\mathrm{C}}}^{d} \|\boldsymbol{u}_{f_{\mathrm{C}}}\|_2^2 \gtrsim \sum_{\sigma\in\Sigma_{\mathrm{ext}}^{\mathrm{W,F}}} h_\sigma^d u_\sigma^2$$

Combining the two bounds together, we get

$$\|u_h\|^2_{L^2(\Omega)} \gtrsim \sum_{\sigma\in\Sigma^{\mathrm{W,F}}} h_\sigma^d u_\sigma^2 \gtrsim \|\boldsymbol{u}\|_\sigma^2.$$

□

Note that the constants in Proposition 3.4.5 depend on the well-posedness threshold via Lemma 3.4.1, but are independent on the cut location. The following result is a direct consequence of Proposition 3.4.5.

**Corollary 3.4.6.** *The mass matrix* $\mathbf{M}$ *related to the aggregated FE space* $\mathcal{V}_h^{\mathrm{ag}}$ *is bounded by* $k(\mathbf{M}) \leq C$*, for a positive constant* $C > 0$ *independent on cut location.*

### 3.4.2 Well-posedness of the unfitted FE Poisson problem

Our goal now is to prove coercivity and continuity of the bilinear form in (3.9). To this end, let us assume that we bound the maximum level of refinement for any triangulation $u_h$ built recursively as a forest-of-trees; this is the case in practice, since available memory is limited. Hence, there exists $h_{\min} > 0$ such that $\min_{T\in u_h} h_T \geq h_{\min} > 0$. We begin with a trace inequality that is key to prove coercivity:

Given $T \in \mathcal{T}_h^{\mathrm{I}}$ and $T_1, \ldots, T_{m_T} \in \mathcal{T}_h^{\mathrm{W}}$, $m_T \geq 1$, the set of constraining well-posed cells (i.e. those constraining at least one DOF of $T$), we let

$$\Omega_T^{\mathrm{act}} \doteq \left( T \cup \bigcup_{i=1}^{m_T} T_i \right) \text{ and } \Omega_T \doteq \Omega \cap \Omega_T^{\mathrm{act}}.$$

Note that $m_T$ is bounded, due to the 2:1 0-balance restriction and the fact that the number of neighbour cells is bounded. In case that $T \in \mathcal{T}_h^{\mathrm{W}}$, the definitions above become $\Omega_T^{\mathrm{act}} = T$ and $\Omega_T = \Omega \cap T$.

**Lemma 3.4.7.** *Given* $u_h \in \mathcal{V}_h^{\mathrm{ag}}$ *and* $T \in \mathcal{T}_h^{\mathrm{act}}$*, there exists* $C(\eta_0) > 0$*, such that*

$$\|\boldsymbol{n} \cdot \nabla u_h\|^2_{L^2(\Gamma_{\mathrm{D}}\cap T)} \leq C(\eta_0) h_T^{-1} \|\nabla u_h\|^2_{L^2(\Omega_T)}.$$

*Proof.* We note first that $|\Gamma \cap T||T|^{-1} \leq C h_T^{-1}$; it can be proven for piecewise smooth boundaries for a constant that depends on the curvature of the surface patches and the maximum number of patches intersecting a cell. Combining this bound with the fact

that constraints are bounded (cf. (3.13)), we can readily use the ideas of the proof in [24, Lemma 5.6], followed by Lemma 3.4.1, to prove the result:

$$\|\boldsymbol{n}\cdot\nabla u_h\|^2_{L^2(\Gamma_{\mathrm{D}}\cap T)} \lesssim h_T^{-1}\|\nabla u_h\|^2_{L^2(\Omega_T^{\mathrm{act}})} \lesssim C(\eta_0)h_T^{-1}\|\nabla u_h\|^2_{L^2(\Omega_T)}.$$

□

We let now $\mathcal{V}(h) \doteq \mathcal{V}_h^{\mathrm{ag}} + H^2(\Omega)\cap H_0^1(\Omega)$ and define the following mesh dependent norms for $v \in \mathcal{V}(h)$:

$$|||v|||_h^2 \doteq \|\nabla v\|^2_{L^2(\Omega)} + \sum_{T\in\mathcal{T}_h^{\mathrm{act}}} \beta_T h_T^{-1}\|v\|^2_{L^2(\Gamma_{\mathrm{D}}\cap T)},$$
$$|||v|||^2_{\mathcal{V}(h)} \doteq |||v|||_h^2 + \sum_{T\in\mathcal{T}_h^{\mathrm{act}}} h_T\|\boldsymbol{n}\cdot\nabla v\|^2_{L^2(\Gamma_{\mathrm{D}}\cap T)}.$$

**Remark 3.4.8.** *By Lemma 3.4.7, norms $|||\cdot|||_h$ and $|||\cdot|||_{\mathcal{V}(h)}$ are equivalent in $\mathcal{V}_h^{\mathrm{ag}}$.*

In what follows, we assume that $\Omega$ has smoothing properties. Then we have the Discrete Poincaré-type inequality (see, e.g. [24, Lemma 5.8])

$$\|v\|_{L^2(\Omega)} \lesssim |||v|||_h, \quad \text{for any } v \in \mathcal{V}(h). \tag{3.14}$$

**Theorem 3.4.9.** *The aggregated unfitted FE problem in* (3.9) *satisfies the following bounds:*

*i) Coercivity:*

$$a(u_h,u_h) \gtrsim |||u_h|||_h^2, \quad \textit{for any } u_h \in \mathcal{V}_h^{\mathrm{ag}}, \tag{3.15}$$

*ii) Continuity:*

$$a(u,v) \lesssim |||u|||_{\mathcal{V}(h)}|||v|||_{\mathcal{V}(h)}, \quad \textit{for any } u,v \in \mathcal{V}(h), \tag{3.16}$$

*if $\beta_T > C(\eta_0)$, for some positive constant $C(\eta_0)$. In this case, there exists one and only one solution of* (3.9).

*Proof.* The proof is analogous to [24, Theorem 5.7]. Hence, we omit details. In order to show coercivity, given $u_h \in \mathcal{V}_h^{\mathrm{ag}}$, since we have that

$$a(u_h,u_h) = |||u_h|||_h^2 - 2\int_{\Gamma_{\mathrm{D}}} u_h(\boldsymbol{n}\cdot\nabla u_h)\mathrm{d}\Gamma,$$

it suffices to show that $2\int_{\Gamma_{\mathrm{D}}} u_h(\boldsymbol{n}\cdot\nabla u_h)\mathrm{d}\Gamma \lesssim |||u_h|||_h^2$. For a (well- or ill-posed) cut cell $T$, usage of the Cauchy-Schwarz inequality, Young's inequality and Lemma 3.4.7 leads to

$$2\int_{\Gamma_{\mathrm{D}}\cap T} u_h(\boldsymbol{n}\cdot\nabla u_h)\mathrm{d}\Gamma \le \alpha_T C(\eta_0)h_T^{-1}\|u_h\|^2_{L^2(\Gamma_{\mathrm{D}}\cap T)} + \alpha_T^{-1}\|\nabla u_h\|^2_{L^2(\Omega_T)}$$

For tree-based meshes, the number of neighbouring cells is bounded and the cell sizes $h_T$ of $T \in \Omega_T$ differ by a bounded value, that depends on the 2:1 0-balance restriction

and the maximum aggregation distance, one can take $\alpha_T > 0$ large enough, but uniform with respect to $h_T$ and cut location, such that:

$$2\int_{\Gamma_D} u_h(\boldsymbol{n}\cdot\nabla u_h)\mathrm{d}\Gamma \leq \sum_{T\in\mathcal{T}_h^{\mathrm{act}}} \alpha_T C(\eta_0) h_T^{-1}\|u_h\|^2_{L^2(\Gamma_D\cap T)} + \frac{1}{2}\|\nabla u_h\|^2_{L^2(\Omega)}$$

Therefore,

$$a(u_h,u_h) \geq \frac{1}{2}\|\nabla u_h\|^2_{L^2(\Omega)} + \sum_{T\in\mathcal{T}_h^{\mathrm{act}}} (\beta_T - \alpha_T C(\eta_0)) h_T^{-1}\|u_h\|^2_{L^2(\Gamma_D\cap T)}$$

For, e.g. $\beta_T > 2\alpha_T C(\eta_0)$, $a(u_h,u_h)$ is a norm. By construction, the lower bound for $\beta_T$ is independent of the $h_T$ and the intersection of $\Gamma_D$ and $\mathcal{T}_h^{\mathrm{act}}$, but it depends on the well-posedness threshold $\eta_0$, which is a user-defined value. It proves the coercivity property in (3.15). Thus, the bilinear form is non-singular. The continuity in (3.16) can be readily proved by repeated use of the Cauchy-Schwarz inequality. Since the problem is finite-dimensional and the corresponding linear system matrix is non-singular, there exists one and only one solution of this problem. □

The linear system matrix that arises from problem (3.9) can be defined as

$$A_{\sigma\sigma'} \doteq a(\tilde{\phi}_\sigma, \tilde{\phi}_{\sigma'}), \quad \text{for } \sigma,\sigma' \in \Sigma^{\mathrm{W,F}},$$

and we have that $\boldsymbol{u}\cdot\mathbf{A}\boldsymbol{u} = a(u_h,u_h)$, for any $u_h \in \mathcal{V}_h^{\mathrm{ag}}$. We can now use Proposition 3.4.5 and Theorem 3.4.9 to show that we have the same bound as the body fitted problem for the linear system matrix. This comes as a consequence of the following:

**Proposition 3.4.10.** *Given $u_h \in \mathcal{V}_h^{\mathrm{ag}}$, the following bound holds:*

$$\|\boldsymbol{u}\|_\sigma^2 \lesssim a(u_h,u_h) \lesssim h_{\min}^{-2}\|\boldsymbol{u}\|_\sigma^2.$$

*Proof.* The lower bound readily follows from coercivity in (3.15), (3.14) and the lower bound of Proposition 3.4.5:

$$a(u_h,u_h) \gtrsim |||u_h|||_h^2 \gtrsim \|u_h\|^2_{L^2(\Omega)} \gtrsim \|\boldsymbol{u}\|_\sigma^2$$

For the upper bound, we first see that the boundary term of $|||\cdot|||_h$ is bounded by $\|\boldsymbol{u}\|_\sigma^2$. Indeed, by scaling arguments and the equivalence of norms for finite-dimensional spaces, we have that

$$\|u_h\|^2_{L^2(\Gamma_D\cap T)} \lesssim h_T^{d-1}\|\boldsymbol{u}_T\|_2^2,$$

where $\boldsymbol{u}_T$ gathers both free and constrained DOFs. Adding up for all cells, invoking the fact that the number of neighbour cells and constraint coefficients are bounded

(see (3.13)), we obtain:

$$\sum_{T\in\mathcal{T}_h^{\text{act}}} \beta_T h_T^{-1} \|u_h\|^2_{L^2(\Gamma_{\text{D}}\cap T)} \lesssim h_{\min}^{-2} \|\boldsymbol{u}\|^2_\sigma. \tag{3.17}$$

On the other hand, using a standard inverse inequality and the upper bound of Proposition 3.4.5, which also holds for $\Omega^{\text{act}}$, we get

$$\|\nabla u_h\|^2_{L^2(\Omega)} \leq \|\nabla u_h\|^2_{L^2(\Omega^{\text{act}})} \lesssim h_{\min}^{-2} \|u_h\|^2_{L^2(\Omega^{\text{act}})} \lesssim h_{\min}^{-2} \|\boldsymbol{u}\|^2_\sigma. \tag{3.18}$$

Combining continuity of the bilinear form (3.9), in (3.16), with Remark 3.4.8, (3.17) and (3.18), we get the sought-after upper bound:

$$a(u_h, u_h) \lesssim |||u_h|||^2_{\mathcal{V}(h)} \lesssim |||u_h|||^2_h \lesssim h_{\min}^{-2} \|\boldsymbol{u}\|^2_\sigma.$$

□

By recalling Corollary 3.4.6, we obtain the following condition number bound.

**Corollary 3.4.11.** *The condition number of the linear system matrix* $\mathbf{A}$*, associated to Problem* (3.9)*, preconditioned by the mass matrix* $\mathbf{M}$*, related to the aggregated FE space* $\mathcal{V}_h^{\text{ag}}$*, satisfies the bound* $k(\mathbf{M}^{-1}\mathbf{A}) \leq C h_{\min}^{-2}$*, for a positive constant* $C > 0$ *independent on cut location.*

A priori error estimates can be proved following the same steps in [24, Section 5.6] and, for conciseness, are not covered here. The key arguments, leading to the estimates, are standard FE arguments, the results above and the fact that the nodal interpolator of a continuous function $u$ in $\mathcal{C}^0(\overline{\Omega})$, defined as $\mathcal{I}_h(u) \doteq \sum_{\sigma\in\Sigma^{\text{W,F}}} u(\boldsymbol{x}_\sigma)\tilde{\phi}_\sigma$, is bounded above, since constraints are also bounded above (see (3.13)).

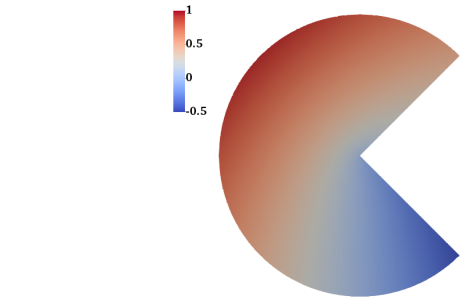
## 3.5 Numerical experiments

Our purpose in this section is to assess numerically the behaviour of $h$-AgFEM. We consider the model problem in Section 3.3, i.e. a Poisson equation with non-homogeneous Dirichlet boundary conditions and a Nitsche-type variational form. We introduce next the experimental benchmarks in Section 3.5.1, composed of several manufactured problems defined in a set of complex geometries. After this, we jump into the numerical experiments themselves. We describe and discuss the results of two sets of experiments, namely convergence tests in Section 3.5.4 and weak-scaling tests in Section 4.4.5.

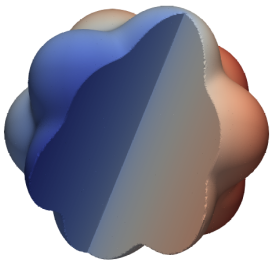
### 3.5.1 Experimental setup

The model problem is defined on five different 2D and 3D non-trivial domains shown in Figure 3.11: (a) a planar “pacman" shape, (b) a popcorn flake with a wedge removed, (c) a hollow block, (d) a 3-by-3 array of (c) and (e) a spiral. These geometries appear often in the literature to study robustness and performance of unfitted FE methods (see,

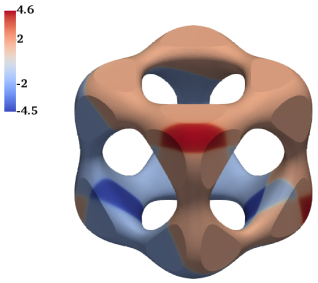
e.g. [24, 35]). The artificial domain $\Omega^{\text{art}}$, on top of which the mesh is generated, is the cuboid $[-1,1]^d$, $d = 2,3$, for cases (a-c), $[0,1]^3$ for case (d) and $[-1,1]^2 \times [0,2]$ for case (e).



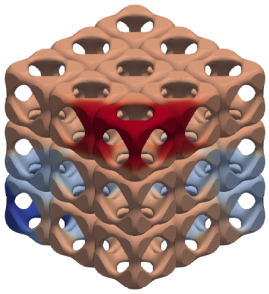

(a) Pacman



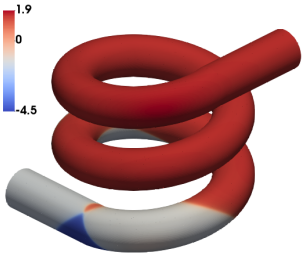
(b) Popcorn




(c) Hollow block



(d) H. b. array




(e) Spiral

Figure 3.11: Geometries and numerical solution to the problems studied in the examples. (a-b) consider the Fichera corner problem in (4.15), whereas (c-e) the multiple "shock" in (3.20).

As illustrated in Figure 3.11, for geometries (a-b), the source term and boundary conditions of the Poisson equation are defined, such that the PDE has the exact solution

$$\begin{aligned} &u(r,\theta) = r^\alpha \sin\alpha\theta,\ r = \sqrt{x^2+y^2},\ \theta = \arctan y/x,\ \alpha = 2/3, \\ &(x,y) \in \Omega \subset \mathbb{R}^2,\ z = 0 \text{ in 2D}, \qquad (x,y,z) \in \Omega \subset \mathbb{R}^3 \text{ in 3D}. \end{aligned} \tag{3.19}$$

The same applies to (c-e), but seeking a different exact solution given by

$$\begin{aligned} u(r) &= \sum_{i=1,3} \arctan \tau^i (r - r_0^i), \\ r &= ||\boldsymbol{x} - \boldsymbol{x}_0^i||_2,\ \boldsymbol{x} = (x,y,z) \in \Omega \subset \mathbb{R}^3, \end{aligned} \tag{3.20}$$

where $\|\cdot\|_2$ denotes the Euclidean norm and

$$\begin{aligned} &\tau^1 = 60, \quad (x_0^1, y_0^1, z_0^1) = (-1,-1,1), \quad r_0^1 = 2.5, \\ &\tau^2 = 80, \quad (x_0^2, y_0^2, z_0^2) = (1,1,-1), \quad r_0^2 = 1.75, \text{ and} \\ &\tau^3 = 120, \quad (x_0^3, y_0^3, z_0^3) = (0.5,-3,-3), \quad r_0^3 = 4.5. \end{aligned}$$

Problems (4.15) and (3.20) correspond to adapted versions of two classical hp-FEM benchmarks, namely, the Fichera corner and "shock" problems (see, e.g. [64]). Derivatives of solution $u$ in (4.15) are singular at the $r = 0$ axis; in particular, $u \in H^{1+\frac{2}{3}}(\Omega)$. Recalling *a priori* error estimates, it is well known that the rate of convergence of the standard FE method with uniform $h$-refinements, when applied to this case, is bounded by regularity only; the energy-norm error[2] satisfies $\|u - u_h\|_a \leq Ch^{-2/3}\|u\|_{H^{1+\frac{2}{3}}(\Omega)}$. However, by combining *a posteriori* error estimation and $h$-adaptive refinements, optimal rates of convergence can be restored [64]. On the other hand, problem (3.20) is characterised by three intersecting shocks. The solution to the problem is smooth, but it sharply varies in the neighbourhood of the shocks. In this case, $h$-adaptive standard FEM does not affect rates of convergence, but potentially yields meshes that minimise the number of cells required to achieve a given discretisation error.

The variety of shapes and benchmarks considered here aims to show (i) the capability of $h$-AgFEM of retaining the same benefits $h$-adaptivity brings, when combined with standard FEM, while being able (ii) to deal with complex and diverse 2D and 3D domains in a robust manner and (iii) to yield remarkable parallel efficiency with state-of-the-art *out-of-the-box* scalable iterative linear solvers for symmetric positive definite matrices. In order to do this, we confront numerical results obtained with $\mathcal{V}^{\text{ag}}$ against those of $\mathcal{V}^{\text{std}}$. In the plots, the two spaces are labelled as *aggregated* (or *ag.*) and *standard* (or *std.*). All examples run on *background Cartesian grids*, with standard isotropic 1:4 (2D) and 1:8 (3D) refinement rules; they are commonly referred to as quad- or octrees in 2D or 3D, resp. Apart from that, continuous FE spaces composed of first order Lagrangian finite elements are employed. We perform convergence tests using three different remeshing strategies (uniform refinements, Li and Bettess (LB), and Oñate and Bugeda (OB) [70]), both in serial and parallel environments. We also assess robustness to ill-conditioning, by computing approximations of condition numbers of the system matrices. Finally, we perform a weak scalability analysis for some selected ag. cases; Table 4.1 summarises the main parameters and computational strategies used in the numerical examples.

### 3.5.2 Experimental environment

Serial experiments are launched at the `TITANI` [181] cluster of the Universitat Politècnica de Catalunya (Barcelona, Spain) in a single DELL PowerEdge R630 node. The node is composed of 2 Intel Xeon E52650L v3 (1.8 GHz) CPUS, with 12 cores per processor and 128 GB of RAM. On the other hand, parallel experiments are carried out at the Marenostrum-IV (MN-IV) supercomputer [130], hosted by the Barcelona Supercomputing Centre. MN-IV is a petascale machine equipped with 3,456 compute nodes interconnected with the Intel OPA HPC network. Each node has 2x Intel Xeon Platinum 8160 multi-core CPUs, with 24 cores each (i.e. 48 cores per node). Among the available

[2]Recall that, for the unit-diffusion Poisson equation, the energy norm is given by $\|u\|_a^2 = \int_\Omega |\nabla u|^2 \, \mathrm{d}\Omega$.

| Description | Considered methods/values |
|---|---|
| Model problem | Poisson equation (Nitsche's formulation) |
| Problem geometry | 2D: Pacman shape; 3D: Popcorn flake,<br>Hollow block, Hollow block array and spiral |
| AMR benchmark | Fichera corner and multiple-shock problem [64] |
| Remeshing strategy | Uniform, Li and Bettess [119], and Oñate and Bugeda [149] |
| Experimental computer environment | Serial and parallel |
| Mesh topology | Single quad- or octree |
| Parallel mesh generation and partitioning tool | `p4est` library [43] |
| Well-posed cut cell criterion | $\eta_0 = 0.25$ |
| FE spaces | Aggregated $\mathcal{V}^{\text{ag}}$ and standard $\mathcal{V}^{\text{std}}$ |
| Cell type | Hexahedral cells |
| Interpolation | Piece-wise bi/trilinear shape functions |
| Linear solver | Sparse direct (serial)<br>Preconditioned conjugate gradients (parallel) |
| Parallel preconditioner | Smoothed-aggregation `GAMG` [1] |
| `GAMG` stopping criterion | $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 < 10^{-9}$ |
| Coef. in Nitsche's penalty term for $\mathcal{V}^{\text{ag}}$ | $\beta = 25.0$ |

Table 3.1: Summary of the main parameters and computational strategies used in the numerical examples.

nodes, our test cases are launched within the subset of 384 GB of RAM *high memory* nodes (216 out of 3,456 nodes). This allows us to generate enough error-curve points in the parallel convergence tests below, without requiring a too large fixed number of processors.

Concerning the software, an MPI-parallel implementation of the *h*-AgFEM method is available at `FEMPAR` [19, 180]. `FEMPAR` is linked against `p4est` v2.2 [43], as the octree Cartesian grid manipulation engine, and `PETSc` v3.11.1 [25, 25] distributed-memory linear algebra data structures and solvers. All software is compiled with Intel v18.0.5 compilers using system recommended optimization flags and linked against the Intel MPI Library (v2018.4.057) for message-passing and the BLAS/LAPACK available on the Intel MKL library for optimised dense linear algebra kernels. All floating-point operations are performed in IEEE double precision. Additionally, condition number estimates are computed outside `FEMPAR` with `MATLAB` function `condest`.[3]

### 3.5.3 Linear solver setup

A linear system of the form $\mathbf{A}\mathbf{x} = \mathbf{b}$ is derived from the FE discretization of the weak form described in Equation (3.9). At large scales, efficient parallel linear solvers are paramount to solve these systems. To show that $\mathcal{V}^{\text{ag}}$ leads to systems that are amenable to well established scalable linear solvers for standard FE analysis on body-fitted meshes, we resort to the broad suite of linear solvers available in the `PETSc` library [25]. In particular, we use a sparse direct solver from the MKL PARDISO package [2], for serial tests, and a preconditioned conjugate gradient method (CGM), for

[3]MATLAB is a trademark of THE MATHWORKS INC.

parallel ones. The selected preconditioner is a smoothed-aggregation AMG scheme called GAMG [1]. Both solvers and preconditioner are readily available through the Krylov Methods KSP module of PETSc.

In the serial environment, arbitrarily large condition numbers of discrete systems obtained with standard unfitted FEM can lead to the breakdown of the sparse direct solver. We mitigate this issue by configuring MKL PARDISO solver parameters, such that it can robustly deal with highly ill-conditioned matrices obtained with $\mathcal{V}^{\text{std}}$ discretizations. Listing 3.1 contains the configuration file informed to PETSc with the list of parameters changed from the default set.

```
1   -ksp_type preonly
2   -pc_type cholesky
3   -pc_factor_mat_solver_type mkl_pardiso
4   -mat_mkl_pardiso_1 1
5   -mat_mkl_pardiso_2 2
6   -mat_mkl_pardiso_8 2
7   -mat_mkl_pardiso_10 8
8   -mat_mkl_pardiso_11 1
9   -mat_mkl_pardiso_13 1
10  -mat_mkl_pardiso_21 1
```

Listing 3.1: Contents of the PETSc configuration file for serial experiments. Line 1 forces the Krylov solver to apply only the preconditioner (MKL PARDISO is available in PETSc as a preconditioner), line 2 prescribes a direct solver based on Cholesky factorization and line 3 MKL PARDISO. See [3] for details on the parameter values set in lines 4-10.

In the parallel environment, conversely, the linear solver is set up in favour of reducing, as much as possible, the deviation from the default configuration given by GAMG, to show that AgFEM blends well with common AMG solvers, whereas std. unfitted FEM does not. Customizations were only carried out to optimise the solver for symmetric positive definite matrices; they are exactly the same as the ones established in [187], which studies AgFEM on distributed uniform meshes, only. In order to advance the convergence test down to low global energy-norm error values, without being polluted by the linear solver accuracy, convergence of GAMG is declared when $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 < 10^{-9}$ within the first 500 iterations, where $\mathbf{r} \doteq \mathbf{b} - \mathbf{A}\mathbf{x}^{\text{cg}}$ is the unpreconditioned residual.

### 3.5.4 Convergence tests

Convergence tests in relative energy-norm error are carried out with three different mesh refinement strategies. The first one is uniform $h$-refinements, in pursuance of both exposing the behaviour of AgFEM, in absence of hanging node constraints, and the limited regularity of the Fichera corner problem. The remaining two are error-driven; they are distinguished by different optimality criteria on the elemental error

indicator $\gamma_T$, for any $T \in \mathcal{T}$. It is not in the scope of this chapter to design *a posteriori* error estimation techniques for AgFEM, although there are some works with other unfitted FE methods that explore this question [39]. Hence, since the target problems have known analytical solution, $\gamma_T$ is taken as the energy norm of the local true error $e = u - u_h$, that is,

$$\gamma_T = \|e\|_{a|_{T\cap\Omega}} = \|u - u_h\|_{a|_{T\cap\Omega}}, \quad T \in \mathcal{T}.$$

Error-driven mesh adaptation seeks an optimal mesh with an iterative procedure. In the examples below, optimality is declared when the global *absolute* discretization error, measured in energy norm, $\|e\|_a$ is below a prescribed quantity $\gamma$, i.e.

$$\|e\|_a \leq \gamma, \quad \gamma > 0. \tag{3.21}$$

Equation (3.21) is referred to as the *acceptability criterion*.

The process starts with an initial guess of the optimal mesh. After finding the approximate solution and the exact cell-wise error distribution, a new mesh is defined with a remeshing strategy. This step consists in comparing each $\gamma_T$ to a given threshold, commonly known as the *optimality criterion*, which is later defined. Depending on the result of the comparison, a different remeshing flag is assigned to the cell. If $\gamma_T$ is above the threshold, $T$ is marked for refinement. Otherwise, $T$ is left unmarked or, optionally, marked for coarsening, when $\gamma_T$ falls well below the threshold. Following this, the mesh is transformed, according to the cell-wise remeshing flags, and partitioned (in parallel experiments). Next, a new FE space is created, by distributing DOFs on top of the new mesh and computing the nonconforming DOF constraints and, if using $\mathcal{V}^{\text{ag}}$, also the ill-posed DOF constraints. After FE integration and assembly, the resulting linear system is solved and the cell-wise error distribution is updated. If the current mesh complies with the acceptability criterion of Equation (3.21), the process is stopped, otherwise it goes back to the application of the optimality criterion.

As mentioned before, two different optimality criteria (thresholds for refinement) are studied, see e.g. [70] for a detailed review. The first one, the LB [118, 119] criterion, establishes that the error distribution in an optimal mesh (denoted with *) is uniform, that is

$$\|e^*\|_{T^*\cap\Omega} = \frac{\gamma}{\sqrt{M^*}}, \quad T^* = 1, \dots, M^*,$$

where $M^*$ is the number of cells in the optimal mesh. At each mesh adaptation step, this quantity is estimated as

$$M^* = \gamma^{-d/m} \left( \sum_{T=1}^{M} \|e\|_T^{d/(m+d/2)} \right)^{(m+d/2)/m},$$

with $d$ the space dimension, $m$ the degree of the interpolation (in second-order elliptic problems) and $M$ the number of cells of the current iteration. On the other hand, the OB [149] criterion considers that the distribution of error *density* in an optimal mesh is

uniform, that is

$$\|e^*\|_{T^*\cap\Omega} = \frac{\gamma\Omega_{T^*\cap\Omega}^{1/2}}{\Omega^{1/2}}, \quad T^* = 1,\dots,M^*,$$

where $\Omega$ is the measure of the domain and $\Omega_{T^*\cap\Omega}$ is the measure of $T^*\cap\Omega$. While the former criterion has been proven [118] to provide standard body-fitted FE meshes satisfying Equation (3.21) with the least number of elements, the latter scales the threshold in terms of the size of the (ill-posed) cell. One of the goals of the following experiments is to see how both strategies perform in the context of unfitted FEs. Note that, with respect to standard body-fitted FEM, both remeshing strategies are almost applied verbatim to an unfitted FE setting; the only difference being that the local quantities in cut cells are computed in the interior part only, in the same way as for the local integration of the weak form stated in Equation (3.9).

Convergence tests with uniform $h$-refinements follow the usual procedure, whereas error-driven tests are controlled with a finite sequence of decreasing error objectives $\gamma_i$, $i > 1$. For each $i > 1$, the iterative procedure described above is carried out to find the mesh that complies with the acceptability criterion of Equation (3.21) with $\gamma = \gamma_i$. If subscript $\gamma_i$ refers to the quantities obtained at the last mesh iteration, at the end of the procedure we can extract the pair

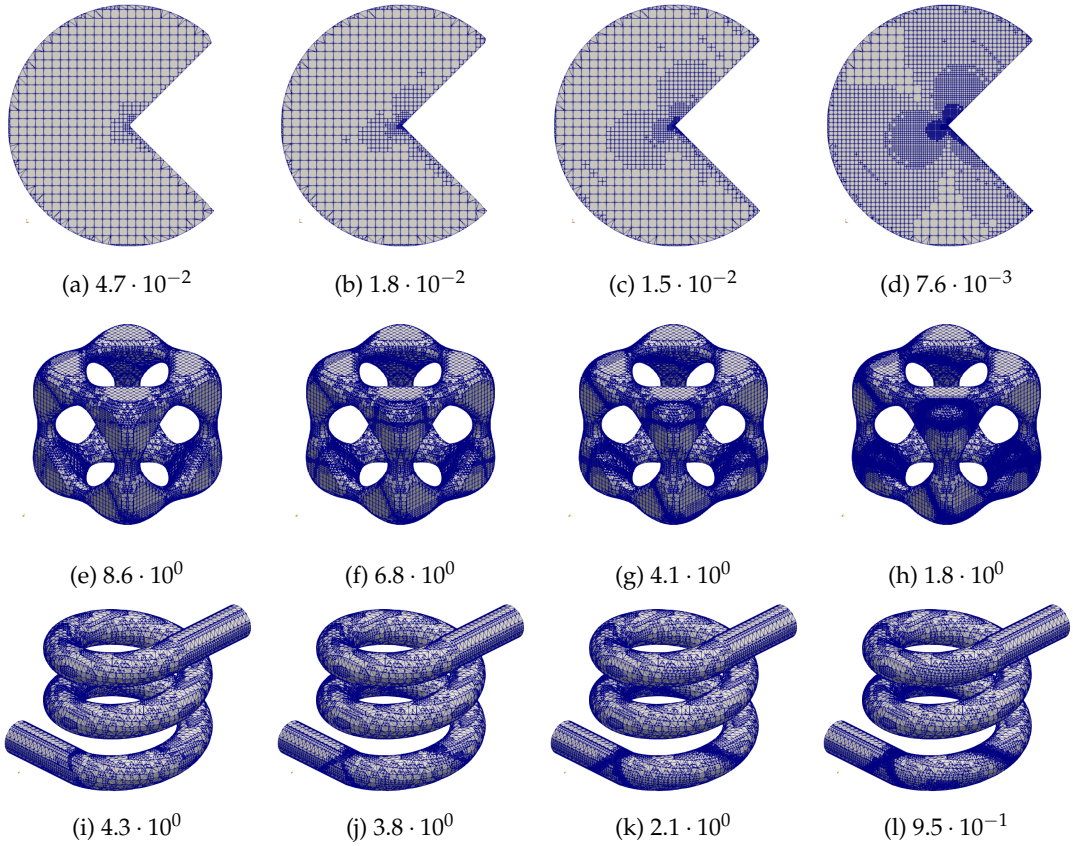
$$\left(\frac{\|e\|_{a,\gamma_i}}{\|u\|_{a,\gamma_i}}, N_{\text{dofs}}^{\gamma_i}\right),$$

that is, a point of the convergence test curve. Figure 3.12 depicts some meshes found with this iterative procedure, using the LB acceptability criterion.

Let us now start the discussion of numerical results obtained with convergence tests. The well-posedness threshold for aggregation $\eta_0$ (see Section 3.2.2) is prescribed to 0.25 in what follows. As shown in Figure 3.13, in serial environment, $h$-AgFEM behaviour consistently mirrors the one of std. $h$-FEM. This includes that (1) $h$-AgFEM always produces more optimal meshes, in terms of the error, than its non-adaptive version; and (2) optimal convergence rates are retained, even for the $h$-AgFEM Fichera problems, where convergence in the non-adaptive version is limited by regularity.

Although the ag. method is more accurate than its std. counterpart for the Fichera problems with uniform refinements, the usual behaviour is that they are very similar in terms of accuracy. Another outcome observed is that the LB criterion is clearly more cost-efficient, in terms of mesh size, than the OB one for both std. and ag. variants. This is also reported in [70] with std. $h$-FEM.

However, for some cases, e.g. Popcorn-Fichera, the sparse direct solver is not able to cope with the std. $h$-FEM 3D matrices, whereas it can robustly withstand the ones of AgFEM. In order to gain more insight into this matter, Figure 3.14 reports the estimated condition numbers, obtained by running `MATLAB` function `condest` with the system matrices associated with the points in the convergence tests. Some condition numbers



(a) $4.7 \cdot 10^{-2}$ (b) $1.8 \cdot 10^{-2}$ (c) $1.5 \cdot 10^{-2}$ (d) $7.6 \cdot 10^{-3}$

(e) $8.6 \cdot 10^{0}$ (f) $6.8 \cdot 10^{0}$ (g) $4.1 \cdot 10^{0}$ (h) $1.8 \cdot 10^{0}$

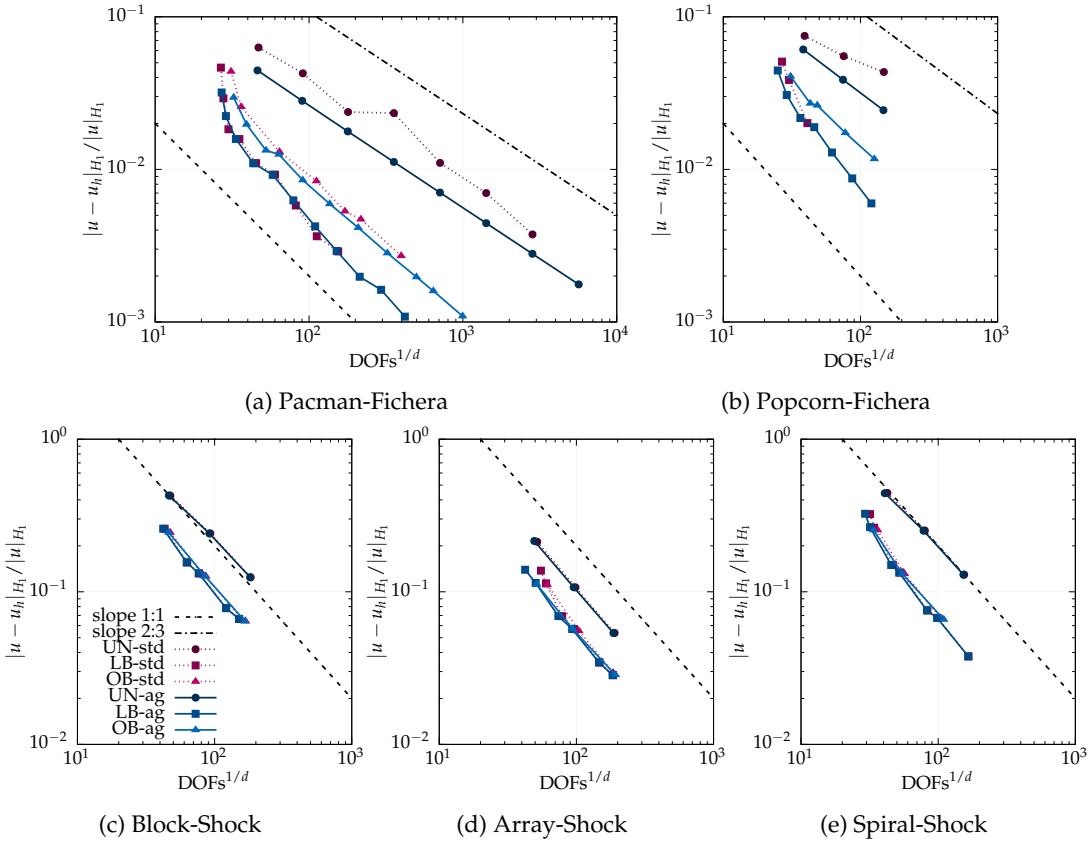
(i) $4.3 \cdot 10^{0}$ (j) $3.8 \cdot 10^{0}$ (k) $2.1 \cdot 10^{0}$ (l) $9.5 \cdot 10^{-1}$

Figure 3.12: Pacman-Fichera, hollow-shock and spiral-shock examples: sequence of optimal meshes obtained with the LB criterion for the convergence test. Relative energy norm of the error $\|u - u_h\|_a / \|u\|_a$ reported at each caption.

associated with the std. variant could not be computed, due to excessive memory demand during the LU factorization within `condest`. In spite of this, it can be clearly seen that matrices obtained with the ag. method are consistently better conditioned than the std. ones, especially in 3D. Besides, the condition number of std. FEM matrices scales erratically, whereas those associated with AgFEM approximately scale with $h_{\text{min}}^{-2}$, as expected.

In parallel tests, poor conditioning of std. FEM matrices frequently leads to failure of the `GAMG` solver. For these type of tests, we repeat the serial (single-threaded) procedure for a fixed number of threads (MPI tasks). We employ six MN-IV high-memory nodes and map each core to a different MPI task. Therefore, the experiments are launched in $6 \cdot 48 = 288$ processors. The partition of the mesh considers 288 subdomains and is defined to seek an equal distribution of the number of cells among processors. This is `p4est`'s default behaviour. As there is more memory available per core and the underlying (iterative) linear solver memory consumption scales optimally (linearly) with the number of DOFs, we expect the convergence tests to reach higher problem sizes than for the sequential experiments.

Indeed, parallel convergence test plots in Figure 3.15 gather more points than the serial ones and confirm all comments made in the sequential experiments, e.g. correct




(a) Pacman-Fichera (b) Popcorn-Fichera

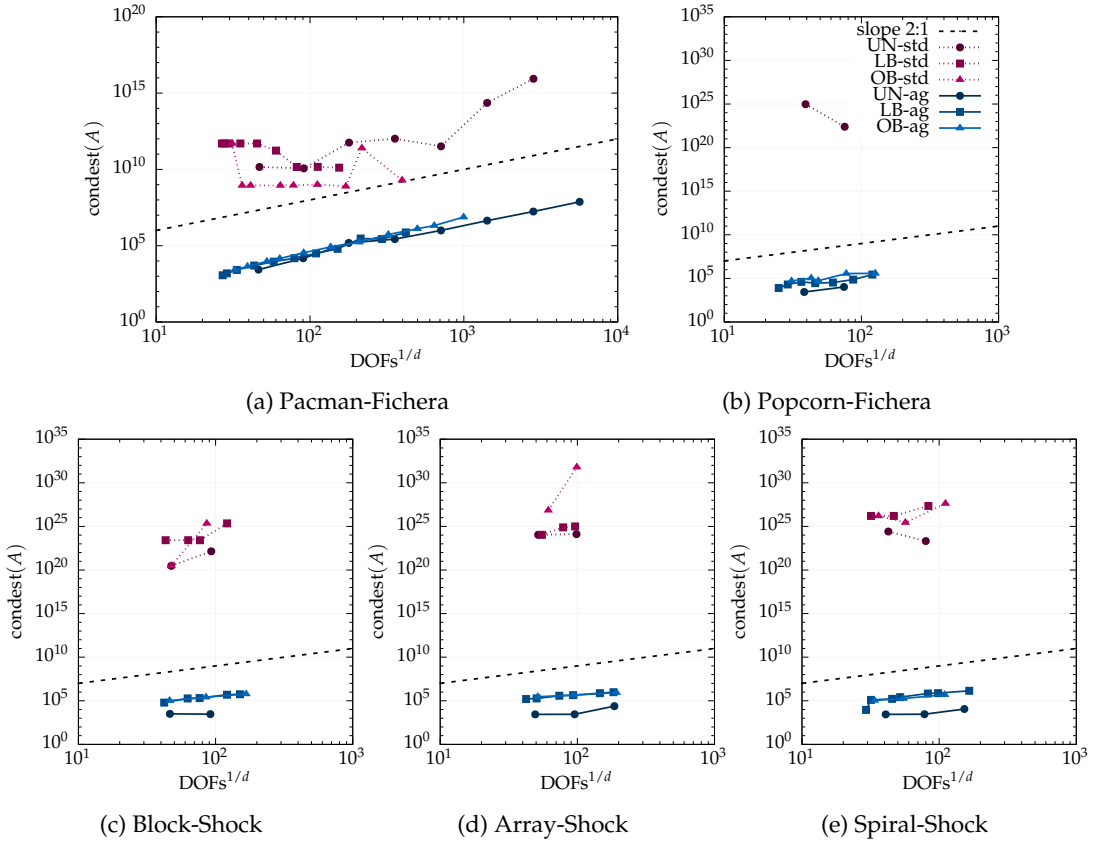
(c) Block-Shock (d) Array-Shock (e) Spiral-Shock

Figure 3.13: Convergence tests in serial environment.

rates of convergence or superiority of the LB criterion. As a result, the parallelization of $h$-AgFEM does not affect the algorithmic behaviour of the method.

However, as anticipated, the linear solver does not manage to generate a solution in many std. FE cases. Either the preconditioner cannot be generated or it is not positive definite (thus, incompatible with the conjugate gradient method). Both issues are directly related to the severe ill-conditioning of matrices obtained with the std. method (in Figure 3.14). On the other hand, when using the ag. method, `GAMG` is fully robust and converges towards the solution at the $10^{-9}$ tolerance.

Despite poor robustness of the solver with the std. method, available results in Figure 3.16 are enough to clearly identify higher growth rates in number of iterations for std. matrices, than for ag. ones. This exposes that, among the two methods, only $h$-AgFEM is potentially scalable, as the number of iterations mildly grows with the size of the problem; even for $h$-AgFEM points in Figure 3.16 with the largest number of DOFs (and condition number), convergence is declared in almost twenty iterations. We have verified that, in this context, the solver achieves single-digit reduction of the residual norm in 2-3 iterations, at most. Textbook multigrid efficiency is attained when the solver uses a modest number of point smoothing steps and convergence nearly advances at one digit in reduction of the residual norm per iteration [25]. We have checked



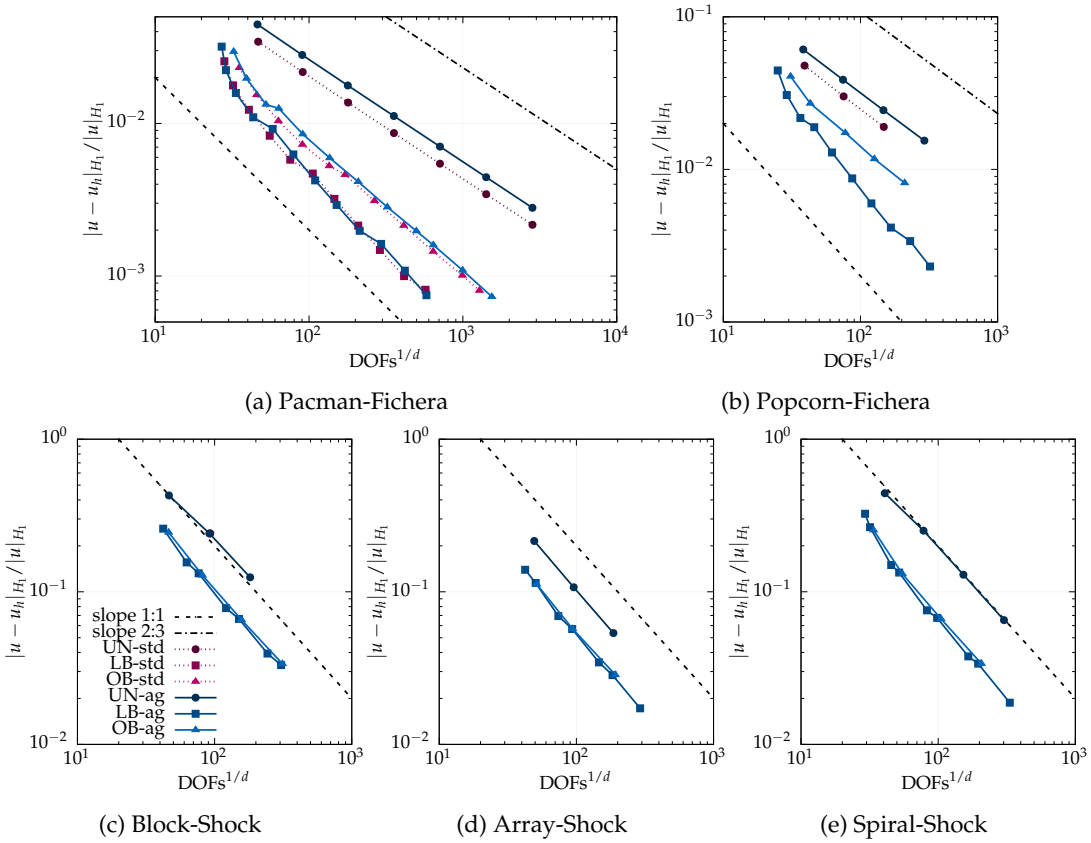

(a) Pacman-Fichera (b) Popcorn-Fichera

(c) Block-Shock (d) Array-Shock (e) Spiral-Shock

Figure 3.14: Condition number estimates of matrices associated with serial convergence tests.

the former is satisfied, by inspecting `PETSc` log data, whereas the latter is broadly fulfilled in AgFEM experiments. Therefore, `GAMG` on $h$-AgFEM matrices is not only robust, but also efficient.

A final experiment with convergence tests looks at the sensitivity of AgFEM to the well-posedness threshold $\eta_0$. As it is shown in Figure 3.17, low values of $\eta_0$ may not bypass the small cut-cell problem and hinder `GAMG` solvability, as shown in the 3D parallel examples. Conversely, high values of $\eta_0$ do not affect robustness, but increase solver iterations and ill-conditioning. They also reduce (local) accuracy. This is most likely an effect of excessive well-posed-to-ill-posed DOF extrapolation. As a result, optimal $\eta_0$ values may be found in the middle of the $[0, 1]$ range. This means that, while enforcing a minimum amount of aggregation is required to guarantee robustness, superfluous aggregation deteriorates solver efficiency. This effect is particularly prominent in $h$-adaptivity; setting $\eta_0 = 1$ on uniform meshes leads to decent results, as demonstrated in a previous work [187].




(a) Pacman-Fichera (b) Popcorn-Fichera

(c) Block-Shock (d) Array-Shock (e) Spiral-Shock

Figure 3.15: Convergence tests in parallel environment for 288 tasks.

### 3.5.5 Weak scaling

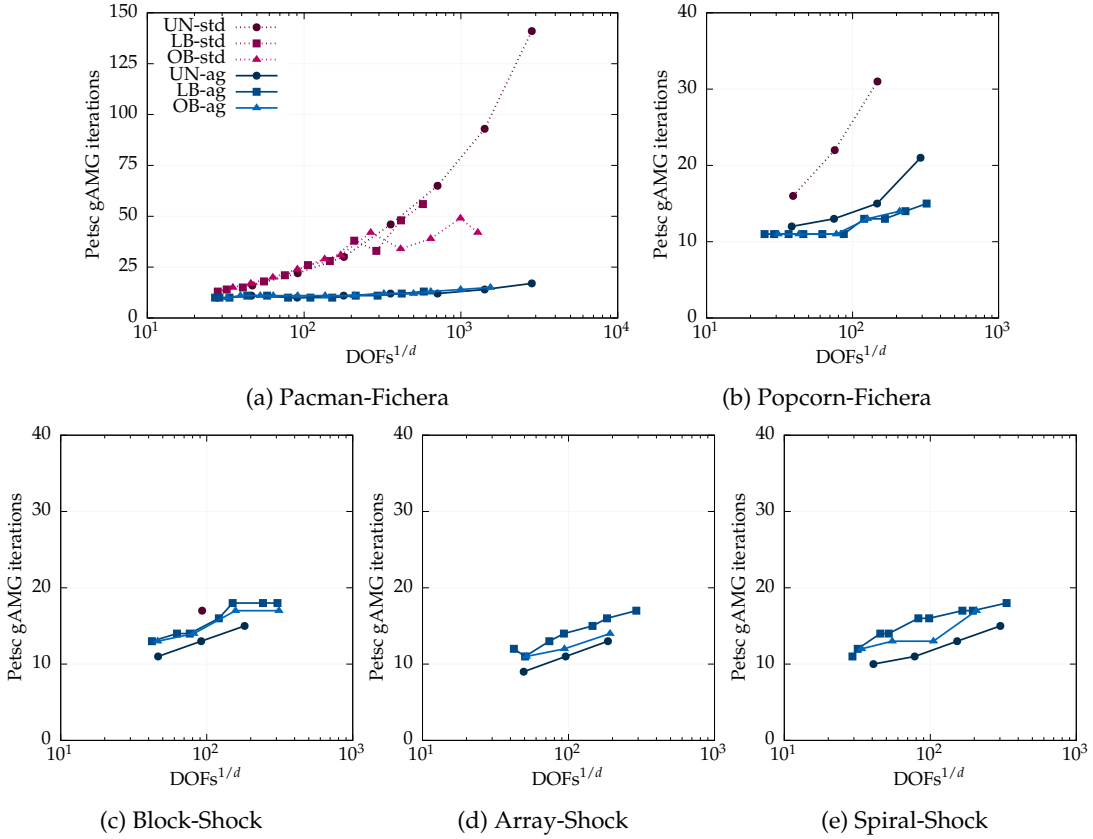
The starting point of weak scaling tests is the parallel convergence test setup of the previous section. As explained, a single convergence test case results in a set of pairs

$$\left\{ \left( \frac{\|e\|_{a,\gamma_i}}{\|u\|_{a,\gamma_i}}, N_{\text{dofs}}^{\gamma_i} \right) \right\}_{\gamma_i > 1},$$

associated with a finite sequence of decreasing target error values $\gamma_i$, $i > 1$. Each test corresponds to an individual curve, e.g. the LB-ag curve for the Pacman-Fichera test case in Figure 3.15(a). Other quantities can be extracted from the test, e.g. the size of the global triangulation $N_{\text{cells}}^{\gamma_i}$. In Section 3.5.4, each pair was obtained for a fixed number of processors $P = 288$. Naturally, as $N_{\text{cells}}^{\gamma_i}$ increases with $i$, so does the size of the local portion of the triangulation $n_{\text{cells}}^{\gamma_i}$, owned by each processor.

Given $\left\{ N_{\text{cells}}^{\gamma_i} \right\}_{i>1}$ associated with a convergence test, a weak scaling one can be derived by adjusting the number of processors $P^i$ for each $\gamma_i$, such that $n_{\text{cells}}^{\gamma_i}$ remains approximately constant for all $i > 1$. This can be achieved, by e.g. prescribing

$$P^i = P^1 \left\lfloor \frac{N_{\text{cells}}^{\gamma_i}}{N_{\text{cells}}^{\gamma_1}} \right\rfloor, \; i > 1,$$




(a) Pacman-Fichera (b) Popcorn-Fichera

(c) Block-Shock (d) Array-Shock (e) Spiral-Shock

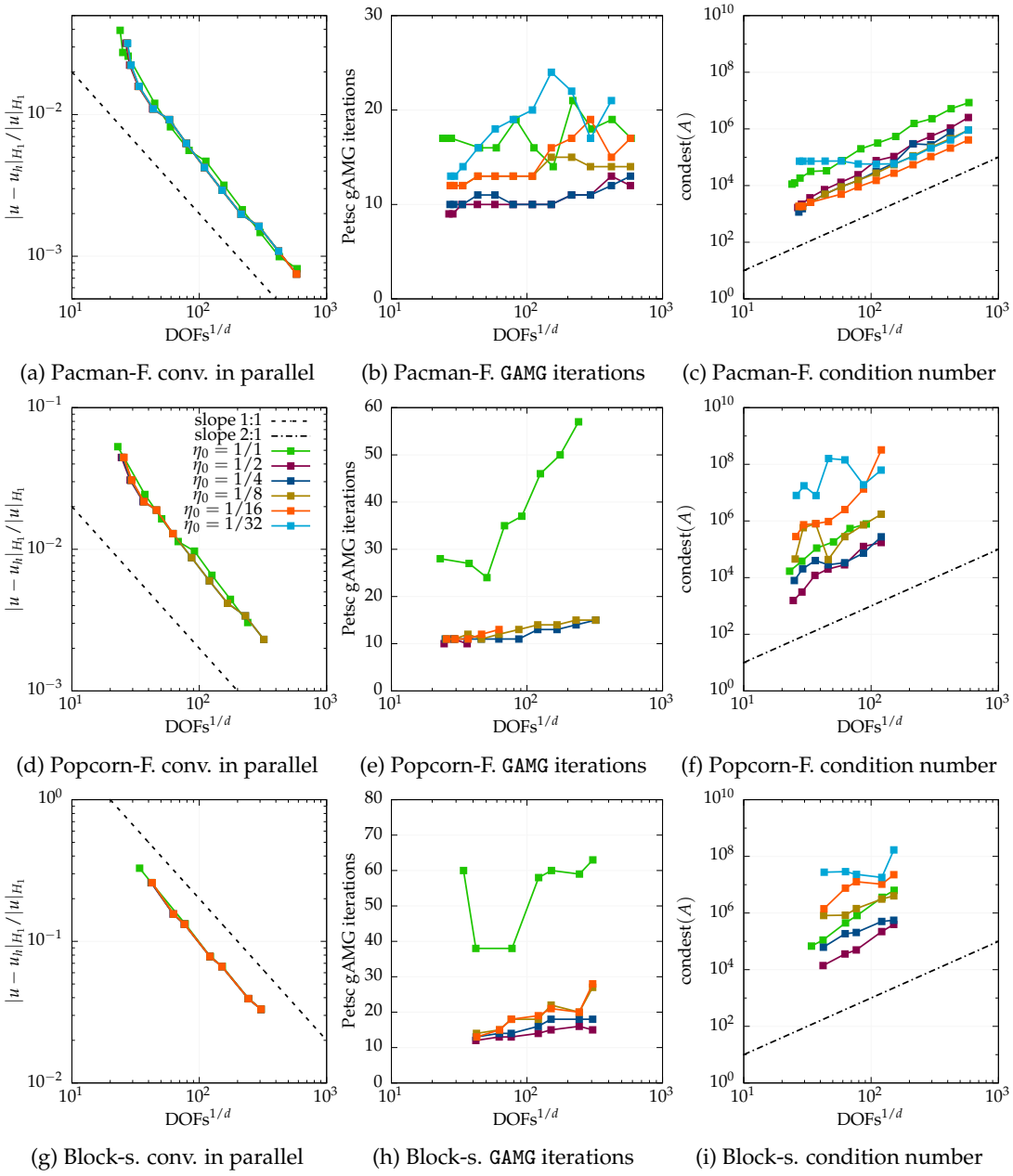
Figure 3.16: GAMG solver iterations in parallel environment for 288 number of tasks.

where $P^1$ is a fixed initial number of processors and $\lfloor \cdot \rfloor$ is the *floor* function; given a real number $x$, $\lfloor x \rfloor$ is the greatest integer less than or equal to $x$. From here, the weak scaling test consists merely in repeating the convergence test, taking $P^i$ processors for each $\gamma_i$. In this way, by keeping the local size of the mesh $n_{\text{cells}}^{\gamma_i}$ constant, we can straightforwardly study how $h$-AgFEM scales with global size of the problem.[4] Table 3.2 gathers the sequences $\left\{P^i\right\}_{i>1}$ obtained following this procedure for the two test cases that will be studied in this section, namely, the Popcorn-Fichera and Hollow-Shock problems for the AgFEM method with the LB remeshing criterion and $\eta_0 = 0.25$.

In weak scaling tests, we monitor wall clock times spent in the main phases of (i) the AgFEM method and (ii) the linear solver. We additionally get (iii) the number of GAMG solver iterations. As finding the optimal mesh for each $\gamma_i$, $i > 1$ is an iterative AMR process, we only report these quantities for the optimal mesh (last iteration). In the FE simulation loop, the starting control point is right after generating and partitioning the optimal mesh. From here, and following the order of the simulation pipeline, we report the time consumed in relevant AgFEM-related phases

1. parallel cell aggregation, i.e. generation of $R_S$ (Algorithm 3.2.2),

[4] We have checked that, using this approach, the local size of the problem (local number of DOFs) increases monotonically, though mildly, for $i > 1$. Thus, this conservative approach allows us to examine how the problem scales, avoiding cumbersome strategies to balance DOFs.




(a) Pacman-F. conv. in parallel (b) Pacman-F. GAMG iterations (c) Pacman-F. condition number

(d) Popcorn-F. conv. in parallel (e) Popcorn-F. GAMG iterations (f) Popcorn-F. condition number

(g) Block-s. conv. in parallel (h) Block-s. GAMG iterations (i) Block-s. condition number

Figure 3.17: $h$-AgFEM sensitivity to $\eta_0$ with the LB criterion. Recall that $\eta_0 = 0.25$ is the reference value in previous experiments (see Figures 3.13-3.16).

2. import data from root cells, i.e. import $\mathcal{T}_{\text{RG}}$ and $\mathcal{T}_{\text{CRG}}$ (Section 3.2.5),

3. setup of the distributed $\mathcal{V}^{\text{std}}$ space (Section 3.2.3), accounting for hanging DOF constraints,

4. setup of the distributed $\mathcal{V}^{\text{ag}}$ space on top of $\mathcal{V}^{\text{std}}$ (Sections 3.2.4 and 3.2.5), with mixed hanging and aggregation DOF constraints.

This is followed (and completed) by gathering the time spent in the linear solver setup and run stages, as well as the number of solver iterations needed to find the approximate solution to the problem on the optimal mesh. The convergence criterion is the

| Popcorn-Fichera LB-ag with $\eta_0 = 0.25$ and $n_{\text{cells}} \approx 15.5k$ | | | | | | | |
|---|---|---|---|---|---|---|---|
| $P$ | 2 | 8 | 19 | 52 | 132 | 349 | 883 |
| $N_{\text{cells}}$ | 31k | 130k | 301k | 800k | 2,025k | 5,354k | 13,553k |
| Hollow-shock LB-ag with $\eta_0 = 0.25$ and $n_{\text{cells}} \approx 21.0k$ | | | | | | | |
| $P$ | 6 | 17 | 29 | 107 | 194 | 790 | 1,484 |
| $N_{\text{cells}}$ | 126k | 369k | 612k | 2,261k | 4,083k | 16,662k | 31,221k |

Table 3.2: Number of subdomains and total cells in the background mesh for the cases considered in the weak scaling tests of Figure 3.18. For each case, local mesh size, given by $n_{\text{cells}}$, remains quasi-constant with the number of subdomains $P$.

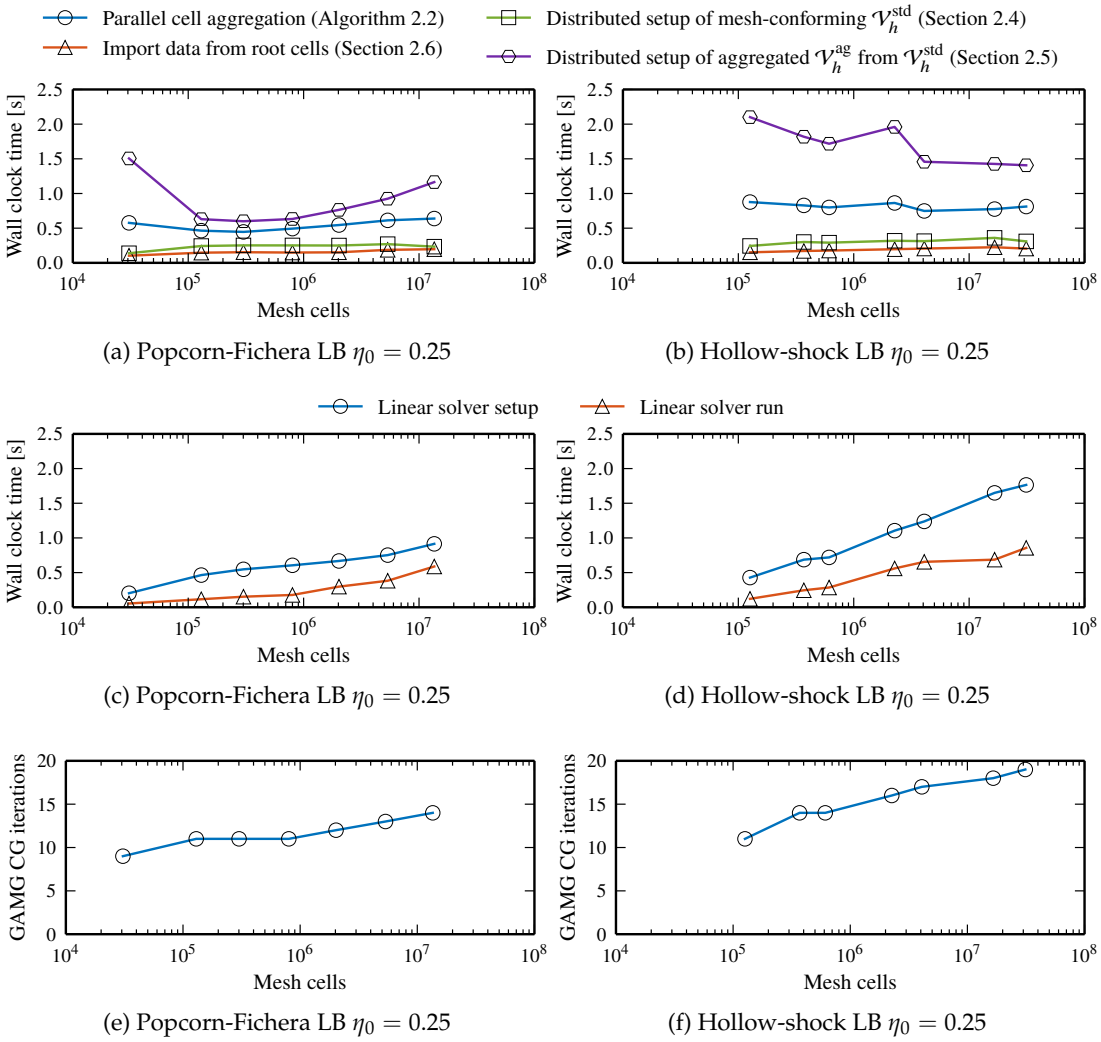
same as the one of the previous section, i.e. $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 < 10^{-9}$.

To allocate the MPI tasks in the MN-IV supercomputer, we resort to the default task placement policy of Intel MPI (v2018.4.057) with partially filled nodes. For each point of the test, the number of nodes $N^i$ is selected as $N^i = \lceil P^i/48 \rceil$, where $\lceil \cdot \rceil$ is the *ceiling* function; given a real number $x$, $\lceil x \rceil$ is the smallest integer more than or equal to $x$. If $P^i$ is not multiple of 48, the placement policy fully populates the first $N-1$ nodes with 48 MPI tasks per node; the remaining $P^i - 48(N-1)$ MPI tasks are mapped to the last node.

Figure 3.18 gathers all the quantities surveyed in weak scaling tests. All main phases of the *h*-AgFEM method exhibit remarkable scalability (Figures 3.18(a) and 3.18(b)). The results are also qualitatively similar for both geometries. Concerning solver performance in Figures 3.18(c)-3.18(f), although times and iterations do not scale as well as *h*-AgFEM-specific phases, results are still sound. Different system matrix conditioning could explain the slight differences between the two problems in solver performance. In any case, growth rate is mild, compared to growth of problem size. For instance, in the Hollow-shock example, total solver wall clock time (setup plus run) scales from 0.55 to 2.34 s, while the problem size scales from 126,232 to 16,619,828 cells. This means the total solver time increases by a factor of $4.3x$, whereas the problem size by a factor of $131.7x$. On the other hand, solver degradation is likely not fully attributed to *h*-AgFEM; see, e.g. the results in [187], showing that `GAMG` loses parallel efficiency even when dealing with body-fitted meshes.

## 3.6 Conclusions

In this chapter, we have introduced the aggregated finite element method on parallel adaptive tree-based meshes, referred to as *h*-AgFEM. The main difficulty is to establish how to combine hanging DOF constraints, arising from mesh non-conformity, with aggregation ones, which are needed to get rid of the small cut cell problem. We have followed a two-level strategy, grounded on building the aggregated FE space on top of an existing conforming FE space.




(a) Popcorn-Fichera LB $\eta_0 = 0.25$

(b) Hollow-shock LB $\eta_0 = 0.25$

(c) Popcorn-Fichera LB $\eta_0 = 0.25$

(d) Hollow-shock LB $\eta_0 = 0.25$

(e) Popcorn-Fichera LB $\eta_0 = 0.25$

(f) Hollow-shock LB $\eta_0 = 0.25$

Figure 3.18: AgFEM weak scaling tests up to 1,484 MPI tasks, as specified in Table 3.2.

As main contributions of the chapter, we have shown that (a) our approach allows one to define a unified AgFE space accounting for both type of constraints, without circular constraint dependencies; the key point is to mark as ill-posed DOFs, those without local support in a well-posed cell. We have also described how, (b) by carefully extending the layer of ghost cells, a distributed-memory version of $h$-AgFEM can be easily incorporated into existing large-scale FE codes. With numerical analysis and experimentation on the Poisson problem, we have studied the behaviour of $h$-AgFEM. It (c) enjoys the same benefits of standard $h$-FEM on body-fitted meshes. In particular, it restores optimal rates of convergence, implied by order of approximation alone, and it is amenable to standard mesh optimality criteria. Likewise, it also (d) inherits good properties from AgFEM on uniform meshes, above all robustness with respect to cut location, with low and well-behaved condition numbers. Finally, we have demonstrated (e) good parallel performance of a distributed-memory implementation of $h$-AgFEM; the main outcome is that it can efficiently exploit well-known AMG preconditioners

available in, e.g. PETSc.

We have successfully managed to bridge unfitted methods and parallel nonconforming tree-based meshes for the first time. *h*-AgFEM has the potential to grow and tackle large-scale multi-phase and multi-physics FE applications on arbitrarily complex geometries, aided by functional and geometrical error-driven mesh adaptation. As future work, it also remains to extend *h*-AgFEM to high-order FE approximations and, more generally, *hp*-adaptivity.

# Chapter 4

# The aggregated unfitted finite element method for interface elliptic problems

*The contents of this chapter correspond to the research publication*

[142] EN AND S. BADIA, *Robust and scalable h-adaptive aggregated unfitted finite elements for interface elliptic problems*, Submitted.

This chapter introduces a novel, fully robust and highly-scalable, *h*-adaptive aggregated unfitted finite element method for large-scale interface elliptic problems. The new method is grounded on a recent distributed-memory implementation of the aggregated finite element method on top of a highly-scalable Cartesian forest-of-tree mesh engine. It follows the classical approach of weakly coupling nonmatching discretisations at the interface to model internal discontinuities at the interface. We propose a natural extension of a single-domain parallel cell aggregation scheme to problems with a finite number of interfaces; it straightforwardly leads to aggregated finite element spaces that have the structure of a Cartesian product. We demonstrate, through standard numerical analysis and exhaustive numerical experimentation on several complex Poisson and linear elasticity benchmarks, that the new technique enjoys the following properties: well-posedness, robustness to cut location and material contrast, optimal (*h*-adaptive) approximation properties, high scalability and easy implementation in large-scale finite element codes. As a result, the method offers great potential as a useful finite element solver for large-scale multiphase and multiphysics problems modelled by partial differential equations.

## 4.1 Introduction

Unfitted FE methods are generating considerable interest in many practical situations. Their ability to handle complex geometries, avoiding cumbersome and time-consuming

body-fitted mesh generation, makes them especially appealing for large-scale simulations. They have been successfully exploited in multiphase and multiphysics applications with moving interfaces, such as fracture mechanics [177], fluid-structure interaction [21], and free surface flows [166], and in applications with varying domains, such as shape or topology optimisation [36], additive manufacturing [144], and stochastic geometry problems [14]. In the numerical community, unfitted FE methods receive different denominations. When the motivation is to capture (moving) interfaces, they are usually referred to as eXtended FE methods (XFEM) [30]. On the other hand, when the goal is to simulate a problem using a (usually simple) background mesh, they are denoted as *unfitted* or *embedded* or *immersed* techniques; see, e.g. the cutFEM method [35], the cutIGA method [74], the immersed boundary method [138] and the finite cell method [167].

This chapter investigates unfitted FE methods in *large scale* simulations of multiphysics problems modelled with PDEs. In particular, it centres upon interface problems. Two main approaches have been considered to model internal discontinuities across the unfitted interface, namely (1) weak coupling of nonmatching discretisations [88, 89] and (2) local partition-of-unity enrichments [135, 176], although both can lead to equivalent formulations [10]. This chapter focuses on the first approach. It broadly consists in dividing the mesh into two (sub)meshes that overlap in cut cells. It leads to FE approximations that have the structure of a Cartesian product. Transmission conditions on the unfitted interface are then weakly enforced by means of penalty [13] or Nitsche [146] formulations, among others.

In the context of unfitted interface methods, the main challenge is to derive robust methods for large material contrast across the interface. Indeed, naive variational formulations may exhibit poor stability in this regime, e.g. average numerical flux weighting in Nitsche methods produces inaccurate and oscillating approximation of interface quantities [9]. On the other hand, large material contrast problems are prone to the so-called *small cut cell problem*. This issue is formally circumscribed to the unfitted boundary case and it is associated with cut cells with arbitrarily small intersection with the physical domain. Unless a specific technique mitigates the problem, numerical integration on these badly-cut cells leads to severe ill-conditioning problems [24, 62]. Since unfitted boundary problems can be interpreted as a limiting case of large contrast interface ones, the latter are not completely immune to the issue.

Despite vast literature on the topic [87, 110, 115, 117], fewer authors achieve formulations that are fully robust and optimal, regardless of cut location and material contrast. A notable exception is the family of methods that rely on *ghost penalty* [35, 37, 38, 86]. These works adopt approach (1) and enrich the variational formulation with suitable stabilization terms defined in the faces of cut cells; the resulting formulation is robust to cut location. Besides, robustness to material contrast is achieved by using the so-called harmonic weights in the Nitsche formulation, a typical approach in body-fitted DG methods [57]. As a result, the condition number of the diagonally-scaled

system matrix becomes independent of the material contrast [35]. However, research in this area has tended to overlook scalability and *hp*-adaptivity, which are essential aspects in applications to large-scale multiphysics problems. These aspects have been considered by the finite cell method community [75, 163], but robustness to material contrast has barely received their attention.

Research over the past few years is turning to an alternative approach to ensure robustness to cut location, the so-called *cell aggregation* or *cell agglomeration* techniques. This approach is based on aggregating cells at the cut interface to remove basis functions associated with small cut cells. It has been recently employed for hybrid-high order (HHO) [34], even though these methods are not considering face aggregation strategies and thus, their trace unknowns can lead to ill-posed problems. Aggregation has also been used for CG [96] methods, but the resulting scheme relies on the assumption that the aggregates can always be rectangles. However, such assumption is wrong, even in two-dimensions; aggregates have more complicated shapes in general geometries and meshes. The authors in [96] picked an elementary 2D circular Poisson problem in a square with a circular inclusion, discretised with a uniform Cartesian grid, in a mesh in which rectangular aggregates only where possible. In contrast, the (CG) aggregated unfitted FEM, referred to as AgFEM [24], is grounded on a general aggregated approach amenable to arbitrarily complex 3D geometries and *h*-adaptivity, see Chapter 3. In spite of this, research has been restricted so far to unfitted boundary elliptic [24] or Stokes [20] problems.

The main goal of this chapter is to present a novel aggregated FE method for interface elliptic BVPs. In contrast with other existing methods, we clearly show that interface AgFEM enjoys *overall* well-behaved numerical properties and remarkable large-scale capability. In particular, we demonstrate, with theoretical results and thorough numerical experimentation, well-posedness, robustness to cut location and material contrast, optimal (*h*-adaptive) approximation properties, high scalability and ease of implementation in HPC FE codes. The chapter gives first insight into AgFEM, as a large-scale FE solver for complex multiphase and multiphysics problems modelled by PDEs. It is also intended to provide guidance in exploiting other unfitted CG methods by aggregation for interface problems.

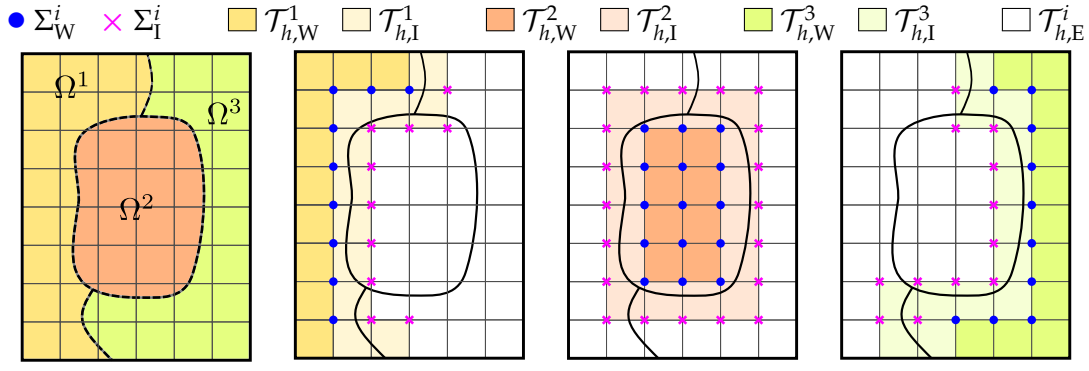
The outline of this chapter is as follows. We assume first an embedded (multiple) *n*-interface geometrical setting in Section 4.2.1. Next, we introduce a natural extension of the single-domain cell aggregation method in [24] to *n*-interface problems, in Section 4.2.2. Cell aggregation can be carried out independently on each subdomain and reuse, with little effort, existing distributed-memory implementations of the single-domain algorithm [187]. In Section 4.2.3, we define AgFE spaces for embedded *n*-interfaces; we see that they easily accommodate the interface-overlapping mesh approach in [88]. Afterwards, we restrict ourselves to the approximation of single interface linear elasticity problems, see Section 4.3.1. We derive a similar formulation to

body-fitted DG methods [11], using the symmetric interior penalty method and harmonic average weights, to weakly enforce interface conditions, see Section 4.3.2. Numerical analysis, proving well-posedness and a priori error estimates, are also covered there; all results are stable to cut location and material contrast. We implement the method in the large-scale FE software package `FEMPAR` [19], which exploits the highly-scalable forest-of-tree mesh engine `p4est` [43] for $h$-adaptivity. In the numerical tests of Section 4.4, we consider both the linear elasticity and Poisson equations as model problems on several complex geometries and several $hp$-FEM standard benchmarks. We numerically assess optimal convergence rates on uniform and $h$-adaptive meshes, robustness to cut location and material contrast, and weak-scalability. Finally, we report the main conclusions and contributions of the chapter in Section 4.5.

## 4.2 The aggregated unfitted finite element method on interface problems

### 4.2.1 Embedded interface geometry setup

Let $\Omega \subset \mathbb{R}^d$, with $d = 2, 3$ denoting the space dimension, be an open, bounded, connected domain, with smooth Lipschitz boundary $\partial\Omega$. Since we seek to analyse problems with multiple physics and/or phases, let $\{\Omega^i\}_{i=1}^N$ be a partition of $\Omega$ into $N$ subdomains $\Omega^i$ with Lipschitz boundaries $\partial\Omega^i$. Let now $\Gamma_0 \doteq \bigcup_{i=1}^N \partial\Omega^i \setminus \partial\Omega$ denote the skeleton of the partition. Equivalently, there is a partition of $\Gamma_0$ into $\Gamma^{ij} \doteq \partial\Omega^i \cap \partial\Omega^j$, such that $\Gamma_0 \doteq \bigcup_{i,j=1}^N \Gamma^{ij}$. Let $N_0$ denote the number of non-empty $\Gamma^{ij}$, for all $i, j = 1, \dots, N$. The setting is represented in Figure 4.1(a).




(a) $\Omega$, $\mathcal{T}_h$ and (dashed) $\Gamma_0$ (b) $\mathcal{T}^1_{h,\mathrm{act}}$ and $\mathcal{V}^1_{h,\mathrm{act}}$ (c) $\mathcal{T}^2_{h,\mathrm{act}}$ and $\mathcal{V}^2_{h,\mathrm{act}}$ (d) $\mathcal{T}^3_{h,\mathrm{act}}$ and $\mathcal{V}^3_{h,\mathrm{act}}$

Figure 4.1: An embedded interface geometry setup for $N = 3$ and $\eta_0 = 1$, i.e. well-posed iff interior and ill-posed iff cut. The boundary of the physical domain $\partial\Omega$ conforms to the mesh $\mathcal{T}_h$, whereas the skeleton $\Gamma_0$ is immersed in it. $\{\mathcal{T}^i_{h,\mathrm{act}}\}_{i=1}^3$ forms a partition of $\mathcal{T}_h$, *overlapping* at cells cut by the skeleton $\Gamma_0$. As a result, degrees of freedom on cut cells are doubled or tripled (assuming linear lagrangian FEs). We consider partitions of DOFs in $\mathcal{V}^i_{h,\mathrm{act}}$ into well-posed $\Sigma^i_{\mathrm{W}}$ and ill-posed $\Sigma^i_{\mathrm{I}}$ DOFs (note that we omit Dirichlet DOFs). Ill-posed DOFs are constrained in terms of well-posed DOFs, see Equation (4.1) and Figure 4.3.

We introduce now a typical embedded interface setup. To focus on the interface problem, we assume that $\Omega$ can be easily meshed with, e.g. Cartesian grids or unstructured $d$-simplexes, such that *the external boundary* $\partial\Omega$ *conforms to the mesh, whereas* $\Gamma_0$ *remains immersed*, as shown in Figure 4.1(a). For simplicity in the exposition, let us consider that the mesh is body-fitted with respect to $\partial\Omega$, even though the general case can readily be tackled using the techniques in [24]. Instead, in this article, we focus on the extension of these techniques to resolve immersed boundaries. According to this, let $\mathcal{T}_h$ be a partition of $\Omega$ into cells, the so-called *background* mesh. Any $T \in \mathcal{T}_h$ is the image of a differentiable homeomorphism $\Phi_T$ over a set of admissible open reference $d$-polytopes [19], such as $d$-simplexes or $d$-cubes. We let $\mathcal{T}_h$ be *non-conforming*, i.e. there can be hanging vertices, edges or faces. We assume that the mesh is *shape-regular* and $h_T$ represents the characteristic size of the cell $T \in \mathcal{T}_h$.

We introduce now a typical embedded interface setup. To focus on the interface problem, we assume that $\Omega$ can be easily meshed with, e.g. Cartesian grids or unstructured $d$-simplexes, such that *the external boundary* $\partial\Omega$ *conforms to the mesh, whereas* $\Gamma_0$ *remains immersed*, as shown in Figure 4.1(a). For simplicity in the exposition, let us consider that the mesh is body-fitted with respect to $\partial\Omega$, even though the general case can readily be tackled using the techniques in [24]. Here, instead, we focus on the extension of these techniques to resolve immersed interfaces. According to this, let $\mathcal{T}_h$ be a partition of $\Omega$ into cells, the so-called *background* mesh. Any $T \in \mathcal{T}_h$ is the image of a differentiable homeomorphism $\Phi_T$ over a set of admissible open reference $d$-polytopes [19], such as $d$-simplexes or $d$-cubes. We let $\mathcal{T}_h$ be *non-conforming*, i.e. there can be hanging vertices, edges or faces. We assume that the mesh is *shape-regular* and $h_T$ represents the characteristic size of the cell $T \in \mathcal{T}_h$.

We assume, without loss of generality, that the immersed skeleton $\Gamma_0$ is represented by the zero level-set of one or several known scalar functions, the so-called level-set functions, or by other means, e.g. from 3D CAD data, using techniques to compute the intersection between cell edges and surfaces (see, e.g. [129]). We also assume that we have suitable techniques (e.g. for local integration) to deal with cells that are intersected by more than one interface $\Gamma^{ij}$. For any cell $T \in \mathcal{T}_h$, we define the quantity

$$\eta_T^i \doteq \frac{\text{meas}_d(T \cap \Omega^i)}{\text{meas}_d(T)}, \quad \eta_T \in [0,1], \quad i = 1, \dots, N,$$

and a user-defined parameter $\eta_0 \in (0,1]$, referred to as the *well-posedness threshold*. To isolate badly cut cells, we classify cells of $\mathcal{T}_h$ in terms of $\eta_T^i$ and $\eta_0$; it leads to subsets of $\mathcal{T}_h$ of the form

$$\mathcal{T}_{h,\text{W}}^i = \{T \in \mathcal{T}_h : \eta_T^i \geq \eta_0\}, \;\; \mathcal{T}_{h,\text{I}}^i = \{T \in \mathcal{T}_h : \eta_0 > \eta_T^i > 0\}, \;\; \mathcal{T}_{h,\text{E}}^i = \{T \in \mathcal{T}_h : \eta_T^i = 0\},$$

for $i = 1, \dots, N$. $\mathcal{T}_{h,\text{W}}^i$, $\mathcal{T}_{h,\text{I}}^i$ and $\mathcal{T}_{h,\text{E}}^i$ are the well-posed (W), ill-posed (I) and exterior cells (E) associated with subdomain $\Omega^i$. $\mathcal{T}_{h,\text{W}}^i$ contains interior cells or those with a

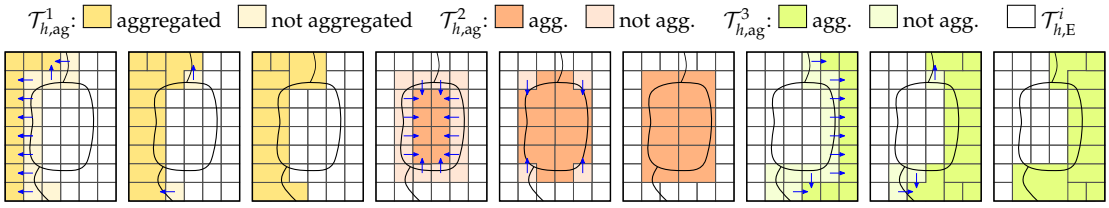
large portion inside $\Omega^i$, $\mathcal{T}^i_{h,\mathrm{I}}$, those with small cut portions in $\Omega^i$, and $\mathcal{T}^i_{h,\mathrm{E}}$ those with empty intersection with $\Omega^i$. We remark that, for $\eta_0 = 1$, well- or ill-posed cells coincide with interior or cut cells. By definition, each triplet $\{\mathcal{T}^i_{h,\mathrm{W}}, \mathcal{T}^i_{h,\mathrm{I}}, \mathcal{T}^i_{h,\mathrm{E}}\}$, $i = 1, \dots, N$, forms a nonoverlapping partition of $\mathcal{T}_h$. We denote the union of cells of $\mathcal{T}^i_{h,\mathrm{W}}$, $\mathcal{T}^i_{h,\mathrm{I}}$ and $\mathcal{T}^i_{h,\mathrm{E}}$ by $\Omega^i_{\mathrm{W}}$, $\Omega^i_{\mathrm{I}}$ and $\Omega^i_{\mathrm{E}}$, e.g. $\Omega^i_{\mathrm{W}} = \bigcup_{T \in \mathcal{T}^i_{h,\mathrm{W}}} \overline{T}$. We also introduce the *active* meshes and domains, given by $\mathcal{T}^i_{h,\mathrm{act}} \doteq \mathcal{T}^i_{h,\mathrm{W}} \cup \mathcal{T}^i_{h,\mathrm{I}}$ and $\Omega^i_{\mathrm{act}} \doteq \Omega^i_{\mathrm{W}} \cup \Omega^i_{\mathrm{I}}$, $i = 1, \dots, N$; note that $\Omega^i \subset \Omega^i_{\mathrm{act}}$. It follows that $\{\mathcal{T}^i_{h,\mathrm{act}}\}_{i=1}^N$ is a partition of $\mathcal{T}_h$, *overlapping* at cells cut by the skeleton $\Gamma_0$, see Figures 4.1(b)-4.1(c)-4.1(d). We observe that our geometrical configuration generalises to multiple interfaces the classical approach adopted in, e.g. [35, 88], for single interface problems. Indeed, for $N = 2$ ($N_0 = 1$), $\{\mathcal{T}^i_{h,\mathrm{act}}\}_{i=1}^2$ is an overlapping partition of $\mathcal{T}_h$, that divides the mesh into two (sub)meshes, where cells cut by the interface are doubled.

Let us note that, for general expressions of the immersed skeleton $\Gamma_0$, it could happen that a cell $T \in \mathcal{T}_h$ is such that $T \cap \Omega$ is a set of connected domains that are disconnected with each other. In this case, we consider each of these disconnected components as a separate cut cell. Thus, we redefine the mesh $\mathcal{T}_h$ replicating these cut cells for each disconnected part, and use the procedure above verbatim. The reason for this is to assure that aggregates are always connected domains and, e.g. the Deny-Lions lemma can be used to prove approximability properties. Alternatively, one could assume that $\Gamma$ has bounded curvature and the mesh is *fine enough* (see, e.g. [88]).

### 4.2.2 Cell aggregation with multiple interfaces

Cell aggregation for single-domain problems is well-covered in previous works [24]; here, we limit ourselves to lay out the extension of the rationale to problems posed in domains with multiple interfaces, introduced in Section 4.2.1. We recall that aggregated FE spaces are grounded on a map, the so-called *root cell map*. This map associates any ill-posed cell with a well-posed cell, by means of a cell aggregation scheme, described in Algorithm 3.2.2.

In our context, we assume we carry out cell aggregation independently on each active mesh $\mathcal{T}^i_{h,\mathrm{act}}$, $i = 1, \dots, N$, as illustrated in Figure 4.2; it yields the $i$-root cell maps $R^i : \mathcal{T}^i_{h,\mathrm{act}} \to \mathcal{T}^i_{h,\mathrm{W}}$. For any $T \in \mathcal{T}^i_{h,\mathrm{W}}$, we refer to $A^i_T \doteq (R^i)^{-1}(T)$ as a cell aggregate rooted at $T$. By construction of $R^i$, aggregates take the form $A^i_T = \{T_j\}_{0 \le j \le m_T}$, where $T_0 = T \in \mathcal{T}^i_{h,\mathrm{W}}$ and $T_j \in \mathcal{T}^i_{h,\mathrm{I}}$, $1 \le j \le m_T$, i.e. they are composed of several ill-posed cells and a unique (root) well-posed cell. Furthermore, aggregates are connected; they are also disjoint in $\mathcal{T}^i_{h,\mathrm{act}}$, i.e. for any $T, T' \in \mathcal{T}^i_{h,\mathrm{act}}$, we have that $A_{R^i(T)} \cap A_{R^i(T')} = \emptyset$ or $R^i(T) \equiv R^i(T')$. It follows that $\mathcal{T}^i_{h,\mathrm{ag}} \doteq \{A^i_T\}_{T \in \mathcal{T}^i_{h,\mathrm{W}}}$ are partitions of $\mathcal{T}^i_{h,\mathrm{act}}$ into cell aggregates, for all $i = 1, \dots, N$. We observe that cell aggregation schemes only use the local information of each $\mathcal{T}^i_{h,\mathrm{act}}$, $i = 1, \dots, N$; there is no coupling between active (sub)meshes. As a result, implementation of a multiple-domain cell aggregation scheme can fully reuse a single-domain counterpart.




(a) Step 1 (b) Step 2 (c) Step 3 (d) Step 1 (e) Step 2 (f) Step 3 (g) Step 1 (h) Step 2 (i) Step 3

Figure 4.2: Cell aggregation on the three active meshes $\mathcal{T}^i_{h,\text{act}}$, $i = 1,2,3$, of Figure 4.1. The algorithm is detailed in Algorithm 3.2.2. First, it marks well-posed cells as individual aggregates (Step 1). Then, aggregates grow iteratively, by attaching adjacent ill-posed cells to them (Step 2). The procedure stops when $\mathcal{T}^i_{h,\text{act}}$, $i = 1,2,3$ is covered by aggregates (Step 3). This operation gives the root cell map $R^i$, $i = 1,2,3$. We observe that the scheme runs independently on each mesh with the local information provided by $\mathcal{T}^i_{h,\text{act}}$, $i = 1,2,3$. Hence, implementation can reuse a single-domain cell aggregation scheme.

### 4.2.3 Aggregated Lagrangian finite element spaces

As stated in Section 4.1, we consider the common approach [9, 35, 88] of building FE spaces on top of interface-overlapping meshes; it leads to FE approximations that have a Cartesian product structure. In our case, we aim to construct a CG AgFE space on top of the aggregated overlapping mesh $\{\mathcal{T}^i_{h,\text{ag}}\}_{i=1}^N$. We will see that we can straightforwardly exploit the single-domain methodology in [24] to derive an AgFE space on each aggregated mesh $\mathcal{T}^i_{h,\text{ag}}$ and, from here, a global FE space in $\mathcal{T}_h$ of the form $\mathcal{V}^1_{h,\text{ag}} \times \ldots \times \mathcal{V}^N_{h,\text{ag}}$.

For the sake of simplicity, we assume that the PDE problem posed in $\Omega$ is such that there is a single scalar-valued field associated to each subdomain $\Omega^i$. We also assume discretisations with Lagrangian FEs. In any case, the exposition can be generalised to other FEs, e.g. Nédélec [148], vector/tensor fields and multiple fields per $\Omega^i$. We also consider same cell topology everywhere in $\mathcal{T}_h$ and $\mathcal{T}_h$ conforming; although AgFE spaces on top of nonconforming meshes are fully covered in Chapter 3.2.2 and numerical tests in Section 4.4 run on Cartesian tree-based (nonconforming) meshes (cf. Chapter 2). Lastly, we omit treatment of strong Dirichlet boundary conditions in the discussion below, although they can be easily taken care of, using standard approaches.

We denote by $\mathcal{V}(T)$ a vector space of functions defined on $T \in \mathcal{T}_h$. For $d$-simplex meshes, we define the local space $\mathcal{V}(T) \doteq \mathcal{P}_q(T)$, i.e. the space of polynomials of order less or equal to $q$ in the variables $x_1, \ldots, x_d$. For $d$-cubes, we define $\mathcal{V}(T) \doteq \mathcal{Q}_q(T)$, i.e. the space of polynomials that are of degree less or equal to $q$ with respect to each variable in $x_1, \ldots, x_d$. In the numerical examples, we limit ourselves to rectangular or hexahedral cells and linear or quadratic shape functions, i.e. $\mathcal{V}(T) \doteq \mathcal{Q}_1(T)$ or $\mathcal{V}(T) \doteq \mathcal{Q}_2(T)$. To simplify notation, we define the elemental functional spaces $\mathcal{V}(T)$ in the physical cell $T \subset \Omega$ (even though our computer implementation relies on reference parametric spaces, as usual). Since we take on Lagrangian FEs, the basis for $\mathcal{V}(T)$ is the Lagrangian basis (of order $q$) on $T$; we assume same order everywhere in $\mathcal{T}_h$. We

denote by $\Sigma_T$ the set of Lagrangian nodes of order $q$ of cell $T$, i.e. the set of local DOFs in $T$. There is a one-to-one mapping between nodes $\sigma \in \Sigma_T$ and shape functions $\phi_T^\sigma(\boldsymbol{x})$ such that $\phi_T^\sigma(\boldsymbol{x}^{\sigma'}) = \delta_{\sigma\sigma'}$, where $\boldsymbol{x}^{\sigma'}$ are the space coordinates of node $\sigma'$ and $\delta$ is the Kronecker delta.

Since we seek a global aggregated FE space of the form $\mathcal{V}_{h,\text{ag}}^1 \times \ldots \times \mathcal{V}_{h,\text{ag}}^N$, we start by defining the subdomain members $\mathcal{V}_{h,\text{ag}}^i$, $i = 1, \ldots, N$. Thus, all notation and definitions in the next paragraphs are subdomain-local, i.e. referred to any subdomain $\Omega^i \subset \Omega$, $i = 1, \ldots, N$, unless stated otherwise. According to this, let $\Sigma_{\text{act}}^i$ refer to the set of (subdomain-)active DOFs of $\mathcal{T}_{h,\text{act}}^i$. We introduce next a local-to-subdomain DOF map $\sigma^i(T, \sigma') \in \Sigma_{\text{act}}^i$, with $\sigma' \in \Sigma_T$ and $T \in \mathcal{T}_h$. In CG methods, $\sigma^i$ is obtained by gluing together DOFs located in the same geometrical position; this operation leads to $\mathrm{C}^0$-continuous approximations. With this notation, we can define a standard FE space in $\mathcal{T}_{h,\text{act}}^i$ of the form

$$\mathcal{V}_{h,\text{act}}^i \doteq \{v^i \in \mathcal{C}^0(\Omega_{\text{act}}^i) : v^i\Big|_T \in \mathcal{V}(T),\ \forall\ T \in \mathcal{T}_{h,\text{act}}^i\}.$$

It is well-known that, when the discrete FE problem is *only* integrated in $\Omega^i$, *direct* usage of $\mathcal{V}_{h,\text{act}}^i$ leads to arbitrarily ill-conditioned linear systems [62]. To solve this issue, we resort to the aggregated FEM [20, 24]. The main idea is to remove from $\mathcal{V}_{h,\text{act}}^i$ problematic DOFs, associated with small cut cells, by constraining them as a linear combination of DOFs with local support in a (well-posed) cell of $\mathcal{T}_{h,\text{W}}^i$. It leads to the aggregated subspace of $\mathcal{V}_{h,\text{act}}^i$, namely $\mathcal{V}_{h,\text{ag}}^i$, that gets rid of the aforementioned ill-conditioning issues.

In order to define $\mathcal{V}_{h,\text{ag}}^i$, the key is to realise that our context is analogous to one considering a single-domain unfitted-boundary problem, taking $\Omega^i$ as the physical domain embedded in $\Omega$. The former case is extensively covered in [24]. Hence, we can follow the same steps to derive $\mathcal{V}_{h,\text{ag}}^i$. According to this, let us define the set of well-posed DOFs as $\Sigma_{\text{W}}^i \doteq \bigcup_{T \in \mathcal{T}_{h,\text{W}}^i} \Sigma_T$ and the set of ill-posed DOFs as $\Sigma_{\text{I}}^i \doteq \Sigma_{\text{act}}^i \setminus \Sigma_{\text{W}}^i$, see Figure 4.1. Obviously, $\{\Sigma_{\text{W}}^i, \Sigma_{\text{I}}^i\}$ forms a partition of $\Sigma_{\text{act}}^i$. $\Sigma_{\text{W}}^i$ gathers all DOFs that have local support in (well-posed) cells of $\mathcal{T}_{h,\text{W}}^i$, while $\Sigma_{\text{I}}^i$ isolates all DOFs, that potentially have arbitrarily small compact support and must be constrained in terms of well-posed DOFs of $\Sigma_{\text{W}}^i$.

To compute ill-posed DOF constraints, we proceed as usual in AgFE methods. First, we compose the root cell map $R^i : \mathcal{T}_{h,\text{act}}^i \to \mathcal{T}_{h,\text{W}}^i$ of Section 4.2.2, with a map between ill-posed DOFs $\Sigma_{\text{I}}^i$ and ill-posed cells $\mathcal{T}_{h,\text{I}}^i$. Specifically, we assign each ill-posed DOF to one of its surrounding ill-posed cells. The chosen cell is then mapped onto a well-posed cell via $R^i$. Thus, the outcome of this composition is a map $K^i : \Sigma_{\text{I}}^i \to \mathcal{T}_{h,\text{W}}^i$, that assigns an ill-posed DOF to a well-posed cell via cell aggregation; see formal definitions in, e.g. [24, 187]. Following this, given $v^i \in \mathcal{V}_{h,\text{act}}^i$ and $\sigma \in \Sigma_{\text{I}}^i$, we linearly extrapolate the nodal value of $\sigma$, namely $v_\sigma^i \in \mathbb{R}$, with the values at the local DOFs of its root cell $K^i(\sigma)$.

It leads to the constraint (see Figure 4.3)

$$v^i_\sigma = \sum_{\sigma' \in \Sigma_{K^i(\sigma)}} C_{\sigma\sigma'} v^i_{\sigma'}, \quad \text{with } C_{\sigma\sigma'} \doteq \phi^{\sigma'}_{K^i(\sigma)}(\boldsymbol{x}^\sigma). \tag{4.1}$$

As a result, the AgFE space can be readily defined as

$$\mathcal{V}^i_{h,\text{ag}} \doteq \{v^i \in \mathcal{V}^i_{h,\text{act}} : v^i_\sigma = \sum_{\sigma' \in \Sigma_{K^i(\sigma)}} C_{\sigma\sigma'} v^i_{\sigma'}, \ \forall \sigma \in \Sigma^i_{\text{I}}\}.$$

It is clear that $\mathcal{V}^i_{h,\text{ag}} \subset \mathcal{V}^i_{h,\text{act}}$. Further details, such as the form of (subdomain-wise) shape functions of $\mathcal{V}^i_{h,\text{act}}$, are not covered here, as they are analogous to those in [24].

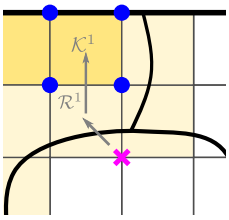



Figure 4.3: Close-up of Figure 4.1(b) illustrating an ill-posed DOF (×) in $u^1_{h,\text{act}}$ mapped to a well-posed cell via $\mathcal{K}^1$. The resulting constraining DOFs, i.e. $\Sigma_{\mathcal{K}^i}$, are marked with ●.

After defining independent AgFE spaces in $\Omega^i$, $i = 1, \ldots, N$, a *global* aggregated FE space $\mathcal{V}_{h,\text{ag}}$ is straightforwardly derived as the Cartesian product of subdomain counterparts, i.e. $\mathcal{V}_{h,\text{ag}} \doteq \mathcal{V}^1_{h,\text{ag}} \times \ldots \times \mathcal{V}^N_{h,\text{ag}}$. We remark that, as $\mathcal{T}^i_{h,\text{act}}$ overlaps in cells cutting the skeleton $\Gamma_0$, DOFs lying on a cut cell are mapped to as many different global DOFs, as active meshes overlapping the cell, via the local-to-subdomain DOF map $\sigma^i$. However, some replicated DOFs may be marked as ill-posed and become constrained. As a result, they do not increase the size of the linear system.[1]

## 4.3 Approximation of unfitted interface elliptic problems

In this section, we address the approximation of compressible linear elasticity problems with the AgFEM. We introduce first the continuous interface problem (4.2) and prove

[1] In this sense, AgFEM departs from other unfitted techniques that rely on the same interface-overlapping mesh approach, such as cutFEM. In those cases, the problem is incremented by the number of replicated DOFs. In particular, the total number of (free) DOFs is $\sum_{i=1,N} \left|\Sigma^i_{\text{act}}\right|$. In contrast, the size of the linear system in AgFEM is always smaller and *bounded above* by $\sum_{i=1,N} \left|\Sigma^i_{\text{act}}\right|$; indeed, the total number of DOFs is regulated by the well-posedness threshold $\eta_0$. For $\eta_0$ equal to zero, we would exactly have $\sum_{i=1,N} \left|\Sigma^i_{\text{act}}\right|$, but this is the standard XFEM case, which is useless because it does not get rid of the small cut cell problem. The larger $\eta_0$ is, the more cells are marked as ill-posed and thus the number of DOFs reduced, because more DOFs are constrained and do not appear in the (reduced) linear system. In the aggregation process, replicated DOFs on the interface cells are eliminated and one can easily end up with a problem even smaller than the original FE problem. In any case, the interface region usually demands more refined meshes due to small scale local effects. This is accomplished by combining AgFEM with adaptive mesh refinement and coarsening (see Chapter 3).

that the weak formulation (4.4)-(4.5) is well-posed. Afterwards, we consider a consistent Nitsche's method (4.7) to discretise the problem with AgFEM. We conclude by examining well-posedness and approximability properties of the discrete problem (4.7), which lead to optimal *a priori* error estimates.

From this point onwards, we restrict ourselves to single interface problems with two subdomains, i.e. there is a unique physical interface $\Gamma_0 \equiv \Gamma^{12}$; henceforth denoted simply by $\Gamma$. This assumption contributes to conciseness and readability; all concepts presented here can be easily extended to the general case with an arbitrary number of subdomains. For the sake of the numerical analysis, let $\Gamma$ be a smooth manifold with bounded curvature. To distinguish the two subdomains, we use superscripts $+, -$ instead of $1, 2$, e.g. the subdomains are denoted by $\Omega^+$ and $\Omega^-$. In addition, we employ superscript $\alpha \in \{+, -\}$ to refer to any of the subdomains and $\pm$ to refer to the broken domain, i.e. $\Omega^\pm \doteq \Omega^+ \cup \Omega^-$.

Before describing the model problem and approximation, we introduce some additional notation. Let $\boldsymbol{v}$ be a smooth enough vector or tensor function defined in $\Omega$. We denote by $\boldsymbol{v}^\alpha \doteq \boldsymbol{v}|_{\Omega^\alpha}$ the restriction of $\boldsymbol{v}$ into $\Omega^\alpha$; conversely, given $\boldsymbol{v}^\alpha$ defined in $\Omega^\alpha$, we identify the pair $\{\boldsymbol{v}^+, \boldsymbol{v}^-\}$ with the function $\boldsymbol{v}$ in $\Omega^\pm$, that is equal to $\boldsymbol{v}^\alpha$ in $\Omega^\alpha$. On the interface, we define $\boldsymbol{v}^+|_\Gamma(\boldsymbol{x}) = \lim_{\epsilon\to 0^+} \boldsymbol{v}(\boldsymbol{x} - \epsilon \boldsymbol{n}^+)$ and $\boldsymbol{v}^-|_\Gamma(\boldsymbol{x}) = \lim_{\epsilon\to 0^-} \boldsymbol{v}(\boldsymbol{x} + \epsilon \boldsymbol{n}^-)$. We define the *jump* of $\boldsymbol{v}$ across $\Gamma$ by $[\![\boldsymbol{v}]\!] \doteq \boldsymbol{v}^+|_\Gamma - \boldsymbol{v}^-|_\Gamma$ and the *weighted average* of $\boldsymbol{v}$ on $\Gamma$ as $\{\{\boldsymbol{v}\}\} \doteq w^+ \boldsymbol{v}^+|_\Gamma + w^- \boldsymbol{v}^-|_\Gamma$, with $0 \leq w^\alpha \leq 1$ and $w^+ + w^- = 1$.

On the other hand, we use standard notation for Sobolev spaces (see, e.g. [185]). For instance, the $L^2(\omega)$ norm is denoted by $\|\cdot\|_{L^2(\omega)}$, the $H^1(\omega)$ norm as $\|\cdot\|_{H^1(\omega)}$ and the $H^1(\omega)$ seminorm as $|\cdot|_{H^1(\omega)}$. Given the two disjoint open connected subdomains $\Omega^+, \Omega^- \subset \mathbb{R}^d$, the Sobolev spaces of the form $H^s(\Omega^+) \times H^s(\Omega^-)$ are represented with $H^s(\Omega^\pm)$, endowed with the norm $\|\cdot\|_{H^s(\Omega^\pm)} \doteq (\|\cdot\|^2_{H^s(\Omega^+)} + \|\cdot\|^2_{H^s(\Omega^-)})^{1/2}$; analogously for seminorms. Vector-valued Sobolev spaces are represented with boldface letters. We use common notation $A \lesssim CB$ or $A \gtrsim CB$ to denote that $A \leq B$ or $A \geq B$ for some positive constant $C$. In this chapter, constants may depend on the order of the FE space and the user-defined value $\eta_0$, but they may not depend on the mesh-interface intersection (i.e. how the cells are intersected), the mesh size of the background mesh, or the contrast of the physical parameters at both sides of the interface.

Moreover, let us assume that the aggregate size is bounded by a constant times $h_T$, where $T$ is the root of the aggregate. This can be shown to hold when assuming that the ratio between the size of two neighbouring cells cannot be arbitrarily large, e.g. using standard 2:1 balance in adaptive non-conforming tree meshes or a patch-local quasi-regularity assumption on unstructured meshes (see also [24, Lemma 2.2]).

Lastly, we introduce the set of faces $\mathcal{F}_h$ that are generated after the intersection of $\Gamma$ and the mesh $\mathcal{T}_h$, i.e. $\mathcal{F}_h \doteq \left\{\bigcup_{T\in\mathcal{T}_h} \Gamma \cap T\right\} \cup \left\{\bigcup_{T,T'\in\mathcal{T}_h \,:\, T\neq T'} \Gamma \cap \left(\overline{T} \cap \overline{T'}\right)\right\}$; a face $F$ in $\mathcal{F}_h$ can be on the boundary of the background mesh cells or intersect the cells. In the subsequent analysis, there is no difference between the two cases and, thus, we do not distinguish among them. Given $F \in \mathcal{F}_h$, we let $T_F^\alpha \in \mathcal{T}^\alpha_{h,\text{act}}$ such that $F \cap \overline{\Omega^\alpha} \subset \overline{T_F^\alpha}$ and

$h_{T_F} \doteq \max\{h_{T_F^+}, h_{T_F^-}\}$. Note that $T_F^+ \equiv T_F^-$ for faces that intersect the cells.

Our main goal is to prove that all constants being used in the analysis are independent of $h$ and the cell-interface intersection. They may depend, though, on the well-posedness threshold $\eta_0$, the shape of $\Gamma$, and the order of the FE approximation. The key strategy in the analysis, in order to prove robustness to the small cut cell problem, is to build upon well-behaved properties, that enjoy AgFE spaces in BVPs posed on unfitted boundaries, i.e. where $\partial\Omega$ is unfitted, instead of $\Gamma$; these properties have been thoroughly covered in [20, 24]. We will often refer to them, without repeating details, to keep the presentation short.

Besides, we also aim to gain some control on the robustness of method (4.7) to material contrast. Since we rule out incompressibility, we adopt the quotient of $\mu$ coefficients at either sides of $\Gamma$ as the measure of material contrast, i.e. we consider $\mu_+/\mu_-$ in the numerical experiments. Therefore, we can follow the usual approach for the Laplacian problem, adopted in body-fitted DG [57] and small-cut-stable unfitted [35] methods. In particular, we employ the so-called *harmonic* average weights, that is $w_+ \doteq \frac{\mu_-}{\mu_+ + \mu_-}$ and $w_- \doteq \frac{\mu_+}{\mu_+ + \mu_-}$. Clearly, $w_\alpha$, $\alpha \in \{+,-\}$, does not depend on cut location, only on material contrast. We will denote the harmonic average of $\mu$ by $\overline{\mu} \doteq \frac{2\mu_+\mu_-}{\mu_+ + \mu_-}$. We have that $\mu_{\min} \leq \overline{\mu} \leq \mu_{\max}$ and $\overline{\mu} \leq 2\mu_{\min}$.

### 4.3.1 Model problem:

We consider the linear isotropic elasticity problem with discontinuous Lamé parameters across $\Gamma$, even though the following discussion and analysis can also be particularised to the Poisson equation, or any other elliptic problem with $H^1$-stability. We adopt a pure-displacement (irreducible) model [77]. For simplicity, we assume homogeneous Dirichlet boundary conditions on $\partial\Omega$. Other options, such as non-homogeneous Dirichlet or Neumann boundary conditions, can readily be considered using standard arguments. We also assume non-homogeneous (immersed) interface transmission conditions, as we deal with them in some examples of Section 4.4, although displacement and normal stress jumps across $\Gamma$ are zero in most practical situations. According to this, the model problem seeks to find the displacement field $\boldsymbol{u} : \Omega^+ \cup \Omega^- \to \mathbb{R}^d$ such that

$$\begin{cases} -\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{f} & \text{in } \Omega^+\cup\Omega^-, \\ \boldsymbol{u} = 0 & \text{on } \partial\Omega, \\ [\![\boldsymbol{u}]\!] = \boldsymbol{j}_\Gamma & \text{on } \Gamma, \text{ and} \\ [\![\boldsymbol{\sigma}(\boldsymbol{u})]\!]\cdot\boldsymbol{n}^+ = \boldsymbol{g}_\Gamma & \text{on } \Gamma, \end{cases} \tag{4.2}$$

where $\boldsymbol{\varepsilon}, \boldsymbol{\sigma} : \Omega^+\cup\Omega^- \to \mathbb{R}^{d,d}$ are the strain tensor $\boldsymbol{\varepsilon}(\boldsymbol{u}) \doteq \frac{1}{2}(\boldsymbol{\nabla}\boldsymbol{u} + \boldsymbol{\nabla}\boldsymbol{u}^T)$ and stress tensor $\boldsymbol{\sigma}(\boldsymbol{u}) = 2\mu\boldsymbol{\varepsilon}(\boldsymbol{u}) + \lambda\mathrm{tr}(\boldsymbol{\varepsilon}(\boldsymbol{u}))\mathbf{Id}$; where $\mathbf{Id}$ denotes the identity matrix in $\mathbb{R}^d$. Apart from that, we let $\boldsymbol{f} \in \boldsymbol{L}^2(\Omega)$ represent the body forces, whereas $\boldsymbol{j}_\Gamma$ and $\boldsymbol{g}_\Gamma$ denote the fixed jump and forcing terms on $\Gamma$. We assume that $\boldsymbol{j}_\Gamma \in \boldsymbol{H}^{1/2}_{00}(\Gamma)$ and $\boldsymbol{g}_\Gamma \in \boldsymbol{H}^{1/2}(\Gamma)$. We recall that $\boldsymbol{H}^{1/2}_{00}(\Gamma)$ is the subspace of functions in $\boldsymbol{H}^{1/2}(\Gamma)$, whose extension by zero on

$\partial\Omega$ is in $\boldsymbol{H}^{1/2}(\partial\Omega \cup \Gamma)$ [185, Appendix A.2]. Since $\boldsymbol{j}_\Gamma \in \boldsymbol{H}^{1/2}_{00}(\Gamma)$, its extension by zero to $\partial\Omega^\alpha$, $\alpha \in \{+,-\}$, is bounded in $\boldsymbol{H}^{1/2}(\partial\Omega^\alpha)$, which we represent with $\boldsymbol{j}_{\partial\Omega^\alpha}$. On the other hand, $\boldsymbol{n}^+$ is the exterior normal to $\Omega^+$.

We assume the Lamé coefficients to be subdomain constant, i.e. $\lambda(\boldsymbol{x}) \doteq \lambda_\alpha \geq 0$ and $\mu(\boldsymbol{x}) \doteq \mu_\alpha > 0$ for $\boldsymbol{x} \in \Omega^\alpha$, $\alpha \in \{+,-\}$, but can have different values across $\Gamma$. Furthermore, we consider the Poisson ratio $\nu_\alpha \doteq \lambda_\alpha/(2(\lambda_\alpha + \mu_\alpha))$ is bounded away from $1/2$, i.e. the material is compressible. Since $\lambda_\alpha = 2\nu_\alpha\mu_\alpha/(1-2\nu_\alpha)$, $\lambda_\alpha$ is bounded above by $\mu_\alpha$, i.e. $\lambda_\alpha \leq C\mu_\alpha$, $C > 0$. Combined with the Cauchy-Schwarz inequality, it leads to the upper bound

$$\int_{\Omega^\alpha} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v}) \, \mathrm{d}\Omega \leq C_\nu \mu_\alpha \|\boldsymbol{\nabla u}\|_{\boldsymbol{L}^2(\Omega^\alpha)} \|\boldsymbol{\nabla v}\|_{\boldsymbol{L}^2(\Omega^\alpha)}, \qquad \forall \boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{H}^1(\Omega^\alpha). \tag{4.3}$$

We can now use (4.3) and the fact that $\boldsymbol{j}_\Gamma \in \boldsymbol{H}^{1/2}_{00}(\Gamma)$ to show that the weak form of (4.2) is well-posed. To this end, we let the decomposition $\boldsymbol{u} \doteq \boldsymbol{w} + \boldsymbol{h}_j \in \boldsymbol{H}^1(\Omega^\pm)$, such that the weak solution of (4.2) becomes: *find* $\boldsymbol{u} = \boldsymbol{w} + \boldsymbol{h}_j \in \boldsymbol{H}^1(\Omega^\pm)$, *where*

$$\boldsymbol{h}_j \in \boldsymbol{H}^1(\Omega^+) \; : \; \int_{\Omega^+} \boldsymbol{\sigma}(\boldsymbol{h}_j) : \boldsymbol{\varepsilon}(\boldsymbol{v}) \, \mathrm{d}\Omega = 0, \qquad \boldsymbol{h}_j = \boldsymbol{j}_{\partial\Omega^+} \text{ in } \partial\Omega^+, \qquad \text{and} \tag{4.4}$$

$$\boldsymbol{w} \in \boldsymbol{H}^1_0(\Omega) \; : \; \int_\Omega \boldsymbol{\sigma}(\boldsymbol{w}) : \boldsymbol{\varepsilon}(\boldsymbol{v}) \, \mathrm{d}\Omega = -\int_\Omega \boldsymbol{\sigma}(\boldsymbol{h}_j) : \boldsymbol{\varepsilon}(\boldsymbol{v}) \, \mathrm{d}\Omega + \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v} \, \mathrm{d}\Omega + \int_\Gamma \boldsymbol{g}_\Gamma \cdot \boldsymbol{v} \, \mathrm{d}\Gamma, \tag{4.5}$$

*for all* $\boldsymbol{v} \in \boldsymbol{H}^1_0(\Omega)$. Thus, $\boldsymbol{u} \in \mathcal{V}$, where

$$\mathcal{V} \doteq \{\boldsymbol{v} \in \boldsymbol{H}^1(\Omega^\pm) : \boldsymbol{v} = \boldsymbol{0} \text{ on } \partial\Omega\}.$$

Continuity of the bilinear form in $\boldsymbol{H}^1(\Omega^\pm)$ is a direct consequence of (4.3). On the other hand, using the Korn inequality, which leads to

$$\int_\omega \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{u}) \, \mathrm{d}\Omega \geq C_\sigma C_\omega \mu \|\boldsymbol{\nabla u}\|^2_{\boldsymbol{L}^2(\omega)}, \tag{4.6}$$

for an open, bounded and connected $\omega \subset \Omega$ and its related Korn constant $C_\omega > 0$, together with the first Poincaré-Friedrichs inequality, we prove coercivity. We can readily apply Lax-Milgram's lemma on (4.4), leading to $\|\boldsymbol{h}_j\|_{\boldsymbol{H}^1(\Omega^+)} \lesssim \|\boldsymbol{j}_{\partial\Omega^+}\|_{\boldsymbol{H}^{1/2}(\partial\Omega^+)} \lesssim \|\boldsymbol{j}_\Gamma\|_{\boldsymbol{H}^{1/2}(\Gamma)}$. Finally, continuity of the right-hand side of (4.5) follows from the Cauchy-Schwarz inequality and a trace theorem:

$$\begin{aligned}
-\int_{\Omega^+} \boldsymbol{\sigma}(\boldsymbol{h}_j) : \boldsymbol{\varepsilon}(\boldsymbol{v}) \, \mathrm{d}\Omega &\lesssim \mu_+^{1/2} \|\boldsymbol{j}_\Gamma\|_{\boldsymbol{H}^{1/2}(\Gamma)} \|\mu^{1/2}\boldsymbol{v}\|_{\boldsymbol{H}^1(\Omega^+)}, \\
\int_\Omega \boldsymbol{f} \cdot \boldsymbol{v} \, \mathrm{d}\Omega &\lesssim \|\mu^{-1/2}\boldsymbol{f}\|_{\boldsymbol{L}^2(\Omega)} \|\mu^{1/2}\boldsymbol{v}\|_{\boldsymbol{L}^2(\Omega)}, \\
\int_\Gamma \boldsymbol{g}_\Gamma \cdot \boldsymbol{v} \, \mathrm{d}\Gamma &\lesssim \|\overline{\mu}^{-1/2}\boldsymbol{g}_\Gamma\|_{\boldsymbol{L}^2(\Gamma)} \|\overline{\mu}^{1/2}\boldsymbol{v}\|_{\boldsymbol{L}^2(\Gamma)} \lesssim \|\overline{\mu}^{-1/2}\boldsymbol{g}_\Gamma\|_{\boldsymbol{H}^{1/2}(\Gamma)} \|\mu^{1/2}\boldsymbol{v}\|_{\boldsymbol{H}^1(\Omega^\pm)}, \quad \forall \boldsymbol{v} \in \mathrm{H}^1_0(\Omega).
\end{aligned}$$

Combining all these results, existence and uniqueness of the weak solution to (4.2) is ensured by the Lax-Milgram theorem. Moreover, the problem is well-posed, since the

unique solution is bounded by the data as follows:

$$\|\mu^{1/2}\boldsymbol{u}\|_{\boldsymbol{H}^1(\Omega^\pm)} \lesssim \mu_+^{1/2}\|\boldsymbol{j}_\Gamma\|_{\boldsymbol{H}^{1/2}(\Gamma)} + \|\mu^{-1/2}\boldsymbol{f}\|_{\boldsymbol{L}^2(\Omega)} + \|\overline{\mu}^{-1/2}\boldsymbol{g}_\Gamma\|_{\boldsymbol{H}^{1/2}(\Gamma)}.$$

### 4.3.2 Discrete formulation

We consider as approximation space of $\mathcal{V}$ the aggregated FE space, see Section 4.2.3,

$$\mathcal{V}_h \doteq \{\boldsymbol{v}_h \in \mathcal{V}_{\text{ag}}^+ \times \mathcal{V}_{\text{ag}}^- : \boldsymbol{v}_h = \boldsymbol{0} \text{ on } \partial\Omega\}.$$

We consider an approximation of (4.5) with this discrete space, which reads:

$$\boldsymbol{u}_h \in \mathcal{V}_h \;:\; a_h(\boldsymbol{u}_h, \boldsymbol{v}_h) = \ell_h(\boldsymbol{v}_h), \quad \forall \boldsymbol{v}_h \in \mathcal{V}_h, \tag{4.7}$$

where the global FE operators $a_h$ and $\ell_h$ are given by

$$\begin{aligned}
a_h(\boldsymbol{u}_h, \boldsymbol{v}_h) &\doteq \int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}_h) : \boldsymbol{\varepsilon}(\boldsymbol{v}_h)\, \mathrm{d}\Omega \\
&\quad + \sum_{F\in\mathcal{F}_h} \left[ \frac{\beta\overline{\mu}}{h_{T_F}} \int_F [\![\boldsymbol{u}_h]\!] \cdot [\![\boldsymbol{v}_h]\!]\, \mathrm{d}\Gamma - \int_F \boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{v}_h)\}\} \cdot [\![\boldsymbol{u}_h]\!]\, \mathrm{d}\Gamma - \int_F \boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u}_h)\}\} \cdot [\![\boldsymbol{v}_h]\!]\, \mathrm{d}\Gamma \right], \\
\ell_h(\boldsymbol{v}_h) &\doteq \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v}_h\, \mathrm{d}\Gamma \\
&\quad + \sum_{F\in\mathcal{F}_h} \left[ \frac{\beta\overline{\mu}}{h_{T_F}} \int_F \boldsymbol{j}_\Gamma \cdot [\![\boldsymbol{v}_h]\!]\, \mathrm{d}\Gamma - \int_F \boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{v}_h)\}\} \cdot \boldsymbol{j}_\Gamma\, \mathrm{d}\Gamma + \int_F \boldsymbol{g}_\Gamma \cdot \left(w_- \boldsymbol{v}_h^+ + w_+ \boldsymbol{v}_h^-\right)\, \mathrm{d}\Gamma \right].
\end{aligned}$$

We observe that $a_h$ and $\ell_h$ contain the usual terms in Nitsche's formulations, i.e. terms that weakly impose the interface conditions, symmetrizing terms and stabilization terms. The latter terms are those premultiplied by $\beta$, which has to be large-enough to ensure coercivity of the bilinear form $a_h$. Furthermore, the above formulation is *consistent*, by the following result:

**Lemma 4.3.1** (Consistency)**.** *Let $\boldsymbol{u} \in \boldsymbol{H}^2(\Omega^\pm) \cap \mathcal{V}$ solve* (4.2)*. Then, it holds $a_h(\boldsymbol{u}, \boldsymbol{v}_h) = \ell_h(\boldsymbol{v}_h)$, $\forall \boldsymbol{v}_h \in \mathcal{V}_h$.*

*Proof.* Since $\boldsymbol{u}$ solves problem (4.2) (in a weak sense), integration by parts leads to:

$$\begin{aligned}
\int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v}_h)\, \mathrm{d}\Omega &= -\int_{\Omega^+\cup\Omega^-} \boldsymbol{v}_h \cdot \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(\boldsymbol{u})\, \mathrm{d}\Omega \\
&\quad + \int_\Gamma \boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u})\}\} \cdot [\![\boldsymbol{v}_h]\!]\, \mathrm{d}\Gamma + \int_\Gamma \boldsymbol{n}^+ \cdot [\![\boldsymbol{\sigma}(\boldsymbol{u})]\!] \cdot \left(w^- \boldsymbol{v}_h^+ + w^+ \boldsymbol{v}_h^-\right)\, \mathrm{d}\Gamma,
\end{aligned}$$

for any $\boldsymbol{v}_h \in \mathcal{V}_h$. Combining this result with $-\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{f}$, $[\![\boldsymbol{u}]\!] = \boldsymbol{j}_\Gamma$ and $\boldsymbol{n}^+ \cdot [\![\boldsymbol{\sigma}(\boldsymbol{u})]\!] = \boldsymbol{g}_\Gamma$, we can check that all terms in the discrete formulation (4.7) cancel out. □

For the sake of proving coercivity, we need the following auxiliary result.

**Lemma 4.3.2.** *Let $T \in \mathcal{T}_{h,\mathrm{W}}^\alpha$ and $\boldsymbol{u}_T \in \mathcal{Q}_q(T)$. There exists $C_{\eta_0} > 0$, dependent on the well-posedness threshold $\eta_0$, such that $\|\boldsymbol{u}_T\|_{\boldsymbol{L}^2(T)}^2 \leq C_{\eta_0}\|\boldsymbol{u}_T\|_{\boldsymbol{L}^2(T\cap\Omega^\alpha)}^2$, $\alpha \in \{+,-\}$.*

*Proof.* The proof is direct for *interior* well-posed cells; we restrict ourselves to well-posed *cut* cells. Let us consider a cell $T$ and its interior portion $T \cap \Omega$. Using the inverse of the geometrical map, which maps $T$ into the reference cell $\hat{T}$, one can map the interior portion to the reference cell, which is represented with $\hat{T}_{\text{in}}$. It is easy to check that $\text{meas}_d(\hat{T}_{\text{in}}) \geq C\eta_0 \text{meas}_d(\hat{T})$. In fact, the constant is 1 for affine maps. $\|\cdot\|^2_{\boldsymbol{L}^2(\hat{T}_{\text{in}})}$ is a norm for $\mathcal{Q}_q(\hat{T})$, since a polynomial that vanishes in a domain of non-zero measure is equal to zero. We prove the result by using the equivalence of norms in finite-dimensional vector spaces and a scaling argument. □

Given $T \in \mathcal{T}^{\alpha}_{h,\text{act}}$, $\alpha \in \{+,-\}$, let us denote by $T_1, \dots, T_{n^{\alpha}_T}$, $n^{\alpha}_T \geq 1$, the set of *constraining* well-posed cells of $T$ in $\mathcal{T}^{\alpha}_{h,\text{W}}$, i.e. the set of well-posed cells that constrain at least one DOF of $T$ in $\mathcal{T}^{\alpha}_{h,\text{W}}$. Let us also define $\Omega_{T^{\alpha}_F} \doteq \Omega^{\alpha} \cap \left(T^{\alpha}_F \cup \bigcup_{i=1}^{n^{\alpha}_T} T_i\right)$. With this notation, we can state the following inequality, which holds for discrete functions in cut cells (see Lemma 3.4.7):

$$\|\boldsymbol{\nabla} \boldsymbol{v}^{\alpha}_h\|^2_{\boldsymbol{L}^2(F)} \lesssim C_{\eta_0} h^{-1}_{T^{\alpha}_F} \|\boldsymbol{\nabla} \boldsymbol{v}^{\alpha}_h\|^2_{\boldsymbol{L}^2(\Omega_{T^{\alpha}_F})}, \quad \alpha \in \{+,-\}, \quad \forall \boldsymbol{v}_h \in \mathcal{V}_h, \quad \forall F \in \mathcal{F}_h. \tag{4.8}$$

We also make use of the following inequality for continuous functions on cut cells (see [88]):

$$\|\psi\|^2_{L^2(\partial(\Omega \cap T))} \lesssim h^{-1}_T \|\psi\|^2_{L^2(\Omega \cap T)} + h_T \, |\psi|^2_{H^1(\Omega \cap T)}, \qquad \forall \psi \in H^1(\Omega \cap T), \tag{4.9}$$

where $\partial(\Omega \cap T)$ is the boundary of $\Omega \cap T$.[2]

Let us define the space $\mathcal{V}(h) \doteq \mathcal{V}_h + \boldsymbol{H}^2(\Omega^{\pm}) \cap \mathcal{V}$. We endow $\mathcal{V}(h)$ with the broken norm:

$$\|\boldsymbol{v}\|^2_{\mathcal{V}(h)} \doteq \sum_{\alpha \in \{+,-\}} \mu_{\alpha} \|\boldsymbol{\nabla} \boldsymbol{v}^{\alpha}\|^2_{\boldsymbol{L}^2(\Omega^{\alpha})} + \sum_{F \in \mathcal{F}_h} \frac{\overline{\mu}}{h_{T_F}} \|[\![\boldsymbol{v}]\!]\|^2_{\boldsymbol{L}^2(F)} + \sum_{\alpha \in \{+,-\}} \sum_{T \in \mathcal{T}^{\alpha}_{h,\text{act}}} \mu_{\alpha} h^2_T \|\boldsymbol{v}\|^2_{\boldsymbol{H}^2(T \cap \Omega^{\alpha})}.$$

It can be checked that $\|\boldsymbol{v}\|_{\boldsymbol{L}^2(\Omega)} \lesssim \|\boldsymbol{v}\|_{\mathcal{V}(h)}$, for $\boldsymbol{v} \in \mathcal{V}(h)$, see, e.g. [24, Lemma 5.8]. The following lemma restricted to the discrete space $\mathcal{V}_h$ provides the well-posedness of the discrete problem. Its extension to $\mathcal{V}(h)$ will be required in the convergence analysis.

**Lemma 4.3.3** (Well-posedness)**.** *The bilinear form in the discrete formulation* (4.7) *satisfies the following properties uniformly w.r.t. the mesh size h of the background mesh and interface intersection:*

*(i) Coercivity:*

$$a_h(\boldsymbol{u}_h, \boldsymbol{u}_h) \gtrsim \|\boldsymbol{u}_h\|^2_{\mathcal{V}(h)}, \qquad \forall \boldsymbol{u}_h \in \mathcal{V}_h,$$

*if $\beta > C$, for some (large-enough) positive constant C.*

[2] We note that the proof in [88] assumes that $\Omega \cap T$ is connected, together with the assumption that $\Gamma$ has a bounded curvature. The connected intersection can be handled either replicating cells (as commented above) or assuming a *fine enough* mesh.

*(ii) Continuity:*

$$a_h(\boldsymbol{u},\boldsymbol{v}) \lesssim \|\boldsymbol{u}\|_{\mathcal{V}(h)}\|\boldsymbol{v}\|_{\mathcal{V}(h)}, \qquad \forall \boldsymbol{u},\boldsymbol{v}\in\mathcal{V}(h).$$

*Therefore, there exists a unique solution to problem* (4.7).

*Proof.* By definition of the bilinear form $a_h$ and (4.6), we have that

$$\begin{aligned} a_h(\boldsymbol{u}_h,\boldsymbol{u}_h) \gtrsim & \sum_{\alpha\in\{+,-\}} C_\sigma C_\omega \mu_\alpha \|\boldsymbol{\nabla} \boldsymbol{u}_h^\alpha\|^2_{\boldsymbol{L}^2(\Omega^\alpha)} + \sum_{F\in\mathcal{F}_h} \frac{\beta\overline{\mu}}{h_{T_F}} \|[\![\boldsymbol{u}_h]\!]\|^2_{\boldsymbol{L}^2(F)} \\ & - 2\sum_{F\in\mathcal{F}_h} \int_F \boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u}_h)\}\}\cdot [\![\boldsymbol{u}_h]\!]\,\mathrm{d}\Gamma, \end{aligned} \tag{4.10}$$

for any $\boldsymbol{u}_h \in \mathcal{V}_h$. In order to prove coercivity, we have to bound the indefinite term. Let us pick an arbitrary $\boldsymbol{u}_h \in \mathcal{V}_h$. Using the fact that $w_\alpha \mu_\alpha = \overline{\mu}$, Cauchy-Schwarz and triangle inequalities and (4.8), we get

$$\|\boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u}_h)\}\}\|^2_{\boldsymbol{L}^2(F)} \lesssim \overline{\mu}^2 \sum_{\alpha\in\{+,-\}} \|\boldsymbol{\nabla}\boldsymbol{u}_h^\alpha\|^2_{\boldsymbol{L}^2(F)} \leq C_{\eta_0}\overline{\mu} \sum_{\alpha\in\{+,-\}} \frac{\mu_\alpha}{h_{T_F^\alpha}} \|\boldsymbol{\nabla}\boldsymbol{u}_h^\alpha\|^2_{\boldsymbol{L}^2(\Omega_{T_F^\alpha})}. \tag{4.11}$$

Usage of the Cauchy-Schwarz and Young inequalities and the previous result leads to

$$\begin{aligned} \left| 2\int_F \boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u}_h)\}\}\cdot [\![\boldsymbol{u}_h]\!]\,\mathrm{d}\Gamma \right| &\lesssim \frac{h_{T_F}}{\gamma\overline{\mu}} \|\boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u}_h)\}\}\|^2_{\boldsymbol{L}^2(F)} + \frac{\gamma\overline{\mu}}{h_{T_F}} \|[\![\boldsymbol{u}_h]\!]\|^2_{\boldsymbol{L}^2(F)} \\ &\lesssim C_{\eta_0} \sum_{\alpha\in\{+,-\}} \frac{\mu_\alpha}{\gamma} \|\boldsymbol{\nabla}\boldsymbol{u}_h^\alpha\|^2_{\boldsymbol{L}^2(\Omega_{T_F^\alpha})} + \frac{\gamma\overline{\mu}}{h_{T_F}} \|[\![\boldsymbol{u}_h]\!]\|^2_{\boldsymbol{L}^2(F)}, \forall \boldsymbol{u}_h \in \mathcal{V}_h, \end{aligned} \tag{4.12}$$

with $\gamma > 0$ an arbitrary positive constant. Combining (4.10) and (4.12), and using the fact that the number of neighbouring cells is bounded, we obtain:

$$a_h(\boldsymbol{u}_h,\boldsymbol{u}_h) \gtrsim \left(C_\sigma C_\omega - \frac{C_{\eta_0}}{\gamma}\right) \sum_{\alpha\in\{+,-\}} \mu_\alpha \|\boldsymbol{\nabla}\boldsymbol{u}_h^\alpha\|^2_{\boldsymbol{L}^2(\Omega^\alpha)} + \left(1-\frac{\gamma}{\beta}\right) \sum_{F\in\mathcal{F}_h} \frac{\beta\overline{\mu}}{h_{T_F}} \|[\![\boldsymbol{u}_h]\!]\|^2_{\boldsymbol{L}^2(F)}.$$

Let us pick $\gamma = \frac{2C_{\eta_0}}{C_\sigma C_\omega}$. Assuming $\beta \geq 2\gamma$, the terms in the right-hand side are positive. In order to check that $a_h(\boldsymbol{u}_h,\boldsymbol{u}_h)$ is also a bound for the $\boldsymbol{H}^2$ broken semi-norm in $\|\cdot\|_{\mathcal{V}(h)}$, we proceed as follows. The local discrete inverse inequality $\|\boldsymbol{\nabla}\boldsymbol{\xi}_h\|_{\boldsymbol{L}^2(T\cap\Omega^\alpha)} \leq Ch^{-1}\|\boldsymbol{\xi}_h\|_{\boldsymbol{L}^2(T)}$ can readily be applied to finite element functions (and its gradients) in agFE spaces (see, e.g. [24, (12)]). On the other hand, by Lemma 4.3.2 we have that $\|\boldsymbol{\xi}_h\|_{\boldsymbol{L}^2(T)} \leq C\|\boldsymbol{\xi}_h\|_{\boldsymbol{L}^2(T\cap\Omega^\alpha)}$. As a result, we have that $h_T\,|\boldsymbol{v}_h|_{\boldsymbol{H}^2(T\cap\Omega^\alpha)} \leq C\|\boldsymbol{\nabla}\boldsymbol{v}_h\|_{\boldsymbol{L}^2(T\cap\Omega^\alpha)}$, for any $\boldsymbol{v}_h \in \mathcal{V}_h$. Hence, bilinear form $a_h$ satisfies coercivity; it is non-singular.

In order to prove continuity, we need a continuous version of (4.11) for functions in $\boldsymbol{H}^2(\Omega^\pm) \cap \mathcal{V}$. Using (4.9), we get the sought-after bound:

$$\begin{aligned}\|\boldsymbol{n}^+ \cdot \{\{\boldsymbol{\sigma}(\boldsymbol{u})\}\}\|^2_{\boldsymbol{L}^2(F)} &\lesssim \overline{\mu}^2 \sum_{\alpha\in\{+,-\}} \|\boldsymbol{\nabla}\boldsymbol{u}^\alpha\|^2_{\boldsymbol{L}^2(F)} \\ &\lesssim \overline{\mu} \sum_{\alpha\in\{+,-\}} \left( \frac{\mu_\alpha}{h_{T_F^\alpha}} \|\boldsymbol{\nabla}\boldsymbol{u}^\alpha\|^2_{\boldsymbol{L}^2(T_F^\alpha\cap\Omega^\alpha)} + \mu_\alpha h_{T_F^\alpha} |\boldsymbol{u}^\alpha|^2_{\boldsymbol{H}^2(T_F^\alpha\cap\Omega^\alpha)} \right).\end{aligned} \tag{4.13}$$

It follows that continuity is a consequence of (4.3), (4.13) and the Cauchy-Schwarz inequality. Since the problem is finite-dimensional and the corresponding linear system matrix is non-singular, there exists a unique solution to problem (4.7). □

Let us assume that the background mesh $\mathcal{T}_h$ is quasi-uniform, with characteristic size $h \doteq \max_{T\in\mathcal{T}_h} h_T$. We adopt now an extended Scott-Zhang interpolant $\Pi_h^{\mathrm{SZ}} : \mathcal{V} \to \mathcal{V}_h$ given by $\Pi_h^{\mathrm{SZ}}(\boldsymbol{u}) = \left\{\Pi_{h,+}^{\mathrm{SZ}}(\boldsymbol{u}), \Pi_{h,-}^{\mathrm{SZ}}(\boldsymbol{u})\right\}$ with $\Pi_{h,\alpha}^{\mathrm{SZ}}(\boldsymbol{u}) \in \mathcal{V}_{\mathrm{ag}}^\alpha$, $\alpha \in \{+,-\}$, defined in [20]. The local approximability property in [20, Theorem 4.4] and the trace inequality (4.9) applied to $\psi = \boldsymbol{u}^\alpha - \Pi_{h,\alpha}^{\mathrm{SZ}}(\boldsymbol{u})$ yield the following result.

**Proposition 4.3.4.** *If $\boldsymbol{u} \in \boldsymbol{H}^m(\Omega^\pm)$, $m \geq 2$, and the order of $\mathcal{V}_h$ is greater or equal than $m-1$, then*

$$\|\boldsymbol{u} - \Pi_h^{\mathrm{SZ}}(\boldsymbol{u})\|_{\mathcal{V}(h)} \lesssim h^{m-1} |\boldsymbol{u}|_{\boldsymbol{H}^m(\Omega^\pm)}.$$

In order to prove *a priori* error estimates, we must assume additional regularity on the solution. For $\Omega$ being a convex polygon, $\Gamma$ of class $\mathcal{C}^2$ and $\boldsymbol{g}_\Gamma \in \boldsymbol{H}_{00}^{1/2}(\Gamma)$, the interface problem enjoys smoothing properties and its solution $\boldsymbol{u} \in \boldsymbol{H}^2(\Omega^\pm)$ (see [50]). Neglecting the geometrical error, the consistency in Lemma 4.3.1, well-posedness in Lemma 4.3.3 and the approximability property in Proposition 4.3.4 can be combined to prove an estimate in the $\mathcal{V}(h)$ norm. Furthermore, under the previous assumptions, a duality argument analogous to [88, Theorem 6] can be used to obtain the $L^2$ estimate. The geometrical error in the approximation could be incorporated into the discussion with the same ideas as, e.g. in [50].

**Proposition 4.3.5.** *If $\boldsymbol{u} \in \boldsymbol{H}^m(\Omega^\pm)$, $m \geq 2$, is the solution of* (4.4)-(4.5) *and $\boldsymbol{u}_h \in \mathcal{V}_h$ is the solution of* (4.7)*, with the order of $\mathcal{V}_h$ greater or equal than $m-1$, then*

$$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{\mathcal{V}(h)} \lesssim h^{m-1} |\boldsymbol{u}|_{\boldsymbol{H}^m(\Omega^\pm)}, \qquad \|\boldsymbol{u} - \boldsymbol{u}_h\|_{\boldsymbol{L}^2(\Omega)} \lesssim h^m |\boldsymbol{u}|_{\boldsymbol{H}^m(\Omega^\pm)}.$$

## 4.4 Numerical experiments

Our goal, in this section, is to analyse numerically the accuracy, optimality, robustness and performance of $h$-AgFEM for interface elliptic BVPs. We consider as model problems the Poisson and linear elasticity equations, with non-homogeneous Dirichlet conditions on the external boundary and discretised with the variational formulation

detailed in Section 4.3.2. We describe first the experimental setup in Section 4.4.1, consisting of several manufactured problems defined in a set of complex geometries. We lay out next the experimental environment of the $h$-AgFEM parallel implementation in FEMPAR [19] in Section 4.4.2. After these preliminaries, we move to report and discuss the numerical results of three different sets of experiments: convergence tests in Section 4.4.3, material contrast and cut location robustness tests in Section 4.4.4 and, finally, weak scaling tests in Section 4.4.5.

### 4.4.1 Experimental benchmarks

Numerical tests consider the variational formulation of Section 4.3.2, with inhomogeneous Dirichlet boundary conditions, applied to the Poisson and linear elasticity problems. Although exposition was restricted to linear elasticity, the formulation for the Poisson equation can be easily derived, as a particular case. This leads to an analogous formulation to the ones in [35, 88, 96]. We observe that, with little effort, the Poisson equation inherits well-posedness and approximability results proven in Section 4.3. Moreover, harmonic weights become $w^+ = \frac{k^-}{k^+ + k^-}$ and $w^- = \frac{k^+}{k^+ + k^-}$, where $k^\alpha > 0$, $\alpha \in \{+, -\}$, represents the subdomain-wise constant diffusion coefficient.

Numerical experiments are carried out on both serial and parallel, distributed-memory, environments. We generally report parallel results; serial ones are only shown when informing about condition numbers. We also observe that parallelisation of interface AgFEM basically reuses ideas that have already been covered in [187]. In addition, all examples run on background Cartesian grids, endowed with standard isotropic 1:4 (2D) or 1:8 (3D) refinement rules; also known as *quadtrees* (2D) or *octrees* (3D). We have also addressed in Chapter 3 how to build AgFE spaces on top of these (generally) non-conforming meshes. In the experiments, we consider both uniform and $h$-adaptive refinements. The latter follow an iterative AMR process that exploits the Li and Bettess convergence (or acceptability) criterion [70, 119]. As usual, the goal of the procedure is to find an optimal mesh, that minimises the number of cells required to achieve a given discretisation error. Nonetheless, we remark that remeshing is not driven by *a posterior* error estimation, since we can compute the exact error in all cases studied, and we do not consider the geometrical error in approximating the interface. In contrast to Chapter 3, we use the *relative* energy norm error in the acceptability criterion to eliminate the influence of material contrast. Seeking to ensure stability, without superfluous aggregation, that degrades accuracy and conditioning, the well-posedness threshold $\eta_0$ to isolate badly-cut cells is set to 0.25.

The FE approximation space for all experiments is $\mathcal{V}_h$, described in Section 4.3.2, as the single-interface version of the general $n$-interface $\mathcal{V}_{h,\text{ag}}$ in Section 4.2.3. Henceforth, we refer to $\mathcal{V}_h$ simply as the AgFE space. We employ both first and second order Lagrangian finite elements. Following discussion in Section 4.3.2, the coercivity coefficient is given by $\beta = 10.0\ q^2$, where $q$ is the FE interpolation order; this value is enough to ensure well-posedness for all the tests below. Apart from that, robustness

tests, in Section 4.4.4, additionally consider a *standard* FE (or StdFE) space defined by $\mathcal{V}_h^{\text{std}} \doteq \mathcal{V}_{h,\text{act}}^{+} \times \mathcal{V}_{h,\text{act}}^{-}$. Although $\mathcal{V}_h^{\text{std}}$ is stable to cut location, under suitable mesh and interface regularity conditions [88], it leads to much more ill-conditioned systems than the AgFE space [62]. For this reason, usage of StdFE space is merely intended to provide a numerical reference to examine the condition number of AgFE space. When using the StdFE space, the coercivity coefficient $\beta$ at each (well- or ill-posed) cut cell is computed by means of solving a generalised eigenvalue problem, detailed in [24, Section 4.2].

The physical domain in all cases is a cuboid (of varying sizes), but the physical interface dividing the two phases is a non-trivial surface, described as the 0-level set of a (piecewise-)smooth function. We consider eight different level-set interfaces: (a) a circle, (b) a flower and (c) a "pacman" shape, in 2D; (d) a cylinder, (e) a popcorn flake, (f) a spiral, (g) a popcorn flake without a wedge (popcorn pacman) and (h) a gyroid, in 3D. All these geometries are covered in the literature [35, 115]; they are typically chosen to examine the behaviour of unfitted FE methods. For illustration purposes, descriptive figures of the considered interfaces (or the interior region that they enclose) are drawn along the convergence plots of Section 4.4.3. Besides, the geometry for the gyroid problem is represented in Figure 4.4.

We study four different analytical benchmarks; all of them are derived with the so-called method of manufactured solutions [158], i.e. we propose a solution of the problem with known analytical solution and then we compute source term and interface conditions from the governing equations (4.2). For the Poisson problem we consider a benchmark for verification (convergence tests), namely the (1) *out-FE-space* benchmark. We add two more Poisson benchmarks, that correspond to adapted versions of two classical $hp$-FEM problems, the (2) *Fichera-corner* and (3) *single-shock* problems. For linear elasticity, we address the (4) *cylindrical inclusion* problem in [176]. Let us next provide the analytical expressions of the solution function for each case.

- The *out-FE-space* benchmark is adapted from [9] and applied to several interface geometries. The solution is given by $u(q,\mathbf{x}) : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ and $q \in \mathbb{N}$ such that

$$u(q;\mathbf{x}) \doteq \begin{cases} \frac{k^+-k^-+(3k^-+k^+)x}{4k^+(k^-+k^+)} - \frac{x^{q+1}}{(q+1)k^+}, & \text{if } \mathbf{x} \in \Omega^+, \\ \frac{(3k^-+k^+)x}{4k^-(k^-+k^+)} - \frac{x^{q+1}}{(q+1)k^-}, & \text{if } \mathbf{x} \in \Omega^-. \end{cases} \tag{4.14}$$

  In our case, we take $q$ as the FE interpolation order, then $u \notin \mathcal{V}_h$. Moreover, $u$ is discontinuous across $\Gamma$, but the jump of normal fluxes is null, i.e. $[\![k\boldsymbol{\nabla} u]\!] \cdot \boldsymbol{n}^+ = 0$.

- The *Fichera-corner* benchmark is adapted from [64] and applied to the pacman and popcorn-pacman interface shapes. The solution $u(r,\theta,z) : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ in cylindrical coordinates is

$$u^\alpha(r,\theta,z) \doteq r^{\omega^\alpha} \sin \omega^\alpha \theta,\ \alpha \in \{+,-\},\ \omega^- = 2/3,\ \omega^+ = 4. \tag{4.15}$$

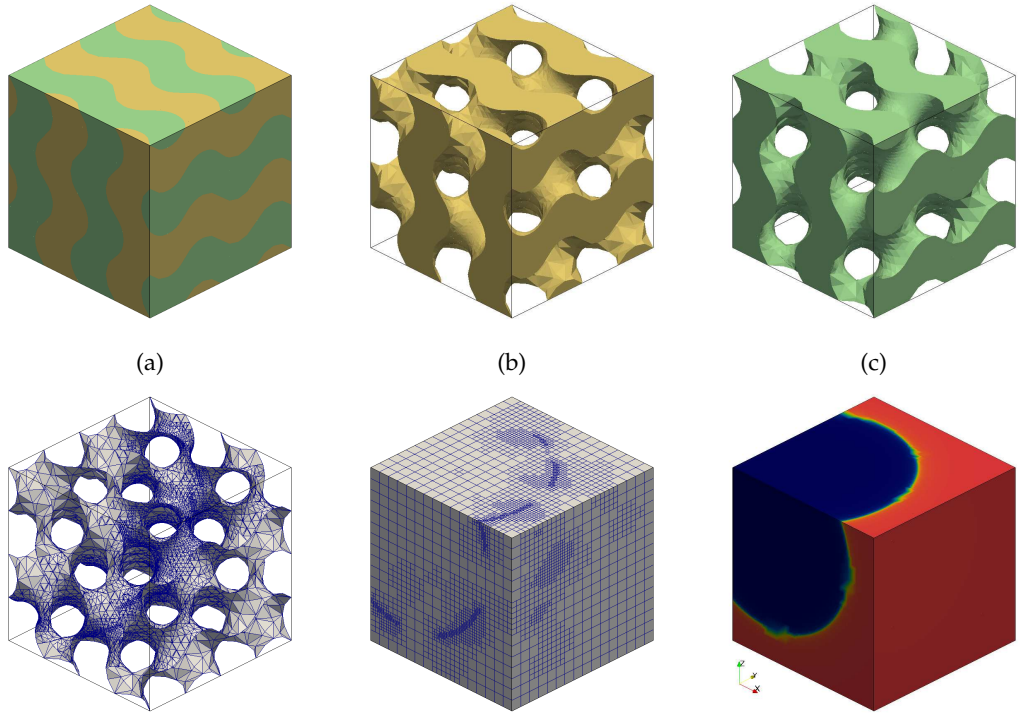
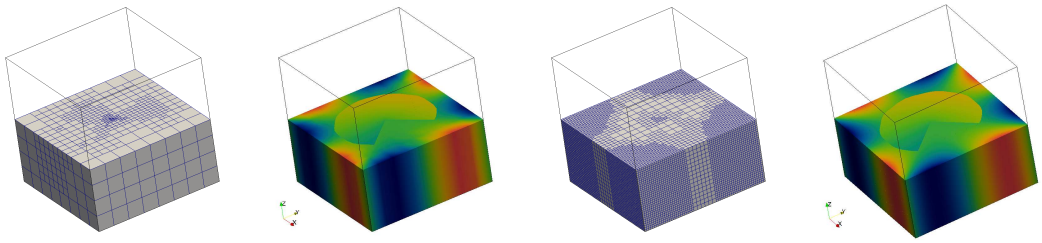


Figure 4.4: The gyroid interface and single-shock benchmark. The top three figures represent the two regions (together and one-by-one) divided by the gyroid level-set function on the region $[-2,2]^3$. The bottom three figures represent the mesh and solution of the single-shock equation (4.16) with $k^+/k^- \neq 1$ on a given $h$-adaptive mesh: the discrete approximated interface in Figure 4.4(d), the mesh in Figure 4.4(e) and the solution in Figure 4.4(f). Different mesh resolution is due to dependency of the energy norm error on material contrast.

Numerical solution of (4.15) in the popcorn flake without a wedge is represented in Figure 4.5. We observe that the problem has fully non-homogeneous interface conditions. Furthermore, $u^+$ is smooth, whereas derivatives of $u^-$ are singular at the $r = 0$ axis; specifically, $u^- \in H^{1+\frac{2}{3}}(\Omega^-)$. When *only* approximating $u^-$, convergence rates of the energy norm with uniform refinements are limited by regularity; they decrease at a rate $\mathcal{O}(h^{-2/3})$. Optimal convergence rates can be restored with $h$-adaptivity [64]. In Section 4.4.3, we argue that, even though $u$ does not explicitly depend on the diffusion coefficients, material contrast determines whether convergence behaviour of $u$ (in the energy norm) is dictated by regularity of $u^+$ or $u^-$.

- The *single-shock* benchmark is also adapted from [64] and applied to the gyroid interface. The solution $u(r) : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ is

$$\begin{aligned} &u(r) \doteq \arctan(\tau(r - r_0)),\ \tau = 60,\ r = \|\mathbf{x} - \mathbf{x}_0\|_2, \\ &r_0 = 2.5,\ \mathbf{x}_0 = (x_0, y_0, z_0) = (-1, -1, 1), \end{aligned} \tag{4.16}$$



(a) $k^+/k^- = 1$ (b) $k^+/k^- = 1$ (c) $k^+/k^- = 10^6$ (d) $k^+/k^- = 10^6$

Figure 4.5: The Fichera-corner benchmark (4.15) on the popcorn-pacman interface in two different situations. We only show mesh and solution at the bottom half of the simulated cube, to show the results on the $z = 0$ plane. Material contrast determines which of the solution sides dominate the numerical error. In the two left plots, $k^+/k^- = 1$ leads to a situation where error and, thus, refinements concentrate in $\Omega^-$. Conversely, in the two right plots, $k^+/k^- = 10^6$ yields higher errors and mesh refinements in $\Omega^+$.

where $\|\cdot\|_2$ denotes the Euclidean norm. Numerical solution of (4.16) in the gyroid is represented in Figure 4.4(f). As in the previous benchmark (4.15), the analytical solution does not depend on the material parameters, but numerical error (in the energy norm) does. Apart from that, $u$ is smooth in $\Omega$, although it sharply varies in the neighbourhood of the shock, and continuous across $\Gamma$, although with a kink if $k^+ \neq k^-$. We notice that the shock may intersect $\Gamma$, e.g. it crosses several times the gyroid 0-level set.

- The *cylindrical inclusion* benchmark is applied to a cylindrical interface. It adapts the linear elasticity problem in [176, Section 7.3]. The displacement in cylindrical coordinates is given by:

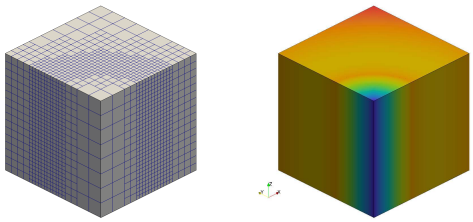
$$u_r(r) \doteq \begin{cases} \left[\left(1 - \frac{b^2}{a^2}\right) c + \frac{b^2}{a^2}\right] r, & 0 \leq r < a, \\ \left(r - \frac{b^2}{r}\right) c + \frac{b^2}{r}, & a \leq r \leq b. \end{cases}, \quad u_\theta \equiv 0, \quad u_z \equiv 0, \tag{4.17}$$

where $a = 0.4$, $b = 2.0$ and

$$c = \frac{(\lambda_- + \mu_- + \mu_+)b^2}{(\lambda_+ + \mu_+)a^2 + (\lambda_- + \mu_-)(b^2 - a^2) + \mu_+ b^2}.$$

In the experiments, $\Omega \subset \{0 \leq r < b\}$ and $\Omega^- = \{0 \leq r < a\}$. The numerical solution is represented in Figure 4.6. As in (4.16), $u$ is continuous across $\Gamma$, but it has a kink if material properties are discontinuous.

Table 4.1 summarizes the main parameters and computational strategies used in the numerical examples. The variety of complex shapes and benchmarks considered above are intended to exhibit the good behaviour of interface AgFEM, in as many situations as possible. Our numerical tests consider first numerical verification of the theoretical results proved in Section 4.3.2. In Section 4.4.3, we carry out convergence



(a) (b)

Figure 4.6: Error-driven adaptive mesh and solution of the linear elasticity problem (4.17) on the cylinder for $\mu_+/\mu_- \neq 1$. The solution has a kink along the interface and error concentrates at the side of the interface outside the cylinder.

tests in uniform and $h$-adaptive meshes to show that interface AgFEM recovers optimal convergence rates. Afterwards, we examine, in Section 4.4.4, robustness to cut location and material contrast, by means of geometry and material perturbations. We show that the condition number of the linear system, after diagonal scaling, is independent of cut location and material contrast. Finally, in Section 4.4.5, we assess good parallel performance and scalability with a weak-scaling analysis of some selected cases from the convergence tests. For each type of numerical test, we perform a subset of the possible matrix of cases in Table 4.1. We provide details for each subset, when dealing with the corresponding test. But before all that, we inform next about the computational infrastructure and software employed.

| Description | Considered methods/values |
|---|---|
| Model problem | interface Poisson, interface linear elasticity |
| Problem geometry | 2D: circle, flower, pacman shape; 3D: cylinder, popcorn flake, spiral, popcorn pacman, gyroid |
| Benchmark | out-FE-space (4.14), Fichera corner (4.15), single shock (4.16), cylindrical inclusion (4.17) |
| Experimental computer environment | serial and parallel |
| Parallel mesh generation and partitioning tool | `p4est` library [43] |
| Mesh topology | single quadtree (2D) or octree (3D) |
| Remeshing strategy | uniform, $h$-adaptive with Li and Bettess [119] rule |
| Well-posed cut cell criterion | $\eta_0 = 0.25$ |
| FE spaces | AgFE and StdFE |
| Cell type and FE interpolation | Q1 and Q2 hexahedral cells |
| Linear solver | sparse direct (serial)<br>preconditioned conjugate gradients (parallel) |
| Parallel preconditioner | smoothed-aggregation `GAMG` [1] |
| `GAMG` stopping criterion | $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 < 10^{-9}$ |
| Weights in averaged normal fluxes | $w^+ = \frac{k^-}{k^+ + k^-}$ and $w^- = \frac{k^+}{k^+ + k^-}$ (Poisson)<br>$w^+ = \frac{\mu^-}{\mu_+ + \mu_-}$ and $w^- = \frac{\mu^+}{\mu_+ + \mu_-}$ (elasticity) |
| Coef. in Nitsche's penalty term for AgFEM | $\beta = 10.0\, q^2$, $q$ is the FE interpolation order |

Table 4.1: Summary of main parameters and computational strategies used in the numerical examples

### 4.4.2 Experimental environment

Serial experiments are launched at the TITANI cluster of the Universitat Politècnica de Catalunya (Barcelona, Spain) in a single DELL PowerEdge R630 node. The node is composed of 2 Intel Xeon E52650L v3 (1.8 GHz) CPUs, with 12 cores per processor and 128 GB of RAM. On the other hand, parallel experiments are carried out at the Marenostrum-IV (MN-IV) supercomputer, hosted by the Barcelona Supercomputing Centre. MN-IV is a petascale machine equipped with 3,456 compute nodes interconnected with the Intel OPA HPC netchapter. Each node has 2x Intel Xeon Platinum 8160 multi-core CPUs, with 24 cores each (i.e. 48 cores per node) and from 96 to 384 GB of RAM.

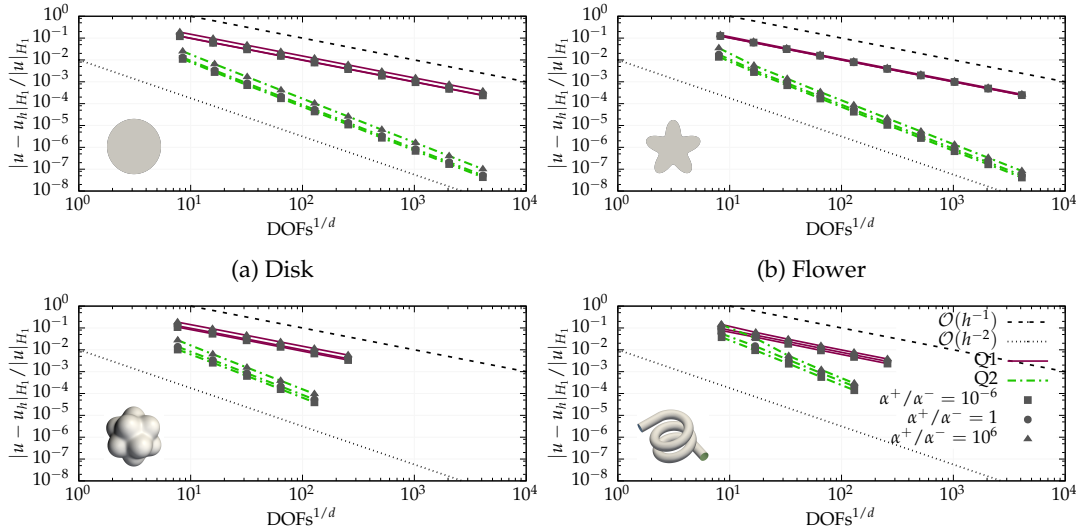
With respect to the software, a MPI parallel implementation of the interface $h$-AgFEM method is available at `FEMPAR` [19]. `FEMPAR` is linked against `p4est` v2.2 [43], as the octree Cartesian grid manipulation engine, and `PETSc` v3.11.1 [25] distributed-memory linear algebra data structures and solvers. All software is compiled with Intel v18.0.5 compilers using system recommended optimisation flags and linked against the Intel MPI Library (v2018.4.057), for message-passing, and the BLAS/LAPACK available on the Intel MKL library, for optimised dense linear algebra kernels. All floating-point operations are performed in IEEE double precision. Additionally, condition number estimates are computed outside `FEMPAR` with `MATLAB` function *condest*.[3]

### 4.4.3 Convergence tests

We study the convergence of interface AgFEM in two stages. In the first one, we choose benchmark (4.14) and examine the rate at which the relative energy norm error decays with uniform mesh refinements. In a second stage, we consider the remaining benchmarks and observe the behaviour for both uniform and error-driven $h$-adaptive mesh refinements. All experiments run on five MN-IV nodes, i.e. we use a total of 240 CPUs, with each CPU mapped to a different MPI task.

For the first part, we consider the circle, flower, popcorn and spiral interface geometries. In the first three cases, the level sets are centred at the origin of coordinates and the physical domain is the unit $[0,1]^3$ cube, while the physical domain of the spiral case is the $[-1,1]^2 \times [0,2]$ cuboid. In all cases, the interface cuts the external boundary. Besides, the circle has radius 0.7 and the flower level-set function in polar coordinates is $\varphi(r,\theta) = r - 0.7(1 + 0.3\sin(5\theta))$. We refer to [20, 35] for the remaining level-set function expressions. The cuboid is initially meshed with a uniform Cartesian grid. Figure 4.7 gathers all convergence tests on uniform meshes for problem (4.14). In agreement to Proposition 4.3.5, we observe that AgFEM consistently recovers optimal convergence rates in the $H^1$-seminorm (equivalent to the energy norm) for all cases considered, including first and second order interpolations and extreme material contrasts.

[3] MATLAB is a trademark of THE MATHchapterS INC.




(a) Disk (b) Flower

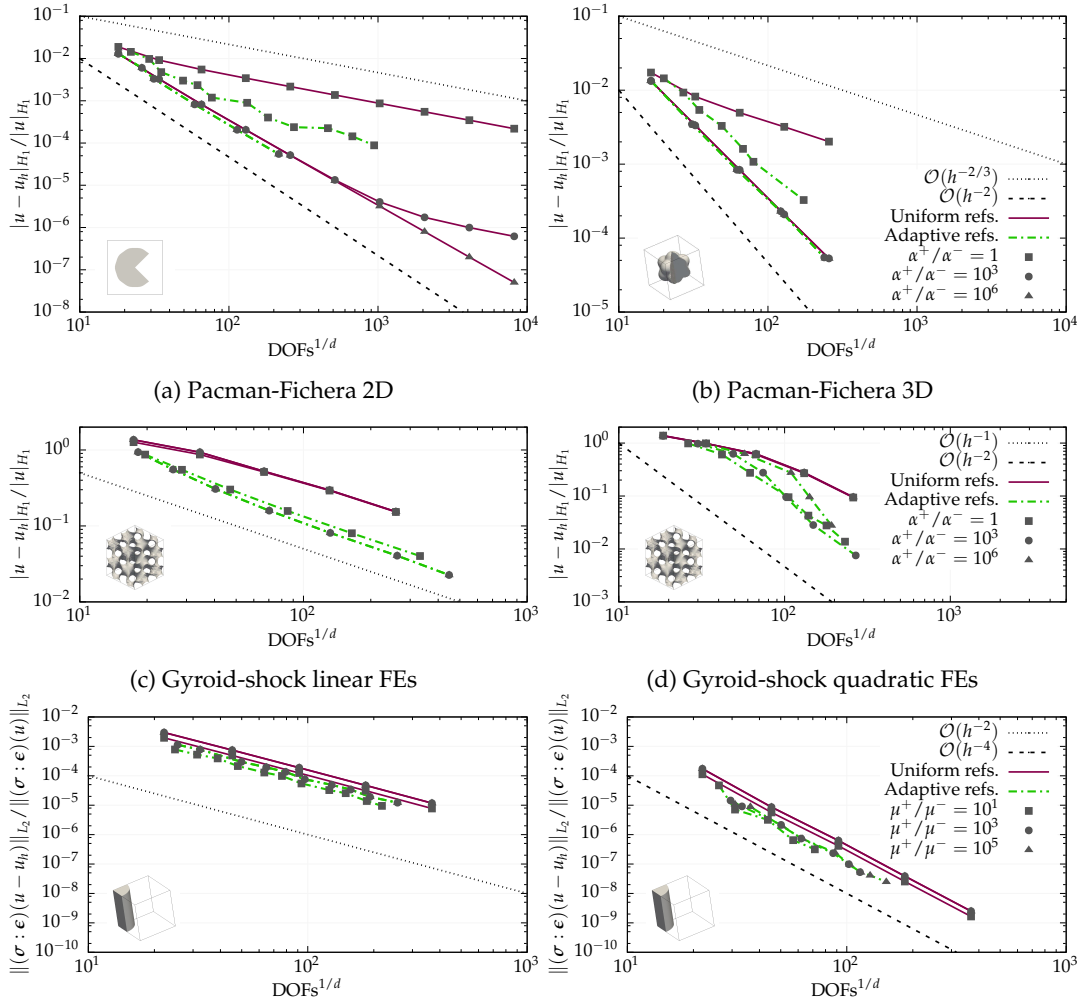
(c) Popcorn (d) Spiral

Figure 4.7: Convergence tests on uniform meshes: For benchmark (4.14) and an initial uniform mesh, AgFEM consistently shows optimal convergence rates as the mesh is uniformly refined.

For tests with uniform and $h$-adaptive mesh refinements, we consider (a) the Fichera-corner (4.15) on the pacman (2D) and popcorn-pacman (3D) shapes, (b) the single-shock (4.16) on the gyroid and (c) the cylindrical inclusion (4.17) on a cylinder. The physical domains are $[0,1]^d$, $[-2,2]^3$ and $[0,1]^3$, resp. Geometry and numerical solutions for each case are represented in Figures 4.5, 4.4(f) and 4.6. We note that, in (a) the interface is in the interior of $\Omega$, while in (c) we exploit radial symmetry. As shown in Figure 4.8, optimal convergence rates are retained with AMR, regardless of extreme material contrast values and order of approximation. We further justify this result in the discussion below.

Even though the solution to the Fichera-corner does not depend on material parameters, convergence rates do. In the Fichera case with uniform refinements, global error decreases at a rate of 2:3, when discrete error concentrates in $\Omega^-$, since $u^- \in H^{1+\frac{2}{3}}(\Omega^-)$ has limited regularity. Conversely, standard convergence rates hold, when discrete error concentrates in $\Omega^+$, where $u^+$ is smooth. Material contrast regulates which side of $\Gamma$ initially contributes more to numerical error, although when $h \to 0$ global error always converges at the slowest rate. We see that, for $k^+/k^- = 1$, global error clearly concentrates in $\Omega^-$, while it concentrates in $\Omega^+$ for $k^+/k^- = 10^6$. For an intermediate value, e.g. $k^+/k^- = 10^3$, discrete error initially concentrates in $\Omega^+$, but for $h$ small enough it shifts to $\Omega^-$.

As expected, $h$-adaptive refinements eliminate the influence of regularity of $u^-$ on the convergence rates. However, as shown in Figure 4.5, different values of the material contrast produce different refinement patterns, in consistence with the discrete error distribution, as discussed above. In particular, mesh refinements concentrate in $\Omega^-$ (or $\Omega^+$), when $k^+/k^-$ is small (or large).




(a) Pacman-Fichera 2D

(b) Pacman-Fichera 3D

(c) Gyroid-shock linear FEs

(d) Gyroid-shock quadratic FEs

(e) Cylinder-elasticity linear FEs

(f) Cylinder-elasticity quadratic FEs

Figure 4.8: Convergence tests on $h$-adaptive meshes: (a)-(b) h-adaptivity test with the Fichera-corner problem (4.15) for quadratic FEs: AgFEM reproduces the behaviour of standard FEM in body-fitted meshes, i.e. convergence rates with uniform refinements is limited by solution regularity, whereas optimal convergence rates are restored with AMR. (c)-(d) h-adaptivity test with the single-shock problem (4.16) on the gyroid: $h$-AgFEM holds (asymptotically) optimal convergence rates. (e)-(f) h-adaptivity test with the cylindrical inclusion problem (4.17) on the cylinder: energy norm error using $h$-AgFEM decays at optimal *superconvergent* rates.

Since the single-shock case in the gyroid is rather intricate, convergence rates are initially slower than usual; optimal convergence rates are reached asymptotically (especially for quadratic FEs). We observe that, in front of uniform refinements, AMR is capable of entering faster into the asymptotic regime. However, the pace at which this is achieved depends on material contrast. In particular, larger values of $k^+/k^-$ slow down reaching optimal rates.

Apart from that, results for the linear elasticity problem also deserve attention. We identify that the energy norm of the error decreases at a rate of 1:2 for linear FEs and 1:4 for quadratic FEs. This means we obtain superconvergence ($\mathcal{O}(h^{q+1})$) for linear FEs and

ultraconvergence ($\mathcal{O}(h^{q+2})$) for quadratic FEs. Although we do not have conclusive evidence, we believe this behaviour is explained by the well-known fact that Gauss-Legendre quadrature points on hexahedral cells are superconvergent stress recovery points [201]. In our case, when the cell is not intersected by $\Gamma$, local errors $\|(\boldsymbol{\sigma} : \boldsymbol{\varepsilon})(\boldsymbol{u} - \boldsymbol{u}_h)\|_{L^2(\Omega)}$ are integrated with standard Gauss-Legendre quadrature rules. As a result, even though the approximated solution is not superconvergent, local error is computed at points that are superconvergent. In contrast, quadrature rules are locally modified in cut cells, as usual in unfitted FE methods [35]; thus, local errors in those cells are not computed at superconvergent points. In spite of this, it is clear from the convergence plots that the behaviour of the global error in the energy norm is not influenced by cut cells, i.e. global error retains the local superconvergence property that (only) holds in non-cut cells.

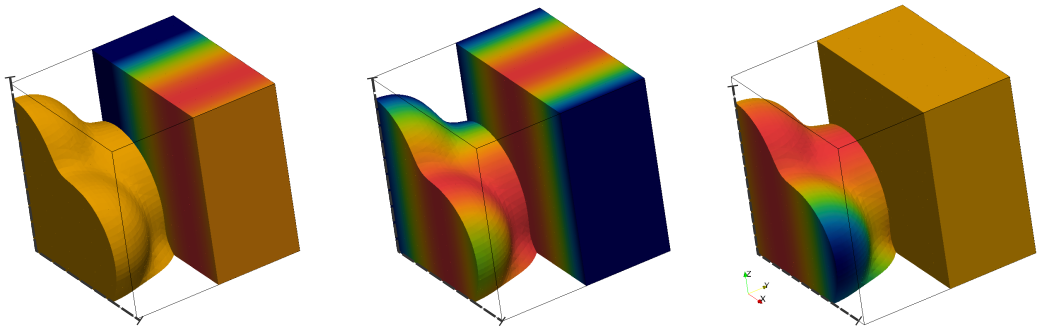
### 4.4.4 Robustness to cut location and material contrast

For the sensitivity of AgFEM to cut location and material contrast, we restrict ourselves to the Poisson benchmark (4.14) in the flower and popcorn interfaces and the linear elasticity benchmark (4.17) in the cylinder.

Our approach is similar to the one in [9]. It consists in carrying out a batch of simulations in a biparametric space, considering different material contrast and cut configurations, as shown in Figure 4.9. The procedure is as follows. We start with a reference simulation in a unit cube $[0,1]^d$, that takes $k^+/k^- = 1$ for (4.14), or $\mu_+/\mu_- = 1$ for (4.17). The unit cube is uniformly meshed with cell size $h = 2^{-6+q}$ for (4.14), and $h = 2^{-5+q}$ for (4.17), where $q$ is the FE interpolation order. The material perturbation simply consists in varying the material contrast $k^+/k^-$ or $\mu_+/\mu_-$ of the reference simulation in the interval $[10^{-6}, 10^6]$. On the other hand, to produce different cut configurations, the unit cube is scaled to $[0, 1+ah]^d$, where $a \in [-1, 1]$. We remark that the number of mesh cells is kept constant, i.e. the cell size after scaling is $\hat{h} = (1+ah)h$.

Given this setting, we launch simulations with AgFEM for different pairs of $(k^+/k^-, a)$ or $(\mu_+/\mu_-, a)$, until we sweep the range $[10^{-6}, 10^6] \times [-1, 1]$. We consider both serial and parallel computations; the latter are carried out in a single MN-IV node, i.e. 48 tasks. Along the sweep, we gather $H^1$-seminorm errors and condition number estimates. Afterwards, we condense the results into colour maps that plot the values each of these quantities in the $(k^+/k^-, a)$ or $(\mu_+/\mu_-, a)$ planes. We discuss next some of the results obtained with this procedure, represented in Figures 4.10, 4.11 and 4.12.

As seen in Figure 4.10, numerical errors in the $H^1$-seminorm are barely sensitive to material contrast and cut location. This behaviour is consistently observed in all three cases and linear/quadratic FEs. Although, for the linear elasticity case (4.17), the error decreases one order of magnitude around $\mu_+/\mu_- = 1$, this is attributed to the fact that the solution is more regular when $\mu_+/\mu_- = 1$ (it does not have a kink), not to the material contrast.



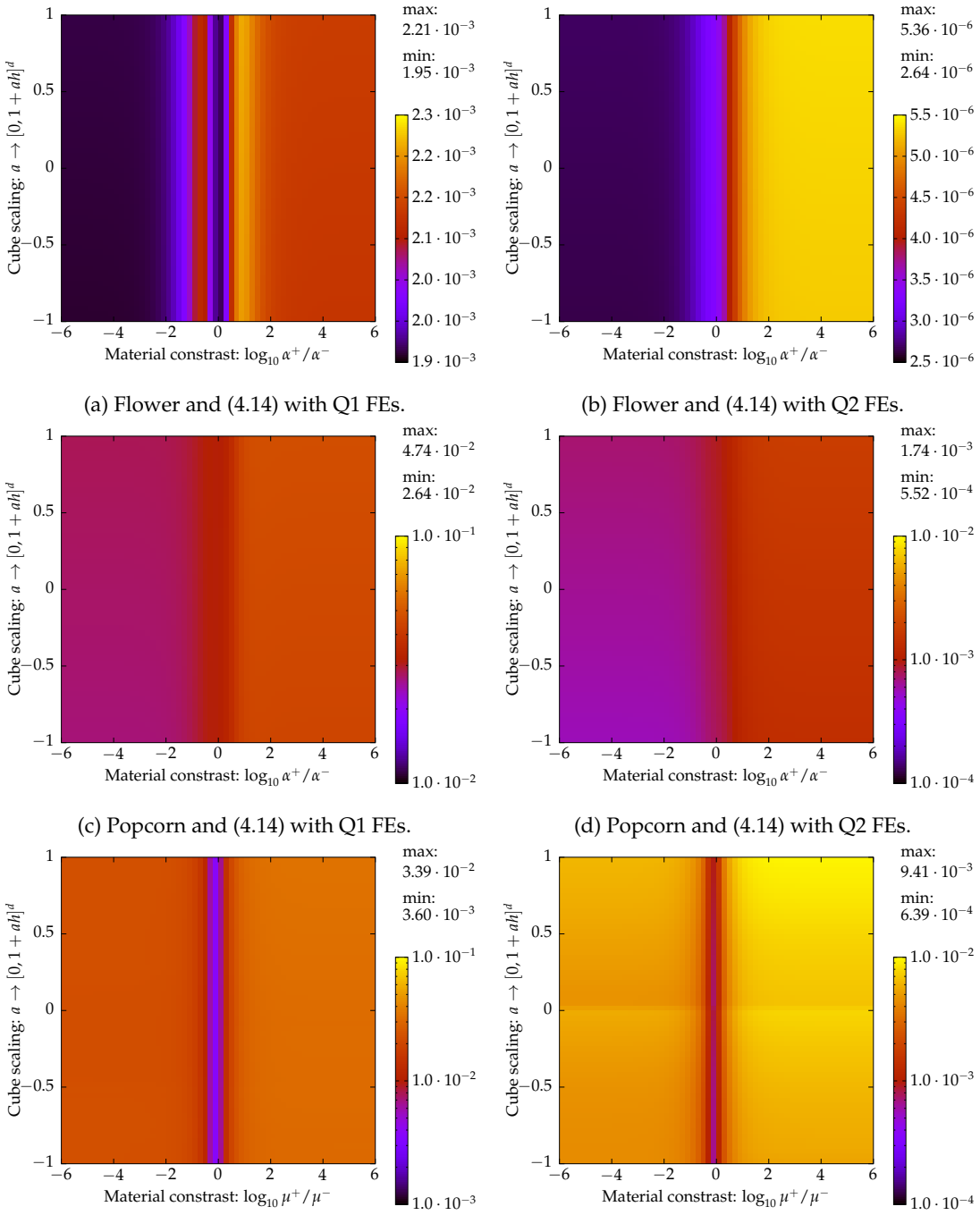
(a) $k^+/k^- = 10^{-6}$ and $a = -1$ (b) $k^+/k^- = 1$ and $a = 0$ (c) $k^+/k^- = 10^6$ and $a = 1$

Figure 4.9: Illustration of the approach to study robustness to cut location and material contrast on the popcorn interface. Note that we only show the right half of the subdomain outside the popcorn flake. To study sensitivity to material contrast, we vary $k^+/k^-$ between $10^{-6}$ and $10^6$. To study sensitivity to cut location, we produce different cut locations by uniformly shrinking (Figure 4.9(a)) or stretching (Figure 4.9(c)) the physical domain with a parameter $a \in [-1, 1]$ (dashed lines show the $x$ and $z$ dimensions of the cube represented in Figure 4.9(b), as reference to compare the different cube scalings).

In Figure 4.11, we plot condition numbers obtained with one of the three cases, namely the Poisson equation (4.14) on the popcorn interface. We have additionally swept the parametric space with StdFE, for comparison with AgFE; it clearly illustrates the effect of the latter on the conditioning of the matrix. As shown in Figures 4.11(a) and 4.11(b), the condition number of the linear system is extremely high for StdFE and shows a predominant dependence on the cut configuration. The problem can be so ill-conditioned that the local eigenvalue solver to compute $\beta$ breaks down. In contrast, AgFEM is fully robust and brings down condition numbers to values that the solvers can cope with, see Figures 4.11(c) and 4.11(d). Besides, dependence on cut location vanishes completely, although there is a clear sensitivity to material contrast. Nonetheless, this dependence is not present in the condition number of the *diagonally-scaled* system matrix. Indeed, as seen in Figures 4.12(a) and 4.12(b), the condition number after diagonal scaling becomes barely sensitive to both cut location and material contrast. Furthermore, condition numbers are around $\mathcal{O}(10^4)$, in the worst case, which is a rather low value for unfitted 3D+Q2 simulations. The same outcome is observed for the linear elasticity case, as shown in Figures 4.12(c) and 4.12(d).

### 4.4.5 Weak-scaling analysis

We carry out weak-scaling tests for three $h$-adaptive cases studied in the convergence tests: (1) the Pacman-Fichera 3D with quadratic FEs for $k^+/k^- = 1$ (Figure 4.8(b)) and the gyroid-shock with (2) linear (Figure 4.8(c)) and (3) quadratic (Figure 4.8(d)) FEs for $k^+/k^- = 10^3$. In the analysis, we aim (a) to deploy a testing methodology that accounts for the fact that cells (and DOFs) that cut the interface are replicated and (b) to demonstrate that both the cell aggregation scheme and the set up of the AgFE




(a) Flower and (4.14) with Q1 FEs.

(b) Flower and (4.14) with Q2 FEs.

(c) Popcorn and (4.14) with Q1 FEs.

(d) Popcorn and (4.14) with Q2 FEs.

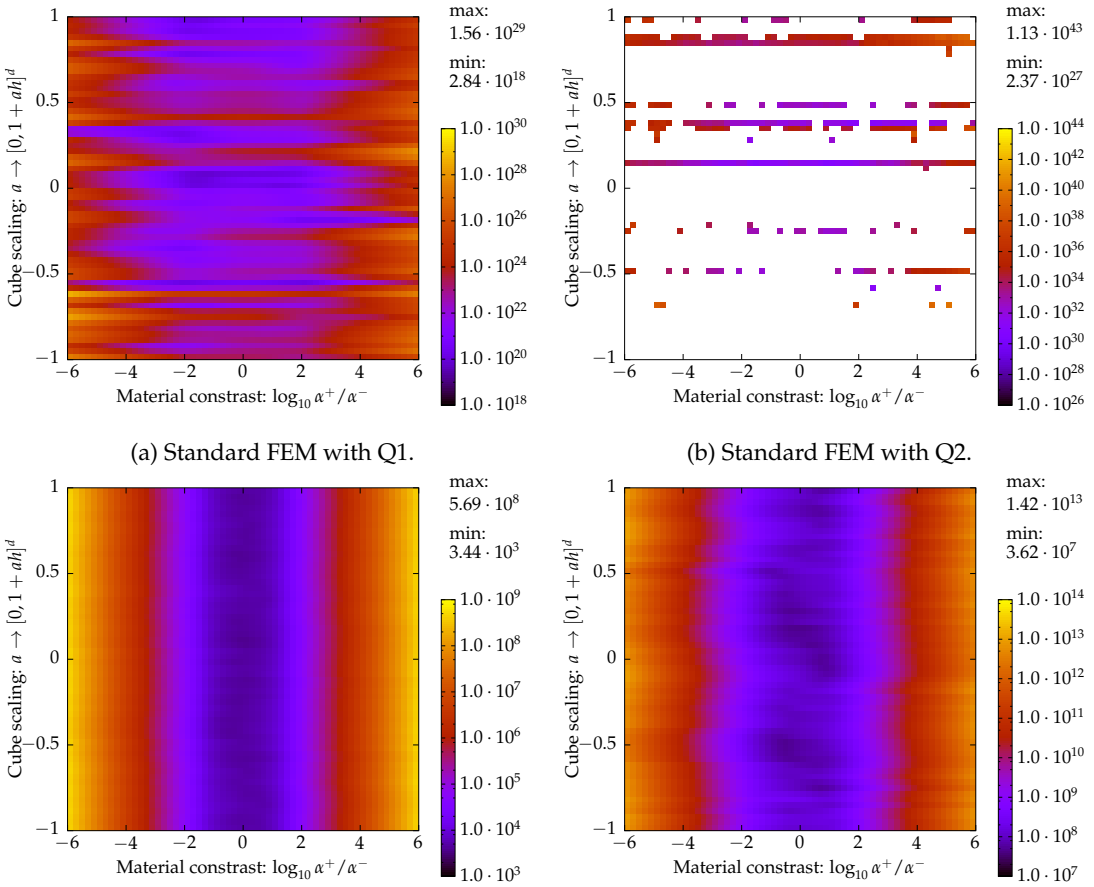
(e) Cylinder and (4.17) for Q1.

(f) Cylinder and (4.17) for Q2.

Figure 4.10: Sensitivity test of Ag FEM w.r.t. material contrast and cut location: For the cases described in Section 4.4.4, the $H^1$-seminorm relative error, i.e. $|u - u_h|_{H^1} / |u|_{H^1}$, is barely sensitive to material contrast and cut location.

space $\mathcal{V}_h$ are computationally (weakly) scalable. In the sequel we use $N_\square$ and $n_\square$ to denote global (i.e. referring to the whole mesh/domain) and local (i.e. referring to the processor-owned submesh/subdomain) sizes/cardinalities of a quantity $\square$.

Our strategy is analogous to the one detailed in Chapter 3; it consists in repeating




(a) Standard FEM with Q1.

(b) Standard FEM with Q2.

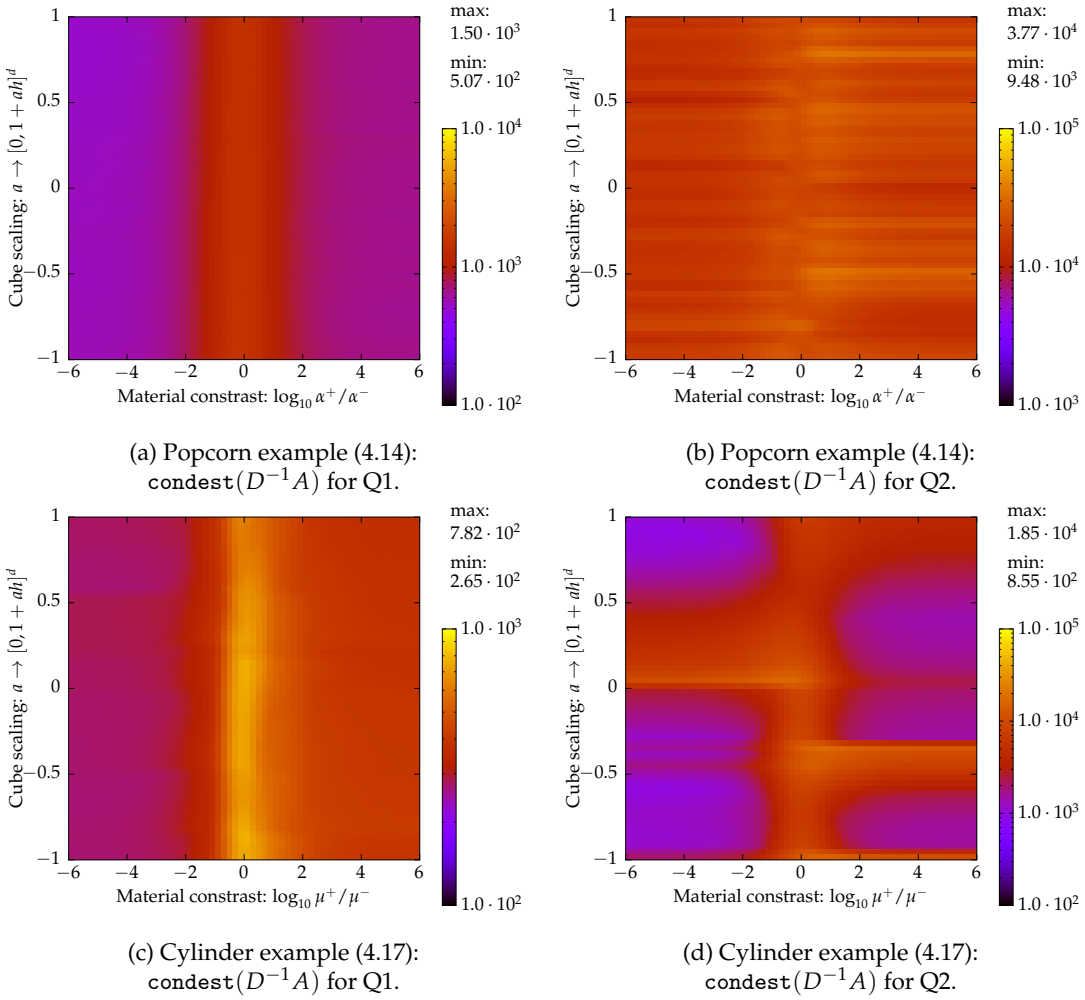
(c) AgFEM with Q1.

(d) AgFEM with Q2.

Figure 4.11: Sensitivity test w.r.t. material contrast and cut location for popcorn example. Examination of $\texttt{condest}(A)$ exposes how lack of robustness and dependency on cut location in standard FEM is not present in AgFEM.

the convergence test, adjusting the number of processors to compute each point in the error plot. The goal is to impose that a suitable quantity remains (approximately) invariant across the whole convergence test. In addition, given a point, it is desirable that the invariant *also* holds across processors, in order to reduce noise in the results due to interprocessor imbalance. In FE simulations, the typical invariant is the (local) number of (free) DOFs each processor owns, since complexity of major phases (e.g. solving the linear system) depends on the number of DOFs. However, it is difficult to balance DOFs across processors in our meshes, which have both free and (hanging and ill-posed) constrained DOFs that overlap at the interface. For this reason, we choose as invariant the local number of active cells $n_{\text{act,cells}}$, where the global counterpart is $N_{\text{act,cells}} = N_{\mathcal{T}^+_{h,\text{act}}} + N_{\mathcal{T}^-_{h,\text{act}}}$, i.e. the number of cells in $\mathcal{T}_h$, but counting cells at the interface twice.

According to this, we consider the sequence of optimal AMR meshes, obtained in




(a) Popcorn example (4.14): $\texttt{condest}(D^{-1}A)$ for Q1.

(b) Popcorn example (4.14): $\texttt{condest}(D^{-1}A)$ for Q2.

(c) Cylinder example (4.17): $\texttt{condest}(D^{-1}A)$ for Q1.

(d) Cylinder example (4.17): $\texttt{condest}(D^{-1}A)$ for Q2.

Figure 4.12: In AgFEM, condition number of the diagonally-scaled system matrix, i.e. $\texttt{condest}(D^{-1}A)$, does not depend on cut location or material contrast and is effectively controlled; all condition numbers are down to $\mathcal{O}(10^4)$, in the worst case.

the convergence test, and compute the number of processors for the weak-scaling analysis as

$$P^i = P^1 \left\lfloor \frac{N^i_{\text{act,cells}}}{N^1_{\text{act,cells}}} \right\rfloor, \; i > 1,$$

where superscript $i > 1$ refers to each element in the sequence of optimal meshes (points in the error curve), $P^1$ is a fixed initial number of processors and $\lfloor \cdot \rfloor$ is the *floor* function; given a real number $x$, $\lfloor x \rfloor$ is the greatest integer less than or equal to $x$. Table 4.2 gathers the sequences $\left\{P^i\right\}_{i>1}$ obtained following this procedure, for the three cases that are studied in this section. We observe that (1) it is clearly more straightforward to equally distribute active cells among processors than DOFs and (2) the (average) local number of free DOFs grows mildly with $i > 1$. Hence, this approach allows us to (conservatively) examine how the problem scales with DOFs, avoiding cumbersome strategies to balance DOFs.

| Pacman-Fichera 3D AMR-Q2 and $n_{\text{act,cells}} \approx 1.2k$ | | | | | | |
|---|---|---|---|---|---|---|
| $P$ | 1 | 5 | 14 | 36 | 58 | 457 |
| $N_{\text{act,cells}}$ | 1.2k | 5.8k | 16k | 43k | 67k | 533k |
| $N_{\text{dofs}}$ | 8.0k | 42k | 118k | 315k | 510k | 4,031k |
| $n_{\text{dofs}}$ | 8.0k | 8.3k | 8.3k | 8.6k | 8.6k | 8.8k |
| Gyroid-shock AMR-Q1 and $n_{\text{act,cells}} \approx 46k$ | | | | | | |
| $P$ | 2 | 9 | 57 | 556 | 2150 | |
| $N_{\text{act,cells}}$ | 92k | 440k | 2,637k | 19,471k | 98516k | |
| $N_{\text{dofs}}$ | 66k | 348k | 2,288k | 18,056k | 89,822k | |
| $n_{\text{dofs}}$ | 33k | 36k | 40k | 42k | 42k | |
| Gyroid-shock AMR-Q2 and $n_{\text{act,cells}} \approx 4.7k$ | | | | | | |
| $P$ | 1 | 4 | 13 | 33 | 99 | 556 |
| $N_{\text{act,cells}}$ | 4.7k | 19k | 62k | 157k | 474k | 2,641k |
| $N_{\text{dofs}}$ | 27k | 118k | 409k | 1,065k | 3,306k | 19,430k |
| $n_{\text{dofs}}$ | 27k | 30k | 32k | 32k | 33k | 35k |

Table 4.2: Number of subdomains $P$, global active cells $N_{\text{act,cells}}$, global DOFs $N_{\text{dofs}}$ and local DOFs $n_{\text{dofs}}$ for the cases considered in the weak scaling tests of Figure 4.13. For each case, local active cells $n_{\text{act,cells}}$, remains quasi-constant with $P$. Besides, $n_{\text{dofs}}$ (slowly) increases monotonically.

Once established the weak-scaling methodology, our purpose is to show that remarkable scalability of ($h$-adaptive) AgFEM, reported in previous chapters for problems with unfitted boundary [15, 187], is preserved for interface problems. As those chapters have already addressed weak scalability of the whole FE simulation pipeline, we focus on reporting wall clock times spent in the two main AgFEM-specific phases, i.e. those phases particular of our approach, not present in other unfitted techniques. The two phases are (1) cell aggregation, see Section 4.2.2, and (2) setup of the AgFE space, see Section 4.2. As finding the optimal mesh for each $i > 1$ is an iterative AMR process, we only monitor these quantities for the optimal mesh (last iteration). We note that, even though (1) and (2) are critical phases of the simulation, from the computational viewpoint, they are not the most prominent ones. Thus, AgFEM does not affect much overall run time with respect to a standard (ill-posed) Galerkin method.

To allocate the MPI tasks in the MN-IV supercomputer, we resort to the default task placement policy of Intel MPI (v2018.4.057) with partially filled nodes. For each point of the test, the number of nodes $N^i$ is selected as $N^i = \lceil P^i/48 \rceil$, where $\lceil \cdot \rceil$ is the *ceiling* function; given a real number $x$, $\lceil x \rceil$ is the smallest integer more than or equal to $x$. If $P^i$ is not multiple of 48, the placement policy fully populates the first $N-1$ nodes with 48 MPI tasks per node; the remaining $P^i - 48(N-1)$ MPI tasks are mapped to the last node.

Figure 4.13 gathers all the quantities surveyed in weak scaling tests. The main phases of $h$-adaptive AgFEM exhibit remarkable scalability for the three cases considered. We observe that the number of local active cells $n_{\text{act,cells}}$ and DOFs $n^i_{\text{dofs}}$, $i > 1$ for the gyroid-shock AMR-Q1 case are significantly larger than for the other two cases.

That is why this case yields the largest computational times.

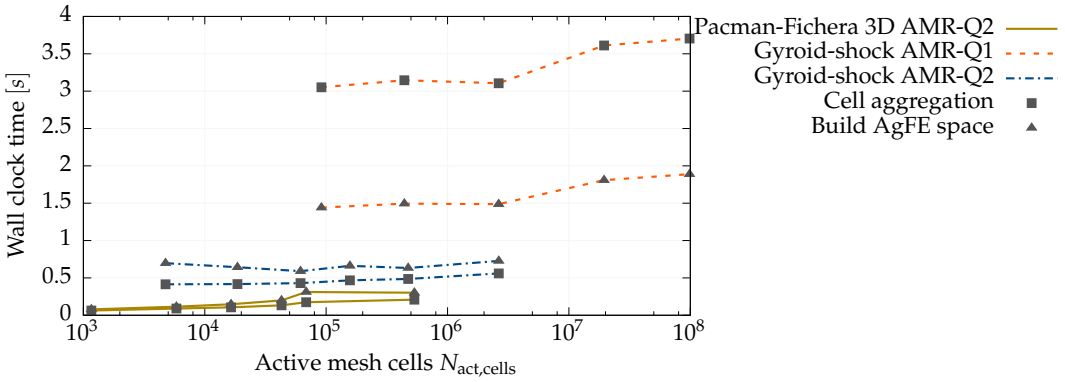



Figure 4.13: Weak scaling tests on selected interface problems from convergence tests in Section 4.4.3 up to 2,150 MPI tasks, as reported in Table 4.2.

## 4.5 Conclusions

This chapter addressed a novel $h$-adaptive aggregated FE method for large-scale (unfitted) interface elliptic boundary value problems. Our methodology is grounded on the well-established approach of weakly coupling interface-overlapping discretisations [88] and the recently developed $h$-adaptive AgFE method (Chapter 3) for unfitted boundary elliptic problems. The study of the new method is accompanied with complete theoretical characterisation and thorough numerical experimentation on a suite of Poisson and linear elasticity ($hp$-FEM) benchmarks with complex interface shapes.

As main contributions of the chapter, we have introduced a (a) natural extension of the (distributed-memory) cell aggregation algorithm in [187] for $n$-interface problems. We have shown that (b) AgFE spaces easily blend to the typical Cartesian-product approximation structures of interface-overlapping meshes. We have proven (c) well-posedness and optimal approximation properties of a SIPM-AgFEM discrete formulation for the irreducible linear elasticity problem. Robustness to cut location is ensured, by inheriting cut-independent estimates from AgFEM in unfitted boundaries, while robustness to material contrast is achieved, by using the same weighted average of body-fitted DG methods. Besides, the resulting method admits (d) straightforward implementation on top of an existing large-scale implementation of AgFEM for unfitted boundary problems. To conclude, exhaustive numerical tests have exposed (e) optimal ($h$-adaptive) approximation capability, robustness to cut location and material contrast and remarkable scalability on parallel adaptive Cartesian tree-based meshes.

Our study offers compelling insight and evidence of the potential of AgFEM as an effective large-scale FE solver for complex multiphase and multiphysics problems modelled by PDEs. Extension to any of those problems is object of future work. Additionally, the chapter provides useful guidance in applying other unfitted CG methods to interface problems, especially those relying on cell aggregation.

# Chapter 5

# A scalable parallel finite element framework for growing geometries

*The contents of this chapter correspond to the research publication*

[143] EN, S. BADIA, A. F. MARTÍN AND M. CHIUMENTI, *A scalable parallel finite element framework for growing geometries. Application to metal additive manufacturing*, International Journal for Numerical Methods in Engineering, 119 (2019), pp. 1098-1125.

This chapter introduces an innovative parallel, fully-distributed finite element framework for growing geometries and its application to metal additive manufacturing. It is well-known that virtual part design and qualification in additive manufacturing requires highly-accurate multiscale and multiphysics analyses. Only high performance computing tools are able to handle such complexity in time frames compatible with time-to-market. However, efficiency, without loss of accuracy, has rarely held the centre stage in the numerical community. Here, in contrast, the framework is designed to adequately exploit the resources of high-end distributed-memory machines. It is grounded on three building blocks: (1) Hierarchical adaptive mesh refinement with octree-based meshes; (2) a parallel strategy to model the growth of the geometry; (3) state-of-the-art parallel iterative linear solvers. Computational experiments consider the heat transfer analysis at the part scale of the printing process by powder-bed technologies. After verification against a 3D benchmark, a strong-scaling analysis assesses performance and identifies major sources of parallel overhead. A third numerical example examines the efficiency and robustness of (2) in a curved 3D shape. Unprecedented parallelism and scalability were achieved in this work. Hence, this framework contributes to take on higher complexity and/or accuracy, not only of part-scale simulations of metal or polymer additive manufacturing, but also in welding, sedimentation, atherosclerosis, or any other physical problem where the physical domain of interest grows in time.

## 5.1 Introduction

AM, broadly known as 3D Printing, is introducing a disruptive design paradigm in the manufacturing landscape. The key potential of AM is the ability to cost-effectively

create *on-demand* objects with complex shapes and enhanced properties, that are near impossible or impractical to produce with conventional technologies, such as casting or forging. Adoption of AM is undergoing an exponential growth lead by the aerospace, defence, medical and dental industries and the prospect is a stronger and wider presence as a manufacturing technology [189].

Nowadays, one of the main showstoppers in the AM industry, especially for metals, is the lack of a software ecosystem supporting fast and reliable product and process design. Part qualification is chiefly based on slow and expensive trial-and-error physical experimentation and the understanding of the process-structure-performance link is still very obscure. This situation precludes further implementation of AM and it is a call to action to shift to a virtual-based design model, based on predictive computer simulation tools. Only then will it be possible to fully leverage the geometrical freedom, cost efficiency and immediacy of this technology.

This chapter addresses the numerical simulation of metal AM processes through *High Performance Computing (HPC) tools*. The mathematical modelling of the process involves dealing with multiple scales in space (e.g. part, melt pool, microstructure), multiple scales in time (e.g. microseconds, hours), coupled multiphysics [63, 107] (e.g. thermomechanics, phase-change, melt pool flow) and arbitrarily complex geometries that grow in time. As a result, high-fidelity analyses, which are vital for part qualification, can be extremely expensive and require vast computational resources. In this sense, HPC tools capable to run these highly accurate simulations in a time scale compatible with the time-to-market of AM are of great importance. By efficiently exploiting HPC resources with scalable methods, one can drastically reduce CPU time, allowing the optimization of the AM building process and the virtual certification of the final component in reasonable time.

Experience acquired in modelling traditional processes, such as casting or welding [48, 53, 122], has been the cornerstone of the first models for metal AM processes [8, 33, 109, 127, 159]. At the part scale, FE modelling has proved to be useful to assess the influence of process parameters [150], compute temperature distributions [54, 69], or evaluate distortions and residual stresses [72, 126, 154]. Recent contributions have introduced microstructure simulations of grain growth [121, 160] and crystal plasticity [106], melt-pool-scale models [107, 195] and even multiscale and multiphysics solvers [104, 124, 165, 194, 196]. Furthermore, advanced frameworks (e.g. grounded on multi-level *hp*-FE methods combined with implicit boundary methods [108]) or applications to topology optimization [175] have also been considered.

However, in spite of the active scientific progress in the field, the authors have detected that very little effort has turned to the design of *large scale* FE methods for metal AM. Even if computational efficiency has been taken into consideration in several works, all approaches have been limited to AMR [67, 151] or simplifications [55, 92, 97, 169] that sacrifice the accuracy of the model. Parallelism and scalability has been generally disregarded (with few exceptions [121, 140]), even if it is fundamental to face

more complexity and/or provide increased accuracy at acceptable CPU times. For instance, for the high-fidelity melt-pool solver in [195], a simulation of 16 ms of physical time with 7 million cells requires 700 h of CPU time on a common desktop with an Intel Core i7-2600.

The purpose of this chapter is to design a novel scalable parallel FE framework for metal AM *at the part scale*. Our approach considers three main building blocks:

1. *Hierarchical AMR with octree meshes* (see Section 5.2). The dimensions of a part are in the order of [mm] or [cm], but relevant physical phenomena tend to concentrate around the melt pool [µm]. Likewise, in powder-bed fusion, the layer size is also [µm], i.e. the scale of growth is much smaller than the scale of the part. Hence, adaptive meshing can be suitably exploited for the highly-localized nature of the problem. Here, the parallel octree-based *h*-adaptive FE framework of Chapter 2 is established.

2. *Modelling the growth of the geometry* (see Section 5.3). In welding and AM processes, the addition of material into the part has been typically modelled by adding new elements into the computational mesh. To this effect, the simulation starts with the generation of a *static* background mesh comprising the substrate and the filling, i.e. the final part. Common practice in the literature essentially considers two different techniques [51, 136]: *quiet*-element method and element-birth method (EBM). The only difference among them is how they treat the elements in the filling at the start of the simulation. While the former assigns to them penalized material properties, perturbing the original problem; in the latter, they have no DOFs. At each time step, the computational mesh is updated: elements inside the incremental growth region are found with a search operation and assigned the usual material properties or new degrees of freedom, respectively. This work extends the EBM to (1). The parallelization approach is designed such that it can be accommodated in a general-purpose FE code. It only requires two interprocessor communications and it is completed with an efficient and embarrassingly parallel intersection test *for rectangular bodies* to drive the update of the computational mesh (Section 5.3.2). Finally, a strategy is devised to balance the computational load among processors, during the growth of the geometry (Section 5.3.3). The parallel and adaptive EBM is central to a parallel FE framework for growing domains and constitutes the main novelty of this work.

3. *State-of-the-art parallel iterative linear solvers*. Compared to sparse direct solvers, iterative solvers can be efficiently implemented in parallel computer codes for distributed-memory machines. However, they must also be equipped with efficient preconditioning schemes, i.e. preconditioners able to keep a bound of the condition number, independent of mesh resolution, while still exposing a high degree of parallelism. Examples of preconditioners that the framework is able to use include balancing domain decomposition by constraints (BDDC) [17] or

AMG [56], although they are not actually exploited in this work, for reasons made clear in Section 5.4, related to the application problem.

As the originality of the framework centres upon the computational aspects to efficiently deal with growing geometries, (1) and (2) are presented in an abstract way, i.e. without considering a reference physical problem. Afterwards, Section 5.4 considers an application to the heat transfer analysis of metal AM processes by powder-bed fusion. Nonetheless, the authors believe that the framework can readily be extended to a coupled thermomechanical analysis, as long as proper treatment of history variables within the AMR framework can be guaranteed. Likewise, it may also be adapted to other metal or polymer technologies, or even be useful to other domains of study, such as the simulation of sedimentation processes.

Computer implementation was supported by `FEMPAR` [19, 180], a general-purpose object-oriented multi-threaded/message-passing scientific software for the fast solution of multiphysics problems governed by PDEs. `FEMPAR` adapts to a range of computing environments, from desktop and laptop computers to the most advanced HPC clusters and supercomputers. The `FEMPAR-AM` module for FE analyses of metal AM processes has been developed on top of this high-end infrastructure. Its main software abstractions are described in Section 5.5. The exposition is intended to help in the customization of any general-purpose FE code for growing domains.

The numerical study of the framework in Section 5.6 starts with a verification of the thermal FE model against a well-known 3D benchmark. Validation of this heat transfer formulation has already been object of previous works [54, 55] and it is not covered here. A strong-scaling analysis follows in Section 5.6.2 to analyse the performance of the computer implementation and expose sources of load imbalance, identified as a major parallel overhead threatening the efficiency of the implementation. The simulation considers the printing of 48 layers in a cuboid, one layer printed per time step (followed by an interlayer cooling step). A relevant outcome is the capability of simulating the printing and cooling of a single layer (two linearized time steps) in a 10 million unknown problem in merely 2.5 seconds average with 6,144 processors. The last example in Section 5.6.3 considers a curved 3D shape and follows the actual laser path point-to-point. A second order adaptive mesh with no geometrical error is transformed during the simulation to accommodate the laser path. Regardless of having nonrectangular cells, the parallel EBM is capable of tracking the laser path, as long as some quasi-rectangularity conditions hold. In conclusion (see Section 5.7), the fully-distributed, parallel framework presented in this chapter is set to contribute to the efficient simulation of AM processes, a critical aspect long identified, though mostly neglected by the numerical community.

## 5.2 Mesh generation by hierarchical AMR

Physical phenomena are often characterized by multiple scales in both space and time. When the smallest ones are highly-localized in the physical domain of analysis, uniformly refined meshes tend to be impractical from the computational viewpoint, even for the largest available supercomputers.

The purpose of AMR is to reach a compromise between the high-accuracy requirements in the regions of interest and the computational effort of solving for the whole system. To this end, the mesh is refined in the regions of the domain that present a complex behaviour of the solution, while larger mesh sizes are prescribed in other areas.

In this work, the areas of interest are known *a priori* and correspond to the growing regions. It is assumed that the geometrical scale of growth is much smaller than the domain of study, as it is the case of welding or AM processes. Besides, the framework is restricted to $h$-adaptivity, i.e. only the mesh size changes among cells, in contrast to $hp$-adaptivity, where the polynomial order $p$ of the FEs may also vary among cells. This computational framework is briefly outlined in this section; the reader is referred to Chapter 2 for a thorough exposition.

### 5.2.1 Hierarchical AMR with octree meshes

Let us suppose that $\Omega \subset \mathbb{R}^d$ is an open bounded polyhedral domain, being $d = 2, 3$ the dimension of the physical space. Let $\mathcal{T}_h^0$ be a conforming and quasi-uniform partition of $\Omega$ into quadrilaterals ($d = 2$) or hexahedra ($d = 3$), where every $T \in \mathcal{T}_h^0$ is the image of a reference element $\hat{T}$ through a smooth bijective mapping $\Phi_T$. If not stated otherwise, these hypotheses are common to all sections of this document.

Hierarchical AMR is a multi-step process. The mesh generation consists in the transformation of $\mathcal{T}_h^0$, typically as simple as a single quadrilateral or hexahedron, into an objective mesh $\mathcal{T}_h$ via a finite number of refinement/coarsening steps; in other words, the AMR process generates a sequence $\mathcal{T}_h^0, \mathcal{T}_h^1, \ldots, \mathcal{T}_h^m \equiv \mathcal{T}_h$ such that $\mathcal{T}_h^i = \mathcal{R}(\mathcal{T}_h^{i-1}, \theta^i)$, $i = 1, \ldots, m < \infty$, where $\mathcal{R}$ applies the refinement/coarsening procedure over $\mathcal{T}_h^{i-1}$ and $\theta^i : \mathcal{T}_h^{i-1} \to \{-1, 0, 1\}$ is an array establishing the action to be taken at each cell: -1 for coarsening, 0 for "do nothing" and 1 for refinement.

A cell marked for refinement is partitioned into four (2D) or eight (3D) children cells by bisecting all cell edges. As a result, $\mathcal{T}_h$ can be interpreted as a collection of quadtrees (2D) or octrees (3D), where the cells of $\mathcal{T}_h^0$ are the roots of these trees, and children cells (a.k.a. quadrants or octants) branch off their parent cells. The leaf cells in this hierarchy form the mesh in the usual meaning, i.e. $\mathcal{T}_h$. Furthermore, for any cell $T \in \mathcal{T}_h$, $\ell(T)$ is the *level* of refinement of $T$ in the aforementioned hierarchy. In particular, $\ell(T) = 0$ for the root cells and $\ell(T) = \ell(\text{parent(T)}) + 1$, otherwise. The level can also be defined for lower dimensional mesh entities (vertices, edges, faces)

as in [99, Definition 2.4]. Figure 5.1 illustrates this recursive tree structure with cells at different levels of refinement stemming from a single root.

Octree meshes admit a very compact computer representation, based on Morton encoding [139] by bit interleaving, which enables efficient manipulation in high-end distributed-memory computers [43]. Moreover, they provide multi-resolution capability by local adaptation, as the leaves in the hierarchy can be at different levels of refinement. However, they are potentially nonconforming by construction, e.g. there can be *hanging* vertices in the middle of an edge or face or *hanging* edges or faces in touch with a coarser geometrical entity.

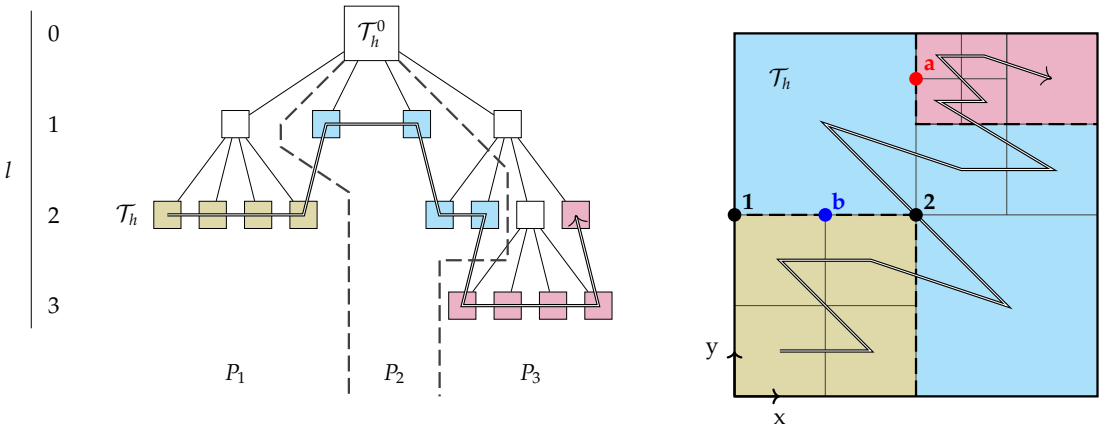



Figure 5.1: The hierarchical construction of $\mathcal{T}_h$ gives rise to a one-to-one correspondence between the cells of $\mathcal{T}_h$ and the leaves of a quadtree rooted in $\mathcal{T}_h^0$. $\mathcal{T}_h$ does not satisfy the *2:1 balance*, e.g. the red hanging vertex $a$ is not permitted. Assuming a discretisation with conforming Lagrangian linear FEs, the value of the DOF at hanging vertex $b$ is subject to the constraint $u_b = 0.5u_1 + 0.5u_2$. $\mathcal{T}_h$ is partitioned into three subdomains, $P1$, $P2$ and $P3$, using the $z$-order curve obtained by traversing the leaves of the octree in increasing Morton index. Adapted from [43].

Nonconformity introduces additional complexity in the implementation of conforming FEs, especially in parallel codes for distributed-memory computers. This degree of complexity is nevertheless significantly reduced by enforcing the so-called *2:1 balance* relation, i.e. adjacent cells may differ at most by a single level of refinement. In this sense, the mesh in Figure 5.1 violates the 2:1 balance, because hanging vertex $a$ is surrounded by cells that differ by two levels.

In order to preserve the continuity of a conforming FE approximation, DOFs sitting on *hanging* geometric entities cannot have an arbitrary value, they are constrained by neighbouring nonhanging DOFs. The approach advocated in this work to handle the hanging node constraints [111] consists in eliminating them from the local matrices, before assembling the global matrix. In this case, hanging DOFs are not associated to global DOFs and, thus, a row/column in the global linear system of equations.

### 5.2.2 Partitioning the octree mesh

Efficient and scalable parallel partitioning schemes for adaptively refined meshes are still an active research topic. Our computational framework relies on the `p4est` library [43]. `p4est` is a MPI library for efficiently handling (forests of) octrees that has scaled up to hundreds of thousands of cores [27]. Using the properties of the Morton encoding, `p4est` offers, among others, parallel subroutines to refine (or coarsen) the octants of an octree, enforce the 2:1 balance ratio and partition/redistribute the octants among the available processors to keep the computational load balanced [43, 99]. Data structures and algorithms involved in the interface between `p4est` and `FEMPAR` are detailed in Chapter 2. They are configured according to a two-layered meshing approach. The first or *inner* layer is a light-weight encoding of the forest-of-trees, handled by `p4est`. The second or *outer* layer is a richer mesh representation, suitable for generic finite elements.

The principle underlying the mesh partitioner is the use of space-filling curves. Octants within an octree can be naturally assigned an ordering by a traversal across all leaves, e.g. increasing Morton index, as shown in Figure 5.1. Application of the one-to-one correspondence between tree nodes and octants reveals that this one-dimensional sequence corresponds exactly to a $z$-order space-filling curve of the triangulation $\mathcal{T}_h$. Hence, the problem of partitioning $\mathcal{T}_h$ can be reformulated into the much simpler problem of subdividing a one-dimensional curve. This circumvents the parallel scaling bottleneck that commonly arises from *dynamic load balancing* via graph partitioning algorithms [101, 102].

However, the simplicity comes at a price related to the fact that, among space-filling curves, $z$-curves have unbounded locality and low bounding-box quality [90]. In our context, this leads to the emergence of poorly-shaped and, possibly, disconnected subdomains (with at most two components [42] for single-octree meshes). Bad quality of subdomains affects the performance of nonoverlapping DD methods [111].

## 5.3 Modelling the growth of the geometry

### 5.3.1 Parallel element-birth method

As mentioned in Section 5.1, the growth of the geometry is modelled in a *background* FE mesh that, even if it is refined or coarsened, always covers the same domain.

Let $\Omega(t)$ be a *growing-in-time* domain. During the time interval $[t_\mathrm{i}, t_\mathrm{f}]$, $\Omega(t)$ transforms from an initial domain $\Omega_\mathrm{i}$ to a final one $\Omega_\mathrm{f}$. For the sake of simplicity, AMR is not considered for now, only later in the exposition, and it is assumed that there exists (1) a *background* conforming partition $\mathcal{T}_h \equiv \mathcal{T}_h^0 = \{T\}$ of $\Omega_\mathrm{f}$, where $\Omega_\mathrm{f}$ and $\mathcal{T}_h^0$ satisfy the hypotheses stated in the first paragraph of Section 5.2.1, and (2) a time discretisation $t_\mathrm{i} = t_0 < t_1 < \ldots < t_{N_t} = t_\mathrm{f}$ such that, for all $j = 0, \ldots, N_t$, a partition $\mathcal{T}_{h,j}$ of $\Omega(t_j)$ can be obtained as a subset of cells of $\mathcal{T}_h$. In other words, a body-fitted mesh of $\Omega_\mathrm{f}$

can be built so that subsets of this mesh can be taken as body-fitted meshes of $\Omega(t_j)$, $j = 0, \ldots, N_t$. As $\Omega(t)$ grows in time, the relation $\mathcal{T}_{h,\mathrm{i}} = \mathcal{T}_{h,0} \subseteq \mathcal{T}_{h,1} \subseteq \ldots \subseteq \mathcal{T}_{h,N_t} = \mathcal{T}_{h,\mathrm{f}}$ holds.

This setting is typical in welding or AM simulations. In AM, for instance, it is frequently required that the mesh of the component conforms to the layers. On the other hand, the method presented below can be adapted with little effort to a more general setting, where the growth-fitting requirement is dismissed by resorting to unfitted or immersed boundary methods [24]. In this case, $\mathcal{T}_h$ is a triangulation of an artificial domain $\Omega_{\mathrm{art}}$, such that it includes the final physical domain, i.e. $\Omega_{\mathrm{f}} \subset \Omega_{\mathrm{art}}$, but it is also characterized by a simple geometry, easy to mesh with Cartesian grids.

Consider now partitions of $\mathcal{T}_h$ of the form $\{\mathcal{T}_{h,j}, \mathcal{T}_h \setminus \mathcal{T}_{h,j}\}$, $j = 0, \ldots, N_t$. In this classification, the cells in $\mathcal{T}_{h,j}$ are referred to as the *active* cells $T_{\mathrm{ac}}$, while the ones in $\mathcal{T}_h \setminus \mathcal{T}_{h,j}$ as the *inactive* cells $T_{\mathrm{in}}$. The key point of the EBM is to assign degrees of freedom only in active cells, that is, the computational domain, at any $j = 0, \ldots, N_t$, is defined by $\mathcal{T}_{h,j} = \{T_{\mathrm{ac}}\}$. In this way, inactive cells do not play any role in the numerical approximation; they have no contribution to the global linear system, in contrast with the quiet-element method [136]. Besides, note the similarities of this approach to the one employed in unfitted FE methods [23, 24], distinguishing interior and cut (active) cells from exterior (inactive) cells.

This representation of a growing domain is completed with a procedure to update the computational mesh during a time increment. The most usual approach is to use a search algorithm to find the set of cells in $\mathcal{T}_{h,j+1} \setminus \mathcal{T}_{h,j}$, $j = 0, \ldots, N_t - 1$, referred to as the *activated* cells $T_{\mathrm{acd}}$. Then, $\mathcal{T}_{h,j+1} = \mathcal{T}_{h,j} \cup \{T_{\mathrm{acd}}\}$ defines the next computational mesh, as illustrated in Figure 5.5. This means that $\mathcal{T}_{h,j+1}$ receives the old DOF global identifiers (and FE function DOF values) from $\mathcal{T}_{h,j}$ and has new degrees of freedom assigned in the activated cells. The initial value of these new DOFs is set with a criterion that depends on the application problem, as seen in Section 5.4.

Therefore, in a parallel distributed-memory environment, there are two different partitions of $\mathcal{T}_{h,j}$ playing a role in the simulation: (1) into subdomains $\mathcal{T}^i_{h,j}$, $i = 1, \ldots, n^{\mathrm{sbd}}$ and (2) into active $T_{\mathrm{ac}}$ and inactive $T_{\mathrm{in}}$ cells. Assuming a one-to-one mapping among subdomains and CPU cores, in our approach of Chapter 2, each processor stores the geometrical information corresponding to the subdomain portion $\mathcal{T}^i_{h,j}$ of the global mesh and one layer of off-processor cells surrounding the local subdomain mesh, the so-called *ghost* cells. With regards to (2), each cell is associated a *status*, e.g. an integer value, that expresses whether it is an active or inactive cell. It follows that $\mathcal{T}^i_{h,j}$ can be composed of both *active* and *inactive* cells.

As explained, the update of (2) follows from finding the subset $T_{\mathrm{acd}}$. *Local* activated cells, i.e. $\mathcal{T}^i_{h,j} \cap \{T_{\mathrm{acd}}\}$, are found with the search algorithm. Afterwards, a nearest-neighbour communication is carried out to update the status at the ghost cells. The second step is necessary to know whether a face sitting on the interprocessor contour is in the interior or at the boundary of the domain $\mathcal{T}_{h,j+1}$.

Bringing now briefly AMR into the discussion, some considerations with regards to the EBM must be taken into account when applying mesh transformations. First, when refining a cell, its status is inherited by the child cells. A cell can only be coarsened, when all siblings have the same status, otherwise the computational domain is perturbed. Finally, after a partition/redistribution of cells across processors, the status of the redistributed cells must be migrated with interprocessor communication. Adding this to the update of the status at ghost cells, the EBM in a parallel AMR framework only demands two extra interprocessor data transfers with respect to a standard FE simulation pipeline.

### 5.3.2 Parallel search algorithm.

The update of the computational mesh consists in finding the cells in the mesh that are inside the *known* growing region of the current time increment. In this sense, the problem is a standard and well known collision detection that can be tackled with any of the many existing algorithms. Our goal is to derive from them a strategy that is both computationally inexpensive and highly-parallelizable for octree-based meshes.

The approach adopted in this work is founded on the hyperplane separation theorem (HST) [32]. It states that given $A$ and $B$ two disjoint, nonempty and *convex* subsets of $\mathbb{R}^n$, there exists a nonzero vector $\boldsymbol{v}$ and a real number $c$, such that $\langle \boldsymbol{x}, \boldsymbol{v} \rangle \geq c$ and $\langle \boldsymbol{y}, \boldsymbol{v} \rangle \leq c$, for any $\boldsymbol{x} \in A$, $\boldsymbol{y} \in B$. In other words, the hyperplane $\langle \cdot, \boldsymbol{v} \rangle = c$, with $\boldsymbol{v}$ its normal vector, separates $A$ and $B$. A corollary of this theorem is that, if $A$ and $B$ are convex *polyhedra*, possible separating planes are either parallel to a face of one of the polyhedra or contain an edge from each of the polyhedra.

In our context, assuming the FE mesh is formed by rectangular hexahedra and the search volume is a cuboid, as in Section 5.4, the purpose is to test the intersection between two cuboids (any mesh cell vs search volume). In this case, the HST narrows down the number of potential separating planes to fifteen [73, 84]: three for the independent faces of the cell, three for the independent faces of the search cuboid and nine generated by all possible independent pairs formed by an edge from the cell and an edge from the search cuboid. It follows that two cuboids intersect, if and only if, none of the fifteen possible separating planes exists.

A separating plane can be tested by comparing the projections of the cuboids onto a line perpendicular to the plane, referred to as the separating axis (Figure 5.2). If the intervals of the projections do not intersect, then the cuboids do not intersect themselves. In [73], it is shown how this test amounts to compare two real quantities that depend on the dimensions and unit directions of the cell, the dimensions and unit directions of the search cuboid and the vector joining the centroids of the two cuboids. For each separating axis, the nonintersection test in terms of these quantities is given in [73, Table 1] or [85, Section 4.6].

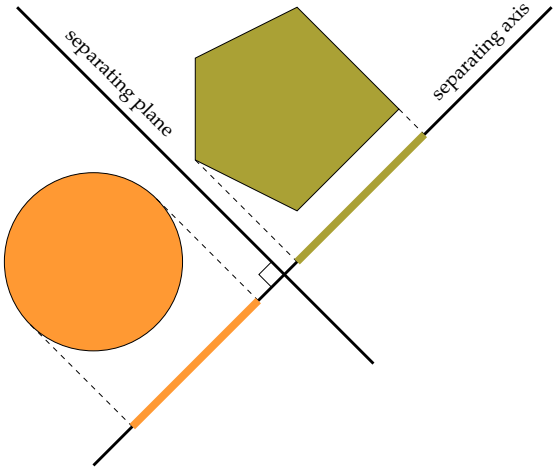
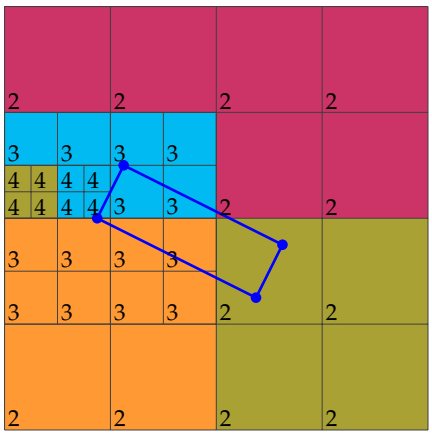



Figure 5.2: Illustration of the HST. A separating plane can be tested by examining whether the projections of the two convex bodies onto a line perpendicular to the plane intersect or not.

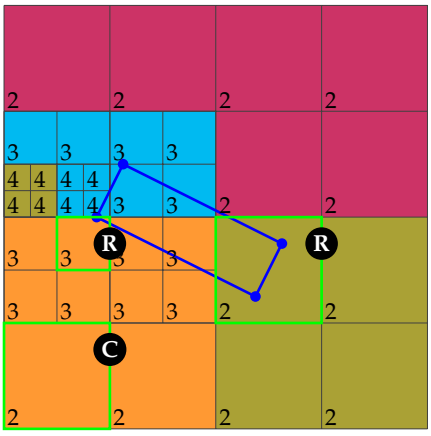
At this point, it remains to see how this test can be exploited for the parallel search algorithm with adaptive octree-based meshes. Dropping the assumption made in Section 5.3.1 of sequential mesh inclusion, i.e. $\mathcal{T}_{h,j} \neq \mathcal{T}_{h,j+1}, \forall j = 0, \ldots, N_t$, now $\mathcal{T}_{h,j}$ is transformed into $\mathcal{T}_{h,j+1}$, by applying several refinement (coarsening) operations to the octants intersecting (nonintersecting) the search cuboid of the $j \rightarrow j+1$ time increment. The transformation finishes when all octants intersecting the search cuboid have a given maximum level of refinement. In fact, these octants form precisely the subset $T_{\text{acd}}$. The number of transformations required is problem-dependent, but it is upper-bounded by the difference between the user-prescribed maximum and minimum levels of refinement.

Therefore, the mesh transformation from $\mathcal{T}_{h,j}$ into $\mathcal{T}_{h,j+1}$ is carried out in a finite number of refinement/coarsening steps, each one determined by a cell-wise search. Specifically, the criterion to decide whether an octant is refined or coarsened is to perform the nonintersection test against the search cuboid. If it passes (fails), the octant is coarsened (refined). An example of this procedure is shown in Figure 5.3. As observed, if all processors know the dimensions of the search cuboid, the algorithm is embarrassingly parallel, in particular, it does not require interprocessor communication. By construction of `p4est` a subdomain can only prescribe refinement/coarsening operations to its own local cells. It follows that the nonintersection test on ghost cells is redundant; the status of ghost cells must be updated at the end of the mesh transformation with a nearest-neighbour communication.

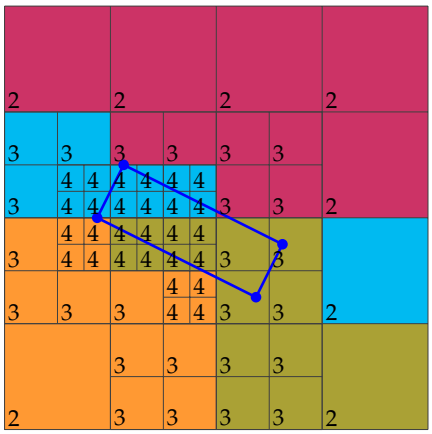
The search algorithm can be further accelerated by intersecting beforehand the search cuboid (or a bounding box of it) against the subdomain limits. In this case, the procedure can be skipped on those subdomains that do not intersect the cuboid. On the other hand, faster tests can be designed for uniformly refined meshes, such as checking whether the centroid of the cell is inside the search cuboid. Note that this test



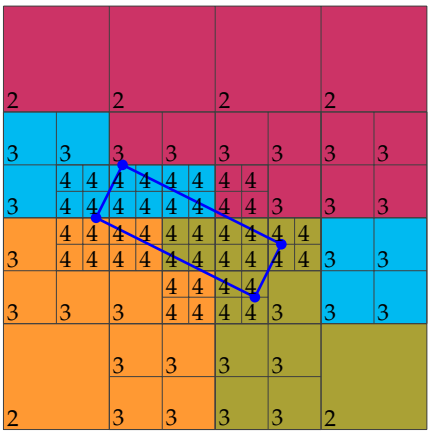
(a) Given $\mathcal{T}_{h,j}, j = 0, \ldots, N_t$, compute the search volume for the time increment $\Delta t^{j \to j+1}$.




(b) Loop over cells in $\mathcal{T}_{h,j}$. If nonintersection test fails (passes), then mark cell for refinement (coarsening). Some examples are highlighted (R = refine, C = coarsen).



(c) Refine, coarsen and redistribute cells.



(d) Repeat (b)-(c) until all cells with maximum level of refinement that intersect the cuboid have been found.

Figure 5.3: 2D example illustrating the iterative procedure to transform $\mathcal{T}_{h,j}$ into $\mathcal{T}_{h,j+1}$. The maximum and minimum levels of refinement are 4 and 2. The level of each cell is written at their lower left corner. Each colour represents a different subdomain. Note that, from one step to the next one, some cells that do not intersect the search volume have to be refined to keep the 2:1 balance.

is not equivalent to the method of separating axes. Besides, it is not suitable for octree meshes. For instance, it may not transform the mesh at all if the search cuboid sits on top of heavily coarsened cells.

Apart from that, the algorithm is limited to rectangular meshes and search volumes. In more general cases, e.g. high-order meshes, the hypothesis of convexity may not hold; i.e. the hyperplane separation theorem cannot be the starting point; a more general method must be adopted instead. However, as shown in the example of Section 5.6.3, a high order mesh can be configured such that the search cuboid always overlaps a region of the mesh, with enough resolution to assume its cells are quasi-rectangular.

### 5.3.3 Dynamic load balancing

When designing a scalable application, the partition into subdomains must be defined such that it evenly distributes among processors the total computational load. However, to this goal, the EBM adds two mutually excluding constraints; indeed, while the size of data structures and the complexity of procedures that manipulate the mesh grow with the total number of cells, those concerning the FE space, the FE system and the linear solver depend on the number of active cells (i.e. number of DOFs).

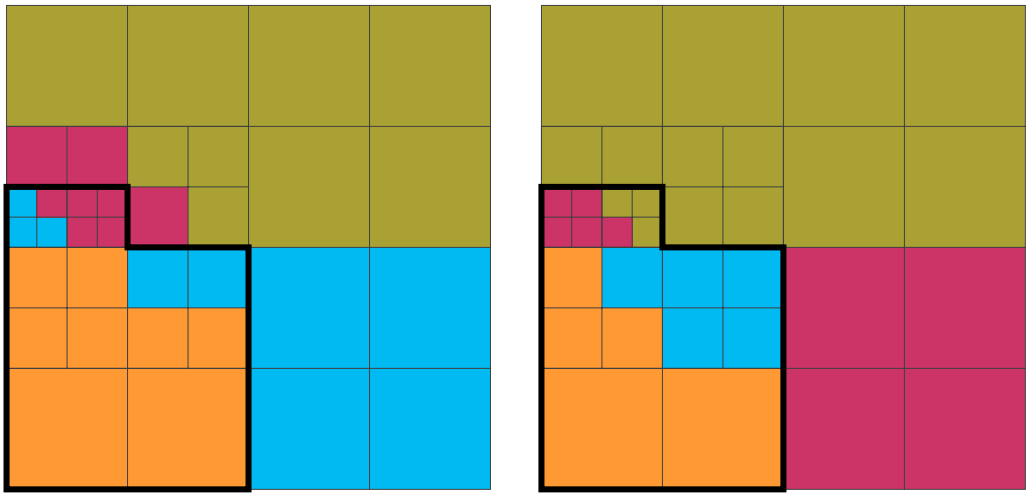
The distribution of computational work can be tuned by allowing for a user specified *weight function* $w$ that assigns a non-negative integer value to each octant. The partition can then be constructed by equally distributing the accumulated weights of the octants that each processor owns, instead of the number of owned octants per processor. As `p4est` provides such capability (see [43, Section 3.3]), the remaining question is to decide how to define $w$, taking into account the constraints above.

The answer depends on how the computational time is distributed among the different stages of the FE simulation pipeline. In the context of growing domains, FE analysis is a long transient and it may be often desirable to reuse the same mesh for several time steps, seeking to minimize the AMR events and, thus, reduce simulation times. In this scenario, the number of time steps (linear system solutions) is greater than the number of mesh transformations and $w$ should favour the balance of active cells. With this idea in mind, the weight function can be defined as

$$w_T = \begin{cases} w_a & \text{if } T \in \{T_{\text{ac}}\} \\ w_i & \text{if } T \in \{T_{\text{in}}\} \end{cases} \tag{5.1}$$

where $w_a \in \mathbb{N}$ and $w_i \in \mathbb{N}^0$.

The effect of this weight function is illustrated in Figure 5.4. A uniform distribution of the octree octants, $(w_a, w_i) = (1, 1)$, may lead to high load imbalance in the number of DOFs per subdomain. There can even be fully inactive parts, as shown in Figure 5.4(a), leaving the processors in charge of them mostly idling during local integration, assembly and solve phases. In contrast, $(w_a, w_i) = (1, 0)$ gives the most uniform distribution of $\{T_{\text{ac}}\}$, but it can also lead to extreme imbalance of $\{T_{\text{in}}\}$ and, thus, the whole set of triangulation cells. Alternatively, pairs $(w_a, w_i)$ satisfying $w_a \gg w_i$ offer good compromise partitions.



(a) A default partition, that is, $(w_a, w_i) = (1, 1)$, can result in a poor balancing of DOFs and even fully inactive parts (e.g. green subdomain).

(b) By setting partition weights to, e.g. $(w_a, w_i) = (10, 1)$, the active cells can be balanced, leading to a more equilibrated parallel distribution of the DOFs.

Figure 5.4: 2D example illustrating how partition weights can be used to balance dynamically the DOFs across the processors. Each colour represents a different subdomain. Active cells are enclosed by a thick contour polygon, representing the computational domain.

## 5.4 Application to metal AM

### 5.4.1 Heat transfer analysis

After introducing the ingredients of the parallel FE framework for growing geometries, the purpose now is to apply it to the thermal analysis of an additive manufacturing process by powder-bed fusion, such as Direct Metal Laser Sintering (DMLS). This manufacturing technology is illustrated in Figure 6.1. This will be the reference problem for the subsequent analysis with numerical experiments.

Let $\Omega(t)$ be a *growing* domain in $\mathbb{R}^3$ as in Section 5.3. Here, $\Omega(t)$ represents the component to be printed. The governing equation to find the temperature distribution $u$ in time is the balance of energy equation, expressed as

$$C(u)\partial_t u - \boldsymbol{\nabla} \cdot (k(u)\nabla u) = r, \quad \text{in } \Omega(t), \quad t \in [t_\text{i}, t_\text{f}], \tag{5.2}$$

where $C(u)$ is the heat capacity coefficient, $k(u) \geq 0$ is the thermal conductivity and $r$ is the rate of energy supplied to the system per unit volume by the moving laser. $C(u)$ is given by the product of the density of the material $\rho(u)$ and the specific heat $c(u)$, but one may consider a modified heat capacity coefficient to also account for phase change effects [54] or compute $C(u)$ with the CALPHAD approach [103, 104, 172].

Equation (5.2) is subject to the initial condition

$$u(\boldsymbol{x}, t_\text{i}) = u^0(\boldsymbol{x})$$

and the boundary conditions in Figure 6.2 are (1) heat conduction through the building platform, (2) heat conduction through the powder-bed and (3) heat convection and radiation through the free surface. After linearising the Stefan-Boltzmann's law for heat radiation [54], all heat loss boundary conditions admit a unified expression in terms of Newton's law of cooling:

$$q_{\text{loss}}(u,t) = h_{\text{loss}}(u)(u - u_{\text{loss}}(t)), \text{ in } \partial\Omega^{\text{loss}}(t), \ t \in [t_{\text{i}}, t_{\text{f}}], \tag{5.3}$$

where *loss* refers to the kind of heat loss mechanism (conduction through solid, conduction through powder or convection and radiation) and the boundary region where it applies (contact with building platform, interface solid-powder or free surface).

The weak form of the problem defined by Equations (5.2)-(5.3) can be stated as: *Find* $u(t) \in \mathcal{V}_t = \mathrm{H}^1(\Omega(t))$, *almost everywhere in* $(t_{\text{i}}, t_{\text{f}}]$, *such that*

$$(C(u)\partial_t u, v) - (k(u)\nabla u, \nabla v) + \langle h_{\text{loss}}(u)u, v\rangle_{\partial\Omega^{\text{loss}}} = \langle f, v\rangle + \langle h_{\text{loss}}(u)u_{\text{loss}}, v\rangle_{\partial\Omega^{\text{loss}}}, \ \forall v \in \mathcal{V}_t. \tag{5.4}$$

Considering now $\mathcal{T}_h$ the triangulation of $\Omega_{\text{f}}$, $\mathcal{V}_h \subset \mathcal{V}_{t_{\text{f}}}$ a conforming FE space for the temperature field and $\{\varphi_j(\boldsymbol{x})\}_{j=1}^{N_h}$ a FE basis of the space $\mathcal{V}_h$, the semi-discrete form of Equation (5.4), after discretisation in space with the Galerkin method and integration in time, e.g. with the semi-implicit backward Euler method, reads

$$\begin{aligned} \left[\frac{\mathbf{M}_{\text{C}}^n}{\Delta t^{n+1}} + \mathbf{A}^n + \mathbf{M}_{\text{loss}}^n\right]\mathbf{U}^{n+1} &= \mathbf{b}_f^{n+1} + \frac{\mathbf{M}_{\text{C}}^n}{\Delta t^{n+1}}\mathbf{U}^n + \mathbf{b}_{\text{loss}}^n, \\ \mathbf{U}(0) &= \mathbf{U}^0, \end{aligned} \tag{5.5}$$

where the time interval of interest $[t_{\text{i}}, t_{\text{f}}]$ has been divided in subintervals $t_{\text{i}} = t_0 < t_1 < \ldots < t_{N_t} = t_{\text{f}}$ with $\Delta t^{n+1} = t_{n+1} - t_n$ variable for $n = 0, \ldots, N_t - 1$. As a result of using the EBM (Section 5.3), Equation (5.5) is only constructed in $\Omega(t_n)$, $n = 0, \ldots, N_t - 1$. In other words, local integration and assembly of Equation (5.4) is only carried out at the subset of active elements $\{T_{\text{ac}}\} = \mathcal{T}_{h,n}$.

Moreover, $\mathbf{U}(t) = (\mathrm{U}_j(t))_{j=1}^{N_h}$ and $\mathbf{U}^0 = (\mathrm{U}_j^0)_{j=1}^{N_h}$ are the components of $\mathrm{U}_h(t)$ and $\mathrm{U}_h^i$ with respect to the basis $\{\varphi_j(\boldsymbol{x})\}_{j=1}^{N_h}$, and the coefficients of $\mathbf{M}_\square$, $\mathbf{A}$ and $\mathbf{b}_\square$ are given by:

$$\begin{aligned} &\mathrm{M}_{\text{C}}^{n,ij} = (C(\mathbf{U}^n)\varphi_i, \varphi_j)_{\Omega(t_n)}, \ \mathrm{M}_{\text{loss}}^{n,ij} = \langle h_{\text{loss}}(\mathbf{U}^n)\varphi_i, \varphi_j\rangle_{\partial\Omega^{\text{loss}}\cap\partial\Omega(t_n)}, \\ &\mathrm{A}^{n,ij} = (k(\mathbf{U}^n)\nabla\varphi_i, \nabla\varphi_j)_{\Omega(t_n)}, \\ &\mathbf{b}_f^{n+1,i} = \langle\varphi_i, f^{n+1}\rangle_{\Omega(t_n)}, \ \mathbf{b}_{\text{loss}}^{n,i} = \langle h_{\text{loss}}(\mathbf{U}^n)\varphi_i, \mathrm{U}_{\text{loss}}(t^{n+1})\rangle_{\partial\Omega^{\text{loss}}\cap\partial\Omega(t_n)}. \end{aligned}$$

An important characteristic of the physical problem is that, due to high heat capacity of metals and small time steps necessary to meet accuracy requirements, Equation (5.5) is often a mass-dominated linear system. As a result, Jacobi (or diagonal) preconditioning is adopted in the numerical examples of Section 5.6. Although this preconditioner does not alter the asymptotic behaviour of the condition number of the

conductivity matrix ($\mathcal{O}(h^{-2})$), it corrects relative scales of Equation (5.5) arising from the fact that meshes resulting from (1) have cells at very different refinement levels, i.e. highly varying sizes.

### 5.4.2 FE modelling of the moving thermal load

As pointed out in Equation (5.2), $r$ is a moving thermal load that models the action of the laser in the system. But the moving heat source also drives the growth of the geometry in time, as the sintering process triggered by the laser transforms the metal powder into new solid material.

Therefore, the FE modelling of the printing process requires a method to apply the volumetric heat source $r$ in space and time and track the growing $\Omega(t)$. The EBM presented in Section 5.3.1 can serve both purposes, as seen in Figure 5.5. In this case, the set of activated cells $T_{\text{acd}}$, representing the incremental growth region, is also affected by the laser during the time increment, i.e. the heat source term is also integrated in these cells.

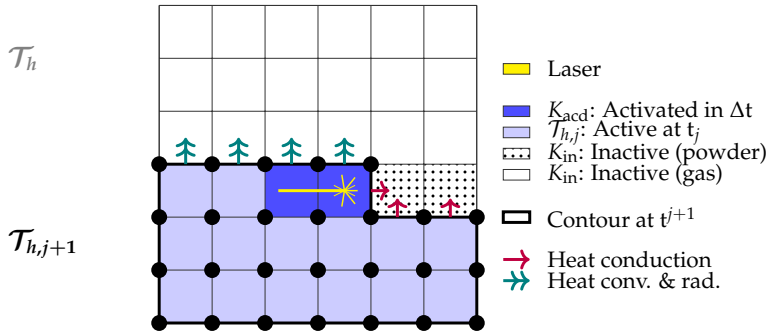



Figure 5.5: Illustration of the *element-birth* method applied to the thermal simulation of an AM process. As shown with this 2D FE cartesian grid, for any $j = 0, \ldots, N_t$, DOFs are only assigned to the set of active cells $\{T_{\text{ac}}\} = \mathcal{T}_{h,j+1}$. A search algorithm is employed to identify the set of activated cells $\{T_{\text{acd}}\} = \mathcal{T}_{h,j+1} \setminus \mathcal{T}_{h,j}$, where the laser is focused during $\Delta t$. The computational mesh is then updated, by assigning new DOFs in $T_{\text{acd}}$. Afterwards, the energy input, as stated by Equation (5.6), is uniformly distributed over $T_{\text{acd}}$.

Another comment arises on the update of the computational mesh. In general, one aims to follow the actual path of the laser in the machine, as faithfully as possible. To this end, the search algorithm of Section 5.3.2 comes into play. The information of the laser path, together with other process parameters, such as the laser spot width, defines the search volume containing the cells affected by the energy input during the time step [54], subsequently referred to as the heat affected volume (HAV).

If $r$ is taken as a uniform heat source, the average density distribution is computed as

$$r = \frac{\eta \mathrm{W}}{\mathrm{V}_{\text{acd}}}, \tag{5.6}$$

where W is the laser power [W] (watt), $\eta$ is the heat absorption coefficient, a measure of the laser efficiency and $\mathrm{V}_{\text{acd}}$ is the volume of activated cells, i.e. intersecting the HAV.

Goldak-based or surface Gaussian distributions may also be considered. In those cases, the heat source is evaluated with their corresponding analytical expressions in $T_{acd}$. On the other hand, the initial temperature of the new DOFs is set to the same value as the initial value at the building platform. More accurate alternatives are analysed in the literature [136], but this aspect of the model is not relevant to the overall performance of the framework.

## 5.5 Computer implementation.

This section describes the software design of a parallel FE framework for growing domains, the so-called `FEMPAR-AM` module, atop the services provided by `FEMPAR` [19]. `FEMPAR` is an open source, general-purpose, object-oriented scientific software framework for the massively parallel FE simulation of problems governed by PDEs. `FEMPAR` software architecture is composed by a set of mathematically-supported abstractions that let its users break FE-based simulation pipelines into smaller blocks that can be used and/or extended to fit users' application requirements. Each abstraction takes charge of a different step in a typical FE simulation, including, among others, mesh generation, adaptation, and partitioning, construction of a global FE space and DOF numbering, numerical evaluation of cell and facet integrals and linear system assembly, linear system solution or generation of simulation output data for later visualization. The reader is referred to [16, 19] for a detailed exposition of the main software abstractions in `FEMPAR`. Although `FEMPAR-AM` exploits most of the software abstractions provided by `FEMPAR`, the discussion in this section mainly focuses on those which had to be particularly set up and/or customized to support growing domains. Apart from this, the section also presents newly introduced software abstractions which are particular to `FEMPAR-AM`. The exposition is intended to help the reader grasp how any general-purpose FE framework can be customized in order to deal with growing domains.

As `FEMPAR-AM` relies on the EBM (Section 5.3.1), the first software requirement that has to be fulfilled by the underlying FE framework is the ability to build global FE spaces that only carry out DOFs for cells which are active at the current simulation step. In `FEMPAR`, a global FE space is built from two main ingredients: (1) `triangulation_t` [19, Section 7], which represents the geometrical discretisation of the computational domain into cells, and (2) `reference_fe_t` [19, Section 6], which represents an *abstract* local FE on top of each of the triangulation cells. A type extension of `reference_fe_t`, referred to as `void_reference_fe_t` [19, Section 6.5], implements a local FE with no degrees of freedom. The data structure in charge of building the global FE space, i.e. `fe_space_t` [19, Section 10], is general enough such that one may use a different local FE on different regions of the computational domain. By mapping *active* cells to standard (e.g. Lagrangian) FEs and *inactive* cells to *void* FEs, `fe_space_t` is constructed such that it assigns global DOF identifiers only to the nodes of *active* cells;

while nodes surrounded by *inactive* cells do not receive a DOF identifier and, as a result, they neither assemble any contributions from local integration nor form part of the global linear system. On the other hand, the `triangulation_t` software abstraction lets its users exploit the so-called `set_id` [19, Section 7.1], a cell-based integer attribute. Each cell of the triangulation can be assigned a `set_id` number to group the cells into different subsets. In our case, this variable is used to store the current status of the cell in the mesh and, by `fe_space_t`, to determine which `reference_fe_t` (i.e. local FE space) to put atop each cell (see discussion above). The update of the `set_id` in the local cells is carried out within the search algorithm (Section 5.3.2), whereas the update in the off-processor ghost cells reuses an existing procedure in `triangulation_t` that invokes a nearest-neighbour communication. Another readily available method of `triangulation_t` takes charge of migrating the `set_id` values when the mesh is redistributed; see Section 5.3.1.

Apart from the special set up of `fe_space_t` described in the previous paragraph, `FEMPAR-AM` also requires to customize `fe_space_t` (as-is in `FEMPAR`) to support growing domains. In particular, one needs to inject DOF values of any field (e.g. temperature in thermal AM) from $\mathcal{T}_{h,j-1}$ into $\mathcal{T}_{h,j}$ and assign initial DOF values to *activated* cells. For this purpose, `FEMPAR-AM` implements `growing_fe_space_t`, a data type extending the standard `fe_space_t`, that provides an special method, referred to as `increment()`, performing this operation. The implementation of this procedure depends on how data structures in charge of handling DOFs and DOF values are laid out and, more importantly, on whether each processor stores DOF values only in its local subdomain portion or also at the ghost cells layer. In the first case, an extra nearest-neighbour communication is needed to update DOF values at nodes sitting on a subdomain interprocessor interface.

The following paragraphs introduce the main software abstractions exclusive to `FEMPAR-AM`. They are in charge of supporting the update of the computational mesh, tracking the growth of the domain, and driving the main top-level AM process simulation loop. The software subsystem is formed by (1) `activation`, a customizable object enclosing data structures and methods necessary to find, at each time step, the subset of *activated* cells $T_{\text{acd}}$, (2) `cli_laser_path`, an AM-specific object to handle the geometrical information of the laser path, and (3) `discrete_path`, also AM-specific, to generate the space-time discretisation of the laser path. To clarify their structure, contents and relationships, an UML class diagram is constructed in Figure 5.6.

`activation` contains the `search_volume` abstract object, a placeholder for the geometrical description of the region to be activated during the time increment. Being a placeholder means that the instance is designed to allocate the minimum required memory space to hold a *single* search volume at a time. As the definition of the activated region depends on the application at hand, the client must specialize the behaviour of the object to its needs. In particular, extensions of this data type must have all the

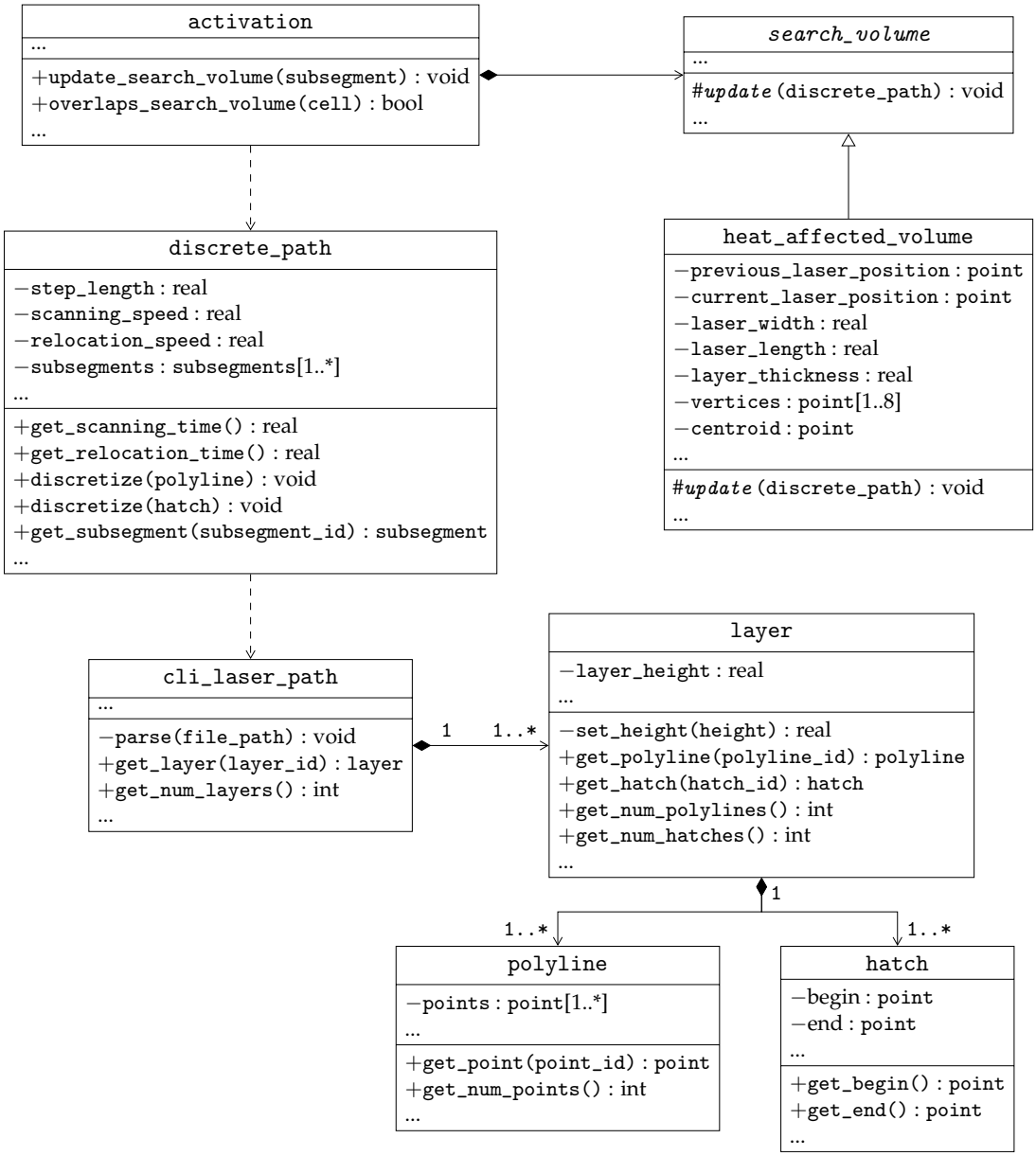



Figure 5.6: UML class diagram of the software subsystem that FEMPAR-AM uses to support the update of the computational mesh in Algorithm 4 and to drive the AM process simulation, tracking the laser path, in Algorithm 5. point is a simple class encapsulating three real-valued coordinates, whereas subsegment encapsulates two point instances.

member variables and implement the methods needed to compute and update the dimensions and vertex coordinates defining the search cuboid at each time increment. In our context, `heat_affected_volume` is the AM-tailored search volume extended object. `activation` implements two public methods to satisfy the user requirements in the update of the computational mesh: (1) `update_search_volume(subsegment)` and (2) `overlaps_search_volume(cell)`. The former fills the search volume coordinates for a given time increment and the latter is used to test for collision between the search cuboid and any cell of the triangulation, using the method described in Section 5.3.2. Algorithm 4 implements the update and increment of the computational domain, i.e. the $\mathcal{T}_{h,j-1}$ into $\mathcal{T}_{h,j}$ transformation, described in Section 5.3.2 and Figure 5.3. For simplicity all steps regarding the treatment of the FE space and global FE functions are omitted, but they must be projected and redistributed. As observed, methods (1) and (2), invoked at Lines 1 and 7, fulfill important steps of the procedure.

**Algorithm 4:** `update_and_increment_computational_domain(subsegment)` (see also Figure 5.3)

```
1  activation.update_search_volume(subsegment) /* fill search volume coordinates for current
     subsegment */
2  found_cell_for_refinement ← true
3  while found_cell_for_refinement do    /* repeat until all max level cells intersecting the
     search volume found */
4  |  found_cell_for_refinement ← false
5  |  for cell ∈ triangulation do
6  |  |  if cell.is_local() then                    /* local = owned by the current MPI task */
7  |  |  |  if activation.overlaps_search_volume(cell) then /* cell intersects the search volume
   |  |  |    (see Section 5.3.2) */
8  |  |  |  |  if cell.get_level() < max_level_of_refinement then
9  |  |  |  |  |  cell.set_for_refinement()
10 |  |  |  |  |  found_cell_for_refinement ← true
11 |  |  |  |  else
12 |  |  |  |  |  cell.set_for_do_nothing()
13 |  |  |  |  end
14 |  |  |  else                                  /* cell does not intersect the search volume */
15 |  |  |  |  if cell.get_level() > min_level_of_refinement then
16 |  |  |  |  |  cell.set_for_coarsening()
17 |  |  |  |  else
18 |  |  |  |  |  cell.set_for_do_nothing()
19 |  |  |  |  end
20 |  |  |  end
21 |  |  end
22 |  end
23 |  triangulation.refine_and_coarsen()
24 |  ...            /* FE space and global FE functions projection/interpolation apply here */
25 |  set_partition_weights()          /* construct weight function as in Equation (5.1) */
26 |  triangulation.redistribute()
27 |  ...                      /* FE space and global FE functions redistribution apply here */
28 end
29 mark_activated_cells()
30 growing_fe_space.increment()    /* inject DOF values of T_{h,j-1} into T_{h,j} and initialize DOF
     values in T_acd */
```

Among the application-specific objects, `cli_laser_path` takes charge of managing

the geometrical information of the laser path. The data comes in the same Common Layer Interface (CLI) ASCII file sent to the numerical control of the machine. CLI [182] is a common universal format for the input of geometry data to model fabrication systems based on layer manufacturing technologies. In the context of AM, a CLI file describes the movement of the laser in the plane of each layer with a complex sequence of *polylines*, to define the (smooth) boundary of the component, and *hatch* rectiliniar patterns for the inner structures (see [182] for several examples). `cli_laser_path` holds a parser for this file format (`parse(file_path)`) and accommodates its hierarchical structure in the following way: First, it aggregates an array of `layer` entities. For each *layer* in the CLI file, a `layer` instance is created and the layer height is stored. Likewise, for each *polyline* or *hatch* associated to the layer, an instance of `polyline` or `hatch` is created and filled with their plane coordinates. The class is supplemented with several query methods (e.g. `get_layer(layer_id)`, `get_hatch(hatch_id)`, `get_point(point_id)`), such that the user can navigate through the laser path subentities and extract their point coordinates. Although not implemented, a polymorphic superclass `laser_path` of `cli_laser_path` may be introduced to consider other file formats. In this way, new file formats can be accommodated with new child extensions of `laser_path`; at the moment, `FEMPAR-AM` can only support CLI files.

Closely associated to `cli_laser_path` is `discrete_path`, an entity that generates the space-time discretisation of the laser path. Given that the printing process is tightly related to the movement of the laser, it is more natural to discretise the laser path with a *step length* $\Delta x$, instead of a time step. `discrete_path` takes the user-prescribed step length and the current *polyline* or *hatch* segment and divides it into subsegments of $\Delta x$ size with the method `discretise(polyline/hatch)`. To support the time integration, `discrete_path` also computes the time increment associated to each subsegment as

$$\Delta t = \Delta x / \mathrm{v}_{\mathrm{scan}},$$

where $\mathrm{v}_{\mathrm{scan}}$ is the scanning speed. At each time step, `discrete_path` feeds `activation` with the current subsegment via `update_search_volume(subsegment)`. After `activation` generates the current search volume and drives the mesh transformation, see Algorithm 4, standard FE simulation steps follow until solving the linear system modelling the printing during the current time increment. The sequence is repeated until all subsegments have been simulated. Following this, the system is allowed to cool down, while the laser moves to the begin point of the next entity in the laser path or a new layer is spread. For the cooling step, `discrete_path` also computes recoat and relocation time as the quotient of the distance between the end point of the current *polyline* or *hatch* and the begin point of the next one divided by the relocation speed $\mathrm{v}_{\mathrm{reloc}}$.

A relevant feature of the design is that `discrete_path` is another placeholder container object, i.e. it only stores the discretisation for the current *polyline* or *hatch* being printed and it is updated while looping over the laser path entities. Using the

`cli_laser_path` recursive construction and the design of `discrete_path`, this loop, which is the one at the top level of `FEMPAR-AM` simulations, can be written in a compact form that reflects the actual printing process, as shown in Algorithm 5 and Algorithm 6. Note that the cost of the operations involved in filling and discretising the laser path is negligible w.r.t. other stages of the simulation that concentrate the bulk of the computational cost (e.g. linear solver). Given this, and the fact that each processor must know the global search cuboid coordinates (see Section 5.3.2), data generated by `cli_laser_path` and `discrete_path` is not distributed, it is replicated in each processor to avoid extra communications.

**Algorithm 5:** Top-level simulation loop of `FEMPAR-AM`

```
for layer ∈ laser_path do
    for polyline ∈ layer do
        discrete_path.discretise(polyline)    /* divide polyline into user-prescribed Δx-sized
          subsegments */
        simulate_printing_process(discrete_path)                    /* see Algorithm 6 */
    end
    for hatch ∈ layer do
        discrete_path.discretise(hatch)       /* divide hatch into user-prescribed Δx-sized
          subsegments */
        simulate_printing_process(discrete_path)                    /* see Algorithm 6 */
    end
end
```

**Algorithm 6:** `simulate_printing_process(discrete_path)`

```
for subsegment ∈ discrete_path do
    update_and_increment_computational_domain(subsegment)        /* see Algorithm 4 */
    ...        /* for simplicity, remaining simulation steps until linear system solution are
      omitted, but follow here */
end
relocation_time ← discrete_path.get_relocation_time()
simulate_cooling(relocation_time)        /* solve Equation (5.5) without heat input (laser
  relocation or layer recoat) */
```

Other minor thermal AM customizations that complete the design are `heat_input`, a polymorphic and extensible entity with a suite of heat source term descriptions (e.g. uniform, surface Gauss or Goldak double ellipsoidal); `property_table`, an auxiliary object to allow the client to linearly interpolate properties that are known in tabular format, in order to evaluate temperature-dependent properties; and `heat_transfer_discrete_integration_t`, a subclass of `discrete_integration_t` [19, Section 11.2], which evaluates the entries of the matrices and vectors defining the linear system of Equation (5.5).

## 5.6 Numerical experiments and discussion

### 5.6.1 Verification of the thermal FE model

First, the thermal FE model presented in Section 5.4 is verified against a 3D benchmark present in the literature [78, 108] that considers a moving single-ellipsoidal heat source on a semi-infinite solid with null fluxes at the free surface.

Assuming a Eulerian frame of reference $(x, y, z)$, consider a heat source located initially at $z = 0$ that travels at constant velocity $v$ along the $x$-axis on top of the semi-infinite solid defined by $z \leq 0$. The heat source distribution, derived from Goldak's double-ellipsoidal model [82], is defined by

$$q(x,y,z,t) = \frac{6\sqrt{3}}{\pi\sqrt{\pi}}\frac{Q}{abc}\exp\left[-3\left(\frac{(x-vt)^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\right)\right],$$

where $Q$ is the (effective) rate of energy supplied to the system, $v$ is the velocity of the laser and $a, b, c$ are the main dimensions of the ellipsoid, as shown in Figure 5.7(a).

The problem at hand is linear and admits a semi-analytical solution using Green's functions [46, 58] given by

$$u(x,y,z,t) = u_0 + \frac{6\sqrt{3}}{\pi\sqrt{\pi}}\frac{\alpha Q}{k}\int_0^t \frac{\exp\left[-3\left(\frac{(x-vt)^2}{a^2+12\alpha(t-\tau)}+\frac{y^2}{b^2+12\alpha(t-\tau)}+\frac{z^2}{c^2+12\alpha(t-\tau)}\right)\right]}{\sqrt{(a^2+12\alpha(t-\tau))(b^2+12\alpha(t-\tau))(c^2+12\alpha(t-\tau))}}\mathrm{d}\tau,$$

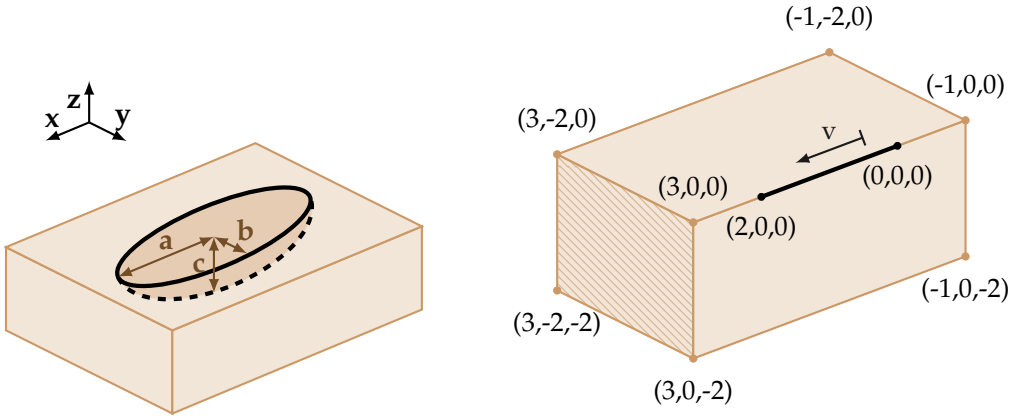
with $u_0$ the initial temperature and $\alpha = k/\rho c$ the thermal diffusivity.

Using the symmetry of the problem, the numerical simulation considers a cuboid with coordinates given in Figure 5.7(b). The path of the laser follows a segment centred along the edge of the cuboid that sits on the $x$-axis. Null fluxes apply at the top surface and the lateral face of symmetry, whereas Dirichlet boundary conditions apply at the remaining contour surfaces to account for the semi-infinite solid.

An $h$-adaptive linear FE mesh is employed, where the smallest size is prescribed around the welding path. Starting with the initial mesh shown in Figure 5.8(a) and assigning $u_0 = 20$, $Q = 50$, $v = 1$, $\alpha = 0.1$, $k = 1$ and $(a, b, c) = (0.3, 0.15, 0.25)$, a convergence test is carried out.

For a parabolic heat equation with sufficiently smooth solution, the error norm in $L^2([0,T]; H_0^1(\Omega))$ with a Backward Euler time integration scheme is proportional to $(h^p + \Delta t)$ (see Theorem 6.29 in [77]), where $p$ is the order of the FE. Since $p = 1$ in this experiment, the time discretisation should be refined at the same rate of the space discretisation.

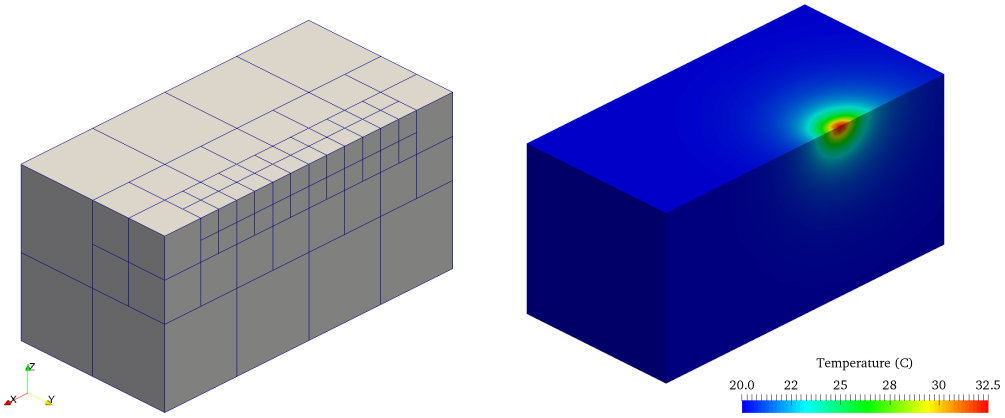
Taking this into consideration with an initial time step of $\Delta t = 0.008$, Figure 5.9 shows that the numerical error in $L^2([0,T]; H_0^1(\Omega))$ of the 3D semi-analytical benchmark decreases at the same rate as the theoretical one. This indicates a correct implementation of the thermal FE model.




(a) Single-ellipsoidal heat source model.

(b) Domain of analysis, boundary conditions and welding path. Homogeneous Neumann boundary conditions apply at the top surface and the lateral face of symmetry. Dirichlet boundary conditions apply elsewhere.

Figure 5.7: 3D semi-analytical benchmark problem. Heat source distribution, geometry and boundary conditions.




(a) Initial adapted octree mesh.

(b) Temperature contour plot at $t = 0.5$ ($3^{\text{rd}}$ refinement step).

Figure 5.8: 3D semi-analytical benchmark problem. Initial mesh and contour plot.

### 5.6.2 Strong-scaling analysis

Next, the focus is turned to analysing the performance of `FEMPAR-AM` with a strong-scaling analysis[1]. The model problem for the subsequent experiments is designed to be geometrically simple, but with a computational load comparable to a real scenario of an industrial application.

According to this, the object of simulation is now the printing of 48 layers of 31.25 [µm] on top of a $32 \times 32 \times 16$ [mm] prism. After printing the 48 layers, the prism has

[1]Strong scalability is the ability of a parallel system (i.e. algorithm, software and parallel computer) to efficiently exploit increasing computational resources (CPU cores, memory, etc.) in the solution of a fixed-size problem. An *ideally* strongly scalable code decreases CPU time exactly as $1/P$, where $P$ is the number of processors being used. In other words, if the system solves a size $N$ problem in time $t$ with a single processor, then it is able to solve the same problem in time $t/P$ with $P$ processors.

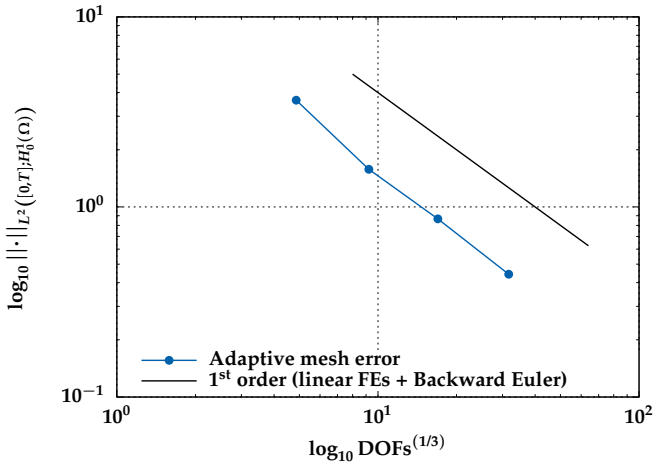



Figure 5.9: 3D semi-analytical benchmark problem. Convergence test.

dimensions of $32 \times 32 \times 17.5$ [mm], as shown in Figure 5.10(a).

Concerning the process parameters, the power of the laser is set to 400 [W], the volumetric deposition rate during scanning is $d_{\mathrm{p}} = 10.0$ [mm$^3$/s] and the time allowed for lowering the platform, recoating and layer relocation between layers is $t_{\mathrm{r}} = 10.0$ [s].

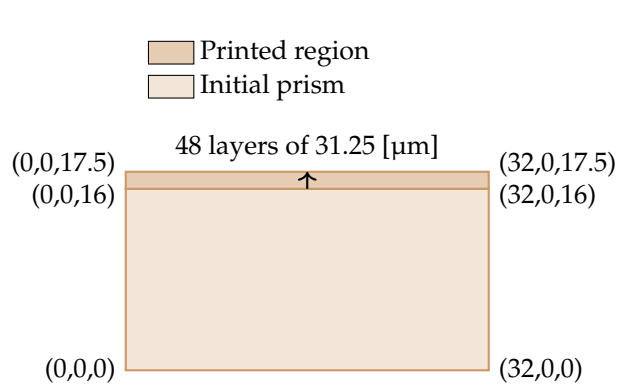
Apart from that, the material chosen is the Ti6Al4V Titanium alloy. The temperature dependent density, specific heat and thermal conductivity are obtained from a handbook and plotted in [54].

A constant heat convection boundary condition applies on the boundary of the cube with $h_{\mathrm{out}} = 50$ [W/m$^2$T] and $u_{\mathrm{out}} = 35$ [°C]. The initial temperature of both the prism and each new layer is $u_0 = 90$ [°C].

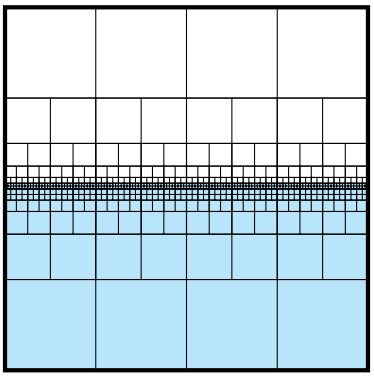
The root octant of the octree mesh is defined to cover a $32 \times 32 \times 32$ [mm] cube region. The octree is transformed during the simulation to model the layer-by-layer deposition process, by prescribing a maximum refinement level of 11, i.e. assigning a mesh size of $h = 32,000/2^{11} = 15.625$ [µm], to the elements inside the layer that is currently being printed. A mesh gradation is then established according to the distribution of thermal gradients (highest at the printing region, lowest at the bottom of the cuboid). The mesh size in the $(x, y)$ plane is fixed, whereas it decreases in both $z$-directions, until reaching a minimum level of refinement of 4 at the top and bottom of the prism. The computational domain is then defined by the initial prism and the layers that have been printed up to the current time. As seen in Figure 5.10(b), most elements end up concentrating around the current layer, due to the coarsening induced by the 2:1 balance, but this is also the region with the highest temperature variations.

This refinement strategy leads to a simulation workflow that, for each layer, comprises the following steps:

1. **Remeshing:** The previous mesh is refined and coarsened to accommodate the current layer.




(a) Setting of the example. The initial prism and the printed region are plotted in light and dark brown.



(b) Illustration of the refinement strategy for maximum and minimum refinement levels of 8 and 3, resp. Computational domain in light cyan.

Figure 5.10: Strong-scaling problem set up. Plane XZ view of the setting and the mesh.

2. **Redistribution:** The new mesh is partitioned and redistributed among all processors to maintain load balance.

3. **Activation:** New DOFs are distributed and initialized over the cells within the current layer.

4. **Printing:** The problem is solved for the printing step. This step consists in the application of the heat needed to fuse the powder of the current layer and the time increment is calculated as $\Delta t = t_{\mathrm{p}} = \mathrm{v}_{\mathrm{layer}}/d_{\mathrm{p}}$ [s].

5. **Cooling:** The problem is solved for the cooling step, accounting for lowering of building platform, recoat time and laser relocation with a time increment of $\Delta t = t_{\mathrm{r}}$ [s]. During the cooling step, the laser is off and the prism is allowed to cool down.

According to this workflow, the simulation of each layer is carried out in two time steps, printing and cooling, so the total number of time steps is $48 \cdot 2 = 96$. However, with the exception of the first layer, each new layer is meshed with a single refine and coarsen step. Hence, the simulation has about half as many AMR events as time steps. Besides, the linear system in Equation (5.5), arising at each time step, is solved with the Jacobi-preconditioned conjugate gradient (PCG) method with unit relaxation parameter.

The numerical experiments for this example were run at the Marenostrum-IV [130] (MN-IV) supercomputer, hosted by the Barcelona Supercomputing Center (BSC). It is equipped with 3,456 compute nodes connected together with the Intel OPA HPC network. Each node has 2x Intel Xeon Platinum 8160 multi-core CPUs, with 24 cores each (i.e. 48 cores per node) and 96 GBytes of RAM.

Apart from that, FEMPAR-AM was compiled with Intel Fortran 18.0.1 using system recommended optimization flags and linked against the Intel MPI Library (v2018.1.163) for message-passing and the BLAS/LAPACK library for optimized dense linear algebra kernels. All floating-point operations were performed in IEEE double precision.

The parallel framework is set up such that each subdomain is associated to a different MPI task, with a one-to-one mapping among MPI tasks and CPU cores. Regarding dynamic load balancing, the partition weights are set to $w_a = 10$ for active cells and $w_i = 1$ for inactive cells. Using linear FEs, the average total number of cells $N_{\text{cells}}$ and global DOFs $N_{\text{dofs}}$ (excluding hanging DOFs) across all time steps are 12,585,216 and 10,273,920. Note that, if a fixed uniform mesh was used, specifying the maximum refinement level of 11 all over the cube, the number of cells would be $(2^{11})^3 = 8.59 \cdot 10^9$ and the number of DOFs would grow from $(2^{11}+1)^2(2^{11}/2+1) = 4.30 \cdot 10^9$, initially, to $(2^{11}+1)^3 = 8.60 \cdot 10^9$, at the end of the simulation. Hence, it is readily exposed how $h$-adaptivity drastically reduces (almost by three orders of magnitude) the size of the problem and the required computational resources, while preserving accuracy around the growing printing region.

Figure 5.11 and Table 5.1 report speed-up and total simulation wall time [s] of FEMPAR-AM using the Jacobi-PCG method, as the number of subdomains is increased. As observed, FEMPAR-AM scales up to 6,144 fine tasks with a peak speed-up of 19.2. Above 6,144 cores, time-to-solution increases due to parallelism related overheads (e.g. interprocessor communication); more computationally intensive simulations (i.e. larger loads per processor) would be required to exploit additional computational resources efficiently. As observed, the total wall time reduces with the number of subdomains to approximately two minutes. This means that it takes 2.5 seconds in average to simulate the printing and cooling of a single layer (in two time steps). However, at larger scales of simulation and/or different problem physics, *weakly scalable* [2] methods, such as AMG or BDDC, may have superior performance.

Another point of interest is to analyse the fraction of total wall time spent in different phases of the simulation and their scalability, shown in Figure 5.12. As observed, the assembly phase dominates at low number of tasks, followed by the triangulation one. However, while assembly is the most scalable simulation phase, the triangulation is the least one. That is why the latter gradually dominates with increasing number of tasks and also leads the degradation of parallel efficiency. It is interesting to see that the solver phase is not relevant, with the exception of a (reproducible) spike in computation time at 786 and 1536 tasks. While the low-importance is caused by solving a relatively simple problem that can be efficiently preconditioned with the Jacobi method (the average number of iterations is merely 68), the abnormal deviation could

[2] Weak scalability is the ability of a parallel system to efficiently exploit increasing computational resources in the solution of a problem *with fixed local size per processor*. An *ideally* weakly scalable code does not vary the time-to-solution with the number of processors and fixed local size per processor. In other words, if the system solves a problem in time $t$ with a given amount of processors, then it is able to solve also in time $t$ an $X$ times larger problem with $X$ times the number of processors. The Jacobi method is not weakly scalable because the number of iterations grows with the global problem size.

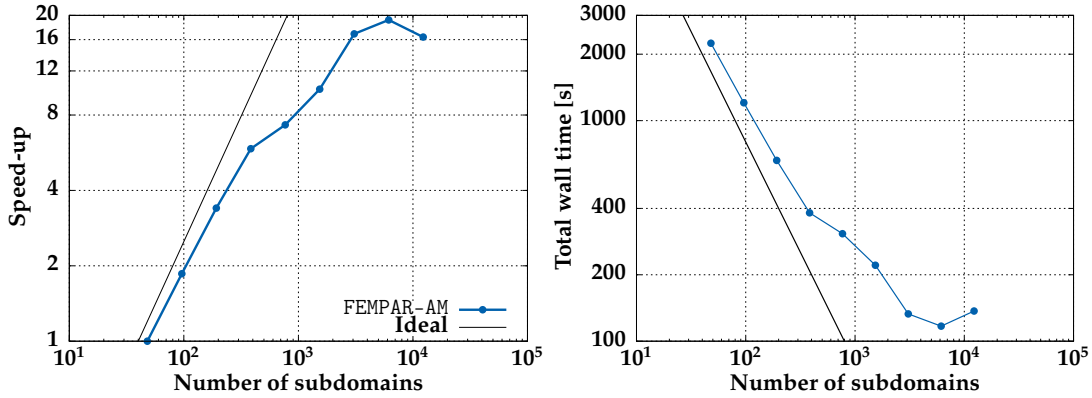



Figure 5.11: Strong-scaling example: Results of FEMPAR-AM for $(w_a, w_i) = (10, 1)$. Maximum speed-up of 19.2 and minimum wall time of 117 [s] is attained at 6,144 processors.

| **P** | **Total wall time [s]** | $\mathbf{S_P} = \frac{t_{48}}{t_P}$ | $\mathbf{E_P} = \frac{S_P}{P/48}$ | $\bar{\mathbf{n}}^{\mathbf{local}}_{\mathbf{dofs}}$ | $\bar{\mathbf{n}}^{\mathbf{iters}}$ |
|---|---|---|---|---|---|
| 48 | 2,244 | 1.00 | 1.00 | 222,355 | |
| 96 | 1,205 | 1.86 | 0.93 | 111,812 | |
| 192 | 660 | 3.40 | 0.85 | 56,390 | |
| 384 | 382 | 5.87 | 0.73 | 28,533 | |
| 768 | 307 | 7.31 | 0.46 | 14,508 | 68 |
| 1,536 | 221 | 10.15 | 0.32 | 7,418 | |
| 3,072 | 133 | 16.87 | 0.26 | 3,827 | |
| 6,144 | 117 | 19.18 | 0.15 | 1,993 | |
| 12,288 | 137 | 16.38 | 0.06 | 1,052 | |

Table 5.1: Strong-scaling analysis results of FEMPAR-AM for $(w_a, w_i) = (10, 1)$. Total wall time accounts for the computational time of all simulation stages. $S_P$ is the speed-up, $E_P$ is the parallel efficiency, $\bar{n}^{\text{local}}_{\text{dofs}}$ is the average size of the local fine problem across processors and time steps and $\bar{n}^{\text{iters}}$ is the average number of iterations of the Jacobi-PCG solver across time steps.

be explained by the irregularity of the partition, although the authors were not able to find clear correlation. Another particularity related to the construction of the problem is that, when following a z-ordering, active and inactive cells are generally mixed. Hence, a standard $(w_a, w_i) = (1, 1)$ partition more or less equally distributes the active cells. It follows that there is a rather low sensitivity to the partition weights.

Further insights are drawn by studying the local distribution of cells and DOFs in space (among processors) and in time (among layers) up to 3,072 processors. As the number of cells and DOFs vary for each layer, all geometrical quantities are studied in terms of the mean $\mu$ and the coefficient of variation $c_v$, also known as relative standard deviation, which is the ratio of the standard deviation $\sigma$ to the mean $\mu$. $c_v$ measures the extent of variability in relation to $\mu$. Thus, it can be used to compare the variability

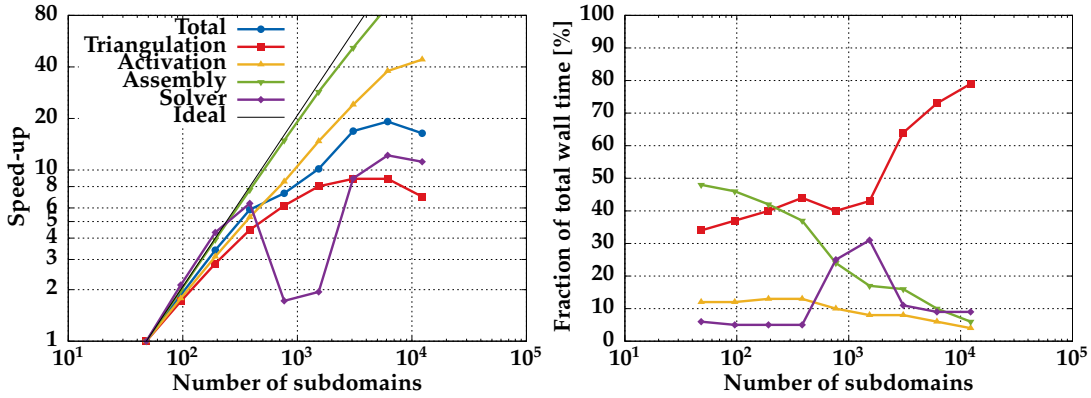



Figure 5.12: Strong-scaling example: Results of FEMPAR-AM for $(w_a, w_i) = (10, 1)$ per simulation phases. The *triangulation* phase accounts for the remesh and redistribute steps of the simulation workflow, including projections and redistributions of the FE solution, whenever the mesh is transformed or redistributed. The *activation* phase accounts for the search of activated cells and the generation of the FE space with new DOFs assigned within the current layer. The *assembly* phase consists of local integration of the weak form and construction of the global linear system, during printing and cooling steps. Finally, the *solver* phase includes the solution of the linear system, also during printing and cooling steps. Although the assembly phase is initially dominant, computational time and scalability are dominated by the triangulation phase.

among different quantities.

Given a quantity $x(t, p)$ that can depend on the time step and processor, the time average at every processor is represented as $\mu^t(x)$, the average among processors at every time step as $\mu^p(x)$ and the mean value across processors and time steps as $\mu(x)$. In what follows, the magnitude of $x$ is studied with $\mu(x)$, whereas dispersion is analysed with the coefficient of variation among processors, i.e. $c_v^p(x) = \sigma^p(x)/\mu^p(x)$. This statistic informs about possible computational load unbalances, due to an uneven distribution of $x$ among processors. As $c_v^p(x)$ depends on the time step, for the sake of simplicity, the average across time steps $\mu^t(c_v^p(x))$ is reported instead.

Table 5.2 gathers the local distribution of cells and degrees of freedom (excluding the hanging ones). The values of $\mu^t(c_v^p(x))$ show that the local number of cells is slightly unbalanced, but the local weighted number of cells, i.e. the sum of the cell weights at each processor for $(w_a, w_i) = (10, 1)$, is perfectly balanced. Apart from that, Figure 5.13 shows that the number of cells oscillates with the height of the layer, but it does not grow in time. This behaviour propagates to other quantities such as the number of active cells or DOFs and it is caused by the 2:1 balance: The cells concentrate at the current layer and immediately below. Even if the domain grows in time, the number of cells away from the layer is much smaller than the number of those close or at the layer.

It is also apparent that the pair $(w_a, w_i) = (10, 1)$ effectively equilibrates the number of active cells among processors. Hence, degrees of freedom are also evenly distributed, though with a slightly higher dispersion. This is especially beneficial for the

| | $\mathbf{n}_{\text{cells}}$ | | $\mathbf{n}_{\text{cells}}^{\text{weighted}}$ | | $\mathbf{n}_{\text{cells}}^{\text{active}}$ | | $\mathbf{n}_{\text{dofs}}$ | |
|---|---|---|---|---|---|---|---|---|
| **P** | $\mu$ | $\mu^t(c_v^p)$ | $\mu$ | $\mu^t(c_v^p)$ | $\mu$ | $\mu^t(c_v^p)$ | $\mu$ | $\mu^t(c_v^p)$ |
| 48 | 262.2k | 0.61 | 2,228k | 0.00 | 218.5k | 0.08 | 222.4k | 0.26 |
| 96 | 131.0k | 0.75 | 1114k | 0.00 | 109.2k | 0.10 | 111.8k | 0.37 |
| 192 | 65.5k | 1.02 | 557k | 0.01 | 54.6k | 0.14 | 56.4k | 0.45 |
| 384 | 32.8k | 1.44 | 279k | 0.01 | 27.3k | 0.20 | 28.5k | 0.63 |
| 768 | 16.4k | 2.28 | 139k | 0.02 | 13,7k | 0.31 | 14.5k | 0.73 |
| 1,536 | 8.2k | 3.46 | 69.6k | 0.05 | 6,8k | 0.48 | 7.4k | 1.10 |
| 3,072 | 4.1k | 5.46 | 35.8k | 0.09 | 3,4k | 0.76 | 3.8k | 1.39 |

Table 5.2: Strong-scaling example. FEMPAR-AM for $(w_a, w_i) = (10, 1)$. Local distribution of cells and DOFs (excluding hanging). $\mu^t(c_v^p)$ expressed in %.

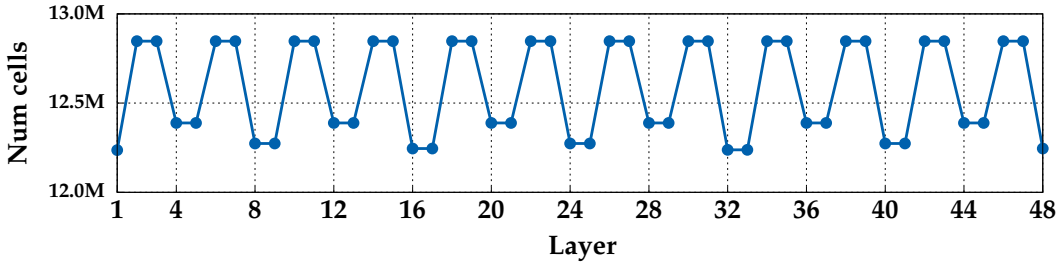



Figure 5.13: Evolution of global number of cells with the height of the layer.

integration and assembly phases, as they are implemented in FEMPAR, such that the bulk of the computational load is concentrated on the active cells set, and also the linear solver phase, as the size of the local systems depend on the number of DOFs that the processor owns. On the other hand, mesh generation, refinement, coarsening and redistribution phases suffer from an uneven distribution of total cells.

Concerning the local number of subdomain neighbours and interface DOFs, in Table 5.3, high variations in space expose the extreme irregularity of Z-curve partitions. Due to the refinement strategy in Figure 5.10(b) and the Z-ordering, most subdomains are embedded in the current layer, while only few are made of bottom coarser cells. This explains why, as seen in the fourth column, listing the maximum number of neighbours across processors and time steps, there can be subdomains touching all the remaining ones. Even if the partition becomes increasingly regular with $P$, the imbalance of neighbours and interface DOFs increases synchronization times in important operations of the Jacobi-PCG, such as the matrix-vector product. It could also be the main cause for the deviation observed in the solver times of Figure 5.12.

### 5.6.3 Extruded wiggle

If the previous example focused in performance, the one in this section aims to show the capabilities of the framework in a more realistic scenario. The setting now is the

| | $\mathbf{n}_{\mathbf{neighbours}}$ | | | $\mathbf{n}_{\mathbf{dofs}}^{\mathbf{interface}}$ | |
|---|---|---|---|---|---|
| **P** | $\mu$ | $\mu^t(c_v^p)$ | **max** | $\mu$ | $\mu^t(c_v^p)$ |
| 48 | 14 | 47.9 | 48 | 7.1k | 35.1 |
| 96 | 16 | 62.5 | 95 | 4.8k | 34.4 |
| 192 | 17 | 79.4 | 191 | 3.3k | 28.3 |
| 384 | 18 | 91.1 | 383 | 2.3k | 25.8 |
| 768 | 19 | 103.6 | 767 | 1.6k | 20.5 |
| 1,536 | 20 | 92.0 | 1,024 | 1.1k | 20.7 |
| 3,072 | 21 | 80.1 | 1,048 | 0.8k | 16.6 |

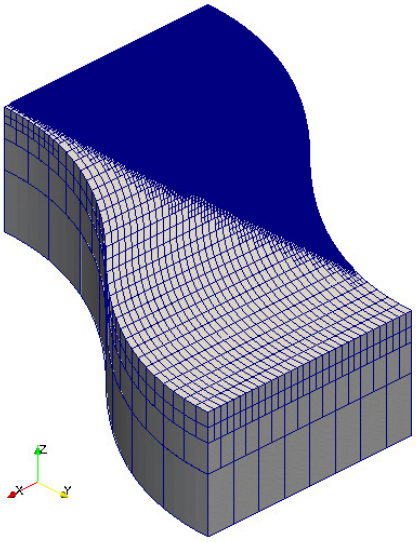
Table 5.3: Strong scaling example. Local distribution of subdomain neighbours and interface DOFs. $\mu^t(c_v^p)$ and $c_v$ expressed in %.

printing of a 3D curved geometry *following the actual scan path*, instead of a layer-averaging approach. In this way, the geometry is no longer rectangular, the temperature field is captured with higher detail and, more importantly, the parallel search algorithm introduced in Section 5.3.2 is more intensively stressed.

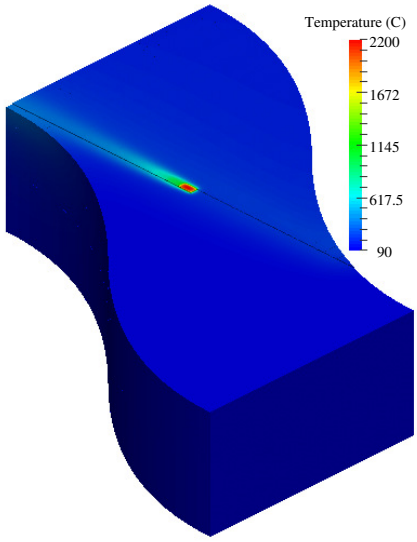
The simulation spans the build of eight 60 [µm] layers on top of a 7.68 [mm] height extruded wiggle. The geometry, represented in Figure 5.14, is obtained in the following way: (1) a one-dimensional wiggle is defined by joining two parabolic functions, the distance between its edges is 30.72 [mm]; (2) extrusion along the $x$-axis yields a 15.36 [mm] thick plane wiggle; (3) extrusion perpendicular to the xy-plane completes the 3D shape.

One feature of the simulation is the use of *second order* lagrangian FEs both for geometry and continuous FE space approximations. In particular, given that the wiggle is parametrized by (piecewise) polynomials of order less or equal than two, an exact discretisation can be easily defined. Although this choice clashes with the rectangularity assumption of the search algorithm in Section 5.3.2, if the HAV overlaps a refined enough region, the mesh cells can be regarded as quasi-rectangular. As a result, the meshing approach considers two different user-prescribed minimum levels of refinement: the (1) *absolute* one, a lower bound of the level of any cell in the mesh, and the (2) *search* one, i.e. the lowest level a cell below the layer being printed must have to ensure quasi-rectangularity. It is apparent that (1) is always lower or equal than (2).

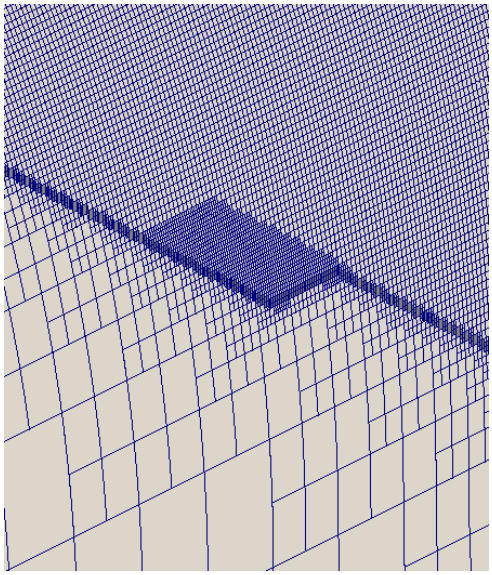
The octree mesh is constructed by working on two different configuration spaces; `p4est` generates and transforms the mesh in a *reference* unit cube, whereas `FEMPAR` maps the unit cube mesh coordinates into the *physical* wiggle. The root octant of the octree mesh is mapped into a 30.72 [mm] height wiggle. The absolute minimum level of refinement is set to three, enough to have an exact discrete geometry; the search one is set to five, after several trial and error experiments; and the maximum one at the HAV is ten, i.e. there are two cells along the thickness of the layer. This setting is apparent in Figures 5.14(a) and 5.14(c). The initial computational domain has a height of 7.68



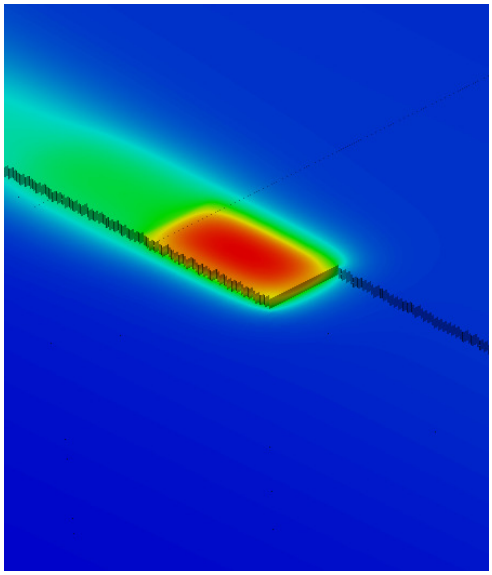
(a) The absolute minimum level of refinement at the bottom of the wiggle is three, while the search one below the current layer is at least five. A blue shaded area indicates the highly-refined cell concentration at the current layer.




(b) Contour plot of temperatures for the given time step. The relevant thermal features are localized around the HAV and its tail.



(c) Close-up of the mesh around the HAV. The search algorithm finds the cells inside the HAV on top of a quasi-rectangular region and assigns to them the maximum level of refinement (10).



(d) Close-up of the contour plot around the HAV. Away from the HAV, highly-refined layer cells are only necessary to capture the growing geometry; they do not increase the resolution.

Figure 5.14: Mesh and temperature contour plot of the 3D wiggle at an intermediate time step of the simulation of the first layer for $(w_a, w_i) = (10, 1)$.

[mm], cells located above this threshold are inactive. After printing the eight layers, the active region reaches 8.16 [mm] height.

Regarding the laser path, odd layers are built with nonoverlapping hatches along

the $x$-axis, whereas even ones along the $y$-axis. Even though the scanning path is orthogonal, the mesh cells are skewed due to the cube-to-wiggle mapping. Therefore, during the simulation, the search algorithm generally tests the intersection of non-aligned cuboids. A uniform heat input of 200 [W] and heat absorption $\eta$ of 0.5 is distributed in a $0.96 \times 0.48 \times 0.06$ [mm] HAV. Note that the HAV has been scaled 1% extra in the $xy$-plane for safety, a practice that is advisable even for linear FEs to avoid failures of the search algorithm in situations that depend on arithmetic precision. The scanning speed is 100 [mm/s] and the relocation speed is 200 [mm/s]. This means that the time step during printing is $0.96/100 = 9.6 \cdot 10^{-3}$ [s].

The material is, once again, Ti6Al4V, and the initial and boundary conditions are also the same of the previous example, i.e. $u_0 = 90$ [°C] and convection on the boundary ($h_{\text{out}} = 50$ [W/m$^2$T] and $u_{\text{out}} = 35$ [°C]). An important simplification is that the powder is not simulated. Although modelling the powder bed has already been object of study by the authors in Chapters 4 and 7, it is left out of this example, because it does not contribute to the main purpose, i.e. the study of the search algorithm.

The simulation workflow is the same as the one in Section 5.6.2, but the time discretisation is set up such that, at each time step, 0.96 [mm] along the laser path are advanced, not a whole layer. Simulation steps (1) to (4), i.e. mesh transformation, activation and printing step, are carried out at every 0.96 [mm] step, while cooling (5) steps only apply between hatches and layers. In this strategy, the mesh explicitly tracks the laser path, but there are many more AMR events than time steps, because the next HAV is not accommodated in a single refine and coarsen step, but (at most) $11 - 5 = 6$ steps. Although not covered in this chapter, other simulation strategies could be analysed. For instance, one could mesh the whole layer as in Section 5.6.2, then simulate printing following the laser path point to point. Compared to the laser-tracking mesh approach, there would be many more time steps than mesh transformations, but the problem solved at each time step would also be bigger.

Numerical experiments were run with 96 MPI tasks one-to-one assigned to 96 cores, distributed in six computing nodes. Each node has 2x Intel Xeon E5-2670 multi-core CPUs, with 8 cores each and 64 GBytes of RAM. The computing nodes were located at the Acuario [4] computer cluster, hosted by the International Centre of Numerical Methods in Engineering (CIMNE) in Barcelona, Spain. Linear systems were solved with a nonrelaxed Jacobi-PCG method. Contour plots were generated in ParaView 5.1.0 [12] with second order visualization cells written in VTK format [168].

Simulation results at several time steps are gathered in Figures 5.14 and 5.16 for $(w_a, w_i) = (10, 1)$. Average global problem size is 2.0M (nonhanging) DOFs and total number of time steps is 8,727. For 96 MPI tasks, the average number of local DOFs is 21.1k and $\mu^t(c_v^p(x)) = 3.3\%$, with $c_v^p(x) = \sigma^p(x)/\mu^p(x)$. In front of the strong-scaling example, dispersion of local DOFs is higher and the total number of DOFs notably oscillates in time (Figure 5.15) due to the laser-tracking refinement strategy.

The duration of the simulation is 13 [h]; each layer is printed in 97 [min] average,

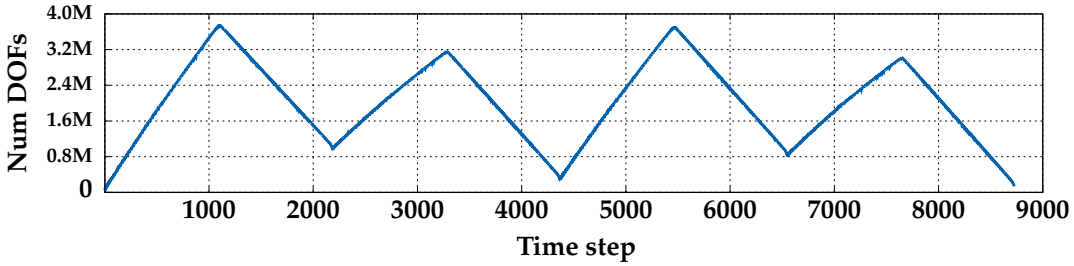
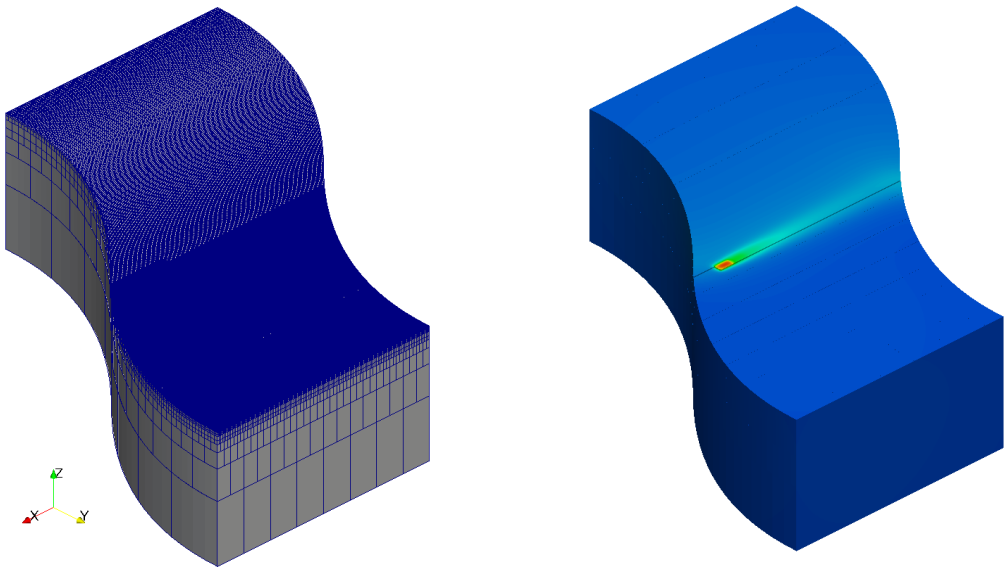



Figure 5.15: Total number of (nonhanging) DOFs heavily oscillates with the laser-tracking refinement approach. Increasing/decreasing trend reverses at the start of a new layer.

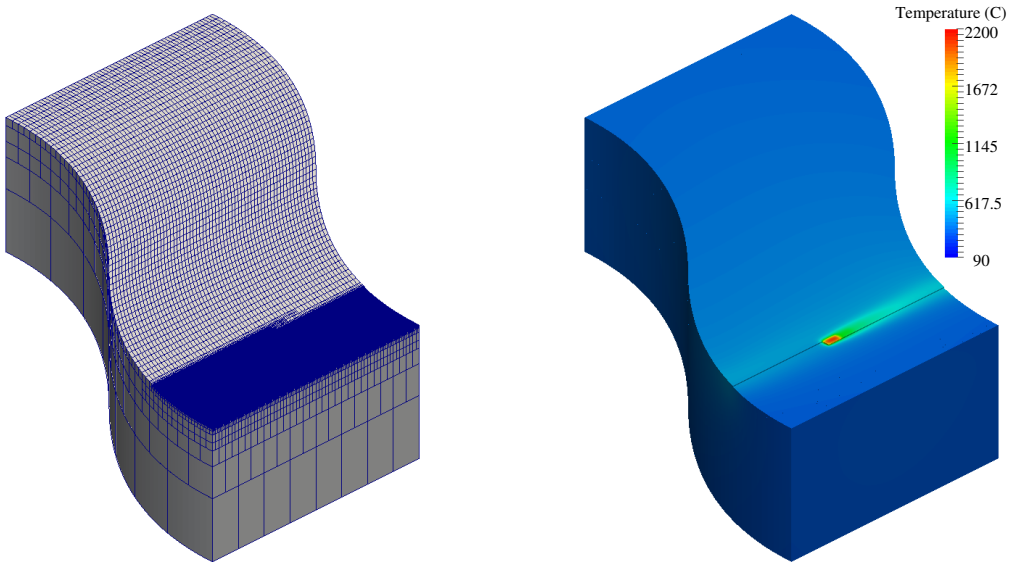
but the individual time varies significantly, due to the oscillation of total number of DOFs. On the other hand, average number of Jacobi-PCG iterations is 381. In spite of being numerous, mesh generation, adaptation and redistribution roughly amounts to 52% of the simulation time. Next phases by runtime are local integration and assembly (22%) and linear solver (24%). An important outcome is that activation with the HST nonitersection test and initialization of new DOFs only occupies 3% of the simulation. Moreover, mesh and contour plots (e.g. Figure 5.16(b)) do not expose any holes during the filling of the layers. Therefore, the parallel search algorithm is both efficient and robust in this example.

Apart from that, there is a pronounced sensitivity to the partition weights. Indeed, Table 5.4 gathers distribution of DOFs and runtimes for several pairs of $(w_a, w_i)$. A weighted partition decreases up to 34% the computation time with respect to a non-weighted one. Runtime decrease affects especially the assembly and solver phases, whose complexity depends on the number of active cells or DOFs, respectively. Reductions could be more significant for larger number of subdomains $P$, because the disequilibrium of local DOFs grows with $P$. Hence, partition weights not only dynamically equilibrate the DOFs among processors, they also have the potential to significantly shorten simulation times. Besides, in relation to the discussion at the end of Section 5.3.3, performance of pairs $(w_a, w_i)$ satisfying $w_a \gg w_i$ is barely distinguished from $(w_a, w_i) = (1, 0)$, i.e. they appear capable of equilibrating the computational load, while precluding extreme imbalance of cells.

A final comment arises on two computational bottlenecks identified in this example that guide how the framework could be further improved, towards dealing with real industrial scenarios. The first and most obvious one is the layer-conforming mesh constraint that precludes coarsening at cells touching the active/inactive interface (e.g. Figure 5.16(a)). Unfitted FE methods could likely get rid of this requirement and dramatically drop the mesh size. On the other hand, even if concurrency in space is efficiently exploited, the framework is still fully sequential in time. Very small time steps must be prescribed to follow the laser with high precision, leading to extremely long transient simulations. This motivates exploring adaptive space-time approximations and solvers [22, 79, 81]. The idea is to provide the solution at all time steps in one shot using




(a) As the mesh must be layer-conforming, many cells concentrate at the last printed layers, even though they do not contribute to capture better the thermal field.




(b) Homogeneous (same level) coarsening at the last layer indicates that there are no interior or boundary holes, i.e. cells that the search algorithm has failed to activate.

Figure 5.16: Mesh and temperature field at intermediate time steps of the sixth 5.16(a) and eighth 5.16(b) layers.

the vast computational resources available nowadays.

## 5.7 Conclusions

This chapter has introduced a novel HPC numerical framework for growing geometries that has been applied to the thermal FE analysis of metal additive manufacturing

| | $\mathbf{n_{dofs}}$ | | Runtime | | Phase percentage [%] | | | |
|---|---|---|---|---|---|---|---|---|
| $(\mathbf{w_a}, \mathbf{w_i})$ | $\mu$ | $\mu^t(c_v^p)$ [%] | T [h] | $T/T_{(1,1)}$ [%] | Triang. | Activ. | Assemb. | Solver |
| (1,1) | 21.1k | 25.6 | 19.7 | 100 | 48 | 3 | 26 | 23 |
| (2,1) | | 15.4 | 15.2 | 77 | 51 | 2 | 23 | 24 |
| (10,1) | | 3.3 | 13.0 | 66 | 52 | 3 | 22 | 24 |
| (100,1) | | 3.0 | 13.0 | 66 | 52 | 2 | 23 | 23 |
| (1,0) | | 2.4 | 13.0 | 66 | 51 | 2 | 23 | 24 |

Table 5.4: Sensitivity of DOF distribution and computation times to the partition weights. Simulation phases correspond to the same ones described in Figure 5.12. Compared to the nonweighted case, partition weights equilibrate the DOF distribution and reduce up to 34% the total simulation runtime.

by powder-bed fusion at the part scale. The abstract framework is constructed from three main blocks (1) hierarchical AMR with octree meshes, distributed across processors with space-filling curves; (2) the extension of the element-birth method to (1); to track the growth of the geometry with an embarrassingly parallel search algorithm, based on the hyperplane separation theorem; and (3) state-of-the-art parallel iterative linear solvers.

After implementation in `FEMPAR`, an advanced object-oriented FE code for large scale computational science and engineering, a strong-scaling analysis, of a problem with 10M unknowns, evaluated the performance of the code up to 12,288 CPU cores. Timing of simulation phases and statistical treatment of cell- and DOF-related quantities revealed uneven DOF distribution and partition irregularity as the main sources of load imbalance, thus increasing parallel overhead.

A second numerical example of 2.0M unknowns in a curved geometry examined the search algorithm, driving the update of the computational domain. The results not only verified efficiency and robustness of the search routine, but also showed that the mesh rectangularity assumption of the algorithm can be slightly relaxed. A further study exposed the potential of partition weights to tune dynamic load balance and bring down simulation times; the weight function has been defined to seek a compromise between equally distributing among processors the number of cells or DOFs.

Average simulation times *per time step* were cut down to 1.2 [s] and 5.4 [s] in the first and second examples. Even at the rate of 1 [s] per time step, fully resolved high-fidelity AM simulations, involving, e.g. other physics than heat transfer, stronger nonlinearities, historical variables, among others, can still take long hours in massively parallel systems. Higher efficiency and parallelism could be attained by resorting to (1) unfitted FE methods [24] to eliminate the mesh body- *and* layer-fitting requirement and (2) weakly scalable adaptive and nonlinear space-time solvers that also exploit concurrency in time.

In spite of the limitations, `FEMPAR-AM` is, to the authors knowledge, the first *fully* parallel and distributed FE framework for part-scale metal AM and the numerical experiments have shown the potential to efficiently address levels of complexity and accuracy

unseen in the literature of metal AM. Apart from turning to other part-scale physics in metal or polymer AM, this HPC framework could also be useful for other growing-geometry problems, as long as the growth is modelled by adding new elements into the computational mesh.

# Chapter 6

# Model reduction by scan-pass lumping in AM by powder-bed fusion

*The contents of this chapter correspond to the research publication*

[55] M. CHIUMENTI, EN, E. SALSI, M. CERVERA, S. BADIA, J. MOYA, Z. CHEN, C. LEE AND C. DAVIES, *Numerical modelling and experimental validation in Selective Laser Melting*, Additive Manufacturing, 18 Supplement C (2017), pp. 171-185.

In this chapter a finite-element framework for the numerical simulation of the heat transfer analysis of additive manufacturing processes by powder-bed technologies, such as Selective Laser Melting (SLM), is presented. These kind of technologies allow for a layer-by-layer metal deposition process to cost-effectively create, from a Computer-Aided Design (CAD) model, complex functional parts such as turbine blades, fuel injectors, heat exchangers, medical implants, among others. The numerical model proposed accounts for different heat dissipation mechanisms through the surrounding environment and is supplemented by a finite-element activation strategy, based on the born-dead elements technique, to follow the growth of the geometry driven by the metal deposition process, in such a way that the same scanning pattern sent to the numerical control system of the AM machine is used. An experimental campaign has been carried out at the Monash Centre for Additive Manufacturing using an EOSINT-M280 machine where it was possible to fabricate different benchmark geometries, as well as to record the temperature measurements at different thermocouple locations. The experiment consisted in the simultaneous printing of two walls with a total deposition volume of 107 $cm^3$ in 992 layers and about 33,500 s build time. A large number of numerical simulations have been carried out to calibrate the thermal FE framework in terms of the thermophysical properties of both solid and powder materials and suitable boundary conditions. Furthermore, the large size of the experiment motivated the investigation of two different model reduction strategies: exclusion of the powder-bed from the computational domain and simplified scanning strategies. All these methods are analysed in terms of accuracy, computational effort and suitable applications.

## 6.1 Introduction

Additive manufacturing (AM) or 3D Printing refers to a group of manufacturing processes that build up a three-dimensional object layer upon layer, directly from a CAD model. This technology has been traditionally used for rapid prototyping using plastic materials. Nowadays, it allows for the 3D printing of metallic components ready to be exploited for many industrial applications.

The most important benefit of AM is the ability to cost-effectively create objects with shapes and properties that were previously near-impossible to produce with conventional manufacturing processes, such as casting or forging. AM can easily print very complex geometries with cavities, thin walls or lattice structures and it is also competitive for customised design in a short production time.

This chapter addresses the numerical simulation of AM processes of metal components by powder-bed technologies, such as SLM, DMLS or Electron Beam Melting (EBM). Many industrial sectors are adopting them to fabricate their products, such as turbine blades, fuel injectors, and microturbines in aerospace and aeronautics; wheel suspensions, heat exchangers, and break disks in the automotive industry; dental bridges and implants in the medical industry, or even jewels in the consumer goods sector.

A typical printing process by powder-bed technology, such as DMLS in Figure 6.1, occurs in a closed chamber with a gas controlled atmosphere and consists of the following steps:

1. A new layer of powder, around *30-60 microns* thick, is spread over the building platform with the levelling blade.

2. A high-energy and focused laser melts the region of powder that belongs to the current cross section of the object. The laser moves according to a predefined *scanning path*.

3. The building platform is lowered to accommodate a new layer.

4. Steps 1. to 3. are repeated until the whole model is created.

5. Loose unfused powder is removed during post processing.

Nowadays, process design and certification of products built by metal AM relies on an expensive and time-consuming trial-and-error procedure. This situation prevents a wider adoption of these technologies by the industry. In order to leverage the freedom in design, cost efficiency and immediacy that AM offers, one possible solution is to shift to a virtual-based design, using predictive computer simulation tools.

Many researchers have already used the Finite-Element (FE) method to analyse metal deposition processes in AM with different technologies [8, 33, 65, 109, 127, 131, 159, 198], often taking advantage from the knowledge acquired in modelling other metal forming processes, such as casting or welding [48, 53, 63, 66, 83, 122].

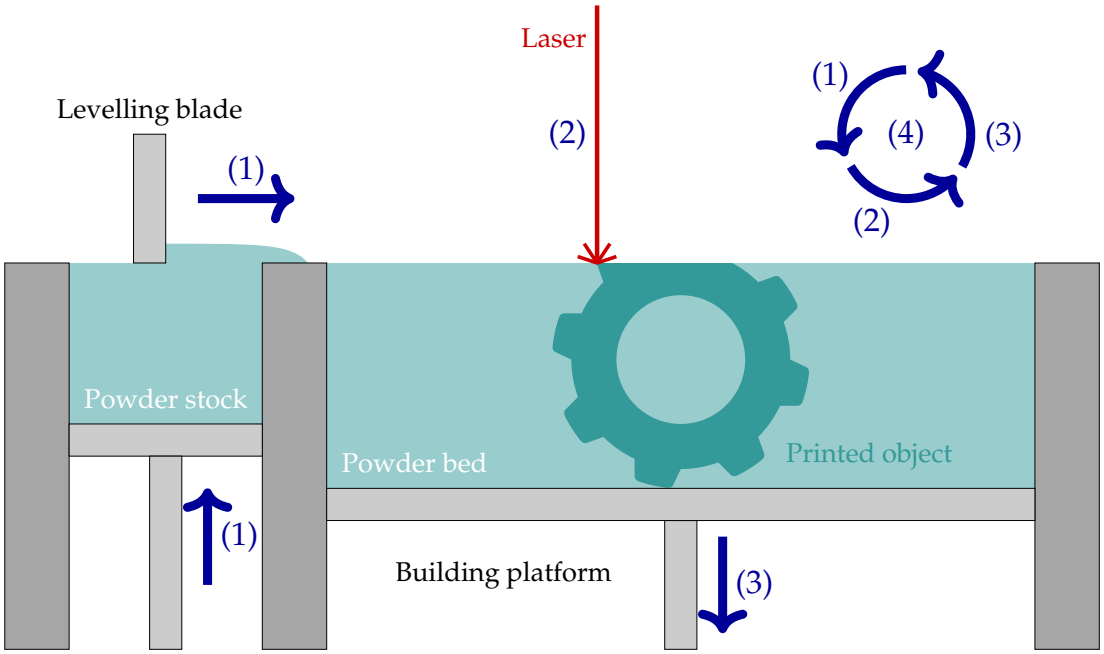



Figure 6.1: A printing process by DMLS. (1) A new layer of powder, around *30-60 microns* thick, is spread over the building platform with the levelling blade. (2) A laser source melts the region of powder that belongs to the current cross section of the object. (3) The building platform is lowered to accommodate a new layer. (4) Steps 1. to 3. are repeated until the whole model is created. (5) Loose unfused powder is removed during post processing.

FE modelling has been mainly employed to assess the influence of process parameters [150, 173, 199] and to evaluate distortions and residual stresses [60, 68, 72, 112, 154]. In this sense, thermal modelling, apart from being an input for the mechanical analysis, is fundamental to characterise the melt pool and the microstructure in an AM process [95, 114, 153, 190] and also guides the selection of the printing process parameters in engineering design [120, 152, 197].

The scope of this chapter is the FE analysis of the AM process by metal deposition at the component scale. Hence the focus is the study of the heat transfer process according to the energy introduced into the system by the heat source (laser, electron beam, etc.), as well as the heat dissipation through the boundaries of the domain which define the component during its fabrication. The phenomena occurring in the melt pool and in the surrounding heat affected zone (HAZ) can also be analysed [51, 82, 113, 155, 161, 183], but are out of the scope of this chapter.

Furthermore, two aspects deserve especial attention when modelling powder-bed technologies. First, the lateral walls of the component are in permanent contact with the unsintered powder throughout the printing and cooling processes. As a consequence, heat conduction through the powder must be modelled. Second, the layer thickness is typically very small (about 30 to 60 microns), so that building industrial parts requires the deposition of thousands of layers. Therefore, computational efficiency is paramount.

Computational complexity is one of the reasons why most experimental studies with powder-bed technologies consider short single-part builds of less than 15 layers and 1 $cm^3$ volume [69, 109, 150, 159]. Fewer works attempt at higher deposition volumes [67, 154], longer processes [72], or multiple parts [92, 154], but barely any of these experimental builds approach the limits of current machines.

Besides, strict discretization requirements [200], specifying mesh sizes smaller than the laser beam spot size, are necessary to obtain an accurate local thermal response, especially, to capture the peak temperature distribution, but they also increase the computational load to a point where engineering applications are out of reach.

Several methods have been introduced to overcome this burden. On the one hand, adaptive mesh refinement and coarsening have been explored to reduce the size of the spatial problem [67, 151]. On the other hand, reduced models with simplified scanning strategies have been introduced [92, 97]. These models are not capable of predicting the complex thermal history (local superheating and supercooling). Therefore, they are not suitable for further mechanical or microscale analyses. However, they are able to capture an accurate global thermal response. For this reason, they are a reliable and efficient alternative for other engineering applications, such as optimisation of the process parameters or process planning. In spite of the benefits, the authors consider that these reduction strategies have been object of few numerical analysis and contrast with physical experiments.

This given, the purpose of this chapter is to enhance the FE framework developed and experimentally validated for both wire-feeding [52] and blown-powder technologies [54] to deal with the thermal analysis of AM processes by powder-bed technology. This task is supported by an exhaustive experimental campaign carried out at the *Monash Centre for Additive Manufacturing* (MCAM) in Melbourne, Australia, using an EOSINT M280 machine and Ti-6Al-4V powder. The scale of the experiment is unprecedented and representative of big industrial cases: simultaneous printing of two 95 $cm^3$ and 12 $cm^3$ walls in 992 layers and about 33,500 s build time.

The computational framework proposed here is calibrated by comparing the temperature evolution obtained at different thermocouple locations during the full duration of the AM process with the corresponding experimental measurements. The experimental setting is also used to investigate different numerical approaches, in order to find the best simulation practice when dealing with powder-bed technologies.

With regards to the thermal loss through the powder, two alternatives are examined: (1) including the powder-bed into the computational model with appropriate estimations of the thermophysical properties of the powder, as done in [69, 154], or (2) a novel approach that excludes the powder-bed and models the corresponding heat loss with an equivalent heat flux through the lateral walls of the component as immersed into the unsintered powder.

As for the computational complexity, this chapter assesses the impact of considering simplified scanning strategies on the accuracy of the solution and the simulation

time. The analysis ends with a comparative evaluation of the advantages and disadvantages of each model reduction approach and recommended applications.

The outline of this chapter is as follows. First, the formulation of the heat transfer problem is detailed in Section 6.2. The FE *activation* technique used to simulate the metal deposition is explained in Section 6.3. Section 6.4 describes the experimental setting at the MCAM. The calibration of the numerical model and the evaluation of the different simulation approaches is addressed in Section 6.5. Finally, Section 6.6 presents the conclusions of this chapter.

## 6.2 Heat transfer analysis

### 6.2.1 Governing equation

Let $\Omega$ be an open bounded domain in $\mathbb{R}^3$ with smooth boundary $\partial\Omega$, representing a thermodynamic continuum. $\Omega$ grows in time during the fabrication process. After the printing, it remains fixed, while cooling down to the room temperature.

The governing equation to describe the temperature evolution during the printing and the cooling phases of the AM process is the balance of energy equation, expressed as

$$\dot{\mathrm{H}} = -\boldsymbol{\nabla} \cdot \mathbf{q} + r, \quad \text{in } \Omega, \ t > 0, \tag{6.1}$$

where $\dot{\mathrm{H}}$ is the rate of enthalpy per unit volume, $r$ the heat supplied to the system per unit volume by the internal sources and $\mathbf{q}$ the heat conduction flux.

For an AM process, the heat source $r(t)$ is the energy input from a very intense and concentrated laser beam that moves in time according to a user-defined deposition sequence, the *scanning path*.

The enthalpy rate per unit volume $\dot{\mathrm{H}}$ is defined, in terms of the temperature $u$ and the rate of the latent heat released/absorbed during the phase-change process $\dot{L}$, as

$$\dot{\mathrm{H}}(u, \dot{L}) = C(u)\dot{u} + \rho(u)\dot{L},$$

where $C(u)$ is the (temperature dependent) heat capacity coefficient, given by the product of the density of the material $\rho(u)$ and the specific heat $c(u)$.

The amount of latent heat is negligible in front of the energy input introduced by the heat source. Moreover, in the HAZ, latent heat is absorbed when the laser fuses the powder particles. Immediately after, the material solidifies and the same amount of latent heat is released. These two phase transformations occur very fast, compared to the thermal diffusion process. As a result, the energy balance due to the phase change is null and very localised at the HAZ, so its global effect is minor in the heat transfer analysis [54].

According to this, the balance of energy equation can be stated as

$$C(u)\dot{u} - \boldsymbol{\nabla} \cdot (k(u)\nabla u) = r, \quad \text{in } \Omega, \quad t > 0,$$

where the conduction heat flux per unit area $\mathbf{q}$ is proportional to the gradient of temperature, according to Fourier's law:

$$\mathbf{q} = -k(u)\,\nabla u,$$

with $k(u) > 0$ the (temperature-dependent) thermal conductivity.

### 6.2.2 Boundary conditions.

Due to the high conductivity of metals, heat conduction through the building platform and thermal loss through the loose powder in which the component is immersed are the predominant heat transfer mechanisms in powder-bed technology. However, heat radiation and convection through the surfaces in contact with the environment must also be accounted for. Figure 6.2 illustrates the boundary conditions of the problem.

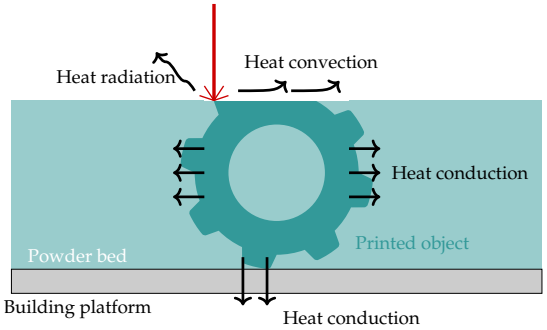



Figure 6.2: A close-up of Figure 6.1 gathering the boundary conditions of the problem: (1) Heat conduction through the building platform. (2) Heat conduction through the powder bed. (3) Heat convection and heat radiation through the free surface.

Considering a partition $(\partial\Omega_{\text{base}}, \partial\Omega_{\text{bed}}, \partial\Omega_{\text{air}})$ of the boundary $\partial\Omega$, where $\partial\Omega_{\text{base}}$ represents the contact surface with the printing platform, $\partial\Omega_{\text{bed}}$ the contact surface with the powder-bed and $\partial\Omega_{\text{air}}$ the external surface in contact with the surrounding environment, the boundary conditions are expressed as:

#### Heat conduction through the building platform.

Typically, the dimensions of the building platform (e.g. its thermal inertia) are much larger than the printed part. Hence, it is possible to prescribe the temperature on the contact surface $\Omega_{\text{base}}$ as

$$u = u_{\text{base}}, \quad \text{on } \partial\Omega_{\text{base}},$$

where $u_{\text{base}}$ is the temperature of the building platform.

**Heat conduction through the powder bed.**

If the powder bed is included in the computational domain, the thermophysical properties of the powder are established in terms of the properties of the solid material and the porosity of the granular bed, $\phi$.

The density and the specific heat are straightforwardly computed as

$$\rho_{\text{pwd}} = \rho_{\text{solid}}(1-\phi), \quad \text{and}$$
$$c_{\text{pwd}} = c_{\text{solid}},$$

but the value of the thermal conductivity, $k_{\text{pwd}}$, of metal powders is frequently estimated with empirical expressions, that also depend on the conductivity of the surrounding air or gas, $k_{\text{gas}}$. Among several models proposed in the literature, the work of Yagi and Kunii [193] for porous beds of metals, revised by Xue and Barlow [191] for low porosity powders, states

$$k_{\text{pwd}} = \left(6.3 + 22\sqrt{0.09 k_{\text{solid}} - 0.016}\right) \frac{k_{\text{solid}}(1-\phi)}{(k_{\text{solid}}/k_{\text{gas}})(10^{0.523-0.594\phi}) - 1}. \tag{6.2}$$

Alternatively, if the powder bed is *not* included in the computational domain, the heat loss by conduction through the powder $q_{\text{bed}}$ can be expressed using an equivalent boundary condition, as

$$q_{\text{bed}}(u) = h_{\text{bed}}(u - u_{\text{bed}}), \quad \text{on } \partial\Omega_{\text{bed}},$$

where $u_{\text{bed}}$ is the temperature of the powder far enough from the HAZ, and $h_{\text{bed}}(u)$ denotes the heat transfer coefficient (HTC) by conduction between the powder and the component.

$u_{\text{bed}}$ should be known or duly estimated in time during the full duration of the AM process, but a constant value can be used, if the thermal interference among the different components, being printed on the same platform at the same time, is negligible.

On the other hand, $h_{\text{bed}}(u)$ is computed as

$$h_{\text{bed}}(u) = \frac{k_{\text{pwd}}(u)}{s_{\text{pwd}}},$$

where $s_{\text{pwd}}$ accounts for the average size of the region around the component, thermally affected by the printing process (e.g. with presence of strong thermal gradients), as shown in Figure 6.3

Introducing an equivalent boundary condition for heat transfer through the powder-bed simplifies the physics of the problem, but has obvious consequences in the error of the predictions. On the one hand, it leads to a reduced computational model with less thermophysical properties of the metal powder to be determined. On the other hand,

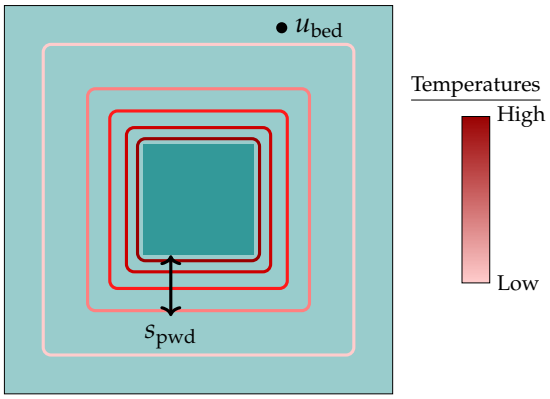



Figure 6.3: $u_{\text{bed}}$ is the average temperature of the powder far from the HAZ, where the thermal field is not much affected by the temperature gradient originated by the printing process, and $s_{\text{pwd}}$ is the average size of the process affected zone.

though suitable for sensitivity analysis and optimisation of the process parameters, this approach is not recommended for applications with strict accuracy requirements.

**Heat convection through the surrounding environment.**

The heat loss by convection through the surrounding environment $q_{\text{conv}}$ can also be expressed by Newton's law as

$$q_{\text{conv}}(u) = h_{\text{conv}}(u - u_{\text{air}}), \quad \text{on } \partial\Omega_{\text{air}},$$

where $h_{\text{conv}}(u)$ denotes the HTC by convection through the surrounding environment and $u_{\text{air}}$ is the temperature of the gas inside the machine chamber. $u_{\text{air}}$ can be assumed constant if the gas temperature is controlled or the component is very small compared to the size of the chamber.

**Heat radiation.**

Radiation is an important heat loss mechanism at the HAZ, due to the high-temperature field induced by the heat source. The radiation heat flux $q_{\text{rad}}$ can be calculated using Stefan-Boltzmann's law:

$$q_{\text{rad}}(u) = \sigma\epsilon\left(u^4 - {u_{\text{air}}}^4\right), \quad \text{on } \partial\Omega_{\text{air}}.$$

Here, $\sigma$ is the Stefan-Boltzmann constant and $\epsilon$ is the emissivity of the radiating surface, a measure of the efficiency of the body as a radiation emitter. The contribution of heat radiation can also be expressed as

$$q_{\text{rad}}(u) = h_{\text{rad}}(u - u_{\text{air}}), \quad \text{on } \partial\Omega_{\text{air}},$$

where

$$h_{\text{rad}}(u) = \sigma\epsilon \left(u^3 + u^2 u_{\text{air}} + u u_{\text{air}}^2 + u_{\text{air}}^3\right).$$

Heat is lost through the environment by a combination of convection and radiation. In practice, it is difficult to discriminate the effects of both heat transfer modes. For this reason, the numerical model assumes a combined heat transfer law, accounting for both heat convection and radiation:

$$q_{\text{air}}(u) = h_{\text{air}}(u - u_{\text{air}}), \quad \text{on } \partial\Omega_{\text{air}}.$$

In this case, $q_{\text{air}}$ represents the heat flux due to the simultaneous convection and radiation mechanisms, and $h_{\text{air}}$ is the corresponding equivalent HTC.

## 6.3 FE modelling of the AM process

Metal deposition is modelled by moving the heat source along a predefined scanning pattern. Hence, the geometry of the component grows in time according to the sintering process that transforms metal powder into a new solid layer.

The numerical simulation of this process requires an ad-hoc procedure to apply the volumetric heat source $r$ in space and time to the elements affected by the moving energy input, as well as to include into the computational domain those elements forming a new layer of material. This procedure is referred to as the *FE activation technique*.

The activation strategy used in this work is the *born-dead-elements technique* [52, 54]. It classifies the elements of the finite-element mesh into: *active*, *activated*, and *inactive* elements:

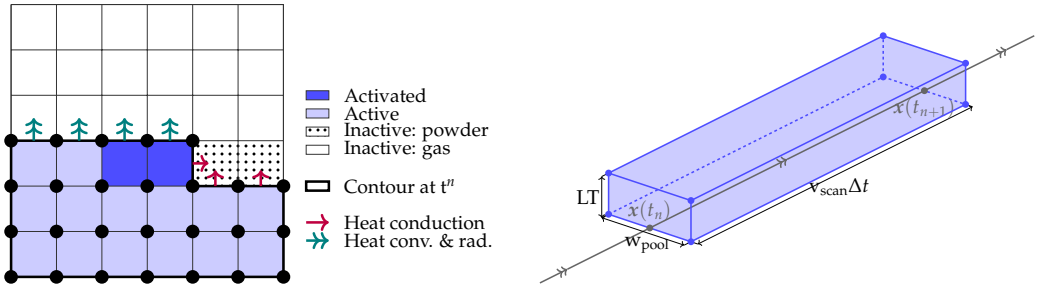
- *Active* elements are those elements representing the building platform, as well as the ones already activated by the metal deposition process.
- *Activated* elements are the ones affected by the energy input during the current time step and inactive previously to this moment.
- *Inactive* elements have not yet been included into the (active) computational domain.

According to this, the computational domain is defined by the set of active and activated elements, as seen in Figure 6.4(a). To update the computational domain from one time step to the next one, a search algorithm is used to identify the elements that are affected by the heat source during the current time increment.

### 6.3.1 Space and time discretisation of the heat source

The representation of the heat source within the FE framework is detailed in [54]. The melt pool moves from a given position $\boldsymbol{x}^n$ to the following position $\boldsymbol{x}^{n+1}$ in the interval $\Delta t = t^{n+1} - t^n$ according to the predefined scanning sequence.

The total volume affected by the power input $V_{pool}^{\Delta t}$ in this interval, referred to as the HAV, can be represented as a cuboid of length $v_{scan}\Delta t$, being $v_{scan}$ the scanning speed, and cross-section given by the average laser spot size $w_{pool}$, and the average layer thickness LT, as shown in Figure 6.4(b). The heat source term $r$ in Equation (6.1) is only applied to the elements inside the HAV.




(a) As shown with this 2D FE cartesian grid, at each time step of the AM simulation, the computational domain is split into active, activated and inactive elements.

(b) Heat affected volume in powder-bed technologies. For a given time step, the laser source moves from position $x^n$ to $x^{n+1}$, according to the pre-defined scanning path.

Figure 6.4: FE activation technology: element classification and HAV.

At each time step, these elements are found with an octree-based search algorithm. If an inactive element belongs to the HAV, then it is activated. The volume of the melt-pool is computed as

$$V_{pool}^{\Delta t,h} = \sum_{e \in hav} V^e$$

where $V^e$ is the volume of element $e$ and the (average) density distribution from the heat source is computed as

$$r = \frac{\eta W}{V_{pool}^{\Delta t,h}},$$

where W is the laser power [W] (watt) and $\eta$ is the heat absorption coefficient, a measure of the laser efficiency. This power redistribution preserves the total energy input, regardless of the FE mesh employed.

The same care devoted to estimate the energy delivered by the laser beam must be placed to compute the heat dissipated through the boundaries of the computational domain. For this reason, another search procedure is used to update the contour surface at each time step of the simulation, in order to update the current boundary surfaces subject to heat radiation and convection ($\Omega_{air}$) and heat conduction through the powder bed ($\Omega_{bed}$).

One of the added features of this FE activation technique is the possibility of specifying the scanning path using the same input data as for the process machine, for instance, with a Common Layer Interface (CLI) file format [54, 182]. A CLI file describes the movement of the laser in the plane of each layer with a complex sequence

of polylines, to define the (smooth) boundary of the component, and hatch patterns, to fill the inner section.

This is a great advantage because it simplifies the end-user work, when integrating the machine directives with the software interface. However, the scanning path only defines the sequence of points along which the power input moves, as well as the reference plane where the laser beam is focused. The scanning path does not contain any information regarding the velocities of the laser, the size of the melt pool, the spot-size of the laser or the thickness of the deposited layer. These values must be separately specified by the end-user.

### 6.3.2 Definition of scanning strategy

As seen in Figure 6.4(b), during a time increment the laser moves $\Delta x = |\boldsymbol{x}^{n+1} - \boldsymbol{x}^{n}| = \mathrm{v}_{\mathrm{scan}}\Delta t$ along the scanning path. From the end-user point of view, it is more convenient to prescribe $\Delta x$, instead of $\Delta t$, and let the software compute the time discretization as $\Delta t = \mathrm{v}_{\mathrm{scan}}/\Delta x$. In this manner, different approximations of the scanning path, i.e. *scanning strategies*, can be defined according to the accuracy requirements.

For instance, taking $\Delta x \approx h$, where $h$ is the element size, leads to a high-fidelity representation of the scanning path, an element-by-element activation at the cost of a high number of time steps. Alternatively, the simulation can be accelerated by defining $\Delta x$ as the length of one hatch, several hatches or even a whole layer. As a counterpart, this strategy only recovers average temperature fields, being not able to capture the local thermal history [54].

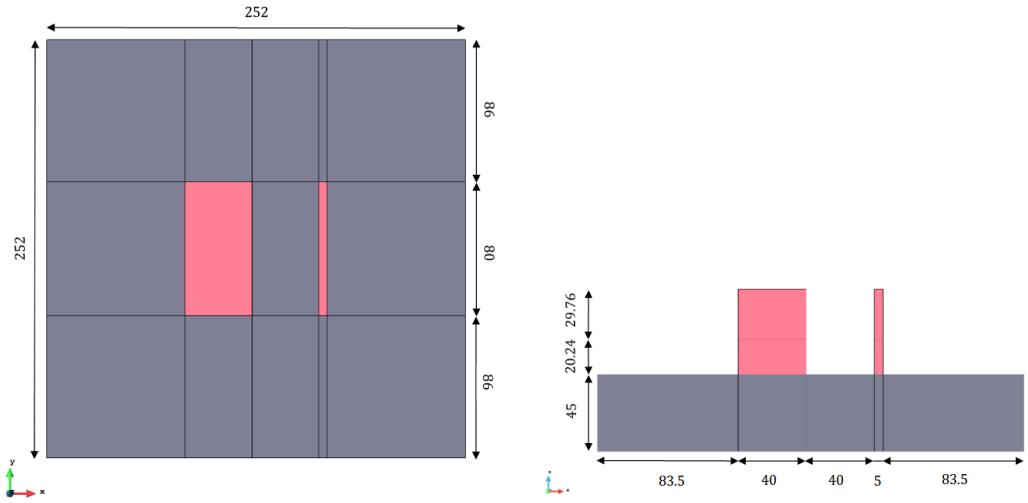
The choice of the scanning strategy depends on the target AM simulation. A high-fidelity approach is affordable when simulating wire-feeding processes, where the layer thickness is around 1 [mm]. However, in powder-bed technologies, the layer thickness reduces to 30-60 [µm]. As a result, thousands of layers of material must be added to build an industrial part and the high-fidelity strategy results in unreasonable computational times. In this case, hatch-by-hatch or layer-by-layer depositions should be considered.

## 6.4 Experimental campaign

An experimental campaign is carried out at the Monash Centre for Additive Manufacturing (MCAM) in Melbourne, Australia, with the purpose of calibrating the thermal analysis FE framework described in Section 6.2 for powder-bed methods.

The printing system employed for the experiments is the EOSINT M280 from Electro Optical Systems (EOS) GmbH. It makes use of an Yb-fibre laser with variable beam width and power up to 400 [W]. The printing process is carried out in a closed $250 \times 250 \times 325$ [mm$^3$] chamber in a controlled argon atmosphere to prevent oxidation of the part. The argon flow is kept laminar.

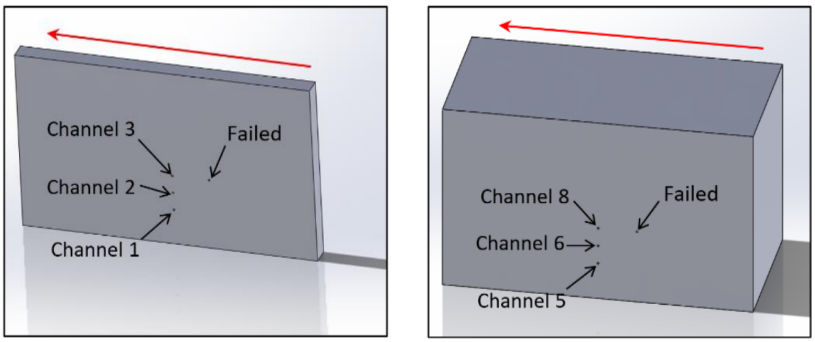
Figure 6.5 shows the samples geometry for the numerical calibration. These consist of two specimens printed simultaneously: a thin wall measuring 5x80x50 [mm$^3$] and a thicker wall measuring 40x80x50 [mm$^3$]. The two walls are separated by a distance of 40 [mm]. The base plate has dimensions of 252x252x45 [mm$^3$]. Ti64 powder is used for the printing operation.




(a) Plane XY view (b) Plane XZ view

Figure 6.5: Base plate and printed walls

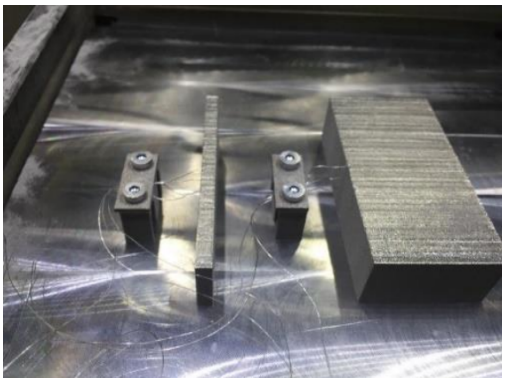
The thermocouples for the temperature measurements are inserted into holes at different locations of the two specimens. For this purpose, the printing job has to be interrupted after an initial deposition of 20.24 [mm] high and the powder bed has to be removed. Four thermocouples are installed in each sample: the first three separated by 5 [mm] in the vertical direction and the fourth one displaced 10 [mm] horizontally from the top one, as shown in Figure 6.6. Next, the powder bed is restored, and the scanning sequence resumed, as described in Figure 6.7. As thermocouples are not welded, their measurements can be affected by air trapped in their holes.



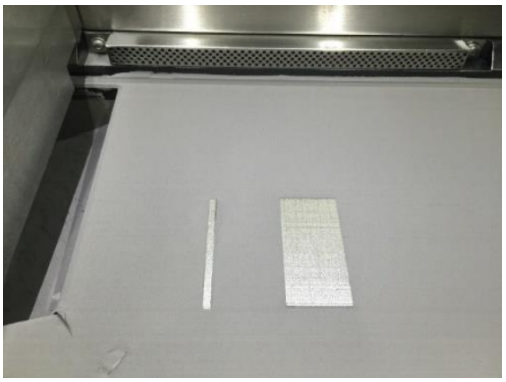

(a) Thermocouples at thin wall (b) Thermocouples at thick wall

Figure 6.6: Location of thermocouple channels

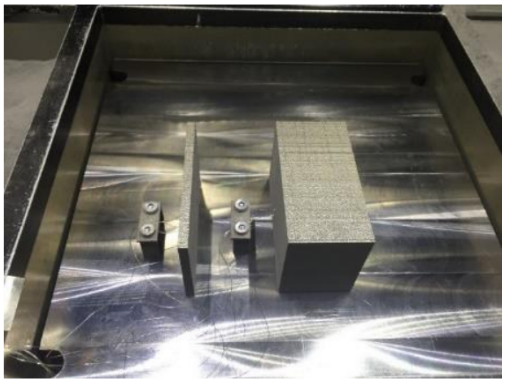
K-type thermocouples and a Graphtec GL-900 8 high-speed data-logger are used for the data gathering. Temperature data could be measured only from six channels,



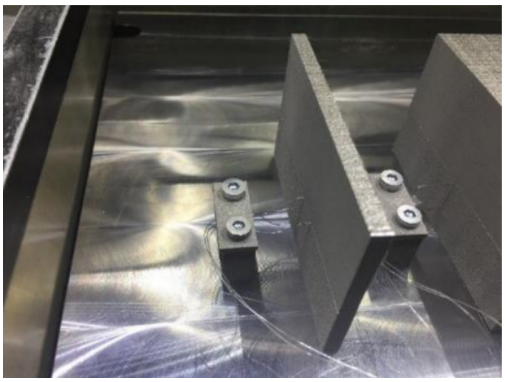
(a) The printing job is interrupted at 20.24 [mm] height, the powder bed is removed, and four thermocouples are inserted into holes of each sample.



(b) The powder bed is restored and the printing job was resumed.



(c) After finishing the printing job, the powder bed is removed from the building platform.



(d) A closer picture shows the position of the thermocouples throughout the printing process.

Figure 6.7: Steps to carry out the temperature measurements

because the fourth channel in both walls was broken during the setup operations. The sampling rate of the data logger is 1 [ms] and the time constant of the thermocouples is 7 [ms].

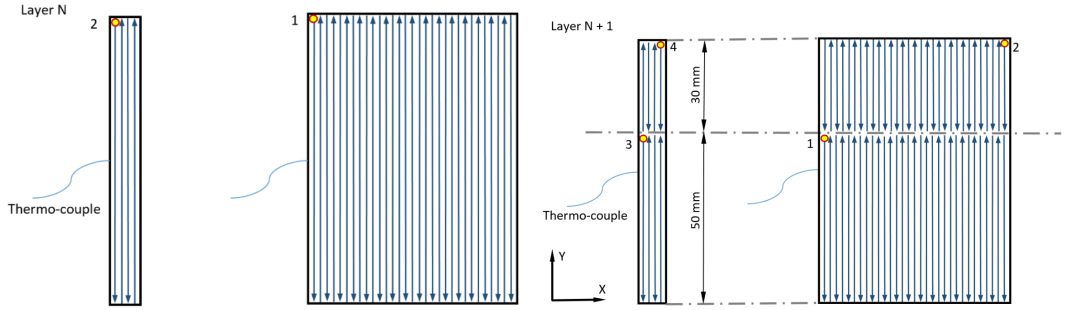
Table 6.1 shows the process parameters used for the printing process. As observed, the layer thickness is set to 30 [µm], meaning that 992 layers are deposited in about 33,500 [s] (a little bit more than 9 [h]) to build the samples.

| | | |
|---|---|---|
| Power input | 280 | [W] |
| Scanning speed | 1200 | [mm/s] |
| Layer thickness | 30 | [µm] |
| Hatch distance | 140 | [µm] |
| Beam offset | 15 | [µm] |
| Recoat time | 9.36 | [s] |
| Relocation time | 0.03 | [s] |

Table 6.1: Process parameters adopted by the EOS Machine

A unidirectional scanning strategy is applied along the longitudinal direction of

both samples. In Figure 6.8 the scanning sequence used for the printing process is described. The scanning path alternates between odd and even layers. Note that the number of hatches drawn does not correspond to the actual number of hatches, which is notably higher according to the power beam size.




(a) Pattern for odd layers. Starting points are indicated with circles.

(b) Pattern for even layers. Starting points are indicated with circles.

Figure 6.8: Scanning pattern

The printed samples are made of Ti6Al4V Titanium alloy. The temperature dependent properties of the bulk material, covering the range from room temperature to fusion temperature, are depicted in Figure 6.9. The base plate is made of CP Ti, a material with similar thermal properties as those of Ti64.

Complementary experiments were done to estimate the density of the porous bed, as it is formed layer-by-layer during the printing process. According to these measurements, the packing density is about 2,405 [kg/m3] at room temperature. Thus, the relative density of Ti64 powder is around 54%, with respect to the density of the bulk material at room temperature.

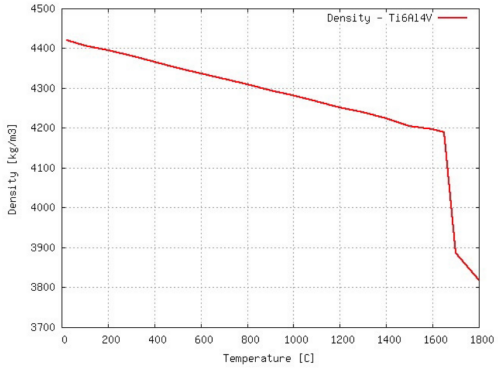
## 6.5 Numerical results and discussion

### 6.5.1 Initial calibration of the model

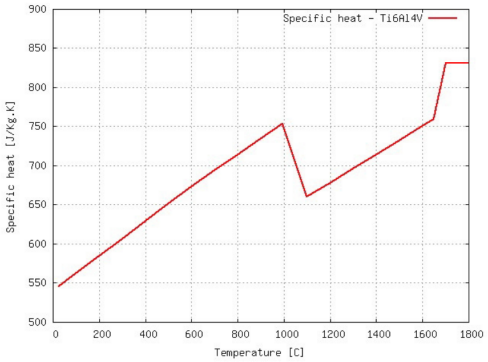
The in-house research software COupled MEchanical and Thermal (COMET) [49] is suitably enhanced to provide a FE framework for the heat transfer analysis of AM by powder-bed technologies. The model is calibrated against the experimental data obtained at the MCAM research centre.

The numerical model selected for the calibration procedure should reproduce as close as possible the physical process. Likewise, the size of the simulation should be chosen to enable a full sensitivity analysis in reasonable computational times.

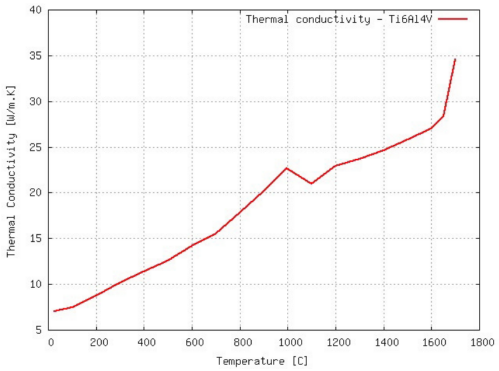
For the most accurate simulation of the metal deposition, the FE mesh must conform to the printed layers, the mesh size must be smaller than the laser beam spot size, and the scanning path must be tracked element by element. However, meeting these requirements in this experiment is extremely difficult from the computational point of view. For instance, assuming a uniform mesh with element size $50 \times 50 \times 30$




(a) Density




(b) Specific heat




(c) Thermal conductivity

Figure 6.9: Ti6Al4V Titanium alloy thermal bulk material properties

[µm], a single layer of the thick wall is composed of 1,280,000 elements to be printed in 1,280,000 time steps.

For these reasons, in a first stage of the calibration process, the powder bed has been excluded from the analysis and the scanning path has been approximated to obtain a computationally affordable numerical model. On the one hand, these two assumptions have an impact on the accuracy, as discussed in Section 6.2.2 and Section 6.1, but it was the only possibility to perform the sensitivity analysis to calibrate the numerical model for the whole build process. On the other hand, the experimental measurements are perturbed by the air trapped in the thermocouple holes and the sampling rate of the

data logger, which delays the thermocouple response when registering peak temperature values.

This given, the FE discretisation consists of a structured mesh of 150,048 hexahedral elements and 194,150 nodes. Figure 6.10 shows the numerical model considered for the numerical simulation of the printing process. This model includes the building platform to account for the heat dissipation by conduction, but excludes the powder bed. Hence, the heat loss by conduction through the powder bed is modelled through the equivalent heat flux described in Section 6.2.

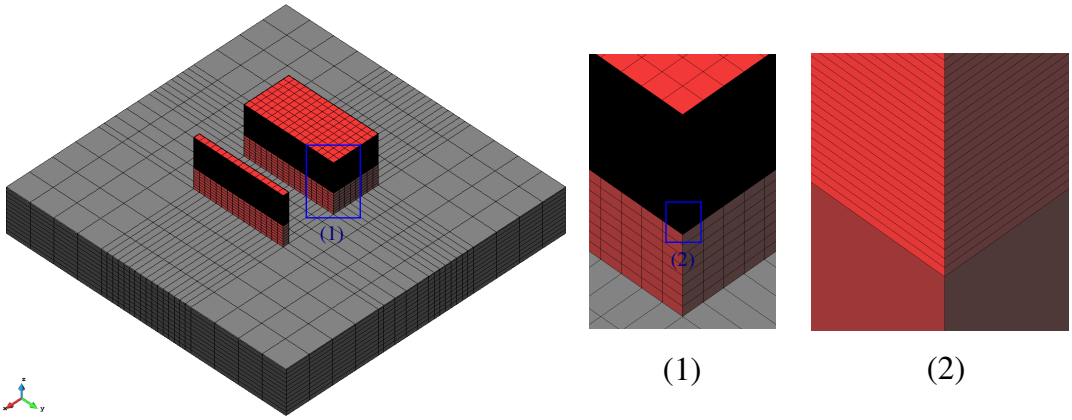



Figure 6.10: The FE mesh conforms to the layers, as seen in the close-up plots.

To further decrease the computational cost, the mesh size is adapted according to the different regions in the model. Small $5 \times 5 \times 0.03$ mm elements are specified at the deposition regions, while a larger mesh size is specified below the deposition regions and the base plate. As a result of approximating the scanning path and using a uniform heat source distribution, the mesh size no longer needs to be smaller than the laser spot size to obtain an average thermal response with a relative error bounded by 10%. It suffices that it conforms to the hatch width (5 mm).

The simulation starts when the printing job is resumed after placing the thermocouples and continues throughout the deposition of the remaining 992 layers, up to the cooling of the whole ensemble. Each new layer is printed in four steps:

1. The scanning sequence corresponding to a new layer of the thin sample is performed;

2. The laser moves from the thin sample to the thick sample (relocation time);

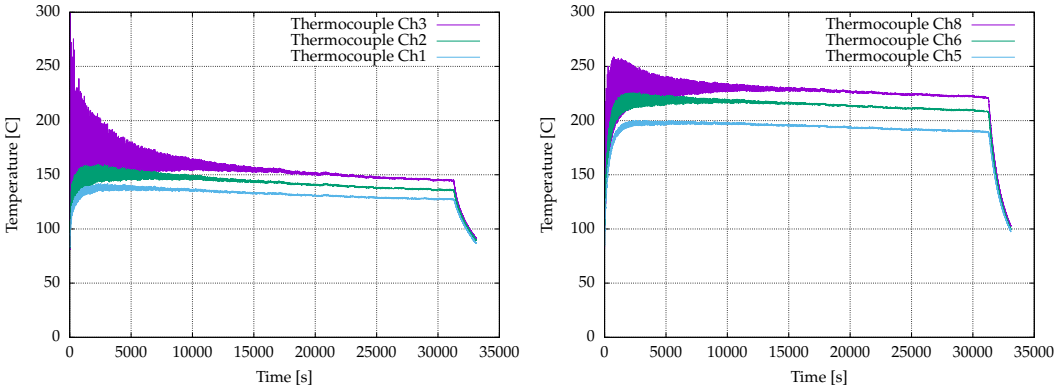
3. The scanning sequence corresponding to a new layer of the thick sample is carried out;

4. The platform is lowered and a new powder layer is spread. During the recoat time, the heat transfer analysis to account for the cooling process of the samples is performed.

The building platform is kept at 100 [°C] during the whole printing process. The average temperature of the powder, as well as the temperature inside the chamber, used for the calibration of the heat transfer coefficients (heat loss by convection and radiation), are set to constant values of 83 [°C] and 35 [°C], respectively, according to on-site measurements.

The HTC for the heat convection model is calibrated to 1.0 [W/m$^2$K]. The powder conductivity required to deal with the heat dissipation through the powder bed is obtained taking $k_{\text{Ti64}} = 7$ [W/mK] and $k_{\text{argon}} = 0.016$ [W/mK] at 20 [°C]. Hence, according to Equation (6.2), $k_{\text{pwd}}$ results in about 0.14 [W/mK]. Repeating the evaluation at 800 [°C], the resulting HTC is $k_{\text{pwd}} = 0.60$ [W/mK]. Hence, according to the average temperature of the powder, the powder conductivity used for the simulations is $k_{\text{pwd}} = 0.20$ [W/mK]. Furthermore, $s_{\text{pwd}} = 40$ [mm], leading to an equivalent HTC by conduction of $h_{\text{bed}} = 5.0$ [W/m$^2$K].

Figure 6.11 describes the experimental data gathered at the six working thermocouples at both samples. The two samples have very different thermal modulus, that is, the ratio between the volume and the area of the external surfaces to dissipate the heat in the surrounding environment. This explains why the thick sample, with higher thermal modulus, presents higher temperatures than the thinner sample.




(a) Measured data at thin sample

(b) Measured data at thick sample

Figure 6.11: Experimental data gathered for both sample locations

Regarding the evolution in time, it starts with a temperature build-up that stabilises at about the hundredth layer. Afterwards, the temperature at the thermocouples decreases slowly until the end of the printing stage, when it drops until cooling down to the initial temperature. This quasi steady-state regime in the middle of the process is a result of the thermocouples being far from the HAZ, i.e. thermal gradients in the neighbourhood are small, and it can be clearly identified here due to the long duration of the experiment, as opposed to the short experiments predominant in the literature.

Apart from that, it can be noticed how the temperature measurements recorded at the thermocouples CH3 and CH8 present extremely high oscillations at the beginning of the process. This is due to the heat radiation during the deposition of the first layers,

| | | |
|---|---|---|
| Power input | 280 | [W] |
| Power absorption | 45 | % |
| Scanning speed | 2.10 | [mm/s] |
| Back speed | 12.40 | [mm/s] |
| Layer thickness | 30 | [µm] |
| Hatch width | 5 | [mm] |

Table 6.2: Process parameters used in the hatch-by-hatch strategy

after resuming the printing process. For this reason, only the data from thermocouples CH1 and CH2 (thinner sample) and thermocouples CH5 and CH6 (thicker sample) are accounted for the calibration process.

The sensitivity analysis has been performed using the following simplified scanning strategy, referred to as *hatch-by-hatch*:

1. One single hatch is used for the layer sintering of the thin sample in a single time step. The total amount of energy input used is uniformly spread at once.

2. Next, the laser source is moved from the thin to the thick sample ready for the following sintering process. The model is allowed to cool down during this relocation time.

3. Later, the layer sintering of the thick sample is performed using 8 hatches, 5 [mm] wide, in 8 time steps.

4. During the final recoat and relocation phases, a new layer of powder is spread and the corresponding cooling process is accounted for.

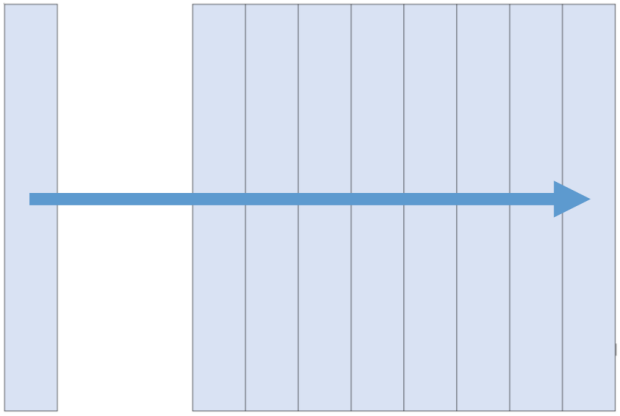
Steps 1 to 4 are repeated until completion of the building process. Finally, the cooling process to the room temperature is analysed.

The hatching pattern of this simulation strategy is described in Figure 6.12(a). The process parameters used for the numerical simulation are gathered in Table 6.2. The scanning speed (printing) and the back speed (relocation, recoat) have been adapted to match the values in Table 6.1.

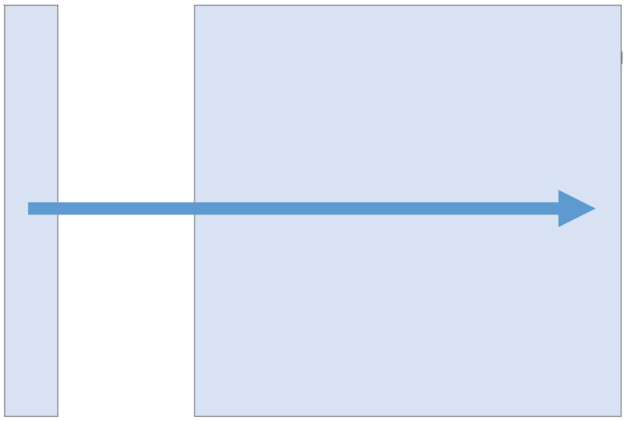
Thermocouples CH1 and CH2 have been used as target to get the most suitable simulation parameters to catch the experimental evidence. In particular, they have been used to calibrate both the power absorption coefficient (responsible of the power input into the system) and the equivalent HTC between the samples and the powder bed (controlling the heat loss through the external surfaces of the components).

Figure 6.13 compares the numerical results with the experimental measurements at CH1 and CH2. A very good agreement can be observed in the thermal response. The only mismatch is detected during the final cooling process, while cooling down to the room temperature.

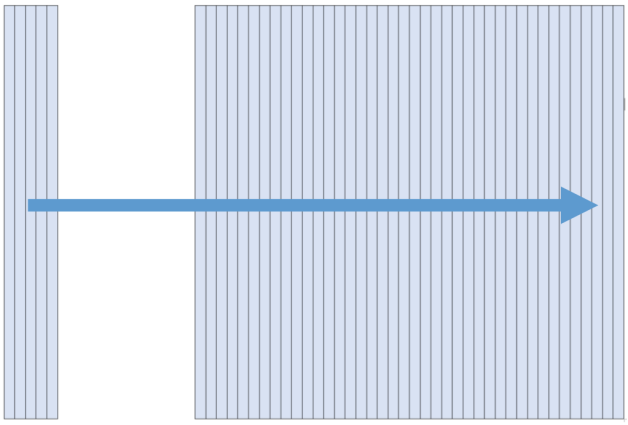
A similar but slightly lower accuracy of the numerical model can be observed in Figure 6.14 when comparing the results obtained for the thick sample. This can be



(a) Hatch-by-hatch: Each layer is printed using 1+8 hatches.



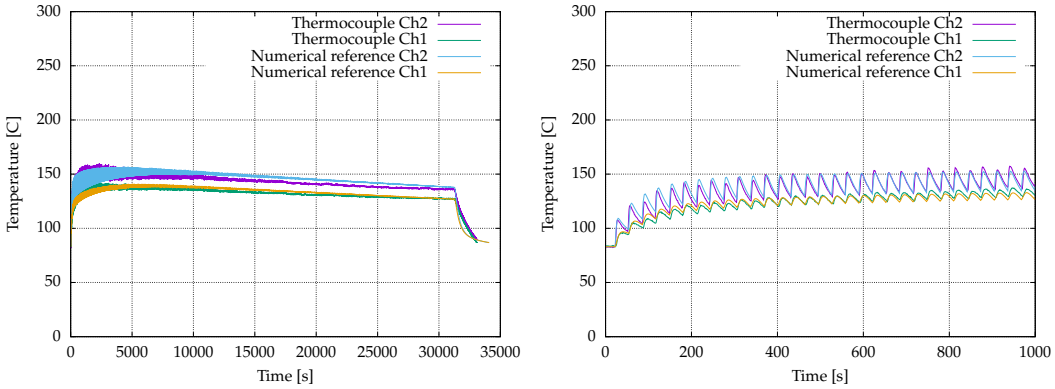
(b) Layer-by-layer: A new layer for the thin and the thick sample samples is simultaneously deposited in a single time step.



(c) High-fidelity: the hatch width is reduced to 1 [mm].

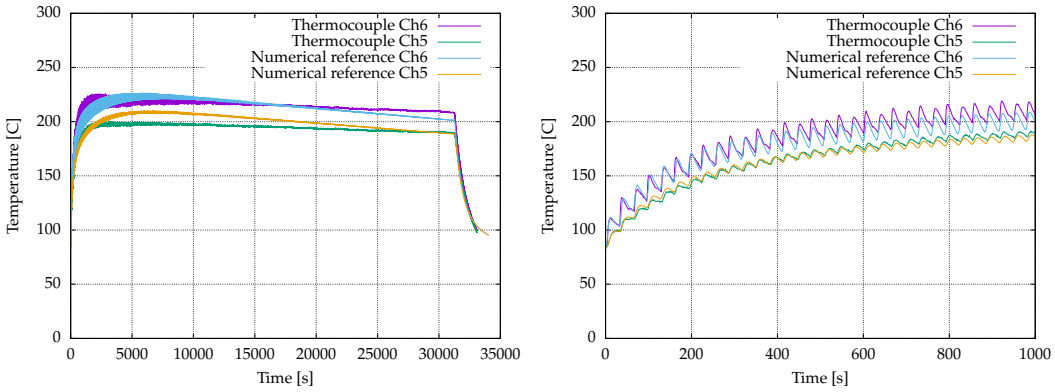
Figure 6.12: Different scanning strategies were considered in the numerical analysis.

attributed to the approximation of the heat loss model through the powder bed. In particular, assuming a uniform value for the powder temperature is not fully reliable (e.g. between the two samples there should be a higher value than the average value used for the entire model). Nevertheless, this mismatch is less than 10%.




(a) The entire sintering process (b) The first phase after resuming the build job

Figure 6.13: Hatch-by-hatch (reference): Numerical results at the thin sample




(a) The entire sintering process (b) The first phase after resuming the build job

Figure 6.14: Hatch-by-hatch (reference): Numerical results at the thick sample

### 6.5.2 Numerical model assessment

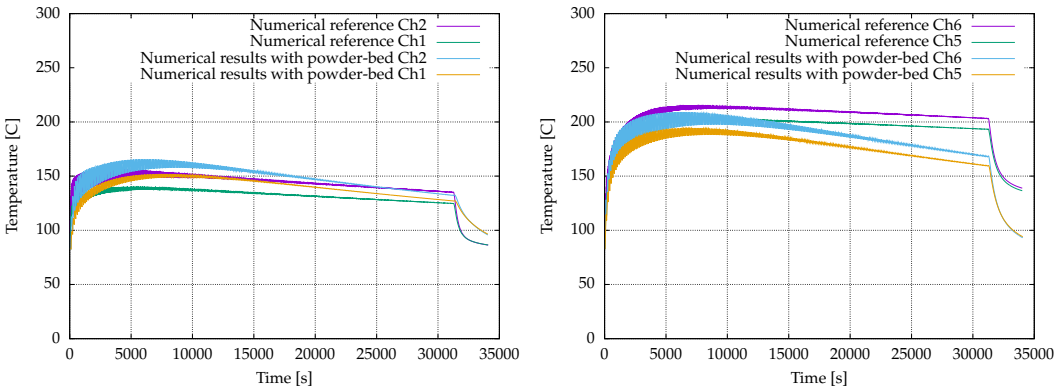
In a next stage, a set of numerical tests is carried out to test the previous hypotheses and investigate different numerical strategies to simulate the AM process by powder-bed methods. The objective is to find the best compromise between computational cost and accuracy.

First, the powder bed is added to the model. In this case, the size of the FE discretisation is much bigger, including 594,368 hexahedral elements and 649,230 nodes. This model is about six times bigger than the previous one (without powder bed) and the simulation time is almost four times longer (from 1 day to about 4 days). The thermophysical properties of the Titanium powder are obtained as defined in Section 6.2, with the values of porosity and thermal conductivity listed in Table 6.3.

| Porosity | 46 | % |
|---|---|---|
| Thermal conductivity | 0.20 | $[W/mK]$ |

Table 6.3: Porosity and thermal conductivity of the Titanium powder

Figure 6.15 compares the numerical results with and without including the powder bed in the simulation. The results are not as good as in the previous case, even if the model is more realistic. This is due to several reasons: (1) the results strongly depend on the thermophysical properties used to characterise the metal powder: density, specific heat and conductivity, all of them should be defined in terms of the actual temperature field. The available limited characterisation of the powder made difficult their calibration; (2) the much higher computational cost made extremely slow the sensitivity analysis.




(a) At the thin sample

(b) At the thick sample

Figure 6.15: Numerical results with or without (reference) including the powder bed into the computational domain

Besides, this simulation is useful to examine the values of $s_{\text{pwd}}$ and the average temperature of the powder. Figure 6.16 shows that $s_{\text{pwd}}$ is around 40 [mm] and the powder temperature is 83 [°C]. However, it is also evident that the average temperature of the powder between the two samples should be approximately 135 [°C], instead of 83 [°C].

The next numerical simulations are intended to assess different scanning strategies. First, a *layer-by-layer* building strategy has been selected to reduce as much as possible the simulation time, while providing reasonable accuracy. Hence:

1. One single time step is used to add simultaneously a new layer for both samples. The energy density used is spread homogeneously at once. The sintering time includes the relocation time, that is, the time used by the laser to move from the ending point of the scanning sequence at thin sample to the starting point at the thick sample.

2. Further time steps are performed to account for the cooling process during the recoating and relocation times, when the building platform is lowered and a new layer of powder is spread.

3. The discretization of the powder bed is avoided.

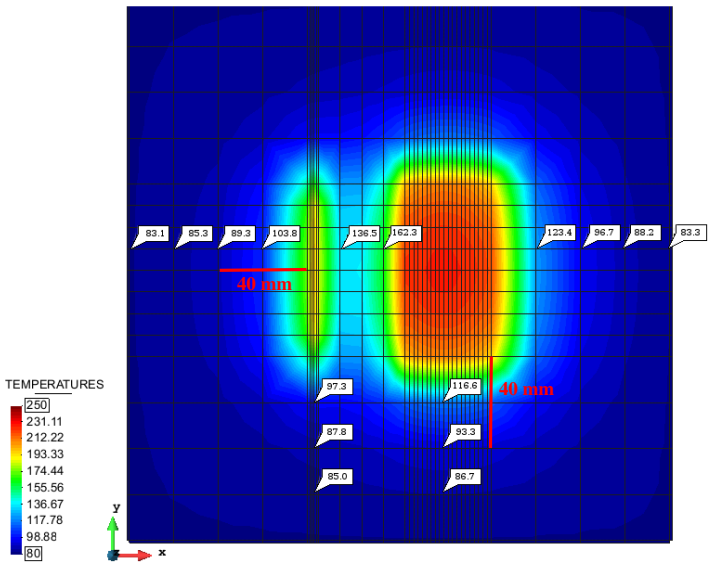



Figure 6.16: Contour plot of temperatures for the simulation including the powder bed. As assumed in the initial calibration stage, $s_{\text{pwd}}$ is around 40 [mm].

Steps 1 and 2 are repeated until completing the building process. Later, the cooling process to the room temperature ends the analysis. The scanning pattern of this simulation strategy is described in Figure 6.12(b). The new values of the scanning speed and the back speed are gathered in Table 6.4.

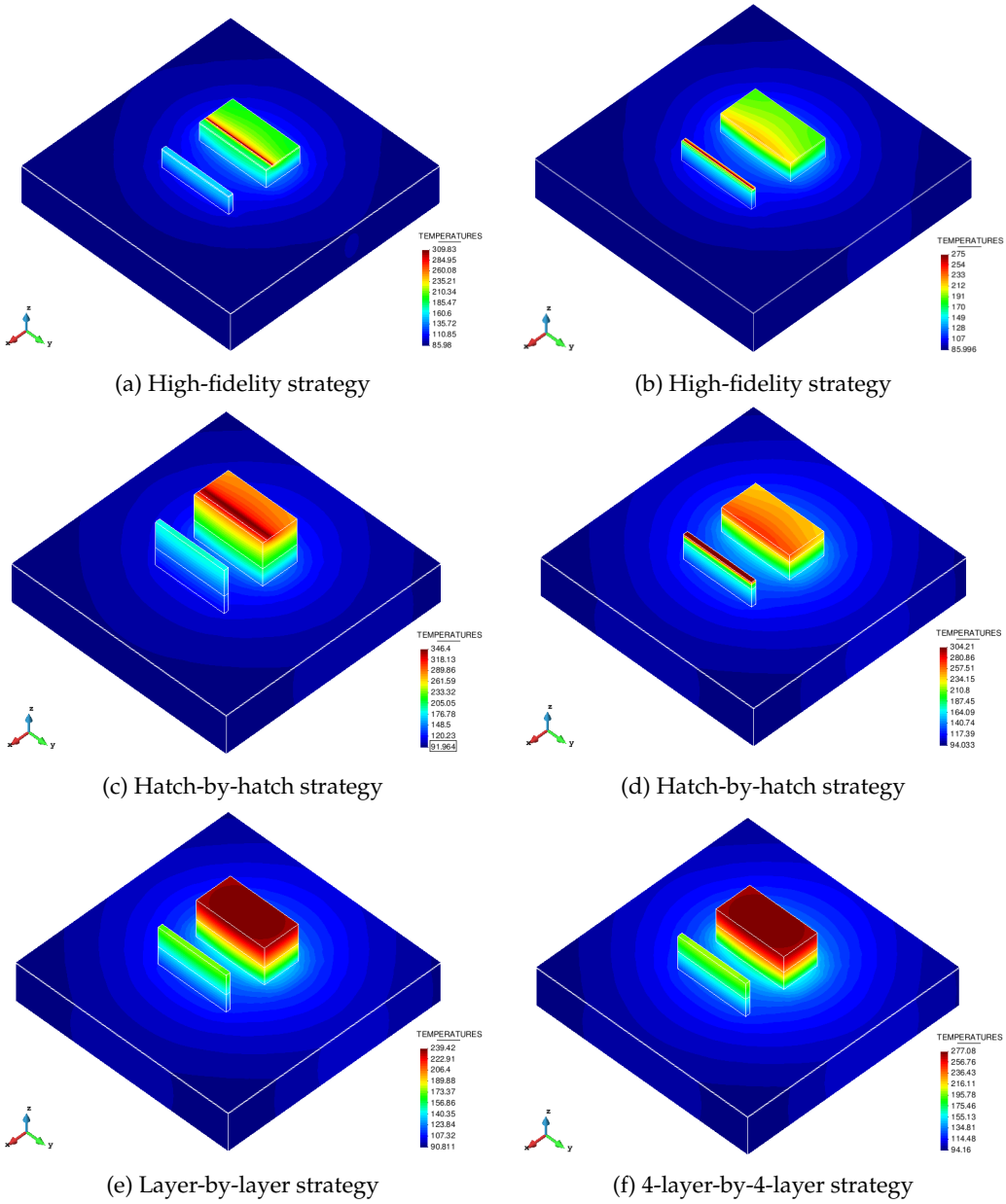
| | | |
|---|---|---|
| Scanning speed | 3.73 | [mm/s] |
| Back speed | 0.69 | [mm/s] |

Table 6.4: New process parameters for the layer-by-layer strategy

Similarly, a *multi-layer-by-multi-layer* simulation strategy can be considered for further reduction of the computational cost. In Figure 6.18, the numerical results with both layer-by-layer and 4-layer-by-4-layer strategies are compared with the hatch-by-hatch result taken as a reference.

Observe that, when using a layer-by-layer strategy, the corresponding energy density is spread uniformly. Hence, the temperature plots must be intended as an average evolution of the temperature field of the layer, with no direct relationship with the measurement locations. Clearly, the CPU-time is notably reduced when adopting the layer-by-layer strategy, as shown in Table 6.5.

Finally, a further scanning strategy has been defined to be closer to the actual hatching sequence of the SLM machine. This new scanning pattern is depicted in Figure 6.12(c) and the resulting simulation strategy is referred to as *high-fidelity*. The process parameters used are the same as those in Table 6.2, except for the hatch width, which has a value of 1 [mm]. Figure 6.19 compares the *high-fidelity* results with the experimental data at thermocouple locations CH1 and CH2.




(a) High-fidelity strategy

(b) High-fidelity strategy

(c) Hatch-by-hatch strategy

(d) Hatch-by-hatch strategy

(e) Layer-by-layer strategy

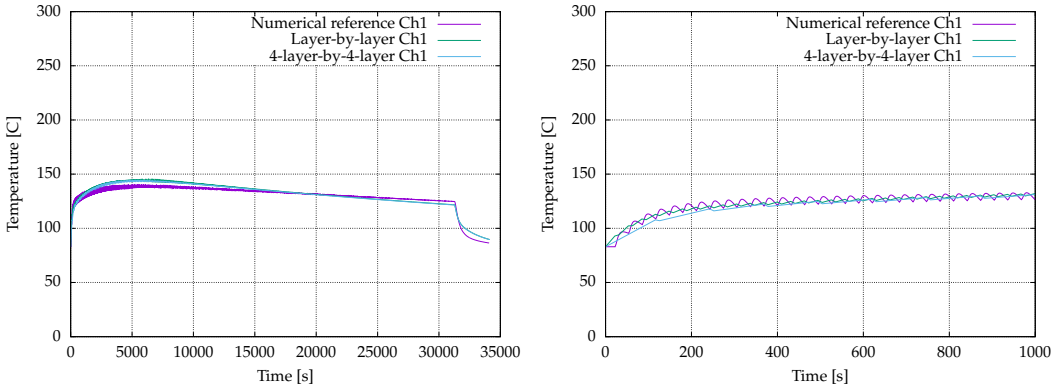
(f) 4-layer-by-4-layer strategy

Figure 6.17: Contour plots of temperatures for the analysed scanning strategies at different time steps. The locality of the temperature distribution decreases as the scanning path simplifies.

| **Strategy** | **CPU-time [h]** | **[%]** |
|---|---|---|
| Reference hatch-by-hatch | 24 | 100 |
| Hatch-by-hatch with powder | 96 | 400 |
| Layer-by-layer | 4 | 17 |
| 4-layer-by-4-layer | 1.33 | 6 |
| High-fidelity | 128 | 533 |

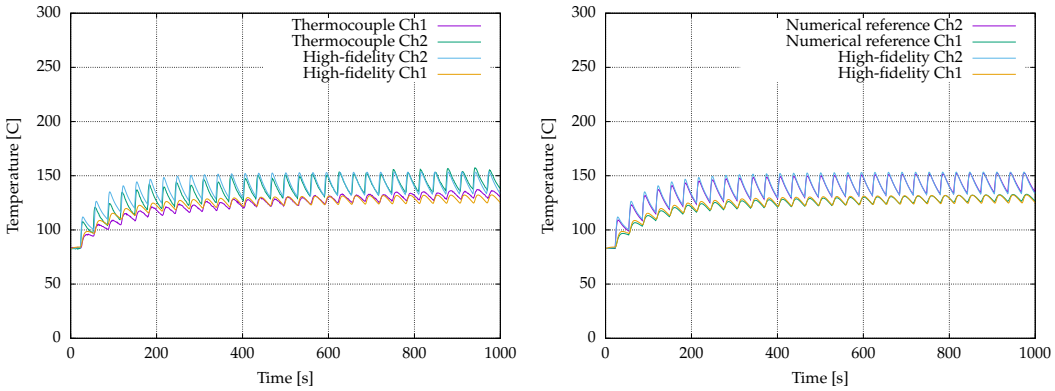
Table 6.5: Simulation CPU times for the different scanning strategies analysed

The similarities between the hatch-by-hatch and the high-fidelity scanning strategies is clear. However, the high-fidelity model required a computational time 5 times




(a) The entire sintering process (b) The first phase after resuming the build job

Figure 6.18: Numerical results obtained by layer-by-layer and 4-layer-by-4-layer strategies




(a) Comparison with experimental data (b) Comparison with numerical reference

Figure 6.19: Numerical results obtained with a high-fidelity simulation

longer, if compared to the hatch-by-hatch strategy: 32 h and 6 h, respectively.

The assessment of the numerical model has given a better insight into the accuracy, computational cost, and applicability of the different model reduction techniques presented in this chapter:

- Excluding the powder-bed from the computational model reduces significantly the size of the spatial discretization and also the uncertainty associated to the approximation of the powder thermophysical properties. Nonetheless, it leads to a less physically representative model with lower accuracy. This modelling strategy should be avoided whenever the local thermal history must be accurately predicted.

- As for the scanning strategies, summarised in Table 6.6, the reduction of CPU-time is accompanied by poorer accuracy. In spite of this, high-fidelity and hatch-by-hatch techniques are already able to match the thermoucouple precision. For

this reason, they are an efficient alternative to assist in many engineering decisions, such as selection of the process parameters, design of the scanning path, orientation of the part, or the number of parts printed in a build.

Apart from that, different domain sizes should be considered for different reduced models. In this sense, the authors recommend to use high-fidelity and hatch-by-hatch strategies from medium build sizes ($> 100$ mm$^3$), layer-by-layer strategies for large builds ($> 10$ cm$^3$ and 100-500 layers), and multi-layered strategies only for very large builds with several components ($> 100$ cm$^3$ and above 500 layers).

| Strategy | Accuracy | CPU-time | Applications | Build size |
|---|---|---|---|---|
| Mesh size ≈ Laser spot size | ●●●●● | ●●●●● | Thermomechanical, mesoscale, and microscale analyses. | Any |
| High-fidelity | ●●●○○ | ●●●●○ | Optimisation of process params., scanning path design. | Medium or large |
| Hatch-by-hatch | ●●●○○ | ●●●○○ | | |
| Layer-by-layer | ●●○○○ | ●●○○○ | Number of parts on single build, location and orientation of parts. | Large |
| n-layer-by-n-layer | ●○○○○ | ●○○○○ | | |

Table 6.6: Comparison of simplified scanning strategies

## 6.6 Conclusions

In this chapter, a FE framework for the numerical simulation of the heat transfer analysis of AM processes by powder-bed technology is detailed. The formulation is supplemented by an apropos FE activation technique to deal with the sintering process which transforms the metal powder into a new solid layer.

The numerical model accounts for the power input and the corresponding power absorption, the temperature dependency of the material properties and the heat dissipation through the boundaries by conduction, convection and radiation.

The experimental calibration of the model is performed by defining a benchmark manufactured using the EOSINT M280 machine available at MCAM laboratories and instrumented with different thermocouples. The scale of the experiment (geometry, number of layers, build time) is close to the current technological limits of the machine, but also a computational challenge, where specifying mesh sizes of the order of the laser spot size leads to extremely large problems.

In a first stage, the powder bed is excluded from the domain of analysis and the scanning sequence is approximated. Both assumptions have an impact on the accuracy of the local complex thermal history, but are necessary to carry out the calibration and sensitivity analysis in reasonable computational times. Even with these reductions, the model is capable of accurately capturing the global thermal response.

Heat dissipation through the powder bed is accounted for with an equivalent boundary condition in terms of Newton's law of cooling. This heat dissipation mechanism and the definition of the power input have been identified as the most sensitive mechanisms to assess the simulation accuracy.

In a next stage, different numerical strategies are analysed to find the best one in terms of computational cost vs simulation accuracy. First, the powder bed is added to the analysis. Afterwards, alternative scanning strategies are investigated by considering: simplified hatch-by-hatch patterns, layer-by-layer and multi-layer-by-multi-layer building sequences.

Excluding the powder bed from the computational domain reduces CPU-time and avoids the characterisation of the material powder. However, it is necessary to define the equivalent HTC between the samples and the powder, as well as the average temperature of the powder far from the HAZ. The powder bed can be split into different regions with different average temperatures. This is relevant when more components are printed on the same building platform at the same time.

Regarding the scanning strategies, a layer-by-layer or a multi-layered approach significantly reduces the computational effort. However, this modelling strategy is only able to capture an average evolution of the temperature field during the manufacturing process. To capture the local thermal history at the thermocouples, the high-fidelity approach is preferred because the energy distribution according to the actual scanning sequence is retained. Finally, simplified hatch-by-hatch patterns strike a good balance between computational effort and accuracy, turning them into a competitive alternative for optimisation of process parameters and process planning.

# Chapter 7

# Model reduction by physics-based localisation in AM by powder-bed fusion

*The contents of this chapter correspond to the research publication*

[144] EN, M. CHIUMENTI, M. CERVERA, E. SALSI, G. PISCOPO, S. BADIA, A. F. MARTÍN, Z. CHEN, C. LEE AND C. DAVIES, *Numerical modelling of heat transfer and experimental validation in Powder-Bed Fusion with the Virtual Domain Approximation*, Finite Elements in Analysis and Design, 168 (2020), p. 103343.

Among metal additive manufacturing technologies, powder-bed fusion features very thin layers and rapid solidification rates, leading to long build jobs and a highly localized process. Many efforts are being devoted to accelerate simulation times for practical industrial applications. The new approach suggested here, the virtual domain approximation, is a physics-based rationale for spatial reduction of the domain in the thermal finite-element analysis at the part scale. Computational experiments address, among others, validation against a large physical experiment of 17.5 [$cm^3$] of deposited volume in 647 layers. For fast and automatic parameter estimation at such level of complexity, a high-performance computing framework is employed. It couples `FEMPAR-AM`, a specialized parallel finite-element software, with `Dakota`, for the parametric exploration. Compared to previous state-of-the-art, this formulation provides higher accuracy at the same computational cost. This sets the path to a fully virtualized model, considering an upwards-moving domain covering the last printed layers.

## 7.1 Introduction

AM or 3D Printing is emerging as a prominent manufacturing technology [189] in many industrial sectors, such as the aerospace, defence, dental or biomedical. These sectors are on the lookout to find the way to exploit its many potentials, including vast geometrical freedom in design, access to new materials with enhanced properties or reduced time-to-market. However, this growth cannot be long-term sustained without the support from predictive computer simulation tools. Only them provide the

appropriate means to jump the hurdle of slow and costly trial-and-error physical experimentation, in product design and qualification, and to improve the understanding of the process-structure-property-performance link.

This work concerns the numerical simulation *at the part scale* of metal AM processes by PBF technologies, such as DMLS, described in Figure 6.1, SLM, or EBM. Compared to other metal technologies, powder-bed methods feature the thinnest layer thicknesses, from 60 to 20 microns (or even below). As a result, building industrial components usually requires depositing thousands of layers; thus, from the modelling viewpoint, computational efficiency should be at the forefront. Besides, they are also characterized by having fast solidification rates. This means that high heat fluxes concentrate in the last printed layers; away from that region, the thermal distribution is much smoother and less physically relevant.

Many researchers have used the FE method to investigate metal AM processes, often aided by their knowledge of modelling other well-known processes, such as casting or welding [48, 122]. At the part scale, early FE models proved their applicability to many engineering problems, e.g. selection of process parameters [173], design of scanning path [150] or evaluation of distortions and residual stresses [60, 126, 154]. However, numerical tests were often limited to short single-part builds and small deposition volumes [109, 159].

More recently, the attention has turned to the design of strategies to accelerate simulations to tackle longer processes, multiple-part builds, higher deposition volumes and, eventually, deal with industrial-scale scenarios in reasonable simulation times. Some authors have attempted to exploit adaptive mesh refinement [69, 108, 151] and/or parallelisation [143], while most of them consider surrogate models that are inevitably accompanied by some sacrifice of accuracy or physical representativeness.

Typical model simplifications [123], alone or combined, consist of time-averaging the history of the process, by lumping welds or layers [97, 125], or reduce the domain of analysis by, e.g. excluding the region of loose powder-bed surrounding the part and/or the base plate. In the case of thermal analyses, heat transfer from the part to the excluded region is then accounted for with an equivalent heat loss boundary condition at the solid-powder interface [55, 116]. However, determination of these boundary conditions has been limited to rather simple approximations and challenged by lack of experimental data, especially concerning heat conduction through the loose powder. In this sense, a path yet to be explored in metal AM is to develop physics-based alternatives, similar to the *virtual mould* approach in the area of casting solidification [61, 94, 141], to replace the heat flow model at regions of less physical interest with a much faster and less memory demanding model.

The purpose of this work is to establish a new *physics-based* rationale for domain reduction in the thermal FE analysis of metal AM processes by powder-bed methods. The new technique, referred to as the virtual domain approximation (VDA) in Section 7.3, approximates the 3D transient heterogeneous heat flow problem (see Section 7.2) at

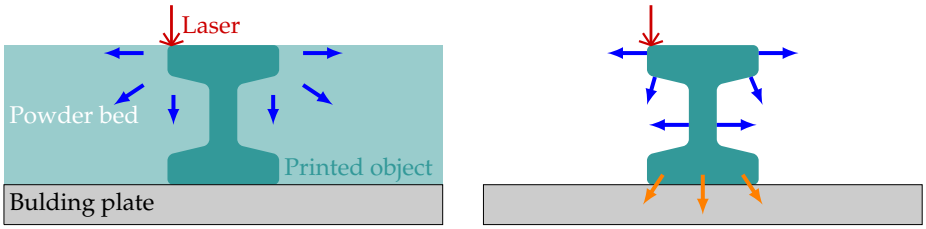
low-relevance subregions (e.g. loose powder, building plate) by a 1D heat conduction problem (see Figure 7.1). This much simpler 1D problem can be analytically solved and reformulated as an equivalent boundary condition for the 3D reduced-domain problem.

Applying this formulation to a simple proof-of-concept example in Section 7.4.1, reduced part-plate (PP) and part-only (P) models are derived and compared with respect to a full powder-bed-part-plate high fidelity (HF) model. In the results of this example, the spatially reduced models are able to accurately approximate the reference response with a computational runtime more than ten times lower than the full model. A second numerical test in Section 7.4.2 addresses experimental validation against physical tests carried out at the MCAM in Melbourne, Australia, using an EOSINT M280 machine and Ti-6Al-4V powder. In this case, an HF model is first calibrated and validated against the experimental data. This step is next repeated for two ancillary PP reduced models. In the first one, the HTC between the part and the powder is assumed constant, as in earlier works [54, 55]. In the second one, the PP adopts the VDA formulation.

Given the scale of the experiment, 17.5 [$cm^3$] of deposited volume in 647 layers and 3.5 [h] of process, the HF model size amounts to 9.8 million unknowns, whereas the PP models to 0.7 million. Sensitivity analysis and parameter estimation at such level of complexity are only practical by means of an advanced computational framework. For this reason, the numerical tests are supported by a high-end parallel computing framework, unprecedented in the simulation of metal AM processes. The framework combines three tools: (1) `FEMPAR-AM` [143], a module of `FEMPAR` [19], a general-purpose object-oriented message-passing/multi-threaded scientific software for the fast solution of multiphysics problems governed by PDEs; (2) `Dakota` [5], for the automatic parametric exploration of the models; and (3) `TITANI` 7.8, a HPC cluster at the Universitat Politècnica de Catalunya (Barcelona, Spain) to support the calculations. Using this innovative methodology for the MCAM experiment, the physics-based VDA-PP model is shown to reproduce the response of the HF model with increased accuracy, with respect to the constant HTC-PP variant. However, the simulation times of both reduced models is practically the same. In other words, the extra cost devoted to evaluate the equivalent boundary condition with the VDA formulation is negligible, in front of the computational cost devoted to the solution of the problem at the reduced domain.

As a result, the VDA formulation offers good compromise between accuracy and efficiency. Indeed, on the one hand, the computational benefit is clear, as the mesh covers a smaller region and the number of degrees of freedom with respect to the complete model is significantly reduced. On the other hand, the impact on accuracy is efficiently controlled, because the neglected physics are taken into account in the evaluation of the equivalent heat loss boundary condition, without affecting the computational cost of the simulation. Even though the VDA formulation does not totally get rid of cumbersome parameter estimations, it can be easily calibrated with respect to an HF model.

This turns it into an appealing alternative for sequential coupling with a part-scale mechanical simulation or optimization problems (e.g. design of minimum-distortion scanning path). More interestingly, by exploiting the locality of the PBF process, the VDA could eventually be useful to reduce the domain down to a few layers, as shown in Figure 7.1(d), while keeping good relation with the thermal response of the HF model.




(a) HF model: Full 3D model accounting for heat conduction through the powder bed region (blue arrows).

(b) PP model: The powder-bed is excluded from the computational domain and heat conduction through the powder bed is accounted for with an equivalent boundary condition (blue arrows).

(c) P model: W.r.t. the PP model, the building plate is also excluded from the computational domain and heat conduction through the plate is accounted for with an equivalent boundary condition (orange arrows).

(d) Local moving model: The analysis only considers the last few layers, where high heat fluxes concentrate, and it is updated to follow the growth of the geometry. When new layers are printed, old ones are left out of the domain of analysis.

Figure 7.1: Application of the VDA technique. The reference HF model includes all the regions involved in the process, but relevant heat transfer in powder-bed methods occurs only at the last printed layers. According to this, the powder-bed, the building plate and previous layers of the solid part can be removed from the computational domain and heat conduction through them can be accounted for with equivalent boundary conditions (blue for powder-bed, orange for plate and violet for part).

## 7.2 Heat transfer analysis in AM

This section describes the transient model for the thermal FE analysis of the printing process *at the part scale*. The contents centre upon (1) an overview of the model variants studied in this work, which depend on the domain of analysis and heat loss model (Section 7.2.1); (2) the governing equation and the determination of the material properties at the powder state (Section 7.2.2); and (3) the treatment of boundary conditions, according to the heat loss model (Section 7.2.3). The reader is referred to [54, 55] for further details on the weak formulation and the FE modelling of the geometry growth during the metal deposition process.

### 7.2.1 Model variants

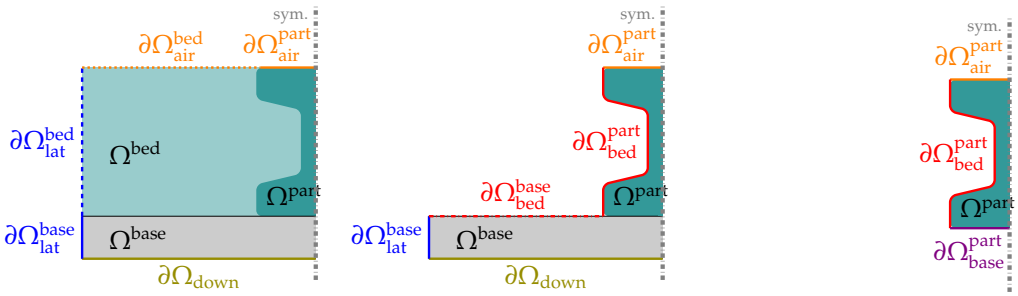
Let $\Omega^{\text{pbf}}$ be an open bounded domain in $\mathbb{R}^3$, representing the system formed by the printed object $\Omega^{\text{part}}$, the building plate $\Omega^{\text{base}}$ and the surrounding powder bed $\Omega^{\text{bed}}$. $\Omega^{\text{pbf}}$ grows in time during the build process. After the printing, it remains fixed, while cooling down to room temperature.

Several variants of the thermal model, represented in Figure 7.2, arise from considering different computational domains $\Omega$:

1. If $\Omega \equiv \Omega^{\text{pbf}} = \Omega^{\text{part}} \cup \Omega^{\text{base}} \cup \Omega^{\text{bed}}$, the model is referred to as **HF**. The contour of $\Omega$ is formed by a region in contact with the air in the chamber $\partial\Omega^{\text{bed}}_{\text{air}} \cup \partial\Omega^{\text{part}}_{\text{air}}$, the lateral wall $\partial\Omega^{\text{bed}}_{\text{lat}} \cup \partial\Omega^{\text{base}}_{\text{lat}}$ and the bottom wall of the plate $\partial\Omega_{\text{down}}$.

2. If the powder bed is excluded from the computational domain, i.e. $\Omega = \Omega^{\text{part}} \cup \Omega^{\text{base}}$, then the so-called **PP** model is obtained. In this case, the solid-powder interface is also in the boundary, i.e. $\partial\Omega = \partial\Omega^{\text{part}}_{\text{air}} \cup \partial\Omega^{\text{part}}_{\text{bed}} \cup \partial\Omega^{\text{base}}_{\text{bed}} \cup \partial\Omega^{\text{base}}_{\text{lat}} \cup \partial\Omega_{\text{down}}$.

3. Finally, taking only $\Omega \equiv \Omega^{\text{part}}$ yields the **P** model, with $\partial\Omega = \partial\Omega^{\text{part}}_{\text{air}} \cup \partial\Omega^{\text{part}}_{\text{bed}} \cup \partial\Omega^{\text{part}}_{\text{base}}$.




(a) HF model. (b) PP model. (c) P model.

Figure 7.2: Close-up of Figure 7.1 to illustrate the thermal model variants studied in this work, along with the boundary conditions.

From the computational viewpoint, HF is the most demanding model, because the part, the plate and the powder bed must be meshed, whereas P is the simplest, because it only needs to mesh the part. On the other hand, HF is the most physically representative model, among those studied here. Therefore, it is established as the *reference* numerical model; i.e. all reduced variants are compared against this one.

As a result of excluding the powder bed (and the building plate), heat conduction through the powder (and the plate) must be accounted for with an equivalent heat loss boundary condition at $\partial\Omega^{\text{part}}_{\text{bed}} \cup \partial\Omega^{\text{base}}_{\text{bed}}$ (or $\partial\Omega^{\text{part}}_{\text{bed}}$ and $\partial\Omega^{\text{part}}_{\text{base}}$). For this purpose, two strategies are studied: (1) constant-valued HTC and (2) VDA (cf. Section 7.3). They lead to further submodels, such as HTC-PP, VDA-PP and VDA-P. Only these three are covered in this work. In particular, their computational cost and accuracy with respect to the reference HF is assessed in Section 7.4.

### 7.2.2 Governing equation

Heat transfer in $\Omega$ at the part scale is governed by the balance of energy equation, expressed as

$$C(u)\partial_t u - \boldsymbol{\nabla} \cdot (k(u)\nabla u) = r, \quad \text{in } \Omega(t), \quad t > 0, \tag{7.1}$$

where $C(u)$ is the heat capacity coefficient, given by the product of the density of the material $\rho(u)$ and the specific heat $c(u)$, and $k(u) \geq 0$ is the thermal conductivity. Furthermore, $r$ is the rate of energy supplied to the system per unit volume by a very intense and concentrated laser or electron beam that moves in time according to a user-defined deposition sequence, referred to as the *scanning path*. After integrating Equation (7.1) in time, $r$ is computed as

$$\begin{cases} r(\boldsymbol{x}) = \frac{\eta \mathrm{W}}{\mathrm{V}_{\mathrm{pool}}^{\Delta t}} & \text{if } \boldsymbol{x} \in \mathrm{V}_{\mathrm{pool}}^{\Delta t} \\ 0 & \text{elsewhere} \end{cases}$$

with W the laser power [W] (watt), $\eta$ the heat absorption coefficient, a measure of the laser efficiency, and $\mathrm{V}_{\mathrm{pool}}^{\Delta t}$ is the region swept by the laser during the time increment $\Delta t$. Note that phase transformations occur much faster than the diffusion process and the amount of latent heat is much smaller than the energy input [54]. That is why, given the scale of analysis, phase-change effects are neglected.

In case of the HF model, the powder-bed is included into the computational domain. As a result, there are two distinct material phases playing a role in Equation (7.1), i.e. solid and powder, which are separated at the $\partial\Omega_{\mathrm{bed}}^{\mathrm{part}} \cup \partial\Omega_{\mathrm{bed}}^{\mathrm{base}}$ interface between the solid part-plate ensemble and the granular powder-bed. To model the material in powder state, the thermophysical properties are determined in terms of the bulk material data at solid state and the porosity of the powder-bed $\phi$. In particular, density and specific heat are obtained as

$$\begin{aligned} \rho_{\mathrm{pwd}} &= \rho_{\mathrm{solid}}(1-\phi), \quad \text{and} \\ c_{\mathrm{pwd}} &= c_{\mathrm{solid}}, \end{aligned}$$

while determination of thermal conductivity $k_{\mathrm{pwd}}$ is more involved and often relies on empirical expressions. Among the models present in the literature, Sih and Barlow [171] establish, for a powder-bed composed of spherical particles [69], the relation

$$\begin{aligned} \frac{k_{\mathrm{pwd}}}{k_{\mathrm{gas}}} = &\left(1 - \sqrt{1-\phi}\right)\left(1 + \phi\frac{k_{\mathrm{rad}}}{k_{\mathrm{gas}}}\right) \\ &+ \sqrt{1-\phi}\frac{2}{1-\frac{k_{\mathrm{gas}}}{k_{\mathrm{solid}}}}\left(\frac{2}{1-\frac{k_{\mathrm{gas}}}{k_{\mathrm{solid}}}}\ln\frac{k_{\mathrm{solid}}}{k_{\mathrm{gas}}} - 1\right) \\ &+ \sqrt{1-\phi}\frac{k_{\mathrm{rad}}}{k_{\mathrm{gas}}} \end{aligned} \tag{7.2}$$

where $k_{\text{gas}}$ is the thermal conductivity of the surrounding air or gas and $k_{\text{rad}}$ is the contribution of radiation amongst the individual powder particles, given by Damköhler's equation:

$$k_{\text{rad}} = \frac{4}{3}\sigma u^3 D_{\text{pwd}},$$

with $D_{\text{pwd}}$ the average diameter of the particles and $\sigma$ the Stefan-Boltzmann constant, i.e. $5.67 \cdot 10^{-8}$ $[\text{Wm}^{-2}\text{K}^{-4}]$.

### 7.2.3 Boundary conditions

Equation (7.1) is subject to the initial condition

$$u(\boldsymbol{x}, 0) = u_0,$$

with $u_0$ the pre-heating temperature of the build chamber, and the boundary conditions applied on the regions shown in Figure 7.2, which depend on the model variant:

1. *HF model:* Heat convection and radiation through the free surface $\partial\Omega_{\text{air}}^{\text{bed}} \cup \partial\Omega_{\text{air}}^{\text{part}}$, heat conduction through the lateral wall on $\partial\Omega_{\text{lat}}^{\text{bed}} \cup \partial\Omega_{\text{lat}}^{\text{base}}$ and heat conduction through the bottom wall of the plate on $\partial\Omega_{\text{down}}$.

2. *PP model:* Heat convection and radiation only through $\partial\Omega_{\text{air}}^{\text{part}}$, heat conduction through the powder bed along $\partial\Omega_{\text{bed}}^{\text{part}} \cup \partial\Omega_{\text{bed}}^{\text{base}}$, heat conduction through the lateral wall only on $\partial\Omega_{\text{lat}}^{\text{base}}$ and heat conduction through the bottom wall of the plate on $\partial\Omega_{\text{down}}$.

3. *P model:* Heat convection and radiation through $\partial\Omega_{\text{air}}^{\text{part}}$, heat conduction through the powder bed along $\partial\Omega_{\text{bed}}^{\text{part}}$ and heat conduction through the building plate on $\partial\Omega_{\text{base}}^{\text{part}}$.

After linearising the Stefan-Boltzmann law for heat radiation [54], all heat loss boundary conditions mentioned above can be expressed in terms of the Newton law of cooling:

$$q_{\text{loss}}(u, t) = h_{\text{loss}}(u)(u - u_{\text{loss}}(t)), \tag{7.3}$$

in $\partial\Omega^{\text{loss}}(t)$, $t > 0$, where *loss* refers to the kind of heat loss mechanism and the boundary region where it applies. Alternatives to Equation (7.3) can also be considered, e.g. the temperature on $\partial\Omega_{\text{down}}$ can be prescribed to $u_0$ [54], taking into account that thermal inertia at the building plate is much larger than at the printed part.

For the HTC model, $h_{\text{loss}}$ and $u_{\text{loss}}$ are always taken as constant, due to the difficulty in describing experimentally the temperature dependency of these quantities. On the other hand, for heat loss through a region modelled with a VDA, $h_{\text{loss}}$ and $u_{\text{loss}}$ are both temperature dependent and computed as described in Section 7.3.

## 7.3 Virtual domain approximation

As mentioned in Section 7.2.1, if the powder-bed (and the building plate) are not included in the domain of analysis, then heat transfer through these regions must be accounted for with equivalent heat conduction boundary conditions. According to Equation (7.3), this leads to the determination of $h_{\text{loss}}$ and $u_{\text{loss}}$ values for heat loss through the solid-powder interface and the part-plate interface.

However, lack of experimental data and physical modelling in the literature are a hurdle in the way to estimate these quantities. The authors are not aware of the existence of, for instance, well-established temperature-dependent empirical correlations for the HTC solid-powder or approximate time evolution laws for the temperature at the building plate.

To overcome these challenges, this work introduces a novel method, i.e. the Virtual Domain Approximation (VDA). The VDA enhances a state-of-the-art thermal contact model for metal casting analysis [53], such that it includes the effect of the thermal inertia of the powder bed (or the building plate). The result of the VDA procedure is a temperature-dependent physics-based way to evaluate $h_{\text{loss}}$ and $u_{\text{loss}}$ in Equation (7.3), a clear improvement with respect to the HTC model.

Assuming the PP variant, the VDA method is outlined in Figure 7.3 and explained in the following paragraphs. It is based on the assumption that the entire 3D heat transfer problem across the powder bed $\Omega^{\text{bed}}$ can be modelled with an equivalent 1D heat conduction problem across a wall on the contour of $\partial\Omega^{\text{part}}_{\text{bed}} \cup \partial\Omega^{\text{base}}_{\text{bed}}$. As a result, heat loss through the powder bed is assumed to be unidimensional and orthogonal to the solid-powder interface, i.e. the VDA neglects diffusion in directions other than the normal to the interface.

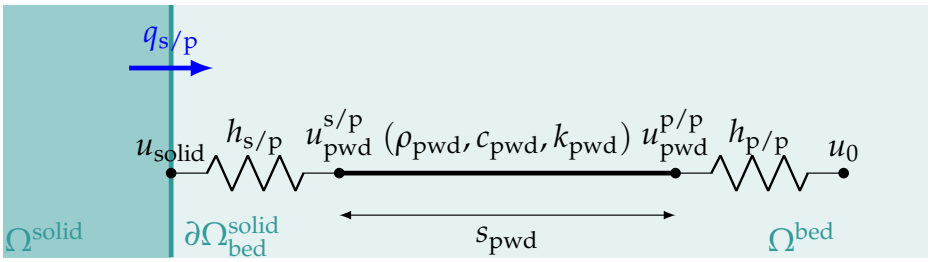



Figure 7.3: Illustration of the VDA method. The main assumption of the method consists in replacing the 3D heat conduction problem across $\Omega^{\text{bed}}$ by an equivalent 1D heat conduction problem with the thermal circuit shown in the figure. Then, Equation (7.1) is only solved in $\Omega^{\text{solid}}$. From the solution of the equivalent 1D problem a boundary condition of the form given in Equation (7.3) (with loss $=$ s/p) is evaluated at each integration point located on $\partial\Omega^{\text{solid}}_{\text{bed}}$.

According to this, the setting considers the 1D time-dependent heat transfer problem through the powder bed given by

$$\rho_{\text{pwd}}(u)c_{\text{pwd}}(u)\partial_t u - \partial_x(k_{\text{pwd}}(u)\partial_x u) = 0, \quad \text{in } (0, s_{\text{pwd}}), \quad t > 0, \tag{7.4}$$

as the classic model for heat conduction through a plane wall [31]. Here, the plane wall in $(0, s_{\text{pwd}})$ represents the region of the powder-bed subject to relevant thermal effects of the printing process (i.e. with presence of significant thermal gradients). The wall is in contact with the surface of the part at $x = 0$ and has average length $s_{\text{pwd}}$, referred to as the effective thermal thickness. A point in the powder bed that distances itself more than $s_{\text{pwd}}$ with respect to the part is barely affected by the thermal gradient and undergoes little changes in temperature during the printing process, i.e. it mostly remains at $u_0$.

The initial condition is the same one as in Equation (7.1), i.e. $u(0) = u_0$. Besides, thermal contact holds at both surfaces of the $s_{\text{pwd}}$-thick powder-bed wall: At one side ($x = 0$), the powder-bed surface is in contact with the solid (part or plate) surface. As the powder-bed is a porous medium, it is reasonable to assume that the contacting surfaces do not match perfectly. In particular, the standard Fourier law does not hold. Hence, heat flux is computed as the product of an HTC $h_{\text{s/p}}$ and the thermal gap $(u_{\text{solid}}(t) - u^{\text{s/p}}_{\text{pwd}}(t))$ between both surfaces, i.e.

$$q_{\text{s/p}}(u,t) = h_{\text{s/p}}(u_{\text{solid}} - u^{\text{s/p}}_{\text{pwd}}), \text{ in } x = 0, \ t > 0. \tag{7.5}$$

At the other side ($x = s_{\text{pwd}}$), thermal powder-to-powder contact applies. The expression for the heat flux is analogous to Equation (7.5); in this case, the HTC is $h_{\text{p/p}}$ and the thermal gap can be taken as $(u^{\text{p/p}}_{\text{pwd}}(t) - u_0)$, in agreement with the discussion above. Hence,

$$q_{\text{p/p}}(u,t) = h_{\text{p/p}}(u^{\text{p/p}}_{\text{pwd}} - u_0), \text{ in } x = s_{\text{pwd}}, \ t > 0.$$

Assuming now a VDA problem attached to each integration point located on a face in $\partial\Omega^{\text{part}}_{\text{bed}} \cup \partial\Omega^{\text{base}}_{\text{bed}}$, $u_{\text{solid}}(t)$ is determined from the solution of Equation (7.1) at any $t > 0$, whereas the material properties of the powder, i.e. $\rho_{\text{pwd}}$, $c_{\text{pwd}}$ and $k_{\text{pwd}}$, and the parameters $s_{\text{pwd}}$, $h_{\text{s/p}}$ and $h_{\text{p/p}}$ are assumed to be known data. This leaves the VDA model as a simple well-posed 1D transient heat transfer problem. The first step of the VDA method is to apply a suitable discretisation of this 1D problem. The resulting linear system is then modified, by adding $u_{\text{solid}}$ as an additional unknown with the extra equation given by Equation (7.5). Following this, static condensation allows one to recover $q_{\text{s/p}}$, such that it no longer depends on $u^{\text{s/p}}_{\text{pwd}}$, only on $u_{\text{solid}}$, the known data and the type of discretisation. In this way, an equivalent expression of Equation (7.5) is obtained, that takes the same form as Equation (7.3) and can be straightforwardly evaluated in the discrete form of Equation (7.1).

Let us illustrate the procedure considering (1) a discretisation in space of Equation (7.4) with a single linear Lagrangian finite element along the thickness $s_{\text{pwd}}$; (2) a forward first order finite difference to approximate the time derivative; and (3) constant material properties. For this setting, the linear system of the VDA problem at the

$n$-th time step ($n \geq 0$) is

$$\begin{bmatrix} h_{\mathrm{s/p}} + \frac{M_{11}}{\Delta t} + K_{11} & K_{12} \\ K_{21} & \frac{M_{22}}{\Delta t} + K_{22} + h_{\mathrm{p/p}} \end{bmatrix} \begin{bmatrix} u_{\mathrm{pwd}}^{\mathrm{s/p},n+1} \\ u_{\mathrm{pwd}}^{\mathrm{p/p},n+1} \end{bmatrix} = \begin{bmatrix} \frac{M_{11}}{\Delta t} u_{\mathrm{pwd}}^{\mathrm{s/p},n} + h_{\mathrm{s/p}} u_{\mathrm{solid}}^{n+1} \\ \frac{M_{22}}{\Delta t} u_{\mathrm{pwd}}^{\mathrm{p/p},n} + h_{\mathrm{p/p}} u_0 \end{bmatrix}, \tag{7.6}$$

where $M_{ij}$ and $K_{ij}$ are the coefficients of the mass $\mathbf{M}$ and conductivity $\mathbf{K}$ matrices for the 1D linear lagrangian FE of the VDA problem, with

$$\mathbf{M} = \rho_{\mathrm{pwd}} c_{\mathrm{pwd}} \frac{s_{\mathrm{pwd}}}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{K} = \frac{k_{\mathrm{pwd}}}{s_{\mathrm{pwd}}} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

If $u_{\mathrm{solid}}^{n+1}$ is reinterpreted as an unknown, then, taking Equation (7.5) as an extra equation of the system, Equation (7.6) is augmented to

$$\begin{bmatrix} h_{\mathrm{s/p}} & -h_{\mathrm{s/p}} & 0 \\ -h_{\mathrm{s/p}} & h_{\mathrm{s/p}} + \frac{M_{11}}{\Delta t} + K_{11} & K_{12} \\ 0 & K_{21} & \frac{M_{22}}{\Delta t} + K_{22} + h_{\mathrm{p/p}} \end{bmatrix} \begin{bmatrix} u_{\mathrm{solid}}^{n+1} \\ u_{\mathrm{pwd}}^{\mathrm{s/p},n+1} \\ u_{\mathrm{pwd}}^{\mathrm{p/p},n+1} \end{bmatrix} = \begin{bmatrix} q_{\mathrm{s/p}}^{n+1} \\ \frac{M_{11}}{\Delta t} u_{\mathrm{pwd}}^{\mathrm{s/p},n} \\ \frac{M_{22}}{\Delta t} u_{\mathrm{pwd}}^{\mathrm{p/p},n} + h_{\mathrm{p/p}} u_0 \end{bmatrix}, \tag{7.7}$$

Identifying now the blocks

$$\begin{aligned} \mathbf{A}_{12} &= [\, -h_{\mathrm{s/p}}\, 0\,], \\ \mathbf{A}_{21} &= \mathbf{A}_{12}^{tr}, \\ \mathbf{A}_{22} &= \begin{bmatrix} h_{\mathrm{s/p}} + \frac{M_{11}}{\Delta t} + K_{11} & K_{12} \\ K_{21} & \frac{M_{22}}{\Delta t} + K_{22} + h_{\mathrm{p/p}} \end{bmatrix} \text{ and} \\ \mathbf{b}_2 &= \begin{bmatrix} \frac{M_{11}}{\Delta t} u_{\mathrm{pwd}}^{\mathrm{s/p},n} \\ \frac{M_{22}}{\Delta t} u_{\mathrm{pwd}}^{\mathrm{p/p},n} + h_{\mathrm{p/p}} u_0 \end{bmatrix}, \end{aligned}$$

Equation (7.7) can be rewritten as

$$\begin{bmatrix} h_{\mathrm{s/p}} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} u_{\mathrm{solid}}^{n+1} \\ \mathbf{u}_{\mathrm{pwd}}^{n+1} \end{bmatrix} = \begin{bmatrix} q_{\mathrm{s/p}}^{n+1} \\ \mathbf{b}_2 \end{bmatrix}. \tag{7.8}$$

Following this, application of static condensation to Equation (7.8) to remove the unknowns of $\mathbf{u}_{\mathrm{pwd}}^{n+1}$, leads to the relation

$$\begin{aligned} &\left[h_{\mathrm{s/p}} - \mathbf{A}_{12}(\mathbf{A}_{22})^{-1}\mathbf{A}_{21}\right] u_{\mathrm{solid}}^{n+1} \\ &\quad = q_{\mathrm{s/p}}^{n+1} - \mathbf{A}_{12}(\mathbf{A}_{22})^{-1}\mathbf{b}_2. \end{aligned} \tag{7.9}$$

From this point, it suffices to compare Equation (7.9) with Equation (7.3) to see that

$$\begin{aligned} h_{\mathrm{loss}} &= h_{\mathrm{s/p}} - \mathbf{A}_{12}(\mathbf{A}_{22})^{-1}\mathbf{A}_{21} \\ u_{\mathrm{loss}} &= -\,\mathbf{A}_{12}(\mathbf{A}_{22})^{-1}\mathbf{b}_2 \,\Big/\, h_{\mathrm{loss}} \end{aligned} \tag{7.10}$$

The VDA method can be applied verbatim (with enhanced accuracy), if several and/or higher order FEs are considered; some examples are listed in Table 7.1. The reason is that the discretisation leads to the same block structure of Equation (7.8). Likewise, the part-plate thermal contact in the P model can be analogously constructed; in this case, the material properties are those of the plate.

One linear FE along the VDA (1E-Q1)

$$h_{\text{loss}} = \frac{4h_{\text{s/p}}(h_{\text{p/p}} + m)\hat{k} + 2mh_{\text{s/p}}h_{\text{p/p}} + m^2h_{\text{s/p}}}{4(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k} + (4h_{\text{s/p}} + 2m)h_{\text{p/p}} + 2mh_{\text{s/p}} + m^2}$$

$$u_{\text{loss}} = \frac{4h_{\text{s/p}}h_{\text{p/p}}\hat{k}u_0 + 2mh_{\text{s/p}}\hat{k}u_{\text{pwd}}^{\text{p/p,n}} + (2\hat{k} + 2h_{\text{p/p}} + m)mh_{\text{s/p}}u_{\text{pwd}}^{\text{s/p,n}}}{4(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k} + 4(h_{\text{s/p}} + \frac{m}{2})h_{\text{p/p}} + 2mh_{\text{s/p}} + m^2}$$

Two linear FE along the VDA (2E-Q1)

$$h_{\text{loss}} = \frac{128(h_{\text{p/p}} + m)h_{\text{s/p}}\hat{k}^2 + (64h_{\text{p/p}} + 24m)mh_{\text{s/p}}\hat{k} + 4m^2h_{\text{s/p}}h_{\text{p/p}} + m^3h_{\text{s/p}}}{128(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k}^2 + ((128h_{\text{s/p}} + 64m)h_{\text{p/p}} + 64mh_{\text{s/p}} + 24m^2)\hat{k} + (16mh_{\text{s/p}} + 4m^2)h_{\text{p/p}} + 4m^2h_{\text{s/p}} + m^3}$$

$$u_{\text{loss}} = \frac{128h_{\text{s/p}}h_{\text{p/p}}\hat{k}^2u_0 + 32mh_{\text{s/p}}\hat{k}^2u_{\text{pwd}}^{\text{p/p,n}} + (64mh_{\text{s/p}}\hat{k}^2 + (32mh_{\text{s/p}}h_{\text{p/p}} + 8m^2h_{\text{s/p}})\hat{k})u_{\text{pwd}}^{\text{m,n}}}{128(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k}^2 + ((128h_{\text{s/p}} + 64m)h_{\text{p/p}} + 64mh_{\text{s/p}} + 24m^2)\hat{k} + (16mh_{\text{s/p}} + 4m^2)h_{\text{p/p}} + 4m^2h_{\text{s/p}} + m^3}$$
$$+ \frac{(32mh_{\text{s/p}}\hat{k}^2 + (32mh_{\text{s/p}}h_{\text{p/p}} + 16m^2h_{\text{s/p}})\hat{k} + 4m^2h_{\text{s/p}}h_{\text{p/p}} + m^3h_{\text{s/p}})u_{\text{pwd}}^{\text{s/p,n}}}{128(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k}^2 + ((128h_{\text{s/p}} + 64m)h_{\text{p/p}} + 64mh_{\text{s/p}} + 24m^2)\hat{k} + (16mh_{\text{s/p}} + 4m^2)h_{\text{p/p}} + 4m^2h_{\text{s/p}} + m^3}$$

One quadratic FE along the VDA (1E-Q2)

$$h_{\text{loss}} = \frac{288(h_{\text{p/p}} + m)h_{\text{s/p}}\hat{k}^2 + (132h_{\text{p/p}} + 36m)mh_{\text{s/p}}\hat{k} + 6m^2h_{\text{s/p}}h_{\text{p/p}} + m^3h_{\text{s/p}}}{288(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k}^2 + ((288h_{\text{s/p}} + 132m)h_{\text{p/p}} + 132mh_{\text{s/p}} + 36m^2)\hat{k} + (36mh_{\text{s/p}} + 6m^2)h_{\text{p/p}} + 6m^2h_{\text{s/p}} + m^3}$$

$$u_{\text{loss}} = \frac{(288\hat{k}^2 - 12m\hat{k})h_{\text{s/p}}h_{\text{p/p}}u_0 + (48m\hat{k}^2 - 2m^2h_{\text{s/p}})\hat{k}u_{\text{pwd}}^{\text{p/p,n}} + (192mh_{\text{s/p}}\hat{k}^2 + (96mh_{\text{s/p}}h_{\text{p/p}} + 16m^2h_{\text{s/p}})\hat{k})u_{\text{pwd}}^{\text{m,n}}}{288(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k}^2 + ((288h_{\text{s/p}} + 132m)h_{\text{p/p}} + 132mh_{\text{s/p}} + 36m^2)\hat{k} + (36mh_{\text{s/p}} + 6m^2)h_{\text{p/p}} + 6m^2h_{\text{s/p}} + m^3}$$
$$+ \frac{(48mh_{\text{s/p}}\hat{k}^2 + (48mh_{\text{s/p}}h_{\text{p/p}} + 22m^2h_{\text{s/p}})\hat{k} + 6m^2h_{\text{s/p}}h_{\text{p/p}} + m^3h_{\text{s/p}})u_{\text{pwd}}^{\text{s/p,n}}}{288(h_{\text{s/p}} + h_{\text{p/p}} + m)\hat{k}^2 + ((288h_{\text{s/p}} + 132m)h_{\text{p/p}} + 132mh_{\text{s/p}} + 36m^2)\hat{k} + (36mh_{\text{s/p}} + 6m^2)h_{\text{p/p}} + 6m^2h_{\text{s/p}} + m^3}$$

One linear FE along the VDA, $u_{\text{pwd}}^{\text{s/p}} \approx u_{\text{solid}}$ and $u_{\text{pwd}}^{\text{p/p}} \approx u_0$ (1E-Q1-D)

$$h_{\text{loss}} = \frac{1}{2}m + \hat{k}$$

$$h_{\text{loss}}u_{\text{loss}} = \frac{1}{2}mu_{\text{solid}}^{n} + \hat{k}u_0$$

Table 7.1: Expressions of $h_{\text{loss}}$ and $u_{\text{loss}}$ in terms of the VDA parameters for different types of discretisations. Forward first order finite difference in time and constant material properties are assumed. $m := \rho_{\text{pwd}}c_{\text{pwd}}s_{\text{pwd}}/\Delta t$, $\hat{k} := k_{\text{pwd}}/s_{\text{pwd}}$ and $u_{\text{pwd}}^{\text{m,n}}$ is the temperature in the middle of the VDA at the previous time step.

But the main advantage of the VDA concerns computational efficiency. As the method ends up extracting a parametrized closed-form expression to evaluate the equivalent boundary condition, there is no need to solve a linear system at each integration point. The additional cost with respect to the HTC model merely consists in extra storage of the temperature values of the VDA at previous time steps (depending on the time integration scheme) and extra floating point operations to evaluate Equation (7.10).

The main limitation of the method is that the contour of the part-powder interface is often a smooth 2D shape and, as a result, heat transfer across the powder-bed is not unidimensional and orthogonal to the interface. Depending on the curvature of the geometry, heat diffusion through directions other than the normal to the discrete boundary may be relevant and the VDA risks of underestimating the amount of heat loss. For simple shapes (e.g. cylinder or sphere) the local curvature of $\partial\Omega^{\text{part}}_{\text{bed}} \cup \partial\Omega^{\text{base}}_{\text{bed}}$ can be taken into account by considering a modified 1D heat transfer problem with

$$\begin{aligned} \hat{\rho}_{\text{pwd}}\hat{c}_{\text{pwd}} &= \rho_{\text{pwd}}c_{\text{pwd}}F_{\text{volume}}, \\ \hat{k}_{\text{pwd}} &= k_{\text{pwd}}F_{\text{volume}}, \text{ and} \\ \hat{h}_{\text{loss}} &= h_{\text{loss}}F_{\text{surface}}, \end{aligned}$$

with the geometrical correction factors $F_{\text{volume}}$ and $F_{\text{surface}}$ defined as

$$F_{\text{volume}} = \frac{V_{\text{pwd}}}{A^{\text{part}}_{\text{ext}} s_{\text{pwd}}} \quad \text{and} \quad F_{\text{surface}} = \frac{A^{\text{pwd}}_{\text{ext}}}{A^{\text{part}}_{\text{ext}}},$$

where, as shown in Figure 7.4, $V_{\text{pwd}}$ is the volume of the fraction of powder bed modelled by the VDA model and $A^{\text{pwd}}_{\text{ext}}$ and $A^{\text{part}}_{\text{ext}}$ are the areas of the external surfaces of the VDA and the part. These corrections are intended to compensate for the nonorthogonal heat loss neglected by the standard approach.

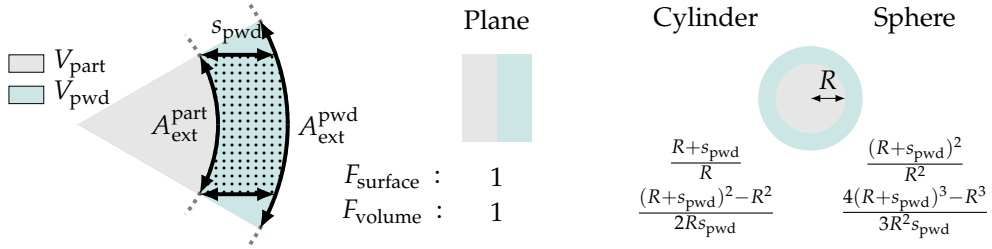



Figure 7.4: Geometrical correction factors to account for non-unidimensional heat transfer.

Another drawback is that the VDA, in spite of adding physics into the computation of the boundary condition, still needs to estimate some unknown quantities, such as $h_{\text{s/p}}$, $h_{\text{p/p}}$ or $s_{\text{pwd}}$. Assuming that $u^{\text{s/p}}_{\text{pwd}} \approx u_{\text{solid}}$ and $u^{\text{p/p}}_{\text{pwd}} \approx u_0$, one can get rid of estimating the HTCs, because the boundary conditions at both ends of the VDA problem become of Dirichlet type. A relation like Equation (7.3) is then obtained following exactly the same procedure outlined above. Additionally, the determination of $s_{\text{pwd}}$ is object of discussion in Section 7.4.1.

## 7.4 Numerical experiments

### 7.4.1 Verification of the VDA against the HF model

The purpose of the first numerical experiment is to verify the new VDA formulation and demonstrate its capabilities and benefits in the thermal simulation of a SLM or DMLS process. Three model variants (cf. Section 7.2.1) are considered: (1) HF, (2) VDA-PP and (3) VDA-P. The first one is taken as the numerical reference for the other two, due to its higher physical representativeness.

The example is designed to be simple, but suitable enough to compare the accuracy and efficiency of models (2) and (3) with respect to model (1). According to this, the object of simulation is the printing of a $10 \times 10 \times 10$ [mm$^3$] cube on top of a $110 \times 110 \times 20$ [mm$^3$] metal substrate. Different materials are selected for the cubic sample and the build plate: while the former is made of Maraging steel M300, the latter is composed of Stainless Steel (SS) 304L. Bulk temperature-dependent thermal properties of SS 304L [137] are represented in Figure 7.5. On the other hand, Table 7.2 prescribes constant density and specific heat values for M300, due to lack of temperature-dependent data, and the M300 powder is assumed to have 54 % of relative density.

| **Temperature [°C]** | **Density [kg/m$^3$]** | **Specific heat [J/gK]** | **Conductivity [W/mK]** |
|---|---|---|---|
| 20.0 | 8,100 | 500 | 14.2 |
| 600.0 | | | 21.0 |
| 1,300.0 | | | 28.6 |
| 1,600.0 | | | 28.6 |

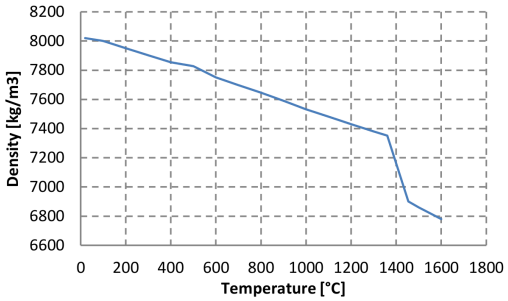
Table 7.2: Maraging steel M300 thermal bulk material properties [156].

The printing process considers a layer thickness of 30 [µm]. Therefore, a total number of 333 layers are deposited to build the sample. The values of the remaining process parameters are detailed in Table 7.3. Note that there is no information regarding the scanning path, because the simulation considers a layer-by-layer deposition sequence. This means that the printing of a complete layer is simulated in a single time step. As a result, the simulation follows an alternating sequence in time consisting of:

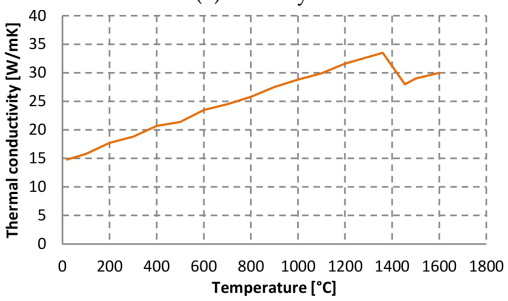
1. **Printing step:** A new layer is activated, i.e. added into the computational domain, and the problem is solved applying the energy input necessary to fuse the powder of the whole layer. The time increment is given by the *scanning time*.

2. **Cooling step:** The problem is solved without heat application to account for the time lowering the plate, recoat time and laser relocation. Here, the time step is the *recoating time*.

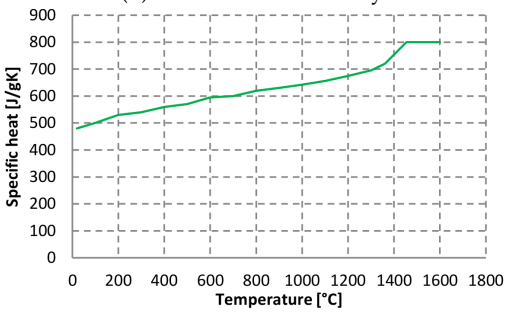
Figure 7.6 shows the FE discretisations of the three tested models. All three cases consider structured meshes of varying size to adequately capture the physics of the process. In particular, a finer layer-conforming mesh is prescribed for the fabricated




(a) Density.




(b) Thermal conductivity.




(c) Specific heat.

Figure 7.5: Stainless Steel 304L thermal bulk material properties [137].

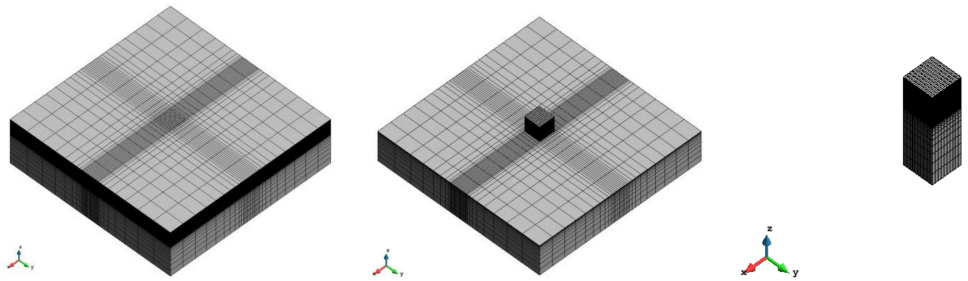
| | | |
|---|---|---|
| Power input | 375 | [W] |
| Scanning time | 2 | [s] |
| Recoating time | 10 | [s] |
| Absorption coefficient | 0.64 | - |
| Layer thickness | 30 | [µm] |
| Plate temperature | 20 | [°C] |

Table 7.3: Process parameters for the example in Section 7.4.1.

part, whereas a coarser mesh is used away from the printed part. As observed in Table 7.4, the mesh size decreases one order of magnitude from the HF model to the VDA-PP and VDA-P ones.

Concerning the boundary conditions, the top surfaces of the three models are subject to heat transfer through the surrounding air with an HTC of 10 [W/m$^2$K] and air temperature of 20 [°C]. On the other hand, the same HTC and environment temperature are assigned at the lateral and bottom walls of the build chamber.

The last ingredient is to characterize the BCs at the solid-powder and part-plate



(a) HF model. (b) VDA-PP model. (c) VDA-P model.

Figure 7.6: FE meshes of the three models tested in the example of Section 7.4.1.

| Model | Mesh size [DOFs] | Runtime [h]-[%] |
|---|---|---|
| HF | 274.5k | 17.6 - 100 |
| VDA-PP | 66.5k | 1.3 - 7.4 |
| VDA-P | 49.9k | 0.8 - 4.5 |

Table 7.4: Example of Section 7.4.1. Mesh sizes and simulation times of the reduced models are one order of magnitude down the reference model.

contacting surfaces, by establishing the VDA parameters for the VDA solid-powder (virtual powder) and the VDA part-plate (virtual plate). Recall that only the first applies to the VDA-PP model, whereas both apply to the VDA-P one.

Both VDAs adopt the full Dirichlet variant (cf. (1E-Q1-D) in Table 7.1) described at the end of Section 7.3 with a single linear Lagrangian FE and without geometrical correction factors, for the sake of reducing the number of unknown parameters. In particular, the only ones that need estimation are the virtual domain thicknesses $s_{\text{pwd}}$ and $s_{\text{plate}}$ and the virtual domain environment temperature $u_0$.

The strategy for such estimation is based on simple calibration with respect to the HF model. Let us explain the procedure step-by-step to model the virtual powder (analogous for the virtual plate). First, the temperature distribution of the HF model is analysed to identify the region concentrating the strongest gradients. As shown in Figure 7.7 for the virtual powder, at about 14 [mm] away from the part, the temperature drops to 90 [°C]. Further away, the thermal gradients are apparently smoother. Therefore, $u_0$ is set to 90 [°C] and $s_{\text{pwd}}$ is initially approximated by 14 [mm]. Calibration with respect to the thermal history of the HF model at a selected point follows to correct $s_{\text{pwd}}$ to the final value. VDA parameter values obtained with this approach are listed in Table 7.5.

| Domain | VDA thickness [mm] | $u_0$ [°C] |
|---|---|---|
| Powder | 10 | 90 |
| Plate | 11 | 20 |

Table 7.5: Example of Section 7.4.1. VDA parameters after calibration with respect to the reference model.

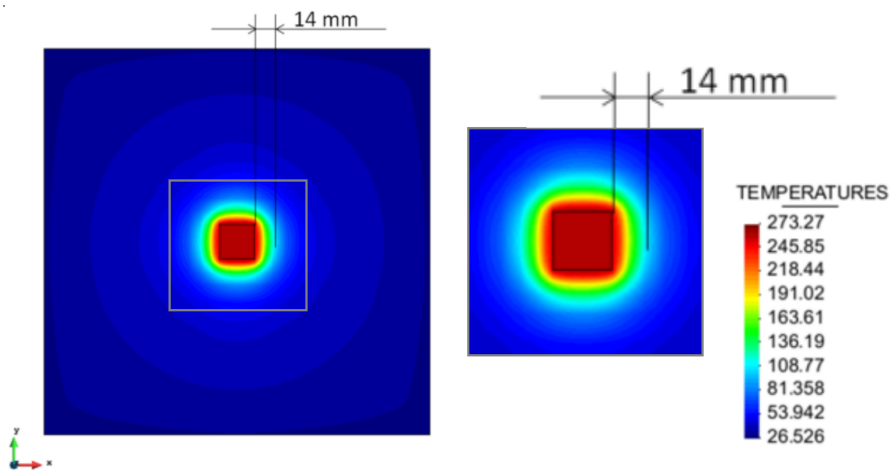



Figure 7.7: Contour plot of temperatures of the HF model (XY view + close up), showing the extent of the region concentrating thermal gradients.

The numerical experiments for this example are supported by the in-house research software COupled MEchanical and Thermal (COMET) [49] and GiD [59, 134] as a pre- and postprocessing software. Figures 7.8 and 7.9 compare the temperature evolutions of the reduced models against the reference HF model, at a point located in the middle of the bottom surface of the part. As observed, both VDA-PP and VDA-P are capable of reproducing the thermal response of the HF model with errors bounded by 10 % and 20 % with the Dirichlet VDA variant. The VDA-P is clearly a little less accurate than the former, as expected. However, as seen in Table 7.4, the computational running times of the VDA models are one order of magnitude less than the HF model. This showcases the ability of the new formulation to approximate the response of a high-fidelity model, but with significantly increased efficiency, and it opens the path for larger scale and experimentally-based simulations, such as the one in the next section.

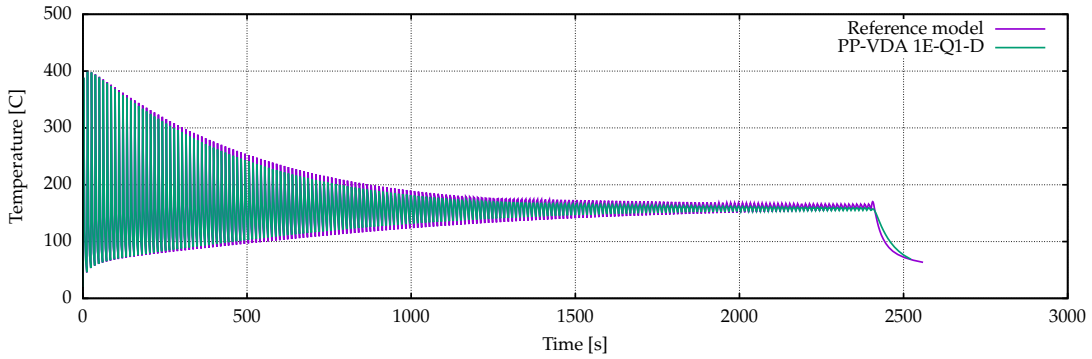
Further experiments were carried out with the VDA-PP model to assess the other VDA discrete forms in Table 7.1. Taking the same parameters as with the Dirichlet case, extra quantities $h_{\mathrm{s/p}}$ and $h_{\mathrm{p/p}}$ were not estimated, they were set to 1,000 [W/m$^2$K] and 10 [W/m$^2$K]. Results in Figure 7.8(b) show that more accurate discretisations, lead to better approximations of the thermal response, as expected.

### 7.4.2 Verification and validation against physical experiments

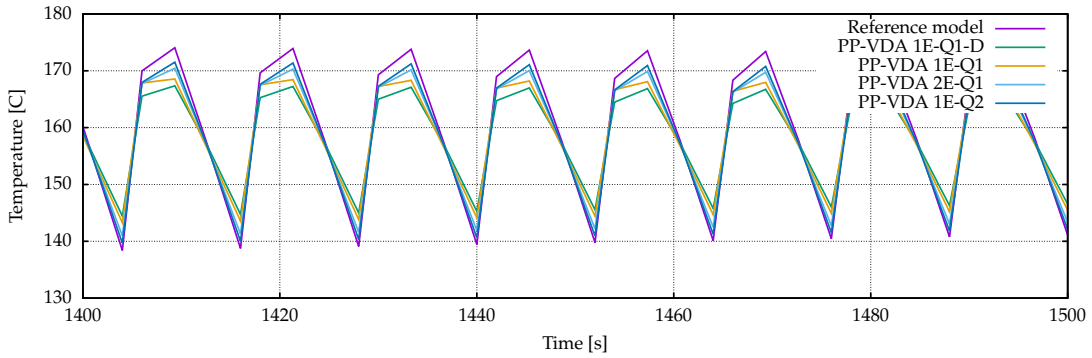
#### Experimental campaign

An experimental campaign took place at the Monash Centre for Additive Manufacturing (MCAM) in Melbourne, Australia, with the purpose of (1) calibrating experimentally the thermal analysis FE framework described in Section 7.2 and (2) assess the novel VDA model presented in Section 7.3.

The printing system employed for the experiments is the EOSINT M280 from Electro Optical Systems (EOS) GmbH. It uses an Yb-fibre laser with variable beam width




(a) VDA-PP model.




(b) Close up of VDA-PP model halfway through the simulation. Comparison against other VDA variants. Variant names in the key are the ones given in Table 7.1.

Figure 7.8: Example of Section 7.4.1. Comparison of the reduced VDA-PP model against the HF model.

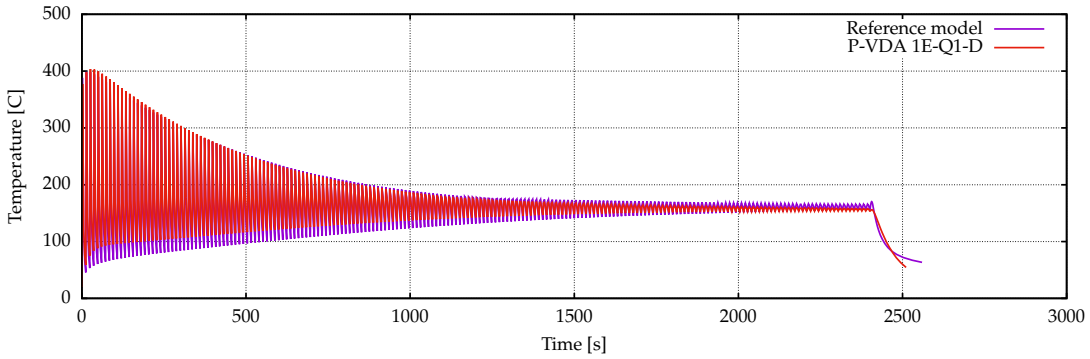
and power up to 400 [W]. The printing process is carried out in a closed $250 \times 250 \times 325$ [mm$^3$] chamber subject to a laminar flow of argon that prevents oxidation.

The printed specimen is an oblique square prism with the lower base located in the centre of the building plate and a 45-degrees slope, as shown in Figure 7.10. Cross section dimensions are $30 \times 30$ [mm$^2$] and the height is 80 [mm].

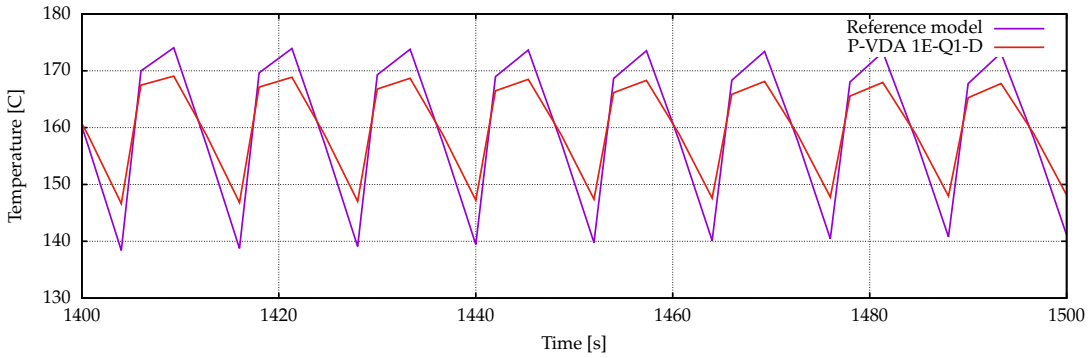
Eight thermocouples for temperature measurements are inserted into 0.78 [mm] diameter holes at the upward and downward facing lateral surfaces of the prism (CH1-4 and CH7-8) or in the powder bed (CH5-6). The position of the channels is indicated in Table 7.6 and Figure 7.11. The printing job was interrupted at 20.58 [mm] height to install the thermocouples on a set of supporting structures that were printed together with the prism, as shown in Figure 7.12.

K-type thermocouples and a Graphtec GL-900 8 high-speed data-logger are used for data gathering. The sampling rate of the data logger is 1 [ms] and the time constant of the thermocouples is 7 [ms]. As thermocouples are not welded and can move inside the hole, their measurements can be perturbed.

The process parameters used for the printing job are described in Table 7.7. The layer thickness is set to 30 [µm]. This means that 647 layers are deposited in about 3.5 [h] to build the samples. As observed, the recoat and laser relocation time varies

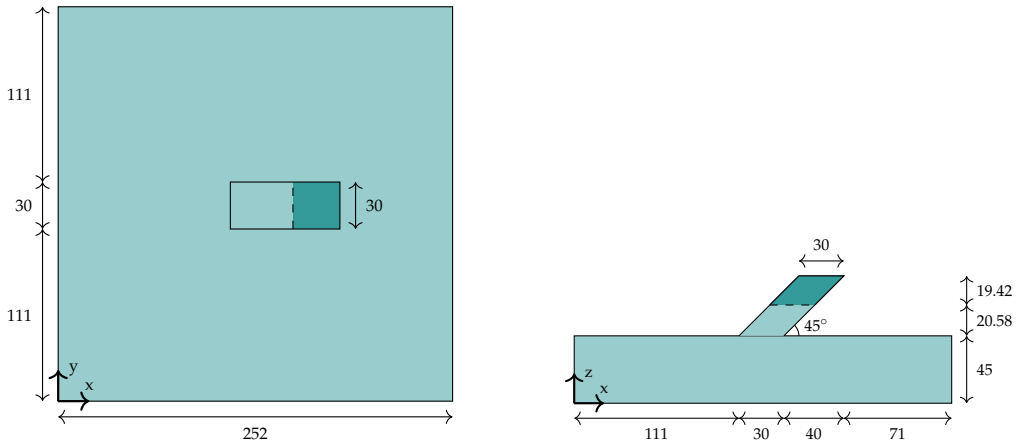



(a) VDA-P model.




(b) Close up of VDA-P model halfway through the simulation.

Figure 7.9: Example of Section 7.4.1. Comparison of the reduced VDA-P model against the HF model.




(a) Plane XY view.

(b) Plane XZ view.

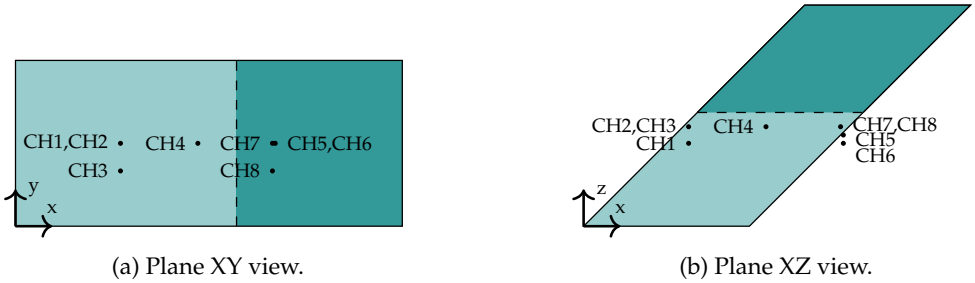
Figure 7.10: MCAM experiment. Base plate and printed specimen (mm). The simulated region is highlighted in dark teal.

between odd and even layers.

Regarding the scanning strategy, the laser travels along the y direction back and forth for each layer, as shown in Figure 7.12. Note that the number of hatches drawn does not correspond to the actual number of hatches, which is a much higher value, according to the laser beam size.

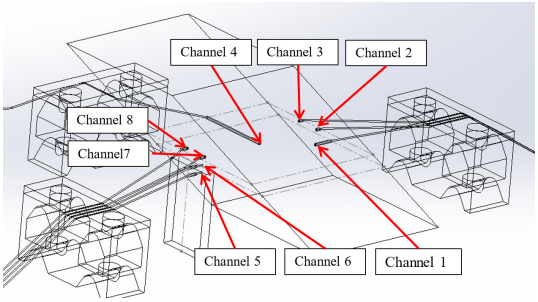
| Channel | (x,y,z) |
|---|---|
| CH1 | (129,125,60) |
| CH2 | (129,125,63) |
| CH3 | (129,120,63) |
| CH4 | (143,125,63) |
| CH5 | (157,125,61.5) |
| CH6 | (157,125,60) |
| CH7 | (156.5,125,63) |
| CH8 | (156.5,120,63) |

Table 7.6: Coordinates (mm) of the thermocouples with respect to the origin of coordinates in Figure 7.10.




(a) Plane XY view.

(b) Plane XZ view.




(c) CAD view.

Figure 7.11: Location of thermocouples at the specimen. The simulated region is highlighted in dark teal.

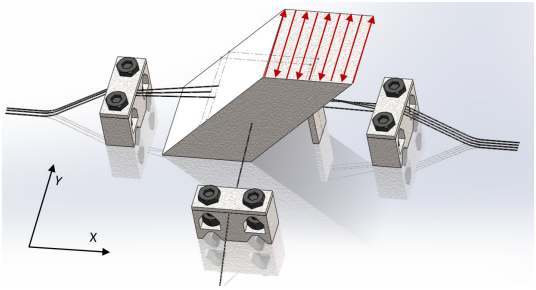



Figure 7.12: CAD view of thermocouple supports and orientative scanning path.

The printed samples are made of Ti6Al4V Titanium alloy. The temperature dependent properties of the bulk material, covering the range from room temperature to fusion temperature, are available in [55]. The base plate is made of CP Ti, a material with similar thermal properties as those of Ti64. Complementary experiments in [55]

| | | |
|---|---|---|
| Power input | 280 | [W] |
| Scanning speed | 1,200 | [mm/s] |
| Layer thickness | 30 | [µm] |
| Hatch distance | 140 | [µm] |
| Beam offset | 15 | [µm] |
| Recoat time (odd layers) | 14.3 | [s] |
| Recoat time (even layers) | 10.7 | [s] |

Table 7.7: MCAM experiment. Process parameters adopted by the EOS Machine.

estimated the relative density of Ti64 powder at around 54 % of the density of the bulk material at room temperature.

Figure 7.13 describes the experimental data gathered at the eight thermocouple channels. During the printing of the first layers, the thermocouples are close to the laser and a sharp and highly oscillatory temperature build-up takes place. This trend stabilizes a little before the hundredth layer, when the thermocouples are far enough from the laser spot. Then, the temperature evolution enters a slowly-cooling quasi steady-state regime, until the process finishes and the temperature falls to room temperature.

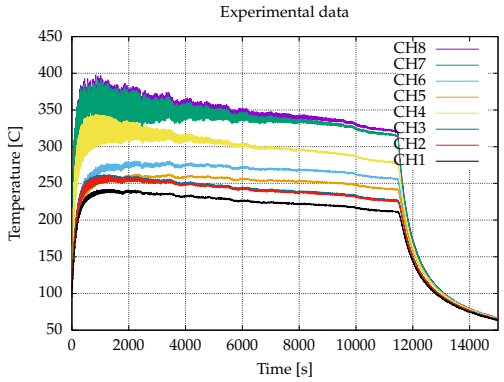



Figure 7.13: Temperature data gathered at the eight thermocouple channels. The evolution is initially ascendant and very oscillatory, but then it stabilizes and decreases slowly. After the printing, temperature drops to room temperature.

### Methodology

The numerical experiments were supported by a framework combining: (1) `FEMPAR` [19], an advanced object-oriented parallel FE library for large scale computational science and engineering, (2) `Dakota` [5], a suite of iterative mathematical and statistical methods for parameter estimation, sensitivity analysis and optimization of computational models, and (3) `TITANI` 7.8, a High Performance Computer (HPC) cluster at the Universitat Politècnica de Catalunya (Barcelona, Spain).

Such advanced computational framework, integrating FEA tools with scientific software for parameter exploration on a HPC platform, has not been previously observed

in the literature related to the numerical simulation of metal AM processes, but it is very convenient to carry out the verification and validation tasks at the scale of the experiment.

| | |
|---|---|
| Nodes | 5 DELL PowerEdge R630 |
| CPUs | 2 x Intel Xeon E52650L v3 (1.8GHz) |
| Cores | 12 cores per processor |
| Cache | 30 MB |
| RAM | 256 GB |
| Local disk | 2 x 250 GB SATA |

Table 7.8: Overview of the architecture of `TITANI`.

An overview of the procedure followed during the numerical tests is:

1. Implementation of the computational model described in Section 7.2 and the VDA model in Section 7.3 in `FEMPAR`.

2. Implementation of an interface to communicate `Dakota` with `FEMPAR`.

3. Design and implementation of a physically accurate reference (HF) heat transfer model.

4. Calibration and validation of the HF model against experimental data generated at the MCAM research centre.

5. Repeat (3) and (4) for two additional HTC-PP and VDA-PP reduced HPC models in `TITANI`. Comparison of the reduced-domain variants with the full-domain HF model.

According to this, three different numerical models were tested, the only difference among them being how heat loss through the powder bed is accounted for. In the HF model, the purpose is to maximize the accuracy of the model, by including the powder bed into the computational domain of analysis, to establish a numerical reference for the reduced models.

In the HTC-PP model, the powder-bed is excluded from the computational domain and heat loss through the powder bed is modelled with a constant heat conduction boundary condition at the solid-powder contact surfaces. The VDA-PP model adopts the same hypotheses of the HTC-PP model, except for the computation of the HTC at the solid-powder interfaces, which is derived from Equation (7.10), considering a single quadratic element (cf. (1E-Q2) in Table 7.1) to solve the VDA 1D heat conduction problem.

The calibration is done w.r.t. the measurements in thermocouple channels CH2 and CH4, separated 14 mm horizontally. Only measurements during printing are considered; the cooling stage is simulated, but not calibrated. The numerical response is then compared with the experimental data of CH8, one of the furthest from CH2 and CH4,

as a way to validate the model. Remaining thermocouples are either close to CH2, CH4, CH8, or outside the sample.

An important feature is that both VDA-PP and HTC-PP models inherit all the simulation data (process parameters, material properties,...) from the HF model and only the relevant parameters of each model are estimated. Table 7.9 gathers all the simulation data from the three models and highlights the calibration parameters of the reduced models.

**Calibration of the powder-bed model**

The reference model that will be used later to assess the accuracy and performance of the VDA model must reproduce as closely as possible the physical process of metal deposition. Likewise, the size of the simulation must be chosen to enable a full sensitivity analysis and iterative parameter estimation in reasonable computational times.

A locally accurate simulation of the metal deposition process must include the powder-bed in the computational domain, the FE mesh must conform to the printed layers, the mesh size must be smaller than the laser beam spot size, and the scanning path must be tracked element by element.

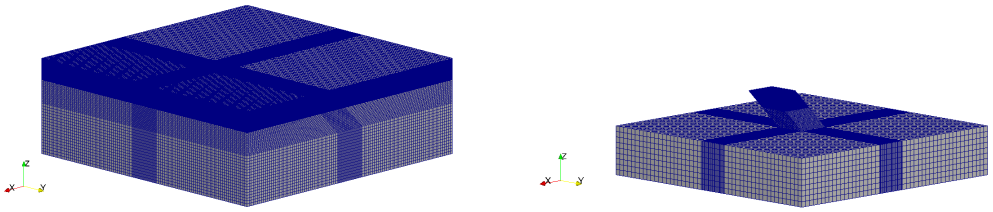
However, complying with these requirements is not always possible from the computational point of view. For instance, in this experiment, assuming a uniform mesh with element size $50 \times 50 \times 30$ [$\mu m^3$], a single layer of the specimen would be composed of 360,000 elements to be printed in 360,000 time steps.

Besides, the focus of this work is on the thermal analysis *at the part scale*, as commented in Section 7.1. This allows us to control the problem size and the number of time steps, by relaxing the previous discretisation requirements with suitable approximations.

The most relevant simplification in this work is the adoption of a layer-by-layer activation strategy, as in Section 7.4.1. Even though this assumption is computationally very appealing, it sacrifices the local thermal history of the response. However, it allows us to recover average thermal responses, as discussed in [54], and enables parametric exploration of the computational model in moderate times. In any case, regardless of the deposition sequence, the reference HF model is, by construction, more accurate than the reduced-domain VDA-PP and HTC-PP versions.

As a result of the previous considerations, the FE discretisation is a structured mesh of 9,565,788 hexahedral elements and 9,742,768 nodes. As observed in Figure 7.14(a), the FE mesh is a box containing the printed specimen, the building plate, and the powder bed. Supporting structures of the thermocouples are not included in the analysis, because they are far and small enough to have little influence in the results at the specimen.

A static mesh is employed throughout the simulation. Element size was determined with a convergence test and it varies, depending on the region of the model. Fine $1 \times 1 \times 0.03$ [$mm^3$] elements are prescribed at the simulated region, while larger mesh



(a) HF model. (b) VDA-PP and HTC-PP models.

Figure 7.14: FE meshes used in the analysis of the MCAM experiment. They conform to the printed layers, that is the reason for the element concentration in the z-direction.

sizes are specified elsewhere. Note that, as a result of the layer-averaging in time and using a uniform heat source distribution, the mesh size no longer needs to be smaller than the laser spot size.

The numerical simulation begins after placing the thermocouples and resuming the job. It continues with the deposition of the remaining 647 layers and finishes with the cooling of the whole ensemble. This amounts to almost 13,000 [s] of deposition process.

All simulation data is listed in Table 7.9. Some comments arise on the parameter values:

1. The heat absorption coefficient, estimated at 70 %, is the most sensitive parameter of the model, as also observed in [54].

2. The temperature-dependent values of the powder thermal conductivity in Table 7.11 were estimated using Equation 7.2.

3. Temperatures of air, powder and plate were measured in the chamber. They suffer variations throughout the process, but the numerical model is barely sensitive to them, with the exception of plate temperature. For this quantity, an estimated evolution law is assumed in Table 7.10, taking into account that the building plate is heated during the printing phase.

4. The dimensions of the numerical experiment (mesh size and number of time steps) can only be appropriately dealt with parallel computing techniques. Still, the calibration procedure was rather slow. For instance, using 20 CPUs of `TITANI`, the execution time of a single evaluation of the HF model was approximately 50 [h].

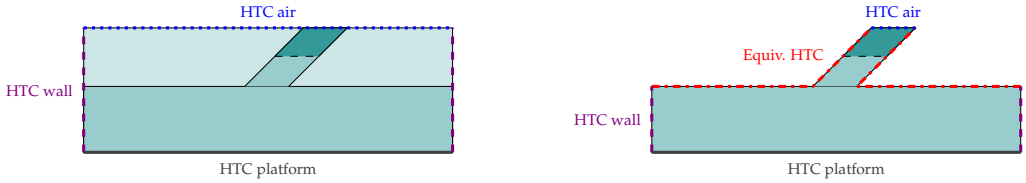
Figure 7.16(a) and Table 7.12 compare the numerical response with the experimental measurements. A very good agreement of predicted values and average rates in time can be observed during the whole printing process, both for the reference calibration channels, namely CH2 and CH4, and the validation channel CH8.

| Parameter | HF model | HTC-PP model | VDA-PP model | Units |
|---|---|---|---|---|
| **Process parameters** | | | | |
| Layer thickness | 30 | | | µm |
| Scanning speed | 5.6 | | | mm/s |
| Backward speed (odd layer) | 2.8 | | | mm/s |
| Backward speed (even layer) | 2.1 | | | mm/s |
| Laser power | 280 | | | W |
| Laser efficiency | 70 | | | % |
| **Material properties** | | | | |
| Bulk thermal properties | In [55] | | | - |
| Powder thermal properties | In Table 7.11 | - | In Table 7.11 | - |
| **Boundary conditions** | | | | |
| HTC air | 10 | | | $W/m^2C$ |
| Air temperature | 35 | | | C |
| HTC at chamber wall | 10 | | | $W/m^2C$ |
| Temperature at chamber wall | 93 | | | C |
| HTC at plate | 10 | | | $W/m^2C$ |
| Platform temperature | In Table 7.10 | | | - |
| Equivalent HTC solid-powder | - | 21 | - | $W/m^2C$ |
| Powder temperature | 93 | | | C |
| Initial temperature | 93 | | | C |
| **Virtual powder model data** | | | | |
| Thickness | - | - | 36 | mm |
| HTC solid-VDA | - | - | 4,150 | $W/m^2C$ |
| HTC VDA-powder | - | - | 1,000 | $W/m^2C$ |
| Volume factor | - | - | 1 | - |
| Surface factor | - | - | 1 | - |
| **Problem size** | | | | |
| Mesh size | 9,742,768 | 696,199 | | nodes |
| Number of layers | 647 | | | layers |
| Number of time steps | 1,434 | | | steps |
| Activation strategy | layer-by-layer | | | - |
| **Computational cost** | | | | |
| Number of CPUs | 20 | | | CPUs |
| Execution time | 50 | 3 | | h |

Table 7.9: Comparison of thermal models analysed in the MCAM experiment. Calibration parameters of the HTC-PP and VDA-PP models marked in blue. Boundary conditions are also described in Figure 7.15.

| Time [s] | Temperature [°C] |
|---|---|
| 0.0 | 93.0 |
| 11,470.0 | 93.0 |
| 12,000.0 | 35.0 |
| 15,000.0 | 35.0 |

Table 7.10: MCAM experiment. Estimated evolution of temperature at the lower surface of the building plate.



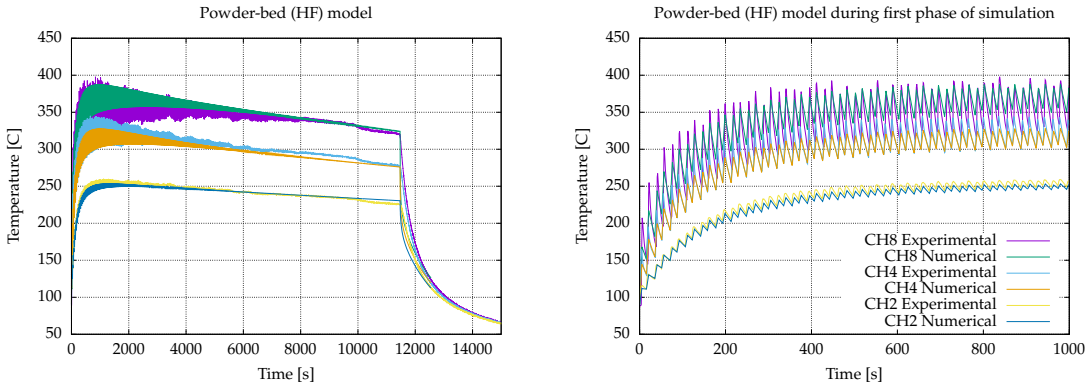

(a) HF model.

(b) VDA-PP and HTC-PP models.

Figure 7.15: Boundary conditions of MCAM experiment.

| Temperature [°C] | Conductivity [W/mC] |
|---|---|
| 20.0 | 0.288 |
| 200.0 | 0.407 |
| 300.0 | 0.466 |
| 400.0 | 0.520 |
| 500.0 | 0.573 |
| 600.0 | 0.630 |
| 700.0 | 0.684 |
| 800.0 | 0.746 |
| 900.0 | 0.808 |
| 1,100.0 | 0.904 |
| 1,227.0 | 0.976 |
| 1,500.0 | 1.115 |
| 1,600.0 | 1.168 |
| 1,660.0 | 1.252 |

Table 7.11: Temperature-dependent thermal conductivity of the Ti64 powder, according to Equation (7.2).




(a) The entire sintering process.

(b) The first phase after resuming the build job.

Figure 7.16: MCAM experiment. Numerical results of the HF model. Close agreement is observed for both calibration channels (CH2 and CH4) and the validation channel (CH8).

| Channel | MAE [°C] | MRE [%] |
|---|---|---|
| CH2 | 2.4 | 1.0 |
| CH4 | 6.3 | 2.1 |
| CH8 | 5.0 | 1.5 |

Table 7.12: MCAM experiment. Mean Absolute Error (MAE) and Mean Relative Error (MRE) of the temperature evolution during the printing phase between the experimental data and the calibrated HF model.

### Assessment of the HTC-PP model

Exclusion of the powder-bed from the computational domain leads to a significant reduction in the size of the problem. In this case, the FE mesh (Figure 7.14(b)) consists of

647,856 elements and 696,199 nodes. As a result, the HTC-PP model can be solved with significantly less computational resources and time than the previous model, i.e. in 3 [h] using 20 CPUs, and the parametric exploration is also much faster.

The HTC model is characterized by modelling heat transfer through the powder-bed with a constant heat conduction boundary condition on the solid-powder contact surfaces. For calibration against the experimental measurements, all the simulation data is the same as the one of the HF model and the only variable is the HTC of the aforementioned boundary condition.

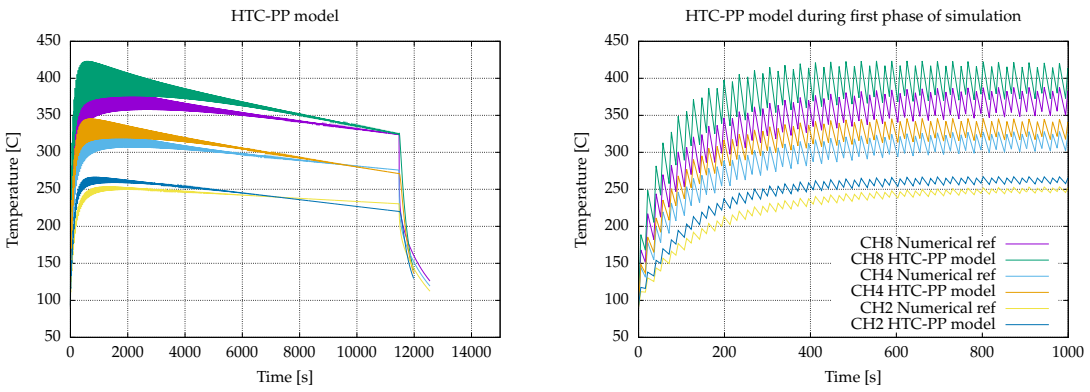
To interact with `Dakota`, the parameter estimation problem is formulated as an optimization problem of minimizing the least-squares error between data and variable-dependent response. The optimization problem is solved with a derivative-free local method (pattern search) until convergence.

To start the least-squares solver, the HTC can be initially estimated as

$$HTC(u) = \frac{k_{\text{pwd}}(u)}{s_{\text{pwd}}},$$

evaluated in the range of observed temperatures (200-400 [°C]). Here, $k_{\text{pwd}}$ is the conductivity of the powder and $s_{\text{pwd}}$ is the virtual loose powder thickness of the VDA. Using this rule of thumb, the HTC is initially set at 13-17 [W/m$^2$K] and converges to 21 [W/m$^2$K].

Concerning the numerical results, as seen in Figure 7.17 and Table 7.13, the simplification of the physics (exclusion of powder-bed + equivalent boundary condition) has obvious negative consequences in the accuracy. Besides, average rate of initial temperature build-up and cooling rate at the quasi-steady state regime are slightly different. In spite of this, the maximum error of the HTC-PP model with respect to the HF one is bounded by 15 %, even for the CH8 channel.




(a) The entire sintering process.

(b) The first phase after resuming the build job.

Figure 7.17: MCAM experiment. Numerical results of the HTC-PP model against the reference HF one. In spite of the physical simplifications, the maximum numerical error is still bounded by 15 % at all channels, but average temperature slopes in time are slightly different.

| | HF model | | experiments | |
|---|---|---|---|---|
| **Channel** | **MAE [°C]** | **MRE [%]** | **MAE [°C]** | **MRE [%]** |
| CH2 | 6.7 | 2.8 | 4.5 | 1.9 |
| CH4 | 8.3 | 2.8 | 5.5 | 1.8 |
| CH8 | 15.0 | 4.3 | 18.8 | 5.4 |

Table 7.13: MCAM experiment. Mean Absolute Error (MAE) and Mean Relative Error (MRE) of the temperature evolution during the printing phase between the HTC-PP model and the reference HF or the measured data.

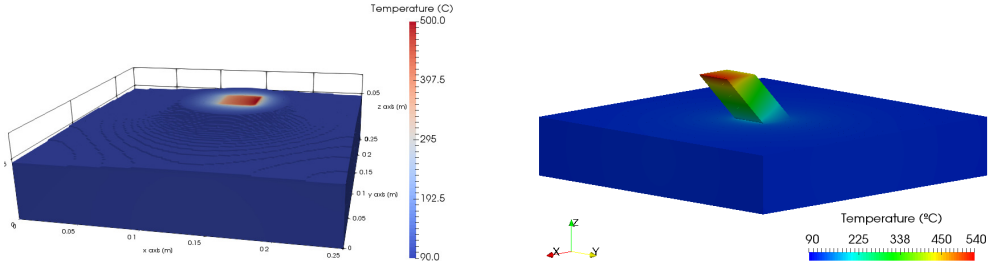
### Assessment of the VDA-PP model

As seen in the contour plot of temperatures in Figure 7.18(a), if the nodal values below the initial temperature of the powder are filtered, it is exposed that thermal gradients concentrate around the printed region.

Taking the FE mesh and the simulation data from the HTC-PP model, the `Dakota` least-squares solver was reformulated for the VDA-PP model. Some parameters of the VDA-PP model can be fixed to reduce the number of calibration variables. For instance, the volume and surface factors $F_{\text{volume}}$ and $F_{\text{surface}}$ can be set to 1, according to the shell-like shape of the VDA region.

As a result of this, the experimental calibration variables are now the thickness of the VDA $s_{\text{pwd}}$ and the HTC at the solid-VDA interface $h_{\text{s/p}}$. The HTC at the VDA-powder interface $h_{\text{p/p}}$ is also ruled out, after detecting low sensitivity to this parameter. However, its value should be large enough to have temperatures close to the initial temperature of the powder at the virtual boundary.

As for the initial approximations, from Figure 7.18(a) the VDA thickness is set to 30 [mm], whereas the HTC solid-VDA is set to 3,000-4,000 [W/m$^2$K], which is in the range of typical values of HTC for metal-sand contact surfaces in sand casting.



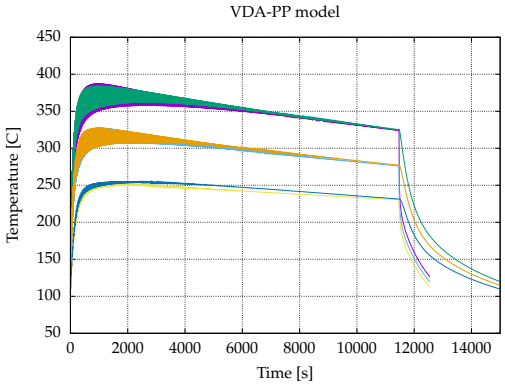

(a) HF model after filtering nodal values below the initial temperature of the powder. This allows one to approximate the thickness of the virtual powder.

(b) VDA-PP model.

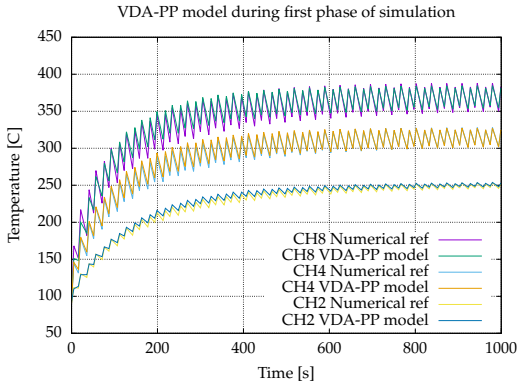
Figure 7.18: MCAM experiment. Contour plots of temperatures of the HF and VDA-PP models, represented at different time steps.

The numerical results in Figure 7.19 and Table 7.14 show that the VDA-PP model is successfully able to recover the accuracy of the HF model, at the same reduced computational cost of the HTC-PP model. In particular, average rates of temperature at

initial build-up and quasi-steady state regime match pretty well; the only mismatch is observed at the cooling phase, where the VDA-PP model overestimates the thermal inertia of the system.




(a) The entire sintering process.




(b) The first phase after resuming the build job.

Figure 7.19: MCAM experiment. Numerical results of the VDA-PP model against the reference HF one. VDA-PP clearly improves the HTC-PP, in the sense that it recovers the numerical response of the reference HF model (values and avarage rates in time).

| | **HF model** | | **experiments** | |
|---|---|---|---|---|
| **Channel** | **MAE [°C]** | **MRE [%]** | **MAE [°C]** | **MRE [%]** |
| CH2 | 3.6 | 1.5 | 3.6 | 1.5 |
| CH4 | 2.0 | 0.7 | 4.5 | 1.5 |
| CH8 | 2.0 | 0.6 | 6.5 | 1.9 |

Table 7.14: MCAM experiment. Mean Absolute Error (MAE) and Mean Relative Error (MRE) of the temperature evolution during the printing phase between the VDA-PP model and the reference HF or the measured data.

## 7.5 Conclusions

This work introduces a new rationale for physics-based model reduction in the thermal FE analysis of metal AM by powder-bed fusion methods, the Virtual Domain Approximation (VDA). In view of the locality of the process, it is reasonable to exclude regions of low physical relevance from the domain of analysis to reduce the size of the problem. However, lack of experimental data hinder proper estimation of heat loss through the neglected regions and rather simple approximations have been considered so far. By contrast, the VDA is thought to integrate the physics being neglected to evaluate heat loss through an excluded region. Inspired by existing methods in casting solidification, it consists in replacing the 3D FE model at the low-relevance region (e.g. loose powder bed, building plate) with a 1D heat conduction problem. Following this, the 1D problem is discretised at convenience and reformulated as a temperature-dependent Robin-type boundary condition for the 3D model in the reduced domain.

Using this approach, reduced models obtained by either excluding the powder bed or the powder bed and the building plate were derived and confronted with (1) a reference complete model and (2) the same reduced model, but with a constant boundary condition, i.e. constant HTC and environment temperature. As observed in the numerical experiments, the computational benefit of meshing a smaller geometry (the simulation time is reduced one order of magnitude) is accompanied by increased accuracy with respect to (2). In fact, the new method mostly recovers the thermal response predicted by (1) with the same computational cost of (2). Hence, this domain reduction strategy arises as an alternative that strikes good balance between efficiency and accuracy.

Even using reduced model variants, experimental validation and industrial applications are still challenged by high computational cost and uncertainty of the material and process parameters. To deal with this issue, this work turns to a methodology unprecedented in the field: an HPC platform that brings together (1) `FEMPAR-AM`, a FE model for the simulation of AM processes, designed to efficiently exploit distributed-memory supercomputers; and (2) `Dakota`, a suite of iterative mathematical and statistical methods for parametric exploration of computational models. As shown in the MCAM experiment, this advanced framework enables fast and automatized sensitivity analysis and parameter estimation. Hence, this kind of synergy may be useful in metal AM, not only to address verification and validation, but also for practical optimization problems.

One remaining question is to study the method for builds with curved shapes. The VDA assumes unidimensional heat loss, normal to the discrete interface. This is optimal for flat surfaces, but for smooth ones, it may lead to an underestimation of heat loss that, in some cases, can be compensated with simple geometrical correction factors proposed in Section 7.3.

Another interesting line of work is to push ahead the VDA capabilities, by reducing the domain up to the last simulated layers. In this scenario, the domain is thought as moving upwards to track the growth of the geometry, while keeping its size controlled either with a fixed or error-based criterion.

# Chapter 8

# Conclusions and future work

## 8.1 Conclusions

This thesis focused on the design of a novel highly-scalable numerical FE framework for PDE problems posed on evolving geometries. We considered as target application metal additive manufacturing processes by powder-bed fusion technologies. We addressed, specifically, a fundamental requirement in AM simulations, which state-of-the-art numerical tools cannot properly satisfy: scalable resolution in space on complex evolving three-dimensional geometries with multiple phases and physics. In order to do this, we have pushed forward current knowledge in tree-based parallel adaptive meshes and unfitted FE methods. We have also provided a blueprint to parallelise FE solvers of PDEs in growing geometries. In parallel, we have used our new computational tools to study and experimentally validate model reduction strategies, by scan pass lumping or physics-based localisation, in the thermal FE analysis of AM-PBF processes. These early successes in application have shown the potential of the new framework to contribute accelerating the understanding of the physics governing AM processes and product design and certification in AM.

In the next paragraphs, we summarise our contributions in the scope of the objectives proposed in Chapter 1. We recall that each chapter in the body of the thesis contains their own detailed conclusions, since they correspond to a different objective and are self-contained, preserving the structure of a paper.

> **Objective 1 (O1)**
> Address *with mathematical rigour* the requirements parallel tree-based meshes must fulfil to lead to correct parallel generic finite element solvers atop them.

In Chapter 2, we considered a two-layered meshing approach: an inner light-weight layer encoding the forest-of-trees, handled by an external specialised mesh engine, and an outer representation of the adaptive mesh suitable for the implementation of generic adaptive FE spaces. We proposed a new approach to construct the outer layer from the inner one and a different wau to represent the outer mesh layer (see Algorithm 1). In contrast to previous works, design assumes that one deals with a subassembled data layout for the linear algebra data structures on the interface among subdomains (see Algorithms 2 and 3). We have *mathematically* proved the computational benefits from

enforcing the 2:1 $k$-balance constraint, i.e. ease of parallel implementation and high scalability, as well as $k$-balance and $s$-ghost cell set minimum requirements that lead to a correct parallel solver for a conforming FE formulation. Implementation in `FEMPAR` and subsequent strong-scaling analysis revealed that the new approach, in terms of performance, is competitive (and sometimes superior) to a different one in the scientific software `deal.II`.

> **Objective 2 (O2)**
> Formulate and analyse the (h-adaptive) aggregated finite element method on parallel tree-based meshes.

In Chapter 3, we have bridged, for the first time, highly-scalable parallel adaptive meshing and unfitted methods. We introduced, specifically, a novel distributed memory implementation of the CG aggregated unfitted finite element method on locally-adapted Cartesian forest-of-trees meshes. We considered a two-tier approach that generates first the hanging DOF constraints and then modifies them with the aggregation DOF constraints, such that it avoids cyclic constraint dependencies. The strategy is backed by satisfactory numerical analysis and requires minimum parallelisation effort, using standard functionality available in existing large-scale FE codes; by attaching a limited amount of ghost cells to a usual single ghost cell layer, we can solve without interprocessor communication all combined aggregation and hanging DOF constraints. We have numerically assessed optimal error-driven mesh $h$-adaptation capability, as in standard body-fitted FEM, robustness to cut location and parallel efficiency and scalability in large-scale FE applications, using a high-end AMG solver. Moreover, we have carried out a sensitivity analysis of the well-posedness threshold. It revealed that, while enforcing a minimum amount of aggregation is required to ensure robustness, excessive aggregation degrades conditioning and solver efficiency.

> **Objective 3 (O3)**
> Formulate and analyse the (h-adaptive) aggregated finite element method on $n$-interface elliptic problems.

In Chapter 4, we have considered the usual approach of weakly coupling nonmatching discretisations to model interface discontinuities. We have naturally extended our cell aggregation scheme to $n$-field problems, with no coupling between fields; other approaches have more rigid aggregation schemes that cannot be easily generalised to any type of geometry and multiple field problems. We have seen that AgFEM easily accommodates the usual structure of approximation in the form of cartesian product FE spaces. We have mathematically proved well-posedness and optimal approximation properties of a symmetric interior penalty method for unfitted interface elliptic BVPs. Robustness to cut location is ensured, by inheriting cut-independent estimates from AgFEM in unfitted boundaries, while robustness to material contrast is achieved, by using the same weighted average of body-fitted DG methods. Moreover, the method can be straightforwardly implemented in parallel from a single-field AgFEM code. Through extensive numerical experimentation we have verified the theoretical results

above: Robustness to cut location and material contrast and robustness, optimal $h$-adaptivity and scalability in parallel tree-based adaptive meshes.

> **Objective 4 (O4)**
> Formulate a *fully* parallel framework to deal with PDE problems posed on evolving geometries, grounded on generic unfitted FEs on tree-based adaptive meshes.

In Chapter 5, we have first designed a distributed-memory element-birth method. With respect to a standard FE simulation pipeline, it only requires two extra nearest-neighbour interprocessor communications. To model the evolution of the domain, we have introduced a robust and embarrassingly parallel search algorithm for rectangular (also parallelepiped) $h$-adaptive meshes, based on the hyperplane separation theorem. To optimise parallel performance, we have proposed a weighted dynamic load balance strategy that can be tuned to balance cells or DOFs, depending on the physical application. All the three ingredients above are combined to deal with PDE problems posed on evolving geometries, atop our unfitted FE framework on parallel tree-based adaptive meshes. In the numerical tests, we have shown that our approach yields unprecedented parallelism and scalability in the (thermal) simulation of metal AM-PBF methods. Moreover, we have assessed that load balance strategies should seek a compromise between balancing cells and DOFs, to yield significant computational savings.

> **Objective 5 (O5)**
> *Experimentally-supported* numerical assessment of (scan pass or layer) lumping methods in *industrial-scale* thermal FE models of metal AM by powder-bed fusion.

In Chapter 6, we have substantially improved existing numerical assessment, sensitivity analysis and contrast with physical experiments, especially, at build scales that approach the limits of current machines, of model reductions by agglomerating the scanning path in time, e.g. by lumping scan passes or layers. We have proposed in Table 7.9 guidelines to help engineers selecting the best lumping strategy to their simulation goals. We have also identified that, at large build scales, layer-by-layer or hatch-by-hatch simulation strategies suffice to predict temperature evolution in time at thermocouple resolution.

> **Objective 6 (O6)**
> Formulate and *experimentally* validate a new *physics-based* thermal contact model, accounting for thermal inertia and suitable for localising AM-PBF models.

In Chapter 7, we have introduced a new *physics-based* rationale, the so-called virtual domain approximation, to reduce (up to localise) the computational domain in thermal FE models of metal AM-PBF. It is inspired by the virtual mould approach in modelling of casting solidification and allows one to disregard low-gradient regions in the

analysis. In contrast to previous works, VDA considers neglected physics, in particular, thermal inertia, in the evaluation of heat loss across the excluded region. Heat loss across such regions is evaluated with a heat conduction boundary condition that is formulated with a parametrised closed form, derived from $p$-Lagrangian FEs discretisations of a 1D rod. As a result, the expression for heat loss can be straightforwardly evaluated (no linear system solution) and accuracy of the VDA model can be easily tuned. Through numerical tests, supported by physical experiments, we assessed that the VDA yields higher accuracy at same computational cost of (naive) constant HTC approaches.

We note that the work of **(O5)** was carried out at the very beginning of the main thesis developments, i.e. before having a parallel framework. Limited available computational capacity, restricted to sequential analyses, led us to adopt model simplifications (e.g. neglect powder-bed or rather coarse meshes) that had an impact in the accuracy of the results. In contrast, we worked with the parallel framework of **(O4)** to meet **(O6)**. In this case, not only those simplifications were no longer necessary, it was also possible to exploit automatic calibration (iterative optimisation) tools atop our FE model, provided by the `Dakota` library. In other words, if we compare the numerical models in Chapter 6 and Chapter 7, it is clearly exposed how, by efficiently using HPC resources, we were able to substantially increase the complexity of our models and the fidelity of our outcomes.

> **Objective 7 (O7)**
> Highly-scalable implementation of the novel framework in high-end large-scale FE codes, exploiting state-of-the-art parallel adaptive tree-based mesh engines and scalable iterative linear solvers.

We have implemented the numerical framework of **(O1-4)** in the large-scale FE open-source software project `FEMPAR`. Through Chapters 2 to 7, we have successfully carried out extensive numerical experimentation in the small clusters `TITANI` and `Acuario` and the top-tier Marenostrum IV. We have both performed strong and weak scaling analysis and the largest problem solved in this thesis with unfitted FEs and tree-based meshes considered 90 million unknowns in 2,150 processors. Finally, we have addressed essential implementation aspects to guide other future works, for instance, in Chapter 5.5.

## 8.2 Future work

This thesis could be a springboard for providing fast and reliable fully-resolved AM simulations. However, this milestone is still a very long way ahead of us. Our numerical framework has several important shortfalls, that can only be addressed with research at the leading edge of numerical methods and scientific computing. We review next the main lines of research to tackle our current limitations.

- **Space-time $hp$-adaptive discretisations**

In this thesis, we have resorted to the ubiquitous method of lines, which segregates spatial and time discretisations, to approximate the transient thermal problem in metal AM. However, the approach is not a good choice for problems posed on evolving geometries; indeed, as it assumes that the domain is constant during the time increment, it leads to poor and nonsmooth *saw-tooth*-shaped space-time approximations of evolving domains. Moreover, it is unclear how to define initial boundary conditions on such *saw-tooth* boundaries. In practice, this approach may provoke spurious oscillations and localised high gradients on the artificial kinks [44]. On top of that, whereas spatial refinement is possible with our framework, the method of lines is not well-suited for variable time steps on different spatial locations, which prevents efficient and accurate simulation of multiscale space-time problems, such as metal AM. For instance, with the current framework for part-scale analysis, we can capture fast dynamics in the melt pool with spatial refinement, but we are forced to prescribe very small time step sizes on the whole part.

For this reason, we propose to extend our generic FE framework atop parallel tree-based meshes, such that it supports hexadecimal-tree 4D mesh representations. In this way, we would be able to solve all the above issues (in combination with unfitted FEs): no *saw-tooth* geometrical approximations, a natural way to impose initial conditions on the boundary and space-time adaptivity. We note that hexadecimal-tree mesh generators are still not available to the general public. On the other hand, we have so far restricted ourselves to solve the geometrical nonconformity, but we must also provide means to deal with nonconformity due to nonmatching functional spaces at cell boundaries (i.e. $p$-refinements) to support high-order unfitted FEs.

- **High-order $hp$-adaptive aggregated FEM**

  We have also restricted the study of our framework to low-order aggregated unfitted FE methods on first order approximations of the physical domain (i.e. at each cell, the cut is a straight line or plane). In the context of complex physics on evolving geometries and, in particular, metal AM processes, one may wish to exploit the smoothness properties of the evolving geometry, as well as the solution of PDEs by considering $p$-refinements, i.e. higher-order approximations. In such situations, a geometrical discretisation that keeps the smoothness of the physical domain is a must. As a result, we consider enhancing the current framework with methods to compute high-order geometrical approximations of the physical domain, as well as high-order submeshes for integration on cut cells. On the other hand, to open the framework to engineering applications, we also propose to explore a CAD/CAE integration paradigm.

  Apart from that, heuristic *a priori* tuning of spatial and temporal refinement is clearly not suitable for complex multiphysics and multiscale simulations. We

require an automatic process to define meshes (in space and time) based on *a posteriori* error estimation. Here, we have bypassed this requirement by considering the true error to drive the AMR, but for practical applications, we need to provide tools for functional and geometrical error-driven space-time adaptivity with unfitted methods, in the direction of works such as [39].

- **Application to powder-bed metal AM**

  Although we could already ascertain the benefits of our novel parallel framework in the application to metal AM-PBF processes, we have not exploited unfitted FE methods in our AM examples. As mentioned in Chapter 5, even for simple geometries, unfitted FE methods would easily eliminate the layer-conforming mesh constraint of body-fitted methods. Nonetheless, towards fully exploiting this technique for complex geometries, we still face a major hurdle in the experimental validation. Indeed, with the limited resources and measurement equipment in most laboratories, physical experiments on simple geometries are already a major endeavour. Furthermore, increasing the geometrical complexity of the experimental samples comes at little benefit for experimentalists, such as the ones we have collaborated with at MCAM; they are generally more concerned about understanding the physical process, rather than exploring geometrical freedom. For this reason, our experimental verification and validation tests have been limited to very simple printed shapes; it remains to see how the full framework performs in practical AM simulations.

  On the other hand, after solving the spatial refinement problem, sequentiality in time becomes the new major computational bottleneck to simulate long printing processes. However, it is to be swiftly avoided by means of space-time discretisations. Moreover, this would allow us to exploit much more efficiently vast computational resources. In particular, dynamic load balance to keep the computational load equilibrated among processors and mitigate parallel overheads, such as idling, would no longer be necessary, since one would solve the problem for all time steps in a single shot. We observe that to this end, one must also design novel space-time (nonlinear) highly-scalable solvers, since traditional segregation between linearisation and linear solver is not an optimal choice. Finally, we propose to consider more complex physics than heat transfer, such as mechanics at the part-scale or melt-pool solvers using the Navier-Stokes equations for the liquid phase, in order to include thermal convection and Marangoni effects.

# Bibliography


[1] *GAMG online documetnation*. `https://www.mcs.anl.gov/petsc/petsc-current/docs/manualpages/PC/PCGAMG.html`.

[2] *Intel MKL PARDISO - Parallel Direct Sparse Solver Interface*. `https://software.intel.com/en-us/articles/intel-mkl-pardiso`.

[3] *petsc MATSOLVERMKL PARDISO online documentation*. `https://www.mcs.anl.gov/petsc/petsc-current/docs/manualpages/Mat/MATSOLVERMKL_PARDISO.html`.

[4] https://hpc.cimne.upc.edu/, (n.d.). Accessed: 2019-03-02.

[5] B. ADAMS, L. BAUMAN, W. BOHNHOFF, K. DALBEY, M. EBEIDA, J. EDDY, M. ELDRED, P. HOUGH, K. HU, J. JAKEMAN, J. STEPHENS, L. SWILER, D. VIGIL, AND T. WILDEY, *Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.0 user's manual*, Sandia Technical Report SAND2014-4633, July 2014. Updated November 2016 (Version 6.5).

[6] M. AINSWORTH, J. T. J. T. ODEN, AND WILEY INTERSCIENCE (ONLINE SERVICE), *A posteriori error estimation in finite element analysis*, Wiley, 2000.

[7] V. AKCELIK, T. TU, J. URBANIC, J. BIELAK, G. BIROS, I. EPANOMERITAKIS, A. FERNANDEZ, O. GHATTAS, E. J. KIM, J. LOPEZ, AND D. O'HALLARON, *High Resolution Forward And Inverse Earthquake Modeling on Terascale Computers*, in Proceedings of the 2003 ACM/IEEE conference on Supercomputing - SC '03, New York, New York, USA, 2003, ACM Press, p. 52.

[8] A. ANCA, V. D. FACHINOTTI, G. ESCOBAR-PALAFOX, AND A. CARDONA, *Computational modelling of shaped metal deposition*, International Journal for Numerical Methods in Engineering, 85 (2011), pp. 84–106.

[9] C. ANNAVARAPU, M. HAUTEFEUILLE, AND J. E. DOLBOW, *A robust nitsche's formulation for interface problems*, Computer Methods in Applied Mechanics and Engineering, 225 (2012), pp. 44–54.

[10] P. M. AREIAS AND T. BELYTSCHKO, *A comment on the article "a finite element method for simulation of strong and weak discontinuities in solid mechanics" by a. hansbo and p. hansbo [comput. methods appl. mech. engrg. 193 (2004) 3523–3540]*,

Computer methods in applied mechanics and engineering, 9 (2006), pp. 1275–1276.

[11] D. N. ARNOLD, F. BREZZI, B. COCKBURN, AND L. D. MARINI, *Unified analysis of discontinuous galerkin methods for elliptic problems*, SIAM journal on numerical analysis, 39 (2002), pp. 1749–1779.

[12] U. AYACHIT, *The paraview guide: a parallel visualization application*, Kitware, Inc., 2015.

[13] I. BABUŠKA, *The finite element method with penalty*, Mathematics of computation, 27 (1973), pp. 221–228.

[14] S. BADIA, J. HAMPTON, AND J. PRINCIPE, *Embedded multilevel monte carlo for uncertainty quantification in random domains*, arXiv preprint arXiv:1911.11965, (2019).

[15] S. BADIA, A. F. MARTÍN, E. NEIVA, AND F. VERDUGO, *The aggregated unfitted finite element method on parallel tree-based adaptive meshes*, arXiv preprint arXiv:2006.05373, (2020).

[16] ——, *A generic finite element framework on parallel tree-based adaptive meshes*, SIAM Journal on Scientific Computing, (in press).

[17] S. BADIA, A. F. MARTÍN, AND J. PRINCIPE, *A highly scalable parallel implementation of balancing domain decomposition by constraints*, SIAM Journal on Scientific Computing, 36 (2014), pp. C190–C218.

[18] S. BADIA, A. F. MARTÍN, AND J. PRINCIPE, *Multilevel Balancing Domain Decomposition at Extreme Scales*, SIAM Journal on Scientific Computing, 38 (2016), pp. C22—-C52.

[19] S. BADIA, A. F. MARTÍN, AND J. PRINCIPE, *FEMPAR: An Object-Oriented Parallel Finite Element Framework*, Archives of Computational Methods in Engineering, 25 (2018), pp. 195–271.

[20] S. BADIA, A. F. MARTÍN, AND F. VERDUGO, *Mixed aggregated finite element methods for the unfitted discretization of the Stokes problem*, SIAM Journal on Scientific Computing, 40 (2018), pp. B1541–B1576.

[21] S. BADIA, F. NOBILE, AND C. VERGARA, *Fluid-structure partitioned procedures based on Robin transmission conditions*, Journal of Computational Physics, 227 (2008), pp. 7027–7051.

[22] S. BADIA AND M. OLM, *Space-time balancing domain decomposition*, SIAM Journal on Scientific Computing, 39 (2017), pp. C194–C213.

[23] S. BADIA AND F. VERDUGO, *Robust and scalable domain decomposition solvers for unfitted finite element methods*, Journal of Computational and Applied Mathematics, 344 (2018), pp. 740–759.

[24] S. BADIA, F. VERDUGO, AND A. F. MARTÍN, *The aggregated unfitted finite element method for elliptic problems*, Computer Methods in Applied Mechanics and Engineering, 336 (2018), pp. 533–553.

[25] S. BALAY, S. ABHYANKAR, M. F. ADAMS, J. BROWN, P. BRUNE, K. BUSCHELMAN, L. DALCIN, A. DENER, V. EIJKHOUT, W. D. GROPP, D. KAUSHIK, M. G. KNEPLEY, D. A. MAY, L. C. MCINNES, R. T. MILLS, T. MUNSON, K. RUPP, P. SANAN, B. F. SMITH, S. ZAMPINI, H. ZHANG, AND H. ZHANG, *PETSc Users Manual*. `http://www.mcs.anl.gov/petsc`, 2019.

[26] S. BALAY, S. ABHYANKAR, M. F. ADAMS, J. BROWN, P. BRUNE, K. BUSCHELMAN, L. DALCIN, V. EIJKHOUT, W. D. GROPP, D. KAUSHIK, M. G. KNEPLEY, L. C. MCINNES, K. RUPP, B. F. SMITH, S. ZAMPINI, H. ZHANG, AND H. ZHANG, *PETSc Users Manual*, Tech. Report ANL-95/11 - Revision 3.7, Argonne National Laboratory, 2016.

[27] 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 (2012), pp. 14:1–14:28.

[28] W. BANGERTH, R. HARTMANN, AND G. KANSCHAT, *deal.II—A general-purpose object-oriented finite element library*, ACM Transactions on Mathematical Software, 33 (2007).

[29] R. BECKER, *Mesh adaptation for Dirichlet flow control via Nitsche's method*, Communications in Numerical Methods in Engineering, 18 (2002), pp. 669–680.

[30] T. BELYTSCHKO, N. MOËS, S. USUI, AND C. PARIMI, *Arbitrary discontinuities in finite elements*, International Journal for Numerical Methods in Engineering, 50 (2001), pp. 993–1013.

[31] T. L. BERGMAN, F. P. INCROPERA, D. P. DEWITT, AND A. S. LAVINE, *Fundamentals of heat and mass transfer*, John Wiley & Sons, 2011.

[32] S. BOYD AND L. VANDENBERGHE, *Convex optimization*, Cambridge university press, 2004.

[33] G. BUGEDA, M. CERVERA, AND G. LOMBERA, *Numerical prediction of temperature and density distributions in selective laser sintering processes*, Rapid Prototyping Journal, 5 (1999), pp. 21–26.

[34] E. BURMAN, M. CICUTTIN, G. DELAY, AND A. ERN, *An unfitted hybrid high-order method with cell agglomeration for elliptic interface problems*, (2019).

[35] E. BURMAN, S. CLAUS, P. HANSBO, M. G. LARSON, AND A. MASSING, *CutFEM: Discretizing Geometry and Partial Differential Equations*, International Journal for Numerical Methods in Engineering, 104 (2015), pp. 472–501.

[36] E. BURMAN, D. ELFVERSON, P. HANSBO, M. G. LARSON, AND K. LARSSON, *Shape optimization using the cut finite element method*, Computer Methods in Applied Mechanics and Engineering, 328 (2018), pp. 242–261.

[37] E. BURMAN, D. ELFVERSON, P. HANSBO, M. G. LARSON, AND K. LARSSON, *Hybridized cutfem for elliptic interface problems*, SIAM Journal on Scientific Computing, 41 (2019), pp. A3354–A3380.

[38] E. BURMAN, J. GUZMÁN, M. A. SÁNCHEZ, AND M. SARKIS, *Robust flux error estimation of an unfitted nitsche method for high-contrast interface problems*, IMA Journal of Numerical Analysis, 38 (2018), pp. 646–668.

[39] E. BURMAN, C. HE, AND M. G. LARSON, *A posteriori error estimates with boundary correction for a cut finite element method*, arXiv preprint arXiv:1906.00879, (2019).

[40] C. BURSTEDDE, O. GHATTAS, M. GURNIS, G. STADLER, EH TAN, T. TU, L. C. WILCOX, AND S. ZHONG, *Scalable adaptive mantle convection simulation on petascale supercomputers*, in 2008 SC - International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE, 2008, pp. 1–15.

[41] C. BURSTEDDE AND J. HOLKE, *A Tetrahedral Space-Filling Curve for Nonconforming Adaptive Meshes*, SIAM Journal on Scientific Computing, 38 (2016), pp. C471–C503.

[42] C. BURSTEDDE, J. HOLKE, AND T. ISAAC, *On the number of face-connected components of morton-type space-filling curves*, Foundations of Computational Mathematics, (2018).

[43] C. BURSTEDDE, L. C. WILCOX, AND O. GHATTAS, *p4est: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees*, SIAM Journal on Scientific Computing, 33 (2011), pp. 1103–1133.

[44] V. CALO, S. BADIA, AND J. PRINCIPE, *Algorithmic aspects of high-performance computing for mechanics and physics*, Journal of Computational and Applied Mathematics, 344 (2018), pp. 739–739.

[45] I. CAMPBELL, O. DIEGEL, J. KOWEN, AND T. WOHLERS, *Wohlers report 2018: 3D printing and additive manufacturing state of the industry: annual worldwide progress report*, Wohlers Associates, 2018.

[46] H. CARSLAW AND J. JAEGER, *Conduction of heat in solids*, Oxford Science Publications: Oxford, UK, 1959.

[47] J. CERVENÝ, V. DOBREV, AND T. KOLEV, *Nonconforming Mesh Refinement for High-Order Finite Elements*, SIAM Journal on Scientific Computing, 41 (2019), pp. C367–C392.

[48] M. CERVERA, C. AGELET DE SARACIBAR, AND M. CHIUMENTI, *Thermo-mechanical analysis of industrial solidification processes*, International Journal for Numerical Methods in Engineering, 46 (1999), pp. 1575–1591.

[49] M. CERVERA, C. AGELET DE SARACIBAR, AND M. CHIUMENTI, *Comet: Coupled mechanical and thermal analysis. data input manual*, Barcelona: International Center for Numerical Methods in Engineering (CIMNE), (2002).

[50] Z. CHEN AND J. ZOU, *Finite element methods and their convergence for elliptic and parabolic interface problems*, Numerische Mathematik, 79 (1998), pp. 175–202.

[51] M. CHIUMENTI, M. CERVERA, N. DIALAMI, B. WU, L. JINWEI, AND C. AGELET DE SARACIBAR, *Numerical modeling of the electron beam welding and its experimental validation*, Finite Elements in Analysis and Design, 121 (2016), pp. 118–133.

[52] M. CHIUMENTI, M. CERVERA, A. SALMI, C. A. DE SARACIBAR, N. DIALAMI, AND K. MATSUI, *Finite element modeling of multi-pass welding and shaped metal deposition processes*, Computer methods in applied mechanics and engineering, 199 (2010), pp. 2343–2359.

[53] M. CHIUMENTI, C. A. DE SARACIBAR, AND M. CERVERA, *On the numerical modeling of the thermomechanical contact for metal casting analysis*, Journal of Heat Transfer, 130 (2008), p. 061301.

[54] M. CHIUMENTI, X. LIN, M. CERVERA, W. LEI, Y. ZHENG, AND W. HUANG, *Numerical simulation and experimental calibration of additive manufacturing by blown powder technology. part i: thermal analysis*, Rapid Prototyping Journal, 23 (2017), pp. 448–463.

[55] M. CHIUMENTI, E. NEIVA, E. SALSI, M. CERVERA, S. BADIA, J. MOYA, Z. CHEN, C. LEE, AND C. DAVIES, *Numerical modelling and experimental validation in selective laser melting*, Additive Manufacturing, 18 (2017), pp. 171 – 185.

[56] E. CHOW, R. D. FALGOUT, J. J. HU, R. S. TUMINARO, AND U. M. YANG, *A survey of parallelization techniques for multigrid solvers*, in Parallel processing for scientific computing, SIAM, 2006, pp. 179–201.

[57] R. CODINA AND S. BADIA, *On the design of discontinuous galerkin methods for elliptic problems based on hybrid formulations*, Computer Methods in Applied Mechanics and Engineering, 263 (2013), pp. 158–168.

[58] K. D. COLE, J. V. BECK, A. HAJI-SHEIKH, AND B. LITKOUHI, *Heat conduction using Green's functions*, CRC Press, 2010.

[59] A. COLL, R. RIBÓ, M. PASENAU, E. ESCOLANO, J. PEREZ, A. MELENDO, A. MONROS, AND J. GÁRATE, *GiD v.13 User Manual*, 2016.

[60] K. DAI AND L. SHAW, *Distortion minimization of laser-processed components through control of laser scanning patterns*, Rapid Prototyping Journal, 8 (2002), pp. 270–276.

[61] J. A. DANTZIG AND J. W. WIESE, *Modeling of heat flow in sand castings: Part II. Applications of the boundary curvature method*, Metallurgical Transactions B, 16 (1985), pp. 203–209.

[62] F. DE PRENTER, C. V. VERHOOSEL, G. J. VAN ZWIETEN, AND E. H. VAN BRUMMELEN, *Condition number analysis and preconditioning of the finite cell method*, Computer Methods in Applied Mechanics and Engineering, 316 (2017), pp. 297–327.

[63] C. A. DE SARACIBAR, M. CERVERA, AND M. CHIUMENTI, *On the formulation of coupled thermoplastic problems with phase-change*, International journal of plasticity, 15 (1999), pp. 1–34.

[64] L. DEMKOWICZ, *Computing with hp-adaptive finite elements: Volume 1 one and two dimensional elliptic and maxwell problems*, Chapman and Hall/CRC, 2006.

[65] D. DENG AND H. MURAKAWA, *Numerical simulation of temperature field and residual stress in multi-pass welds in stainless steel pipe and comparison with experimental measurements*, Computational materials science, 37 (2006), pp. 269–277.

[66] D. DENG, H. MURAKAWA, AND W. LIANG, *Numerical simulation of welding distortion in large structures*, Computer methods in applied mechanics and engineering, 196 (2007), pp. 4613–4627.

[67] E. R. DENLINGER, M. GOUGE, J. IRWIN, AND P. MICHALERIS, *Thermomechanical model development and in situ experimental validation of the Laser Powder-Bed Fusion process*, Additive Manufacturing, 16 (2017), pp. 73–80.

[68] E. R. DENLINGER, J. C. HEIGEL, AND P. MICHALERIS, *Residual stress and distortion modeling of electron beam direct manufacturing Ti-6al-4v*, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 229 (2015), pp. 1803–1813.

[69] E. R. DENLINGER, V. JAGDALE, G. V. SRINIVASAN, T. EL-WARDANY, AND P. MICHALERIS, *Thermal modeling of Inconel 718 processed with powder bed fusion and experimental validation using in situ measurements*, Additive Manufacturing, 11 (2016), pp. 7–15.

[70] P. DÍEZ AND A. HUERTA, *A unified approach to remeshing strategies for finite element h-adaptivity*, Computer Methods in Applied Mechanics and Engineering, 176 (1999), pp. 215–229.

[71] C. R. DOHRMANN, *A Preconditioner for Substructuring Based on Constrained Energy Minimization*, SIAM Journal on Scientific Computing, 25 (2003), pp. 246–258.

[72] A. J. DUNBAR, E. R. DENLINGER, M. F. GOUGE, AND P. MICHALERIS, *Experimental validation of finite element modeling for laser powder bed fusion deformation*, Additive Manufacturing, 12, Part A (2016), pp. 108–120.

[73] D. EBERLY, *Dynamic collision detection using oriented bounding boxes*, Geometric Tools, Inc, (2002).

[74] D. ELFVERSON, M. G. LARSON, AND K. LARSSON, *CutIGA with basis function removal*, Advanced Modeling and Simulation in Engineering Sciences, 5 (2018), p. 6.

[75] M. ELHADDAD, N. ZANDER, T. BOG, L. KUDELA, S. KOLLMANNSBERGER, J. KIRSCHKE, T. BAUM, M. RUESS, AND E. RANK, *Multi-level hp-finite cell method for embedded interface problems with application in biomechanics*, International journal for numerical methods in biomedical engineering, 34 (2018), p. e2951.

[76] H. C. ELMAN, D. J. SILVESTER, AND A. J. WATHEN, *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*, Oxford University Press, USA, 2014.

[77] A. ERN AND J.-L. GUERMOND, *Theory and practice of finite elements*, vol. 159, Springer Science & Business Media, 2013.

[78] V. D. FACHINOTTI, A. A. ANCA, AND A. CARDONA, *Analytical solutions of the thermal field induced by moving double-ellipsoidal and double-elliptical heat sources in a semi-infinite body*, International Journal for Numerical Methods in Biomedical Engineering, 27 (2011), pp. 595–607.

[79] R. D. FALGOUT, S. FRIEDHOFF, T. V. KOLEV, S. P. MACLACHLAN, AND J. B. SCHRODER, *Parallel time integration with multigrid*, SIAM Journal on Scientific Computing, 36 (2014), pp. C635–C661.

[80] R. A. FINKEL AND J. L. BENTLEY, *Quad trees a data structure for retrieval on composite keys*, Acta Informatica, 4 (1974), pp. 1–9.

[81] M. J. GANDER, *50 years of time parallel time integration*, in Multiple Shooting and Time Domain Decomposition Methods, Springer, 2015, pp. 69–113.

[82] J. GOLDAK, A. CHAKRAVARTI, AND M. BIBBY, *A new finite element model for welding heat sources*, Metallurgical transactions B, 15 (1984), pp. 299–305.

[83] J. A. GOLDAK AND M. AKHLAGHI, *Computational welding mechanics*, Springer Science & Business Media, 2006.

[84] S. GOTTSCHALK, M. C. LIN, AND D. MANOCHA, *Obbtree: A hierarchical structure for rapid interference detection*, in Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM, 1996, pp. 171–180.

[85] S. GOTTSCHALK, D. MANOCHA, AND M. C. LIN, *Collision queries using oriented bounding boxes*, PhD thesis, University of North Carolina at Chapel Hill, 2000.

[86] C. GÜRKAN AND A. MASSING, *A stabilized cut discontinuous galerkin framework for elliptic boundary value and interface problems*, Computer Methods in Applied Mechanics and Engineering, 348 (2019), pp. 466–499.

[87] J. GUZMÁN, M. A. SÁNCHEZ, AND M. SARKIS, *A finite element method for high-contrast interface problems with error estimates independent of contrast*, Journal of Scientific Computing, 73 (2017), pp. 330–365.

[88] A. HANSBO AND P. HANSBO, *An unfitted finite element method, based on nitsche's method, for elliptic interface problems*, Computer methods in applied mechanics and engineering, 191 (2002), pp. 5537–5552.

[89] ——, *A finite element method for the simulation of strong and weak discontinuities in solid mechanics*, Computer methods in applied mechanics and engineering, 193 (2004), pp. 3523–3540.

[90] H. HAVERKORT AND F. VAN WALDERVEEN, *Locality and bounding-box quality of two-dimensional space-filling curves*, in European Symposium on Algorithms, Springer, 2008, pp. 515–527.

[91] M. A. HEROUX, E. T. PHIPPS, A. G. SALINGER, H. K. THORNQUIST, R. S. TUMINARO, J. M. WILLENBRING, A. WILLIAMS, K. S. STANLEY, R. A. BARTLETT, V. E. HOWLE, R. J. HOEKSTRA, J. J. HU, T. G. KOLDA, R. B. LEHOUCQ, K. R. LONG, AND R. P. PAWLOWSKI, *An overview of the Trilinos project*, ACM Transactions on Mathematical Software, 31 (2005), pp. 397–423.

[92] N. E. HODGE, R. M. FERENCZ, AND R. M. VIGNES, *Experimental comparison of residual stresses for a thermomechanical model for the simulation of selective laser melting*, Additive Manufacturing, 12, Part B (2016), pp. 159–168.

[93] J. HOLKE, *Scalable algorithms for parallel tree-based adaptive mesh refinement with general element types*, arXiv preprint arXiv:1803.04970, (2018).

[94] C. HONG, T. UMEDA, AND Y. KIMURA, *Numerical models for casting solidification: Part I. The coupling of the boundary element and finite difference methods for solidification problems*, Metallurgical Transactions B, 15 (1984), pp. 91–99.

[95] D. HU AND R. KOVACEVIC, *Modelling and measuring the thermal behaviour of the molten pool in closed-loop controlled laser-based additive manufacturing*, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 217 (2003), pp. 441–452.

[96] P. HUANG, H. WU, AND Y. XIAO, *An unfitted interface penalty finite element method for elliptic interface problems*, Computer Methods in Applied Mechanics and Engineering, 323 (2017), pp. 439–460.

[97] J. IRWIN AND P. MICHALERIS, *A Line Heat Input Model for Additive Manufacturing*, Journal of Manufacturing Science and Engineering, 138 (2016), pp. 111004–111004–9.

[98] T. ISAAC, C. BURSTEDDE, AND O. GHATTAS, *Low-Cost Parallel Algorithms for 2:1 Octree Balance*, in 2012 IEEE 26th International Parallel and Distributed Processing Symposium, IEEE, may 2012, pp. 426–437.

[99] T. ISAAC, C. BURSTEDDE, L. C. WILCOX, AND O. GHATTAS, *Recursive Algorithms for Distributed Forests of Octrees*, SIAM Journal on Scientific Computing, 37 (2015), pp. C497–C531.

[100] J. N. JOMO, F. D. PRENTER, M. ELHADDAD, D. D. ANGELLA, C. V. VERHOOSEL, S. KOLLMANNSBERGER, J. S. KIRSCHKE, E. H. V. BRUMMELEN, AND E. RANK, *Robust and parallel scalable iterative solutions for large-scale finite cell analyses*, Finite Elements in Analysis and Design, 163 (2019), pp. 14–30.

[101] G. KARYPIS AND V. KUMAR, *A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs*, SIAM Journal on Scientific Computing, 20 (1998), pp. 359–392.

[102] ——, *Parallel Multilevel series k-Way Partitioning Scheme for Irregular Graphs*, SIAM Review, 41 (1999), pp. 278–300.

[103] L. KAUFMAN AND H. BERNSTEIN, *Computer calculation of phase diagrams with special reference to refractory metals*, Academic Press New York, 1970.

[104] T. KELLER, G. LINDWALL, S. GHOSH, L. MA, B. M. LANE, F. ZHANG, U. R. KATTNER, E. A. LASS, J. C. HEIGEL, Y. IDELL, ET AL., *Application of finite element, phase-field, and calphad-based methods to additive manufacturing of ni-based superalloys*, Acta materialia, 139 (2017), pp. 244–253.

[105] D. W. KELLY, J. P. DE S. R. GAGO, O. C. ZIENKIEWICZ, AND I. BABUSKA, *A posteriori error analysis and adaptive processes in the finite element method: Part I—error analysis*, International Journal for Numerical Methods in Engineering, 19 (1983), pp. 1593–1619.

[106] A. KERGASSNER, J. MERGHEIM, AND P. STEINMANN, *Modeling of additively manufactured materials using gradient-enhanced crystal plasticity*, Computers & Mathematics with Applications, (2018).

[107] S. A. KHAIRALLAH, A. T. ANDERSON, A. RUBENCHIK, AND W. E. KING, *Laser powder-bed fusion additive manufacturing: Physics of complex melt flow and formation mechanisms of pores, spatter, and denudation zones*, Acta Materialia, 108 (2016), pp. 36–45.

[108] S. KOLLMANNSBERGER, A. ÖZCAN, M. CARRATURO, N. ZANDER, AND E. RANK, *A hierarchical computational model for moving thermal loads and phase changes with applications to selective laser melting*, Computers and Mathematics with Applications, 75 (2018), pp. 1483 – 1497.

[109] S. KOLOSSOV, E. BOILLAT, R. GLARDON, P. FISCHER, AND M. LOCHER, *3d FE simulation for temperature evolution in the selective laser sintering process*, International Journal of Machine Tools and Manufacture, 44 (2004), pp. 117–123.

[110] F. KUMMER, *Extended discontinuous Galerkin methods for two-phase flows: the spatial discretization*, International Journal for Numerical Methods in Engineering, 109 (2017), pp. 259–289.

[111] P. KÜS AND J. ŠÍSTEK, *Coupling parallel adaptive mesh refinement with a nonoverlapping domain decomposition solver*, Advances in Engineering Software, 110 (2017), pp. 34–54.

[112] M. LABUDOVIC, D. HU, AND R. KOVACEVIC, *A three dimensional model for direct laser metal powder deposition and rapid prototyping*, Journal of materials science, 38 (2003), pp. 35–49.

[113] P. LACKI AND K. ADAMUS, *Numerical simulation of the electron beam welding process*, Computers & Structures, 89 (2011), pp. 977–985.

[114] Y. LEE AND W. ZHANG, *Modeling of heat transfer, fluid flow and solidification microstructure of nickel-base superalloy fabricated by laser powder bed fusion*, Additive Manufacturing, 12 (2016), pp. 178–188.

[115] C. LEHRENFELD, *High order unfitted finite element methods on level set domains using isoparametric mappings*, Computer Methods in Applied Mechanics and Engineering, 300 (2016), pp. 716–733.

[116] C. LI, M. F. GOUGE, E. R. DENLINGER, J. E. IRWIN, AND P. MICHALERIS, *Estimation of part-to-powder heat losses as surface convection in laser powder bed fusion*, Additive Manufacturing, (2019).

[117] K. LI, N. M. ATALLAH, G. A. MAIN, AND G. SCOVAZZI, *The shifted interface method: A flexible approach to embedded interface computations*, International Journal for Numerical Methods in Engineering, (2019).

[118] L.-Y. LI AND P. BETTESS, *Notes on mesh optimal criteria in adaptive finite element computations*, Communications in numerical methods in engineering, 11 (1995), pp. 911–915.

[119] L.-Y. LI, P. BETTESS, J. BULL, T. BOND, AND I. APPLEGARTH, *Theoretical formulations for adaptive finite element computations*, Communications in Numerical Methods in Engineering, 11 (1995), pp. 857–868.

[120] Y. LI AND D. GU, *Thermal behavior during selective laser melting of commercially pure titanium powder: Numerical simulation and experimental study*, Additive Manufacturing, 1-4 (2014), pp. 99–109.

[121] Y. LIAN, S. LIN, W. YAN, W. K. LIU, AND G. J. WAGNER, *A parallelized three-dimensional cellular automaton model for grain growth during additive manufacturing*, Computational Mechanics, 61 (2018), pp. 543–558.

[122] L.-E. LINDGREN, *Computational welding mechanics*, Elsevier, 2014.

[123] L.-E. LINDGREN AND A. LUNDBÄCK, *Approaches in computational welding mechanics applied to additive manufacturing: Review and outlook*, Comptes Rendus Mécanique, 346 (2018), pp. 1033–1042.

[124] L.-E. LINDGREN, A. LUNDBÄCK, M. FISK, R. PEDERSON, AND J. ANDERSSON, *Simulation of additive manufacturing using coupled constitutive and microstructure models*, Additive Manufacturing, 12 (2016), pp. 144–158.

[125] J. LINDWALL, A. MALMELÖV, A. LUNDBÄCK, AND L.-E. LINDGREN, *Efficiency and accuracy in thermal simulation of powder bed fusion of bulk metallic glass*, JOM, (2018), pp. 1–6.

[126] X. LU, X. LIN, M. CHIUMENTI, M. CERVERA, J. LI, L. MA, L. WEI, Y. HU, AND W. HUANG, *Finite element analysis and experimental validation of the thermomechanical behavior in laser solid forming of ti-6al-4v*, Additive Manufacturing, 21 (2018), pp. 30–40.

[127] A. LUNDBÄCK AND L.-E. LINDGREN, *Modelling of metal deposition*, Finite Elements in Analysis and Design, 47 (2011), pp. 1169–1177.

[128] A. MALMELÖV, A. LUNDBÄCK, AND L.-E. LINDGREN, *History reduction by lumping for time-efficient simulation of additive manufacturing*, Metals, 10 (2020), p. 58.

[129] O. MARCO, R. SEVILLA, Y. ZHANG, J. J. RÓDENAS, AND M. TUR, *Exact 3D boundary representation in finite element analysis based on Cartesian grids independent*

*of the geometry*, International Journal for Numerical Methods in Engineering, 103 (2015), pp. 445–468.

[130] `https://www.bsc.es/marenostrum/marenostrum`. Accessed: 2020-22-04.

[131] S. MARIMUTHU, D. CLARK, J. ALLEN, A. KAMARA, P. MATIVENGA, L. LI, AND R. SCUDAMORE, *Finite element modelling of substrate thermal distortion in direct laser additive manufacture of an aero-engine component*, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 227 (2013), pp. 1987–1999.

[132] A. MASSING, M. G. LARSON, A. LOGG, AND M. E. ROGNES, *A Nitsche-based cut finite element method for a fluid-structure interaction problem*, Communications in Applied Mathematics and Computational Science, 10 (2015), pp. 97–120.

[133] D. MEAGHER, *Geometric modeling using octree encoding*, Computer Graphics and Image Processing, 19 (1982), pp. 129–147.

[134] A. MELENDO, A. COLL, M. PASENAU, E. ESCOLANO, AND A. MONROS, *Gid home*, 2016. [Online; accessed Dec-2018].

[135] J. M. MELENK AND I. BABUŠKA, *The partition of unity finite element method: basic theory and applications*, in Research Report/Seminar für Angewandte Mathematik, vol. 1996, Eidgenössische Technische Hochschule, Seminar für Angewandte Mathematik, 1996.

[136] P. MICHALERIS, *Modeling metal deposition in heat transfer analyses of additive manufacturing processes*, Finite Elements in Analysis and Design, 86 (2014), pp. 51–60.

[137] K. C. MILLS, *Recommended values of thermophysical properties for selected commercial alloys*, Woodhead Publishing, 2002.

[138] R. MITTAL AND G. IACCARINO, *Immersed Boundary Methods*, Annual Review of Fluid Mechanics, 37 (2005), pp. 239–261.

[139] G. M. MORTON, *A computer oriented geodetic data base and a new technique in file sequencing*, (1966).

[140] M. MOZAFFAR, E. NDIP-AGBOR, S. LIN, G. J. WAGNER, K. EHMANN, AND J. CAO, *Acceleration strategies for explicit finite element analysis of metal powder-based additive manufacturing processes using graphical processing units*, Computational Mechanics, (2019), pp. 1–16.

[141] L. NASTAC, *A monte carlo approach for simulation of heat flow in sand and metal mold castings (virtual mold modeling)*, Metallurgical and Materials Transactions B, 29 (1998), pp. 495–499.

[142] E. NEIVA AND S. BADIA, *Robust and scalable h-adaptive aggregated unfitted finite elements for interface elliptic problems*, arXiv preprint arXiv:2006.11042, (2020).

[143] E. NEIVA, S. BADIA, A. F. MARTÍN, AND M. CHIUMENTI, *A scalable parallel finite element framework for growing geometries. application to metal additive manufacturing*, International Journal for Numerical Methods in Engineering, 119 (2019), pp. 1098–1125.

[144] E. NEIVA, M. CHIUMENTI, M. CERVERA, E. SALSI, G. PISCOPO, S. BADIA, A. F. MARTÍN, Z. CHEN, C. LEE, AND C. DAVIES, *Numerical modelling of heat transfer and experimental validation in powder-bed fusion with the virtual domain approximation*, Finite Elements in Analysis and Design, 168 (2020), p. 103343.

[145] L. NGUYEN, S. STOTER, T. BAUM, J. KIRSCHKE, M. RUESS, Z. YOSIBASH, AND D. SCHILLINGER, *Phase-field boundary conditions for the voxel finite cell method: Surface-free stress analysis of CT-based bone structures*, International Journal for Numerical Methods in Biomedical Engineering, 33 (2017), p. e2880.

[146] J. NITSCHE, *Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind*, Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg, 36 (1971), pp. 9–15.

[147] M. OLM, S. BADIA, AND A. F. MARTÍN, *Simulation of High Temperature Superconductors and experimental validation*, Computer Physics Communications, 237 (2018), pp. 154–167.

[148] ——, *On a general implementation of $h$- and $p$-adaptive curl-conforming finite elements*, Advances in Engineering Software, 132 (2019), pp. 74–91.

[149] E. OÑATE AND G. BUGEDA, *A study of mesh optimality criteria in adaptive finite element analysis*, Engineering computations, 10 (1993), pp. 307–321.

[150] L. PARRY, I. ASHCROFT, AND R. WILDMAN, *Understanding the effect of laser scan strategy on residual stress in selective laser melting through thermo-mechanical simulation*, Additive Manufacturing, 12 (2016), pp. 1–15.

[151] N. PATIL, D. PAL, H. KHALID RAFI, K. ZENG, A. MORELAND, A. HICKS, D. BEELER, AND B. STUCKER, *A Generalized Feed Forward Dynamic Adaptive Mesh Refinement and Derefinement Finite Element Framework for Metal Laser Sintering—Part I: Formulation and Algorithm Development*, Journal of Manufacturing Science and Engineering, 137 (2015), p. 041001.

[152] R. B. PATIL AND V. YADAVA, *Finite element analysis of temperature distribution in single metallic powder layer during metal laser sintering*, International Journal of Machine Tools and Manufacture, 47 (2007), pp. 1069–1080.

[153] P. PEYRE, P. AUBRY, R. FABBRO, R. NEVEU, AND A. LONGUET, *Analytical and numerical modelling of the direct metal deposition laser process*, Journal of Physics D: Applied Physics, 41 (2008), p. 025403.

[154] P. PRABHAKAR, W. SAMES, R. DEHOFF, AND S. BABU, *Computational modeling of residual stress formation during the electron beam melting process for Inconel 718*, Additive Manufacturing, 7 (2015), pp. 83–91.

[155] M. RAHMAN, W. MAURER, W. ERNST, R. RAUCH, AND N. ENZINGER, *Calculation of hardness distribution in the haz of micro-alloyed steel*, Welding in the World, 58 (2014), pp. 763–770.

[156] RENISHAW(PLC), *Data sheets - additive manufacturing*, 2018. [Online; accessed Dec-2018].

[157] W. C. RHEINBOLDT AND C. K. MESZTENYI, *On a Data Structure for Adaptive Finite Element Mesh Refinements*, ACM Transactions on Mathematical Software (TOMS), 6 (1980), pp. 166–187.

[158] P. J. ROACHE, *Code verification by the method of manufactured solutions*, J. Fluids Eng., 124 (2002), pp. 4–10.

[159] I. A. ROBERTS, C. J. WANG, R. ESTERLEIN, M. STANFORD, AND D. J. MYNORS, *A three-dimensional finite element analysis of the temperature field during laser melting of metal powders in additive layer manufacturing*, International Journal of Machine Tools and Manufacture, 49 (2009), pp. 916–923.

[160] T. M. RODGERS, J. D. MADISON, AND V. TIKARE, *Simulation of metal additive manufacturing microstructures using kinetic Monte Carlo*, Computational Materials Science, 135 (2017), pp. 78–89.

[161] D. ROSENTHAL, *Mathematical theory of heat distribution during welding and cutting*, Welding journal, 20 (1941), pp. 220s–234s.

[162] J. RUDI, O. GHATTAS, A. C. I. MALOSSI, T. ISAAC, G. STADLER, M. GURNIS, P. W. J. STAAR, Y. INEICHEN, C. BEKAS, AND A. CURIONI, *An extreme-scale implicit solver for complex PDEs: highly heterogeneous flow in earth's mantle*, in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis on - SC '15, New York, New York, USA, 2015, ACM Press, pp. 1–12.

[163] M. RUESS, D. SCHILLINGER, A. I. OEZCAN, AND E. RANK, *Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries*, Computer Methods in Applied Mechanics and Engineering, 269 (2014), pp. 46–71.

[164] J. W. RUGE AND K. STÜBEN, *4. Algebraic Multigrid*, in Multigrid Methods, Society for Industrial and Applied Mathematics, 1987, pp. 73–130.

[165] E. SALSI, M. CHIUMENTI, AND M. CERVERA, *Modeling of microstructure evolution of ti6al4v for additive manufacturing*, Metals, 8 (2018), p. 633.

[166] H. SAUERLAND AND T. P. FRIES, *The extended finite element method for two-phase and free-surface flows: A systematic study*, Journal of Computational Physics, 230 (2011), pp. 3369–3390.

[167] D. SCHILLINGER AND M. RUESS, *The Finite Cell Method: A Review in the Context of Higher-Order Structural Analysis of CAD and Image-Based Geometric Models*, Archives of Computational Methods in Engineering, 22 (2015), pp. 391–455.

[168] W. J. SCHROEDER, B. LORENSEN, AND K. MARTIN, *The visualization toolkit: an object-oriented approach to 3D graphics*, Kitware, 2004.

[169] I. SETIEN, M. CHIUMENTI, S. VAN DER VEEN, M. SAN SEBASTIAN, F. GARCIANDÍA, AND A. ECHEVERRÍA, *Empirical methodology to determine inherent strains in additive manufacturing*, Computers & Mathematics with Applications, (2018).

[170] M. S. SHEPHARD, *Linear multipoint constraints applied via transformation as part of a direct stiffness assembly process*, International Journal for Numerical Methods in Engineering, 20 (1984), pp. 2107–2112.

[171] S. S. SIH AND J. W. BARLOW, *The prediction of the emissivity and thermal conductivity of powder beds*, Particulate Science and Technology, 22 (2004), pp. 427–440.

[172] J. SMITH, W. XIONG, J. CAO, AND W. K. LIU, *Thermodynamically consistent microstructure prediction of additively manufactured materials*, Computational mechanics, 57 (2016), pp. 359–370.

[173] B. SONG, S. DONG, H. LIAO, AND C. CODDET, *Process parameter selection for selective laser melting of Ti6al4v based on temperature distribution simulation and experimental sintering*, The International Journal of Advanced Manufacturing Technology, 61 (2012), pp. 967–974.

[174] G. STADLER, M. GURNIS, C. BURSTEDDE, L. C. WILCOX, L. ALISIC, AND O. GHATTAS, *The dynamics of plate tectonics and mantle flow: from local to global scales.*, Science (New York, N.Y.), 329 (2010), pp. 1033–8.

[175] J. C. STEUBEN, A. P. ILIOPOULOS, AND J. G. MICHOPOULOS, *Towards Multiscale Topology Optimization for Additively Manufactured Components Using Implicit Slicing*, in Volume 1: 37th Computers and Information in Engineering Conference, ASME, 2017, p. V001T02A024.

[176] N. SUKUMAR, D. L. CHOPP, N. MOËS, AND T. BELYTSCHKO, *Modeling holes and inclusions by level sets in the extended finite-element method*, Computer Methods in Applied Mechanics and Engineering, 190 (2001), pp. 6183–6200.

[177] N. SUKUMAR, N. MOËS, B. MORAN, AND T. BELYTSCHKO, *Extended finite element method for three-dimensional crack modelling*, International Journal for Numerical Methods in Engineering, 48 (2000), pp. 1549–1570.

[178] H. SUNDAR, R. S. SAMPATH, S. S. ADAVANI, C. DAVATZIKOS, AND G. BIROS, *Low-constant parallel algorithms for finite element simulations using linear octrees*, in Proceedings of the 2007 ACM/IEEE conference on Supercomputing - SC '07, New York, New York, USA, 2007, ACM Press, p. 1.

[179] H. SUNDAR, R. S. SAMPATH, AND G. BIROS, *Bottom-Up Construction and 2:1 Balance Refinement of Linear Octrees in Parallel*, SIAM Journal on Scientific Computing, 30 (2008), pp. 2675–2708.

[180] `https://github.com/fempar/fempar`. Accessed: 2020-22-04.

[181] `https://caminstech.upc.edu/ca/calculintensiu`. Accessed: 2020-22-04.

[182] Common Layer Interface (CLI) Version 2.0 File Description, (n.d.). Accessed: 2016-08-26.

[183] Y. TIAN, C. WANG, D. ZHU, AND Y. ZHOU, *Finite element modeling of electron beam welding of a large complex al alloy structure by parallel computations*, journal of materials processing technology, 199 (2008), pp. 41–48.

[184] T. TIANKAI TU, D. O'HALLARON, AND O. GHATTAS, *Scalable Parallel Octree Meshing for TeraScale Applications*, in ACM/IEEE SC 2005 Conference (SC'05), IEEE, 2005, pp. 4–4.

[185] A. TOSELLI AND O. B. WIDLUND, *Domain Decomposition Methods — Algorithms and Theory*, vol. 34 of Springer Series in Computational Mathematics, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[186] P. VANEK, M. BREZINA, AND J. MANDEL, *Convergence of algebraic multigrid based on smoothed aggregation*, Numerische Mathematik, 88 (2001), pp. 559–579.

[187] F. VERDUGO, A. F. MARTÍN, AND S. BADIA, *Distributed-memory parallelization of the aggregated unfitted finite element method*, Computer Methods in Applied Mechanics and Engineering, 357 (2019), p. 112583.

[188] L. C. WILCOX, G. STADLER, C. BURSTEDDE, AND O. GHATTAS, *A high-order discontinuous Galerkin method for wave propagation through coupled elastic–acoustic media*, Journal of Computational Physics, 229 (2010), pp. 9373–9396.

[189] T. WOHLERS, *Wohlers report 2017*, Wohlers Associates, Inc, 2017.

[190] B. XIAO AND Y. ZHANG, *Laser sintering of metal powders on top of sintered layers under multiple-line laser scanning*, Journal of Physics D: Applied Physics, 40 (2007), p. 6725.

[191] S. XUE AND J. W. BARLOW, *Models for the Prediction of the Thermal Conductivities of Powders*, in Solid Freeform Fabrication Symposium Proceedings, Center for Materials Science, University of Texas at Austin Austin, Tex., 1991, pp. 62–69.

[192] I. YADROITSAU, *Selective laser melting: Direct manufacturing of 3D-objects by selective laser melting of metal powders*, Lambert Academic Publishing, 2009.

[193] S. YAGI AND D. KUNII, *Studies on effective thermal conductivities in packed beds*, AIChE Journal, 3 (1957), pp. 373–381.

[194] W. YAN, S. LIN, O. L. KAFKA, Y. LIAN, C. YU, Z. LIU, J. YAN, S. WOLFF, H. WU, E. NDIP-AGBOR, M. MOZAFFAR, K. EHMANN, J. CAO, G. J. WAGNER, AND W. K. LIU, *Data-driven multi-scale multi-physics models to derive process–structure–property relationships for additive manufacturing*, Computational Mechanics, 61 (2018), pp. 521–541.

[195] W. YAN, Y. QIAN, W. GE, S. LIN, W. K. LIU, F. LIN, AND G. J. WAGNER, *Meso-scale modeling of multiple-layer fabrication process in selective electron beam melting: Inter-layer/track voids formation*, Materials and Design, 141 (2018), pp. 210 – 219.

[196] Y. P. YANG, M. JAMSHIDINIA, P. BOULWARE, AND S. M. KELLY, *Prediction of microstructure, residual stress, and deformation in laser powder bed fusion process*, Computational Mechanics, 61 (2018), pp. 599–615.

[197] J. YIN, H. ZHU, L. KE, W. LEI, C. DAI, AND D. ZUO, *Simulation of temperature distribution in single metallic powder layer for laser micro-sintering*, Computational Materials Science, 53 (2012), pp. 333–339.

[198] S. ZEKOVIC, R. DWIVEDI, AND R. KOVACEVIC, *Thermo-structural finite element analysis of direct laser metal deposited thin-walled structures*, in Proceedings SFF Symposium,. Austin, TX, 2005.

[199] D. Q. ZHANG, Q. Z. CAI, J. H. LIU, L. ZHANG, AND R. D. LI, *Select laser melting of W–Ni–Fe powders: simulation and experimental study*, The International Journal of Advanced Manufacturing Technology, 51 (2010), pp. 649–658.

[200] L. ZHANG, E. W. REUTZEL, AND P. MICHALERIS, *Finite element modeling discretization requirements for the laser forming process*, International Journal of Mechanical Sciences, 46 (2004), pp. 623–637.

[201] O. C. ZIENKIEWICZ AND J. ZHU, *The superconvergent patch recovery (spr) and adaptive finite element refinement*, Computer Methods in Applied Mechanics and Engineering, 101 (1992), pp. 207–224.