

A High-Order Discontinuous Galerkin Tool for the Simulation of Multiphase Problems

Application to Space Debris Ablation by Melting

Doctoral dissertation presented by
David Henneaux
in partial fulfillment of the requirements for the degree
of Doctor in Engineering Sciences from UCLouvain

Louvain-la-Neuve (Belgium), February 2024

Thesis committee

Prof. Philippe **Chatelain** (Co-supervisor, UCLouvain)
Prof. Thierry **Magin** (Co-supervisor, ULB)
Dr. Pierre **Schrooyen** (Advisor, Cenaero)
Prof. Paul **Fisette** (Chair, UCLouvain)
Prof. Juray **De Wilde** (Examiner, UCLouvain)
Prof. Jean-François **Remacle** (Examiner, UCLouvain)
Prof. Andrea **Beck** (Examiner, University of Stuttgart)
Dr. Florian **Kummer** (Examiner, Technische Universität Darmstadt)

Manuscript writing

Credits	`LaTeX Cookbook` written by Alex Povel, modified by the present author.
Compiled on	**January 17, 2024 8:51am Z**
	`https://github.com/henneauxd/my-latex-cookbook`
Engine	LuaHBTeX, Version 1.15.0 (TeX Live 2022)
LaTeX Version	LaTeX 2_ε (2022-11-01)
Glossary	`glossaries-extra` + `bib2gls` + `Zotero with Better BibTeX`
Bibliography	biblatex + biber
Generator	`latexmk`
Class	2022/10/12 v3.38 KOMA-Script

Open-source software used

`Python`	Pre- and post-processing
`Git`	Code versioning
`Paraview + GmshReader plug-in`	Rendering
`Inkscape`	Drawing
`Gmsh`	Meshing
`MAdLib`	Mesh management and adaptation
`Algoim`	Quadrature rules
`PHG`	Quadrature rules
`SU2`	CFD computations
`LAMMPS`	MDS computations
`VS Code`	IDE
`GNU/Linux`	

©David Henneaux 2023. All Rights Reserved.
Université catholique de Louvain, EPL-iMMC
Place Sainte-Barbe, 2
1348 Louvain-la-Neuve
Belgium

Thesis committee

Prof. Philippe **Chatelain** (Co-supervisor, UCLouvain)
Prof. Thierry **Magin** (Co-supervisor, ULB)
Dr. Pierre **Schrooyen** (Advisor, Cenaero)
Prof. Paul **Fisette** (Chair, UCLouvain)
Prof. Juray **De Wilde** (Examiner, UCLouvain)
Prof. Jean-François **Remacle** (Examiner, UCLouvain)
Prof. Andrea **Beck** (Examiner, University of Stuttgart)
Dr. Florian **Kummer** (Examiner, Technische Universität Darmstadt)

Funding

This research was financially supported by a VKI fellowship from August 2018 to January 2019, by a UCLouvain-FSR fellowship from January 2019 to October 2019 and by a fellowship from the Fund for Research Training in Industry and Agriculture (FRIA), grant no. 1.E.077.20F, provided by the Belgian Fund for Scientific Research (F.R.S.-FNRS) from October 2019 to September 2023. The ten weeks scientific stay at NASA Ames was supported by the Fédération Wallonie-Bruxelles (FWB) and by a VKI-AA Research Travel Grant.

Computational resources have been provided by the Consortium des équipements de Calcul Intensif (CECI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S-FNRS) under grant no. 2.5020.11.







Rest of this page intentionally left blank.

Foreword

Acknowledgements

I've always loved math classes in primary and secondary school. When the time came to choose my studies, I chose engineering, more because "when you like math, engineering is for you" than because of the amazing scientific and technological feats which are based on the concepts I would be lucky enough to discover during my studies. It turns out that it was the best decision I could have made, as I truly loved learning the many subjects that are the foundation of the engineering sciences (applied mathematics, physics, chemistry, computer science, etc.). I would like to start this section by acknowledging all the professors I have had during my entire studies, from the beginning of pre-primary school to the end of university. Having the chance to be trained and to learn is an incredibly precious gift, and teachers are at the heart of this essential transmission of knowledge.

Among the numerous subjects covered in engineering classes, I quickly fell in love with fluid mechanics. The main reason was, I think, the possibility to gain one million dollar if I could one day solve the Navier-Stokes existence and smoothness question, part of the Millenium Prize problems. I'm kidding of course, I will never be able to solve such problem. The true reason, which is nonetheless linked to the nature of this problem, is the complexity of this field which I've always found challenging and motivating. I also enjoyed the multi-disciplinary nature of fluid mechanics, which combines applied mathematics, physics and programming. At this stage, I would like to thank specifically all the UCLouvain's teachers who passed on their passion to me about fluid mechanics. Profs. Grégoire Winckelmans and Philippe Chatelain have played a particularly important role by giving me the opportunity to use to their imposing VPM Computational Fluid Dynamics code during my master thesis. I would also like to say a special thank you to Denis-Gabriel for all the time he devoted to introducing me to the world of High-Performance Computing.

Spurred on by my new passion (that replaced the one of my adolescence, basketball), it was only natural that I should knock on the door of the von Karman Institute for Fluid Dynamics. I was lucky enough to be accepted into the Research Master program, which will remain an adventure engraved in my memory forever. It is kind of exceptional to have such a top-notch research institute in the middle of the small Belgium. What is even more incredible is that they offer grants to study in this unique environment. I'm really thankful for the opportunity I got and to all the VKI's professors who contributed to

deepen my passion for fluid mechanics and to introduce me to the scientific research. Thank you Bruno, Alessandro, Pierre and Thierry for supervising so well during my master thesis there.

The launch of my PhD thesis is inextricably linked with my adventure at the VKI and concerns one person in particular, namely Prof. Thierry Magin. You not only proposed me this fantastic opportunity of starting a PhD, you also convinced me to take one of the best decisions of my life. For this, I'll be forever grateful to you. And there are so many other reasons why I'm grateful to you. You're one of the kindest and most devoted people I know. You've always been there to explain me technical concepts, find reassuring words when I had doubts and highlight the value of what I was doing. I'm convinced that all the students who come under your tutelage grow from it.
I also owe a lot to my second supervisor, Prof. Philippe Chatelain, who accepted to be part of this UCLouvain-VKI collaborative project. Every discussion we've had together has left me this same feeling: the extent of your knowledge and your adaptability are astonishing. High-order methods and ablation are in principle not your research field, but you were still always able to give me invaluable advice and ideas for overcoming the most difficult situations. And doing all this with your unrivalled sense of humour added even more pleasure to our discussions.
The third member of my awesome supervising team was Dr. Pierre Schrooyen. Pierre, you're the person who has inspired me the most during my PhD. In a sense, I think that everything I've tried to accomplish, I did it to try to tend toward the achievements you've accumulated during your research career. I'm still a long way from reaching that goal, but you remain the lantern that lights up my research choices and gives me the necessary motivation. Above all, you're simply a super friendly and humble guy and I'm so glad I'll have the chance to continue collaborating with you in the future.

In direct connection with this manuscript, I would also like to express my sincere thanks to the other thesis committee members. First, my heartfelt thanks to the two "external" members, Prof. Andrea Beck and Dr. Florian Kummer. I asked you to be part of this committee because your work and that of your teams you lead inspired me throughout my thesis. The famous quote of Isaac Newton, "if I have seen further, it is by standing on the shoulders of giants", perfectly applies to your case. Thank you for your feedback and your insightful questions.
On the UCLouvain side, Prof. Juray De Wilde and Prof. Jean-François Remacle have also my utmost gratitude for agreeing to review the manuscript. Thank you Prof. De Wilde for your very valuable remarks about the physical modeling proposed in this work and putting it into perspectives with respect to the very complex physics occurring in flight. Thank you Prof. Remacle for your very relevant (and challenging) questions about some numerical aspects. Finally, I would like to thank Prof. Paul Fisette for agreeing to chair the committee with a lot of benevolence.

Besides my supervising team, I've also drawn a lot of inspiration from my VKI colleagues and friends. The presence around me of motivated and talented

people that you all are and the healthy emulation it generated acted as a catalyst for my progress. I genuinely appreciated all the coffee and lunch breaks with you, during which we shared our thoughts about the quality of the food (mostly with Italians I must say), the somewhat peculiar way Belgium is organized and how things are working here, or the (almost) high-tech buildings of VKI and all their amenities. I will also always remember the long parties we had, whether at VKI or in Brussels, and how we had a lot of fun chatting, dancing, drinking (and sometimes eating some plants ;-)). I would also like to give a special mention to all the students I interacted with, particularly the ones I supervised during their master theses. I learned a lot from our numerous discussions.
My UCLouvain partners, thank you so much for always welcoming me with open arms, despite my infrequent visits in Louvain-la-Neuve. You always teased me that I only came when drinks were organized. Well, it is kind of true. My real motivation behind this was that I knew that during those events, I would have the opportunity to talk with you. And I cherished everyone of of those moments. At UCLouvain, I also took part in an activity that was probably the most enriching of my PhD from a human point of view: the "Gender +" CORSCI working group. Collaborating with committed people sharing convictions that I value in order to address gender-related issues in the scientific community of our university has been a life-changing experience.

This thesis has also been fed by some important collaborations. I first would like to deeply acknowledge Cenaero for giving me this great opportunity of using and developing into their in-house `Argo` software. I'm well aware of how lucky I was to have access to this cutting-edge tool. I wouldn't have been able to achieve the results of this work without this code, nor would I have learned as much about the development of large industrial codes and the implementation of the discontinuous Galerkin method. My thanks also go to Nagi Mansour and Justin Haskins for hosting me during my research stay at Nasa Ames. It has been a huge honor to work under the supervision of both of you. I've learned so many things. I also would like to thank all the members of the team there who made me feel part of the group straight away, whether at Ames or outside during our trips and parties around.
I also owe a debt of gratitude to the Fonds de la Recherche Scientifique (FRS-FNRS) of Belgium that financially supported this work. I sincerely admire their commitment to supporting young researchers. I realized that their support enabled doctoral students like me to carry out their research in exceptional conditions, most probably among the best in the world.

Permettez-moi de basculer au français pour mettre en avant d'autres personnes qui ont été absolument essentielles au bon déroulement de cette thèse.
Je pense d'abord à tous mes amis, que ce soit ceux que je connais depuis mon enfance et adolescence ou bien ceux que j'ai eu la chance de rencontrer durant mon parcours académique. Au groupe d'Arlon, les NRV et toutes les personnes s'y rattachant, j'aimerais vous remercier pour tous les bons moments partagés. Derrière vos boutades à propos de mon sujet de thèse, j'ai toujours ressenti votre intérêt pour ce que je faisais. La simple idée de revenir en province le week-end

pour vous voir quand quelque chose s'organisait me galvanisait pour toute la semaine. Et les soirées "à l'ancienne" pour nous remémorer nos joyeuses années à l'unif avaient cette faculté de me vider la tête de toutes les préoccupations que je pouvais avoir au sujet de l'avancement de mon travail (et de me le faire ressentir jusqu'au lundi matin quand la semaine recommençait).
D'Arlon, ma ville natale, j'ai également d'autres amis extrêmement précieux à mes yeux ; des personnes que je connais depuis plus de 20 ans et avec qui j'ai partagé, et continuerai de le faire, tellement de choses. Vous aussi avez toujours manifesté votre enthousiasme pour mes recherches et je vous suis très reconnaissant pour toutes les discussions privilégiées que nous avons pu avoir. C'est une grande chance de pouvoir compter sur des amis comme vous depuis si longtemps.
Enfin, j'ai rencontré des personnes merveilleuses durant mes études d'ingénieur et dans mon kot à Louvain-la-Neuve. Nos soupers à nous refaire les histoires d'avant et à partager nos projets d'avenir ont égayé cette période de thèse et j'ai déjà hâte des prochaines occasions de vous revoir.

Ensuite, il y a ma famille qui n'a jamais cessé de m'encourager et de me soutenir dans tout ce que j'ai entrepris. La réalisation de cette thèse est en grande partie grâce à vous. J'aimerais en particulier remercier mes parents qui, par leurs marques d'affection et leurs gestes de soutien, m'ont insufflé en permanence cette énergie nécessaire tout au long de cette aventure. J'aimerais vous dire à quel point mes retours au foyer familial, ainsi que le séjour prolongé durant la période de Covid, ont été des sources de joie pour moi. Merci à mes frères pour leur indéfectible bienveillance et leurs authentiques encouragements. Thomas, nos débats de nature philosophique étaient un halo de lumière dans l'obscurité parfois impénétrable des équations que je m'échinais à résoudre. Lucas, ta vivacité et ta curiosité ont toujours rendu nos discussions sacrément intéressantes ; n'en déplaise aux autres personnes qui nous entouraient qui ne semblaient guère porter de l'intérêt à la programmation orienté-objet en Python ou à la formulation faible en éléments finis.
Je voudrais terminer par remercier du fond de mon coeur Victoria, la personne avec qui j'ai clairement partagé le plus de ressentis durant cette thèse. Peut-être encore plus que ta complaisance et tes encouragements inconditionnels, ta compréhension par rapport à mon emploi du temps souvent incompatible à une vie à deux équilibrée a été un élément central à mon épanouissement. Je voudrais donc te remercier pour les nombreux sacrifices consentis. Merci aussi d'avoir manifesté ton intérêt (presque) non feint pour les résultats que je te montrais, comme par exemple pour ce magnifique cas test de la bulle statique dont tu me parles encore (ou pas...).

Reader's note

The general organization of this manuscript is based on the following pattern. Each chapter, although linked to others in certain respects, can be read es-

sentially independently. The first section of each chapter (or sometimes even subsections inside the chapter) is devoted to a focalized state-of-the-art, which makes it possible to read the rest of the chapter that is about the development of models, methods or algorithms. Each chapter is closed by a summary of the contributions made by this work to the subject covered by the chapter.
If you find typos and/or errors, please don't hesitate to report them to me via henneauxdavid@gmail.com. The most up-to-date version of this dissertation can be found in my ResearchGate page: https://www.researchgate.net/profile/David-Henneaux/research.
I hope you enjoy reading this thesis an I thank you already for your interest.

David Henneaux
Brussels, January 2024

Rest of this page intentionally left blank.

Abstract

Keywords Compressible two-phase flows, phase transformations and mass transfer, melting ablation, sharp interface method, high-order extended discontinuous Galerkin, level-set method

Interest in the space debris proliferation has been growing substantially over the last decade because of the ever-increasing threat posed by those derelict spacecrafts on the sustainability of future space activities and the risks for on-ground casualties and the environment. A promising technology under development to mitigate this problem consists in designing the spacecrafts in such a way that they can safely re-enter the Earth's atmosphere at the end of their mission and be completely destroyed before reaching the ground. Setting up this new design paradigm necessitates advanced prediction capabilities about the degradation process of re-entering debris. However, the simplified models currently in use in engineering tools lump most of the physics, disabling the detailed comprehension of the involved phenomena, yet essential for the design for demise principle. The work carried out in this thesis aims to remediate this lack of knowledge by paving the way to a numerical framework able to produce simulations with an unprecedented level of fidelity.

This thesis is organized around three main pillars framing the structure of this manuscript. The first one concerns the modeling of some of the most important phenomena in the aerothermal ablation of melting objects. We start by formulating justified and consistent simplifying assumptions in order to focus our efforts on the identified phenomena that are the phase transformation of the condensed phase (melting-solidification) and the interactions between the gas and the liquid melt layer involving mass transfer and surface tension effects. From there, we lay down general jump conditions at the gas-liquid interface taking into account all the considered phenomena via suitable constitutive laws. We then obtain compressible volume-averaged equations with appropriate closure and equations of state to model the liquid-solid phases with phase transformation.

The models are implemented in the `Argo` multiphysics platform developed by the research center Cenaero which is based on a discontinuous Galerkin discretization of the conservation laws. The second and central pillar of this thesis is the extension of this available high-order framework to a compressible two-phase sharp interface solver able to preserve its accuracy while resolving the tremendous jumps across the gas-condensed phase interface. We start by improving the accuracy of a level-set method responsible for tracking the interface motion. We then adapt the spatial and temporal schemes such as to obtain

an unfitted discretization of the moving interface problem we are considering. This implies dealing with numerical challenges such as the cell-agglomeration procedure and the time integration over time-dependent domains. We finally devise a novel coupling strategy enforcing in a monolithic approach the jump conditions between viscous compressible phases. This is realized by altering the assembly of the interface term in the variational formulation such that the correct balance of fluxes across the interface is achieved.
The last part is about the verification of the implemented models and numerical methods. To do so, we select benchmark test cases assessing several aspects of the solver. We also build up a manufactured solution framework generating reference data satisfying prescribed jump conditions across static or moving interfaces. Finally, we conceive an experimental setup in a cold hypersonic environment which allows us to isolate and observe the ablation mechanisms targeted in this work.
The achievements of this work lay the foundation for extensibility of the framework to models with increasing level of complexity, toward an accurate description of the ablation by melting process.

Contents

Publications, presentations and supervision xvii

List of Figures xxi

List of Tables xxvii

List of Algorithms xxix

Nomenclature xxxi
Symbols . . . xxxi
Subscripts . . . xxxiii
Acronyms . . . xxxiv

I Research context

1 Introduction 3
1.1 Space debris and the design for demise strategy . . . 3
1.1.1 The space debris problem and its mitigation . . . 3
1.1.2 Space debris mitigation: the design for demise paradigm 8
1.1.3 Design for Demise: the need for high-fidelity predictive capabilities . . . 11
1.2 Physics of the aerothermochemical degradation process . . . 17
1.2.1 Hypersonic entry of space debris . . . 17
1.2.2 Shock layer region and high-temperature effects . . . 20
1.2.3 Various ablation processes depending on the material . . . 22
1.2.4 Boundary layer region and ablation by melting . . . 24
1.3 Numerical simulation of melting ablation . . . 31
1.3.1 Simplifying assumptions . . . 31
1.3.2 Characteristics of the simplified problem and choice of research paths . . . 33
1.4 Objectives and outline of the thesis . . . 37

II Physical modeling

2 Equations in the bulk of each phase 45
2.1 Fundamentals of continuum mechanics . . . 45
2.1.1 The spectrum of descriptions and the continuum hypothesis 45

2.1.2 Kinematical descriptions 48
2.1.3 Physical dynamics 56
2.2 The compressible Navier-Stokes equations and their closure . . 57
2.2.1 Derivation of conservation laws from first principles . . 57
2.2.2 Closure of the equations by constitutive models 61
2.3 Specific models for the liquid phase 66
2.3.1 Stiff compressible model 66
2.3.2 Thermodynamic and transport properties 70
2.3.3 Derivation of a thermodynamic consistent EOS 74
2.4 Summary of the equations implemented in this work 78

3 Modeling of the sharp gas-condensed phase interface 81
3.1 Formulations of multiphase flows 81
3.1.1 Mathematical formulations and level of description . . . 82
3.1.2 Review and choice of local instant compressible multi-phase models 84
3.2 Derivation of the interfacial jump conditions 89
3.2.1 Balance equations for generic quantities 90
3.2.2 Jump conditions for viscous compressible two-phase flows 96
3.3 Constitutive equations at the interface 100
3.3.1 Entropy production and mass transfer 100
3.3.2 Surface tension 104
3.4 Summary of the equations implemented in this work 105

4 Modeling of the diffuse liquid-solid phase interface 109
4.1 An overview of liquid-solid phase change modeling 110
4.1.1 Melting and glass transition processes 110
4.1.2 Range of models and associated numerical methods . . . 113
4.1.3 The choice of a macroscale diffuse interface model 115
4.2 The volume-averaged two-fluid equations 115
4.2.1 Volume averaging and derivation of the general model . 117
4.2.2 Closure of the equations and hierarchy of diffuse interface models 121
4.3 The equilibrium homogeneous mixture model for liquid-solid system with phase transformation 126
4.3.1 The mixture equations and the assumptions 126
4.3.2 Mass equation 131
4.3.3 Momentum equation 133
4.3.4 Energy equation 138
4.4 Summary of the equations implemented in this work 140

III Numerical methods
The Argo DG platform 149

5 High-order spatial discretization with the discontinuous Galerkin method 151
5.1 General overview of high-order methods for CFD 152
5.1.1 Weighted residual method: a unified framework to derive different classes of methods 152
5.1.2 Benefits, limitations and characteristics of DG schemes . 156
5.2 The discontinuous Galerkin method 160
5.2.1 Mesh and function space definitions 161
5.2.2 From the strong to the variational formulation of the Navier-Stokes equations 165
5.2.3 Ensuring stability of the discrete form through the definition of numerical fluxes 167
5.2.4 Assembly of the variational form 172
5.3 Implementation of an all-speed fluid solver 179
5.3.1 State-of-the-art overview and numerical challenges . . . 179
5.3.2 Versatile choice of working variables sets 183
5.3.3 Use of general equations of state 186
5.3.4 Treatment of low-Mach number flows by a density-based method 188

6 Spatial discretization of the level-set advection for interface motion 197
6.1 An overview of the methods to follow interfaces and the choice of the level-set method 198
6.1.1 State-of-the-art 198
6.1.2 The choice of the level-set method 200
6.2 The level-set method and its DG discretization 204
6.2.1 Governing equations 204
6.2.2 DG discretization 208
6.2.3 Temporal discretization 210
6.3 Reinitialization and computation of the closest points on the interface 211
6.3.1 Some background 211
6.3.2 Implementation 212
6.3.3 General algorithm and additional remarks 216
6.3.4 Verification test case 217
6.4 Velocity extension 219
6.4.1 Some background 219
6.4.2 Implementation 221
6.4.3 Verification test cases 223
6.5 The narrow band strategy 229
6.5.1 Some background 229
6.5.2 Implementation 229
6.6 Enforcement of continuous level-set iso-0 contour 232
6.6.1 Two distinct level-set functions 232

6.6.2 Four approaches to enforce continuity 232
6.7 Organization and summary of the various methods 238

7 Curvature filtering strategies for high-order sharp interface surface tension treatment 241
7.1 Overview of techniques to handle surface tension forces 242
7.1.1 Modeling and overview of methods to treat surface tension forces 242
7.1.2 Curvature filtering strategies in high-order FE-based methods 243
7.1.3 Objectives of the analysis 245
7.2 Description of the implemented filtering framework 246
7.2.1 Global flow chart 246
7.2.2 The artificial viscosity method 250
7.2.3 The patch-recovery method 250
7.2.4 The partition of unity method 252
7.3 Assessment of the various strategies 253

8 Sharp interface spatial discretization with the eXtended DG method 265
8.1 Overview of discretization approaches for discontinuous fields across interfaces and the choice of the XDG method 266
8.1.1 General overview about discontinuities handling 266
8.1.2 Overview of Finite Element type geometrically unfitted methods 269
8.1.3 Main challenges of unfitted methods 271
8.1.4 Advantages of high-order unfitted methods and the choice of the XDG method 274
8.2 The eXtended discontinuous Galerkin method 277
8.2.1 Cut mesh and extended function space definitions 277
8.2.2 Modifications of the discrete variational form 281
8.2.3 Enforcement of continuous level-set iso-0 surface 285
8.2.4 Integration over implicitly-defined surfaces 285
8.2.5 Cell agglomeration: treatment of small cut elements . . . 288
8.3 Implementation aspects of the XDG framework 291
8.3.1 Assembly of agglomerated residuals and jacobians . . . 291
8.3.2 Artificial viscosity technique in cut elements 303
8.3.3 Mesh-to-mesh projection 306

9 Enforcement of jump conditions between two compressible viscous phases 313
9.1 Overview of interface coupling approaches for compressible two-phase flows 314
9.1.1 Monolithic versus partitioned approaches 314
9.1.2 Enforcement of the jumps with a monolithic approach for two-phase flows 315

9.2 Enforcement of the jump conditions into the variational form . . 318
9.2.1 Continuous strong form of the equations 319
9.2.2 Interface convective numerical fluxes 320
9.2.3 Interface diffusive numerical fluxes 326
9.3 Resulting discretization implemented and its analysis 330
9.3.1 Resulting discretization 330
9.3.2 Jacobians of interface terms 332
9.3.3 Weak consistency 333

10 Temporal discretization, system solving and coupling between interface and fluid dynamics 339
10.1 Implicit time integration applied to standard DG discretization 340
10.1.1 Overview of time stepping schemes 340
10.1.2 Dual time-stepping and time preconditioning 345
10.1.3 Use of non-conservative sets of variables 351
10.2 Time integration with moving interfaces for the XDG method . 353
10.2.1 Overview of methods to handle evolving cut-mesh topologies 353
10.2.2 Splitting and moving interface approaches to integrate over time-varying domains 354
10.2.3 Cell-agglomeration to ensure constant cut topology over time levels 359
10.2.4 Assembly of the temporal residual for general sets of variables 364
10.2.5 General procedure 366
10.3 Coupling strategy between the fluid and level-set solvers 370
10.3.1 Overview of possible coupling strategies 370
10.3.2 Algorithm of the implemented segregated approach . . 371

IV Verification and experimental testing

11 Verification based on benchmark test cases 377
11.1 Overview of the test cases 378
11.2 Verification of the general EOS models on single-phase problems 379
11.2.1 One-dimensional Riemann 379
11.2.2 Steady flow around a cylinder 380
11.3 Spatial convergence study of the XDG solver on steady configurations 382
11.3.1 Compressible Couette 386
11.3.2 Manufactured inclined interface 387
11.3.3 Manufactured spinning bubble 393
11.4 Benchmark two-phase test cases in the incompressible regime . 402
11.4.1 Rising bubble 403
11.4.2 Rayleigh-Plesset oscillating bubble 404

11.5 Benchmark two-phase test cases in the compressible regime . . 405
11.5.1 Sod Riemann test case 407
11.5.2 Gas-liquid Riemann test case 408
11.5.3 Shock-bubble interaction 409
11.6 Benchmark two-phase test cases with surface tension 412
11.6.1 Oscillation of a deformed liquid iron droplet 414
11.6.2 Buoyancy-driven rise of a bubble 416
11.6.3 Standing capillary waves 419
11.7 Test cases with phase transformation 422
11.7.1 Mass transfer at the gas-liquid interface 422
11.7.2 Phase transformation at the liquid-solid interface 427

12 Development of a cold melting ablation experimental setup 437
12.1 Similarity analysis for replication of reentry conditions 438
12.1.1 Similarity concepts for reentry flows 438
12.1.2 Specific dimensional analysis for melting ablation 440
12.2 Setup of a cold hypersonic experiment with melting 442
12.2.1 Our strategy 442
12.2.2 Cold hypersonic VKI H3 facility 443
12.2.3 Sample design and manufacturing 448
12.3 Experimental measurements 451

V Conclusions

13 Conclusions and outlook 461
13.1 Summary and main achievements 461
13.2 Directions for future work and perspectives 469

A Expression of the fluxes Jacobians 481
A.1 Expression of the homogeneity tensor for compressible equations closed by general EOS 481
A.2 Construction of the Jacobian of the convective flux with respect to conservative variables 485

B Molecular Dynamics Simulation to estimate thermophysical properties of liquid metals 487
B.1 Theory of Molecular Dynamics 487
B.1.1 Molecular Modeling Approaches 487
B.1.2 Main elements of a MD simulation 489
B.1.3 Computation of thermophysical properties 497
B.2 Results of Molecular Dynamics Simulations with LAMMPS . . 503
B.2.1 Thermophysical properties of pure copper 503
B.2.2 Thermodynamic properties of an alloy: steel AISI 316L . 508

C A general methodology for generating manufactured solutions satisfying prescribed conditions **513**
C.1 Introduction . . . 513
C.2 Building manufactured solutions satisfying prescribed conditions 516
C.2.1 General manufacturing and verification procedure . . . 516
C.2.2 Limitations concerning the constraints definition and the MS free parameters computation . . . 519
C.3 Description of the manufacturing solution software . . . 520
C.4 Application examples . . . 521
C.4.1 Test case 1 - Unsteady inclined interface . . . 523
C.4.2 Test case 2 - Steady circular interface . . . 525
C.5 Conclusions and perspectives . . . 526

D Some semi-analytical reference solutions **529**
D.1 Exact Riemann solver for general EOS . . . 529
D.1.1 Determination of the intermediate states . . . 530
D.1.2 Computation of the global solution . . . 531
D.2 Rayleigh-Plesset problem with mass transfer . . . 532
D.2.1 Governing equations in the bulk . . . 532
D.2.2 Interface jump conditions . . . 533
D.2.3 ODE derivation for the bubble radius . . . 534

E Dimensional analysis for the melting ablation problem **537**

Bibliography **547**

Rest of this page intentionally left blank.

Publications, presentations and supervision

Published journal article

1. David Henneaux et al. (2023f). "High-order enforcement of jumps conditions between compressible viscous phases: An extended interior penalty discontinuous Galerkin method for sharp interface simulation." In: *Computer Methods in Applied Mechanics and Engineering* 415, p. 116215

Submissions under review

1. David Henneaux et al. (2023d). "Assessment of curvature filtering strategies for high-order sharp interface simulation of surface tension driven flows." Submitted to Flow, Turbulence and Combustion
2. David Henneaux et al. (2023b). "A General Methodology for Symbolically Generating Manufactured Solutions Satisfying Prescribed Conditions - Application to Two-phase Flows Equations." Submitted to Mathematics in Computer Science

Conference proceedings

1. David Henneaux et al. (2019a). "Towards a high-fidelity multiphase solver with application to space debris aerothermal ablation modeling." In: *AIAA Aviation 2019 Forum*, p. 2876
2. David Henneaux et al. (2020b). "Extended discontinuous Galerkin method for solving gas-liquid compressible flows with phase transition." In: *AIAA AVIATION 2020 FORUM*, p. 2971
3. David Henneaux et al. (2021b). "A High-Order Level-Set Method Coupled with an Extended Discontinuous Galerkin Method for Simulating Moving Interface Problems." In: *AIAA AVIATION 2021 FORUM*, p. 2742

Oral presentations at international conferences

1. David Henneaux et al. (2019b). "Towards a high-fidelity multiphase solver with application to space debris aerothermal ablation modeling."

Oral presentation at the AIAA Aviation 2019 Forum (Dallas)

2. David Henneaux et al. (2020c). "Extended discontinuous Galerkin method for solving gas-liquid compressible flows with phase transition." Oral presentation at the AIAA Aviation 2020 Forum (virtual)
3. David Henneaux et al. (2021d). "Sharp Interface Compressible Two-Phase Flow Solver Based On An Extended Discontinuous Galerkin Method." Oral presentation at the 14th World Congress on Computational Mechanics & ECCOMAS Congress (virtual)
4. David Henneaux et al. (2021c). "A High-Order Level-Set Method Coupled with an Extended Discontinuous Galerkin Method for Simulating Moving Interface Problems." Oral presentation at the AIAA AVIATION 2021 FORUM (virtual)
5. David Henneaux et al. (2022b). "A High-Order Extended Discontinuous Galerkin Method for Coupled Multi-Material Sharp Interface Problems." Oral presentation at the 15th World Congress on Computational Mechanics & 8th Asian Pacific Congress on Computational Mechanics (virtual)
6. David Henneaux et al. (2022c). "Enforcing Interface Coupling Conditions within a Discontinuous Galerkin Sharp Interface Method: Application to Viscous Compressible Gas-Liquid Flows." Oral presentation at the 10th International Conference on Numerical Methods for Multi-Material Fluid Flow (Zurich)
7. David Henneaux et al. (2023e). "General Methodology for Symbolically Generating Manufactured Solutions Satisfying Prescribed Conditions Application to Two Phase Flows." Oral presentation at the 6th International Conference on Numerical and Symbolic Computation (virtual)
8. David Henneaux et al. (2023c). "An All-Speed Extended Discontinuous Galerkin Sharp Interface Solver for Simulating Hydrodynamics of Melting Materials." Oral presentation at the 22nd IACM Computational Fluids Conference (Cannes)

Oral presentations and proceedings at symposium

1. David Henneaux et al. (2020a). "Towards a High-Fidelity Multiphase Solver with Application to Space Debris Aerothermal Ablation Modeling." In: *11th Symposium of VKI PhD Research*
2. David Henneaux et al. (2021a). "A high-order level-set method to follow the gas-liquid interface of melting materials." In: *12th Symposium of VKI PhD Research*
3. David Henneaux et al. (2022a). "Enforcing Interface Coupling Conditions within a High-Order Sharp Interface Method: Application to Viscous Compressible Gas-Liquid Flows." In: *13th Symposium of VKI PhD Research*
4. David Henneaux et al. (2023a). "Assessment of curvature filtering strategies for high-order sharp interface simulation of surface tension driven flows." In: *14th Symposium of VKI PhD Research*

Supervised master theses

1. Pedro Jorge (2019). *Development of oxidation models for space debris demise*
2. Aaricia Lamonte (2020). *Evaporation modeling based on thermogravimetric analysis*
3. Claudio Miccoli (2021). *Development of a low-Mach number compressible solver to simulate liquid metal flows for space debris ablation*
4. Ward Haegeman (2022). *Implementation of a curvature reconstruction algorithm for a high-order and sharp interface multiphase flow solver*
5. Andrea Perin (2023). *Wind Tunnel Testing of Space Debris Demise* (*operation and application*)

Rest of this page intentionally left blank.

List of Figures

1.1 Evolution of the number of objects in geocentric orbit by object class . . . 5
1.2 Evolution of the number of re-entering objects in each year by object type without human spaceflight . . . 5
1.3 Estimated number of space debris objects as function of the object size in Earth orbit . . . 5
1.4 Strategy behind the space debris mitigation measures . . . 9
1.5 Details of the main mitigation measures presented in Figure 1.4 9
1.6 Structure space debris re-entry prediction tools . . . 12
1.7 Main characteristics of a hypersonic re-entering object . . . 21
1.8 Chemical processes and temperature evolution in the shock layer region of a hypersonic re-entering object . . . 23
1.9 Boundary layer phenomena contributing to ablation by melting 30
1.10 Illustration of the thesis structure with the chapter numbers and their links . . . 40

2.1 Diagram of relations between systems and domains considered for the kinematical description . . . 49
2.2 Illustration of the thermal dilation of a liquid aluminum slab with linear temperature gradient . . . 78

3.1 Material volume cut by a moving interface for the derivation of the jump conditions . . . 91

4.1 Classification of solid-liquid phase change based on the interfacial morphology . . . 111
4.2 Classification of solid-liquid phase transition based on the change of properties . . . 113
4.3 Phase diagram of a binary isomorphous alloy . . . 132

5.1 Illustrative sketch of some elements involved in the DG discretization . . . 164

6.1 Embedding of a one-dimension interface as the zero iso-contour of the two-dimensional level-set function . . . 205
6.2 Embedding (extension) of the interface velocity to the two-dimensional level-set function . . . 205
6.3 Example of a level-set function not satisfying the signed distance property . . . 207

6.4 Illustration of the extended velocity and the associated source term . . . 209
6.5 Dual-circle level-set reinitialization test case . . . 219
6.6 Dual-circle reinitialization test case - Spatial convergence study 220
6.7 Rotating circle test case – Extended velocity field . . . 225
6.8 Rotating circle test case – Convergence study . . . 226
6.9 Channel flow test case – Iso-0 contours with extended velocity 227
6.10 Vortex-in-a-box test case – Iso-0 contours with extended velocity 228
6.11 Narrow band implementation . . . 231
6.12 Illustration of the narrow band concept . . . 231
6.13 Visual comparison of the different level-set continuity enforcement techniques . . . 237
6.14 Illustration of the level-set reinitialization and extension steps . 238

7.1 Elements for the curvature computation and filtering procedure 249
7.2 Interface geometries for the static geometrical analysis of the curvature computation . . . 254
7.3b Bar chart of the results of the three geometrical curvature accuracy test cases . . . 261
7.4 Sampling of the curvature along the level-set iso-0 of the circular droplet configuration . . . 262
7.5 Velocity and elevated pressure fields for the circular droplet test case . . . 262

8.1 Illustrative sketch of the main approaches to discretize a discontinuous solution . . . 266
8.2 Setting of the two-phase interface problem to be discretized . . 278
8.3 Geometrical discretization of the continuous domain in XDG . 280
8.4 Illustration of the way discontinuous fields are represented in XDGM . . . 282
8.5 Quadrature points on cut element . . . 287
8.6 Illustration of the cell agglomeration procedure on a triangular mesh . . . 289
8.7 Illustration of the cell agglomeration procedure on a quadrilateral mesh . . . 290
8.8 Illustrative example of the agglomeration flagging procedure . 293
8.9 Illustrative example of the assembly of the agglomerated residual 299
8.10 Example of meshes for the mesh-to-mesh projection test case . 308
8.11 Spatial convergence study of the mesh-to-mesh projection . . . 309

9.1 Waves structure of the approximate two-phase Riemann solvers 320
9.2 Computation of geometric factor of the penalization term for an agglomerated cells pair . . . 330

10.1 Illustrative example of the splitting and moving interface approaches for time integration of moving interface problem . . . 363

10.2 Illustrative example of the two approaches to discretize the space-time domain formed by the intersection between agglomerated mesh and the moving interface 364

11.1 Comparison of the numerical van der Waals Riemann test case solution with the analytical one 381
11.2 Comparison of the meshes used for the van der Waals steady cylinder flow between `Argo` and `SU2` 382
11.3 Comparison of the Mach number distribution between `Argo` and `SU2` on the van der Waals steady cylinder flow 383
11.4 Comparison of the velocity distribution between boundary conforming mesh and immersed boundary on the van der Waals steady cylinder flow 383
11.5 Steady compressible Couette test case setup 388
11.6 Steady compressible Couette test case grids 388
11.7 Steady inclined interface test case setup 389
11.8 Steady inclined interface test case grids 389
11.9 Steady spinning bubble test case setup 394
11.10 Steady spinning bubble test case structured grids 394
11.11 Steady spinning bubble test case unstructured grids 396
11.12 Steady spinning bubble test case convergence curves on unstructured grids 397
11.13 Steady flower shape test case setup 400
11.14 Rising bubble test case setup 403
11.15 Bubble shape, velocity and streamlines of the rising bubble test case 404
11.16 Bubble shape comparison at different non-dimensional times for the rising bubble test case 405
11.17 Rayleigh-Plesset oscillating bubble test case setup 406
11.18 Time evolution of the radius of the Rayleigh-Plesset oscillating bubble 406
11.19 Sod Riemann test case setup 408
11.20 Comparison of the numerical Sod Riemann test case solution with the analytical one 408
11.21 Gas-liquid Riemann test case setup 409
11.22 Comparison of the numerical gas-liquid Riemann test case solution with the analytical one 409
11.23 Illustration of the effects of artificial viscosity on the velocity profile of the gas-liquid Riemann test case 410
11.24 Shock-bubble interaction test case setup 410
11.25 Numerical Schlieren visualization of the shock-bubble interaction test case 411
11.26 Comparison of the numerical shock-bubble interaction test case results with experimental ones 411

11.27 Velocity and elevated pressure fields for the surface-tension driven oscillating droplet . . . 416
11.28 Temporal evolution of droplet radius for the surface-tension driven oscillating droplet . . . 417
11.29 Velocity fields for the surface-tension driven rising bubble test case . . . 418
11.30 Temporal evolution of the center of mass for the surface-tension driven rising bubble test case . . . 418
11.31 Temporal evolution of the degree of circularity for the surface-tension driven rising bubble test case . . . 419
11.32 Velocity and elevated pressure fields for the surface-tension driven standing waves test case . . . 421
11.33 Temporal evolution of the interface amplitude for the surface-tension driven standing waves test case . . . 421
11.34 Elevated temperature fields for the splitting and moving interface approaches on the 1D manufactured solution test case with mass transfer . . . 425
11.35 Temporal convergence study for the 1D manufactured solution test case with mass transfer . . . 426
11.36 Rayleigh-Plesset collapse bubble test case with mass transfer setup . . . 427
11.37 Velocity and elevated pressure fields of the Rayleigh-Plesset collapse bubble test case with mass transfer . . . 429
11.38 Temporal evolution of the normalized vapor bubble radius for the Rayleigh-Plesset collapse bubble test case with mass transfer 429
11.39 Stefan melting test case setup . . . 431
11.40 Temperature and elevated enthalpy fields for the Stefan melting problem . . . 431
11.41 Comparison between numerical and semi-analytical temperature profiles for the Stefan melting problem . . . 432
11.42 Velocity and temperature fields for the solidifying cylinder test case . . . 433
11.43 Elevated density fields in the gas and condensed phases for the solidifying cylinder test case . . . 434
11.44 Close-up view on the gas-condensed phase interface for the solidifying cylinder test case with velocity fields and volume fraction fields . . . 435

12.1 Schematic representation of VKI H3 hypersonic wind tunnel facility . . . 444
12.2 Unstructured mesh of the GCA5 sample geometry with symmetry boundary conditions . . . 446
12.3 Steady state results of the flow around the sample geometry with inlet conditions corresponding to the reservoir conditions of the H3 facility . . . 447

12.4 Two-dimensional cut along the axial direction of the rounded cone-shaped sample geometry 449
12.5 Three-dimensional cuts along the axial direction on the two main geometries of molds used to manufacture the samples tested in the H3 facility 450
12.6 Pre- and post-manipulations in the casting process of the samples to be tested in the H3 facility 451
12.7 Two-dimensional cuts with dimensions along the axial direction on the two main geometries of molds used to manufacture the samples tested in the H3 facility 452
12.8 Pictures of two samples mounted in the H3 wind tunnel, before and after the test 454
12.9 Pictures of all the samples just after the casting process and after having been tested in H3 455
12.10 Shadowgraph images of the GCA5 sample at the start, middle and end of the run 455
12.11 Shadowgraph images of the WC1 and GC2 samples at the start, middle and end of the run 456
12.12 Shadowgraph images of the WH4 and GH3 samples at the start, middle and end of the run 457

B.1 Initial configuration of the particles in a MD simulation 490
B.2 Schematic representation of periodic boundary conditions in MDS 491
B.3 Schematic representation of the electron sea model explaining metallic bonding 492
B.4 Geometry of a simple chain molecule, illustrating the definition of interatomic distance, bend angle and torsion angle 494
B.5 Constant energy hypersurface in phase space with a typical trajectory followed by a MD simulation 498
B.6 Density of copper as a function of temperature obtained with MDS and comparison with experimental results 504
B.7 Enthalpy of copper as a function of temperature obtained with MDS and comparison with experimental results 504
B.8 Isobaric specific heat of copper as a function of temperature obtained with MDS and comparison with experimental results 505
B.9 Volumetric thermal expansion coefficient of copper as a function of temperature obtained with MDS and comparison with experimental results 505
B.10 Bulk modulus of copper as a function of temperature obtained with MDS and comparison with experimental results 506
B.11 Dynamic viscosity of liquid copper as a function of temperature obtained with MDS and comparison with experimental results 506

B.12 Self-diffusion coefficient of liquid copper as a function of temperature obtained with MDS and comparison with empirical relation 507
B.13 Surface tension of liquid copper as a function of temperature obtained with MDS and comparison with experimental results 507
B.14 $p - v - T$ surface of pure copper built from a multivariate quadratic regression on the reference points computed via MDS 508
B.15 Initial configuration of the particles for the steel 316 alloy 509
B.16 Density of steel 316 as a function of temperature obtained with MDS and comparison with experimental results 510
B.17 Enthalpy of steel 316 as a function of temperature obtained with MDS and comparison with experimental results 510
B.18 Isobaric specific heat of steel 316 as a function of temperature obtained with MDS and comparison with experimental results 511
B.19 Volumetric thermal expansion coefficient of steel 316 as a function of temperature obtained with MDS and comparison with experimental results. Error bars representing 95 % confidence intervals are drawn 511

C.1 Modules constituting the pyManufSol numerical framework . . 522
C.2 Surface plot of the density field over the two phases for the inclined interface test case 524
C.3 One-dimensional profiles of the pressure field along the y-axis for four different x coordinates values 525
C.4 Velocity field over the two phases for the circular interface test case 526
C.5 Surface plot of the pressure field over the two phases for the circular interface test case 527
C.6 One-dimensional profile along the radial coordinate of the projection of the shear stress tensor along the normal-tangent direction with respect to the interface 527

List of Tables

2.1 Spectrum of physical modeling approaches 47

3.1 Characteristics of models and methods to simulate multiphase flows 85
3.2 Overview and physical meaning of the different parameters used in the jump conditions 106

4.1 Hierachy of two-fluid models 123

6.1 Errors for velocity extension test cases 227
6.2 Comparison of the different level-set continuity enforcement techniques based on several error measures 237

7.1 Best identified curvature computation combinations for the drop test case 256
7.2 Best identified curvature computation combinations for the starfish test case 257
7.3 Best identified curvature computation combinations for the peanut test case 258

9.1 Strategies for splitting the jump conditions between convective and diffusive fluxes 321

11.1 Numerical and physical parameters for the general EOS single-phase test cases 379
11.2 Parameters of the test cases used for the spatial grid convergence study 384
11.3 Discretization errors for the steady compressible Couette test case 386
11.4 Discretization errors for the steady inclined interface test case . 390
11.5 Interior penalty choice influence on the convergence rate 391
11.6 Scalar and tensorial interior penalty influence on the convergence rate 392
11.7 Choice of working variables influence on the convergence rate . 393
11.8 Discretization errors for the steady spinning bubble test case . . 395
11.9 Changes in parameters for the spinning bubble test case 398
11.10 Choice of working variables influence on the convergence rate for the steady spinning bubble test case 398
11.11 Discretization errors for the steady flower shape test case 401
11.12 Parameters for the unsteady incompressible two-phase test cases 402

11.13 Parameters for the unsteady compressible two-phase test cases 407
11.14 Numerical and physical parameters for the surface-tension driven test cases 413
11.15 Parameters for the unsteady evaporation two-phase test cases . 422
11.16 Parameters for the unsteady melting two-phase test cases 427

12.1 Test matrix for the preliminary melting ablation experiments . 445
12.2 Test matrix for the final melting ablation experiments 446
12.3 Some properties of the two materials used for the melting ablation experiments 449

C.1 Physical parameters selected for the test cases of Appendix C.4.1 and Appendix C.4.2. 522

List of Algorithms

1 Reinitialization by high-order computation of the closest point, $\mathbf{x}_{\Gamma,\mathrm{cp}}$, on the iso-0 of the level-set, φ, for each target mesh point, $\mathbf{x}_q$. The process is initialized with the closest point in the cloud of sampling points `cloudPts` that is found by a search tree `searchTree`. The reinitialized level-set field $\varphi^{\mathrm{reinit}}$ as well as the set of exact closest points are returned. 218

2 Extension of the velocity of the iso-0 of the level-set φ to each target mesh point $\mathbf{x}_q$. The velocity of a point in the domain is taken equal to the velocity at the closest point on the interface $\mathbf{x}_{\Gamma,\mathrm{cp}}$, determined from the provided fluid states $\mathbf{u}_k$ in each phase k. The extended velocity field over the domain $\mathbf{v}_{\mathrm{ext}}$ is returned. . 223

3 Main steps for the reinitialization and extension of the velocity based on a direct geometric approach. It takes as inputs the level-set field $\boldsymbol{\Phi}$ and the solution from the fluid solver $\underline{\boldsymbol{U}}^{\mathrm{fluid}}$ . . 238

4 Level-set solver for one time step taking as inputs the fluid solutions from the current time step $\underline{\boldsymbol{U}}^{\mathrm{fluid}}(t_n)$ and a history of level-set solutions over previous time steps and computing the level-set at the next time step $\boldsymbol{\Phi}(t_{n+1})$ based on a spatial DG discretization and explicit temporal integration with N_s stages. . 239

5 XDG fluid solver for one time step taking as inputs a history of topologies $\{\mathtt{Topo}^{t_{n+i}}\}_{i=-N_{\mathrm{step}}+1,\ldots,1}$, quadratures $\{\mathtt{Quad}^{t_{n+i}}\}_{i=-N_{\mathrm{step}}+1,\ldots,1}$ and solutions as well as information about the interface at the newt time step $\Gamma_{\mathrm{info}}^{t_{n+1}}$, and computing the solution at the next time step $\underline{\boldsymbol{U}}^{t_{n+1}}$ based on a spatial XDG discretization combined with cell-agglomeration and single-stage multi-step implicit temporal integration. 367

6 Coupling of level-set and XDG fluid solvers over one time step by an explicit first-order Lie operator splitting. 371

7 A simple molecular dynamics program 489

Rest of this page intentionally left blank.

Nomenclature

Symbols

Roman	Description	Unit
$\mathbb{B}_b$	Boundary with index b	–
$\mathcal{B}_b$	Spatial contribution of boundary face b to the global residual	–
C_p	Isobaric heat capacity	J/(kg K)
c	Speed of sound	m/s
C_v	Isochoric heat capacity	J/(kg K)
e,e_{int}	Internal/thermal energy	J/kg
$\mathcal{E}_e$	Spatial contribution of internal face e to the global residual	–
$\mathbb{E}_e$	Internal face with index e	–
e_{kin}	Kinetic energy	J/kg
E	Total energy	J/kg
H	Total enthalpy	J/kg
$\mathcal{I}_m$	Spatial contribution of phase interface in element m to the global residual	–
Ma	Mach number	-
$\mathbb{T}_m$	Element with index m	–
$\underline{\boldsymbol{M}}$	Mass matrix of the discretization	–
$\mathbb{I}_m$	Phase interface with index m	–
$\dot{m}_\Gamma$	Interface mass flux	kg/(m^2 s)
$\mathbf{n}$	Normal vector	-
$\mathcal{N}^{\text{sp}}$	Spatial contribution to the global weak form	–
$\mathcal{N}^{\text{tp}}$	Temporal contribution to the global weak form	–
$\mathcal{P}_p$	Function space of order p	–
p	Pressure or Interpolation order	Pa, -

$\underline{\boldsymbol{Q}}$	Cell-agglomeration coupling matrix	–
$\hat{\mathbb{T}}$	Reference parametric element	–
$\mathtt{B}$	Set of boundaries over the domain	–
$\mathtt{T}$	Set of elements over the domain	–
$\mathtt{E}$	Set of internal faces over the domain	–
$\mathtt{I}$	Set of phase interfaces over the domain	–
$\mathcal{T}_m$	Temporal contribution of element m to the global residual	–
T	Temperature	K
t	Time	s
$\mathbf{u}$	Approximate solution	–
$\underline{\boldsymbol{U}}$	Global set of unknown nodal degrees of freedom	–
$\mathbf{v}_{\text{ext}}$	Extnded velocity field for the level-set	m/s
$\mathcal{V}_m$	Spatial contribution of element m to the global residual	–
$\mathcal{W}$	Finite-dimensional functional space	–
$\mathbf{w}$	Test function	–
$\mathbf{x}$	Physical coordinates	-
ρ	Density	$\mathrm{kg/m^3}$

Greek	**Description**	**Unit**
$\mathbf{q}$	Heat flux vector	$\mathrm{W/m^2}$
Γ	Geometrical approximation of the interface	–
Ω	Geometrical approximation of the domain	–
α_p	Isobaric thermal expansion	1/K
β_T	Isothermal compressibility	1/Pa
$\gamma_{\mathtt{E}}, \gamma_{\mathtt{B}}, \gamma_{\mathtt{I}}$	Penalty parameters for Nitsche's term	–
γ	Specific heat ratio	–
$\underline{\boldsymbol{\delta}}$	Nitsche's penalization term	–
η	Entropy	$\mathrm{J/(kg\,K)}$
θ	Coefficient determining type of interior penalty method	–
κ	Interface curvature	1/m
λ	Thermal conductivity	$\mathrm{W/(m\,K)}$
μ	Dynamic viscosity	$\mathrm{kg/(m\,s)}$

$\boldsymbol{\xi}$	Parametric coordinates	-
σ_m	Mapping of physical element m to its reference parametric element	–
σ	Surface tension coefficient	N/m
$\underline{\boldsymbol{\tau}}$	Viscous stress tensor	Pa
φ	Level-set function	–
ω_q	Quadrature weights	–
$\boldsymbol{\phi}$	Shape functions	–

Other	**Description**	**Unit**
$\widetilde{\bullet_k}$	Favre average (mass weighted)	–
$\overline{\bullet_k}$	Phase average (volume weighted)	–
$\underline{\mathbf{X}}$	Tensor	–
$\mathbf{x}$	Vector	–
Δ_Γ	Jump across phase interface	–

Subscripts

Notation	**Description**
-,-,-	Tensor of quantities over all interpolation points of all elements
b	Refers to boundary face with index b
e	Refers to internal face with index e
m,-,-	Tensor of quantities over all interpolation points of element m
m	Refers to element or phase interface with index m
m(b)	Refers to boundary face with index b with a dependence to the solution over the adjacent element m

Notation	Description
m(e)	Refers to internal face with index e with a dependence to the solution over the adjacent element m
m,-,k,-	Tensor of quantities over all interpolation points of element m in phase k
m,n,k,f	Quantity f at element m, interpolation point n, in phase k

Acronyms

Notation	Description
CFD	Computational Fluid Dynamics
D4D	Design for Demise
DG	Discontinuous Galerkin
DSMC	Direct Simulation Monte Carlo
FD	Finite Differences
FE	Finite Element
FV	Finite Volume
MDS	Molecular Dynamics Simulation
XDG	eXtended Discontinuous Galerkin

Part I

Research context

1

Introduction

1.1 Space debris and the design for demise strategy 3
1.1.1 The space debris problem and its mitigation 3
1.1.2 Space debris mitigation: the design for demise paradigm 8
1.1.3 Design for Demise: the need for high-fidelity predictive capabilities 11
1.2 Physics of the aerothermochemical degradation process 17
1.2.1 Hypersonic entry of space debris 17
1.2.2 Shock layer region and high-temperature effects 20
1.2.3 Various ablation processes depending on the material . . 22
1.2.4 Boundary layer region and ablation by melting 24
1.3 Numerical simulation of melting ablation 31
1.3.1 Simplifying assumptions 31
1.3.2 Characteristics of the simplified problem and choice of research paths 33
1.4 Objectives and outline of the thesis 37

1.1 Space debris and the design for demise strategy

1.1.1 The space debris problem and its mitigation

Status of space environment

The ESA's annual space environment report presents the status of the space environment in various facets with the goal of estimating the impact of space

activities on the space debris status and quantifying the effect of applied mitigation measures. The figures and plots presented below are directly taken from T. E. S. D. O. ESA (2023). The taxonomy used in Figure 1.1 and Figure 1.2 is clarified here:

- *PL* = payload: operational object performing a specific function excluding the launch.
- *PF* = payload fragmentation debris: objects originating from a fragmentation of a payload due to an explosion or a collision with another orbiting object, or which were unintentionally released from the original payload.
- *PM* = payload mission related debris objects: objects which served purpose for the functioning of a payload and which were intentionally released afterward as debris (e.g. overs for optical instruments).
- *PD* = payload debris: all other payload related debris for which the genesis is unclear.
- *RB* = rocket body: object designed to perform launch relate functionality.
- *RF* = rocket fragmentation debris: objects originating from a fragmentation of a rocket due to an explosion, or which were unintentionally released from the original rocket.
- *RM* = rocket mission related debris objects: objects which served purpose for the functioning of a rocket and which were intentionally released afterward as debris (e.g. shrouds).
- *RD* = rocket debris: all other rocket related debris for which the genesis is unclear.
- *UI* = unidentified: all the remaining tracked objects in orbit that cannot be traced back to a specific launch event.

The two most striking observations to be made about Figure 1.1 is that (i) there are way more non-operational and uncontrolled space debris in orbit than functional objects, (ii) the number of those debris has increased in an almost exponential way over the past decade. Not all objects stay in orbit forever. The ones in the lowest orbits end up re-entering into the Earth's atmosphere. As Figure 1.2 shows, almost 2500 reentry events of tracked objects occurred over the 2021-2023 period alone. Among those reentries, it is almost certain that some objects survived and reached the ground, even if fragments have not been discovered yet. Based on statistical models, it is possible to estimate the number of space debris smaller than the tracking cut-off size. Following the ESA's predictions, 900,000 objects from 1 cm to 10 cm are orbiting the Earth and this model estimates the number of objects between 1 mm and 1 cm at a staggering 128 million. Finally, we can remark that on average over the last two decades, 11.2 non-deliberate fragmentation continue to occur in every year. This contributes to fuel the Kessler's syndrome according to which collisions between objects could cause a cascade in which each collision generates space debris that increases the likelihood of further collisions.

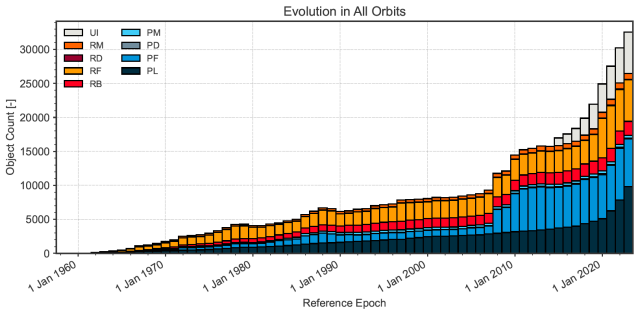


Figure 1.1 / Evolution of the number of tracked objects greater than 10 cm in geocentric orbit by object class. Please consult the text for the definitions. Figure taken from T. E. S. D. O. ESA (2023, Fig. 2.1.a).

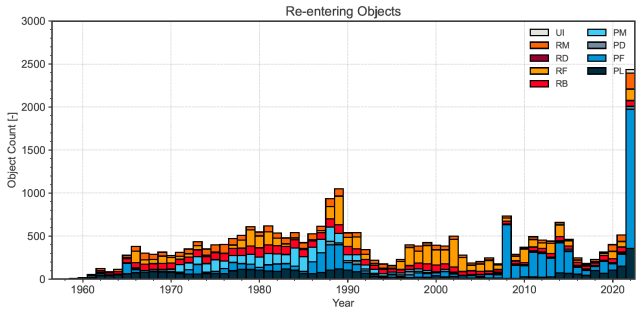


Figure 1.2 / Evolution of the number of re-entering objects in each year by object type without human spaceflight. Please consult the text for the definitions. Figure taken from T. E. S. D. O. ESA (2023, Fig. 2.30.a).

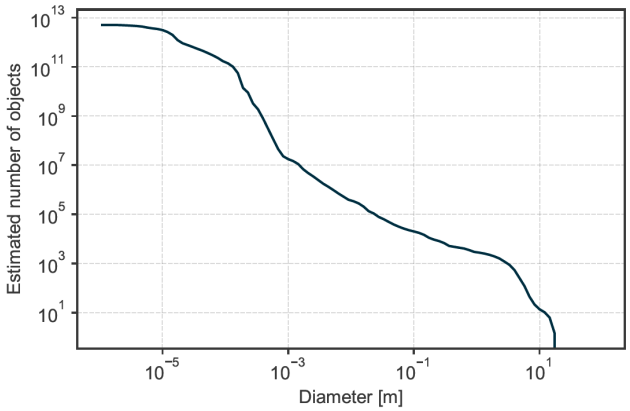


Figure 1.3 / Estimated number of space debris objects as function of the object size in Earth orbit. Figure taken from T. E. S. D. O. ESA (2023, Fig. 2.8).

Space sustainability and risks

The accumulation of derelict objects in orbit is threatening in two ways. The first one relates to the sustainability of space activities in the future. Several important space applications could be rapidly affected or even lost, such as weather forecasting, climate monitoring, Earth sciences, or satellite communications. Pursuing space exploration could also be potentially disturbed. Under this scenario, socio-economic and scientific impacts could be severe (OECD and Development (2020)). The second threat is linked to reentry hazards to human health, Earth environment, and damages to assets caused by impacting or floating fragments, pressurized or explosive fragments, hazardous chemical and radioactive substances (E. R.-e. S. w. g. ESA (2017)).

Space debris mitigation guidelines

Several national and international organizations of the space faring nations have established Space Debris Mitigation standards. The contents may be slightly different between them but their fundamental principles are the same and follow the guidelines issued by IADC (2020). Following here the European Cooperation for Space Standardization (ECSS) standard, those principles can be summed up as follows:

- *The limitation of debris released during normal operations*: the release of objects such as connectors, fairings, covers, etc. shall be avoided or retained by design. This also applies to particles larger than 1 mm, e.g. used in pyrotechnic devices or solid rocket motors.
- *The minimization of the potential for on-orbit break-ups*: accidental in-orbit break-up resulting from stored energy shall be minimized and a quantitative risk assessment shall be conducted. Moreover, intentional in-orbit fragmentation shall be prohibited.
- *Post mission disposal*: actions shall be taken to remove spacecraft and orbital stages that have reached the end of their mission operations from the useful densely populated orbit regions, namely the Low Earth Orbit (LEO) and Geostationary Orbit (GEO). For LEO, no space system shall be left for more than 25 years after the end of their mission. For GEO, space system shall be disposed to another orbit sufficiently far away such that the decommissioned spacecraft will not enter the GEO protected region within 100 years.
- *Prevention of on-orbit collisions*: the design and mission profile of a spacecraft shall be designed such that the probability of collisions with known orbiting objects are limited. Moreover, counter-measures such as collisions avoidance shall be applied when required.

In addition to those guidelines, ESA established re-entry safety requirements setting a threshold of 10^{-4} on the global re-entry casualty risk. It should be the role of the spacecraft manufacturers and operators to ensure that the space system are designed and operated such that this threshold is not exceeded.

In the context of this work, we are interested in the second and third guidelines, i.e. the minimization of the on-orbit break-ups and especially in the post mission disposal with its associated re-entry safety requirements. The main mitigation measures to put in place in accordance with those two guidelines are organized in Figure 1.4 and Figure 1.5 and commented below.

Protected orbits Because of their level of congestion and their strategic importance, low Earth orbit (LEO), from Earth surface up to approximately 2,000 km, and the Geostationary orbit (GEO) with equatorial latitudes of ± 15 deg and altitudes from 34,786 to 36,786 km, are considered as protected regions with regard to the generation of space debris. Other crowded regions such as Medium Earth Orbit (MEO) or navigation satellite orbits are also under threat from space debris, but are considered less critical.

Uncontrolled reentry, controlled reentry and graveyard orbit Two types of reentry into Earth's atmosphere are possible. In a controlled reentry, the spacecraft is maneuvered in a controlled manner into a targeted reentry path with a well-defined impact footprint on the surface of the Earth. In contrast, uncontrolled reentry implies that the spacecraft falls back to Earth in an unguided fashion, which leads to uncontrolled reentry time and impact zone. A graveyard orbit can be defined as an orbit that lies away from common operational orbits and in which it is unlikely that collisions will be provoked in the future.

On-ground casualty risk The casualty/fatality risks posed by re-entering debris for humans depend essentially on two main factors. The first one is the total casualty area, which can be seen as an equivalent impact area depending on both the average projected area of each fragment and the cross-section of a human being on the Earth surface. This first factor thus depends on the space system design and the fragmentation process and can be directly reduced by a proper design for demise strategy (discussed after). The second factor is the average population density which depends on the orbit inclination, latitude step size and argument of perigee. It is only possible to reduce the importance of this second factor in a controlled reentry strategy.

Deorbit maneuver Deorbit maneuvers are meant to initiate the reentry phase or to lower the orbit such that the presence of a spacecraft in LEO is ensured to be below the 25 years limit. Passive deorbiting can be achieved via four approaches: (i) natural decay due to atmospheric drag, (ii) deployment of deorbiting systems such as drag augmentation devices (sails, balloons, foams) electro-dynamic tethers interacting with Earth's magnetosphere or electro-static tethers interacting with Earth's ionosphere, (iii) low thrust propulsion systems such as Hall effect thrusters and arcjets, and (iv) high thrust propulsion systems such as monopropellant Hydrazine-based engines. The three first techniques are typically used in case of uncontrolled reentry while the steep reentry angle

required by controlled reentry necessitates to the use of high thrust propulsion engines. They can also be combined. Active deorbiting on the other hand can be conducted in two main ways: (i) attachment of a deorbiting kit (with solid rocket engines e.g.) to the space debris by a mothership or (ii) the capture of the debris by a chaser spacecraft. Active deorbiting involves complex rendezvous and capture steps.

Reorbit maneuver The goal of reorbit maneuvers is to move a satellite from its operational orbit to another one, typically a graveyard orbit. This can also be done following a passive low thrust propulsion approach or an active removal with a second servicer.

Passivation During the disposal phase, a spacecraft shall permanently deplete r make safe all remaining on-board sources of stored energy in a controlled sequence. Those sources include power and propulsion systems. The goal of the power passivation is to avoid battery failure and subsequent break-up due to a thermal runaway induced by over-temperature, over-charge or over-discharge, short-circuit or structural issues. This is usually done by isolating the battery from the other components. There are two main risks associated to the left-over of propellant in the tanks after the mission: (i) propellant dissociation causing tank burst, (ii) hypersonic velocity impacts causing explosion. Propulsion passivation consists in venting the propellant by means of valves or microperforators.

Design for Demise The Design for Demise (D4D) principle resides in the intentional design, assembly, integration and testing of spacecraft components such that the 1-in-10 000 probability threshold for the casualty risk of any reentry event is respected. As a rule of thumb, this threshold may currently be difficult to comply with for mid to large satellites (typically above 2000 kg). But for space systems that allow it, the D4D is essential as it avoids the need to plan a very costly and difficult controlled reentry.

1.1.2 Space debris mitigation: the design for demise paradigm

Among the various technologies covered above to be developed in order to sustain the implementation of space debris mitigation measures, the Design for Demise has certainly a key role and it represents the motivation behind this work. The interest of this emerging paradigm and its principle are exposed below.

Why is D4D important?

As discussed in the previous section, a controlled or uncontrolled reentry can be considered for a space system in LEO at the end of its life. In both cases, the

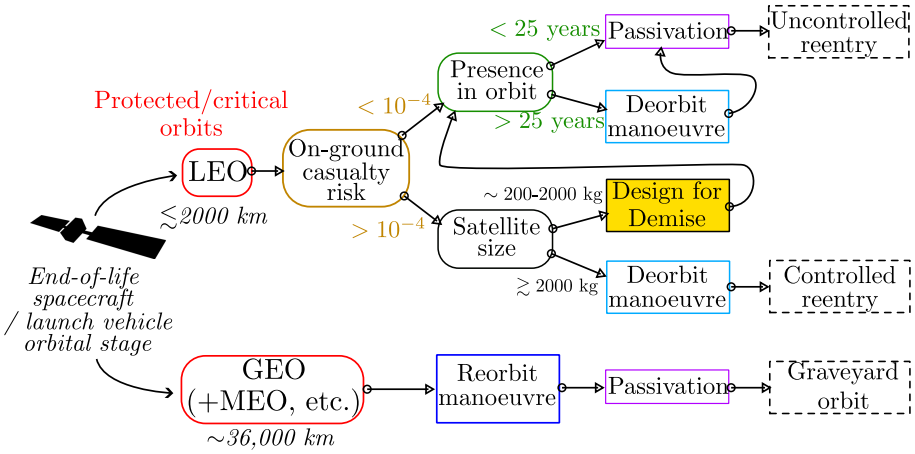


Figure 1.4 / Strategy behind the mitigation measures to avoid that a spacecraft at the end of its mission generates additional debris.

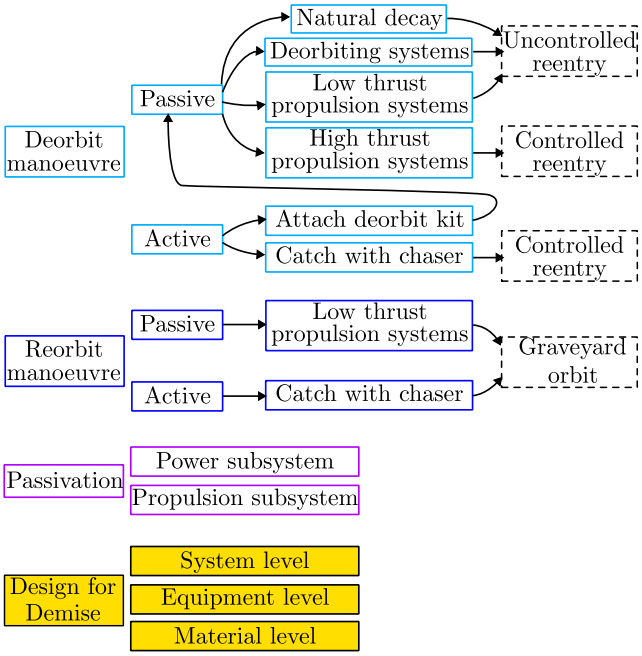


Figure 1.5 / Details of the main mitigation measures presented in Figure 1.4.

estimated on-ground casualty risk for any space object upon its atmospheric reentry shall be lower than 10^{-4} according to safety regulations initiated by ESA and adopted by other Agencies. Controlled reentry, if successful, allows complying with this rule by targeting a sparsely populated impact area. However, this strategy results in significant impacts on the space system mass, complexity and cost. For example, the addition of high thrust propulsion systems as well as bigger tanks should be considered in the design, which could in turn require a heavier structure. The resulting additional mass will have a direct impact on launch costs and might even require the satellite to be launched on a larger vehicle. Alternatively, uncontrolled reentry has much more limited system impacts. However, it calls for specific measures to adhere to the on-ground casualty risk requirements. That is where D4D comes into play.

How can D4D be implemented?

The D4D techniques can be classified according to two elements. The first one concerns the level at which the modifications are applied, i.e. either at a system level with a global impact on the whole spacecraft or at a component level with a focus on specific equipment. The second aspect is about the strategies adopted to maximize the chance that all the spacecraft components will be completely destroyed before reaching the ground. We can identify four main approaches to do so (Cattani et al. (2021), Grassi et al. (2017)):

- *Minimization of the heat load required for ablation*: this goal is typically pursued at the component level by reducing the mass of the component (e.g. a smaller reaction wheel but that spin faster to keep the same inertia effect) or replacing the material (e.g. a material with a lower melting or lower emissivity).
- *Maximization of the available heat*: another option is to provide additional energy to the object and channel it into the ablation processes (in order to avoid its dispersion in unwanted ways). Three approaches are possible. The first one at the system level is to change the ballistic coefficient of the spacecraft such that an optimal trade-off is found between the time spent during the reentry (associated to the total heat flux stored by the object) and the trajectory profile, i.e. velocity and flight path angle (which will determine the peak heat flux). The second approach at the component level is to change the local heat fluxes by modifying the shapes of the components. For instance, sharper corners are likely to increase the local heat flux. The last possibility is to trigger some exothermic reactions, e.g. by means of thermite mixture.
- *Optimization of the heat transfer*: this is mostly related to the fact that internal pieces of equipment are shielded from the heat flux by external structures. Achieving an earlier break-up by weakening the joints that hold together some parts could expose them earlier to the reentry heat flux and maximize the chance of destruction. With the same objective, orifices in the outer panels of the spacecraft or lattice structures are also

possible.

- *Minimization of the casualty area*: in order to reduce the overall on-ground casualty area, one can try to keep some fragments together through containment techniques such as containment box, nets or non-demisable joints. One should be careful however that by keeping the fragments together, an increment of the impact energy is induced, possibly making it higher than the safety threshold of 15J.

The implementation of these techniques should of course preserve the structural integrity of the spacecraft during all the mission and should not compromise its nominal functionalities. This combination of unaffected spacecraft performance and enhanced "demisability" makes the conception of a spacecraft following a D4D scheme an ambitious endeavor.

1.1.3 Design for Demise: the need for high-fidelity predictive capabilities

The multi-level and multidisciplinary natures of the D4D strategy make its implementation into the design phase of new spacecrafts a very challenging task. That is why it must be supported by the development of tools for space debris re-entry prediction and ground-risk assessment. In the following, we focus on space debris re-entry prediction.

Engineering tools

High-fidelity methods such as Direct Simulation Monte Carlo (DSMC) or Computational Fluid Dynamics (CFD) are capable of reliably simulating the challenging hypersonic flow field around a space debris and its ablation. However, these methods solve numerically for the complex fluid dynamics equations and are therefore characterized by extremely long computational times, even for performing aero-thermodynamic simulations for a single point along the vehicles trajectory. For this reason, such tools cannot be extensively used neither in the preliminary design phase of the spacecraft nor in the fast, daily, analysis of potentially hazardous debris. Less computationally expensive approaches, that rely on simplified physical and mathematical models, should be used instead, obviously at the cost of less accurate results (Benedetti et al. (2018)). This has led to the development of engineering tools by space agencies and subcontractors, which has become an important and active research area in the recent years. Based on the way the spacecraft is represented, the available tools can be divided into two categories: object-oriented and spacecraft-oriented codes (Tobias Lips and Fritsche (2005)). Object-oriented methods simulate the re-entry of individual parts of the spacecraft assumed to be represented by primitive shapes with associated motion capabilities. The SESAM part of the global ESA DRAMA software belongs to this category. Spacecraft-oriented codes, on the other hand, model the complete spacecraft as close as possible

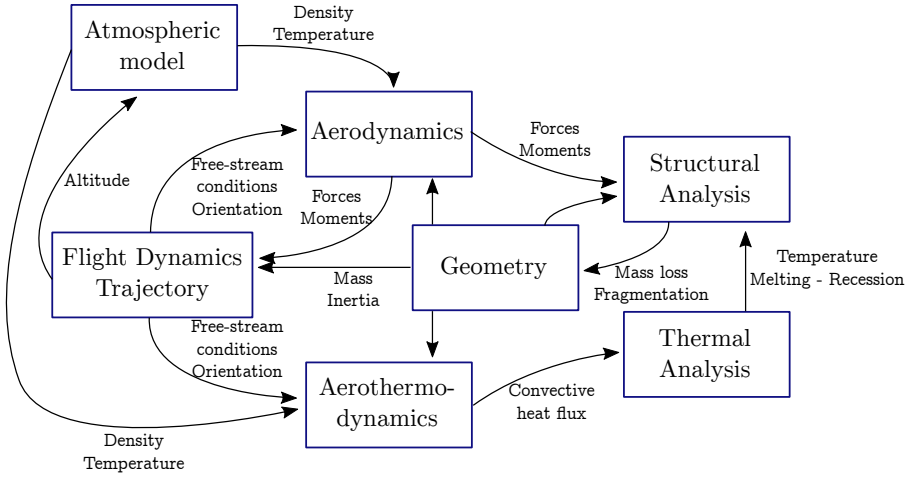


Figure 1.6 / Typical structure of space debris re-entry prediction tools with the different analysis modules and their dependences.

to the real design as one consistent object. This is the case for the SCARAB tool developed by HTG or the PAMPERO tool of CNES. This difference in the geometrical representation, leads also to some differences in the physical models between the two categories of tools. These tools however share a similar structure which is summarized in Figure 1.6. A complete prediction tool for spacecraft destruction needs to be multidisciplinary by involving different analysis modules continuously exchanging results at each step of the re-entry. Those are briefly commented based on Figure 1.6.

Atmospheric model Modelling the Earth atmosphere, particularly the density and temperature profiles as function of the altitude, plays an important role in the simulation of the re-entry. Indeed, the aerodynamic forces and the heat fluxes depend on the density while the Mach number is function of the temperature. Different atmospheric models exist and require the altitude to return the desired quantities.

Flight dynamics – trajectory The trajectory is predicted by integrating the equations of motion with three (for object-oriented codes) or six (for spacecraft-oriented codes) degrees of freedom. The aerodynamic forces and moments influencing the velocity and the attitude of the debris are provided by the aerodynamic module. For object-oriented tools, different attitudes modes can be chosen like randomly tumbling (the attitude is not computed as it is done in spacecraft-oriented tools). Atmospheric and gravitational forces can also be taken into account. The updated mass and inertia of the object are given by the geometry module.

Aerodynamics For object-oriented tools, the drag coefficient is computed based on hypersonic correlations or simplified theory and is then given to the trajectory module. In the new version of SESAM, lift and side force coefficients are also computed. In spacecraft-oriented tools, the forces and the moments are obtained by integration of the pressure and shear coefficients on the surface of the debris. These coefficients are also given by correlations. In both cases, the correlations are different depending on the flow regime (continuum, transitional or free-molecular). To compute the coefficients, the geometry and the density are needed and they are provided by the appropriate modules.

Aerothermodynamics This module estimates the convective heat flux given the free stream conditions, the geometry and the attitude of the object. To do so, hypersonic correlations for the stagnation point heat flux are used and are then extrapolated or integrated over the surface of each primitive shape to obtain the heat flux distribution over the entire surface of a body. Shape-dependent effective radii of curvature and motion-dependent averaging factors are applied in order to use these equations for all object shapes and motion. Pre-computed values for common simple geometrical shapes can also be available, like in the new version of SESAM.

Geometry In most object-oriented codes, the spacecraft is initially modelled as a single simple object which virtually contains all the other spacecraft components, without presenting any type of relationship between them. This is called the parent/child concept. The components are based on a pre-defined object list of simple shaped primitives (sphere, box, cylinder, etc.). In the upgraded SESAM version, the spacecraft is modelled as a combination of multiple primitives with two types of relationships between them: "included in" (one primitive is fully shielded by another one as in the parent/child concept) or "connected to" (two primitives are both partially exposed to the flow field and share a thermal conductive area), as explained by Fuentes et al. (2017). In SCARAB, the spacecraft modelling has distinct hierarchy levels, which allows composing a complex system by subsystems, compounds, elements and finally primitives as the lowest level, as pointed out by Lips et al. (2004). The primitive geometries are discretized into panels to reconstruct internal and external spacecraft components. Arbitrary size, thickness and orientation can be assigned to the primitive shapes that can be combined to represent complex components. In PAMPERO, the geometrical discretization is even more advanced as a mesh is generated over the entire object. The geometry is updated at each time step based on the output of the structural analysis: fragmentation or mass decrease due to partial melting are, for example, taken into account.

Structural analysis In the most simple object-oriented tools, a single spacecraft break-up event is assumed at a given altitude or when the temperature of the surface reaches a threshold. For each critical element of the spacecraft

decomposed into its primitive components, a thermal destructive re-entry analysis is then performed, considering partial melting of the fragments or not. In more elaborated object-oriented codes like SESAM, the spacecraft fragmentation is the result of the evolution of the relationships established between the primitives. When a relationship is broken after the complete melting of a link between two elements, a list of fragments is generated. User-defined triggers can also be used, like the heat flux or the dynamic pressure (Fuentes et al. (2017)). In addition to thermal failure analysis, a stress analysis is performed in spacecraft-oriented tools like SCARAB by determining the loads resulting from aerodynamic and inertia forces (Koppenwallner et al. (2005)).

Thermal analysis and melting This last module is of primary interest for this work. For each primitive shape in SESAM, for each panel in SCARAB and for each cell in PAMPERO, the following heat flow balance is considered at the surface:

$$\dot{q}_{tot} = \dot{q}_{conv} + \dot{q}_{rad} + \dot{q}_{cond}\,, \qquad [1.1]$$

where $\dot{q}_{tot}$ is the resulting heating rate, and $\dot{q}_{conv}$, $\dot{q}_{rad}$ and $\dot{q}_{cond}$ are the convective, radiative and conductive heating rates, respectively. The convective heat flux is obtained from hypersonic correlations in continuum and rarefied regimes, the radiative heat transfer from the Stefan-Boltzmann law weighted by some view factors and the conductive heat flux is usually computed by finite differences along a one-dimensional line, taking into account or not the conduction between connected shapes. It has to be noted that some codes, like the NASA ORSAT tool, also consider the contribution of active oxidation in the computation of the total heating rate. Based on this total heat flux, the local body temperature can be estimated and a melting model can be applied to take into account the ablation. Koppenwallner et al. (2005) explain that in SCARAB, the panels are divided into sublayers over which temperature gradients can be neglected such that a lumped mass model is used to compute the temperature via

$$\dot{q}_{tot} = \rho\, c_p\, s\, A_{ref} \frac{\mathrm{d}T}{\mathrm{d}t} \qquad \text{for} \quad T < T_{melt}\,.$$

Once the melting temperature T_{melt} of the panel material is reached, the sublayer temperature stays at T_{melt} and the incoming heat is used to ablate mass by melting at a rate which is determined by the latent heat h_{melt}:

$$\dot{q}_{tot} = \rho\, h_{melt}\, A_{ref} \frac{\mathrm{d}s_{melt}}{\mathrm{d}t} \qquad \text{for} \quad T = T_{melt}\,,$$

where s_{melt} is the thickness of the melted layer. When $s_{melt} = s$, the sublayer is completely ablated and is removed from the panel. PAMPERO relies on a more sophisticated 3D thermal module using finite volume method to calculate the resulting heating rate on each volume i in the following way (Annaloro et al.

(2014)):

$$\rho\, c_p\, V_i \frac{\mathrm{d}T}{\mathrm{d}t} = \dot{q}_{tot} = \sum_{j=1}^{3} \left[\dot{q}_{conv_j} + \dot{q}_{rad_j} + \dot{q}_{cond_{i,j}} \right] \qquad \text{for} \quad T < T_{melt}\,,$$

where the index j runs on the dimension and V_i is the volume of the cell i. Concerning the ablation, a cell is removed when its melting temperature is reached.

The need for high-fidelity numerical tools

There are two main reasons why engineering tools shall be supplemented by high-fidelity numerical tools: (i) to improve the simplified models of the low-fidelity approaches, (ii) to fill knowledge gaps about some complex degradation processes.

Improvement of low-fidelity models The use of case-dependent correlations and simplified physical models in the different analysis modules of engineering tools described above inevitably lead to the generation of important uncertainties in the final estimation of the ground casualty risk. It is therefore important to investigate the effects of simplifying assumptions on the inferred casualty risk. The lack of experimental or high-fidelity numerical data combined with the multidisciplinary aspect of space debris re-entry make the validation and verification of spacecraft demise tools a very challenging task. J. Beck et al. (2017) highlight the fact that there is basic data on real re-entries, but this is at a level where only a basic correlative verification can be performed. It is only recently that significant effort has started to be made to provide accurate data against which simplified engineering models can be assessed consistently. The development of the SAM tool (Merrifield et al. (2015)) and the organization of workshops, such as the R.Tech/CNES/ESA Spacecraft Demise Workshop", the VKI/ESA Workshop on Space Debris Mitigation," or the ESA/CNES ATD3 Working Group, are examples of initiatives taken to improve the reliability of space debris re-entry prediction tools. A convenient way to exploit the richness of high-fidelity data is to build efficient surrogate-based models which encompass complex physics while being suited to be incorporated in a low-fidelity tool.

Production of missing knowledge In order to sustain the development of the design for demise technology, gaining insight into the degradation process of a debris during its re-entry is essential. The determination of the design changes to be applied to new spacecraft components in order to minimize the on-ground casualty risks require a better understanding and prediction of the complex ablation phenomena. Indeed, J. Beck et al. (2017) report that it has been demonstrated "that the current physical modeling of the spacecraft demise does not necessarily capture the correct material behavior or breakup processes

necessary to understand whether particular design changes will be effective in reducing the casualty risk". Therefore, advancement in the predictive but also in the descriptive capabilities of spacecraft demise tools reveals necessary. However, as it has been mentioned earlier, since these tools have to stick to their low-fidelity approach, they will never be able to describe accurately the multi-physical material response of re-entering space objects, yet decisive in the implementation of the design for demise. That is why high-fidelity numerical tools, in conjunction with detailed experimental measurements, have the potential to understand ablation phenomena that are not accessible by low-fidelity simulations and improving the fidelity of the engineering tools models, e.g. by building surrogate-based models encompassing. In the last decades, extensive work has been done on the characterization of Thermal Protection Materials (TPM). On the contrary, numerical or experimental investigations of demise phenomena is quite a recent field, still under development and at the heart of this project. In fact, Fujimoto et al. (2017) point out that there are still technological gaps to establish a highly accurate CFD tool able to simulate the detailed aerothermochemical ablation process.

On the role of uncertainties in the predictive capabilities

What we are trying to accomplish by developing a high-fidelity numerical tool is to achieve new prediction capabilities about the space debris ablation process. To quote Oberkampf and Roy (2010, Ch. 2.2.4), we want to "use a computational model to foretell the state of a physical system under conditions for which the computational model has not been validated".

Sources of uncertainties Due to the complexity of the physics and the different aspects that need to be modeled, this process is inevitably subjected to uncertainties. Each of the three pillars of scientific inference has some sources of uncertainties. To obtain experimental data, the first step is to acquire raw data (e.g. voltages measured in time) from experimental apparatus. There will be a first source of uncertainties stemming from the limited resolution of the measurement devices themselves. Then, the raw data need to be converted to meaningful physical quantities such that they can be exploited to derive conceptual and physical models. This conversion is done by a calibration process which itself involves some sort of theoretical modeling and sometimes computational models. This adds other sources of epistemic and aleatory uncertainties to the whole experimental process. On the modeling side, the assumptions we need to make to formulate the conceptual models manifest themselves either in the model parameters or the model structure itself. According to A. d. Val (2021, Sec. 1.2), both characteristics are can carry two main types of model uncertainties. The first one is aleatory uncertainties whose fundamental nature is randomness. The second type is epistemic uncertainties associated to the fact that there exists a lack of knowledge about the physical phenomena under consideration. Further, epistemic uncertainties can come in two flavors: conscious

and blind. Regardless of what level of physical modeling fidelity is chosen, there are always spatial and temporal scales of the phenomena that are ignored. If we are aware of the assumptions we make, we should account for those conscious uncertainties. On the other hand, blind uncertainties concern unknown unknowns for which it is almost impossible to bound their magnitude. Finally, the translation and discretization of the conceptual model into a numerical program is also disturbed by several uncertainties such as programming errors, discretization errors as well as round-off and iterative errors.

Dealing with uncertainties The outcomes that a numerical tool produces should therefore be analyzed through the lens of the different sources of uncertainties. Two objectives should be pursued: (i) evaluating the impact of the uncertainties on the outcomes, (ii) reducing the sources of uncertainties that can be reduced. The first goal can be addressed with stochastic methodologies in the framework of uncertainty quantification by propagating along the simulation chain different uncertainties, whether of aleatory or epistemic nature. Some confidence intervals can then be added to the outcomes of the simulations and the impact of the different sources of uncertainties can be ranked. The second goal is highly case-dependent and can be achieved through experiments, improved numerical approximations, expert opinion, higher fidelity physics modeling, etc. Neither of these two objectives are considered in this work. However, we can already point out that the choice of high-order numerical methods can be considered as going in the direction of reducing the sources of errors associated to the discretization of the physical models.

1.2 Physics of the aerothermochemical degradation process

At the end of Section 1.1.2, we have seen that most D4D techniques are based on the redistribution of heat flux in order to enhance the ablation of a re-entering space debris. That is the reason why the present study, which aims at improving the efficiency of the D4D, focuses specifically on the heat transfer and ablation phenomena. Those are mostly confined in the boundary layer region adjacent to the body surface. There are however important phenomena taking place upstream, which are described by the two first subsections below.

1.2.1 Hypersonic entry of space debris

We start by looking at the big picture of a space debris re-entering the atmosphere at a hypersonic regime, with the help of Figure 1.7. The following comments, with the numbers corresponding to the one of the sketch, can be formulated:

1. *Hypersonic flow*: as discussed above, a major part of the reentry is in the hypersonic regime, i.e. that in the body's frame of reference, the incoming gas particles velocities and atmospheric pressure and temperature are such that the Mach number is beyond the hypersonic limit.
2. *Strong detached bow shock*: Hypersonic flows are characterized by very strong shock waves[1] across which a dramatic transfer between the high kinetic energy content of the gas to thermodynamic energy is operated. The shape and position of this curved detached shock will evolve as the freestream conditions change along the reentry and because of the shape change of the object. The influence of the shock on the aerodynamic of an object in hypersonic flow depending on its orientation has been investigated experimentally in Guillaume Grossir et al. (2020).
3. *Entropy layer and vorticity generation*: across a shock, entropy is generated. Anderson Jr. (2006, Sec. 5.2) explains that the high curvature of the detached shock in hypersonic regime induces strong entropy gradients between adjacent streamlines. From Crocco's theorem, this translates into the introduction of a large amount of vorticity into the flow. So the inviscid part is strongly rotational, which can complicate the imposition of the boundary layer edge conditions in some methods.
4. *Inviscid flow region*: the inviscid flow region contains a subsonic flow pocket around the stagnation point region and supersonic flow in regions where the upstream shock is more curved. This region can be notably thin due to the thin shock layer and thick boundary layer characteristic of hypersonic flows.
5. *Viscous flow region and viscous interaction*: In the side regions of the object where the flow has gone through an oblique shock wave such that its kinetic energy content is still very high, its slow-down by viscous effects significantly increases its internal energy. In turn, the temperature increases. Because the pressure is more or less constant along the normal direction through the boundary layer, the increase in temperature results in a decrease in density. As explained by Anderson Jr. (2006, Sec. 1.3.3), in order to pass the same mass flow through the boundary layer at reduced density, its thickness must be larger. Those two effects are amplified in the hypersonic regime, which makes the boundary layer grows more rapidly than at slower speeds. The thick boundary layer in hypersonic flow can exert a major displacement effect on the outer inviscid flow which thus sees a much thicker object. The modifications of the inviscid flow in turn

[1] A physical intuition for the formation of shock waves is given in Anderson (1990, Sec. 3.6). Flows are made of molecules for which the presence of a body will change their momentum and energy. The random motion of the molecules communicate those changes to neighboring flow regions and that is how the perturbations brought by the body are propagated through space. Those perturbations waves are related to the mean molecular velocity because molecular collisions are transmitting the perturbations. The velocity at which those waves travel is defined as the sound speed. Now, if the flow upstream of the object is supersonic, the sound waves can no longer freely propagate upstream and will tend to coalesce a short distance ahead of the body. This will in turn create a shock wave, "blocking" the information transfer such that the flow upstream of the shock has no idea of the presence of the body.

affects the boundary layer growth. This interaction between the inviscid and viscous zones is called the viscous interaction.

6. *Separation and recirculation*: the shapes of space debris do not follow in general any aerodynamic rules. That is why sharp corners can be present almost anywhere around the object. In regions with convex corners, the flow turns and expand rapidly through Prandtl-Meyer expansion fans. The boundary later detaches and forms a free shear layer that separates the recirculating flow region behind the corner from the outer flow field. The latter is recompressed and turned back to the freestream direction (Mehta (2017)). Flow reattachement in the recirculating region is often marked by high local pressure and heat transfer (Sinha and Dey (2010)).
7. *Deceleration*: the strong deceleration experienced by a re-entering body during some part of its trajectory acts on it as a gravitational body force. Tellep (1959) notes that while for the gas phase, this body force is negligible in comparison with inertia forces, it may be of comparable magnitude to shear and inertia forces for the dense and viscous liquid molten layer developing on the body surface.
8. *Tumbling*: contrary to a space vehicle designed to survive reentry, a space debris typically exhibits a random tumbling motion when the atmosphere density is such that the aerodynamic forces destabilize the initial object configuration. This behavior has three effects. The first one is that the aerodynamic and aerothermal loads are not concentrated on a single face of the body but are rather affecting most if not all of them. What is commonly assumed is a random tumbling movement resulting in averaged quantities, like in Annaloro et al. (2020). The second effect is due to the additional body force induced by the rotating motion. This "centrifugal" force is likely to influence the development of the molten liquid layer on the body surface. This effect was studied by Cochran and Ostrach (1968). The last consequence of the tumbling motion is a modification of the original aerodynamic coefficients of the body.
9. *Aerodynamic forces*: the pressure and viscous forces of the gas flow impacting the body surface produce aerodynamic forces and moments influencing the trajectory and the orientation of the object.
10. *Fragmentation*: the combination of aerodynamic forces, body forces and thermal loads may lead to the fracture of structural joints and the breakup of the debris in several pieces. The main object and the crumbled pieces are initially in contact but under the influence of the severe flow conditions, they separate and interfere with each other. These interactions influence the trajectory as well as the mechanical and thermal stresses experienced by the different bodies. Interactions between fragmenting objects in hypersonic flow have been studied experimentally in Kovács et al. (2023) and numerically in Register et al. (2020).
11. *Shock-shock interaction*: Two or more shock waves can meet and interact if several objects are sufficiently close to each other or if a single object has a geometry producing that event, such as analyzed by Guillaume Grossir

et al. (2020). Shock-shock interaction may lead to very complex shocks patterns involving a mixture of straight and curved shock waves resulting in mixed subsonic-supersonic flows. Thus, the flowfield can be strongly modified by the presence of such interactions.

12. *Shock-boundary layer interaction*: shock waves can also impinge boundary layer regions, which results in the occurrence of local peaks in the heat transfer and pressure loads. Anderson Jr. (2006, Sec. 7.5) explains that the large pressure rise across the shock acts as a strong adverse pressure gradient on the boundary layer, which causes its local separation and the formation of an induced separation shock wave. The separated boundary layer subsequently reattaches to the surface at some downstream location, accompanied by a reattachement shock and a localized high pressure, causing a high aerodynamic heating. This effect is examined in Garbacz et al. (2022).

1.2.2 Shock layer region and high-temperature effects

In a second step, we focus our attention on the shock layer region pictured in Figure 1.8. Most of the explanations given below come from Potter (2011, Sec. 1.3).

1. *Undisturbed Earth's atmosphere*: considering only entry in the lower orbits of Earth's atmosphere here, the incoming gas before the shock is essentially composed by molecular nitrogen and oxygen. The freestream conditions in terms of temperature, pressure and density will be function of the altitude along the trajectory.
2. *Vibrational excitation*: across the strong shock formed in front of the body, the kinetic energy of the freestream particles is very rapidly converted into translational energy due to their collisions with the more dense shock layer gas. Inter-particle collisions then excite rotational and vibrational energy modes of the air molecules. Translational and rotational energy modes quickly equalize due to efficient energy transfer but vibrational excitation requires much more collisions.
3. *Dissociation and electronic excitation*: as the vibrational temperature continues to increase, the electronic modes of the molecules gets also excited, to the point where the inter-nuclear bonds are overcome and dissociation occurs.
4. *Reactions, ionization and radiation*: further collisions lead to an excitation of the upper internal energy levels. After a characteristic lifetime, the atoms or molecules spontaneously relax to a lower energy level, releasing a photon and radiating electromagnetic energy. For sufficiently energetic collisions, bound electrons can be ejected from their parent nuclei, leaving behind ions and free electrons. The gas becomes a plasma in which exchange reactions are promoted.

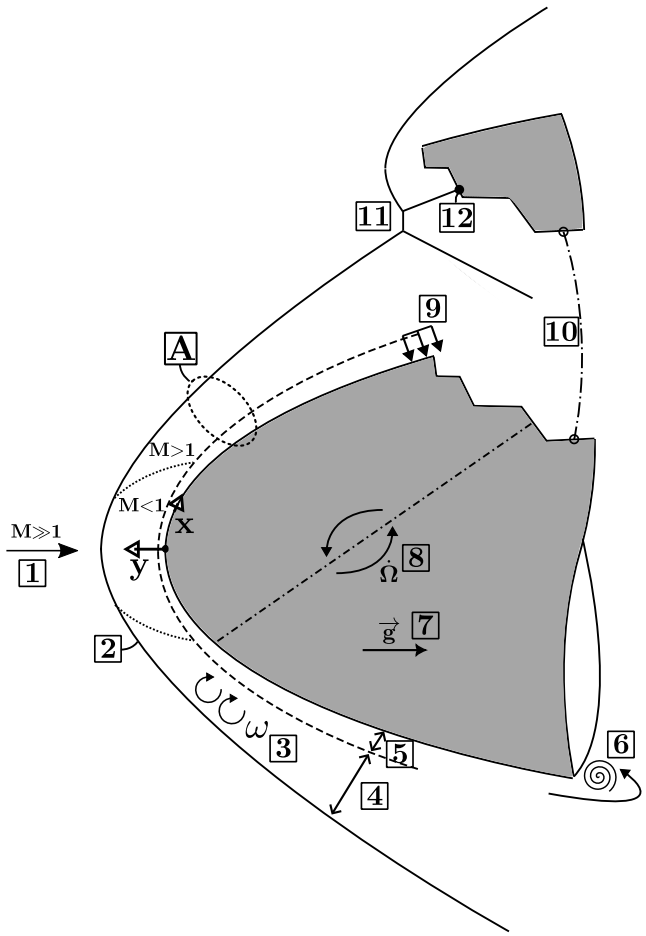


Figure 1.7 / Illustration of the main characteristics of a hypersonic re-entering object. See Section 1.2.1 for the explanations of the numbers.

5. *Relaxation to equilibrium*: sufficiently far behind the shock, the number of collisions undergone by the molecules and atoms is such that the gas tends toward a state of local thermochemical equilibrium.
6. *Cooling and recombination*: the high-enthalpy gas is convected into the boundary layer adjacent to the object where, in addition to radiation, the surface is also heated by temperature and species diffusion[2]. The gas temperature drops and exothermic recombination reactions occur within the cooler regions of the boundary layer, releasing additional chemical energy that is transported to the surface by thermal conduction. Some thermal non-equilibrium may also occur due to faster cooling of translational and rotational modes.
7. *Thermal non-equilibrium*: the disparity in time scales for the excitation of the different energy modes requires a separate treatment and leads to the formulation of multi-temperature models. It is only after a sufficient amount of collisions that the plasma tends toward a situation where all energy modes can be described by equilibrium distributions at single temperature.
8. *Chemical non-equilibrium*: in the chemical non-equilibrium region after the shock, the chemical composition of the plasma continues to change due to chemical reactions between the various species present. Note that the chemical non-equilibrium region is typically more extended than the thermal non-equilibrium region as more collisions are required to reach equilibrium.

1.2.3 Various ablation processes depending on the material

Before describing in details all the phenomena involved in the specific ablation mechanism we are considering here, we first give an overview of all the possible ablation processes based on the type of materials, based on the categorization proposed in E. S. A. ESA (2020).

Metals The most present metallic materials in space debris are aluminum alloys, steel alloys, nickel alloys, copper alloys and titanium. They can be found in most critical items of a spacecraft which are typically the tank, the reaction wheels or the ballast mass. Metallic components undergo melting when subjected to high heat loads and various correlated phenomena such as evaporation, spraying or oxidation. It must be noted that they are most of the time coated or protected by a surface treatment in order to improve their corrosion behavior or their tribological properties. This can mostly influence their emissivity properties and possibly their degradation behavior.

[2] The heat transfer contribution from the species diffusion is associated to the heat of formation of each species representing the important chemical energy associated to the reactions that occur in the shock layer.

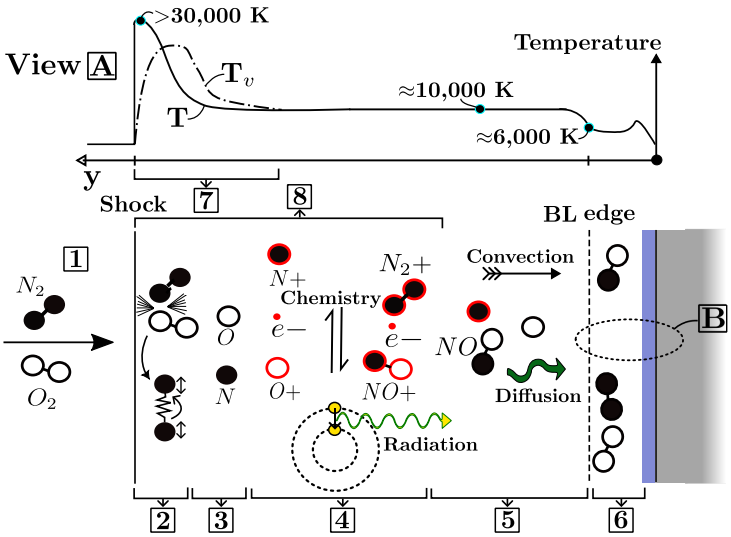


Figure 1.8 / Illustration of the chemical processes and temperature evolution in the shock layer region of a hypersonic re-entering object. Zoom in region A of Figure 1.7. See Section 1.2.2 for the explanations of the numbers. Figure inspired from Potter (2011, Fig. 1.5).

Composites Composite materials are made of a reinforcement that provides the strength and stiffness and a matrix surrounding it such that it holds in place. For space applications, the reinforcement is usually carbon or glass fibers while the matrix is typically epoxy or cyanate ester. Due to their advantageous strength-to-weight ratio, they are mostly used on the main structure of the spacecraft such as face skins of the sandwich panels and as thicker monolithic parts (e.g. structural booms) and for composite overwrapped pressure vessels (COPV). In the case of debris made of composite materials like Carbon Fibers Reinforced Polymers (CFRP), the ablation process is similar to the one encountered in TPS shielding the spacecraft from high heat fluxes during re-entry. First, the material resin is progressively pyrolyzed producing gases which are blown out of the porous material and injected into the boundary layer. The remaining charred material (fibers and residual carbon from resin) is then eroded by oxidation, sublimation or spallation (Pierre Schrooyen (2015)).

Polymers In addition of being present in the matrix of composite parts, polymer materials are also widely used in spacecraft as adhesives, tapes and potting agents for insert bonding in sandwich panels. Most polymers are of semi-crystalline nature, which makes their phase transformation particularly complex with a combination of melting and glass transition accompanied by a non-Newtonian behavior.

Ceramics Thanks to their favorable thermal stability and very high melting point, ceramics such as silicon carbide are used for high temperature applications (e.g. propulsion) or for structures such as optical payload. They typically undergo an amorphous phase transition such as glass transition.

Glasses Glassy materials, like silicate, are used for high-precision optics (lenses, mirrors) and scientific equipment. Some polymers can also have a glassy behavior. A peculiarity of glassy materials is the high viscosity of the molten layer and the low thermal conductivity, which promote higher resistance to the decomposition. It is concluded by Bethe and M. Adams (1959) that for this class of materials, evaporation tends to be the driving ablation phenomenon, contrary to metals for which the melting and the removal of the molten layer by shear stresses are expected to play a more significant role.

The developments carried on in this thesis concern the continuum regime of the reentry trajectory and the ablation of both metals and materials undergoing glass transition with Newtonian behavior such as some ceramics and glasses. We focus on the continuum regime because it is expected that for objects re-entering from LEO, the most determinant portion of the trajectory for the destruction of the debris will be roughly between 90 and 50 km, according to E. S. A. ESA (2020). The choice of the material type has been driven by the fact that the ablation of metals, ceramics and glassy materials have received far less attention in the past decades than the one of composite materials which are typically used in TPS. However, similarly to composite-made debris, those materials show strong resistance to high-enthalpy environments and are classified as critical for demisability.

1.2.4 Boundary layer region and ablation by melting

Since our interest lies in the ablation of metallic, ceramics and glassy materials in the continuum regime, we shall now detail all the possible phenomena involved in the ablation of those materials and taking place in the boundary layer region adjacent to the body surface. In the following, we will employ the term "ablation by melting" to designate a degradation process not only involving the transition from a solid to a liquid state, but also subsequent phenomena like evaporation or shear removal. A complete list of the possible phenomena covered by this term are described below and illustrated in Figure 1.9. On this sketch, the distinction between the solid, the liquid and the mushy region in-between is done. This represents therefore the ablation process of a metallic alloy. However, even if ceramics and glassy materials do not clearly show this phase distinction, their ablation involves, just as metals, the presence of three phases, i.e. the incoming air, the pure solid body and a region in-between which is flowing like a liquid.

1. *Laminar boundary layer growth*: the main characteristics of boundary layer in hypersonic flow were discussed above: thick boundary layer and intense viscous dissipation. We can specify here that those characteristics

will be particularly marked in regions downstream of the shock where the flow is still in a supersonic or hypersonic regime, i.e. for which the flow has gone through a weak oblique shock wave. On the contrary, in the subsonic pocket around the stagnation region, the development of a compressible subsonic boundary layer takes place.

2. *Laminar-turbulent transition*: at some point along the surface, any disturbance in the flow will start to get amplified. This is the beginning of the transition region. Predicting this transition in "standard" subsonic conditions over non-ablating solid surfaces is still nowadays the subject of intense research. In our case, this becomes even more complex as the combined effects of phenomena like mass blowing, liquid layer unsteady wavy behavior, chemical reactions or surface temperature will influence the transition process. This was the topic of the work conducted by Miró Miró (2020).
3. *Turbulent boundary layer*: similarly to boundary layers developing in subsonic flows, the ones in hypersonic regime also have a fuller velocity profile than their laminar counterparts due to the enhanced mixing brought by turbulence (see Anderson Jr. (2006, Sec. 6.8)). This results in larger velocity gradients near the surface and consequently larger skin friction and heat transfer. Thus, the boundary layer nature directly impacts the ablation process.
4. *Liquid layer shearing*: by "shearing" of the melt layer, we refer to the total amount of momentum imparted by the gas flow on the liquid layer, i.e. due to velocity and pressure gradients. The importance of each contribution depends on the location around the body and the local flow conditions.
5. *Convection of liquid layer*: the momentum transfer from the gas-liquid interface into the liquid film establishes convection and contributes to transport the liquid in the gas flow direction along the body solid surface.
6. *Melting and recession of solid*: once the melting point of the solid is reached and the heat of fusion is consumed, the solid melts and the liquid-solid surface recedes. Depending on the local flow conditions in the liquid layer, the opposite phase transformation, i.e. solidification might also occur. Note that on the sketch, the liquid layer is already present but at the onset of melting, the gas is directly in contact with the solid surface.
7. *Gas-liquid interface instabilities*: Despite large values of dynamic viscosities for the considered demising materials, the corresponding kinematic viscosity is expected to be smaller than the one of the hot air which has a very low density. In such circumstances, the boundary layer thicknesses in the gas and liquid can be expected to be comparable. Now, the unstable interface wave growth in high-speed flows depends essentially on the interactions taking place within the two conjugated boundary layers (i.e. on the gas and liquid sides). O. Girin (2017) thus shows that the driving mechanism for the interfacial hydrodynamic instabilities differs from the Kelvin-Helmholtz one and it is identified as "gradient" instability. A sim-

ilar distinction is made in Marmottant and Villermaux (2004). O. Girin (2017) identifies, based on the value of the Weber number, a stable region in the vicinity of the stagnation point and an unstable region elsewhere in which interface instabilities will grow and eventually lead to the spraying of droplets.

8. *Surface tension effects*: because of the imbalance of intermolecular attractive forces that appear at the interface between a liquid and another medium, a resulting localized surface force is present. This force, called the surface tension force, tends to resist an increase in the interface area and induces a pressure jump across the interface. Because the surface tension coefficient depends on the temperature and composition, gradients of those quantities along the interface generates a flow in the liquid phase. This is called the Marangoni effect. Note that the surface tension coefficients of liquid metals are particularly high because the strong metallic bonding influence the surface tension force (Faghri and Yuwen Zhang (2006, Sec. 2.6.2)). As such, surface tension effects might play an important role in the liquid layer dynamics.
9. *Evaporation*: besides the mechanical removal of material via melting followed by spraying, evaporation of the molten layer is expected to be the second ablation driving factor, as discussed by Bronshten (1983, Ch. 3) and shown by Dias et al. (2016). The rate at which the liquid surface will evaporate can be predicted by means of complex kinetic considerations. It is essentially determined by the local temperature and pressure conditions which influence the amount of particles evaporating from and condensing into the liquid as well as the ones reflected on the interface. The local partial pressure of the liquid vapor in the gas mixture adjacent to the interface plays a crucial role in this net mass flux across the interface. Note that the total liquid evaporated mass (hence contributing directly to the ablated mass) is likely to be greatly increased with the spraying of liquid droplets inside the gas because of the increased total surface area which is a determinant factor for evaporation. Let us also remark that evaporation is an endothermic process which will absorb some heat received by the body. This is referred as evaporation cooling.
10. *Vaporized species diffusion and shielding effect*: when the vaporized liquid species are injected into the gas phase, they diffuse and form a protective gaseous film around the body absorbing a portion of the incoming heat flux (Herman and Melnik (1962)). This has also the effect of increasing the gas boundary layer thickness and modifying the absorption and scattering properties of the medium for the incoming radiation. O. Girin (2017) makes the remark that the injection of the vapor inside the boundary layer decreases the local Knudsen layer and can contribute to raise the height from which the continuum assumption breaks down.
11. *Homogeneous reactions between air and vaporized species*: the injected vapor liquid species might react with some gas species, similarly to what happens with the pyrolysis products of degrading porous thermal protection

systems (see Helber et al. (2014)).

12. *Boiling*: while evaporation can happen at any temperature, boiling of the liquid layer will only occur if the boiling point of the liquid is reached. In this case, gas pockets inside the liquid form, rise to the interface and burst, which can greatly affect the interface topology. In experiments on ordinary chondrites, Helber et al. (2019) observe boiling and frothing at the surface of the sample. Bronshten (1983, Sec. 2.9) notices that since the melt layer is usually very thin and heat is supplied to it from the interface, the rate of mass loss due to boiling should be expected to be only slightly higher than the one due to evaporation.
13. *Radiative cooling*: as noted by Anderson Jr. (2006, Sec. 18.8), continuum radiation emitted by the hot surface (in liquid or solid state) can be a significant cooling mechanism.
14. *Heterogeneous reactions between air and surface* (*oxidation*): in addition to homogeneous reactions inside the gas phase, heterogeneous reactions between gas and condensed phase species occur as well. Those are in general exothermic and increase the total heat flux on the ablating object. In our case, oxidation reactions are expected to be the dominant heterogeneous reactions. It is found in Helber et al. (2019) in the context of plasma flows around meteoroid samples that the modeling of the release of the formation enthalpy associated to oxidation reactions is necessary to match the experimental data. Two main types of oxidation mechanisms exist. Active oxidation is very fast and forms a volatile oxide while in passive oxidation, which is a longer process, the gas and condensed phase species react to form a solid oxide film. In the latter case, the oxide layer acts as a protective layer against the aerodynamic stresses and thermal loads from the gas. The development of models to estimate the growth of this oxide layer by passive oxidation has been the topic of the work by Jorge (2019). The formation of an oxide layer at the surface plays also an important role in the radiative cooling process. Indeed, oxides being generally more refractory than metallic alloys, higher radiative cooling rates are expected. This is studied experimentally in Barka et al. (2019). Let us also mention that in certain circumstances, it is possible that because the melting point of the oxide layer is higher than the one of the virgin solid, melting occurs below the solid oxide layer. In this case, the flow in the melted layer is mainly driven by buoyancy effects.
15. *Oxide layer peeling*: Prévereaud et al. (2016) stress out the fact that the oxide layer developed can be friable, peels off and finally detaches to the surface under the influence of the gas flow. This mass removal from the main body can be considered as yet another ablation mechanism.
16. *Species diffusion in condensed phase*: even if diffusion rates are lower than in the gas, the transport of species due to local concentration gradients is also possible in the liquid layer and can influence the thermophysical properties. It is interesting to remark that in order to be able to predict the growth of the oxide layer, taking into account the diffusion of species

in the vicinity of the interface with the gas is crucial.

17. *Shock layer incoming heat flux*: as illustrated in Figure 1.8, the total heat flux coming from the shock layer is determined by convection, species and temperature diffusion, and radiation contributions.
18. *Boundary layer heat flux contributions*: the shock layer heat flux is supplemented by various contributions (added or subtracted) from the boundary layer, reviewed in the previous points.
19. *Heat conduction in condensed phase*: the total resulting heat flux is finally conducted inside the condensed phase. This will induce a bit of swelling due to the thermal expansion of the material.
20. *Liquid layer spraying*: from the shearing of a smooth liquid film to the detachment of droplets, there is a complex sequence of events. Marmottant and Villermaux (2004) summarize the process as follows. The gas flow initiates a shear instability on the liquid and roll waves are formed. The transient acceleration experienced by the liquid then leads to a Rayleigh–Taylor type of instability at the wave crests, producing liquid ligaments which further stretch in the air stream and break into droplets. This is the primary atomization. The secondary atomization mechanism is due to the break-up of the droplets into smaller ones and the impact of some droplets on the liquid film. From the comments made before, we can realize that the mechanical removal of the liquid layer from the object involves four related aspects: (i) the forces exerted by the gas on the liquid, (ii) the development of hydrodynamic instabilities at the interface, (iii) the stabilizing surface tension effects, and (iv) the tearing-off of liquid particles from the melted layer.
21. *Liquid droplets convection*: the liquid droplets stripped from the film are caught by the air flow. As they evaporate, they form a mixture of air-liquid vapor mixture containing liquid droplets. Those become hardened afterward due to cooling and form small solid balls, as explained by A. Girin and Kopyt (1994).
22. *Gas-condensed phase properties jumps*: one of the peculiar characteristics of this multiphase situation is the very high properties jumps (density and thermophysical properties) that exist at the gas-condensed phase interface because of the very dissimilar nature between the gas and the condensed phases. Concerning the temperature, the assumption of continuity across the interface is often made. However, it must be noted that for high evaporation rates, non-equilibrium effects can make the temperature jump non-negligible. In continuum regime, the continuity of the tangential components of the velocities at the interface is applicable. Note that a discontinuity in the derivative of the two profiles is present because of the different viscosities. Regarding the normal velocities[3], an important

[3]The normal velocity profile sketched in Figure 1.9 is just an example of a possible velocity profile. Its actual shape could be quite different, especially in the liquid layer. For the pressure profile that is assumed to be constant along the boundary layers, it must be mentioned that Anderson Jr. (2006, Sec. 6.4) pointed out that for very large freestream Mach number, this assumption is not always valid.

jump will exist when evaporation takes place. Finally, there will also be an appreciable discontinuity in the pressure profiles because of the recoil pressure effect due to evaporation and the surface tension forces.

23. *Re-solidification*: as pointed out by Duffa (2013, Ch. 9), the molten layer will migrate back under the effect of the shear and may re-solidify in important thickness in expansion areas. The same holds for liquid droplets sprayed from the surface, as mentioned earlier. This re-solidification will modify the initial shape of the object and affect its aerodynamics.

To close this section, it is important to recall that in addition to all the phenomena listed above, the influence of body forces due to deceleration and tumbling motion can have a substantial influence on the melt layer dynamics, as already mentioned in Section 1.2.1. Finally, we insist on the fact that under the heading "ablation by melting", we include all the phenomena described above, even though we consider only the ones highlighted by the purple boxes on the sketch in the framework of this thesis.

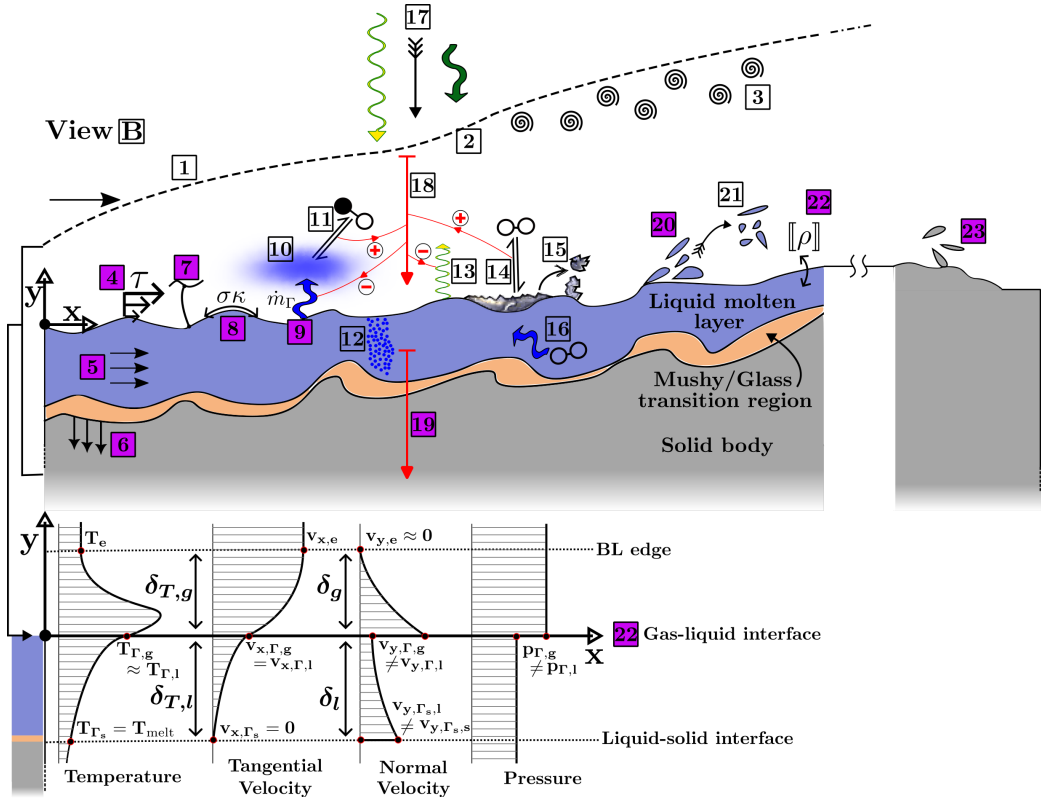


Figure 1.9 / Illustration of the phenomena occurring in the boundary layer region and contributing to the ablation by melting of a re-entering object. The profiles of quantities in the bottom part are inspired from Dorrance (2017, Fig. 3.3). Zoom in region B of Figure 1.8. See Section 1.2.4 for the explanations of the numbers. The ones in the filled purple boxes are those considered in this thesis.

1.3 Numerical simulation of melting ablation

As already mentioned earlier, our aim is to develop a high-fidelity numerical tool to study in details the ablation of re-entering space debris. Our focus lies on the "liquefying" materials for which a liquid layer appears in the ablation process.

1.3.1 Simplifying assumptions

Given the extreme complexity and variety of the phenomena involved in space debris reentry and their ablation, illustrated by Figure 1.7, Figure 1.8 and Figure 1.9, we focus our efforts by relying on the following simplifying assumptions.

- *Space debris attitude and structure*
  - *No dynamic freestream variations*: we don't consider any modeling that could represent the variations of the freestream conditions as the object is progressing along its reentry trajectory. Doing so would require to have a coupling with a trajectory prediction code and a very efficient CFD code allowing computation over at least a dozen of seconds.
  - *No fluid-structure interaction*: although fragmentation events play a crucial role in the overall destruction process of a debris, it is not taken into account here as this would require to develop a Fluid-Structure Interaction (FSI) framework including fracture computations.
  - *Fixed attitude*: since no FSI capabilities are foreseen, we consider objects with a fixed orientation, which prevents the study of the tumbling effects.
- *Flow*
  - *Focus on boundary layer region*: we restrict our attention to the boundary layer region where the detailed ablation mechanisms we want to study take place. From a numerical point of view, the simulation of the shock is made difficult by the very abrupt changes of properties across it and the fact that its position cannot be predicted in advance. This difficulty is even more acute for the very high Mach number regime involved here and the use of high-order discretization techniques. The numerical treatment of the shock is already the topic of recent ongoing research in our group.
  - *No body forces*: we don't consider any additional body forces that could model tumbling or deceleration effects.
  - *No several species*: we will consider multicomponent medium such as air and metallic alloys but they will be described by a "single species" model, meaning that the properties will be the ones of the

homogeneous mixture as a whole. This is obviously a very limiting hypothesis as it is well known that high temperature effects can only be accurately modeled by treating the species separately. We decided in this thesis to devote more effort to implementing the necessary numerical methods rather than rigorous physical models. This choice was motivated by the fact that previous work in our research group such as Pierre Schrooyen (2015) have already established solid models. Their addition in the present numerical framework will be a natural follow-up to this work.
  - *No turbulence*: the transition of the boundary layer from a laminar to a turbulent regime, which is expected to be promoted by the surface irregularities and the formation of liquid waves, is known to strongly influence the heat transferred to the body. That is why its prediction is probably important to accurately simulate the ablation. However, as already mentioned earlier, this represents a complete research field on its own and is beyond the scope of this work.
  - *No phase transformation inside the bulk of each phase*: although we consider evaporation and melting at the phase interfaces, we do not include the modeling of phase transformations inside the bulk, such as boiling or cavitation. This would have required the implementation of dedicated equations of state (most probably through a coupling with external libraries) and of techniques to be able to accommodate appearing phase interfaces.
  - *No tracking of droplets cloud*: depending on the material properties and flow regimes, we might expect the formation and transport of clouds of highly dispersed liquid droplets. The strategy selected in this work based on a local instant formulation is not appropriate for such scenarios. Volume-averaged models coupled with kinetic-based models would probably have been required.

- *Thermal effects*
  - *No radiation*: the incoming and outgoing radiative heat fluxes are not computed in this work. This is again a very strong assumption that is not really in line with the expected physics. Indeed, radiative effects can have a significant influence on the heating rates of re-entering objects. But the resolution of the radiative transfer equation is not a straightforward task and was not foreseen as part of this work.
  - *No thermal non-equilibrium*: thermal non-equilibrium effects are expected to be dominant in the region just after the shock, even though they could also manifest in the boundary layer regions. This is not included in our model but has already been investigated by members of our research group.
- *Chemistry*
  - *No homogeneous reactions*: because we don't take the presence of several species into account, we don't consider chemical reactions

inside the bulk of each phase either. This will also strongly impact the fidelity of our models and will prevent us from modeling for instance the reactions between the vaporized species and the gas ones.
- *No heterogeneous reactions except evaporation/condensation*: similarly, reactions between the gas and the condensed phases are not included. The main limitations brought by this assumption in our situation concerns the oxidation reactions and the formation of a surface oxide layer.

1.3.2 Characteristics of the simplified problem and choice of research paths

From the phenomena identified in Section 1.2.4 and the simplifying assumptions reviewed above, we are now in a position to list the phenomena to be modeled and to identify the possible numerical methods to discretize their governing equations. This identification remains very general at this stage, and allows us to situate the various elements of our framework from a global point of view. In this respect, it does not constitute a proper review of the literature. This is delegated to the beginning of every chapter or section for the sake of clarity given the variety of concepts and methods addressed in this work, which can be presented separately. We proceed in a step-by-a-step manner while (i) justifying each research direction and confronting them with other existing approaches, and (ii) highlighting the requirements in terms of modeling, numerics or verification and validation.

Dealing with three phases

Possible approaches The first element to consider is the way to deal with the presence of the three phases inside the domain. Two main strategies are possible. In the partitioned approach, a weak coupling between the different systems is adopted, i.e. information between the separated solvers are exchanged at specified instants and/or spatial locations. This can be done with explicit or implicit approach, the latter one involving subiterations between the problems such that some prescribed interface conditions are satisfied up to a tolerance level. In the context of the simulation of melting ablation, S. Peluchon et al. (2022) propose an explicit weak coupling between the gas-liquid equations and the solid ones. In the other monolithic approach, a strong coupling is achieved by including the equations for each phase into single global system, with the jump conditions acting as coupling constraints which are parts of the system as well.

The choice of a monolithic strong coupling The multiphysics nature of the targeted application leads to a very stiff non-linear coupled problem. Based on this observation, we opted for the monolithic approach for its incorporated

robustness, temporal stability and conservativity properties, which we believe is essential to deal with the aforementioned challenges. More justifications can be found in Section 9.1.1.

Requirements The implementation of a monolithic approach requires both the derivation of physically compatible coupling conditions between the different systems and the design of a strategy to incorporate in a consistent and stable way those conditions into the global system.

Governing equations in the bulk of each phase

Possible approaches The second point to discuss concerns the choice of the equations governing the behavior of each phase. We recall that we do not consider several species in the equations. For the gas phase, the use of the standard compressible Navier-Stokes equations is clear, especially bearing in mind that one perspective of this work would be to consider the simulation of the highly compressible shock region. For the condensed phase, it is customary to adopt an incompressible model. However, the use of compressible equations with appropriately tuned closure models is also perfectly valid from a physical point of view.

The choice of compressible models for all phases There are two main reasons why we chose a compressible model to describe all phases. From a numerical point of view first, because of the monolithic approach followed here, having models of a similar nature allows a more straightforward coupling strategy. The second physical reason has to do with the flow conditions which are likely to induce sufficient density variations such that an incompressible model for the liquid phase does not hold. Those two reasons are further discussed in Section 2.3.1.

Requirements While the perfect gas law can be used for the gas, pertinent closure models are required to fit the equations to the stiff behavior of the condensed phase. The use of such specialized models put some challenging constraints on the numerical methods which need to be augmented by dedicated techniques to ensure stability and accuracy of the scheme.

Treatment of the gas–condensed phase interface

Possible approaches The two main prototypes to handle phase interface are the sharp and diffuse interface methods. In the former case, the interface is considered as a singular surface across which jump conditions apply. In the latter case, the interface is smeared over a heterogeneous region in which the different phases mix together. Because of the compressible models considered here, such diffusive interface approach would require the derivation of volume-averaged equations.

The choice of a sharp interface representation Once again, we can justify this choice based on modeling and numerical arguments. From modeling perspective, the interface topology in the flow regions of interest between the gas and the condensed phase is expected to be such that a separated or mixed multiphase flow configuration will be present. This makes it possible to consider a sharp interface treatment, which is superior to the diffuse thanks to the rigorous and exact imposition of the jump conditions across the interface. More details are available in Section 3.1.2. From a numerical point of view, the huge properties jumps that exist between the gas and the condensed phases can be very challenging to capture in a diffuse interface setting without generating spurious oscillations. This is avoided with an appropriate sharp interface representation, as explained in Section 8.1.4.

Requirements To benefit from the high level of fidelity offered by the sharp interface treatment, the derivation of the jump conditions based on physical arguments is necessary. In particular, those jumps shall include the interfacial phenomena of interest, namely the transfer of momentum to account for the shear removal of the liquid layer, the surface tension effects and the evaporative mass flux. The numerical challenge is to enforce the jumps into the discretized form on an interface whose motion depends on the flow quantities from either side of the interface.

Motion of the gas–condensed phase interface

Possible approaches Because we are considering a sharp interface treatment of the gas–condensed phase interface via a so-called local instant formulation, the position and motion of the interface are not part of the fluid system and need to be explicitly determined. This can be done either by deforming the mesh such that mesh faces coincide with the interface location or by solving an additional transport equation for an auxiliary field from which the interface position can be inferred. In this second category, the advected field can either explicitly represent the interface (e.g. via Lagrangian marker-particles) or it can implicitly contained the interface definition through a scalar quantity.

The choice of a fixed mesh with auxiliary field In this thesis, we decided to go for a fixed mesh discretization accompanied by the transport of an auxiliary scalar field which implicitly identifies the interface at each time step. The choice of a fixed mesh approach over a moving one is mainly dictated by the simplicity of implementation. As far as the implicit representation is concerned, it has been chosen for the inherent ability of such methods to handle complex interface deformations and topological changes. More information is provided in Section 6.1.2.

Requirements The accuracy with which the interface geometry and its associated quantities such as normal and curvature are represented should be

consistent with the one of the fluid quantities. Moreover, a strategy to couple the additional transport equation for the scalar field with the fluid equations needs to be devised. This applies in particular to the exchange of information required by each of the two solvers.

Treatment of the liquid–solid phase interface

Possible approaches Similarly to the gas–condensed phase interface, sharp and diffuse approaches are possible.

The choice of a diffuse interface representation In contrast to the the gas–condensed phase interface, the one separating the liquid and solid regions is not clearly defined. Indeed, for metallic alloys, a mixture region made of liquid and solid particles is present (the mushy zone), while for ceramics and glassy materials, a zone of progressive softening can be identified. Thus, the choice of a diffuse interface representation appears natural. More comments can be found in Section 4.1.3.

Requirements In the diffuse interface approach, the difficulties of the sharp setting of representing and tracking the discontinuous interface is avoided. Instead, they are shifted to the modeling of source terms representing the interactions between both phases and their numerical treatment. In our case, those terms should model the phase transformation between the liquid and solid phases and the associated transfer of momentum and energy. There is no need to consider an additional method to describe the interface motion here since its explicit position is not required.

Discretization of the governing equations

Possible approaches Finally, the last important choice to be made in assembling our numerical tool relates to the discretization techniques, i.e. to the way the continuous models can be approximated by numerical algorithms. In Computational Fluid Dynamics (CFD), it is common to distinguish between low-order and high-order methods. In the first category, the Finite Volume Method (FVM) is by far the most widely used. In the second class, Finite-Element (FE) based methods continue to arouse great interest in the community.

The choice of a high-order method One of the most striking features of this work is that it is based on a FE-based high-order discretization method, with a local representation of the solution per element. This brings unmatched accuracy per degrees of freedom compared to low-order methods while preserving the flexibility of handling complex geometries. Additional benefits are discussed in Section 5.1.2. The well-known advantages of high-order methods to discretize the equations in the bulk of a phase are at least as valuable when

it comes to discretize flows with sharp interfaces, as detailed in Section 8.1.4. This is an extra motivation for this choice.

Requirements The advantages of high-order accuracy come at the expense of a decrease of robustness of the global scheme. One of the main causes for this is linked to the presence of under-resolved flow features that can degrade the theoretical error behavior and stability of high-order schemes. That is why special measures need to be taken.

Verification and validation of the numerical tool

Possible approaches Once the numerical tool has been assembled, a crucial step is to assess the implementation of the numerical methods and the suitability of the physical models in order to demonstrate the predictive capabilities of the numerical tool. These assessment strategies fall into two complementary categories: verification and validation. Validation evaluates the accuracy of the physical models implemented in the code for a given application. It is typically performed through comparison with experimental data. On the other hand, verification is a purely mathematical enterprise which attempts to evaluate if the equations are correctly solved. It is commonly done by reproducing benchmark literature test cases or by building an analytical reference solution following the so-called Method of Manufactured Solutions (MMS).

Combination of benchmark test cases, manufactured solutions and simplified experiments In this work, we combined the use of benchmark reference and manufactured solutions to verify our implementation on reference solutions. We also conducted exploratory experiments which isolate some of the phenomena we want to study in this work.

Requirements For the comparison with benchmark test cases, we should be careful to select test cases in line with the governing equations solved by our solver and to adapt, if necessary, the available semi-analytical solutions that can be found in the literature. Concerning the manufactured solutions, their construction is made challenging by the multiphase nature and complex equations involved. Finally, the design of the experiments should be aligned with the prediction capabilities of the developed code.

1.4 Objectives and outline of the thesis

Objectives

We should start by recalling that the proposed framework is not appropriate, considering the current computational capabilities, to the simulation of a whole

re-entering spacecraft with all the phenomena identified previously. It is rather intended for the fully resolved Direct Numerical Simulations (DNS) of local ablation phenomena which may considerably support the development of new Design for Demise technologies, as discussed in Section 1.1.3. Following the previously elaborated modeling and numerical ingredients of our solver as well as the requirements they imply, we can now precisely state the objectives of this thesis.

Overall objective: **Develop a unified high-order discontinuous Galerkin interface-resolved solver to simulate ablation by melting**:

- ***Obj.1 - Modeling*** **Derive compatible models to describe the gas, liquid and solid phases, as well as their coupling**:
  - ***Obj. 1.a*** Lay down general jump conditions at the gas-liquid interface taking into account all the considered phenomena via suitable constitutive laws.
  - ***Obj. 1.b*** Obtain compressible volume-averaged equations with appropriate closure to model the liquid-solid phases including phase transformation.
- ***Obj.2 - Numerical methods*** **Extend a discontinuous Galerkin framework to a compressible two-phase sharp interface solver**:
  - ***Obj. 2.a*** Improve the accuracy of the interface-capturing method and couple it with the fluid solver.
  - ***Obj. 2.b*** Adapt the spatial and temporal schemes to the unfitted discretization of moving interface problems.
  - ***Obj. 2.c*** Devise a coupling strategy enforcing jump conditions between viscous compressible phases and preserving the high-order features of the method.
- ***Obj.3 - Verification and experimental testing*** **Design test cases to verify the implementation and an experimental setup to reproduce specific ablation phenomena**:
  - ***Obj. 3.a*** Build up a manufactured solution framework generating reference data satisfying prescribed jump conditions.
  - ***Obj. 3.b*** Conceive an experimental setup isolating some of the phenomena of interest in this work.

From the important assumptions made in Section 1.3.1, one can legitimately wonder how the proposed tool will achieve its claimed "high-fidelity" level. We do not pretend that the methods and models investigated as part of this work will be able to shed the light on all the driving ablation phenomena for which a better understanding reveals necessary. We can only testify to their relevance to certain isolated phenomena, highlighted in Figure 1.9. In this regard, the research undertaken in this thesis can be considered as laying the foundation for extensibility to models with increasing level of complexity, toward an accurate description of the ablation by melting process. It is instructive to note that since

the numerical methods developed in this work are meant to accommodate new models, the solver could equally be applied to any physical process involving separated two-phase flows, with or without phase transformations. This observation leads us to formulate a final transversal objective for this research.

Transversal objective: **Assess the capabilities of interface-resolved high-order methods for the simulation of multiphase flows in a regime spanning from almost incompressible to compressible**

Outline

The rationale behind the structure of this thesis is the partitioning of the whole numerical framework into related yet quite independent building blocks that are presented in each chapter. This makes it possible to devote part of each chapter, or even sections, to a focalized state-of-the-art review. Based on this review, the concepts and methods required for the development of each block are then exposed. The chapters are organized into three main parts, each one of them corresponding to one of the objectives stated above. The first and last parts contain the introduction and conclusions chapters, respectively. A schematic of the thesis structure is given in Figure 1.10 and detailed in what follows.

Part II – Physical modeling Chapter 2 is devoted to basic concepts of continuum mechanics from which the Navier-Stokes equations are derived. The last part explores the ways to equip the standard equations with models describing the liquid phase behavior. In Chapter 3, a review of the existing multiphase formulations is first provided. The general jump conditions are then inferred from the concepts introduced in the previous chapter and specialized transport theorems. They are finally specified to the viscous compressible system and different constitutive equations for the gas-liquid interface are discussed. Chapter 4 starts with an overview of the liquid-solid phase change modeling. The general volume-averaged equations are then presented before being transformed to the homogeneous mixture model with phase transformation employed in this work.

Part III – Numerical methods The first part of Chapter 5 is dedicated to an introduction to variational high-order methods. The variational form of the Navier-Stokes system discretized by means of a nodal discontinuous Galerkin (DG) method is then defined. The chapter closes by giving implementation details about the adaptation of the method to treat stiff condensed phase models. The focus of Chapter 6 is turned to the interface capturing method developed in this thesis, namely the level-set method. It addresses successively the DG discretization of the level-set transport equation, the reinitialization and velocity extension procedures, the narrow band strategy and the enforcement of a continuous interface. Chapter 7 continues with an overview of the curvature filtering techniques for the level-set method in a high-order context. A general

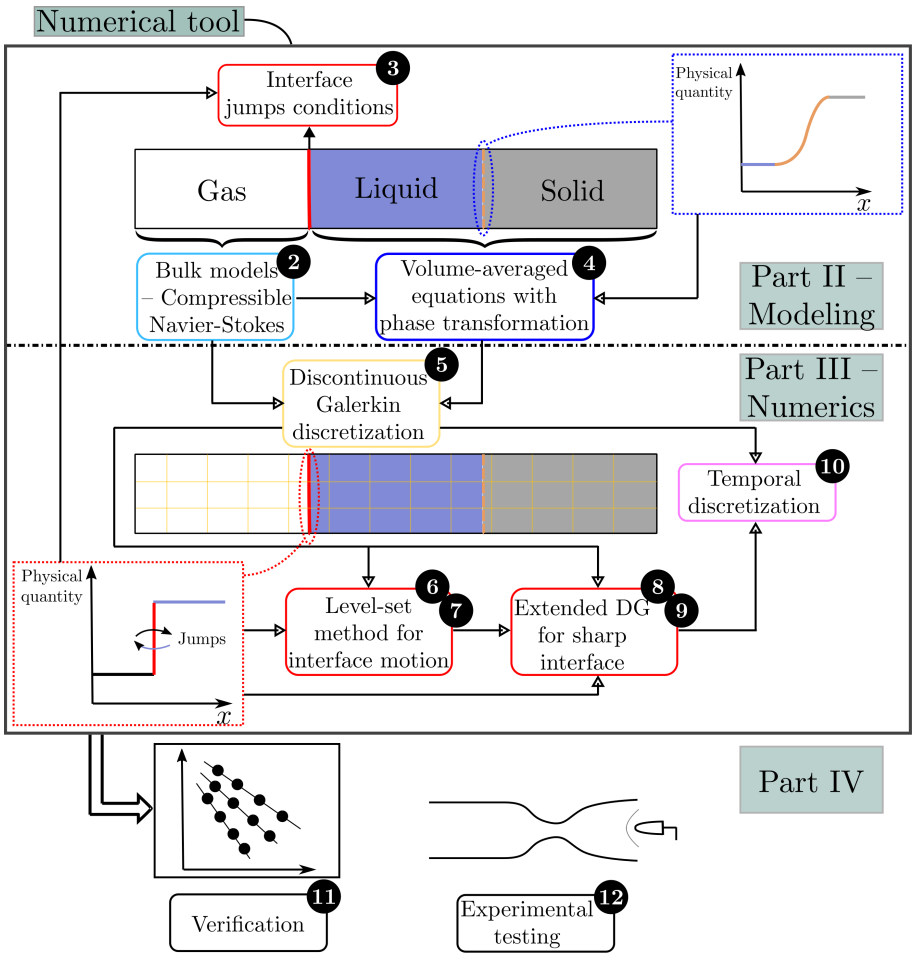


Figure 1.10 / Illustration of the thesis structure with the chapter numbers and their links.

flow chart composed of decisive steps in the curvature estimation process is then proposed, and the ability of several methodologies to provide robust and stable surface tension computation are evaluated. Chapter 8 looks at the extension of the DG method to unfitted sharp interface discretizations. A review of the existing approaches, the challenges and the motivations to rely on the extended DG method in this work are first exposed commented. The different techniques to handle the challenges posed by the method are then detailed before illustrating some implementation aspects via concrete examples. Chapter 9 starts with an overview of the coupling approaches for compressible two-phase flows. The coupling strategy to enforce the jump conditions through specific interfacial fluxes is then explained. All the pieces from this chapter and the previous are then put together to analyze the resulting variational formulation. Finally, the part III is concluded by Chapter 10 which presents all the numerical aspects linked to the temporal discretization. It begins by describing implicit time stepping applied to standard discontinuous Galerkin discretizations. It then elaborates on the time integration for moving interface problems. Lastly, the coupling procedure between the level-set and the fluid solvers is given, which serves as a recapitulation of the various developed methods.

Part IV – Verification and experimental testing Chapter 11 demonstrates the successful application of the proposed methods on reference test cases and manufactured solutions. A description of the experimental setup and the obtained flow visualization results are the subject of Chapter 12.

Part V – Conclusions Finally, Chapter 13 summarizes major results and developments achieved in this thesis and gives perspectives of potential and promising research fields in the future.

Rest of this page intentionally left blank.

Part II

Physical modeling

2

Equations in the bulk of each phase

Some parts of Section 2.3 were presented in Henneaux et al. (2023c).

2.1 Fundamentals of continuum mechanics 45
2.1.1 The spectrum of descriptions and the continuum hypothesis 45
2.1.2 Kinematical descriptions 48
2.1.3 Physical dynamics 56
2.2 The compressible Navier-Stokes equations and their closure . . 57
2.2.1 Derivation of conservation laws from first principles . . 57
2.2.2 Closure of the equations by constitutive models 61
2.3 Specific models for the liquid phase 66
2.3.1 Stiff compressible model 66
2.3.2 Thermodynamic and transport properties 70
2.3.3 Derivation of a thermodynamic consistent EOS 74
2.4 Summary of the equations implemented in this work 78

2.1 Fundamentals of continuum mechanics

2.1.1 The spectrum of descriptions and the continuum hypothesis

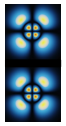
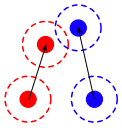
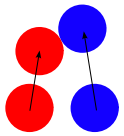
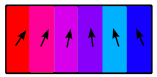
For any physical system for which we want to predict the behavior, three essential elements are required: (i) the choice of the level of description, (ii) the definition of physical quantities describing the state of the system, (iii) the derivation of evolution equations for those quantities. A broad overview of the

existing physical modeling approaches is given in Table 2.1[1]. A step-by-step animated description of those different approaches is provided by braintruffle (2022). In this work, we rely on the field of continuum mechanics and consider the solution of the Navier-Stokes equations by means of Computational Fluid Dynamics (CFD) methods. The reasons for this choice are threefold:

- *Level of description*: for the prediction of the ablation of an object from a macroscale perspective, the use of molecular dynamics and *a fortiori* quantum mechanical models appears clearly to be inappropriate. The details those approaches could deliver at a microscopic to nanoscopic scales are indeed not relevant for that objective. At the other end of the spectrum, it was explained in Section 1.4 that the methods developed in this work aim at improving the fidelity which with engineering models currently address the problem. Those are therefore not sufficiently reliable for the intended purpose.
- *Flow regime*: as mentioned in Section 1.2, this thesis focuses on the portion of the re-entry trajectory during which the gas flow around the object can be considered to be in the continuum regime. Concerning the condensed phase, the continuum hypothesis is obviously acceptable. If the rarefied regime was of interest, kinetic-based models combined with Direct Simulation Monte Carlo (DSMC) would have been a more suitable choice. This approach is adopted in Bariselli (2021).
- *Computational feasibility*: the last reason which is related to the two first ones is about the computational cost. DSMC methods become very computationally demanding for low Knudsen numbers. Solving the Schrödinger equation or applying Molecular Dynamic Simulation (MDS) is simply out of reach nowadays to study the type of problem envisaged in this work.

[1]Note that we do not include in this table fields involving relativistic effects such as relativistic mechanics and quantum field theory.

Table 2.1 / Spectrum of physical modeling approaches.

	Discrete/Particle-based			Continuum	
					
Field	Quantum mechanics	Molecular dynamics	Kinetic theory	Continuum mechanics	Engineering models
Description	Elementary particles	Atoms/molecules	Set of molecules, averaging in phase space	Continuous quantities, averaging in position space	Continuous quantities
Quantities	Wave function	Individual positions/velocities	Probability density function	Macroscopic	Macroscopic
Theory	Quantum mechanics	Quantum & Classical mechanics	Statistical mechanics	Classical mechanics	Surrogate modeling
Equations	Schrödinger eqs	Newton's 2nd law, with quantum potentials	Boltzmann eqs	Mass conservation, Newton's 2nd law, 1st law of thermod.	Correlations
Numerics (examples)	Finite Differences (FD), Quantum Monte Carlo (QMC)	Molecular Dynamics Simulation (MDS)	Direct simulation Monte Carlo (DSMC)	Computational Fluid Dynamics (CFD)	(Semi)-analytical expressions

2.1.2 Kinematical descriptions

The elements exposed in this section are a summary mainly combining the extensive theory presented in Knobbe (2010), Gamnitzer (2010) and Abeyaratne (2012). The point of view adopted here, which is intended to be general, borrows a number of concepts to the Arbitrary Lagrangian-Eulerian (ALE) community. The reader is referred to references such as Donea et al. (2004), Askes et al. (2004) or Barlow et al. (2016) for an overview of the ALE concepts. The notations employed in this section are supported by Figure 2.1.

Frame of reference Kinematics is the science of motions and deformations, which is applied here to continua. The causes of motions and deformations is the essence of dynamics which is left for the next section. The description of the phenomena studied in this work requires a mathematical formulation that depends both on space and time. Space dependence will be formulated in a d-dimensional Euclidean vector space $\mathbb{E}^d$. In this space, any vector $\mathbf{v} \in \mathbb{E}^d$ and tensor $\underline{\boldsymbol{T}} \in \mathbb{E}^d \otimes \mathbb{E}^d$ can be decomposed into components $v_i \in \mathbb{R}$ and $T_{ij} \in \mathbb{R}$, respectively, by using a set of linearly independent orthonormal base vectors $\{\hat{\mathbf{e}}_i\}$ such that

$$\mathbf{v} = v_i \hat{\mathbf{e}}_i \quad \text{and} \quad \underline{\boldsymbol{T}} = T_{ij} \hat{\mathbf{e}}_i \hat{\mathbf{e}}_j \,, \quad \text{where} \quad i,j = 1,\dots,n \,, \tag{2.1}$$

and the Einstein summation convection is used. The Euclidean point space $\mathcal{E}$ is also introduced such that each point can be identified by its coordinates. It is assumed that transient problems have to be evaluated over a closed time interval, i.e.

$$\Delta = [t_0,\, t_{end}] \subset \mathbb{R} \quad \text{with} \quad t_{end} > t_0 \,. \tag{2.2}$$

Treating separately space and time as independent entities leads to the concept of Newtonian space-time continuum[2], $\mathfrak{N}$. The mapping establishing the correspondence between $\mathfrak{N}$ and the space and time entities introduced before, viz.

$$\mathfrak{E} : \mathfrak{N} \to \mathbb{E}^d \otimes \Delta \,, \tag{2.3}$$

defines the *frame of reference* with respect to which the governing equations will be expressed. In this work, the description of the positions inside that frame of reference will be restricted to inertial Cartesian coordinate systems[3].

Fluid system and properties The continuum hypothesis underpinning all models presented in the following implies that any *system* $\mathcal{S}$ to be observed is

[2] Newtonian space-time continuum can be put into perspective with the four-dimensional space-time manifold in the Einstein's theory of relativity where space and time are dependent upon the reference frame's motion. Because of the targeted applications in this work, relativistic effects are not considered.

[3] We can note that true inertial systems are idealized constructs because various factors such as the rotation of the Earth or gravitational forces can introduce small accelerations. However, the scales over which the phenomena studied in this work will occur are such that this kind of external perturbations can be reasonably disregarded.

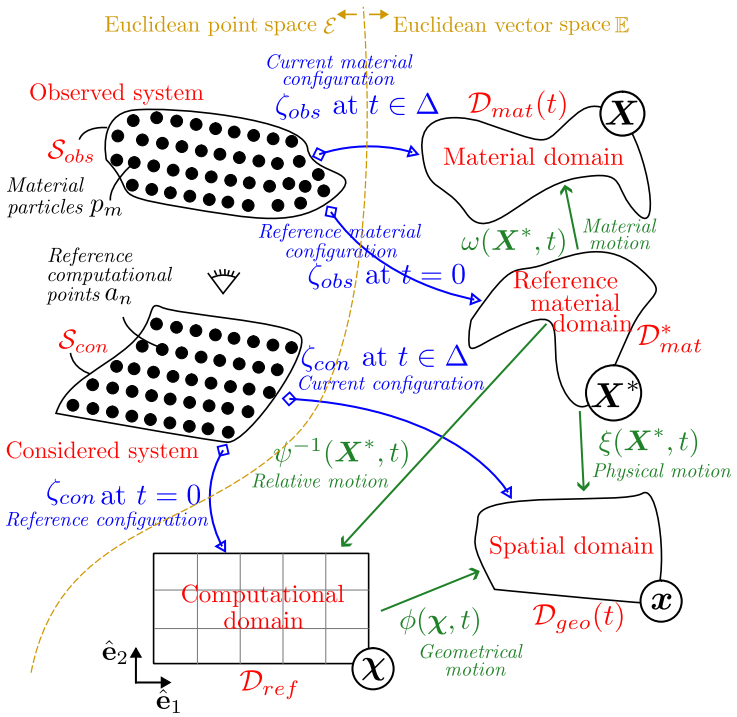


Figure 2.1 / Diagram showing the relations between the different systems and domains considered for the kinematical description. Inspired from Knobbe (2010, Fig. 3.2).

composed by an infinite set of *material particles* [4] p,

$$\mathcal{S} = \{p_m\}\,, \quad m = 1, \dots, \infty\,, \quad p_m \in \mathcal{E}.$$

In a fluid flow, a particle can be seen as a collection of elementary particles (e.g. molecules) occupying a small volume. The number of those constituent particles is such that fluid property can be defined by averaging over the fluid particle volume without molecular fluctuations becoming apparent. At this stage, it is instructive to remark that there are two types of physical characteristics associated with a fluid system. *Intensive properties* like density or temperature are linked to individual fluid particles while *extensive properties* like mass or internal energy are related to "parts of the fluid system" and the amount of considered fluid. To make the link between both, any intensive fluid property associated to a given fluid particle p_m is specified by a continuous tensor field φ_i that can be function of time,

$$\varphi_i \colon \mathcal{S} \times \Delta \to \mathcal{U}\,, \tag{2.4}$$

where $\mathcal{U}$ is a normed vector space that depends on the property considered: $\mathcal{U} = \mathbb{R}$ for a scalar, $\mathcal{U} = \mathbb{E}^d$ for a vector and $\mathcal{U} = \mathbb{E}^d \otimes \mathbb{E}^d$ for a tensor (Knobbe (2010, Sec. 2.2)). An extensive property over a specified fluid system can then be determined by the integration of the corresponding field density φ_i, which defines a time-dependent functional φ_e

$$\varphi_e(t) = \int_{\mathcal{S}} \varphi_i(p,t)\, \mathrm{d}p \colon \mathcal{S} \times \Delta \to \mathcal{U}. \tag{2.5}$$

By writing Equation 2.4 and Equation 2.5, we have implicitly assumed that there is a correspondence between any material particle p_m located in a given system $\mathcal{S}$ at an instant $t \in \Delta$ and the spatial Euclidean space $\mathbb{E}^d$. This correspondence is called a *configuration* ζ. This is a one-to-one mapping between the set of particles defining the fluid system and the space $\mathbb{E}^d$,

$$\zeta \colon \mathcal{S} \to \mathbb{E}^d\,, \quad \mathbf{r} = \zeta(p_m)\,, \quad \text{with} \quad p_m \in \mathcal{S}\,,$$

where the vector $\mathbf{r}$ identifies the position of a particle with respect to $\mathbb{E}^d$. Applying this mapping to the whole fluid system leads to the definition of a fluid *domain* $\mathcal{D}$ which can depend on time:

$$\mathcal{D} = \zeta(\mathcal{S}, t)\,.$$

Since the equations to be implemented in a code are expressed in the space $\mathbb{E}^d$, it will be more convenient to rewrite Equation 2.5 as

$$\varphi_e(t) = \int_{\zeta(\mathcal{S},t)} \varphi_i(\mathbf{r},t)\, \mathrm{d}\mathbf{r}\,. \tag{2.6}$$

[4] The term "particle" refers in the context of continuum mechanics to an abstract entity, different from the concept of a particle with a mass.

Let us note that a domain can be in two different configurations while still occupying the exact same spatial volume. This would be the case for example for a circular domain undergoing a rigid-body rotation: the domain would occupy the same volume before and after the rotation but the associated particles would be at different locations. While a fluid domain can be seen as a concrete geometrical volume defined by spatial coordinates, a fluid system is an abstract conception. Manipulating both entities through the notion of configuration will help us to introduce later the concepts of observed and considered systems, reference and current configurations, as well as material, computational and spatial domains.

Configurations and domains A system $\mathcal{S}$ can be of two types: closed or open. A closed system comprises the same set of material particles at any time; it moves and deforms without any particle entering or leaving it. On the contrary, the transfer of matter is allowed through the boundaries of an open system. The physical integral conservation laws that will be described in Section 2.2 are based on the time rate of change of extensive fluid properties over moving closed systems, often referred as *material volumes*. At the same time, an extensive fluid property can be evaluated by considering any arbitrary set of particles such that the integration domain in Equation 2.6 does not have to be a closed system. Those two considerations suggest the definition of two distinct systems (Knobbe (2010, Sec. 3.1)):

- A first physical system made up of material points. It permanently contains the same material particles that are tracked as the system moves through space. It is the *observed system* with respect to which the physical laws will be derived. A material point is still called p_m and the system is noted $\mathcal{S}_{obs} = \{p_m\}$.
- A second computational "system" consisting of a fixed set of arbitrary reference points. It is from this system, whose motion is completely decoupled from the first one, that the physical problem is observed. It is also with respect to this *considered system* that the physical properties will be expressed. A reference point is called a_n and the system is noted $\mathcal{S}_{con} = \{a_n\}$.

It is with respect to the *considered* system that local motions and deformations[5] can be mathematically formulated. In order to do so, we must necessarily consider (at least) two configurations defining two fluid domains (Abeyaratne (2012, Sec. 1.2)):

- The *computational* or *referential* domain is linked to the considered system through a fixed *reference configuration* which is usually chosen as the initial configuration,
$$\mathcal{D}_{ref} = \zeta_{con}(\mathcal{S}_{con}, t = 0) .$$

[5] The motion and deformation of a fluid parcel can be locally decomposed into four elementary movements: rigid-body translation, rigid-body rotation, isotropic expansion and pure straining without change of volume (Papanastasiou et al. (2000, Sec. 2.4)).

The goal of the reference configuration is to identify/designate a fixed set of reference computational points (e.g. the grid points) in the Euclidean vector space $\mathbb{E}^d$. Thus, a given reference point $a_n \in \mathcal{E}$ has a reference position $\mathbf{r}_0 = \zeta_{con}(a_n, t = 0) \in \mathbb{E}^d$.

- The *spatial* or *geometrical* domain is obtained through the possibly time-dependent *current configuration*,

$$\mathcal{D}_{geo} = \zeta_{con}(\mathcal{S}_{con}, t) \,.$$

This second domain is thus formed by following the points corresponding to the reference configuration and the displacement of a given point can be obtained by subtracting $\mathbf{r}_0$ from the current position $\mathbf{r} = \zeta_{con}(a_n, t)$: $\mathbf{r} - \mathbf{r}_0$. A prerequisite to express the displacement in this way is that the reference configuration is at one particular time $t \in \Delta$ collocated with the current one, hence the usefulness of selecting the reference configuration as the initial one.

A reference configuration is also introduced to map the physical observed system to the initial *material* domain,

$$\mathcal{D}^*_{mat} = \zeta_{obs}(\mathcal{S}_{obs}, t = 0) \,,$$

as well as a current configuration to track the fixed set of material particles as they move through space,

$$\mathcal{D}_{mat} = \zeta_{obs}(\mathcal{S}_{obs}, t) \,.$$

The main kinematical formulations Now that the various mathematical concepts have been introduced, we can turn our attention to the different ways in which the functional $\varphi_e(t)$ in Equation 2.6 can be specified. This translates into the selection of (i) the integral limits and (ii) the representation of the integrand (Knobbe (2010, Sec. 3.4)). Choosing the integral domain implies the selection of the configuration ζ and the considered system $\mathcal{S}$. This selection determines the point of view with which the physical system is observed. The four most common approaches are commented here:

- *Lagrangian*: in the Lagrangian perspective, for every time $t \in \Delta$, the same set of moving material points forming the observed system $\mathcal{S}_{obs}$ is tracked. This is accomplished by taking the computational domain as being equal to the initial material domain,

$$\mathcal{D}_{ref} = \mathcal{D}^*_{mat} \,.$$

This choice allows to consider the same set of moving material points from a time-independent reference configuration of the observed system. In other words, the extensive physical properties of the flow are determined by integrating the corresponding field density over $\mathcal{D}^*_{mat}$. The local

intensive property is then represented as

$$\varphi_i = \varphi_i(\mathbf{X}^*, t)\Big|_{\mathbf{X}^*} \triangleq \varphi_{i,\mathbf{X}^*} \quad \forall \mathbf{X}^* \in \mathcal{D}^*_{mat}\,,$$

where the position vector $\mathbf{X}^* = \zeta_{obs}(p_m, t = 0)$ refers to the reference Lagrangian coordinates in the following and the $\bullet\big|_{\mathbf{X}^*}$ notation indicates that the function is evaluated keeping a constant $\mathbf{X}^*$.

- *Eulerian*: the Eulerian viewpoint is radically different as the moving material system is observed from a non-moving region. Stated differently, the observed physical problem is moving "over" the considered geometrical problem so that at two distinctive times, different moving subregions of the observed system are considered. It is therefore not possible to track the complete initial set of material particles, which means that there is an efflux of those points in this fixed considered region. In this case, there is no direct correspondence between the computational and material domains. Instead, the computational domain is equivalent to the spatial domain,

$$\mathcal{D}_{ref} = \mathcal{D}_{geo}\,,$$

and the representation of the local intensive property is given as

$$\varphi_i = \varphi_i(\mathbf{x}, t)\Big|_{\mathbf{x}} \triangleq \varphi_{i,\mathbf{x}} \quad \forall \mathbf{x} \in \mathcal{D}_{geo}\,,$$

where the position vector $\mathbf{x} = \zeta_{con}(a_n, t = 0)$ denotes the Eulerian coordinates.

- *Total ALE*: The Arbitrary Lagrangian Eulerian (ALE) concept combines the features of the two above classic approaches into a hybrid approach. The first part of the name (arbitrary Lagrangian) refers to the selection of a set of moving computational points whose motion is arbitrary (Knobbe (2010, Sec. 3.1.2)). In the total ALE, the reference configuration is taken as the computational domain so that the same set of points is considered at every time (i.e. the integral limits of $\varphi_e(t)$ are fixed by the computational domain) and the representation of the intensive property follows as

$$\varphi_i = \varphi_i(\boldsymbol{\chi}, t)\Big|_{\boldsymbol{\chi}} \triangleq \varphi_{i,\boldsymbol{\chi}} \quad \forall \boldsymbol{\chi} \in \mathcal{D}_{ref}\,,$$

where the position vector $\boldsymbol{\chi} = \zeta_{con}(a_n, t = 0)$ is the ALE coordinates. The Eulerian characteristic of the ALE concept is used to observe a part of the moving material domain from a considered system. Thus, the actual part of the material system that can be observed is governed by the considered region. Behind this viewpoint lies a postulate of arbitrary collocation according to which, at some instant in time, the spatial and material domains coincide (Knobbe (2010, Sec. 3.1.2)). The same postulate is also required for the Eulerian approach. It is easily satisfied by choosing the

material, the computational and the spatial domains to be collocated at the initial time. Since the time evolution of the spatial domain is not determined by the material domain, there is a geometrical problem to be solved, in addition to the physical problem.

- *Updated ALE*: in the updated ALE approach (see e.g. Askes et al. (2004, Sec. 2.2)), the observer is not located in the reference but in the current configuration of the moving considered system, i.e. in $\mathcal{D}_{geo}$ with the representation of φ_i given by

$$\varphi_i = \varphi_i(\boldsymbol{\chi}, t)\Big|_{\boldsymbol{\chi}} \triangleq \varphi_{i,\boldsymbol{\chi}} \quad \forall \boldsymbol{\chi} \in \mathcal{D}_{geo}\,.$$

  Since the spatial domain coincides with the material domain initially, this approach allows to extend the capabilities of the total ALE: in addition to handle problems with moving boundaries, the updated ALE also offers the possibility to adapt the computational mesh based on the simulated local phenomena.

Let us note that the major advantage of formulating the problem on the computational domain for the three first approaches described above is that it is obtained from a reference configuration which is a time-independent map. Hence, the computational mesh, resulting from the spatial discretization of the computational domain, does not change during a transient simulation (Knobbe (2010, Sec. 3.2)).

Material time derivative and transport theorem In order to obtain the time evolution of a physical problem (see Section 2.2), the material/substantial time derivative of the selected formulation of the functional $\varphi_e(t)$,

$$\frac{D\varphi_e(t)}{Dt} \triangleq \frac{\partial \varphi_e(t)}{\partial t}\Big|_{\mathbf{X}} = \frac{\partial}{\partial t}\int_{\mathcal{D}_{mat}(t)} \varphi_{i,\mathbf{X}}\,\mathrm{d}\mathbf{X}\,, \qquad [2.7]$$

has to be evaluated because physical laws are formulated with respect to the current moving material domain. In Equation 2.7, the position vector in the current material domain is $\mathbf{X} = \zeta_{obs}(p_m)$ and the representation of φ_i is given by

$$\varphi_i = \varphi_i(\mathbf{X}, t)\Big|_{\mathbf{X}} \triangleq \varphi_{i,\mathbf{X}} \quad \forall \mathbf{X} \in \mathcal{D}_{mat}\,.$$

However, because the spatial dependence of φ_i depends on the considered system in the most general case (and not the material one), a transformation from material to computational time derivative is required. This conversion can be operated starting by the fact that a quantity φ_i to be observed carries equal values independent of the viewpoint, identified by $\varphi_{i,\mathbf{X}^*}$, $\varphi_{i,\boldsymbol{\chi}}$, $\varphi_{i,\mathbf{x}}$, $\varphi_{i,\boldsymbol{\chi}}$ before. The links between the different viewpoints are contained in the four mappings outlined in Figure 2.1, $\omega(\mathbf{X}^*, t)$, $\phi(\boldsymbol{\chi}, t)$, $\xi(\mathbf{X}^*, t)$ and $\psi^{-1}(\mathbf{X}^*, t)$, defining the material, the geometrical, the physical and the relative motions, respectively

(Gamnitzer (2010, Sec. 2.1.1)). In particular, we have the relation

$$\varphi_{i,\mathbf{X}^*} = \varphi_i(\mathbf{X}^*,t)\Big|_{\mathbf{X}^*} = \varphi_i(\psi^{-1}(\mathbf{X}^*,t),t)\Big|_{\boldsymbol{\chi}} = \left(\varphi_i\Big|_{\boldsymbol{\chi}} \circ \psi^{-1}\right)(\mathbf{X}^*,t)\,. \qquad [2.8]$$

Applying the chain rule for differentiation to Equation 2.8, we can obtain that (Benedikt Schott (2017, Sec. 2.1.2))

$$\frac{\partial \varphi_{i,\mathbf{X}^*}}{\partial t} = \frac{\partial \varphi_{i,\boldsymbol{\chi}}}{\partial t} + \frac{\partial \varphi_{i,\boldsymbol{\chi}}}{\partial \boldsymbol{\chi}}\mathbf{w}_{\mathbf{X}^*} = \frac{\partial \varphi_{i,\boldsymbol{\chi}}}{\partial t} + \frac{\partial \varphi_{i,\mathbf{x}}}{\partial \mathbf{x}}\mathbf{c}_{\mathbf{X}^*}\,, \qquad [2.9]$$

where $\mathbf{w}_{\mathbf{X}^*} = \frac{\partial \boldsymbol{\chi}}{\partial t}\Big|_{\mathbf{X}^*}$ is the velocity of a fixed material particle in the computational domain and $\mathbf{c}_{\mathbf{X}^*} = \frac{\partial \mathbf{x}}{\partial \boldsymbol{\chi}}\Big|_{\boldsymbol{\chi}}\mathbf{w}_{\mathbf{X}^*}$ is the relative velocity between the material particle velocity in the spatial domain and the grid velocity. The next steps in the conversion of Equation 2.7 involve

1. The change of coordinates to go from the time-dependent integral domain $\mathcal{D}_{mat}(t)$ to the constant reference material domain $\mathcal{D}^*_{mat}$[6].
2. The application of the postulate of arbitrary collocation to change $\mathcal{D}^*_{mat}$ to $\mathcal{D}_{ref}$.
3. The use of the chain rule formula Equation 2.9.

This procedure, which is not detailed here and can be found e.g. in Knobbe (2010, Sec. 3.2.2), leads to the derivation of the generalized ALE Reynolds transport theorem,[7]

$$\frac{D\varphi_e(t)}{Dt} = \int_{\mathcal{D}_{ref}} \frac{\partial \varphi_{i,\boldsymbol{\chi}}}{\partial t}(\boldsymbol{\chi},t) + \nabla_{\boldsymbol{\chi}} \cdot \left(\varphi_{i,\boldsymbol{\chi}}\mathbf{w}_{\boldsymbol{\chi}}\right)(\boldsymbol{\chi},t)\,\mathrm{d}\boldsymbol{\chi}\,, \qquad [2.10]$$

$$= \int_{\mathcal{D}_{ref}} \frac{\partial \varphi_{i,\boldsymbol{\chi}}}{\partial t}(\boldsymbol{\chi},t)\,\mathrm{d}\boldsymbol{\chi} + \int_{\partial\mathcal{D}_{ref}} (\varphi_{i,\boldsymbol{\chi}}\mathbf{w}_{\boldsymbol{\chi}})(\boldsymbol{\chi},t)\cdot\mathbf{n}(\boldsymbol{\chi},t)\,\mathrm{d}\mathbf{S}_{\boldsymbol{\chi}}\,, \qquad [2.11]$$

where the second line is obtained by applying the Gauss-Ostrogradskii divergence theorem, with the normal vector to the domain $\mathbf{n}$, the computational domain velocity $\mathbf{w}_{\boldsymbol{\chi}}(\boldsymbol{\chi},t) = \mathbf{w}_{\mathbf{X}^*}(\psi(\boldsymbol{\chi},t),t)$ and the differential volume and surface elements of the domain $\mathcal{D}_{ref}$, $\mathrm{d}\boldsymbol{\chi}$ and $\mathrm{d}\mathbf{S}_{\boldsymbol{\chi}}$, respectively.

[6] To change an integral of $\varphi_{i,\mathbf{X}}$ over $\mathcal{D}_{mat}(t)$ to an integral of $\varphi_{i,\mathbf{X}^*}$ over $\mathcal{D}^*_{mat}$, the determinant of the gradient of deformation tensor appears. The time derivative can then be moved inside the integral and the time derivative of the determinant needs to be expressed. This is detailed e.g. in Gamnitzer (2010, App. A.1).

[7] The ALE Reynolds transport theorem for the updated ALE formulation is slightly different from the one written here since the integral limits are fixed by the time-dependent domain $\mathcal{D}_{geo}(t)$ and the spatial gradient is expressed with respect to the spatial coordinates and not the computational ones, see Knobbe (2010, Secs. 2.4 & 3.3.3)

2.1.3 Physical dynamics

Generic balance law Equipped with the kinematical formulation developed in Section 2.1.2, we can study what causes the time rate of changes of any extensive properties. This is expressed by global balance laws whose generic form can be stated as follows. In the frame of reference $\mathfrak{E}$, Equation 2.3, a global balance law states that

$$\frac{D\varphi_e(t)}{Dt} = \int_{\partial\mathcal{D}_{mat}(t)} \mathfrak{F}(\mathbf{X},t)\cdot\mathbf{n}(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} + \int_{\mathcal{D}_{mat}(t)} \mathfrak{P}(\mathbf{X},t)\,\mathrm{d}\mathbf{X}\,, \qquad [2.12]$$

where

- The material derivative on the left-hand side is the time rate of change of the physical state variable defined over a moving material volume $\mathcal{D}_{mat}(t)$ containing a fixed set of constitutive particles.
- The normal *flux* $\mathfrak{F}\cdot\mathbf{n}$ across the material volume boundaries $\partial\mathcal{D}_{mat}(t)$ contributes to the dynamic supply of φ_e through interactions with the external environment.
- The volumetric *production* $\mathfrak{P}$ can be decomposed into a local supply of φ_e from a source outside the considered material domain, noted $\mathfrak{P}_{out}$, and a production inside the material domain, $\mathfrak{P}_{in}$.

Now, replacing the left-hand side of Equation 2.12 by Equation 2.11 and applying the postulate of arbitrary collocation between the material and computational domains, we obtain the generic global balance law for the total ALE kinematical formulation,

$$\begin{aligned}\int_{\mathcal{D}_{ref}} \frac{\partial\varphi_{i,\boldsymbol{\chi}}}{\partial t}(\boldsymbol{\chi},t)\,\mathrm{d}\boldsymbol{\chi} &= \int_{\partial\mathcal{D}_{ref}} \left[\mathfrak{F}(\boldsymbol{\chi},t) - (\varphi_{i,\boldsymbol{\chi}}\mathbf{w}_{\boldsymbol{\chi}})(\boldsymbol{\chi},t)\right]\cdot\mathbf{n}(\boldsymbol{\chi},t)\,\mathrm{d}\mathbf{S}_{\boldsymbol{\chi}} \\ &\quad + \int_{\mathcal{D}_{ref}} \mathfrak{P}(\boldsymbol{\chi},t)\,\mathrm{d}\boldsymbol{\chi}\,, \qquad [2.13]\\ &= \int_{\mathcal{D}_{ref}} \nabla_{\boldsymbol{\chi}}\cdot\left[\mathfrak{F}(\boldsymbol{\chi},t) - (\varphi_{i,\boldsymbol{\chi}}\mathbf{w}_{\boldsymbol{\chi}})(\boldsymbol{\chi},t)\right] + \mathfrak{P}(\boldsymbol{\chi},t)\,\mathrm{d}\boldsymbol{\chi}\,. \qquad [2.14]\end{aligned}$$

We can notice the addition of a convective flux across the domain boundaries compared to the Lagrangian form, which emanates from the relative motion between the computational and material domains. It is often useful to derive an equivalent statement of such global principle in terms of a local differential equation that must hold for each $\boldsymbol{\chi}\in\mathcal{D}_{ref}$. It can be shown (see e.g. Abeyaratne (2006, Sec. 5.3)) that if the integrand in Equation 2.14 is *continuous* everywhere in $\mathcal{D}_{ref}$, the global statement can be localized to give

$$\frac{\partial\varphi_{i,\boldsymbol{\chi}}}{\partial t}(\boldsymbol{\chi},t) = \nabla_{\boldsymbol{\chi}}\cdot\left[\mathfrak{F}(\boldsymbol{\chi},t) - (\varphi_{i,\boldsymbol{\chi}}\mathbf{w}_{\boldsymbol{\chi}})(\boldsymbol{\chi},t)\right] + \mathfrak{P}(\boldsymbol{\chi},t) \quad \forall\boldsymbol{\chi}\in\mathcal{D}_{ref},\ \forall t\in\Delta\,. \qquad [2.15]$$

Specification to the Eulerian approach In this work, a pure Eulerian approach is chosen and can be obtained as a particular case of the ALE formulation derived above. The reasons for this choice are twofold: (i) the simplicity of the Eulerian formulation compared to the ALE one, (ii) the spatial and temporal discretizations techniques already implemented in the code in which the developments have been carried out during this thesis. As explained in Section 2.1.2, in the Eulerian approach, the reference computational domain corresponds to the geometrical domain so that in Equation 2.14 and Equation 2.15, the coordinates $\boldsymbol{\chi}$ will be replaced by $\mathbf{x}$ and the velocity,

$$\mathbf{w}_{\boldsymbol{\chi}}(\boldsymbol{\chi}, t) = \mathbf{v}_{\mathbf{x}}(\mathbf{x}, t) \triangleq \frac{\partial \mathbf{x}}{\partial t}(\xi^{-1}(\mathbf{x}, t), t)\Big|_{\mathbf{X}},$$

is now equal to the velocity of a given material particle with respect to the spatial domain. Equation 2.15 with those adaptations will be used in Section 2.2.

2.2 The compressible Navier–Stokes equations and their closure

After explaining the ways to describe motion in Section 2.1.2 and its relationship with surface and volume fluxes in Section 2.1.3, our focus is directed to the establishment of the equations governing the physical phenomena to be studied in this thesis. The equations presented in this chapter apply inside the bulk of the gas and liquid phases and are formulated following an Eulerian point of view. The interface jumps connecting those equations together are elaborated in Chapter 3.

2.2.1 Derivation of conservation laws from first principles

We start by establishing the equations based on three sound physical principles: conservation of matter, balance of linear and angular momentum, and first and second laws of thermodynamics. The equations in this section are independent of constitutive relations and therefore valid for all materials.

Mass conservation

The conservation of mass expresses that the total amount of matter contained in a moving material volume remains constant,

$$\frac{D}{Dt}\int_{\mathcal{D}_{mat}(t)} \rho(\mathbf{X}, t)\, \mathrm{d}\mathbf{X} = 0\,,$$

where ρ is the fluid *density* expressed in $\mathrm{kg/m^3}$. Based on Equation 2.15 written in Eulerian form, the corresponding local partial differential equation (also called the continuity equation) writes as

$$\frac{\partial \rho}{\partial t}(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot [\rho\, \mathbf{v}]\,(\mathbf{x},t) = 0\ . \qquad [2.16]$$

Remarks on partial densities balance Even if multispecies models are not considered in this work, it is instructive to write the balance equations that applies to each partial density ρ_s of the sth-species in the mixture. Contrary to the total mass, there is a species diffusion flux, $\mathfrak{F} = \mathbf{J}_s$, given by a generalized Fick's model, and an internal production or destruction of species, $\mathfrak{P} = \mathfrak{P}_{in} = \dot{\omega}_s$, generated by chemical reactions. This leads to

$$\frac{\partial \rho_s}{\partial t}(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot [\rho_s\, \mathbf{v}]\,(\mathbf{x},t) = -\nabla_{\mathbf{x}} \cdot \mathbf{J}_s(\mathbf{x},t) + \dot{\omega}_s(\mathbf{x},t)\,,$$

where $\mathbf{J}_s = \rho_s\, \mathbf{V}_s$, with the diffusion velocity $\mathbf{V}_s$ depending on the species concentration gradients, and the production rate $\dot{\omega}_s$ expressed based on a set of chemical reactions with their associated reaction rates. A module of the platform used in this work implements the multispecies equations. More information can be found in Pierre Schrooyen (2015).

Linear momentum balance

Linear momentum quantifies the amount of translational motion of matter in the system and is given by the product between the mass and the velocity of the system. The corresponding field density is then $\varphi_i = \rho\, \mathbf{v}$ in $\mathrm{kg/(m^2\,s)}$. According to Newton's second law, the change of linear momentum of a transported material volume equals the sum of the resulting forces on that volume. The resulting forces are a combination of contact forces on the volume boundaries and body forces acting on its interior. The contact forces are determined by the surface traction $\mathbf{t}_m(\mathbf{X},t,\mathbf{n})$ which, according to Cauchy's stress theorem, is related to a unique second-order tensor field $\underline{\boldsymbol{\sigma}} \in \mathbb{E}^d \otimes \mathbb{E}^d$, called the Cauchy stress tensor, via

$$\mathbf{t}_m(\mathbf{X},t,\mathbf{n}) = \underline{\boldsymbol{\sigma}}(\mathbf{X},t) \cdot \mathbf{n}(\mathbf{X},t)\,,$$

with the boundary normal vector $\mathbf{n}$ (Abeyaratne (2012, Sec. 4.5)). Based on Equation 2.12, we then have $\mathfrak{F} = \underline{\boldsymbol{\sigma}}$. Since there is no internal production of momentum, $\mathfrak{P} = \mathfrak{P}_{out} = \rho\mathbf{b}$, where $\mathbf{b}$ is the specific body force density. The original integral equation over a material volume reads

$$\frac{D}{Dt}\int_{\mathcal{D}_{mat}(t)} (\rho\, \mathbf{v})(\mathbf{X},t)\, \mathrm{d}\mathbf{X} = \int_{\partial\mathcal{D}_{mat}(t)} \underline{\boldsymbol{\sigma}}(\mathbf{X},t) \cdot \mathbf{n}(\mathbf{X},t)\, \mathrm{d}S_{\mathbf{X}} + \int_{\mathcal{D}_{mat}(t)} (\rho\, \mathbf{b})(\mathbf{X},t)\, \mathrm{d}\mathbf{X}\,,$$

and is translated into the following local Eulerian form

$$\frac{\partial(\rho\,\mathbf{v})}{\partial t}(\mathbf{x},t) + \nabla_{\mathbf{x}}\cdot\left[\rho\,\mathbf{v}\otimes\mathbf{v}\right](\mathbf{x},t) = \nabla_{\mathbf{x}}\cdot\underline{\boldsymbol{\sigma}}(\mathbf{x},t) + (\rho\,\mathbf{b})(\mathbf{x},t)\,. \qquad [2.17]$$

The stress tensor is at this stage still undefined and its expression will depend on the material under consideration. This is covered in Section 2.2.2.

Remarks on angular momentum balance The angular momentum (or moment of momentum) of a system around a fixed point is defined as the cross product between the linear momentum and the position vector $\mathbf{r}$ from the center of rotation, i.e. $\mathbf{r}\times(\rho\mathbf{v})$. The balance of angular momentum states that in an inertial frame, the resultant moment on any part of the system about a fixed point equals the rate of increase of its angular momentum about that same point. This results in a symmetry requirement for the Cauchy stress tensor, i.e.

$$\underline{\boldsymbol{\sigma}} = \underline{\boldsymbol{\sigma}}^T\,,$$

where $\underline{\boldsymbol{\sigma}}^T$ is the transpose of the stress tensor (Abeyaratne (2012, Sec. 4.6)).

Total energy balance

In physics and thermodynamics, there are many forms of energy, e.g. chemical, elastic, internal, kinetic, potential, etc. On the other hand, there is only one universal physical law for the total energy of the considered system. The first law of thermodynamics stipulates that in a closed system, total energy cannot be created or destroyed, but it can be transferred or transformed from one form to another. In engineering thermodynamics, the *total energy* per unit of mass, noted E and expressed in J/kg, is considered to be made up of three macroscopic contributions. The first one is the kinetic energy associated with the motion of the system as a whole and given by $e_{kin} = \frac{1}{2}\mathbf{v}\cdot\mathbf{v}$. Another is the change in gravitational potential energy. However, for our applications, this second form of energy will be neglected. All other energy contributions are lumped together in the internal energy of the system, noted e_{int}, which collects the thermal energy of all constituents[8]. The field density is then obtained by $\varphi_i = \rho\,E = \rho\,(e_{kin} + e_{int})$. Because total energy is conserved, there is no internal production, i.e. $\mathfrak{P}_{in} = 0$. The rate of change of total energy is the consequence of the net heat flux into the system and the mechanical power done on the system. Similar to the resulting forces for the momentum balance, both the heat supply and mechanical power combine surface and volume contributions. The heat transferred across the system boundaries is determined by the contact heat supply $\mathbf{t}_e(\mathbf{X},t,\mathbf{n})$ which, analogously to Cauchy's stress theorem, can be expressed as a linear combination between a heat flux vector $\mathbf{q}\in\mathbb{E}^d$ and the

[8] Depending on the level of modeling required, the internal energy can be further decomposed into microscopic contributions due to the motions and configurations of individual molecules, atoms and subatomic particles making up the matter of the system (Moran et al. (2010, Sec. 2.3)).

boundaries normal (I.-S. Liu et al. (2002, Sec. 2.4)), viz.

$$\mathbf{t}_e(\mathbf{X}, t, \mathbf{n}) = -\mathbf{q}(\mathbf{X}, t) \cdot \mathbf{n}(\mathbf{X}, t) \, .$$

The heat transfer inside the volume of the system (e.g. due to radiation) is here not taken into account. With regard to the mechanical power, the rate of work done by the contact forces on the fluid system moving at velocity $\mathbf{v}$ is $\underline{\boldsymbol{\sigma}} \cdot \mathbf{v}$. The rate of work done by the body forces is similarly obtained by $\rho \, \mathbf{b} \cdot \mathbf{v}$. Given Equation 2.12, we thus have $\mathfrak{F} = \underline{\boldsymbol{\sigma}} \cdot \mathbf{v} - \mathbf{q}$, $\mathfrak{P} = \mathfrak{P}_{out} = \rho \, \mathbf{b} \cdot \mathbf{v}$ and the physical law translates into the global balance equation

$$\begin{aligned} \frac{D}{Dt} \int_{\mathcal{D}_{mat}(t)} (\rho \, E)(\mathbf{X}, t) \, \mathrm{d}\mathbf{X} = & \int_{\partial \mathcal{D}_{mat}(t)} (\underline{\boldsymbol{\sigma}} \cdot \mathbf{v} - \mathbf{q})(\mathbf{X}, t) \cdot \mathbf{n}(\mathbf{X}, t) \, \mathrm{d}S_{\mathbf{X}} \\ & + \int_{\mathcal{D}_{mat}(t)} (\rho \, \mathbf{b} \cdot \mathbf{v})(\mathbf{X}, t) \, \mathrm{d}\mathbf{X} \, , \end{aligned}$$

which is equivalent to the local equation

$$\frac{\partial (\rho \, E)}{\partial t}(\mathbf{x}, t) + \nabla_{\mathbf{x}} \cdot \left[\rho \, \mathbf{v} \, E \right](\mathbf{x}, t) = \nabla_{\mathbf{x}} \cdot (\underline{\boldsymbol{\sigma}} \cdot \mathbf{v} - \mathbf{q})(\mathbf{x}, t) + (\rho \, \mathbf{b} \cdot \mathbf{v})(\mathbf{x}, t) \, . \quad [2.18]$$

Remarks on other forms of energy equation The balance equations for the other forms of the energy can also be determined. The one for the kinetic energy is readily obtained by dotting the linear momentum equation with the velocity vector. The equation for the internal energy can then be derived by subtracting from the total energy equation the one for the kinetic energy. Let us note that because there is an internal production process present in the kinetic and internal energies balances (i.e. $\mathfrak{P}_{in} \neq 0$), they are not conservative quantities, contrary to the total mass, the linear momentum, the angular momentum and the total energy.

Entropy inequality

Before diving into the constitutive models in Section 2.2.2, it is important to present here the last physical principle about entropy inequality. Even if this last relation is not explicitly solved, it implicitly enters into the governing equations because it is required that the constitutive models comply with the entropy inequality principle. The most tangible physical interpretation of entropy is perhaps to be found from a statistical thermodynamics point of view. The Boltzmann's entropy formula relates the number of possible microstates of a system for a particular macrostate[9] to the entropy of that system. As the number of microstates corresponding to a macrostate increases, the probability of finding the system in that macrostate also increases and so does the entropy. Hence, systems tend to evolve toward states with greater "disorder". This

[9] A microstate refers to a particular arrangement of the constitutive microscopic particles of a system and their corresponding energy levels. A given macrostate, which defines the macroscopic properties like temperature or pressure, can be achieved by different microstates.

interpretation is consistent with the second law of thermodynamics imposing during any process in a closed system that the internal entropy production is strictly positive when irreversibilities are involved and zero only in the limit as irreversibilities are reduced to zero. This can be formulated as

$$\begin{aligned}\frac{D}{Dt}\int_{\mathcal{D}_{mat}(t)}(\rho\,\eta)(\mathbf{X},t)\,\mathrm{d}\mathbf{X} - \int_{\partial\mathcal{D}_{mat}(t)}\mathfrak{F}_\eta(\mathbf{X},t)\cdot\mathbf{n}(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} \\ - \int_{\mathcal{D}_{mat}(t)}\mathfrak{P}_{out,\eta}(\mathbf{X},t)\,\mathrm{d}\mathbf{X} \\ = \int_{\mathcal{D}_{mat}(t)}\mathfrak{P}_{in,\eta}(\mathbf{X},t)\,\mathrm{d}\mathbf{X} \geq 0\,,\end{aligned}$$

where η is the specific entropy density in $\mathrm{J}/(\mathrm{kg\,K})$, $\mathfrak{F}_\eta$ is the entropy flux across the system boundaries, $\mathfrak{P}_{out,\eta}$ is the external entropy supply and $\mathfrak{P}_{in,\eta}$ is the internal entropy generation which must be greater or equal to zero. The exploitation of this principle to develop constraints on the constitutive models can be found in I.-S. Liu (2019, Sec. 9.3). In a *reversible* process ($\mathfrak{P}_{in,\eta} = 0$), it is often assumed that the entropy flux is proportional to the heat flux which is divided by an absolute temperature to make the flux path independent, i.e. $\mathfrak{F}_\eta = \mathbf{q}/T$. Neglecting any external entropy supply, we can obtain the common differential local form (in a Lagrangian fashion, without the convective flux, as we will not use this equation as a balance law)

$$\frac{\partial S}{\partial t} + \nabla\cdot\left(\frac{\mathbf{q}}{T}\right) = 0\,, \qquad [2.19]$$

where S is the entropy per unit volume in $\mathrm{J}/(\mathrm{K\,m}^3)$. Under this formulation, the entropy S can also be viewed as the quantity of heat exchanged by the system during a reversible process taking place at the equilibrium temperature T.

2.2.2 Closure of the equations by constitutive models

One of the principal objectives of continuum mechanics is to predict the behavior of a system once the external causes of its change of state are specified. This amounts to solve the system of equations formed by the conservation of mass, balance of linear momentum and balance of total energy. So far, we have identified the causes behind the changes of the conservative quantities but we have not yet specified them to the type of material we want to model. This is covered in this section.

Transport laws and properties

In addition to the entropy inequality presented at the end of Section 2.2.1, constitutive mathematical models should obey to the principle of frame-indifference and material symmetry (I.-S. Liu (2019, Secs. 5&6)). The first requirement ensures that constitutive functions are indifferent to a change of reference frame, i.e. independent of the way the system is observed. This leads to the condition

of material objectivity. The second is concerned with the invariance of the material's response under certain transformations. Fluids can be considered as viscoelastic materials. In the most general case, for a viscous heat-conducting fluids, the constitutive relations for the Cauchy stress tensor and heat flux vector are function of the deformation gradient $\underline{\boldsymbol{\epsilon}}$, the rate of deformation $\dot{\underline{\boldsymbol{\epsilon}}}$, the temperature T and the gradient of temperature ∇T (I.-S. Liu (2019, Sec. 8.5)), i.e.

$$\underline{\boldsymbol{\sigma}} = \underline{\boldsymbol{\sigma}}(\underline{\boldsymbol{\epsilon}}, \dot{\underline{\boldsymbol{\epsilon}}}, T, \nabla T)\,, \quad \mathbf{q} = \mathbf{q}(\underline{\boldsymbol{\epsilon}}, \dot{\underline{\boldsymbol{\epsilon}}}, T, \nabla T)\,.$$

The material symmetry condition requires that the dependence on $(\underline{\boldsymbol{\epsilon}}, \dot{\underline{\boldsymbol{\epsilon}}})$ be reduced to the determinant of $\underline{\boldsymbol{\epsilon}}$ and the velocity gradient $\nabla \mathbf{v}$ (I.-S. Liu (2019, Sec. 8.3)). From conservation of mass, the dependence on $\det(\underline{\boldsymbol{\epsilon}})$ can be replaced by the density ρ. The velocity gradient tensor can be decomposed into a symmetric part, $\underline{\boldsymbol{S}}_{sym}$, and a skew-symmetric part, $\underline{\boldsymbol{S}}_{ske}$, such that

$$\nabla \mathbf{v} = \underline{\boldsymbol{S}}_{sym} + \underline{\boldsymbol{S}}_{ske}\,.$$

By material objectivity and isotropicity of the fluid, $\dot{\underline{\boldsymbol{\epsilon}}}$ cannot depend on $\underline{\boldsymbol{S}}_{ske}$ (I.-S. Liu (2019, Sec. 8.1)), so the rate of deformation only depends on the rate of strain, $\dot{\underline{\boldsymbol{\epsilon}}} = \dot{\underline{\boldsymbol{\epsilon}}}(\underline{\boldsymbol{S}}_{sym})$. Writing the most general expressions that $\underline{\boldsymbol{\sigma}}$ and $\mathbf{q}$ can take and keeping only up to linear terms both in the strain rate $\underline{\boldsymbol{S}}_{sym}$ and the temperature gradient ∇T, the most widely-used Newtonian-Fourier model is obtained

$$\underline{\boldsymbol{\sigma}} = -p\,\underline{\boldsymbol{I}} + \mu_2 \mathrm{tr}(\underline{\boldsymbol{S}}_{sym})\,\underline{\boldsymbol{I}} + 2\mu\,\underline{\boldsymbol{S}}_{sym}\,, \tag{2.20}$$

$$\mathbf{q} = -\lambda \nabla T\,, \tag{2.21}$$

where $\underline{\boldsymbol{I}}$ is the identity tensor, μ is the *dynamic viscosity* coefficient and μ_2 is the second viscosity coefficient, both in $\mathrm{kg}/(\mathrm{m\,s})$, and λ is the *thermal conductivity* in $\mathrm{W}/(\mathrm{m\,K})$. The symmetric part of the velocity gradient tensor can be further decomposed into an isotropic and deviatoric part as

$$\begin{aligned}
\underline{\boldsymbol{S}}_{sym} &= \underline{\boldsymbol{S}}_{sym,iso} + \underline{\boldsymbol{S}}_{sym,dev} \text{ with} \\
\underline{\boldsymbol{S}}_{sym,iso} &= \frac{1}{3}\mathrm{tr}(\underline{\boldsymbol{S}}_{sym})\,\underline{\boldsymbol{I}} = \frac{1}{3}(\nabla \cdot \mathbf{v})\,\underline{\boldsymbol{I}}\,, \\
\underline{\boldsymbol{S}}_{sym,dev} &= \underline{\boldsymbol{S}}_{sym} - \underline{\boldsymbol{S}}_{sym,iso} \triangleq \underline{\boldsymbol{D}}\,.
\end{aligned}$$

The stress tensor in Equation 2.20 can now be rewritten as

$$\underline{\boldsymbol{\sigma}} = (-p + \kappa \nabla \cdot \mathbf{v})\underline{\boldsymbol{I}} + 2\mu \underline{\boldsymbol{D}}\,, \tag{2.22}$$

where $\kappa = (\mu_2 + 2/3\mu)$ is introduced as the *bulk viscosity* coefficient. Equation 2.22 shows that, in the most general case, the isotropic part of the complete stress tensor contains a viscous stress that is additive to the thermodynamic pressure. It is customary to make the Stokes hypothesis according to which the bulk viscosity is zero, i.e. that the two coefficients of viscosity are related

by $\mu_2 + 2/3\mu = 0$. However, it has been demonstrated that for most fluids of practical interest, $\kappa \neq 0$, and that the value of κ may even be much larger than the one of μ. Instead of the Stokes hypothesis which is not satisfactory from a physical point of view, Buresti (2015) suggests that we should assume that

$$|\kappa \nabla \cdot \mathbf{v}| \ll p\,,$$

which amounts to neglect any difference between the thermodynamic and mechanical pressures. This is the assumption done in this work that leads to a stress tensor of the form

$$\underline{\boldsymbol{\sigma}} = -p\underline{\boldsymbol{I}} + 2\mu\underline{\boldsymbol{D}}\,, \qquad [2.23]$$

with $\underline{\boldsymbol{D}} = \frac{1}{2}\left[\nabla\mathbf{v} + (\nabla\mathbf{v})^T\right] - \frac{1}{3}(\nabla \cdot \mathbf{v})\,\underline{\boldsymbol{I}}$. The three contributions to the stress in a fluid flow are commented in Papanastasiou et al. (2000, Sec. 1.3.3). The mechanical pressure p arises due to the perpendicular motion of the particles toward a material surface, the viscous normal stresses, which are the diagonal components of $2\mu\underline{\boldsymbol{D}}$, are caused by accelerating or decelerating perpendicular motions toward material surfaces, and the viscous shear stresses, which are the non-diagonal components, are induced by velocity gradients in directions perpendicular to the streamlines.

Thermodynamic properties and equation of state

The specification of the stress tensor in Equation 2.20 and the heat flux vector in Equation 2.21 introduces two additional intensive properties: the pressure p and temperature T. Inserting those constitutive relations in the Navier-Stokes equations, Equation 2.16, Equation 2.17 and Equation 2.18, we end up with $1 + d + 1 = 2 + d$ equations to be solved (d being the spatial dimension) for $4 + d$ unknown quantities, namely the conservative quantities, i.e. mass, the d components of linear momentum and total energy, and the pressure and temperature. Two additional constitutive relations should therefore supplement the set of governing equations. In practice, those two relations will be two so-called incomplete *equations of state* (EOS) that embody the thermodynamic properties of a given material. Hence, the dynamics of the fluid is not solely determined by the form of the governing equations but also by the choice of those two EOS. The derivation of those EOS necessitate to introduce a few thermodynamic concepts beforehand.

Local thermodynamic equilibrium A system is said to be in thermodynamic equilibrium if, when isolated from its surroundings, there is no observable changes in its macroscopic properties[10]. In practice, the whole system to be studied will be out of equilibrium since we are interested in studying its dynamics which involves transport of quantities associated to spatial gradients

[10] Let us note that the concept of equilibrium is to some extent subjective as the state variables do not keep rigorous fixed values because of the presence of unavoidable microscopic fluctuations. But the equilibrium state is stable because irreversible processes tend to drive the system toward a unique state for which entropy is maximized.

(e.g. heat transport due to temperature gradients). That is where the local thermodynamic equilibrium hypothesis comes into play. It states that at a given instant of time, equilibrium is achieved locally in each individual material particles of the system. The state of equilibrium is allowed to be different between neighboring particles and to change in the course of time for each individual particle (Lebon et al. (2008, Sec. 2.2)).

Gibbs' relation and fundamental potentials If we apply the first law of thermodynamics to a fixed closed system undergoing a reversible process where the only work performed is due to pressure forces, we obtain the local differential equation

$$\mathrm{d}e^* = \delta Q - p\mathrm{d}V\,,$$

where e^* is the internal energy in J, δQ is the heat exchanged, which depends on the path taken during the process (hence the inexact differential "δ"), and $\mathrm{d}V$ is the change in volume. Then, using the second law of thermodynamics, Equation 2.19, we can replace the heat supply, $\delta Q = T\mathrm{d}S^*$, which leads to the well-known Gibbs' equation[11] (Lebon et al. (2008, Sec. 1.4))

$$\mathrm{d}e^* = T\mathrm{d}S^* - p\mathrm{d}V\,, \qquad [2.24]$$

where S^* is the entropy in J/K. Let us remark that Equation 2.24 assumes infinitesimal changes of a system under equilibrium conditions. It can be employed when modeling a fluid flow by invoking the local thermodynamic equilibrium hypothesis stated before. It follows directly from Gibbs' relation that

$$e^* = e^*(S^*, V)\,, \qquad [2.25]$$

which is one of the four fundamental thermodynamic functions. The other three, also called the thermodynamic potentials, can be obtained by applying a Legendre transformation to the differential of e^*. Those are the enthalpy, the Gibbs' free energy and the Helmholtz's free energy (Lebon et al. (2008, Sec. 1.5.1)).

Definition of incomplete EOS Equation 2.25, or more precisely the set of variables (e^*, S^*, V), defines a complete EOS because it provides a complete description of the thermodynamic state and all the other state functions can be computed from it. For example, the pressure and temperature are obtained as first derivatives,

$$p(V, S^*) = -\left.\frac{\partial e^*}{\partial V}\right|_{S^*}\,, \quad T(V, S^*) = \left.\frac{\partial e^*}{\partial S}\right|_{V}\,.$$

This illustrates also that at thermodynamic equilibrium, we can select two independent state functions to express any other thermodynamic quantities.

[11] The generalized Gibbs equation for a system with several species should also account for the reversible chemical work through the chemical potentials of every species.

The two extensive properties S^* and V are not convenient to work with, thus we would like to select two other quantities which should be intensive instead. This choice is arbitrary and will depend on the type of EOS we want to build. Let's say we take as independent state functions the temperature T and density ρ. The pressure can thus be obtained by a relation like $p = p(T, \rho)$. This relation is called an incomplete EOS because the set of variables (p, T, ρ) does not allow determining other quantities like e.g. the internal energy by taking derivatives of the relation $p(T, \rho)$, as explained in Cowperthwaite (1969). It was pointed out at the beginning of this section that there are two more unknowns than the number of equations to solve. Therefore, we will need to define two incomplete EOS, typically a volumetric and caloric EOS,

$$p = p(\rho, T)\,, \quad e = e(\rho, T)\,. \tag{2.26}$$

An overview of the possibilities to define such incomplete EOS is given in Section 2.3.3.

Thermodynamic properties We have seen that primary quantities like pressure, internal energy, density or temperature can be obtained as first-order partial derivatives of the four fundamental thermodynamic potentials, i.e. internal energy, enthalpy, Helmholtz and Gibbs free energies. In addition to the primary state variables, it may be necessary to compute other secondary thermodynamic quantities. In particular, five fundamental quantities are often required as inputs to a model as they are the ones that can be measured experimentally:

$$C_v \triangleq \left.\frac{\partial e}{\partial T}\right|_{\rho}, \qquad \text{isochoric heat capacity}\,, \tag{2.27}$$

$$C_p \triangleq \left.\frac{\partial h}{\partial T}\right|_{p}, \qquad \text{isobaric heat capacity}\,, \tag{2.28}$$

$$\alpha_p \triangleq \frac{-1}{\rho}\left.\frac{\partial \rho}{\partial T}\right|_{p}, \qquad \text{isobaric thermal expansion}\,, \tag{2.29}$$

$$\beta_T \triangleq \frac{1}{\rho}\left.\frac{\partial \rho}{\partial p}\right|_{T}, \qquad \text{isothermal compressibility}\,, \tag{2.30}$$

$$K_S \triangleq \frac{1}{\rho}\left.\frac{\partial \rho}{\partial p}\right|_{S}, \qquad \text{isentropic compressibility}\,. \tag{2.31}$$

Let us remark that those quantities can be expressed as second-order partial derivatives of the potentials and that all other second derivatives can be expressed in terms of them, making non-measurable properties accessible. In Menikoff and Plohr (1989, App. A), three dimensionless parameters linking those five measurable properties are presented. Rosenholm (2021) gives a complete and condensed overview of all the interconnections between the various properties through a "thermodynamic wheel of connections" and recalls the useful mathematical relations to build such map, including the well-known

Maxwell relations. Finally, the last property of interest in the context of this work is the speed of sound whose squared value is defined as the isentropic partial derivative of the pressure with respect to the density,

$$c^2 \triangleq \left.\frac{\partial p}{\partial \rho}\right|_S = \left.\frac{\partial p}{\partial \rho}\right|_e + \frac{p}{\rho^2}\left.\frac{\partial p}{\partial e}\right|_\rho . \qquad [2.32]$$

The second equality in Equation 2.32 has been obtained by using the chain rule (Menikoff and Plohr (1989, App. A)),

$$\left.\frac{\partial}{\partial y}\right|_z = \left.\frac{\partial x}{\partial y}\right|_z \left.\frac{\partial}{\partial x}\right|_y + \left.\frac{\partial}{\partial y}\right|_x ,$$

and the Gibbs equation, Equation 2.24.

2.3 Specific models for the liquid phase

The use of the compressible Navier-Stokes equations to describe internal or external gas flows, laminar or turbulent, at low-subsonic or hypersonic regimes, has become a standard. However, this is much less the case for liquid flows which are much more often predicted by means of incompressible models. Therefore, the selection of a compressible model in this treatise brings some elements to be discussed. Moreover, contrary to the gas phase for which a sound theory exists to estimate the thermophysical properties, for the liquid phase, there is no overarching model available. Those aspects are covered in the following of this section.

2.3.1 Stiff compressible model

Let us start by defining the terminology that will be used in what follows. The mass balance, Equation 2.16, can alternatively be written in terms of the material derivative,

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{v}\cdot\nabla ,$$

such that

$$\frac{1}{\rho}\frac{D\rho}{Dt} + \nabla\cdot\mathbf{v} = 0 . \qquad [2.33]$$

From Equation 2.33, it can be deduced that the well-known divergence-free constraint on the velocity field can only be assumed if the rate at which the density of a material particle changes can be neglected. This will be the case only if the density variations due to both pressure and temperature variations can be assumed to be negligible. The rigorous derivation of the conditions to be satisfied to make such assumption are presented in Batchelor (1967, Sec. 3.6) and are summarized here:

1. When density variations due to pressure are insignificant, the flow is said to be incompressible. It is important to remark that an incompressible flow does not imply a constant density field. The three conditions to be met are:

a) The pressure changes propagate much faster than the fluid velocity, so that compressibility effects are not apparent. This translates into the well-known condition on the Mach number:

$$\mathrm{Ma} = \frac{v_n}{c} \ll 1\,,$$

with v_n, the normal velocity.

b) The frequency of the velocity fluctuations in the flow, noted f, over a region of characteristic dimension L, is much smaller than the frequency of the acoustic waves, such that the flow field is not perturbed by the propagation of acoustic waves:

$$f \ll \frac{c}{L}\,.$$

This condition is in general satisfied, except in the situations studied in acoustics.

c) The difference of static pressure between two points is much smaller than the absolute pressure:

$$g\,L \ll c^2\,,$$

with the gravitational acceleration g. The only circumstances in which this situation is not met occur in atmospheric flows.

2. The second condition concerns the variations of density of a material element due to internal dissipative heating or due to molecular heat diffusion which should be negligible. This translates into two conditions:

a) The viscous dissipation is small enough not to generate enough heat affecting density:

$$\frac{\alpha_p\, v_n^2}{C_p}\frac{\mu}{\rho\, L\, v_n} \ll 1\,.$$

This contribution can in practice be neglected, except in the events of highly viscous fluids or very strong shear flows.

b) The heat conduction is small enough not to induce a temperature gradient, ∇T, impacting the density:

$$\alpha_p \nabla T \frac{\lambda}{\rho\, C_p\, L\, v_n} \ll 1\,.$$

Physical arguments

From this analysis, we can provide some justifications for our choice of the compressible models for both the gas and liquid phases. For the gas phase, outside the boundary layer region in atmospheric entry conditions, it is evident that condition 1.a will not be respected and that compressibility effects must be taken into account. In the boundary layer region with locally small velocities, the density variations due to pressure will become less pronounced. However, the strong velocity gradients may induce sufficient viscous dissipation such that condition 2.a is no longer valid. Even though those considerations are interesting, the high-temperature effects present in reentry flows will in any case make the density strongly dependent on pressure and temperature. Concerning the liquid phase, the density dependence on pressure is expected to be minimal. However, the liquid layer might be subjected to important temperature gradients such that condition 2.b is violated, making the divergence-free velocity condition no longer valid.

Various mathematical models

Depending on the flow conditions to be simulated and the conditions presented above, several versions of the governing equations are possible.

- *Constant density flow*: in this situation, the density is constant everywhere, $\rho =$ constant, and the solenoidal constraint on the velocity is applicable. If temperature evolution is not of interest, the continuity and momentum equations can be solved solely.
- *Incompressible thermally dilatable flow with small temperature variations*
  - *Boussinesq approximation*: the velocity is still considered as divergence free because the temperature dependence of the density is assumed to only have a notable influence on the buoyancy forces. According to De Boer (1984), this is a fair assumption for weakly heated flows.
  - *Anaelastic equations*: this model, mentioned in Martinez and Gartling (2004), extends the applicability of the Boussinesq approximation by allowing unrestricted length scales. To do so, the density is expressed as a sole function of the temperature, $\rho = \rho(T)$, and its local time rate of change is assumed negligible compared to the spatial divergence, i.e. $\nabla \cdot (\rho \mathbf{v}) = 0$. This model is well suited for strongly heated deep flows, as pointed out in De Boer (1984).
- *Incompressible thermally dilatable flow with large temperature variations*
  - *Acoustically filtered equations*: in this model, also presented in Martinez and Gartling (2004), the pressure is decomposed into a spatially independent thermodynamic part, p_0, and an asymptotically smaller dynamical part. The density is decoupled from pressure and undergoes isobaric thermal expansion via the EOS $\rho = \rho(p_0, T)$. In this case, the full continuity equation is retained and the pressure gradient in the momentum equation is expressed based on the dynamic

pressure (but is too small to induce any density variations). The model properties make it appropriate for strongly heated shallow flows.
  - *Fully compressible equations*: although tiny for some materials like liquid metals, any material possesses a non-zero isothermal compressibility coefficient. In theory, this means that using an EOS for the density of the form $\rho = \rho(p, T)$ can always be done. This choice in turn implies the solution of the full compressible Navier-Stokes equations, a more complex system than the simplified models listed above, but that is theoretically valid in any of the situations mentioned before.

- *Compressible thermally dilatable flow*: if the pressure variations induce significant density changes, the natural choice is to consider the full compressible Navier-Stokes equations.

The simulation of the dynamics of the thin liquid layer resulting from the melting of a solid body immersed in a high-temperature gas jet is expected to fall in the category of incompressible thermally dilatable flow with large temperature variations. Although the use of the acoustically filtered equations to describe the liquid layer would have been a judicious choice, we decided to rely on the full compressible equations for some numerical reasons exposed below. In the following, we refer to the liquid incompressible thermally dilatable flow as being a *stiff compressible* flow. The term variable density nearly incompressible flow would be just as legitimate. We avoid the name "weakly compressible" as the techniques commonly associated to this name keep the compressible equations (as we do) but select an arbitrary value for the speed of sound so that the numerical stiffness of the problem is relaxed (while we consider the physical speed of sound value), as explained in Vittoz et al. (2019).

Numerical arguments

The arguments presented above suggest the use of either (i) the full compressible Navier-Stokes equations for both phases or (ii) the full compressible models for the gas phase and a filtered version of the compressible equations for the liquid phase retaining the density variations while decoupling them from the thermodynamic pressure. The first option is selected here because of two main reasons related to numerical aspects:

- The first reason is the availability of numerical schemes specifically designed to treat low-Mach number flows governed by the compressible Navier-Stokes equations and their demonstrated capabilities to handle low-speed stiff liquid phases. This is discussed in Section 5.3.1. Moreover, the fact that the numerical platform used in this work already contained an implicit time discretization made it possible to consider simulations in which an extreme time step restriction would be avoided.

- In this treatise, we adopt a sharp interface coupling between the gas and the liquid phases (see Chapter 8). The second reason is the complexity of the algorithms to be able to couple two sets of equations of different nature across a sharp interface. Various studies have succeeded in coupling compressible and incompressible equations: Caiden et al. (2001), Daru et al. (2010), Billaud et al. (2011), Boger et al. (2015), Ghasemi and Nordström (2019). Because the equations are of different nature (elliptic VS hyperbolic), distinct procedures to obtain the solution (e.g. pressure-based VS density-based) have to be implemented. It is also needed to adapt each of the original algorithm used to discretize the bulk equations in such a way that they communicate between them the flow quantities across the interface at the right moment. One of the most complicated aspects of this coupling is linked to the dissimilar meanings of pressure between the compressible and incompressible models. This makes the waves transmission at the interface particularly complex to handle.

2.3.2 Thermodynamic and transport properties

To close the system of governing equations, expressions for the thermodynamic and transport properties as well as the incomplete EOS, Equation 2.26, need to be obtained. While there are well established theories for ideal gas to accurately estimate properties of pure gas or mixture of gases, this is not the case for liquids, or condensed phase in general. In this section, an overview of the available models for the liquid phase, put in parallel to the ones for ideal gases, is given. The ideal gas model is an idealized concept assuming the absence of any intermolecular forces and a negligible volume for the molecules. Gases at very high pressure or very low temperature tend to experience intermolecular interactions and depart from the ideal gas model. Real gas models are used in those situations. Similarly, the repulsive and attractive intermolecular forces in liquids and solids are very important, which makes the ideal gas model not adapted to describe their behavior. It must be remembered that in this work, we are only considering constant thermo-transport properties, both in the gas and the condensed phases. We are also not taking into account the presence of distinct chemical species in a given phase, even if this phase is a mixture of several components. Thus, the models presented in the following to express mixture properties as a function of primary flow variables have not been implemented as part of this work. For the sake of completeness, we still present an overview here.

Appropriate EOS

Ideal gas The ideal or perfect gas law relates the pressure, density and temperature at local thermodynamic equilibrium according to

$$p = \rho R T, \quad R = \frac{\mathcal{R}}{\mathcal{M}},$$

where R is the specific gas constant, $\mathcal{R}$ the universal gas constant and $\mathcal{M}$ the molar mass. For a mixture of perfect gases composed by N_s chemical species, each ith-species obeys the perfect gas law such that[12]

$$p_i = \rho_i R_i T, \tag{2.34}$$

which results in

$$p = \sum_{i=1}^{N_s} p_i, \quad \rho = \sum_{i=1}^{N_s} \rho_i. \tag{2.35}$$

Real gas and condensed phase For a pure real gas or condensed phase (i.e. with a single chemical component), different possibilities to define an EOS exist:

- *Analytical*: the most simple way is to specify the EOS as an analytical expression. A large variety of models exist, each of them being characterized by its complexity and its range of applicability. The simplest EOS that takes into account the three molecular effects present in matter, i.e. agitation, repulsion and attraction, is the Noble-Abel Stiffened Gas (NASG) EOS presented in Olivier Le Métayer and Saurel (2016):

$$p(\rho, e) = (\gamma - 1)\frac{(e - q)}{(1/\rho - b)} - \gamma p_\infty, \tag{2.36}$$

  where the product $(\gamma-1)(e-q)$ represents thermal agitation, the covolume parameter b accounts for repulsive short distance effects by defining an excluded volume around the molecule (which is equivalent to consider finite size molecules) and the term γp_∞ models attractive interactions. The derivation of a class of analytical EOS extending this NASG model is presented after in Section 2.3.3.
- *Multi-parameter EOS based on the free Helmholtz energy*: this more advanced approach, reviewed e.g. in Span et al. (2001), consists in formulating the EOS based on the nondimensional free Helmholtz energy, F,

$$\frac{F(T, \rho)}{\mathcal{R}T} = \alpha^0(\tau, \delta) + \alpha^r(\tau, \delta),$$

[12] We assume here a single temperature model where all the energy modes are defined with a common temperature. Other models using several temperatures to partition the energy modes exist.

with the inverse reduced temperature, $\tau = T_r/T$, and density, $\delta = \rho/\rho_r$, where the reduced quantities usually correspond to the critical parameters of the described substance. The parameters describing the functions α^0 and α^r are fitted to empirical data. The elegance of this formulation is that all other thermodynamic properties can be obtained through analytic derivatives of F since the Helmholtz's free energy is a fundamental thermodynamic function (see Section 2.2.2). Several libraries relying on this approach, like CoolProp (Bell et al. (2014)) and REFPROP (Huber et al. (2022)), are available. The main disadvantage is the massive computation time. That is why EOS look-up tabulation methods have been developed to compute and store the EOS prior to the simulation, like in De Lorenzo et al. (2017) and F. Föll et al. (2019).

- *Molecular simulations*: since experimental data might be scarce for some materials and/or limited to a narrow range of conditions, an alternative approach is to generate reference data based on simulations at the microscopic level. In this regard, the use of *Molecular Dynamics Simulation* (MDS) has been investigated as part of this work. The theoretical foundations, the results and conclusions of this study can be found in Appendix B.

If the presence of multiple species is accounted for in a mixture of real gases or condensed phases, it is no longer valid to consider that each species in the mixture obeys it owns EOS, independent of the others, as Equation 2.34 does. Moreover, Dalton's law for partial pressures, Equation 2.35, does not hold anymore. Poling et al. (2001, Sec. 5.2) note that there are two principal ways to obtain an analytical EOS for the mixture. The first one is to add "higher-order" terms to the EOS to consider the interactions or correlations of properties among pairs, trios, etc. of the components. The virial EOS is an example. The other approach, which is more convenient, uses the individual EOS of each component and the composition dependence is included by making the parameters of the EOS vary with composition through some *mixture rules*. Two illustrative examples of basic mixture rules for the mixture density are given. The first one is

$$\rho = \sum_{i=1}^{N_s} Y_i \rho_i \,,$$

where Y_i is the mass fraction of the ith-species in the mixture. This rule assumes that the volume of the mixture is proportional to the mass and that the proportionality constant is the same for both pure species i and the mixture. The second example, known as Amagat's law, assumes instead that the partial volumes of the different species in the mixture are additive such that

$$\frac{1}{\rho} = \sum_{i=1}^{N_s} \frac{Y_i}{\rho_i} \,.$$

Thermodynamic properties

Ideal gas The most accurate way to estimate properties of ideal gas mixtures over a wide range of conditions is to rely on statistical thermodynamics using the so-called partition function describing the population of internal energy modes. The partition function can be split into separate energy types assuming there is no coupling between the different energy modes. For each energy mode, an expression for the partition function is provided by the Rigid-Rotor Harmonic-Oscillator (RRHO) model exploiting quantum mechanical results. Internal energies and enthalpies of each mode can then be computed as derivatives of the partition functions. This is done in the MUlticomponent Thermodynamics And Transport properties for IONized gases library in C++ (Mutation++), described in Scoggins and T. E. Magin (2014).

Real gas and condensed phase Contrary to perfect gas, it is not possible to obtain thermodynamic properties from the partition functions as there are no available expressions. Instead, experimental measurements of quantities such as the heat capacities over a range of temperatures (and sometimes pressures) are collected for pure components and dedicated mixture rules are then applied when mixture quantities need to be obtained. For example, for a liquid alloy, a reasonable estimate of the specific heat capacity can be calculated from the Kopp-Neumann rule of mixtures (Valencia and Quested (2001)):

$$C_p = \sum_{i=1}^{N_s} Y_i \, C_{p,i} \, .$$

Primary thermodynamic quantities like the enthalpy or internal energy can then be obtained by integrating measurable quantities, like the heat capacities.

Transport properties

Ideal gas For ideal gas mixtures, the dynamic viscosity and thermal conductivity can be obtained from kinetic theory by solving linear transport systems as an approximate result of the Boltzmann equation by the Chapman-Enskog expansion (see T. Magin (2004)). This is also the approach adopted in the library Mutation++.

Real gas and condensed phase As described in Poling et al. (2001, Secs. 9 and 10), there is no comparable unified theory to estimate the transport properties of liquids. Experimental measurements remain essential and mixture properties can also be estimated based on mixture rules from the elemental properties of the components. For liquid, the most simple mixture rule for viscosity is known as the Grunberg-Nissan rule (Poling et al. (2001, Sec. 9.13)),

$$\log(\mu) = \sum_{i=1}^{N_s} X_i \log(\mu_i) \, ,$$

with the molar fraction X_i, while for thermal conductivity, the power law rule can be considered (Poling et al. (2001, Sec. 10.12)),

$$\lambda = \left(\sum_{i=1}^{N_s} Y_i \lambda_i^{-2} \right)^{-1/2} .$$

2.3.3 Derivation of a thermodynamic consistent EOS

In this section, we present the derivation of analytical EOS expressions that have been implemented in our solver. This is done following a general procedure ensuring thermodynamic consistency of the derived EOS. The starting point is to consider the most general form of the Mie-Grüneisen model which describes many "real" materials like liquids and solids and from which a variety of EOS can be obtained. Heuzé (2012) indicates that the underlying assumption behind the Mie-Grüneisen model, originally developed for solids by Mie (1903) and Grüneisen (1912), is that the pressure is a linear function of the internal energy such that

$$p(v,e) = p_r(v) + \frac{\Gamma(v)}{v}(e - e_r(v)) , \qquad [2.37]$$

where $v = 1/\rho$ is the specific volume, $\Gamma(v)$ is the Grüneisen coefficient depending solely on v and p_r and e_r are reference pressure and internal energy. Those reference states can be taken along different reference curves, like the isentropic or Hugoniot curves (Shyue (2001)), leading to different expressions for $p_r(v)$ and $e_r(v)$. For instance, Wang et al. (2021) uses MDS to compute the Hugoniot curve of iron and to estimate the reference states. Inspired by the Noble-Abel Stiffened Gas (NASG) EOS of Olivier Le Métayer and Saurel (2016), we add a co-volume parameter to Equation 2.37 such that the general EOS expression we manipulate is

$$p(v,e) = p_r(v) + \frac{\Gamma(v)}{v-b}(e - e_r(v)) . \qquad [2.38]$$

To keep the formulation as generic as possible, we express the density dependent parameters in the following way:

$$\Gamma(v) = \Gamma_0 \left(\frac{v}{v_0} \right)^{\alpha_\Gamma} , \quad \alpha_\Gamma \in [0,1] \text{ (according to Shyue (2001))} , \qquad [2.39]$$

$$p_r(v) = d_p \left[\left(\frac{v_0}{v} \right)^{\alpha_p} - 1 \right] + p_{r,0} , \qquad [2.40]$$

$$e_r(v) = d_e \left[\left(\frac{v_0}{v} \right)^{\alpha_e} - 1 \right] + e_{r,0} , \qquad [2.41]$$

with Γ_0, v_0, $p_{r,0}$ and $e_{r,0}$ being constant reference states and α_Γ, α_p, d_p, α_e, d_e constant coefficients. Equation 2.38 is the prototype for an incomplete EOS. We need two incomplete EOS (see Equation 2.26) to close the system of governing equations. This second EOS should be related in a way to the first EOS. This

link will be established based on some thermodynamic constraints such that the resulting set of incomplete EOS is thermodynamic consistent. The procedure followed after is inspired by the work of Le Métayer et al. (2004), Heuzé (2012), Latige (2013, Sec. 4.4.3) and Radulescu (2020). Since we want to find $e = e(v, T)$, we write the differential

$$\mathrm{d}e = \frac{\partial e}{\partial v}\bigg|_T \mathrm{d}v + \underbrace{\frac{\partial e}{\partial T}\bigg|_v}_{C_v} \mathrm{d}T\,. \tag{2.42}$$

We also have that

$$\mathrm{d}S = \frac{\partial S}{\partial T}\bigg|_v \mathrm{d}T + \frac{\partial S}{\partial v}\bigg|_T \mathrm{d}v = \frac{\partial S}{\partial T}\bigg|_v \mathrm{d}T + \frac{\partial P}{\partial T}\bigg|_v \mathrm{d}v\,, \tag{2.43}$$

where the second equality uses one of the Maxwell relations (see Moran et al. (2010, Sec. 11.3.1)). Inserting Equation 2.42 and Equation 2.43 into the Gibbs' relation, Equation 2.24, we obtain

$$\left[\left(\frac{\partial e}{\partial v}\right)\bigg|_T + p - T\left(\frac{\partial p}{\partial T}\right)\bigg|_v\right]\mathrm{d}v = \left[T\left(\frac{\partial s}{\partial T}\right)\bigg|_v - C_v\right]\mathrm{d}T\,. \tag{2.44}$$

Because T and v are chosen as independent variables, they can be varied independently. In particular, we can let $\mathrm{d}T = 0$ and $\mathrm{d}v \neq 0$ in Equation 2.44, from which it follows that

$$\left(\frac{\partial e}{\partial v}\right)\bigg|_T = T\left(\frac{\partial p}{\partial T}\right)\bigg|_v - p\,, \tag{2.45}$$

which is the constraint the second incomplete EOS needs to satisfy to be thermodynamically consistent. The first assumption we do at this stage is to assume a constant specific heat capacity, $C_v = C_{v,0}$, so that

$$\left(\frac{\partial e}{\partial T}\right)\bigg|_v = C_v \Rightarrow e(v, T) = C_{v,0}(T - T_0) + f(v)\,, \tag{2.46}$$

where $C_{v,0}$ and T_0 are reference specific heat capacity at constant volume and temperature, and $f(v)$ is the function coming from the integration. The next step is to determine $f(v)$ such that Equation 2.45 is satisfied. The second assumption we do, in order to simplify the derivation, is to set $\alpha_\Gamma = 0$ in Equation 2.39, i.e. $\Gamma = \Gamma_0$. With this assumption, we can write that

$$T\left(\frac{\partial p}{\partial T}\right)\bigg|_v = \frac{\Gamma_0}{(v-b)}C_{v,0}T\,, \quad \left(\frac{\partial e}{\partial T}\right)\bigg|_v = \frac{\mathrm{d}f}{\mathrm{d}v}\,,$$

and we find from Equation 2.45 the ODE that the function $f(v)$ should satisfy

$$\frac{\mathrm{d}f}{\mathrm{d}v} + [\Gamma_0 f(v) + C_1]\frac{1}{(v-b)} + \frac{C_2}{(v-b)v^{\alpha_e}} + \frac{C_3}{v^{\alpha_p}} + C_4 = 0\,, \tag{2.47}$$

where the constants are given by

$$
\begin{aligned}
C_1 &= -\Gamma_0 \left(C_{v,0}\, T_0 + e_{r,0} - d_e \right) , \\
C_2 &= -\Gamma_0\, d_e\, v_0^{\alpha_e} , \\
C_3 &= d_p\, v_0^{\alpha_p} , \\
C_4 &= p_{r,0} - d_p .
\end{aligned}
$$

Equation 2.47 is solved symbolically with `WolframAlpha`. However, with no further assumption, the expression of $f(v)$ involves hypergeometric functions, which would make the implementation of the EOS into the code too complex. The distinction is then made between:

- The case for which the co-volume $b = 0$. The expression of $f(v)$ is in this case given by

$$
f(v) = \frac{C_3\, v^{(1-\alpha_p)}}{\alpha_p - \Gamma_0 - 1} + \frac{C_2\, v^{-\alpha_e}}{\alpha_e - \Gamma_0} - \frac{C_4\, v}{\Gamma_0 + 1} - \frac{C_1}{\Gamma_0} + k_1\, v^{-\Gamma_0} .
$$

- The case for which the two exponents $\alpha_p = \alpha_e = 0$, i.e. $p_r = p_{r,0}$ and $e_r = e_{r,0}$. The expression of $f(v)$ is found to be

$$
f(v) = \frac{b\, C_3}{\Gamma_0 + 1} + \frac{b\, C_4}{\Gamma_0 + 1} - \frac{C_3\, v}{\Gamma_0 + 1} - \frac{C_4\, v}{\Gamma_0 + 1} - \frac{C_1}{\Gamma_0} - \frac{C_2}{\Gamma_0} + k_1\, (b - v)^{-\Gamma_0} .
$$

In both expressions of $f(v)$ above, there is still an integration constant k_1 to be determined. This can be done by imposing $f(v_0) = e_{r,0}$,

$$
k_1 : f(v_0, k_1) - e_{r,0} = 0 .
$$

The last step is to express the free parameters present in the constants C_1, C_2, C_3, C_4 in terms of measurable quantities. Those parameters are Γ_0, d_p, d_e, $p_{r,0}$ and $e_{r,0}$. Le Métayer et al. (2004) remark that those constants are systematically determined from experimental data but the nature of the considered data depends on the targeted application. For example, experimental results of shock dynamics inside a material would be considered if the application is the propagation of shock waves inside a solid medium. On the other hand, if we want to simulate the behavior of a liquid in contact with its vapor, the saturation curves will be considered instead. In this work, we adopt the following approach. For a given material, the reference states $p_{r,0}$, $e_{r,0}$ and T_0 are fixed based on the initial flow conditions of the simulations. The reference specific heat, $C_{v,0}$, is set accordingly. The two observable quantities we consider here to determine the free parameters are the isobaric thermal expansion, α_p, and the isothermal compressibility, β_T. Those are particularly relevant for us since we want the material to be simulated to have the correct physical compressibility and thermal expansion. We choose their values based on the selected reference conditions, i.e. $\alpha_p = \alpha_{p,0}$ and $\beta_T = \beta_{T,0}$. Starting from the definitions of α_p and

β_T, Equation 2.29 and Equation 2.30, we obtain two conditions

$$\left(\frac{\partial p}{\partial v}\right)\Big|_T = \frac{-1}{\beta_{T,0}\, v_0}\,, \quad \left(\frac{\partial p}{\partial T}\right)\Big|_v = \frac{\alpha_{p,0}}{\beta_{T,0}}\,. \tag{2.48}$$

The second condition directly leads to the expression of the reference Grüneisen coefficient,

$$\Gamma_0 = \frac{(v_0 - b)\,\alpha_{p,0}}{C_{v,0}\,\beta_{T,0}}\,. \tag{2.49}$$

The first condition is used to determine d_p. Since we only consider two conditions, we set $d_e = d_p$. Finally, the choice of exponents α_p and α_e leads to various EOS models, which makes the proposed approach very flexible:

- $\alpha_p = \alpha_e = b = 0$: this gives an EOS with a form similar to the stiffened gas EOS which reads

$$\begin{aligned} p(\rho, T) &= \rho(\gamma - 1)e - \gamma p_\infty\,, \\ e(\rho, T) &= C_v\, T + \frac{p_\infty}{\rho}\,, \end{aligned} \tag{2.50}$$

  where $\gamma = C_p/C_v$ is the heat capacity ratio and p_∞ is a pressure constant modeling molecular attractive effects.
- $\alpha_p = 1$ and $\alpha_e = b = 0$: this gives an EOS with a form similar to that derived in Latige (2013, Sec. 4.4.3) and used in Simon Peluchon (2017, Sec. 5.4.1) which restores the physical thermal dilatation rate of the substance, contrary to the more simple stiffened gas EOS which over-amplifies the dilatation.
- $\alpha_p = 2$ and $\alpha_e = b = 0$: this type of EOS is a prototype of cubic EOS with the reference pressure having a quadratic dependence on density.
- $b \neq 0$ and $\alpha_p = \alpha_e = 0$: this choice leads to an EOS similar to the NASG EOS, Equation 2.36.

To close this section, Figure 2.2 shows the thermal dilatation behavior of a liquid aluminum slab at atmospheric pressure, modeled by the standard stiffened gas EOS and the second Mie-Gruneisen EOS mentioned above (i.e. $\alpha_p = 1$ and $\alpha_e = b = 0$). The parameters for the stiffened gas EOS are the following ones:

$$\begin{aligned} \gamma &= 3.16\,, \\ C_v &= 970.0\mathrm{J}/(\mathrm{kg\,K})\,, \\ p_\infty &= 1.651\mathrm{e}10\mathrm{Pa}\,. \end{aligned}$$

The parameters injected into the Mie-Gruneisen EOS can be found in Table 11.16, for the liquid phase of the solidifying cylinder test case. This comparison highlights the exaggerated thermal dilatation of the liquid with the common stiffened gas EOS. This is not the case with the Mie-Gruneisen EOS which imposes, by construction, the thermal dilatation coefficient introduced by the user.

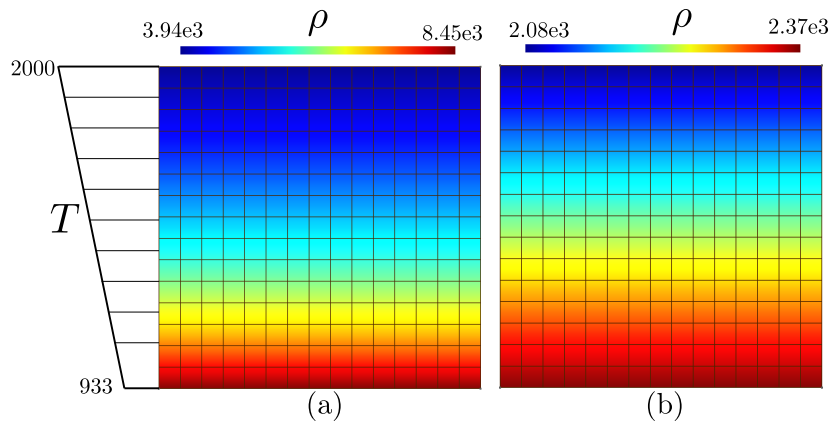


Figure 2.2 / Illustration of the thermal dilation of a liquid aluminum slab with linear temperature gradient modeled by (a) the stiffened gas EOS, (b) the new Mie-Grunseinsen EOS.

2.4 Summary of the equations implemented in this work

The governing equations The compressible Navier-Stokes equations with constant thermophysical properties that are solved in this thesis are written under a flux form which is adapted for its discretization presented in Section 5.2.2:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \underline{\boldsymbol{F}}^c(\mathbf{u}) = \nabla \cdot \underline{\boldsymbol{F}}^d(\mathbf{u}, \nabla \mathbf{u}) + \mathbf{S}(\mathbf{u}, \nabla \mathbf{u}) \,, \qquad [2.51]$$

with the vector of conservative variables $\mathbf{u}$, the convective flux tensor $\underline{\boldsymbol{F}}^c$, the diffusive flux tensor $\underline{\boldsymbol{F}}^d$ and the source term vector $\mathbf{S}$. While the expression of the source term depends on the application, the three other terms have the following general expressions:

$$\mathbf{u} = \begin{bmatrix} \rho \\ \rho \mathbf{v} \\ \rho E \end{bmatrix}, \quad \underline{\boldsymbol{F}}^c = \begin{bmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + p \underline{\boldsymbol{I}} \\ \rho H \mathbf{v} \end{bmatrix}, \quad \underline{\boldsymbol{F}}^d = \begin{bmatrix} 0 \\ \underline{\boldsymbol{\tau}} \\ \underline{\boldsymbol{\tau}} \cdot \mathbf{v} - \mathbf{q} \end{bmatrix}, \qquad [2.52]$$

in which the density in $\mathrm{kg/m^3}$ is noted ρ, the velocity vector in (x, y) Cartesian coordinate system and expressed in m/s

$$\mathbf{v} = \left[v_x, \, v_y \right] ,$$

the static pressure p in $\mathrm{N/m^2}$ and the identity matrix $\underline{\boldsymbol{I}}$. The total energy E and the total enthalpy H per unit mass, with units of J/kg, both include thermal

and kinetic contributions,

$$E = e_{int} + e_{kin} = e_{int} + \frac{1}{2}(\mathbf{v}\cdot\mathbf{v}) \,,$$
$$H = E + \frac{p}{\rho} \,,$$

with the specific internal energy e_{int} and the specific kinetic energy e_{kin}. It can already be remarked here that although the system of equations is written for the unknown conservative variables, this system can be solved in theory for any other valid set of variables. This is discussed in Section 5.3.1.

The transport laws and properties The constitutive equations for the shear stress tensor $\underline{\boldsymbol{\tau}}$ and the heat flux vector $\mathbf{q}$,

$$\underline{\boldsymbol{\tau}} = \mu \left[\nabla\mathbf{v} + (\nabla\mathbf{v})^T - \frac{2}{3}(\nabla\cdot\mathbf{v})\,\underline{\boldsymbol{I}} \right] \,,$$
$$\mathbf{q} = -\lambda\,\nabla T \,,$$

follow the assumptions of a Newtonian fluid and the Fourier law, respectively. The temperature in K is noted T. The dynamic viscosity μ in $\mathrm{kg}/(\mathrm{m\,s})$ and the thermal conductivity λ in $\mathrm{W}/(\mathrm{m\,K})$ are both taken as constant.

The thermodynamic properties and equation of state The system of equations is closed by a volumetric and a caloric EOS,

$$p = p(\rho, T) \,,$$
$$e = e(\rho, T) \,. \qquad [2.53]$$

Inside the EOS expressions lie the isochoric and isobaric heat capacities, respectively noted C_v and C_p, and both expressed in $\mathrm{J}/(\mathrm{kg\,K})$. The isobaric thermal expansion α_p in $1/\mathrm{K}$ and the isothermal compressibility β_T in $1/\mathrm{Pa}$ might also be involved in the thermodynamic relations. All the thermodynamic properties are assumed to be constant.

For the two-phase simulations conducted in this project involving two fluids (gas-gas, gas-liquid or liquid-liquid), two separated sets of governing equations, with distinct EOS and thermodynamic and transport properties, will be considered. They will be coupled at the phase interface through the jump conditions summarized in Section 3.4. For the simulations featuring a gas and a condensed phase undergoing phase transformation, the gas phase will obey the equations above while the condensed liquid-solid system will be modeled by a mixture version of the compressible Navier-Stokes equations presented in Section 4.4.

Contribution 2.1: Equations in the bulk of each phase

The main contributions that can be identified from this chapter are:

- Detailed kinematical and dynamical descriptions of the problem, starting from the most generic models up to the governing equations at stake, including the statement of the required hypotheses.
- A justified choice of the closure models adopted for the liquid phase.
- An original unifying procedure to define new complex EOS compatible with material properties.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- A new internal library providing several thermodynamic models to specify the EOS.
- A new class defining the compressible Navier-Stokes system closed by general EOS.

Those features extend the capabilities of the code which was up to now only able to deal with perfect gas EOS or incompressible models.

3

Modeling of the sharp gas–condensed phase interface

Section 3.2 and Section 3.3 are at the basis of the models implemented into the numerical tool that was the subject of the publication Henneaux et al. (2023f).

3.1 Formulations of multiphase flows 81
3.1.1 Mathematical formulations and level of description . . . 82
3.1.2 Review and choice of local instant compressible multi-phase models 84
3.2 Derivation of the interfacial jump conditions 89
3.2.1 Balance equations for generic quantities 90
3.2.2 Jump conditions for viscous compressible two-phase flows 96
3.3 Constitutive equations at the interface 100
3.3.1 Entropy production and mass transfer 100
3.3.2 Surface tension 104
3.4 Summary of the equations implemented in this work 105

3.1 Formulations of multiphase flows

It was introduced in Section 1.2 that the type of ablation phenomena considered in this treatise involves the presence of several phases. After having derived the governing equations for a single phase in Section 2.2, this chapter shall cover the extension of those models to multiphase flows configurations. Multiphase

flows can be regarded as any fluid flow systems consisting of two or more distinct phases, having some level of phase separation at a scale well above the molecular level. This makes the distinction between immiscible phases, considered here, and miscible phases, which are completely soluble in one another. Let us note that the term "phase" will be used here in the most general sense: two different phases can refer to two different chemical substances (air and water e.g.) or to the same chemical substance with two different phases of matter (steam and liquid water e.g.). Some authors prefer to use two different names, namely multi-fluid and multiphase flows; here we will refer to multiphase flows to designate both situations. Immiscible multiphase flows are characterized by both the combinations of the present phases and the geometry of the interfaces between them (Ishii and Hibiki (2011, Sec. 1.3)). In the physical situation considered in this work, some regions of the flows correspond to a separated flow configuration of a liquid film flowing on a solid surface due to the interaction with a gas jet, others coincide with transitional or even dispersed flows where the liquid film is sprayed into droplets. In this chapter, we shall focus on the way to model the interactions occurring between the gas and condensed phases across the physically sharp interface separating them. The equations employed in this chapter are valid for two interacting fluid phases. However, all the jump conditions remain valid to model the interaction between the gas phase and the condensed phase under its pure solid state. This flexibility is made possible thanks to the unified model selected to describe the liquid-solid region, which is described in Chapter 4.

3.1.1 Mathematical formulations and level of description

The physical nature of a phase interface makes it a three-dimensional region, the thickness of which may be several molecular diameters or more (Slattery et al. (2007, Sec. 1.2.1)). From this observation, three main mathematical formulations of multiphase flows are possible and discussed below.

Phase field formulation

The first kind of formulations considers the physically microscopic diffused nature of a phase interface and can be regrouped under the *phase field methods* (PFM) name. As explained in Nayigizente (2021, Sec. 4.1.2), the starting point of all PFM is the definition of a free energy functional taking into account the local anisotropy of the fluid caused by the presence of the interface. For hydrodynamics problems addressed by the second gradient theory, the free energy typically takes the form of the van der Waals free energy,

$$\mathcal{F} = \mathcal{F}_0(\rho, T) + \frac{1}{2}\chi|\nabla\rho|^2 ,$$

where the energy is decomposed into a bulk energy $\mathcal{F}_0$ and an excess free energy depending on the squared density gradient whose contribution is only important in the interfacial region (Giovangigli (2020)). The governing equations can then be obtained using a variational approach to minimize this energy under the constraint imposed by the second law of thermodynamics. PFM are particularly interesting to study intrinsically local phenomena such as contact angles or phase transition in very small systems or situations where the interface is actually physically thick, namely for near-critical fluids (Nayigizente (2021, Sec. 4)). However, the applicability of PFM to more practical flow situations is limited and will therefore not be further discussed in this work.

Local instant formulation

In the local instant formulation, the interface region is treated as a singular surface. A singular surface is a mathematical idealization of a physical finite-thickness region across which very large changes in some field properties takes place. Because of the continuum assumption made in this work and the targeted applications, it is expected that such a macroscopic *sharp interface* representation will accurately describe the interface, as noted by Tryggvason et al. (2011, Sec. 2.1). Such formulation will result in a moving boundary problem with the position of the interface being an unknown due to the coupling between the field variables and the interface dynamics. Across the interface, the enforcement of jump conditions is required to couple, based on physical laws, the sets of single-phase equations applying in the bulk of each phase. The high level of fidelity of those approaches comes at the expense of mathematical and numerical difficulties, which might be insurmountable, e.g. in the case of highly dispersed flows. By construction, they are however very well suited to address separated two-phase flow configurations, which is one of the motivations to rely on this formulation to couple the gas and condensed phase regions. This chapter presents the derivation of the jump conditions which are at the basis of the local instant formulation. The numerical implementation is detailed in Chapter 8 and Chapter 9.

Averaging formulations

This last family of models, often referred to as *multi-fluid* methods (MFM), tends to treat the interface as a heterogeneous region where the different phases mix together. Contrary to PFM where the diffusion of the interface is justified by physical arguments, the interface region in MFM is diffused numerically, i.e. the variables used to describe the interface in the equations are smeared out but physically, the interface is still considered to be sharp. The diffuse character of the interface is achieved by averaging (typically volume averaging) the one-fluid Navier-Stokes equations such that a single set of equations is solved in the entire computational domain. There exists a wide variety of reduced-order models, ranging from fully out-of-equilibrium seven-equations models where each phase possesses its own temperature, pressure and velocity

to equilibrium homogeneous mixture models with common properties shared by the two phases. It should be noted that the local instant formulation is the fundamental basis for all the MFM, regardless of the type of averaging, since the closure of the terms appearing from the averaging process should integrate the constraints imposed by the jump conditions. The difficulties of explicitly representing discontinuities across moving interfaces are avoided with MFM such that they can be applied to complex multiphase industrial flows. On the other hand, the smearing of the interface may lead to excessive errors in the interface region or numerical instabilities for high contrast interface problems. A deeper overview of this field as well as the derivation of the MFM used in this work for the liquid-solid interface are given in Chapter 4.

To close this section, we present in Table 3.1 a concise and general comparison of the two main modeling paradigms to address multiphase flows. Most of the elements, whether of a numerical or modeling nature, will be dealt with in the following of this treatise.

3.1.2 Review and choice of local instant compressible multiphase models

The choice of a sharp interface local instant formulation

We emphasized in the previous section the two main existing approaches to tackle multiphase flows. We shall now expose the two main reasons for which a sharp interface local instant formulation is chosen in this work.

Targeted physics The focus of this work is on the prediction of the ablation of space debris objects made of materials undergoing melting. As explained in Section 1.2.4, the main driving mechanisms of this ablation process are expected to be the shear stresses exerted by the gas phase on the liquid molten layer and the evaporation of this liquid layer into the gas. Due to the gas stresses and the body forces, the liquid layer will be progressively sprayed into droplets in the wake. Being able to simulate the flow of this kind of droplets cloud with a local instant formulation seems out of reach (at least without any very efficient adaptive strategies). However, our interest lies instead in the melting of the solid surface and the apparition of the molten layer and the dynamics of the liquid layer under the influence of the gas flow until the onset of atomization. Simulating those phenomena is key to have a good prediction of subsequent ones like spraying or break-up. In the flow regions of interest, the interface topology between the gas and the condensed phase is expected to be such that a separated or mixed multiphase flow configuration will be present. It turns out that a local instant formulation to study such flows is possible and this is the first motivation to follow that path.

Level of fidelity As already mentioned, the averaging procedure involved in the multi-fluid models leads to some loss of information that need to be

Table 3.1 / Characteristics of the two main families of models and associated numerical methods to simulate multiphase flows.

	Local instant	Averaging
Modeling strategy	Separated single phase eqs in each phase Jump conditions at the interface	Common averaged eqs over the whole domain Interfacial exchange source terms
Modeling strengths	"Exactly" resolve the interface Best option for separated phases, OK for mixed flows	Work for any number of phases Appropriate for all configurations[a]
Modeling difficulties	Inappropriate for highly dispersed flows	Closure of interfacial terms Thermodynamic mixing between phases
Numerical facilities	Standard single-phase techniques in the bulk	Single set of eqs over the domain, no interface tracking
Numerical difficulties	Enforcement of jump conditions on moving interfaces Tracking of interface location	Treatment of nonconservative terms Increasing interface smearing Smoothing of significant discontinuities

[a]Restricted to monodispersed description of dispersed flows

recovered through the closure of some terms. In some situations, this extra level of modeling is not satisfactory enough. This could be the case for the targeted application. Indeed, it was pointed out in Chang et al. (2013), R.R. Nourgaliev et al. (2008) and Robert Nourgaliev et al. (2005) that a rigorous and exact imposition of the jump conditions is a necessary condition to simulate accurately interfacial instabilities and breakup in a shear-dominated environment. This is the main reason for the adoption of a local instant formulation in this work. Furthermore, the closure of the equations in reduced-order approaches is not universal and should be adapted to the type of flows to be simulated in order to achieve adequate modeling quality. To enable the establishment of closure laws, the data produced by fully resolved Direct Numerical Simulations (DNS) of local interfacial phenomena are very valuable or even indispensable. This is exemplified by the work of Hardy et al. (2022) who developed a particle-resolved DNS framework to feed averaged models of particle flows. One of the goal of the proposed numerical tool could therefore be seen as the production of high-fidelity valuable simulations data of a certain class of two-phase flows that could be exploited by lower fidelity approaches afterward. Besides those two motivations, relying on a local instant formulation naturally offers the possibility to combine optimal single field discretizations within a partitioned approach. Although not exploited in this work, this interesting feature is worth mentioning.

The main challenges and modeling approaches

To benefit from the high resolution features of the local instant formulation, a series of challenges must be handled. We review below the most relevant ones for our application.

Coupled equations The local instant formulation leads to a system of coupled PDEs with dynamical interfaces whose motion often depend on the solution itself. This leads to two levels of complex non-linear couplings: (i) the spatial coupling of the bulk equations of the different phases across the interface, (ii) the temporal coupling between the fluid equations and the transport equation for the interface. Both couplings can be handled either in a partitioned fashion or a monolithic way. This differentiation was introduced in Section 1.3.2 and is further discussed for the spatial coupling in Section 9.1 and for the temporal coupling in Section 10.3.1.

Compressible inviscid effects The compressible treatment of two-phase flows bears some additional difficulties compared to their incompressible counterparts. Stefan Fechter (2015, Sec. 1) indeed notes that the decomposition and separate solution of hydrodynamics and thermodynamics as in the incompressible case is no longer possible. This is because all conservation equations are coupled via an equation of state. This makes the use of the local instant formulation with a diffuse interface region much more complicated. Indeed, applying

a traditional shock-capturing technique to smooth the discontinuities across the interface would lead to the appearance of spurious velocity and pressure fluctuations resulting in computational failure. Abgrall (1996) identifies the cause of those oscillations to be the non-physical nature of the intermediate states in the diffuse region, which lead to an erroneous pressure field when reconstructed from the mixture EOS. In order to avoid the numerical mixing, a sharp interface treatment has to be developed[1], which is more complex to put in place than a diffuse interface from a numerical perspective. On the contrary, for incompressible two-phase flows, the local instant formulation can flawlessly be used with a diffuse interface treatment, like for example in Lee et al. (2017). The most common approach in the compressible two-phase flows community to treat the inviscid effects at the phase interface is to formulate a two-phase Riemann problem incorporating the jumps on the inviscid fluxes, i.e. taking into account surface tension and mass transfer effects. The theory behind this formulation can be found e.g. in Rohde and Zeiler (2018).

Viscous effects The discontinuous nature of the physical quantities at the interface combined with the dependency of the viscous and thermal effects on the quantities gradients make the extension from inviscid to viscous model particularly challenging. Instead of Dirichlet-type jump conditions, in this case Robin-type constraints, which are moreover non-linear, need to be imposed. The compressible nature of the model makes the situation even more difficult. In incompressible two-phase flows, it is usually possible to limit the jumps to the continuity of velocities and the balance between the pressure and normal viscous forces, as done e.g. in Kummer (2017). In a compressible framework, the pressure depends on the solution vector and the energy balance needs to be solved along with the mass and momentum balances. One way to achieve that coupling is by constructing a diffusive generalized Riemann problem including both convective and viscous fluxes, like in Jöns et al. (2023). The more common approach, followed in this work, is to split the convective and diffusive parts in the momentum jumps, similarly to what is done in Houim and Kuo (2013).

Surface tension forces Including the surface tension forces into the jump conditions is easy as they can simply be modeled by one or two (depending on whether Marangoni effects are considered or not) additional terms. What makes the modeling of surface tension effects particularly difficult is the correct estimation of the interface curvature on which the normal pressure jump induced by surface tension forces depend (see Section 3.3.2). This curvature computation is more a numerical challenge and a full chapter, Chapter 7, is dedicated to that aspect.

[1]The other option already mentioned before is to rely on consistent averaging formulations with well-behaved relaxation of the mixture thermodynamics as one traverses from one phase to the other, as noted by Saurel and Pantano (2018).

Phase change Taking into account phase change effects significantly complicates the jump conditions as a non-zero net mass flux induces jumps in normal velocities (meaning that the interface velocity is not equal anymore to the fluid velocities), in pressure (e.g. the recoil pressure in liquid-vapor system) and in heat fluxes. The jump conditions also need to be closed as the apparition of the additional terms introduce new unknowns. The different possibilities are discussed in Section 3.3.1. The distinction between two "types" of phase change models is worth highlighting here:

- *Based on models within the jumps*: the derivation of the local instant formulation implicitly assumes the presence of a phase interface in the domain of computation, at the beginning of the simulation. This interface can separate any type of substances: a liquid and its vapor (e.g. water and steam) or a liquid and a gas from different substances (e.g. water and air). The most rigorous way to accommodate for phase transfer effects is to add an under-compressive evaporation/condensation wave in the waves structure of the two-phase Riemann problem, as done e.g. in Stefan Fechter et al. (2015). In such cases, the jump conditions do not suffice to control the interfacial dynamics completely and an additional macroscopic constraint, encompassing microscale information, need to be added. It should be noted that for a liquid-gas system with different substances, the presence of all the chemical species of the system need to be considered in at least one of the phase, usually the gas phase. Indeed, for evaporation and condensation between a gas and a liquid, the mass transfer rate typically depends on the concentration of the vaporized liquid in the gas region near the interface. Thus, the advection and diffusion of the vaporized liquid species mixed with original gas species need to be accounted for.
- *Based on the EOS*: the other possibility to model phase change is to close the bulk equations by a multi-parameter EOS able to model the behavior of both the liquid and its vapor phases, as well as evaporation/condensation effects. Such complex EOS are commonly based on the free Helmholtz energy and included in the solver via an external fluid library (see Section 2.3.2). Special care must be taken when approaching the conditions (pressure, temperature) at which the phase change occurs, more precisely in between the two metastable regions. Indeed, in the intermediate region in-between, called the spinodal region, the pressure curve may become non-convex, resulting in non-physical states and changing the hyperbolic part of the system to an elliptic part. The transition between the vapor and liquid phases can be accomplished in a diffuse interface manner, as in F. Föll et al. (2019), or via a sharp interface approach, like in S. May and Thein (2023). In the former case, the spinodal region can be avoided by using the homogeneous equilibrium model. In the latter case, the jump conditions of the local instant formulation need to be enforced and physical states are selected based on an entropy inequality.

The combination of the two situations listed above into a single formulation

is possible and is presented e.g. in Houim and Kuo (2013). This allows phase change to occur across the initially present phase interface as well as inside the bulk of each phase. In this work, we limit ourselves to the first approach and do not consider any EOS able to model phase transition.

Multispecies As it will be commented in Section 3.2.2, the jump conditions can be extended to take into account the balance of partial densities at the interface, including mass transfer, species diffusion and interfacial source production/depletion. This is realized by Houim and Kuo (2013) in the framework of a sharp interface compressible method.

Multiscale nature It should be mentioned that in turbulent flows, we can avoid the introduction of any extra modeling in the Navier-Stokes equations if the numerical resolution is such that it is able to capture the smallest turbulent structures at the Kolmogorov scale. In dispersed multiphase flows, there is no such smallest scale to be resolved since for instance a liquid droplet immersed in gas flow can become infinity small if it evaporates. In such configurations, there will also be the need for additional modeling to be incorporated into the local instant formulation.

In this thesis, we build upon a local instant formulation based on the compressible Navier-Stokes equations where the jump conditions are enforced in a strongly coupled and sharp manner, where both inviscid and viscous effects are considered, and for which surface tension forces and phase change are included. On the other hand, the presence of several species in the flow is not considered as part of this work. The implementation of multispecies models clearly represent one of the priority perspectives of our work.

We focused in this section on challenges related to the modeling. The sharp interface treatment implied by the use of the local instant formulation poses also important numerical difficulties: the tracking of the interface, the representation of discontinuous solutions and the enforcement of the physical jump conditions. The first aspect is covered in Section 6.1, the second in Section 8.1 and the last one in Section 9.1.

3.2 Derivation of the interfacial jump conditions

In many branches of continuum mechanics, the concept of singular surface across which jump discontinuities occur in some mechanical or thermodynamical properties variables, or in their spatial or temporal partial derivatives of some order, plays a major role. The local partial differential equations developed in Section 2.2.1 are valid at every point in the system for which the solution is continuous. For the points belonging to the interface, discontinuous properties prevent the use of the classical formulation. Instead, jump conditions

relating the quantities on both sides of the interface should be computed. The derivation of the interface jump conditions can be achieved following different paths, although they all lead to similar results. The most common ones are:

- *Material volume*: the first approach consists in considering a material volume around the interface, which is divided into two non-material volumes by the moving interface inside it. This is the approach followed by Delhaye (1974) and Guédon (2013, Sec. 2.1) and extended by I. Müller (1985, Sec. 3.1) and Dreyer (2005) to consider interfacial density.
- *Control volume*: the second approach uses a control volume that includes the two phases and moves at the interface velocity. It is followed by Ishii and Hibiki (2011, Sec. 1.2) and Faghri and Yuwen Zhang (2006, Sec. 3.3.6). The former includes interfacial density while the second does not.
- *Distribution*: the last identified possibility, presented in Kataoka (1986), is to write the local differential equations valid in all the domain by multiplying the properties by an indicator function. The discontinuous functions present in the equations are then treated in the sense of distributions such that differentiation is possible. This avoids the need for integral theorems, as is the case with the first two approaches.

In this treatise, the material volume approach is adopted for the sake of consistency with the derivation of the bulk equations in Section 2.2.1. Let us recall also that the master balance laws are originally expressed with respect to a material volume. As done in Section 2.1, the generic expression of the jump conditions is first developed before being stipulated to the system of the Navier-Stokes equations considered as part of this work.

3.2.1 Balance equations for generic quantities

The development of the jump conditions in this section is inspired by Dreyer (2005), Abeyaratne (2012) and Casey (2011). Similarly to the derivation of the bulk equations, we consider an observed system/body of material particles, $\mathcal{S}_{obs}$, which is embedded in the Euclidean space by a set of mappings called the configurations, noted ζ_{obs}. We introduce two configurations: a fixed reference one defining the reference material domain in space, $\mathcal{D}^*_{mat}$, and the current configuration to track the fixed set of moving material particles, $\mathcal{D}_{mat}$. The position vector in the reference configuration is denoted by $\mathbf{X}^*$ and the corresponding one in the current configuration by $\mathbf{X}$, as before. Both are linked by a time-dependent homeomorphism, $\omega(\mathbf{X}^*, t)$. At any instant of time, $\omega(\mathbf{X}^*, t)$ may be used to "push-forward" sets of points from the domain $\mathcal{D}^*_{mat}$ to $\mathcal{D}_{mat}$. Conversely, $\omega^{-1}(\mathbf{X}, t)$ may be used to "pull-back" points from $\mathcal{D}_{mat}$ to $\mathcal{D}^*_{mat}$. This is illustrated in Figure 3.1.

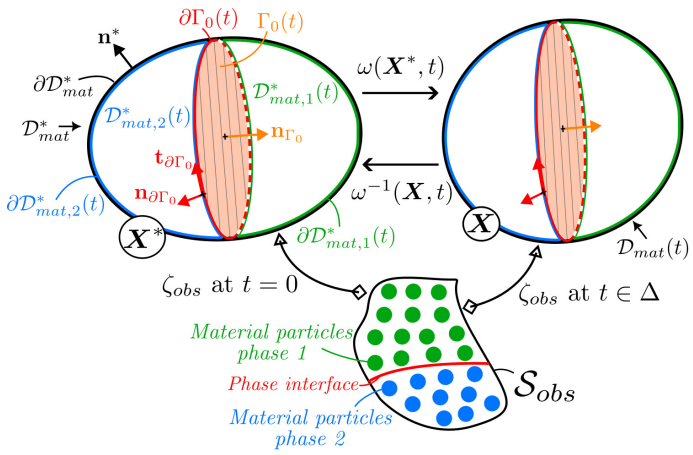


Figure 3.1 / Material volume cut by a moving interface with the notations used in the derivation of the jump conditions.

Decomposition of the integral balance

The peculiarity of the situation considered here is the presence of a smooth orientable singular surface $\Gamma(t)$ that divides the current material domain into two contiguous parts $\mathcal{D}_1(t)$ and $\mathcal{D}_2(t)$ such that $\mathcal{D}_{mat}(t) = \mathcal{D}_1(t) \cup \mathcal{D}_2(t)$. The boundaries of the subdomain containing phase k can be decomposed into $\partial\mathcal{D}_k(t) = \partial\mathcal{D}'_k(t) \cup \Gamma(t)$, where $\partial\mathcal{D}'_k(t) = \partial\mathcal{D}_{mat}(t) \cap \mathcal{D}_k(t)$. While $\partial\mathcal{D}'_k$ is a material boundary as a part of the boundary of the material domain $\mathcal{D}_{mat}$, the interface Γ is, in the most general case, not a material boundary since exchange of mass is possible between $\mathcal{D}_1$ and $\mathcal{D}_2$. Because $\mathcal{D}_{mat}(t)$ is material, the generic balance law derived for the bulk equations, Equation 2.12, still holds. To stay the most generic possible and following Dreyer (2005, Sec. 5.1) and Ishii and Hibiki (2011, Sec. 1.2.1), we split the field density φ_i into a volumetric part, φ_i^v, defined over $\mathcal{D}_{mat}$ and a surface density, φ_i^s, defined over Γ, such that the material derivative of the corresponding extensive property φ_e becomes

$$\begin{aligned}\frac{D\varphi_e(t)}{Dt} = &\frac{\partial}{\partial t}\int_{\mathcal{D}_1(t)} \varphi^v_{i,\mathbf{X},1}(\mathbf{X},t)\,\mathrm{d}\mathbf{X} + \frac{\partial}{\partial t}\int_{\mathcal{D}_2(t)} \varphi^v_{i,\mathbf{X},2}(\mathbf{X},t)\,\mathrm{d}\mathbf{X}\\ &+ \frac{\partial}{\partial t}\int_{\Gamma(t)} \varphi^s_{i,\mathbf{X}}(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\,,\end{aligned} \qquad [3.1]$$

where $\varphi^v_{i,\mathbf{X},k}$ is the volumetric field density associated to phase k. Accordingly, we decompose the flux into the normal flux across the material boundaries of $\mathcal{D}_{mat}$, noted $\mathfrak{F}^v$, and an interface flux, $\mathfrak{F}^s$, which is tangential to Γ and normal

to its boundary $\partial\Gamma$,

$$\begin{aligned}\int_{\partial\mathcal{D}_{mat}(t)} \mathfrak{F}(\mathbf{X},t)\cdot\mathbf{n}(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} &= \int_{\partial\mathcal{D}_1'(t)} \mathfrak{F}_1^v(\mathbf{X},t)\cdot\mathbf{n}_1(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\\ &+ \int_{\partial\mathcal{D}_2'(t)} \mathfrak{F}_2^v(\mathbf{X},t)\cdot\mathbf{n}_2(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\\ &+ \int_{\partial\Gamma(t)} \mathfrak{F}^s(\mathbf{X},t)\cdot\mathbf{n}_{\partial\Gamma}(\mathbf{X},t)\,\mathrm{d}l_{\mathbf{X}}\,,\end{aligned} \tag{3.2}$$

where $\mathbf{n}_{\partial\Gamma}$ is the normal vector to the interface boundary and $\mathrm{d}l_{\mathbf{X}}$ is the line differential element. The production term is also written as the sum between a volumetric part, $\mathfrak{P}^v$, and a surface part, $\mathfrak{P}^s$, leading to

$$\int_{\mathcal{D}_{mat}(t)} \mathfrak{P}(\mathbf{X},t)\,\mathrm{d}\mathbf{X} = \int_{\mathcal{D}_1(t)} \mathfrak{P}_1^v(\mathbf{X},t)\,\mathrm{d}\mathbf{X} + \int_{\mathcal{D}_2(t)} \mathfrak{P}_2^v(\mathbf{X},t)\,\mathrm{d}\mathbf{X} + \int_{\Gamma(t)} \mathfrak{P}^s(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\,. \tag{3.3}$$

Before going further, we can rearrange the two first terms in the right-hand side of Equation 3.2 such that

$$\begin{aligned}\int_{\partial\mathcal{D}_k'(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_k(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} &= \int_{\partial\mathcal{D}_k'(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_k(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\\ &+ \int_{\Gamma(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_\Gamma(\mathbf{X},t)t_k\,\mathrm{d}S_{\mathbf{X}}\\ &- \int_{\Gamma(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_\Gamma(\mathbf{X},t)t_k\,\mathrm{d}S_{\mathbf{X}}\\ &= \int_{\partial\mathcal{D}_k(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_k(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\\ &- \int_{\Gamma(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_\Gamma(\mathbf{X},t)t_k\,\mathrm{d}S_{\mathbf{X}}\\ &= \int_{\mathcal{D}_k(t)} \nabla\cdot\mathfrak{F}_k^v(\mathbf{X},t)\,\mathrm{d}\mathbf{X}\\ &- \int_{\Gamma(t)} \mathfrak{F}_k^v(\mathbf{X},t)\cdot\mathbf{n}_\Gamma(\mathbf{X},t)t_k\,\mathrm{d}S_{\mathbf{X}}\,,\end{aligned} \tag{3.4}$$

where

$$t_k = \begin{cases} 1 & \text{if } k=1\,,\\ -1 & \text{if } k=2\,,\end{cases}$$

because the interface normal, $\mathbf{n}_\Gamma$, is defined as pointing outwards from phase $k=1$ and the normal to the boundary $\mathbf{n}_k$ is also defined as pointing outwards. Adding and subtracting the integrals over the interface implicitly assumes that at each instant of time, a finite one-sided limit of $\mathfrak{F}_k^v$ exists as $\Gamma(t)$ is approached from one side or the other, depending on the phase k. Based on those manipulations, the sum of the two first terms on the right-hand side of Equation 3.2

turns into

$$\int_{\partial\mathcal{D}_1'(t)} \mathfrak{F}_1^v(\mathbf{X},t)\cdot\mathbf{n}_1(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} + \int_{\partial\mathcal{D}_2'(t)} \mathfrak{F}_2^v(\mathbf{X},t)\cdot\mathbf{n}_2(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} = \int_{\mathcal{D}_{mat}(t)} \nabla\cdot\mathfrak{F}^v(\mathbf{X},t)\,\mathrm{d}\mathbf{X} - \int_{\Gamma(t)} \left(\mathfrak{F}_1^v(\mathbf{X},t)-\mathfrak{F}_2^v(\mathbf{X},t)\right)\cdot\mathbf{n}_\Gamma(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\,.$$

The last term in Equation 3.2 can be transformed from a line integral over the contour of the interface to a surface integral over the interface by using the Green's formula for a surface (see e.g. Aris (1962, Sec 9)):

$$\int_{\partial\Gamma(t)} \mathfrak{F}^s(\mathbf{X},t)\cdot\mathbf{n}_{\partial\Gamma}(\mathbf{X},t)\,\mathrm{d}l_{\mathbf{X}} = \int_{\Gamma(t)} \nabla_{\Gamma,t}\cdot\mathfrak{F}^s(\mathbf{X},t)(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}}\,, \qquad [3.5]$$

where the tangential gradient operator on the interface is obtained as

$$\nabla_{\Gamma,t} = \nabla - \mathbf{n}_\Gamma(\mathbf{n}_\Gamma\cdot\nabla)\,. \qquad [3.6]$$

Transport theorems with singular interface

As far as Equation 3.1 is concerned, we can notice that time derivatives of integrals over time-dependent *non-material* domains need to be evaluated. This requires us to reconsider the transport theorem presented in Equation 2.11 which was developed for time derivatives of integrals over time-dependent material domains. The starting point is to realize that none of the three steps listed at the end of Section 2.1.2 imposes $\mathcal{D}_{mat}(t)$ to be a material domain to develop the transport theorem. It is only the physical principles we invoked in Section 2.1.3 that are formulated on material domains (and which necessitate a transport theorem to change the formulation type). This observation therefore suggests us that to reconsider the transport theorem, instead of selecting the material motion $\omega(\mathbf{X}^*,t)$, we could just as easily use an alternative mapping $\omega_{nm}(\mathbf{X}^*,t)$ defining a motion which is non-material. More precisely, given the situation we are dealing with, let us define the mapping $\omega_{nm}(\mathbf{X}^*,t)$ such that the current non-material domain, $\mathcal{D}_{nm}(t)$, has a part of its boundary, $\partial\mathcal{D}_{nm}^a(t)$, which moves with the material particles velocity (hence it is a material boundary) and the other part, $\partial\mathcal{D}_{nm}^b(t)$, which moves with a prescribed velocity. The time derivative of this new mapping defines a velocity,

$$\mathbf{u}_{\mathbf{X}^*} = \frac{\mathrm{d}\omega_{nm}}{\mathrm{d}t}\,,$$

which is equal to the material velocity on $\partial\mathcal{D}_{nm}^a(t)$ and the prescribed velocity on $\partial\mathcal{D}_{nm}^b(t)$. If we (i) use this mapping in the steps to derive the generalized ALE Reynolds transport theorem, (ii) take $\partial\mathcal{D}_{nm}^a(t) = \partial\mathcal{D}_k'(t)$, $\partial\mathcal{D}_{nm}^b(t) = \Gamma(t)$ and the prescribed velocity to be equal to the interface velocity $\mathbf{v}_\Gamma$, (iii) stipulate

the ALE transport theorem to an Eulerian formulation, we can show[2] that the time derivative of the integrals over the moving non-material domains $\mathcal{D}_k(t)$ in Equation 3.1 can be expressed with respect to the fixed reference Eulerian domain $\mathcal{D}_{ref} = \mathcal{D}_{geo}$ as

$$\begin{aligned}\frac{\partial}{\partial t}\int_{\mathcal{D}_k(t)} \varphi^v_{i,\mathbf{X},k}(\mathbf{X},t)\,\mathrm{d}\mathbf{X} = &\int_{\mathcal{D}_{ref,k}} \frac{\partial \varphi^v_{i,\mathbf{x},k}}{\partial t}(\mathbf{x},t)\,\mathrm{d}\mathbf{x} \\ &+ \int_{\partial\mathcal{D}'_{ref,k}} (\varphi^v_{i,\mathbf{x},k}\mathbf{v}_{\mathbf{x},k})(\mathbf{x},t)\cdot\mathbf{n}_k(\mathbf{x},t)\,\mathrm{d}\mathbf{S}_\mathbf{x} \\ &+ \int_{\Gamma_{ref}(t)} (\varphi^v_{i,\mathbf{x},k}\mathbf{v}_{\mathbf{x},\Gamma})(\mathbf{x},t)\cdot\mathbf{n}_\Gamma(\mathbf{x},t)t_k\,\mathrm{d}\mathbf{S}_\mathbf{x}\,.\end{aligned} \tag{3.7}$$

Equation 3.7 allows to transform the material time derivative of a field density integrated over a moving non-material domain into the time derivative of this same field represented with respect to the spatial (Eulerian) coordinates $\mathbf{x}$ and integrated over the fixed reference configuration. A direct consequence of this transformation is the addition of the convective fluxes accounting for the efflux of material particles for the observer located in the reference domain. The second term on the right-hand side is the standard normal flux of φ^v_i across the fixed reference boundaries of the control volume with the local Eulerian fluid velocity $\mathbf{v}_\mathbf{x}$. The third term is the normal convective flux originating from the motion of the interface with a velocity $\mathbf{v}_{\mathbf{x},\Gamma}$. It is worth mentioning that the reference image of the interface, $\Gamma_{ref}(t)$, will be time-dependent in the most general case where mass transfer occurs across it. This means that for every fixed control volume crossed by an interface we will consider in our Eulerian formulation, we will need to pay attention to the way in which this propagating boundary is handled in comparison to the other fixed boundaries. This issue is addressed in Section 10.2. Similarly to the rearrangement operated in Equation 3.4, the second term on the right-hand side of Equation 3.7 can be rewritten as

$$\begin{aligned}\int_{\partial\mathcal{D}'_{ref,k}} (\varphi^v_{i,\mathbf{x},k}\mathbf{v}_{\mathbf{x},k})(\mathbf{x},t)\cdot\mathbf{n}_k(\mathbf{x},t)\,\mathrm{d}\mathbf{S}_\mathbf{x} = &\int_{\mathcal{D}_{ref,k}(t)} \nabla\cdot(\varphi^v_{i,\mathbf{x},k}\mathbf{v}_{\mathbf{x},k})(\mathbf{x},t)\,\mathrm{d}\mathbf{x} \\ &- \int_{\Gamma_{ref}(t)} (\varphi^v_{i,\mathbf{x},k}\mathbf{v}_{\mathbf{x},k})(\mathbf{x},t)\cdot\mathbf{n}_\Gamma(\mathbf{x},t)t_k\,\mathrm{d}\mathbf{S}_\mathbf{x}\,.\end{aligned} \tag{3.8}$$

We also need a transport theorem to put the time derivative inside the last integral in Equation 3.1. The derivation of the required surface transport theorem is not detailed here and can be consulted e.g. in Dziuk and Elliott (2007, Sec. 9) or Cermelli et al. (2005). As sketched in Figure 3.1, let us consider the evolving interface $\Gamma(t)$ oriented by its unit outward pointing normal $\mathbf{n}_\Gamma(\mathbf{x},t)$ with the

[2] A detailed derivation of the transport theorem involving the successive steps of proving the theorem on (i) material volume without singular interface, (ii) non-material volume without singular interface, and (iii) material volume with singular interface is provided by Casey (2011, Sec. 4). A more intuitive development considering an arbitrary moving control volume can also be found in Sonin (2001).

boundary-curve $\partial\Gamma(t)$ with unit outward pointing normal $\mathbf{n}_{\partial\Gamma}(\mathbf{x},t)$. $\mathbf{n}_{\partial\Gamma}$ is thus normal to $\partial\Gamma$ but tangent to Γ. Let the interface velocity, $\mathbf{v}_\Gamma$, be decomposed into a normal and tangential components such that

$$\mathbf{v}_\Gamma = v_{\Gamma,n}\mathbf{n}_\Gamma + \mathbf{v}_{\Gamma,t}\,. \tag{3.9}$$

In particular, we can select the interface tangential velocity $\mathbf{v}_{\Gamma,t}$ such that the velocity of the interface boundary, $\mathbf{v}_{\partial\Gamma}$, can be written as

$$\mathbf{v}_{\partial\Gamma} = (\mathbf{v}_{\Gamma,t}\cdot\mathbf{n}_{\partial\Gamma})\mathbf{n}_{\partial\Gamma} + v_{\Gamma,n}\mathbf{n}_\Gamma\,.$$

Let us note that the mean curvature of the interface, noted κ_Γ, is obtained as

$$\kappa_\Gamma = -\nabla_{\Gamma,t}\cdot\mathbf{n}_\Gamma\,. \tag{3.10}$$

Applying the surface transport theorem to the last term in Equation 3.1, we get

$$\begin{aligned}\frac{\partial}{\partial t}\int_{\Gamma(t)}\varphi^s_{i,\mathbf{X}}(\mathbf{X},t)\,\mathrm{d}S_{\mathbf{X}} &= \int_{\Gamma_{ref}(t)}\frac{\partial\varphi^s_{i,\mathbf{X}}}{\partial t}(\mathbf{x},t) + \mathbf{v}_\Gamma\cdot\nabla\varphi^s_{i,\mathbf{X}}(\mathbf{x},t)\\ &\quad + \varphi^s_{i,\mathbf{X}}\nabla_{\Gamma,t}\cdot\mathbf{v}_\Gamma(\mathbf{x},t)\,\mathrm{d}S_{\mathbf{x}}\\ &= \int_{\Gamma_{ref}(t)}\frac{\partial\varphi^s_{i,\mathbf{X}}}{\partial t}(\mathbf{x},t) + v_{\Gamma,n}\left[\mathbf{n}_\Gamma\cdot\nabla\varphi^s_{i,\mathbf{X}}(\mathbf{x},t) - \kappa_\Gamma\varphi^s_{i,\mathbf{X}}(\mathbf{x},t)\right]\\ &\quad + \nabla_{\Gamma,t}\cdot(\varphi^s_{i,\mathbf{X}}\mathbf{v}_{\Gamma,t})(\mathbf{x},t)\,\mathrm{d}S_{\mathbf{x}}\,,\end{aligned} \tag{3.11}$$

where the passage from the first to the second equality uses the two following relations (see Dziuk and Elliott (2007, Sec. 2.2)),

$$\begin{aligned}\nabla_\Gamma\cdot\mathbf{v}_\Gamma &= \nabla_\Gamma\cdot(v_{\Gamma,n}\mathbf{n}_\Gamma) + \nabla_\Gamma\cdot\mathbf{v}_{\Gamma,t} = -v_{\Gamma,n}\kappa_\Gamma + \nabla_\Gamma\cdot\mathbf{v}_{\Gamma,t}\,,\\ \mathbf{v}_\Gamma\cdot\nabla\varphi^s_{i,\mathbf{X}} &= v_{\Gamma,n}\mathbf{n}_\Gamma\cdot\nabla\varphi^s_{i,\mathbf{X}} + \mathbf{v}_{\Gamma,t}\cdot\nabla\varphi^s_{i,\mathbf{X}} = v_{\Gamma,n}\mathbf{n}_\Gamma\cdot\nabla\varphi^s_{i,\mathbf{X}} + \mathbf{v}_{\Gamma,t}\cdot\nabla_{\Gamma,t}\varphi^s_{i,\mathbf{X}}\,.\end{aligned}$$

Local jump conditions

We are now in a position to gather all the previously derived relations and extract the generic form of the local jump conditions across the interface. By inserting into Equation 2.12 the various relations valid for a material domain containing a moving singular interface separating the domain into two phases and using the fact that

$$\varphi_e = \varphi_{e,1} + \varphi_{e,2}\,,$$

to assemble integrals over the respective phases into a single integral, we reach the following balance law expression[3]:

$$\begin{aligned}
&\int_{\mathcal{D}_{ref}} \frac{\partial \varphi_i^v}{\partial t} + \nabla \cdot (\varphi_i^v \mathbf{v})\, \mathrm{d}\mathbf{x} + \int_{\Gamma_{ref}(t)} \left[\varphi_{i,1}^v (\mathbf{v}_\Gamma - \mathbf{v}_1) - \varphi_{i,2}^v (\mathbf{v}_\Gamma - \mathbf{v}_2)\right] \cdot \mathbf{n}_\Gamma \, \mathrm{d}S_\mathbf{x} \\
&+ \int_{\Gamma_{ref}(t)} \frac{\partial \varphi_i^s}{\partial t} + \mathbf{v}_\Gamma \cdot \nabla \varphi_i^s + \varphi_i^s \nabla_{\Gamma,t} \cdot \mathbf{v}_\Gamma \, \mathrm{d}S_\mathbf{x} \\
&= \int_{\mathcal{D}_{ref}} \nabla \cdot \mathfrak{F}^v \, \mathrm{d}\mathbf{x} - \int_{\Gamma_{ref}(t)} \left[\mathfrak{F}_1^v - \mathfrak{F}_2^v\right] \cdot \mathbf{n}_\Gamma \, \mathrm{d}S_\mathbf{x} + \int_{\Gamma_{ref}(t)} \nabla_{\Gamma,t} \cdot \mathfrak{F}^s \, \mathrm{d}S_\mathbf{x} \\
&+ \int_{\mathcal{D}_{ref}} \mathfrak{P}^v \, \mathrm{d}\mathbf{x} + \int_{\Gamma_{ref}(t)} \mathfrak{P}^s \, \mathrm{d}S_\mathbf{x} \,.
\end{aligned} \tag{3.12}$$

Noting that Equation 3.12 has been obtained considering a material domain on which the bulk balance laws presented in Section 2.1.3 also hold, the terms integrated over $\mathcal{D}_{ref}$ vanish. Moreover, since the equation must apply to any $\mathcal{D}_{ref}$, and therefore to any Γ_{ref}, we can obtain the pointwize differential formulation of the jump conditions by localization:

$$\underbrace{[\![\varphi_i^v (\mathbf{v} - \mathbf{v}_\Gamma)]\!] \cdot \mathbf{n}_\Gamma}_{(a)} \underbrace{- \frac{\partial \varphi_i^s}{\partial t} - \mathbf{v}_\Gamma \cdot \nabla \varphi_i^s}_{(b)} \underbrace{- \varphi_i^s \nabla_{\Gamma,t} \cdot \mathbf{v}_\Gamma}_{(c)} = \underbrace{[\![\mathfrak{F}^v]\!] \cdot \mathbf{n}_\Gamma}_{(d)} \underbrace{- \nabla_{\Gamma,t} \cdot \mathfrak{F}^s}_{(e)} \underbrace{- \mathfrak{P}^s}_{(f)} , \tag{3.13}$$

where the jump operator across the interface at a position $\mathbf{x} \in \Gamma$ for some quantity η defined in the bulk of each phase is given by

$$[\![\eta]\!] = (\eta_{1|_\Gamma} - \eta_{2|_\Gamma}) \,, \quad \text{with} \quad \eta_{k|_\Gamma} = \lim_{\epsilon \to 0, \epsilon > 0} \eta(\mathbf{x} \pm \epsilon \mathbf{n}_\Gamma) \,, \tag{3.14}$$

where the adequate sign is related to the phase considered according to the determined orientation of the normal. The interpretation of the different terms in Equation 3.13 is as follows: (a) is the net normal flux of the quantity φ_i^v across the interface (which is null if the interface moves with the fluid velocity), (b) is the time rate of change of the surface field density φ_i^s from an observer moving at the interface velocity, (c) is due to the surface dilatation/contraction, (d) is the net normal flux due to interactions with the external environment, (e) is the line flux along the interface, and (f) is the interface source.

3.2.2 Jump conditions for viscous compressible two-phase flows

The generic jump conditions in Equation 3.13 can now be specified to the system of Navier-Stokes equations we are considering in this work. The first hypothesis we make consists in neglecting the interfacial inertia for all the conserved quantities, i.e. $\varphi_i^s = 0$. In the momentum jump, taking into account that effect is

[3] The $\bullet_\mathbf{x}$ subscript is omitted from now on for the sake of readability as all the equations presented in the following will be based on an Eulerian formulation.

typically important for surfactant dynamics applications, see Manikantan and Squires (2020) for an extensive overview of the topic. Concerning the energy jump, Ishii and Hibiki (2011, Sec. 5.1.1) notify about the fact that even though the mass and momentum associated to the interface itself can be neglected, the surface energy may not be insignificant due to the energy associated to the surface tension. However, we make the assumption of neglecting also the interface energy in this treatise. Not disregarding the interfacial terms would mean that a surface PDE should also be solved in addition to the one in the bulk and this would require a dedicated numerical approach like e.g. the trace finite element method for coupled bulk-interface transport problems described in Gross et al. (2015). The generic form of the jump conditions used in the following thus becomes

$$[\![\varphi_i^v(\mathbf{v}-\mathbf{v}_\Gamma)]\!]\cdot\mathbf{n}_\Gamma = [\![\mathfrak{F}^v]\!]\cdot\mathbf{n}_\Gamma - \nabla_{\Gamma,t}\cdot\mathfrak{F}^s - \mathfrak{P}^s\,. \tag{3.15}$$

Mass

For the mass balance, we set $\varphi_i^v = \rho$ and all the terms on the right-hand side of Equation 3.15 to zero. This results in the equality (in absolute value) of the net mass fluxes across the interface,

$$\rho_1(\mathbf{v}_1-\mathbf{v}_\Gamma)\cdot\mathbf{n}_\Gamma = \rho_2(\mathbf{v}_2-\mathbf{v}_\Gamma)\cdot\mathbf{n}_\Gamma \triangleq \dot{m}_\Gamma\,. \tag{3.16}$$

Remarks on partial densities If the presence of multispecies was accounted for, some interfacial species balance would be needed. The volumetric field density would be the partial densities of each species s, $\varphi_i^v = \rho_s$. The species diffusion flux is noted, like in the bulk, as $\mathfrak{F}^v = \mathbf{J}_s = \rho_s\mathbf{V}_s$ and there may be some chemical reactions occurring on the interface (e.g. heterogeneous reactions between the two phases) such that $\mathfrak{P}^s = \dot{\omega}^s$. This yields the following jumps:

$$\left[\rho_{s,1}(\mathbf{v}_1-\mathbf{v}_\Gamma) - \rho_{s,2}(\mathbf{v}_2-\mathbf{v}_\Gamma)\right]\cdot\mathbf{n}_\Gamma = \left[\rho_{s,1}\mathbf{V}_{s,1} - \rho_{s,2}\mathbf{V}_{s,2}\right]\cdot\mathbf{n}_\Gamma + \dot{\omega}^s\,,$$

which can be rewritten as

$$\dot{m}_\Gamma(Y_{s,1}-Y_{s,2}) = \left[\rho_{s,1}\mathbf{V}_{s,1} - \rho_{s,2}\mathbf{V}_{s,2}\right]\cdot\mathbf{n}_\Gamma + \dot{\omega}^s\,,$$

using the mass fraction definition of the sth-species in phase k, $Y_{s,k} = \rho_{s,k/\rho_k}$.

Linear momentum

For the linear momentum balance at the interface, we set $\varphi_i^v = \rho\,\mathbf{v}$, the flux is due to the stresses on the interface, $\mathfrak{F}^v = \underline{\boldsymbol{\sigma}}$, and there is an interface flux due to an interface stress tensor which will be noted $\mathfrak{F}^s = \mathfrak{F}^s_{\text{mom}}$ for the moment and be specified in Section 3.3. There is no interface source of momentum, $\mathfrak{P}^s = 0$.

Thus, the vector jump conditions for the momentum vector are

$$[\![\rho \mathbf{v}(\mathbf{v}-\mathbf{v}_\Gamma)]\!]\cdot\mathbf{n}_\Gamma + [\![p\underline{\boldsymbol{I}}]\!]\cdot\mathbf{n}_\Gamma = [\![\underline{\boldsymbol{\tau}}]\!]\cdot\mathbf{n}_\Gamma - \nabla_{\Gamma,t}\cdot\mathfrak{F}^s_{\text{mom}}\,.$$

Using the mass jump condition, Equation 3.16, the jumps can be reformulated as

$$\dot{m}_\Gamma[\![\mathbf{v}]\!] + [\![p]\!]\mathbf{n}_\Gamma = [\![\underline{\boldsymbol{\tau}}]\!]\cdot\mathbf{n}_\Gamma - \nabla_{\Gamma,t}\cdot\mathfrak{F}^s_{\text{mom}}\,. \qquad [3.17]$$

Remarks on angular momentum The interface stress tensor can be decomposed in normal and tangential components, $\mathfrak{F}^s_{\text{mom}} = \mathfrak{F}^s_{n,\text{mom}}\mathbf{n}_\Gamma + \mathfrak{F}^s_{t,\text{mom}}\mathbf{t}_\Gamma$. Based on the expression derived by I. Müller (1985, Sec. 3.2.1), Dreyer (2005) shows that, in the absence of interface internal angular momentum, the jump on the angular momentum leads to the conclusion that the tangential interface stress tensor is symmetric while the normal interface stress tensor should be null,

$$\mathfrak{F}^s_{n,\text{mom}} = 0\,, \quad \mathfrak{F}^s_{t,\text{mom}} = \left(\mathfrak{F}^s_{t,\text{mom}}\right)^T\,. \qquad [3.18]$$

Total energy

For the total energy, $\varphi^v_i = \rho\, E$, both the rate of work done on the interface and the heat flux contribute to the interface flux, i.e. $\mathfrak{F}^v = \underline{\boldsymbol{\sigma}}\cdot\mathbf{v} - \mathbf{q}$, and there may be some surface energy source (e.g. radiation), noted $\mathfrak{P}^s = \dot{q}_\Gamma$. The rate of work due to interface line flux is given by $\mathfrak{F}^s = \mathfrak{F}^v_{\text{mom}}\cdot\mathbf{v}_\Gamma$. The jump condition on the total energy then reads

$$[\![\rho E(\mathbf{v}-\mathbf{v}_\Gamma)]\!]\cdot\mathbf{n}_\Gamma + [\![p\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma = [\![\underline{\boldsymbol{\tau}}\cdot\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma - [\![\mathbf{q}]\!]\cdot\mathbf{n}_\Gamma - \nabla_{\Gamma,t}\cdot(\mathfrak{F}^s_{\text{mom}}\cdot\mathbf{v}_\Gamma)\,.$$

Using the mass jump condition, Equation 3.16, the jump can be reformulated as

$$\dot{m}_\Gamma[\![E]\!] + [\![p\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma = [\![\underline{\boldsymbol{\tau}}\cdot\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma - [\![\mathbf{q}]\!]\cdot\mathbf{n}_\Gamma - \nabla_{\Gamma,t}\cdot(\mathfrak{F}^s_{\text{mom}}\cdot\mathbf{v}_\Gamma) + \dot{q}_\Gamma\,. \qquad [3.19]$$

An important implicit assumption that has been made in the derivation of the energy jump is about the neglect of the internal interface energy. Even if the interface inertia is neglected (as stated at the beginning of this section), this does not imply that the interface energy can also be. But here, we do this hypothesis to simplify the equations.

Remarks on other forms of energy Similarly to the balance equations in the bulk, it is possible to obtain jump conditions on other non-conserved forms of energy by combining the jumps on momentum and total energy. The mechanical energy jump condition can be found by dotting the linear momentum jump conditions by the interface velocity. Subtracting the mechanical jump condition from the total energy jump condition, the jump on the internal energy is deduced, which can be expressed based on the enthalpy using $E = h + e_{kin} - p/\rho$:

$$\dot{m}_\Gamma[\![h + e_{kin} - \mathbf{v}\cdot\mathbf{v}_\Gamma]\!] = [\![\underline{\boldsymbol{\tau}}\cdot(\mathbf{v}-\mathbf{v}_\Gamma)]\!]\cdot\mathbf{n}_\Gamma - [\![\mathbf{q}]\!]\cdot\mathbf{n}_\Gamma + \dot{q}_\Gamma\,. \qquad [3.20]$$

In many practical situations, the viscous dissipation and the kinetic energy are negligible compared to the heat flux and enthalpy, so that the jump on the enthalpy takes the widespread expression

$$\dot{m}_\Gamma [\![h]\!] = -[\![\mathbf{q}]\!] \cdot \mathbf{n}_\Gamma + \dot{q}_\Gamma \,.$$

Entropy

The entropy inequality in the bulk, Equation 2.19, translates the second thermodynamic principle. Likewise, there exists an entropy inequality at the interface, under the form of a jump condition. We set $\varphi_i^v = \rho\,\eta$, and the entropy flux as $\mathfrak{F}^v = -\mathbf{q}/T$. There is also an interface entropy production, noted $\mathfrak{P}^s = \dot{\eta}_\Gamma$, which is only zero in the case of reversible processes. Thus, the jump on entropy is

$$\dot{\eta}_\Gamma = -\dot{m}_\Gamma [\![\eta]\!] - [\![\frac{\mathbf{q}}{T}]\!] \cdot \mathbf{n}_\Gamma \geq 0 \,. \tag{3.21}$$

In order to obtain a jump condition from which it will be possible to extract some constitutive relations at the interface (see Section 3.3), we will combine the jump conditions on enthalpy and entropy. A similar approach is followed in Delhaye (1974). We start by dividing Equation 3.20 by the interface temperature, T_Γ, and we transform the heat flux term like

$$\frac{[\![\mathbf{q}]\!] \cdot \mathbf{n}_\Gamma}{T_\Gamma} = [\![\mathbf{q} \cdot \mathbf{n}_\Gamma \left(\frac{1}{T_\Gamma} - \frac{1}{T} \right)]\!] + [\![\frac{\mathbf{q}}{T}]\!] \cdot \mathbf{n}_\Gamma \,. \tag{3.22}$$

Expressing the last term of Equation 3.22 based on Equation 3.21 and inserting the result into Equation 3.20, by replacing the enthalpy h by $g + T\eta$, with the Gibbs free energy g, we obtain

$$\begin{aligned} \dot{\eta}_\Gamma = {} & [\![(\mathbf{q} \cdot \mathbf{n}_\Gamma + \dot{m}_\Gamma T \eta) \left(\frac{1}{T_\Gamma} - \frac{1}{T} \right)]\!] \\ & + \frac{\dot{m}_\Gamma}{T_\Gamma} [\![g + e_{kin} - \mathbf{v} \cdot \mathbf{v}_\Gamma]\!] \\ & - \frac{1}{T_\Gamma} [\![(\underline{\boldsymbol{\tau}} \cdot \mathbf{n}_\Gamma) \cdot (\mathbf{v} - \mathbf{v}_\Gamma)]\!] \,. \end{aligned} \tag{3.23}$$

We can reformulate two terms of the above equation in the following way:

$$\begin{aligned} (\underline{\boldsymbol{\tau}} \cdot \mathbf{n}_\Gamma) \cdot (\mathbf{v} - \mathbf{v}_\Gamma) &= (\underline{\boldsymbol{\tau}} \cdot \mathbf{n}_\Gamma) \cdot (\mathbf{v}_t - \mathbf{v}_{t,\Gamma}) + \frac{1}{\rho} (\underline{\boldsymbol{\tau}} \cdot \mathbf{n}_\Gamma \cdot \mathbf{n}_\Gamma) \dot{m}_\Gamma \,, \\ [\![e_{kin} - \mathbf{v} \cdot \mathbf{v}_\Gamma]\!] &= [\![\frac{1}{2} \mathbf{v} \cdot \mathbf{v} + \frac{1}{2} \mathbf{v}_\Gamma \cdot \mathbf{v}_\Gamma - \mathbf{v} \cdot \mathbf{v}_\Gamma]\!] = [\![(\mathbf{v} - \mathbf{v}_\Gamma)^2]\!] \,, \end{aligned} \tag{3.24}$$

where $\mathbf{v}_t$ and $\mathbf{v}_{t,\Gamma}$ are the tangential components of the bulk and interface velocities, respectively. Inserting Equation 3.24 into Equation 3.23, we retrieve the same expressions as the ones in Ishii and Hibiki (2011, Sec. 1.2) and Faghri

and Yuwen Zhang (2006, Sec. 3.3.6), i.e.

$$\begin{aligned}\dot{\eta}_\Gamma = &\underbrace{[\![\left(\mathbf{q}\cdot\mathbf{n}_\Gamma + \dot{m}_\Gamma T\eta\right)\left(\frac{1}{T_\Gamma}-\frac{1}{T}\right)]\!]}_{(a)} \\ &- \frac{1}{T_\Gamma}\underbrace{[\![\left(\underline{\boldsymbol{\tau}}\cdot\mathbf{n}_\Gamma\right)\cdot\left(\mathbf{v}_t-\mathbf{v}_{t,\Gamma}\right)]\!]}_{(b)} - \underbrace{\frac{\dot{q}_\Gamma}{T_\Gamma}}_{(c)} \\ &+ \frac{\dot{m}_\Gamma}{T_\Gamma}\underbrace{[\![g+\frac{1}{2}(\mathbf{v}-\mathbf{v}_\Gamma)^2-\frac{1}{\rho}\left(\underline{\boldsymbol{\tau}}\cdot\mathbf{n}_\Gamma\cdot\mathbf{n}_\Gamma\right)]\!]}_{(d)} \geq 0\,.\end{aligned} \tag{3.25}$$

Finally, we can indicate that the generic jumps, Equation 3.15, incorporate the well-known Rankine-Hugoniot shock relations by taking the interface velocity as the shock speed, setting the viscosity and thermal conductivity to zero and disregarding the interface line flux and source. However, the nature of a material and shock wave discontinuity is different. Shock waves dynamically emanates from the non-linearities present in the hyperbolic terms of the equations while phase interfaces are initially present or arises due to phase transition. Moreover, in contrast to shock waves, phase boundaries usually occur as subsonic waves. Hence, additional conditions need to be imposed to ensure a unique solution, as noted by Magiera and Rohde (2022). This is explained in the next section.

3.3 Constitutive equations at the interface

3.3.1 Entropy production and mass transfer

The jump conditions presented above are not closed. The normal momentum jump provides a condition on the pressure jump but there is no such condition on the temperature. Moreover, if the net mass flux across the interface is different from zero, then the interface normal velocity is not equal to the fluid normal velocities and becomes an additional unknown. Finally, no information is embedded into the jumps about the jump on the tangential velocities. Therefore, some constitutive interface models are required to define the problem uniquely. Similarly to the constitutive equations in the bulk (see Section 2.2.2), the interface constitutive laws should satisfy some thermodynamic constraints. They may be obtained from some irreversible thermodynamic arguments, as done by Standart (1968). However, Ishii and Hibiki (2011, Sec. 1.2.2) note that the level of details of this kind of advanced analysis is generally not needed in practice and leads to conditions which are too complex to be applied. A simpler theory turns out to be necessary. The logical starting point is to consider the entropy inequality at the interface, Equation 3.25. According to Dreyer (2005, Sec. 15), the interfacial entropy inequality can be brought into a sum of binary

products:

$$T_\Gamma \dot{\eta}_\Gamma = \sum_n F_n D_n \,,$$

where the quantities F_n and D_n are called the driving forces and fluxes, respectively. For example, in the last term (d) of Equation 3.25, we can identify the mass flux $\dot{m}_\Gamma$ as the interfacial flux, which is driven by the combination of the jumps on the Gibbs free energy, the kinetic energy and the normal viscous stress tensor. To provide the three missing constitutive relations from the entropy jump, there are two possibilities:

1. *Equilibrium process*: the first one is to consider the limiting case of equilibrium process at the interface, which is equivalent to assuming a zero interfacial entropy production or that all the driving forces governing the flux across the interface are zero. Applying those considerations in Equation 3.25 yields three constitutive relations:

$$(a) : [\![T]\!] = 0 \Rightarrow T_1 = T_2 = T_\Gamma \,, \qquad [3.26]$$

$$(b) : [\![\mathbf{v}_t]\!] = 0 \Rightarrow \mathbf{v}_{t,1} = \mathbf{v}_{t,2} = \mathbf{v}_{t,\Gamma} \,, \qquad [3.27]$$

$$(d) : [\![g]\!] = [\![-\frac{1}{2}(\mathbf{v} - \mathbf{v}_\Gamma)^2 + \frac{1}{\rho}\,(\underline{\boldsymbol{\tau}} \cdot \mathbf{n}_\Gamma \cdot \mathbf{n}_\Gamma)]\!] \,. \qquad [3.28]$$

   The two first conditions specify the continuity of temperatures and tangential velocities across the interface. The last condition means that the difference in chemical potential[4] compensates for the mechanical effects induced by the differences of kinetic energy and normal stresses. Let us remark that the condition on tangential velocities only applies for viscous flows. Concerning the last relation, it is only needed if there is a net mass flux across the interface. In this situation, the interface is considered as ideal, which means that there is no resistance to interfacial transfer of quantities. Hence, the exchanges between two phases are governed by the conditions of the bulk fluid at each side, but not by the interface itself (Ishii and Hibiki (2011)). With such assumptions, the interface mass flux is commonly computed based on the jump of heat flux or the species concentration gradients at the interface. The two approaches are compared in Irfan and Muradoglu (2017) and combined in Boniou et al. (2022b).
2. *Non-equilibrium process*: the second less restrictive possibility is to allow for some non-equilibrium process, hence some production of entropy at the interface. Different approaches have been presented in the literature. Two main trends can be identified:
   - In the first family of approaches, all the models start from the entropy

[4] The chemical potential, μ_s, of a species s in a system is the energy that can be absorbed or released due to a change of the number of constituent particles, expressed in number of moles N_s. For a pure substance, the chemical potential is related to the Gibbs free energy via the relation

$$\mu = \left(\frac{\partial g}{\partial N}\right)\Big|_{T,p} \,.$$

jump condition at the interface (by neglecting more or less terms depending on the flow conditions). A few examples are given here. In Stefan Fechter et al. (2015), where an inviscid model is used, the entropy jump is reformulated as

$$[\![g + \frac{1}{2}(\mathbf{v} - \mathbf{v}_\Gamma)^2]\!] + [\![T\eta]\!] - T_\Gamma[\![\eta]\!] = -k^* \dot{m}_\Gamma \,,$$

where k^* is an entropy production coefficient that can be approximated using molecular simulations like Density Functional Theory (DFT). Assuming a linear relation between the fluxes and driving forces in Equation 3.25, for an inviscid model but with the presence of heat flux, Jöns and Claus-Dieter Munz (2022) derive expressions for the mass flux and the heat flux via the Onsager theory. Finally, Juric and Tryggvason (1998) exploit the entropy principle to derive an expression for the departure of the interface temperature from the saturation temperature.

- In the second class of approaches, the entropy condition is not explicitly enforced. Instead, an expression for the interface mass flux, which should models the interfacial resistance to mass transfer, is directly provided. Models of this kind are usually rooted in kinetic theory. The justification for this switch between classical continuum and kinetic models is to be found in the physical nature of phase change process like evaporation. Gerasimov and Yurin (2018, Sec. 1.4) explain that most of the determining processes defining the evaporation phenomenon as a whole take place in the vicinity of the phase interface, at scales which are approximately equal to several mean free paths of the molecules. This thin region in the gas phase next to the interface is called the Knudsen layer. In this layer, Dias (2020, Sec. 3.4) points out that due to few collisions between the particles emerging from the liquid surface and the ones from the edge of the Knudsen layer, the velocity distribution function is bimodal and deviates from a Maxwellian distribution. This makes the Navier-Stokes not valid anymore and one has to resort to the Boltzmann equations. Based on a Chapman-Enskog expansion of the Boltzmann equations, M. Bond and Struchtrup (2004) derive a generalized expression for the net mass flux across the interface, from which well-known models like the Hertz-Knudsen or Schrage models can be extracted. The general expression for the net interfacial mass flux of the sth-species can be written as

$$\dot{m}_{\Gamma,s} = \frac{1}{\sqrt{2\pi}} \left(\Theta_e(T_l) \frac{p_{\text{sat},s}(T_l)}{\sqrt{R_s T_l}} - \Theta_c(T_l, T_g) \frac{p_s}{\sqrt{R_s T_g}} \right) , \qquad [3.29]$$

where Θ_e and Θ_c are evaporation and condensation coefficients

depending on the temperatures of the gas and liquid phases, T_g and T_l, $p_{\text{sat},s}$ is the saturation pressure of species s and p_s the vapor pressure of species s in the gas mixture, and R_s is the gas constant. This model has been implemented in our solver. Equation 3.29 provides an estimation for the mass flux but not for the potential temperature discontinuities across the interface. In order to have a whole description of the phase transition effects, the approach of Ytrehus and Østmo (1996) solving the Boltzmann equations by the moment method can be followed. This has been done in Paula et al. (2019). The approach has been extended to multicomponent flows by Dias (2020, Sec. 3.4.2). In accordance with the kinetic-based modeling of mass transfer effects, the jumps on temperature and tangential velocities can also be estimated based on some other extensions of the Boltzmann equations like the Burnett equations, as conducted in Agarwal et al. (2001). This has been applied in Houim and Kuo (2013).

Jumps on temperature and tangential velocity It remains to clarify which jump conditions the temperatures and tangential velocities have to satisfy. If an equilibrium mass transfer model is considered, we showed from the entropy jump that the continuity of temperatures (Equation 3.26) and tangential velocities (Equation 3.27) across the interface should be assumed. If a non-equilibrium phase transition model is used instead, we should distinguish between inviscid and viscous models.

- Tangential velocity:
  - Inviscid model: in this case, term (b) in Equation 3.25 is automatically zero and there is no explicit condition to be prescribed on the tangential velocities. If there is mass transfer and Marangoni effects, a discontinuity in the tangential velocities might even develop. To rebuild the interface velocity components from the normal and tangential fluid velocities components, a choice has therefore to be made. In Stefan Fechter et al. (2018), the left or right tangential velocity is assigned to the tangential interface velocity depending on their magnitude.
  - Viscous model: in the viscous case, it is in general physically relevant to impose the continuity of tangential velocities across the interface. A partial-slip condition to extend the validity of the Navier-Stokes model is also possible, like in Houim and Kuo (2013).
- Temperature: if a complete model like the one Ytrehus and Østmo (1996) and Dias (2020) based on the Boltzmann equations is considered, the temperature discontinuities at the interface should be obtained from it. On the other hand, if only a single equation to model the mass flux is used (which is the most common approach), two situations are possible:

- Inviscid model: if the diffusive fluxes are not taken into account in the balance at the interface, there is in theory no need to impose anything on the temperatures. We should mention that although the heat flux is neglected in the bulk, there exists approaches still incorporating a heat flux in the jump conditions. This is for example the case in Stefan Fechter et al. (2018) where the interface temperature needs therefore to be specified and is simply taken as the average between the temperatures of the two phases.
- Viscous model: for viscous models, a condition on the temperatures is required. This can be done by imposing a jump, equal to zero like in Chang et al. (2013) or different from zero based on some non-continuum considerations like in Houim and Kuo (2013). A similar approach would be to consider, in addition to a mass flux model, an interfacial heat flux model from which the temperature jump can be obtained. This is explained in G. Chen (2022). The last possibility is to include an equation to estimate the interface temperature, like in Juric and Tryggvason (1998).

3.3.2 Surface tension

The last term which remains to be closed is the interface stress tensor $\mathfrak{F}^s$. Focusing on the momentum surface flux contribution, $\mathfrak{F}^s_{\text{mom}}$, the surface stress tensor can be decomposed into an isotropic and deviatoric part such that $\mathfrak{F}^s_{\text{mom}} = \mathfrak{F}^{s,i}_{\text{mom}} + \mathfrak{F}^{s,d}_{\text{mom}}$. Slattery et al. (2007, Sec. 4.9.4) note that in the case of a single-component system with clean interface, the dominant interfacial momentum flux is due to the isotropic part since we should not expect any surface viscous effects (which would necessitate the use of Boussinesq-Scriven type surface models, like in A. Reusken et al. (2015)), i.e.

$$\mathfrak{F}^s_{\text{mom}} \approx \mathfrak{F}^{s,i}_{\text{mom}} = \sigma_\Gamma \underbrace{(\underline{\boldsymbol{I}} - \mathbf{n}_\Gamma \mathbf{n}_\Gamma^T)}_{\underline{\boldsymbol{P}}_\Gamma}, \tag{3.30}$$

where $\underline{\boldsymbol{P}}_\Gamma$ is the surface projection operator, consistent with the requirements of a tangential surface stress tensor imposed by the angular momentum jump, Equation 3.18 and σ_Γ has to be the thermodynamic surface tension according to the interface entropy inequality, as shown by Slattery et al. (2007, Sec. 4.9.5). Applying the surface divergence operator to this momentum contribution, the interface curvature κ_Γ appears:

$$\nabla_{\Gamma,t} \cdot \mathfrak{F}^s_{\text{mom}} = -\sigma_\Gamma \kappa_\Gamma \mathbf{n}_\Gamma + \nabla_{\Gamma,t} \sigma_\Gamma, \tag{3.31}$$

where the first term gives the normal component force due to the effect of the curved interface whereas the second term represents the tangential force due to surface tension gradient, also known as Marangoni effects.

3.4 Summary of the equations implemented in this work

In this work, we consider the general jump conditions on the total mass, Equation 3.16, on the linear momentum, Equation 3.17, and on the total energy, Equation 3.19, derived in this chapter. The entropy inequality at the interface is not explicitly enforced and when mass transfer is present, we use a kinetic-based model of the form of Equation 3.29 instead of a model based on the entropy production. Thus, in the following, we will write

$$\dot{m}_\Gamma := \rho_k(v_{n,k} - v_{n,\Gamma}) = \dot{m}_\Gamma^{\text{model}}(\mathbf{u}_1, \mathbf{u}_2)\,, \tag{3.32}$$

to indicate that in case of phase transition, a model should be provided, under the form of an analytical equation for the interfacial mass flux and which is potentially dependent on the solutions in both phases. Finally, for the conditions on the temperature and tangential velocities, we impose them to be continuous at the interface in practice. However, we included the possibility to impose constants jumps, $\Delta_\Gamma T$ on the temperatures, and $\Delta_\Gamma v_t$ on the tangential velocities, i.e.

$$[\![v_t]\!] = \Delta_\Gamma v_t\,, \tag{3.33}$$
$$[\![T]\!] = \Delta_\Gamma T\,. \tag{3.34}$$

Combining Equation 3.33 with the mass balance, Equation 3.16, relations on velocity jumps across the interface, both on the normal-tangential and Cartesian components, can be written as:

$$\begin{aligned} [\![v_n]\!] &= \dot{m}_\Gamma\,[\![1/\rho]\!] \\ [\![v_t]\!] &= \Delta_\Gamma v_t \end{aligned} \iff \begin{aligned} [\![v_x]\!] &= \dot{m}_\Gamma\,[\![1/\rho]\!]\,n_{x,\Gamma} + \Delta_\Gamma v_t\,t_{x,\Gamma} \\ [\![v_y]\!] &= \dot{m}_\Gamma\,[\![1/\rho]\!]\,n_{y,\Gamma} + \Delta_\Gamma v_t\,t_{y,\Gamma}\,. \end{aligned} \tag{3.35}$$

Gathering the jumps and the supplemental closure conditions, the jumps equations implemented in the solver are listed below. To get more insights about those relations, we develop them in the two-dimensional setting we are considering. Moreover, to simplify the notations and obtain a generic model, we replace the normal and tangential contributions of the surface stress tensor such that

$$\Delta_\Gamma p\,\mathbf{n}_\Gamma = \sigma_\Gamma\,\kappa_\Gamma\,\mathbf{n}_\Gamma\,, \tag{3.36}$$
$$\Delta_\Gamma \tau_{nt}\,\mathbf{t}_\Gamma = -\nabla_{\Gamma,t}\sigma_\Gamma\,. \tag{3.37}$$

An overview of the various parameters introduced in our formulation of the jump conditions is given in Table 3.2.

Table 3.2 / Overview and physical meaning of the different parameters used in the jump conditions.

Parameters	Physical meaning
$\dot{m}_\Gamma$	net mass flux across interface, non-zero when evaporation or condensation occurs at the phase interface
$\Delta_\Gamma p$	$= \sigma\kappa$, normal surface stress due to surface tension effects with surface tension coefficient σ and interface curvature κ
$\Delta_\Gamma \tau_{nt}$	$= \nabla\sigma \cdot \mathbf{t}_\Gamma$, tangential surface stress due to surface tension gradients known as Marangoni effects
$\Delta_\Gamma q$	surface energy flux, due for example to surface radiation or heterogeneous chemical reactions
$\Delta_\Gamma T$	temperature jump at the interface due to molecular effects, e.g. the presence of the Knudsen layer during evaporation
$\Delta_\Gamma v_t$	partial-slip at the interface, e.g. due to non-continuum flow regimes effects

Jumps in Cartesian frame In the Cartesian reference frame, the implemented jumps read:

$$[\![\dot{m}_\Gamma]\!] = 0 \,, \tag{3.38a}$$

$$\dot{m}_\Gamma\,[\![v_x]\!] + [\![p]\!]\,n_{x,\Gamma} - \Delta_\Gamma p\,n_{x,\Gamma} - \Delta_\Gamma\tau_{nt}\,t_{x,\Gamma} = [\![\tau_{xx}]\!]\,n_{x,\Gamma} + [\![\tau_{xy}]\!]\,n_{y,\Gamma}\,, \tag{3.38b}$$

$$\dot{m}_\Gamma\,[\![v_y]\!] + [\![p]\!]\,n_{y,\Gamma} - \Delta_\Gamma p\,n_{y,\Gamma} - \Delta_\Gamma\tau_{nt}\,t_{y,\Gamma} = [\![\tau_{yx}]\!]\,n_{x,\Gamma} + [\![\tau_{yy}]\!]\,n_{y,\Gamma}\,, \tag{3.38c}$$

$$\begin{aligned}\dot{m}_\Gamma\,[\![E]\!] + [\![p\,v_n]\!] - \Delta_\Gamma p\,v_{n,\Gamma} - \Delta_\Gamma\tau_{nt}\,v_{t,\Gamma} - \dot{q}_\Gamma = {}& [\![\tau_{xx}\,v_x]\!]\,n_{x,\Gamma} + [\![\tau_{yx}\,v_y]\!]\,n_{x,\Gamma} \\ & + [\![\tau_{xy}\,v_x]\!]\,n_{y,\Gamma} + [\![\tau_{yy}\,v_y]\!]\,n_{y,\Gamma} \\ & - [\![q_x]\!]\,n_{x,\Gamma} - [\![q_y]\!]\,n_{y,\Gamma}\,,\end{aligned} \tag{3.38d}$$

supplemented by

$$[\![v_x]\!] = \dot{m}_\Gamma\,[\![1/\rho]\!]\,n_{x,\Gamma} + \Delta_\Gamma v_t\,t_{x,\Gamma}\,, \tag{3.39a}$$

$$[\![v_y]\!] = \dot{m}_\Gamma\,[\![1/\rho]\!]\,n_{y,\Gamma} + \Delta_\Gamma v_t\,t_{y,\Gamma}\,, \tag{3.39b}$$

$$[\![T]\!] = \Delta_\Gamma T\,, \tag{3.39c}$$

$$\dot{m}_\Gamma = \dot{m}_\Gamma^{\text{model}}(\mathbf{u}_1, \mathbf{u}_2)\,. \tag{3.39d}$$

The Cartesian jumps will be useful for the enforcement of the viscous fluxes at the interface in Section 9.2.3.

Jumps in interface normal frame It is also convenient to project the jump conditions along the normal and tangential directions to the interface:

$$\begin{aligned}
&&&&[\![\dot{m}_\Gamma]\!] &= 0 &, \quad [3.40a]\\
\dot{m}_\Gamma\,[\![v_n]\!] + &[\![p]\!] - \Delta_\Gamma p &&&= [\![\tau_{nn}]\!] &, \quad [3.40b]\\
\dot{m}_\Gamma\,[\![v_t]\!] &&- \Delta_\Gamma \tau_{nt} &&= [\![\tau_{nt}]\!] &, \quad [3.40c]\\
\dot{m}_\Gamma\,[\![E]\!] + &[\![p\,v_n]\!] - \Delta_\Gamma p\,v_{n,\Gamma} &- \Delta_\Gamma \tau_{nt}\,v_{t,\Gamma} - \Delta_\Gamma q &= [\![\tau_{nn}\,v_n]\!] + [\![\tau_{nt}\,v_t]\!] - [\![q_n]\!] &, \quad [3.40d]
\end{aligned}$$

supplemented by

$$\begin{aligned}
[\![v_n]\!] &= \dot{m}_\Gamma\,[\![1/\rho]\!]\,, && [3.41a]\\
[\![v_t]\!] &= \Delta_\Gamma v_t\,, && [3.41b]\\
[\![T]\!] &= \Delta_\Gamma T\,, && [3.41c]\\
\dot{m}_\Gamma &= \dot{m}_\Gamma^{\text{model}}(\mathbf{u}_1, \mathbf{u}_2)\,. && [3.41d]
\end{aligned}$$

The jumps in the interface attached frame will be useful for the enforcement of the convective fluxes at the interface in Section 9.2.2.

Contribution 3.1: Models at the sharp gas–condensed phase interface

The main contributions that can be identified from this chapter are:

- The detailed derivation of the generic jump conditions and their specification to the case of two-phase compressible viscous flows, including the statement of the required hypotheses.
- The elaborated presentation of the constitutive models to close the set of jump conditions.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- The complete jump conditions with the possibility of easily specifying new constitutive equations to close the various terms.

- A new internal library providing several mass transfer models derived from kinetic theory.

The modularity of the implementation offers the possibility to model a wide range of phenomena and to couple phases governed by distinct constitutive laws.

4

Modeling of the diffuse liquid–solid phase interface

The models in Section 4.3 were presented in Henneaux et al. (2023c).

4.1 An overview of liquid-solid phase change modeling 110
4.1.1 Melting and glass transition processes 110
4.1.2 Range of models and associated numerical methods . . . 113
4.1.3 The choice of a macroscale diffuse interface model 115
4.2 The volume-averaged two-fluid equations 115
4.2.1 Volume averaging and derivation of the general model . 117
4.2.2 Closure of the equations and hierarchy of diffuse interface models 121
4.3 The equilibrium homogeneous mixture model for liquid-solid system with phase transformation 126
4.3.1 The mixture equations and the assumptions 126
4.3.2 Mass equation 131
4.3.3 Momentum equation 133
4.3.4 Energy equation 138
4.4 Summary of the equations implemented in this work 140

4.1 An overview of liquid–solid phase change modeling

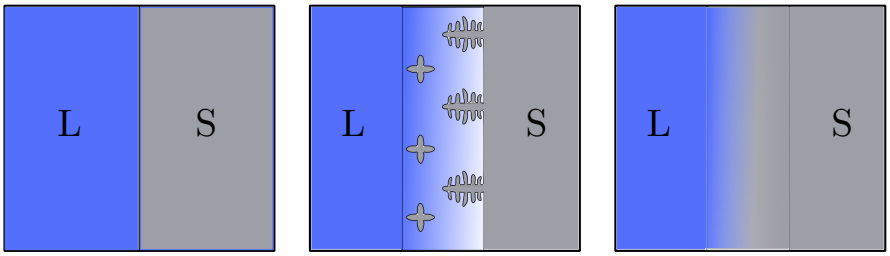
4.1.1 Melting and glass transition processes

In this thesis, the term "melting" is in fact employed not only to designate melting phase transition process but also its reverse, solidification, and another distinct phase transition process which is the glass transition.

Crystalline and amorphous materials

The molecules in a solid phase are compactly arranged in a closely-spaced, fixed pattern, and are bonded together by intermolecular attractive forces. The way the constituents of a solid are arranged depends on its nature: they can form a regular repeating three-dimensional structure called a crystal lattice, thus producing a *crystalline solid*, or they can aggregate with no particular order, in which case they form an *amorphous solid*. Most metals and metallic alloys are crystalline while the glassy materials found in some parts of a spacecraft are amorphous. When heat is applied to a solid, the temperature of the solid rises as the vibrational energy of the molecules also increases. When this thermal energy reaches a level that overcomes the intermolecular attractive force, molecules no longer remain in a fixed pattern. In the case of a crystalline solid, this threshold level is reached uniformly for all the molecules because of their regular organization. Relative motion among them becomes then possible at the same time and the solid melts[1]. The situation is quite different for an amorphous solid, such as polymers, wax or glass. Due to the random organization of their molecules, different amounts of thermal energy are needed to overcome the different interactions. Consequently, amorphous solids tend to soften slowly over a wide temperatures range rather than having a well-defined melting point like a crystalline solid, a process called the glass transition. These differences in phase transition process have an impact on the morphology of the interface between the solid and the liquid phases. Their phase change shares some common physical aspects but presents also some significant differences which are explained below. A deeper physical description can be found in Fredriksson and Åkerlind (2012).

[1]It must be pointed out here that atoms in metals have the peculiarity of being hold together through metallic bonds, which are of a different nature to covalent or ionic bonds. On melting, metallic bonds are loosened, not broken. So, although the ordered solid structure has been broken down, metallic bonds are still present between the atoms inside a molten metal. Metallic bonds are not fully broken until the metal boils. Metallic bonds are discussed in more in details in Appendix B.



a / Pure substances. **b /** Alloys. **c /** Glassy materials.

Figure 4.1 / Classification of solid-liquid phase change based on the interfacial morphology.

Melting

Interfacial morphology The thickness of the interface region where the solid-liquid phase transition occurs may vary from few Angstroms to few centimeters, and its microstructure may be very complex, depending on several factors, such as the nature of the material, the rate of heating/cooling, the temperature and concentrations gradients, surface tension, etc., as explained by Alexiades and A. D. Solomon (1993, Sec. 1.1). The morphology of the solid-liquid interface during solidification has received considerable attention in the literature while information on the interfacial properties during melting is quite sparse in comparison. However, it has been established that atom movements at the interface during solidification are essentially reversed during melting, such that the interface exhibits similar features in both situations (Woodruff (1968)). Based on the nature of the interface, the solid-liquid phase change process can be classified into three categories according to Bennon and F. Incropera (1987), which are sketched in Figure 4.1. The different interfacial morphologies are compared here from a macroscopic point of view, which corresponds to the scale at which melting shall be investigated in this work. For most *pure materials* changing phase at a fixed temperature, the interface appears locally planar and of negligible thickness. Thus, it may be seen as a sharp front separating the solid and liquid bulk regions. This will typically be the case when a pure metal melts, for instance.

For multicomponent materials like alloys undergoing phase transition with *separable phases*, since the melting point differs between substances, phase transition occurs over a range of temperatures instead of at a single melting point. When the temperature of the multicomponent material is within this range, the substance is a mixture of solid and liquid, called the *mushy zone*[2]. From a macroscopic point of view, the interfacial region has a finite thickness and the

[2]To be more accurate, a single mushy zone is present only in isomorphous alloys for which the different components are completely miscible over the entire compositional range in both liquid and solid states. For eutectic and peritectic alloys for which the solubility between the different constituents is limited, we can identify several mushy regions (Nikrityuk (2011, Sec. 5.1.2)). This distinction goes however out of the modeling scope of this work.

properties of the two bulk regions transition smoothly over this zone (and not in a sharp manner like for pure substances). From a microscopic perspective, the morphology of the mushy region can be described as a combination between (i) dendrites[3], which can exist in either columnar or equiaxed forms and which are constrained to any translation, and (ii) solid grains floating in the liquid and totally free to move. Those characteristics are at the basis of different models discussed in the following.

Latent heat The first fundamental concept in phase change process to mention is the latent heat. While a crystalline solid has not reached its melting point, its temperature can be increased by adding some amount of heat, named the sensible heat. After the temperature of a solid reaches its melting point, continued heating does not result in temperature increase since the heat provided, called the latent heat, is entirely used to overcome the intermolecular attractive forces. This jump in the enthalpy of the system has to be taken into account in the derivation of the mathematical models, as it will be detailed in this chapter. Once the solid has melted, the temperature begins to rise again, but now at a different rate, due to the modification of the heat capacity associated to phase transition.

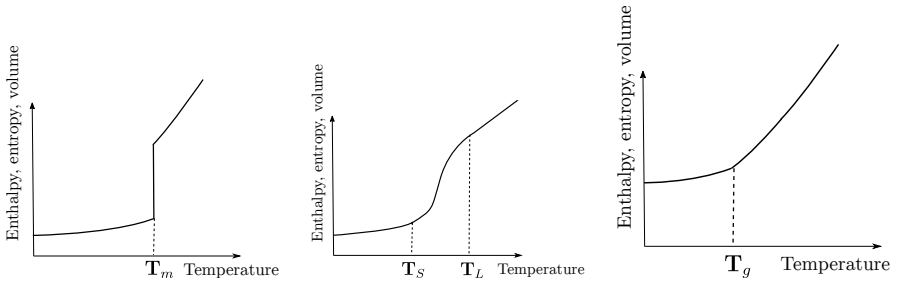
Phase-change temperature As already mentioned previously, pure substance melts at a well-defined temperature which is a function of pressure. For multicomponent crystalline materials, the phase transition occurs over a range of temperature that depends on the concentration of each component in the system. For a binary system, this temperature range is delimited by the liquidus temperature of the interface between the mushy zone and the liquid phase and by the solidus temperature of the interface between the solid and the mushy zone. The differences between pure substance, alloys and glassy materials are shown in Figure 4.2.

Changes in thermophysical properties Crystalline materials undergo first-order phase transition, meaning that thermodynamic quantities obtained from first derivatives of the free energy are discontinuous between the phases (entropy, density, enthalpy for example). *A fortiori*, quantities obtained from second derivatives of the free energy are discontinuous as well (heat capacity, isothermal compressibility, isobaric expansivity for example).

Glass transition

Interfacial morphology The last type of phase transformation process to be mentioned concerns phase transition between *non-separable phases* in amorphous materials. In this case, it is not possible to make a clear distinction between the solid and liquid phases in the interfacial region. The slow softening of the solid

[3]Dendrites are treelike solid structures coming from the anisotropic behavior of solid crystals which have a preferred growth direction depending on the basic lattice structure of the metal.



a / Pure substances. **b /** Alloys. **c /** Glassy materials.

Figure 4.2 / Classification of solid-liquid phase transition based on the change of properties.

is accompanied by a smooth decrease in the viscosity and there is no distinct interface between the two phases.

Latent heat For amorphous materials, there is no jump in enthalpy during the glass transition process. Providing heat to the system always results in a temperature increase. Like for crystalline materials, the rate at which temperature rises changes after phase transformation due to a change in the heat capacity.

Phase-change temperature It is not possible to identify a precise phase-change temperature range for glassy materials. Instead, a glass transition temperature T_g is often defined, corresponding to the temperature at which thermodynamic properties like the heat capacity show a step change (see Figure 4.2c).

Changes in thermophysical properties Amorphous materials show second-order phase transition, which leads to sudden changes in heat capacity, isothermal compressibility or isobaric expansivity but not in entropy, density or enthalpy, as explained in G. Zhang and Zhou (2017).

4.1.2 Range of models and associated numerical methods

The solid-liquid phase transition modeling approaches can be divided following the same global categories as for multiphase flows discussed in Section 3.1.1, i.e. phase-field, local instant and averaging formulations. The two former ones are typically used when the evolution of the complex microstructures in the interface region during phase transition needs to be accurately predicted (e.g. to be able to predict the influence on the mechanical properties). The latter approach deals with a macroscopic representation of the interface.

Microscale models

Local instant formulation In the local instant formulation, the jump conditions derived in Section 3.2 are imposed at the liquid-solid interface which is considered as a sharp discontinuity. To be more precise, the jumps are simplified compared the ones considered at the gas-liquid interface since the velocity field in the solid phase is set to zero. It is by following this approach that the well-known Stefan problem can be formulated. An application of this approach with the level-set as interface capturing method can be found for instance in Theillard et al. (2015). A review is provided in Jaafar et al. (2017).

Phase field formulation In the phase field approach, the mathematical difficulty associated to the satisfaction of jump conditions on interfaces whose location is part of the unknown solution is avoided. Those methods employ an additional state variable, the phase-field variable, which is a function of position and time, to describe whether the material is liquid or solid. The behavior of this variable is governed by an additional equation that is coupled to the equations of balance for mass, momentum and energy. The interface is described by a steep, but continuous, transition of this variable between the two states. An overview of this method in the context of melting and solidification can be found e.g. in Boettinger et al. (2002).

Macroscale models

The starting point of all macroscale models, sometimes called one-domain methods, are the volume-averaged equations, which are discussed in details in the next section. The idea behind this class of models is to consider the single-phase conservation equations for each phase and apply a volume averaging operator to obtain a set of equations valid all over the domain. The interface region becomes diffuse and there is no need to explicit track its position. There are two main classes of single-domain approaches: multi-fluid and mixture (see e.g. P. J. Prescott and F. P. Incropera (1996, Sec. 3.B)).

Multi-fluid approach In the multi-fluid/two-fluid approach, a set of averaged equations is considered per phase. The influence of microscopic phenomena on the macroscopic behavior is incorporated through closure terms modeling the interfacial exchange between the phases. An overview can be consulted in Beckermann and Viskanta (1993).

Mixture / One-fluid approach The mixture formulation, also called the continuum model, is obtained by summing the separate averaged equations of each phase to obtain a single set of equations. Interfacial exchange terms cancel out and the impact of the microscopic quantities is taken into account via effective mixture variables. A detailed derivation of such models can be found in Jun Ni and F. P. Incropera (1995b). It is interesting to note that simplifying assumptions

are usually made with this kind of models such that the resulting equations are easier to handle. This will be the case in the present work.

4.1.3 The choice of a macroscale diffuse interface model

This introductory section concludes with a presentation of the reasons that led us to choose a macroscale diffuse interface model rather than one of the others introduced above.

- *Physical diffuse nature of the interface*: from a macroscopic point of view, the liquid-solid interface, contrary to the one separating the gas and the condensed phases, can be seen as a diffuse mixture region over which the properties smoothly transition from one phase to the other. It would only make sense to consider a local instant formulation if melting of pure materials were to be studied, which will rarely be the case for space debris ablation, or if a microscopic description of the interfacial phenomena was required, which is *a priori* not the case.
- *Applicability of the model*: similarly to the use of volume-averaged equations for highly dispersed two-phase situations, the choice a macroscale model is the only one leading to equations in form which are amenable for computation, considering the intricate nature of the interface in the mushy region.

4.2 The volume-averaged two-fluid equations

In the last chapter, the advantages, challenges and limitations of the local instant formulation of two-phase flows have been highlighted. In order to overcome the main limitation of the interface-resolved approach, namely its application to realistic multiphase flows including dispersed regions, another strategy is sought. This alternative strategy starts from the observation that a macroscopic description of the flow, modeling some interfacial phenomena instead of explicitly resolving them, might be sufficient as long as it is still predictive for the targeted applications. The idea is at the heart of the numerous multi-fluid models that have been developed in the literature. Cordesse (2020, Sec. 1.1) identifies three approaches to derive the general equations behind all those models:

- *Averaging from extended bulk equations*: the averaging approach can be broken down in three steps: (i) the bulk equations in each homogeneous subregions of the flows are considered separately, (ii) those equations are extended to the whole domain by multiplying the physical quantities

with an indicator function, (iii) an averaging operator is finally applied to the resulting equations to obtain a macroscopic description of the flow, both in the interface and in the homogeneous flow regions. The averaging introduces some additional unclosed terms that need to be modeled. This modeling determines the specific final model. This is the approach adopted for example in Drew (1983) and Ishii and Hibiki (2011).

- *Closure of averaged equations from second principle*: the second approach works the other way around. It first postulates averaged conservation equations for each phase with the presence of unknown exchange terms. The second principle of thermodynamics is then invoked to close the exchange terms. This is the approach followed by Landau and Lifshitz (1987) or Baer and Nunziato (1986).
- *Variational principle*: this last derivation approach relies on the Hamilton's principle of stationary action. It states that any system will follow the physical path that minimizes his action. Battista (2021, Sec. 3.3) explains that the first step is to set up the assumptions that govern the phenomena to be represented, in particular the type and form of the kinetic and potential energies. Those energies are then cast into a Lagrangian whose integral over space and time constitutes the "Hamiltonian action" of the system. The physical path of the system from one state to another, among the infinite possible ones, is the one that minimizes the action. This leads to a system of equations to which a dissipative part must be added in a second step exploiting the second principle of thermodynamics.

The development of the general two-phase flow models that is carried out below follows the first approach. This is also the approach from which the equations of the liquid-solid system will be derived in Section 4.3.

Ishii and Hibiki (2011) interpret the averaging procedure as a low-pass filtering which excludes the unwanted high frequency signals contained in the local instant formulation. By doing so, we can tackle the difficulties mentioned in Section 3.1.2. Indeed, the tracking of a large amount of deformable interfaces is made possible thanks to the implicit identification of the interfaces and we can account, via closure models, for the fluctuating quantities coming from turbulent effects or small scales interfacial phenomena. Concerning the capturing of important discontinuities across the interfaces, those are smeared out by the averaging process. From a numerical point of view however, this last aspect can be double-edged because a smooth representation of an important discontinuity might lead to some numerical instabilities and inaccuracies. This will be further discussed in Section 8.1 when the sharp and diffusive interface methods will be put into perspectives. There exist various forms of possible averaging to be applied to the equations (see e.g. Morel (2015, Sec. 3.2)): time averaging, spatial averaging, ensemble averaging. A. Prosperetti (2007, Sec. 8.1) notes that the superiority of one model versus the others is still an open debate. Here we will consider spatial volume averaging because it is the averaging used in the majority of solid-liquid models and because we can extract from it a strong physical interpretation.

4.2.1 Volume averaging and derivation of the general model

The steps toward the derivation of a general two-fluid model presented below are based on Morel (2015, Secs. 2.6 & 3.3), Cordesse (2020, Sec. 1.1.1) and Le Touze (2016, Sec. 3.3.1)

Homogenization of the single-phase equations

We start by introducing the support phase function associated to phase k:

$$X_k(\mathbf{x},t) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is in phase } k \text{ at time } t\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{4.1}$$

The phase function is advected at the interfacial speed such that:

$$\frac{\partial X_k}{\partial t} + \mathbf{v}_\Gamma \cdot \nabla X_k = 0\,, \tag{4.2}$$

with the gradient ∇X_k to be interpreted in the distributional sense since X_k is not continuous, i.e.

$$\nabla X_k = \mathbf{n}_{\Gamma,k}\delta_{\Gamma,k}\,, \tag{4.3}$$

with $\delta_{\Gamma,k}$, the Dirac distribution of the interface associated to phase k. The first step of the derivation consists in extending the validity of the single-phase bulk equations to the entire domain occupied by the two phases by multiplying each system by its corresponding phase function X_k. By introducing X_k into the differential operators and using Equation 4.2 and Equation 4.3 to express the additional terms that appear, we get

$$\begin{aligned}
\frac{\partial X_k \rho_k}{\partial t} + \nabla\cdot(X_k\rho_k\mathbf{v}_k) &= 0 \\
&\quad - [\rho_k(\mathbf{v}_k - \mathbf{v}_\Gamma)]\cdot\mathbf{n}_{\Gamma,k}\delta_{\Gamma,k}\,, \\
\frac{\partial X_k \rho_k \mathbf{v}_k}{\partial t} + \nabla\cdot(X_k\rho_k\mathbf{v}_k\otimes\mathbf{v}_k) &= \nabla\cdot[X_k(-p_k\underline{\underline{\boldsymbol{I}}} + \underline{\underline{\boldsymbol{\tau}}}_k)] + X_k\rho_k\mathbf{b}_k \\
&\quad - [\rho_k\mathbf{v}_k\otimes(\mathbf{v}_k-\mathbf{v}_\Gamma) + p_k\underline{\underline{\boldsymbol{I}}} - \underline{\underline{\boldsymbol{\tau}}}_k]\cdot\mathbf{n}_{\Gamma,k}\delta_{\Gamma,k}\,, \\
\frac{\partial X_k \rho_k E_k}{\partial t} + \nabla\cdot(X_k\rho_k H_k\mathbf{v}_k) &= \nabla\cdot[X_k(\underline{\underline{\boldsymbol{\tau}}}_k\cdot\mathbf{v}_k - \mathbf{q}_k)] + X_k\rho_k\mathbf{b}_k\cdot\mathbf{v}_k \\
&\quad - [\rho_k E_k(\mathbf{v}_k-\mathbf{v}_\Gamma)]\cdot\mathbf{n}_{\Gamma,k}\delta_{\Gamma,k} \\
&\quad + [(p_k\underline{\underline{\boldsymbol{I}}} - \underline{\underline{\boldsymbol{\tau}}}_k)\cdot\mathbf{v}_k + \mathbf{q}_k]\cdot\mathbf{n}_{\Gamma,k}\delta_{\Gamma,k}\,.
\end{aligned} \tag{4.4}$$

Those equations are valid over the entire domain and not restricted anymore to their respective domain. It is instructive to remark that by summing the equations over both phases and by introducing mixture quantities, we can obtain a single-fluid formulation of the problem with right-hand side source terms localized at the interface and which incorporate the jump conditions

presented in the local instant formulation. At this stage, we still have a model with the same limitations as the local instant formulation since we would need to know all the interfacial details to be able to define the function X_k and use the derived equations. In order to filter out under-resolved flow features like turbulence or the presence of small droplets, we average the equations.

Volume averaging of the homogeneous model

The second step consists in averaging the equations.

Averaging operator and definitions We use the volume averaging operator $\langle\phi\rangle$ defined by

$$\langle\phi\rangle = \frac{1}{V}\int_V \phi \, \mathrm{d}\mathbf{x}\,.$$

Yeoh and Tu (2010, Sec. 2.2) make the remark that the underlying assumption for a meaningful result is that the volume V over which the averaging is performed is larger than the characteristic length scale of individual phase elements (e.g. particle spacing or bubble diameter). The averaging operator satisfy the following properties:

- Linearity
$$\langle a_1\phi + a_2\rangle = a_1\langle\phi\rangle + \langle a_2\rangle\,.$$
- Idempotency
$$\langle\langle\phi\rangle\rangle = \langle\phi\rangle\,.$$
- Gauss and Leibniz rules
$$\frac{\partial\langle\phi\rangle}{\partial t} = \langle\frac{\partial\phi}{\partial t}\rangle\,, \quad \frac{\partial\langle\phi\rangle}{\partial x} = \langle\frac{\partial\phi}{\partial x}\rangle\,.$$

To decouple the mean flow quantities of interest from the fluctuating ones, we introduce the fluctuations ϕ',

$$\phi = \langle\phi\rangle + \phi' \quad \text{with } \langle\phi'\rangle = 0\,.$$

Because we are using a spatial volume averaging operator here, we have that

$$\alpha_k := \langle X_k\rangle \tag{4.5}$$

is the volume fraction of phase k. We also introduce three averages:

- Phase average (volume weighted):
$$\overline{\phi_k} = \frac{\langle X_k\,\phi_k\rangle}{\alpha_k}\,.$$

- Favre average (mass weighted):

$$\widetilde{\phi_k} = \frac{\langle X_k \rho_k \phi_k \rangle}{\alpha_k \overline{\rho_k}} .$$

- Interfacial phase average:

$$\overline{\phi_{k,\Gamma}} = \frac{\langle \|\nabla X_k\| \phi_k \rangle}{\|\nabla \alpha_k\|} .$$

The fluctuations from the Favre average will be noted by ϕ'',

$$\phi = \widetilde{\phi} + \phi'' .$$

Averaged equations with unclosed terms Applying the operator $\langle \bullet \rangle$ to Equation 4.4, using its properties listed above and introducing the phase and Favre averages, we obtain the general averaged equations:

$$\frac{\partial \alpha_k \overline{\rho_k}}{\partial t} + \nabla \cdot (\alpha_k \overline{\rho_k} \widetilde{\mathbf{v}_k}) = S_{\text{mass},k} , \tag{4.6a}$$

$$\frac{\partial \alpha_k \overline{\rho_k} \widetilde{\mathbf{v}_k}}{\partial t} + \nabla \cdot (\alpha_k \overline{\rho_k} \widetilde{\mathbf{v}_k} \otimes \widetilde{\mathbf{v}_k} + \alpha_k \overline{p_k} \underline{\boldsymbol{I}}) = \nabla \cdot \left[\alpha_k (\overline{\underline{\boldsymbol{\tau}}_k} + \underline{\boldsymbol{\tau}}_k^*)\right] + \alpha_k \overline{\rho_k} \widetilde{\mathbf{b}_k} + \mathbf{S}_{\text{mom},k} , \tag{4.6b}$$

$$\frac{\partial \alpha_k (\overline{\rho_k} \widehat{E_k} + E_k^*)}{\partial t} + \nabla \cdot \left[\alpha_k (\overline{\rho_k} \widehat{E_k} + E_k^* + \overline{p_k}) \widetilde{\mathbf{v}_k}\right] = \nabla \cdot \left[\alpha_k (\overline{\underline{\boldsymbol{\tau}}_k} \cdot \widetilde{\mathbf{v}_k} - \overline{\mathbf{q}_k} + \mathbf{J}_k^*)\right] + \alpha_k \overline{\rho_k} \widetilde{\mathbf{b}_k} \cdot \widetilde{\mathbf{v}_k} + S_{\text{ener},k} , \tag{4.6c}$$

where the averaged total specific energy is defined according to Haegeman (2021, Sec. 2.2.1) as

$$\overline{\rho_k} \widehat{E_k} = \overline{\rho_k} \widetilde{e_{int,k}} + \frac{1}{2} \overline{\rho_k} \, (\widetilde{\mathbf{v}_k})^2 .$$

We can also add a transport equation for the volume fraction from Equation 4.2, i.e.

$$\frac{\partial \alpha_k}{\partial t} + \mathbf{v}_\Gamma \cdot \nabla \alpha_k = 0 , \tag{4.7}$$

complemented by the constraint that the sum of the volume fractions from both phases is equal to unity,

$$\alpha_1 + \alpha_2 = 1 . \tag{4.8}$$

Fluctuating terms The averaging procedure on the non-linear terms leads to the apparition of fluctuations terms which bear the $\bullet^*$ superscript and are

expressed as:

$$\begin{aligned}\underline{\boldsymbol{\tau}}_k^* &= -\overline{\rho_k \, \mathbf{v}_k'' \otimes \mathbf{v}_k''}\,,\\ E_k^* &= \frac{1}{2}\overline{\rho_k \, (\mathbf{v}_k'')^2}\,,\\ \mathbf{J}_k^* &= -\overline{\rho_k \, H_k'' \, \mathbf{v}_k''} + \overline{\underline{\boldsymbol{\tau}}_k \cdot \mathbf{v}_k''}\,.\end{aligned} \qquad [4.9]$$

Le Touze (2016) remarks that those terms are very similar to the ones that appear when the Navier-Stokes equations are averaged to isolate the turbulent fluctuating terms. However, it is important to note that the terms here represent the variables fluctuations caused by under-resolved moving interfaces and not by the turbulent structures. It would also be possible to make the turbulent fluctuating terms appear but this would require a second averaging procedure which is not carried out here.

Interfacial exchange terms The source terms on the right-hand side of Equation 4.6, noted $S_\bullet$, contain all the additional terms that are created by the averaging process and are expressed as:

$$S_{\text{mass},k} = \langle \rho_k \, (\mathbf{v}_k - \mathbf{v}_{\Gamma,k}) \cdot \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle = \underbrace{\langle \dot{m}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle}_{S^1_{\text{mass},k}}\,, \qquad [4.10a]$$

$$\mathbf{S}_{\text{mom},k} = \underbrace{\langle \dot{m}_{\Gamma,k} \, \mathbf{v}_k \, \delta_{\Gamma,k} \rangle}_{\mathbf{S}^1_{\text{mom},k}} + \underbrace{\langle p_k \, \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle - \langle \underline{\boldsymbol{\tau}}_k \cdot \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle}_{\mathbf{S}^2_{\text{mom},k}}\,, \qquad [4.10b]$$

$$S_{\text{ener},k} = \underbrace{\langle \dot{m}_{\Gamma,k} \, E_k \, \delta_{\Gamma,k} \rangle}_{S^1_{\text{ener},k}} + \underbrace{\langle p_k \, \mathbf{v}_k \cdot \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle - \langle \underline{\boldsymbol{\tau}}_k \cdot \mathbf{v}_k \cdot \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle}_{S^2_{\text{ener},k}} + \underbrace{\langle \mathbf{q}_k \cdot \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle}_{S^3_{\text{ener},k}}\,. \qquad [4.10c]$$

The terms with the superscripts 1, $S^1_{-,k}$, are linked to the interfacial transfer of mass, momentum and energy due to phase change. They represent the advection of an interfacial quantity due to the relative motion of the solid-liquid interface and become zero when the net mass flux $\dot{m}_\Gamma$ is zero. The terms $S^2_{-,k}$ take into account the interfacial transfer of momentum and work due to the stresses exerted by each phase on the interface. The term $S^3_{ener,k}$ is related to the conduction of heat across the interface due to temperature gradients on each side of the interface.

Jump conditions The averaging process can also be applied to the jump conditions. The result is not detailed here but can be found in Morel (2015, Sec. 3.3.1). Exploiting the averaged jump conditions, with the same assumptions as the ones stated in Section 3.2.2, the links between the interfacial exchange terms and the jumps can be established as:

$$\sum_{k=1}^{2} S_{\text{mass},k} = 0\,, \qquad [4.11a]$$

$$\sum_{k=1}^{2} \mathbf{S}_{\mathrm{mom},k} = \mathbf{S}_{\mathrm{mom},\Gamma}\,, \tag{4.11b}$$

$$\sum_{k=1}^{2} S_{\mathrm{ener},k} = S_{\mathrm{ener},\Gamma}\,. \tag{4.11c}$$

Equation 4.11a expresses the conservation of mass at the interface. Equation 4.11b is about the balance between the averaged momentum transfers from the bulk and the mixture momentum source at the interface given by

$$\mathbf{S}_{\mathrm{mom},\Gamma} = \langle \sigma_\Gamma\, \kappa_\Gamma\, \mathbf{n}_\Gamma\, \delta_\Gamma \rangle\,,$$

if only surface tension effects are taken into account in the interface line fluxes and Marangoni effects are disregarded. Finally, Equation 4.11c formulates the balance between the net interfacial supply of energy from both phases and the mixture total energy source at the interface, which is given as

$$S_{\mathrm{ener},\Gamma} = \langle [(\sigma_\Gamma\, \kappa_\Gamma\, \mathbf{n}_\Gamma) \cdot \mathbf{v}_\Gamma + \dot{q}_\Gamma]\, \delta_\Gamma \rangle\,, \tag{4.12}$$

and for which the internal energy associated to the interface has been neglected (as in the local instant formulation).

4.2.2 Closure of the equations and hierarchy of diffuse interface models

Identification of the unknowns and family of models

While averaging allows the mathematical solution of the problem to be tractable, there is an apparent need to recover lost information since the microscopic variables on which they are based are not known from the solution of the macroscopic equations. More precisely, the unknown variables appearing in the general two-fluid model derived so far can be identified as:

- Volume fraction transport: α_k, $\mathbf{v}_\Gamma = 1 \times 2 + 1 = 3$ unknowns.
- Mass balances: $\overline{\rho_k}$, $\widetilde{\mathbf{v}_k}$, $S_{\mathrm{mass},k} = 3 \times 2 = 6$ unknowns.
- Momentum balances: $\overline{p_k}$, $\overline{\boldsymbol{\tau}_k}$, $\boldsymbol{\tau}_k^*$, $\overline{\rho_k\, \mathbf{b}_k}$, $\mathbf{S}_{\mathrm{mom},k}$, $\mathbf{S}_{\mathrm{mom},\Gamma} = 5 \times 2 + 1 = 11$ unknowns.
- Energy balances: $\widetilde{e_k}$, E_k^*, $\overline{\mathbf{q}_k}$, $\mathbf{J}_k^*$, $S_{\mathrm{ener},k}$, $S_{\mathrm{ener},\Gamma} = 5 \times 2 + 1 = 11$ unknowns.

There are 31 unknowns for only 11 equations identified so far, namely the balance equations for each phase, Equation 4.6, the 3 jump conditions, Equation 4.11, one of the volume fraction transport relations, Equation 4.7 and the constraint on the sum of the volume fractions, Equation 4.8. We therefore need 20 additional closure relations. Those relations and the associated assumptions will specify the type of models.

Relaxation processes Many models have been proposed in the literature and the purpose here is not to present a complete overview but simply to classify the models according to their main assumptions. To do so, we introduce the concept of characteristic relaxation time that determine the time for a quantity to reach an equilibrium value common to both phases. Indeed, two-phase flow systems naturally evolve toward mechanical and thermodynamic equilibrium according to four principal relaxation times which can generally be ordered, following Labois (2008, Sec. 1.3), like

$$\tau_p \ll \tau_v \ll \tau_T \ll \tau_\mu \,, \tag{4.13}$$

where τ_p is the relaxation time associated to the pressures in the system, τ_v the one associated to the velocities, τ_T to the temperatures and τ_μ to the chemical potentials. It can be shown that some interfacial exchange terms that need to be modeled are inversely proportional to those relaxation times. The sequence in Equation 4.13 thus yields a classification of two-fluid models which is summarized in Table 4.1. In the field of gas-gas or gas-liquid flows, a lot of different two-fluid models have been developed following this categorization. An overview can be consulted in Saurel and Pantano (2018) and Maltsev et al. (2022).

Distinction between two-fluid and mixture models In the context of liquid-solid two-phase flows with phase change, it was mentioned in Section 4.1.2 that the fidelity of available models ranges from the complete two-fluid model (i.e. 7 equations) to the mixture model. Although the mixture model only contains three equations, it is preferable not to identify it as a "three-equations" models in the sense of the relaxation processes described above. Indeed, as it will be shown in Section 4.3, the mixture model is obtained by summing the complete two-fluid model over the two phases and by defining the appropriate mixture variables based on the phase-averaged quantities. As such, it does not directly rely on the concept of variables relaxing to an equilibrium state (although some equilibrium assumptions will be formulated to simplify the equations).

Closure of some terms

The expressions of some unclosed terms are here detailed as they will be useful in the derivation of the homogeneous mixture model in the next section.

Surface tension and pressure relations Following Y. Sun and Beckermann (2004), the pressure part in $\mathbf{S}^2_{\text{mom},k}$ in Equation 4.10b can be split in two parts, the first one representing the average interfacial pressure and the second one the unbalanced pressures at the interface,

$$\langle p_k \, \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle = \overline{p_{k,\Gamma}} \nabla \alpha_k + \langle (p_k - \overline{p_{k,\Gamma}}) \, \mathbf{n}_{\Gamma,k} \, \delta_{\Gamma,k} \rangle \,, \tag{4.14}$$

Table 4.1 / Hierachy of two-fluid models from relaxation of the general volume-averaged model.

Model name	Quantities at equilibrium	Equations considered
7 equations	-	2 mass 2 momentum 2 energy 1 volume fraction
6 equations	Pressure	2 mass 2 momentum 2 energy
5 equations	Pressure, Velocity	2 mass 1 momentum 2 energy
4 equations	Pressure, Velocity Temperature	2 mass 1 momentum 1 energy
3 equations	Pressure, Velocity Temperature, Chemical potential	1 mass 1 momentum 1 energy

where $\overline{p_{k,\Gamma}}$ uses the interfacial phase average definition. Assuming instantaneous microscopic pressure equilibrium, $\overline{p_{k,\Gamma}}$ can be taken equal to the phase-averaged pressure of phase k,

$$\overline{p_{k,\Gamma}} = \overline{p_k}\,. \tag{4.15}$$

Inserting Equation 4.15 in Equation 4.14 and combining the first term of the right-hand-side of Equation 4.14 with the pressure gradient in the averaged momentum equation, Equation 4.6b becomes

$$\frac{\partial \alpha_k \overline{\rho_k}\,\widetilde{\mathbf{v}_k}}{\partial t} + \nabla\cdot(\alpha_k \overline{\rho_k}\,\widetilde{\mathbf{v}_k}\otimes\widetilde{\mathbf{v}_k}) = -\alpha_k\nabla\overline{p_k} + \nabla\cdot\left[\alpha_k(\overline{\underline{\boldsymbol{\tau}}_k} + \underline{\boldsymbol{\tau}}_k^*)\right] + \alpha_k\overline{\rho_k\,\mathbf{b}_k} + \mathbf{S}^*_{\mathrm{mom},k}\,, \tag{4.16}$$

with the modified interfacial exchange source term, $\mathbf{S}^*_{\mathrm{mom},k}$, given by

$$\mathbf{S}^*_{\mathrm{mom},k} = \mathbf{S}^1_{\mathrm{mom},k} + \underbrace{\langle(p_k - \overline{p_k})\,\mathbf{n}_{\Gamma,k}\,\delta_{\Gamma,k}\rangle - \langle\underline{\boldsymbol{\tau}}_k\cdot\mathbf{n}_{\Gamma,k}\,\delta_{\Gamma,k}\rangle}_{\mathbf{S}^{2,*}_{\mathrm{mom},k}}\,.$$

The jump condition, Equation 4.11b, can then be restated as

$$\begin{aligned}\sum_{k=1}^{2}\mathbf{S}^*_{\mathrm{mom},k} &= \sum_{k=1}^{2}\mathbf{S}_{\mathrm{mom},k} - \sum_{k=1}^{2}\langle\overline{p_k}\,\mathbf{n}_{\Gamma,k}\,\delta_{\Gamma,k}\rangle\,,\\ &= \mathbf{S}_{\mathrm{mom},\Gamma} - \sum_{k=1}^{2}\langle\overline{p_{k,\Gamma}}\,\mathbf{n}_{\Gamma,k}\,\delta_{\Gamma,k}\rangle\,.\end{aligned}$$

The difference between the averaged interfacial pressures, $\overline{p_{k,\Gamma}}$, can be assumed to balance the surface tension force at the interface, $\mathbf{S}_{\mathrm{mom},\Gamma}$, such that

$$\sum_{k=1}^{2}\mathbf{S}^*_{\mathrm{mom},k} = 0\,, \tag{4.17}$$

as it is done in J. Ni and Beckermann (1991).

Macroscopic bulk stresses and heat fluxes Contrary to the interfacial exchange terms, the macroscopic viscous stress and heat flux represent interactions within the phase itself. Constitutive relations for the average of those two terms can be proposed, making the assumptions listed in J. Ni and Beckermann (1991):

- The viscous stresses for both the solid and liquid phases are proportional to the rates of deformation, meaning that the solid is assumed to be a pseudo-fluid (this assumption will be discussed in Section 4.3.3). The contribution due to displacement gradients to the viscous stresses (arising from density differences and phase change) is neglected. The heat flux is given by the Fourier law.
- The fluctuating components of the viscosity and thermal conductivity are neglected.

Based on those assumptions, Y. Sun and Beckermann (2004) derive an expression for the averaged viscous stress associated to phase k:

$$\begin{aligned}\overline{\underline{\boldsymbol{\tau}}_k} = \mu_k^{\text{eff}} \left[\nabla \widetilde{\mathbf{v}_k} + (\nabla \widetilde{\mathbf{v}_k})^T - \frac{2}{3}(\nabla \cdot \widetilde{\mathbf{v}_k}) \underline{\boldsymbol{I}} \right] \\ + \frac{\mu_k^{\text{eff}}}{\alpha_k} \left[\nabla \alpha_k (\widetilde{\mathbf{v}_k} - \widetilde{\mathbf{v}_{k,\Gamma}}) + (\widetilde{\mathbf{v}_k} - \widetilde{\mathbf{v}_{k,\Gamma}}) \nabla \alpha_k - \frac{2}{3} \nabla \alpha_k \cdot (\widetilde{\mathbf{v}_k} - \widetilde{\mathbf{v}_{k,\Gamma}}) \underline{\boldsymbol{I}} \right],\end{aligned} \tag{4.18}$$

with the effective viscosity μ_k^{eff} that depends on the interfacial topology and the averaged interfacial velocity of phase k defined as

$$\widetilde{\mathbf{v}_{k,\Gamma}} = \frac{\langle \|\nabla X_k\| \rho_k \mathbf{v}_k \rangle}{\|\nabla \alpha_k\| \overline{\rho_k}}.$$

Y. Sun and Beckermann (2004) propose a simple viscosity weighted model for the averaged interfacial velocity,

$$\widetilde{\mathbf{v}_{k,\Gamma}} = \frac{\mu_1 \alpha_2 \widetilde{\mathbf{v}_1} + \mu_2 \alpha_1 \widetilde{\mathbf{v}_2}}{\mu_1 \alpha_2 + \mu_2 \alpha_1}.$$

The first part of Equation 4.18 accounts for the viscous stress of phase k to itself as in a single phase system. The second part represents the contribution due to relative motion between the two phases and is non-zero only inside the diffuse interface zone. Concerning the average heat flux, it is typically modeled by an effective thermal conductivity in each phase, λ_k^{eff}, such that

$$\overline{\mathbf{q}_k} = -\lambda_k^{\text{eff}} \nabla \overline{T_k} - \frac{\lambda_k^{\text{eff}}}{\alpha_k} \nabla \alpha_k \left[\overline{T_k} - \overline{T_{k,\Gamma}} \right], \tag{4.19}$$

which is consistent with the model in Ishii and Hibiki (2011, Sec. 9.1.2.4). Similarly to the viscous stress, the first part takes into account the heat flux coming from the phase k itself while the second part, proportional to the difference between the phase-averaged and interfacial temperature, models the heat transfer due to the interaction between the two phases.

Closures for the interfacial transfers due to phase change and for the interfacial stresses and heat transfers are not detailed here since we will not need them in the derivation of the mixture model. The interested reader can consult e.g. Ishii and Hibiki (2011, Sec. 1.2.6) for general closures and J. Ni and Beckermann (1991) for closures related to liquid-solid systems.

4.3 The equilibrium homogeneous mixture model for liquid–solid system with phase transformation

In this work, we made the choice of facility over fidelity, i.e. we adopted the mixture model rather than any two-fluid models, for the following reasons:

- *Closure of the equations*: although two-phase models have the potential to concurrently resolve a lot of microscopic and macroscopic features, there are many uncertainties involved in their closure. Thus, if we do not correctly model the unclosed terms in the equations, we may even lose the increased modeling fidelity theoretically offered by the two-phase models.
- *Computational expense*: the additional equations present in the two-phase models as well as the complex exchange terms inevitably makes their solution procedure more computationally expensive than mixture models.
- *Dedicated algorithms*: the presence of the non-conservative exchange terms in the equations pose some mathematical and numerical issues as they are required to enforce hyperbolicity and because specialized numerical schemes dealing with the relaxation operators need to be designed. In order to use the same algorithm both in the gas and condensed phases, it is therefore preferable to avoid those additional difficulties.
- *Level of description*: finally, we believe that the level of modeling fidelity brought by the mixture model is sufficient for the targeted application where we are not interested in a fine description of the interfacial phenomena between the liquid and solid phases.

4.3.1 The mixture equations and the assumptions

Simplifying assumptions

In order to facilitate the derivation of the mixture model and anticipating the fact that the model we want to use for the liquid-solid system does not need to incorporate all the microscopic information, we formulate the following simplifying assumptions:

- We neglect the fluctuating terms, Equation 4.9, because their modeling is very uncertain:

$$\boldsymbol{\tau}_k^* = E_k^* = \mathbf{J}_k^* = 0\,. \qquad [4.20]$$

  This limits the applicability of the model to creeping flow in the mushy region and laminar flow in the pure liquid region, as pointed out by Beckermann and Viskanta (1993).

- In the averaged total energy source at the interface, Equation 4.12, the work done by the surface tension forces is assumed to be negligible in comparison to the heat generation at the interface such that

$$S_{\text{ener},\Gamma} = \langle \dot{q}_\Gamma \, \delta_\Gamma \rangle \, . \tag{4.21}$$

- The next assumption, very common in liquid-solid system, consists in assuming thermal equilibrium between the phases, not only at the interface but also in the volume, such that the averaged interfacial temperature are equal,

$$\overline{T_1} = \overline{T_2} = \overline{T_{1,\Gamma}} = \overline{T_{2,\Gamma}} = T_v \, ,$$

with the mixture temperature T_v. This assumption is made e.g. in Jun Ni and F. P. Incropera (1995b) or Beckermann and Viskanta (1993) and can be justified by the high thermal diffusivity of the materials considered in this work and the expected laminar behavior of the flow in the liquid-solid interfacial region. This assumption simplifies the expression for the averaged heat flux per phase, Equation 4.19, into

$$\overline{\mathbf{q}_k} = -\lambda_k^{\text{eff}} \nabla T_v \, . \tag{4.22}$$

- The last assumption concerns the expression of the averaged macroscopic viscous stress tensor, Equation 4.18. The second part of the tensor, which takes into account the effects of the interfacial motions and the mass transfers on the average deformation, is neglected. In other words, we assume that in both phases the difference between the interfacial fluid velocity and the mean velocity in the flow is sufficiently small for the second term to be negligible compared to the first one. This leads to

$$\overline{\underline{\boldsymbol{\tau}}_k} = \mu_k^{\text{eff}} \left[\nabla \widetilde{\mathbf{v}_k} + (\nabla \widetilde{\mathbf{v}_k})^T - \frac{2}{3} (\nabla \cdot \widetilde{\mathbf{v}_k}) \underline{\boldsymbol{I}} \right] . \tag{4.23}$$

This last assumption is clearly the most debatable and could easily be relaxed if needed by implementing the second part of the stress tensor in our model. For example, another hypothesis proposed by Beckermann and Viskanta (1993) is to set $\widetilde{\mathbf{v}_{1,\Gamma}} \approx \widetilde{\mathbf{v}_{2,\Gamma}} = 0$, which can be interpreted as neglecting the interfacial momentum transfer due to density changes upon phase change.

Mixture quantities

We start by introducing definitions of mixture quantities that will be useful in what follows:

- The volume fraction of phase k, already introduced before, is still noted α_k. All the mixture quantities obtained from a volume average are identified by the subscript $\bullet_v$.

- The mixture density, ρ_v, is given by

$$\rho_v = \alpha_1 \overline{\rho_1} + \alpha_2 \overline{\rho_2}\,. \tag{4.24}$$

- The mass fraction of phase k is noted f_k and given by

$$f_k = \alpha_k \frac{\overline{\rho_k}}{\rho_v}\,. \tag{4.25}$$

  All the mixture quantities obtained from a mass average are identified by the subscript $\bullet_m$.

It should be noted at this point that the mixture equations will need to be solved for the mixture quantities and not the phase-averaged variables.

Summing over both phases

We then sum the averaged bulk equations, Equation 4.6a, Equation 4.16 and Equation 4.6c, over the two phases. While doing so, we include the jumps relations, Equation 4.11a, Equation 4.17 and Equation 4.11c combined with Equation 4.21, to eliminate the unknown interfacial exchange terms. This yields, taking into account the assumptions listed above:

$$\underbrace{\frac{\partial}{\partial t}\left(\underbrace{\sum_{k=1}^{2} \alpha_k \overline{\rho_k}}_{(a)}\right)}_{} + \underbrace{\sum_{k=1}^{2} \nabla\cdot(\alpha_k \overline{\rho_k}\,\widetilde{\mathbf{v}_k})}_{(b)} = 0\,, \tag{4.26a}$$

$$\underbrace{\frac{\partial}{\partial t}\left(\underbrace{\sum_{k=1}^{2} \alpha_k \overline{\rho_k}\,\widetilde{\mathbf{v}_k}}_{(a)}\right)}_{} + \underbrace{\sum_{k=1}^{2} \nabla\cdot(\alpha_k \overline{\rho_k}\,\widetilde{\mathbf{v}_k}\otimes\widetilde{\mathbf{v}_k})}_{(b)} + \underbrace{\sum_{k=1}^{2} \alpha_k \nabla\overline{p_k}}_{(c)} = \underbrace{\sum_{k=1}^{2} \nabla\cdot(\alpha_k \overline{\underline{\boldsymbol{\tau}}_k})}_{(d)} + \underbrace{\sum_{k=1}^{2} \alpha_k \overline{\rho_k}\,\widetilde{\mathbf{b}_k}}_{(e)}\,, \tag{4.26b}$$

$$\underbrace{\frac{\partial}{\partial t}\left(\underbrace{\sum_{k=1}^{2} \alpha_k \overline{\rho_k}\,\widetilde{E_k}}_{(a)}\right)}_{} + \underbrace{\sum_{k=1}^{2} \nabla\cdot\left(\alpha_k \overline{\rho_k}\,\widetilde{H_k}\,\widetilde{\mathbf{v}_k}\right)}_{(b)} = \underbrace{\sum_{k=1}^{2} \nabla\cdot(\alpha_k \overline{\underline{\boldsymbol{\tau}}_k}\cdot\widetilde{\mathbf{v}_k})}_{(c)} - \underbrace{\sum_{k=1}^{2} \nabla\cdot(\alpha_k \overline{\mathbf{q}_k})}_{(d)} + \underbrace{\sum_{k=1}^{2} \alpha_k \overline{\rho_k}\,\widetilde{\mathbf{b}_k}\cdot\widetilde{\mathbf{v}_k}}_{(e)}$$

$$+ \underbrace{S_{\text{ener},\Gamma}}_{(\tilde{f})} , \qquad [4.26c]$$

with the averaged total enthalpy defined as

$$\widehat{H_k} = \widehat{E_k} + \frac{\overline{p_k}}{\overline{\rho_k}} .$$

The next step is to develop each term appearing in the equations above.

Mass equation For the mass equation, we have:

- Using the definition of the mixture density, Equation 4.24,

$$\textit{Equation 4.26a} - (a) = \rho_v .$$

- Introducing the mixture velocity $\mathbf{v}_m$ as

$$\mathbf{v}_m = f_1 \widetilde{\mathbf{v}_1} + f_2 \widetilde{\mathbf{v}_2} , \qquad [4.27]$$

and using the definition of the mass fraction, Equation 4.25, we get

$$\textit{Equation 4.26a} - (b) = \nabla \cdot (\rho_v \mathbf{v}_m) . \qquad [4.28]$$

Momentum equation Concerning the momentum equation, the following developments can be made:

- Similarly to Equation 4.28, we have

$$\textit{Equation 4.26b} - (a) = \rho_v \mathbf{v}_m .$$

- For the second term, we follow the decomposition in P. Prescott et al. (1991) and first write

$$\sum_{k=1}^{2} \alpha_k \overline{\rho_k} \widetilde{\mathbf{v}_k} = \rho_v \mathbf{v}_m \otimes \mathbf{v}_m + \sum_{k=1}^{2} \alpha_k \overline{\rho_k} (\widetilde{\mathbf{v}_k} - \mathbf{v}_m) \otimes (\widetilde{\mathbf{v}_k} - \mathbf{v}_m) . \qquad [4.29]$$

Then, from Equation 4.27, we can write

$$\widetilde{\mathbf{v}_1} = \frac{\mathbf{v}_m}{f_1} - \frac{f_2}{f_1} \widetilde{\mathbf{v}_2} . \qquad [4.30]$$

Inserting this relation in the previous one to re-express the second term, we arrive at

$$\textit{Equation 4.26b} - (b) = \nabla \cdot \left[\rho_v \mathbf{v}_m \otimes \mathbf{v}_m + \rho_v \frac{f_2}{f_1} (\mathbf{v}_m - \widetilde{\mathbf{v}_2}) \otimes (\mathbf{v}_m - \widetilde{\mathbf{v}_2}) \right] .$$

- The third term is simply written as

$$\textit{Equation 4.26b} - (c) = \alpha_1 \nabla \widetilde{p_1} + \alpha_2 \nabla \widetilde{p_2} \,.$$

- For the sum of the stress tensors, we only re-express here the stress tensor associated to phase $k = 1$ as a function of the mixture velocity and the phase-averaged velocity in phase $k = 2$, in anticipation of the manipulations that will be done in the following. To do so, we re-use Equation 4.30 and writes

$$\nabla \widetilde{\mathbf{v}_1} = \frac{1}{f_1} \nabla \mathbf{v}_m + \mathbf{v}_m \nabla \left(\frac{1}{f_1} \right) - \frac{f_2}{f_1} \nabla \widetilde{\mathbf{v}_2} - \widetilde{\mathbf{v}_2} \nabla \left(\frac{f_2}{f_1} \right) ,$$

which, using Equation 4.23, leads to

$$\begin{aligned} \alpha_1 \overline{\underline{\boldsymbol{\tau}}_1} = \alpha_1 \mu_1^{\text{eff}} \Bigg\{ & \frac{1}{f_1} \left[\nabla \mathbf{v}_m + (\nabla \mathbf{v}_m)^T - \frac{2}{3} (\nabla \cdot \mathbf{v}_m) \underline{\boldsymbol{I}} \right] \\ & + \mathbf{v}_m \nabla \left(\frac{1}{f_1} \right) + \nabla \left(\frac{1}{f_1} \right) \mathbf{v}_m - \frac{2}{3} \left[\mathbf{v}_m \cdot \nabla \left(\frac{1}{f_1} \right) \right] \underline{\boldsymbol{I}} \\ & - \frac{f_2}{f_1} \left[\nabla \widetilde{\mathbf{v}_2} + (\nabla \widetilde{\mathbf{v}_2})^T - \frac{2}{3} (\nabla \cdot \widetilde{\mathbf{v}_2}) \underline{\boldsymbol{I}} \right] \\ & - \widetilde{\mathbf{v}_2} \nabla \left(\frac{f_2}{f_1} \right) + \nabla \left(\frac{f_2}{f_1} \right) \widetilde{\mathbf{v}_2} - \frac{2}{3} \left[\widetilde{\mathbf{v}_2} \cdot \nabla \left(\frac{f_2}{f_1} \right) \right] \underline{\boldsymbol{I}} \Bigg\} \,. \end{aligned} \qquad [4.31]$$

- The last term is simply developed as

$$\textit{Equation 4.26b} - (e) = \alpha_1 \overline{\rho_1} \widetilde{\mathbf{b}_1} + \alpha_2 \overline{\rho_2} \widetilde{\mathbf{b}_2} = \rho_v \mathbf{b}_m \,.$$

Energy equation Regarding the energy equation, we propose the following manipulations:

- The first term becomes

$$\textit{Equation 4.26c} - (a) = \rho_v E_m \,,$$

with the mass averaged mixture total energy defined by

$$E_m = \underbrace{f_1 \widetilde{e_{int,1}} + f_2 \widetilde{e_{int,2}}}_{e_{int,m}} + \underbrace{1/2 \left(f_1 \widetilde{\mathbf{v}_1} \cdot \widetilde{\mathbf{v}_1} + f_2 \widetilde{\mathbf{v}_2} \cdot \widetilde{\mathbf{v}_2} \right)}_{e_{kin,m}} ,$$

where $e_{int,m}$ is the mixture internal energy and $e_{kin,m}$ the mixture kinetic energy. However, we should be careful that this latter cannot be entirely expressed based on the mixture velocity since

$$e_{kin,m} = \frac{1}{2} \left[\mathbf{v}_m \cdot (\widetilde{\mathbf{v}_1} + \widetilde{\mathbf{v}_2}) - \widetilde{\mathbf{v}_1} \cdot \widetilde{\mathbf{v}_2} \right] . \qquad [4.32]$$

- Using the same decomposition as in Equation 4.29, the second term writes

$$\textit{Equation 4.26c} - (b) = \nabla \cdot \left[\rho_v H_m \mathbf{v}_m + \sum_{k=1}^{2} \alpha_k \overline{\rho_k} \left(\widehat{H_k} - H_m \right) \left(\widetilde{\mathbf{v}_k} - \mathbf{v}_m \right) \right] ,$$

  where the mixture total enthalpy is defined as

$$H_m = E_m + \frac{1}{\rho_v} \underbrace{\left(\alpha_1 \overline{p_1} + \alpha_2 \overline{p_2} \right)}_{p_v} ,$$

  with the mixture pressure p_v.
- From Equation 4.22, the sum on the heat fluxes is simply given by

$$\textit{Equation 4.26c} - (d) = -\nabla \cdot \left[\left(\alpha_1 \lambda_1^{\text{eff}} + \alpha_2 \lambda_2^{\text{eff}} \right) \nabla T_v \right] .$$

- The assumption on the interfacial energy source, Equation 4.21, is such that

$$\textit{Equation 4.26c} - (e) = \langle \dot{q}_\Gamma \, \delta_\Gamma \rangle ,$$

  and will be modeled in Section 4.3.4.

4.3.2 Mass equation

From the developments presented above, the mixture mass equation is readily obtained as

$$\frac{\partial \rho_v}{\partial t} + \nabla \cdot \left(\rho_v \mathbf{v}_m \right) = 0 . \qquad [4.33]$$

Remarks about multispecies models The apparition of the mushy region around the liquid-solid interface necessarily involves the presence of at least two distinct species in the system. Since the local thermophysical properties, including the liquidus and solidus temperatures, will depend on the local concentration of species, models including transport equations for each species present in the system have been developed. There are still two phases in the domain (liquid and solid) but the presence of several species mixed at the molecular level is allowed. The mixture equations are supplemented by a transport equation for each species. Many of the models available in the literature consider a binary alloy (i.e. with two species). A phase diagram is required to provide the relations between the chemical composition, the temperature and the physical state of the alloy (pure liquid, pure solid or both at the same time). For a binary system, the phase diagram is often presented as a temperature-composition diagram under isobaric conditions. As an illustrative example, let us consider two species, A and B, with liquid and solid mole fractions noted respectively by $\chi_{A,l}$, $\chi_{A,s}$, $\chi_{B,l}$, $\chi_{B,s}$. A sketch of a typical phase diagram for a

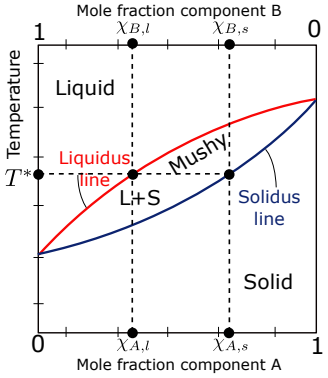


Figure 4.3 / Phase diagram of a binary isomorphous alloy.

binary isomorphous[4] alloy is shown in Figure 4.3. Nikrityuk (2011, Sec. 5.1.2) explains that a stable two-component liquid phase exists above the liquidus line while a two-component stable solid phase lies below the solidus line. In the region between two lines, called the mushy region, the system is composed of both liquid and solid phases of both components. At a given temperature T^*, the composition of the liquid and solid phases are different, leading to a phase composition gradients in the mushy region and the definition of a segregation coefficient,

$$k_A = \left.\frac{\chi_{A,s}}{\chi_{A,l}}\right|_{T,p} .$$

Closure for the volume fraction Before presenting the momentum and energy equations, it is important at this point to realize that since we are not considering any additional transport equation for the volume or mass fraction of one phase, we need a closure model. Indeed, all the mixture quantities are obtained from the volume or mass fraction. Typically, a model for the liquid volume fraction, α_l, is provided. For a binary system, this model usually depends both on the temperature and the segregation coefficient mentioned above. An overview of such models can be found in Basu and Date (1988, Sec. 2.7.b). In our case where the transport of species is not considered, a model for α_l solely dependent on temperature is sought. We choose the smooth differentiable profile proposed in Cagnone et al. (2014), viz.

$$\alpha_l(T) = \frac{1}{2}\left[\tanh\left(2\pi\,\frac{T - T_{\text{melt}}}{T_{\text{liq}} - T_{\text{sol}}}\right) + 1\right], \qquad [4.34]$$

with T_{melt}, the melting temperature, T_{liq}, the liquidus temperature and T_{sol}, the solidus temperature. Below the solidus temperature, the mixture is fully solid,

[4] Because of the limited solubility between the components in eutectic and peritectic alloys, the shape of their phase diagrams is different.

above the liquidus temperature, the mixture is a pure liquid, and in between the two, the volume fractions of both phases are different from zero, indicating the presence of the two phases in the mixture. The choice of the liquidus and solidus temperatures therefore determines the extent of the mushy region over which the fluid properties smoothly transition from one phase to the other. Contrary to the local instant formulation of Section 3.2.2 for which the interface position needs to be explicitly tracked by an additional scalar quantity, in this diffuse interface approach, the interfacial region is identified based on a fluid quantity, namely the local temperature in this case.

4.3.3 Momentum equation

The specification of the momentum equation requires considering three distinct physical situations involving specific modeling assumptions. In what follows, the phase $\bullet_{k=1}$ will be the liquid phase $\bullet_l$ while the phase $\bullet_{k=2}$ will be the solid phase $\bullet_s$.

Phase transition with non-separable phases or with floating dispersed solid microstructures

The first case concerns two types of liquid-solid phase transformation and interactions:

- The case of floating solid microstructures fully dispersed in the liquid.
- The case of amorphous materials exhibiting a glassy phase transition, i.e. for which the distinction between liquid and solid phases is not clearly defined.

For the two situations, the assumptions of mechanical equilibrium and equality of velocities, as done in Voller et al. (1989), seem reasonable:

$$\overline{\mathbf{v}_l} = \overline{\mathbf{v}_s} = \mathbf{v}_m \,, \tag{4.35a}$$

$$\widetilde{p_l} = \widetilde{p_s} = p_v \,. \tag{4.35b}$$

Those two assumptions greatly simplify the different terms in the mixture momentum equation which becomes:

$$\begin{aligned} &\frac{\partial(\rho_v \mathbf{v}_m)}{\partial t} + \nabla \cdot [\rho_v \mathbf{v}_m \otimes \mathbf{v}_m] + \nabla p_v \\ &= \nabla \cdot \left[(\alpha_l \mu_l^{\text{eff}} + \alpha_s \mu_s^{\text{eff}}) \left[\nabla \mathbf{v}_m + (\nabla \mathbf{v}_m)^T - \frac{2}{3} (\nabla \cdot \mathbf{v}_m) \underline{\boldsymbol{I}} \right] \right] + \rho_v \mathbf{b}_m \,. \end{aligned} \tag{4.36}$$

A requirement for the proposed model is that the mixture velocity progressively approaches zero as the solid liquid fraction tends to unity. Because the interfacial exchange terms have been removed by the summation over the two phases, this velocity *penalization* must come from either the viscosity or the source term.

A source term under the form of a drag coefficient may be appropriate to model the interactions between the floating solid microstructure and the liquid flow. This has been done in Jun Ni and F. P. Incropera (1995b). During solidification of amorphous materials, to represent the slow softening of the solid without a clear phase change, a model acting on the effective mixture viscosity seems more physically relevant. This could be simply achieved by specifying a very large value for the effective solid viscosity. More sophisticated models based on rheological considerations have also been proposed. An overview of some of them can be found in Nikrityuk (2011, Sec. 5.2.2). In all the cases, the idea is to write

$$\alpha_l \, \mu_l^{\text{eff}} + \alpha_s \, \mu_s^{\text{eff}} = \mu_v(\alpha_l, \mu_l^{\text{eff}}, \mu_s^{\text{eff}}) \,, \tag{4.37}$$

where μ_v is the mixture viscosity whose expression depends on the selected model.

Phase transition with floating solid structures

The difference with the first case is that here the averaged size of the free-floating solid particles is such that the assumption on the equality of velocities between the two phases cannot be made. Mechanical equilibrium is however still assumed, as done in Jun Ni and F. P. Incropera (1995b):

$$\widetilde{p_l} = \widetilde{p_s} = p_v \,. \tag{4.38}$$

The mixture momentum equation becomes:

$$\begin{aligned}
&\frac{\partial(\rho_v \, \mathbf{v}_m)}{\partial t} + \nabla \cdot \left[\rho_v \, \mathbf{v}_m \otimes \mathbf{v}_m + \rho_v \frac{f_s}{f_l} \, (\mathbf{v}_m - \widetilde{\mathbf{v}_s}) \otimes (\mathbf{v}_m - \widetilde{\mathbf{v}_s}) \right] + \nabla p_v \\
&= \nabla \cdot \Big[\alpha_l \, \mu_l^{\text{eff}} \left\{ \frac{1}{f_l} \left[\nabla \mathbf{v}_m + (\nabla \mathbf{v}_m)^T - \frac{2}{3} (\nabla \cdot \mathbf{v}_m) \underline{\boldsymbol{I}} \right] \right. \\
&\qquad + \mathbf{v}_m \nabla \left(\frac{1}{f_l} \right) + \nabla \left(\frac{1}{f_l} \right) \mathbf{v}_m - \frac{2}{3} \left[\mathbf{v}_m \cdot \nabla \left(\frac{1}{f_l} \right) \right] \underline{\boldsymbol{I}} \\
&\qquad - \frac{f_s}{f_l} \left[\nabla \widetilde{\mathbf{v}_s} + (\nabla \widetilde{\mathbf{v}_s})^T - \frac{2}{3} (\nabla \cdot \widetilde{\mathbf{v}_s}) \underline{\boldsymbol{I}} \right] \\
&\qquad \left. - \widetilde{\mathbf{v}_s} \nabla \left(\frac{f_s}{f_l} \right) + \nabla \left(\frac{f_s}{f_l} \right) \widetilde{\mathbf{v}_s} - \frac{2}{3} \left[\widetilde{\mathbf{v}_s} \cdot \nabla \left(\frac{f_s}{f_l} \right) \right] \underline{\boldsymbol{I}} \right\} \Big] \\
&+ \nabla \cdot \left[\alpha_s \, \mu_s^{\text{eff}} \left\{ \left[\nabla \widetilde{\mathbf{v}_s} + (\nabla \widetilde{\mathbf{v}_s})^T - \frac{2}{3} (\nabla \cdot \widetilde{\mathbf{v}_s}) \underline{\boldsymbol{I}} \right] \right\} \right] \\
&+ \rho_v \, \mathbf{b}_m \,.
\end{aligned} \tag{4.39}$$

The problem with Equation 4.39 is that it contains the averaged solid velocity $\widetilde{\mathbf{v}_s}$ which is an additional unknown to the problem. To overcome this difficulty, there are three possible solutions:

- An extra relationship is added to the system of equations to account for the movement of the dispersed solid crystals in the melt. Such model is

presented in Jun Ni and F. P. Incropera (1995a).

- The shortcomings of the homogeneous mixture model can be avoided by noticing that in this kind of situation, a two-way coupling between the two phases is probably more appropriate. Bennon and F. Incropera (1987) indeed point out that in such circumstances, explicit consideration must be given to phase interactions and description of relative phase motions, which motivates the use of a two-fluid model instead of a more simplified mixture model.
- The last possibility is to formulate additional simplifying assumptions such that the unknown averaged solid velocity disappears from the equation.

  – The first assumption is to neglect the second term in the convective term, i.e.

$$\rho_v \frac{f_s}{f_l} (\mathbf{v}_m - \widetilde{\mathbf{v}_s}) \otimes (\mathbf{v}_m - \widetilde{\mathbf{v}_s}) \approx 0 . \qquad [4.40]$$

    This term is non-zero only within the mushy region and it is a measure of the inertial forces established s a consequence of variations in relative phase velocities. P. Prescott et al. (1991) and Bennon and F. Incropera (1987) note that its influence compared to the other terms can be neglected.

  – P. Prescott et al. (1991) also suggests to neglect, in the liquid viscous stress tensor, the terms involving the gradients of the mass fractions, in comparison to the ones with the velocity gradients:

$$\nabla\left(\frac{1}{f_l}\right) = \nabla\left(\frac{f_s}{f_l}\right) \approx 0 . \qquad [4.41]$$

  – Finally, Jun Ni and F. P. Incropera (1995b) remark that for a dispersed flow with a low solid volume fraction, the effective solid viscosity can be assumed to be approximately equal to the effective liquid viscosity, $\mu_l^{\text{eff}} \approx \mu_s^{\text{eff}}$. If in addition, the densities of the liquid and solid phases are equivalent, $\overline{\rho_l} \approx \overline{\rho_s}$, we have that the two following terms cancel each other:

$$\begin{aligned} &- \alpha_l \mu_l^{\text{eff}} \left\{ \frac{f_s}{f_l} \left[\nabla\widetilde{\mathbf{v}_s} + (\nabla\widetilde{\mathbf{v}_s})^T - \frac{2}{3}(\nabla \cdot \widetilde{\mathbf{v}_s})\underline{\boldsymbol{I}} \right] \right\} \\ &+ \alpha_s \mu_s^{\text{eff}} \left\{ \left[\nabla\widetilde{\mathbf{v}_s} + (\nabla\widetilde{\mathbf{v}_s})^T - \frac{2}{3}(\nabla \cdot \widetilde{\mathbf{v}_s})\underline{\boldsymbol{I}} \right] \right\} \approx 0 . \end{aligned}$$

With those assumptions, the simplified momentum equation is now free of unknown variables:

$$\begin{aligned} &\frac{\partial(\rho_v \mathbf{v}_m)}{\partial t} + \nabla \cdot [\rho_v \mathbf{v}_m \otimes \mathbf{v}_m] + \nabla p_v \\ &= \nabla \cdot \left[\frac{\alpha_l}{f_l} \mu_l^{\text{eff}} \left[\nabla\mathbf{v}_m + (\nabla\mathbf{v}_m)^T - \frac{2}{3}(\nabla \cdot \mathbf{v}_m)\underline{\boldsymbol{I}} \right] \right] + \rho_v \mathbf{b}_m . \end{aligned} \qquad [4.42]$$

Phase transition with constrained solid motion

As explained by Jun Ni and F. P. Incropera (1995b), for stationary solids such as columnar dendrites attached to the solid body or packed equiaxed crystals, the solid phase cannot be anymore treated as a pseudo-fluid and should rather be considered as a permeable/porous matrix across which the molten liquid can flow. Since the solid is stationary and assumed non-deformable, the solid momentum equation does not apply anymore and the liquid equation governs the momentum balance for the mixture. One approach is therefore to directly start from the volume-averaged momentum liquid equation (and not from the mixture one) and to express the interfacial momentum exchange terms with a drag coefficient (to model dispersed solid flow) or with a permeability tensor (to model stationary solid). This is one of the approach developed in Jun Ni and F. P. Incropera (1995b). The second (almost) equivalent approach proposed by the same authors and that can also be found in P. Prescott et al. (1991) is to start from the momentum mixture equation and include the stresses resulting from the interactions between both phases by using the solid volume-averaged momentum equation. The second approach is here followed. We start by writing the volume-averaged solid momentum equation from Equation 4.16, including all the previous assumptions:

$$\begin{aligned}&\frac{\partial \alpha_s \overline{\rho_s}\,\widetilde{\mathbf{v}_s}}{\partial t} + \nabla\cdot(\alpha_s \overline{\rho_s}\,\widetilde{\mathbf{v}_s}\otimes\widetilde{\mathbf{v}_s}) + \alpha_s \nabla\overline{p_s}\\ &= \nabla\cdot\left[\alpha_s \mu_s^{\text{eff}}\left\{\nabla\widetilde{\mathbf{v}_s} + (\nabla\widetilde{\mathbf{v}_s})^T - \frac{2}{3}(\nabla\cdot\widetilde{\mathbf{v}_s})\underline{\underline{I}}\right\}\right] + \alpha_s\overline{\rho_s\,\mathbf{b}_s} + \mathbf{S}^1_{\text{mom},s} + \mathbf{S}^{2,*}_{\text{mom},s}\,,\end{aligned} \tag{4.43}$$

We then make an assumption on the averaged velocity of the solid phase which is set to zero (because of the stationary nature of the solid), which implies that the averaged liquid velocity can be considered as the mixture velocity:

$$\widetilde{\mathbf{v}_s} = 0 \Rightarrow \widetilde{\mathbf{v}_l} = \mathbf{v}_m\,.$$

Equation 4.43 can thus be simplified into

$$\alpha_s\nabla\overline{p_s} = \alpha_s\overline{\rho_s\,\mathbf{b}_s} + \mathbf{S}^1_{\text{mom},s} + \mathbf{S}^{2,*}_{\text{mom},s}\,, \tag{4.44}$$

which provides an expression for the solid pressure gradient to be inserted in the mixture momentum equation. Before doing so, we use the jumps condition, Equation 4.17, to write

$$\mathbf{S}^1_{\text{mom},s} + \mathbf{S}^{2,*}_{\text{mom},s} = -\mathbf{S}^1_{\text{mom},l} - \mathbf{S}^{2,*}_{\text{mom},l}\,, \tag{4.45}$$

such that the mixture momentum equation, with $\overline{\mathbf{v}_s} = 0$ and the solid pressure gradient replaced by Equation 4.44 and Equation 4.45, becomes:

$$\begin{aligned}
&\frac{\partial(\rho_v \mathbf{v}_m)}{\partial t} + \nabla \cdot \left[\rho_v \mathbf{v}_m \otimes \mathbf{v}_m + \rho_v \frac{f_s}{f_l} \mathbf{v}_m \otimes \mathbf{v}_m\right] + \alpha_l \nabla \overline{p_l} \\
&= \nabla \cdot \left[\alpha_l \mu_l^{\text{eff}} \left\{\frac{1}{f_l}\left[\nabla \mathbf{v}_m + (\nabla \mathbf{v}_m)^T - \frac{2}{3}(\nabla \cdot \mathbf{v}_m)\underline{\boldsymbol{I}}\right]\right.\right. \\
&\qquad \left.\left. + \mathbf{v}_m \nabla\left(\frac{1}{f_l}\right) + \nabla\left(\frac{1}{f_l}\right)\mathbf{v}_m - \frac{2}{3}\left[\mathbf{v}_m \cdot \nabla\left(\frac{1}{f_l}\right)\right]\underline{\boldsymbol{I}}\right\}\right] \\
&+ \alpha_l \overline{\rho_l}\, \widetilde{\mathbf{b}_l} + \mathbf{S}^1_{\text{mom},l} + \mathbf{S}^{2,*}_{\text{mom},l}\,.
\end{aligned} \tag{4.46}$$

Like for the previous situation, several additional unknown terms are present in the equation and must be modeled or neglected based on the following assumptions:

- We do the same assumption as before on the second part of the liquid viscous stress tensor, Equation 4.41.
- We follow P. Prescott et al. (1991) by assuming that the momentum contribution due to phase change is negligible compared to the other terms, i.e.

$$\mathbf{S}^1_{\text{mom},l} = \mathbf{0}\,.$$

- Momentum transfer due to interfacial interactions between the liquid and solid phases can be expressed, in analogy with Darcy's law, in terms of a permeability tensor $\underline{\boldsymbol{K}}$:

$$\mathbf{S}^{2,*}_{\text{mom},l} = -\mu_l \underline{\boldsymbol{K}}^{-1} \cdot (\widetilde{\mathbf{v}_l} - \widetilde{\mathbf{v}_s})\,. \tag{4.47}$$

  Jun Ni and F. P. Incropera (1995b) explain that if the solid structures are assumed to be isotropic, the Carman-Kozeny expression to estimate a scalar permeability can be used:

$$\underline{\boldsymbol{K}} \Rightarrow K = \frac{\alpha_l^3\, d_s^2}{K_0\,(1-\alpha_l)^2}\,,$$

  where K_0 is the Kozeny coefficient and d_s a characteristic length of the solid structure. This dissipative interfacial stress should model the resistance the liquid experiences when flowing through a region of continuous solid structure. Note that it implicitly means that the drag interaction effect is supposed to be dominant compared to all other interfacial phenomena that could contribute to the exchange of momentum.

Applying those assumptions to Equation 4.46, it comes:

$$\begin{aligned}
&\frac{\partial(\rho_v \mathbf{v}_m)}{\partial t} + \nabla\cdot\left[\frac{\rho_v}{f_l}\mathbf{v}_m \otimes \mathbf{v}_m\right] + \alpha_l \nabla\overline{p_l} \\
&= \nabla\cdot\left[\frac{\alpha_l}{f_l}\mu_l^{\text{eff}}\left[\nabla\mathbf{v}_m + (\nabla\mathbf{v}_m)^T - \frac{2}{3}(\nabla\cdot\mathbf{v}_m)\underline{\underline{\boldsymbol{I}}}\right]\right] \\
&+ \alpha_l\,\overline{\rho_l}\,\widetilde{\mathbf{b}_l} - \mu_l\frac{\alpha_l^3\, d_s^2}{K_0\,(1-\alpha_l)^2}\mathbf{v}_m\,.
\end{aligned} \tag{4.48}$$

It must be highlighted that hybrid approaches have been proposed to accommodate the three sets of momentum equations presented above in a single model. That is the logic followed by Oldenburg and Spera (1992) who employ a switching function to change the model from a porous Darcy type flow to a non-Newtonian flow with mixture viscosity. This switching depends on a critical value of the liquid volume fraction. Below that threshold, the liquid is considered to flow across a rigid array of packed solid. Above the threshold, the mushy region consists of a slurry of free-floating equiaxed dendrites.

4.3.4 Energy equation

From the manipulations presented in Section 4.3.1, the mixture total energy equation can be written as

$$\begin{aligned}
&\frac{\partial(\rho_v E_m)}{\partial t} + \nabla\cdot\left[\rho_v H_m \mathbf{v}_m + \sum_{k=l}^{s}\alpha_k\,\overline{\rho_k}\,(\widehat{H_k} - H_m)\,(\widetilde{\mathbf{v}_k} - \mathbf{v}_m)\right] \\
&= \nabla\cdot(\alpha_l\,\overline{\underline{\boldsymbol{\tau}}_l}\cdot\widetilde{\mathbf{v}_l} + \alpha_s\,\overline{\underline{\boldsymbol{\tau}}_s}\cdot\widetilde{\mathbf{v}_s}) \\
&+ \nabla\cdot\left[\left(\alpha_l\lambda_l^{\text{eff}} + \alpha_s\lambda_s^{\text{eff}}\right)\nabla T_v\right] \\
&+ \rho_v\,(f_l\,\widetilde{\mathbf{b}_l}\cdot\widetilde{\mathbf{v}_l} + f_s\,\widetilde{\mathbf{b}_s}\cdot\widetilde{\mathbf{v}_s}) + \langle\dot{q}_\Gamma\,\delta_\Gamma\rangle\,.
\end{aligned} \tag{4.49}$$

As with the momentum equations, some additional simplifying assumptions must be made for the energy equation to be exploitable:

- Similarly to the inertia associated to the relative velocities, Equation 4.40, the energy flux associated to relative phase motion is neglected:

$$\sum_{k=l}^{s}\alpha_k\,\overline{\rho_k}\,(\widehat{H_k} - H_m)\,(\widetilde{\mathbf{v}_k} - \mathbf{v}_m) \approx 0\,.$$

- The body forces are assumed to be the same for both phases:

$$\widetilde{\mathbf{b}_l} = \widetilde{\mathbf{b}_s} = \mathbf{b}_m\,.$$

- For the viscous dissipation term, the distinction should be made between the three scenarios considered for the momentum equation:

– For the amorphous material and floating dispersed solid microstructures, we have

$$\begin{aligned}&\nabla\cdot(\alpha_l\,\overline{\underline{\boldsymbol{\tau}}_l}\cdot\widetilde{\mathbf{v}}_l+\alpha_s\,\overline{\underline{\boldsymbol{\tau}}_s}\cdot\widetilde{\mathbf{v}}_s)\\&=\nabla\cdot\left[(\alpha_l\,\mu_l^{\text{eff}}+\alpha_s\,\mu_s^{\text{eff}})\left[\nabla\mathbf{v}_m+(\nabla\mathbf{v}_m)^T-\frac{2}{3}(\nabla\cdot\mathbf{v}_m)\underline{\boldsymbol{I}}\right]\cdot\mathbf{v}_m\right].\end{aligned}$$

– For the stationary solid porous matrix, with the same assumptions as for the momentum equations, we get

$$\begin{aligned}&\nabla\cdot(\alpha_l\,\overline{\underline{\boldsymbol{\tau}}_l}\cdot\widetilde{\mathbf{v}}_l+\alpha_s\,\overline{\underline{\boldsymbol{\tau}}_s}\cdot\widetilde{\mathbf{v}}_s)\\&=\nabla\cdot\left[\frac{\alpha_l}{f_l}\,\mu_l^{\text{eff}}\left[\nabla\mathbf{v}_m+(\nabla\mathbf{v}_m)^T-\frac{2}{3}(\nabla\cdot\mathbf{v}_m)\underline{\boldsymbol{I}}\right]\cdot\mathbf{v}_m\right].\end{aligned}$$

– For the moving solid structures where the averaged liquid and solid velocities are distinct, the situation is more tricky as an important number of terms are involved. A detailed analysis is not conducted here because it goes beyond the scope of this work.

- Finally, it remains to model the averaged interfacial energy flux production. The assumption we make here is that this energy transfer between the two phases is only caused by the release or absorption of latent heat during phase transformation, all the other potential energy sources (like radiation e.g.) being neglected. We follow Voller et al. (1989) and model this energy source as

$$\langle\dot{q}_\Gamma\,\delta_\Gamma\rangle=-\frac{\partial(\rho_v f_l\,\Delta h)}{\partial t}-\nabla\cdot(\rho_v f_l\,\Delta h\,\mathbf{v}_m)\,, \tag{4.50}$$

where the jump in enthalpy is only activated when the local mass fraction of liquid, f_l, is different from zero and it is expressed as

$$\Delta h=\widetilde{h_s}(T_{\text{melt}})-\widetilde{h_l}(T_{\text{melt}})+\Delta h_{\text{lat}}\,, \tag{4.51}$$

with Δh_{lat}, the latent heat of fusion, and $\widetilde{h_s}(T_{\text{melt}})$ and $\widetilde{h_l}(T_{\text{melt}})$ the averaged solid and liquid enthalpies evaluated at the melting temperature. To interpret this expression, it is convenient to transfer the energy source term to the left-hand side of the energy equation and split its contributions between the temporal and convective terms. Doing so, we can realize that this enthalpy jump is added to the sensible liquid enthalpy:

$$\begin{aligned}H_m+\Delta h&=h_m+e_{kin,m}+f_l\,\Delta h\,,\\&=f_l\,\widetilde{h_l}(T)+f_s\,\widetilde{h_s}(T)+e_{kin,m}+f_l\left[\widetilde{h_s}(T_{\text{melt}})-\widetilde{h_l}(T_{\text{melt}})+\Delta h_{\text{lat}}\right],\\&=f_l\left[\widetilde{h_l}(T)+\widetilde{h_s}(T_{\text{melt}})-\widetilde{h_l}(T_{\text{melt}})+\Delta h_{\text{lat}}\right]+f_s\,\widetilde{h_s}(T)+e_{kin,m}\,.\end{aligned}$$

An important remark to make here is that the latent heat associated to the liquid-solid phase transformation should be set to zero for materials

exhibiting a second-order phase transition, such as amorphous materials.

Taking into account the considerations above, the mixture energy equation transforms into:

$$\begin{aligned}&\frac{\partial(\rho_v E_m)}{\partial t} + \nabla\cdot\left[\rho_v H_m \mathbf{v}_m\right]\\&= \nabla\cdot\left(\alpha_l \overline{\underline{\boldsymbol{\tau}}_l}\cdot\widetilde{\mathbf{v}_l} + \alpha_s \overline{\underline{\boldsymbol{\tau}}_s}\cdot\widetilde{\mathbf{v}_s}\right)\\&+ \nabla\cdot\left[\left(\alpha_l\lambda_l^{\text{eff}} + \alpha_s\lambda_s^{\text{eff}}\right)\nabla T_v\right]\\&+ \rho_v \mathbf{b}_m\cdot\mathbf{v}_m - \frac{\partial(\rho_v f_l \Delta h)}{\partial t} - \nabla\cdot\left(\rho_v f_l \Delta h\,\mathbf{v}_m\right)\,.\end{aligned} \tag{4.52}$$

4.4 Summary of the equations implemented in this work

In this thesis, the homogeneous equilibrium mixture model considered blends two models derived above, the one for non-separable phases and floating dispersed solid microstructures and the one for porous solid structure with constrained motion. The implemented equations rely on a compressible model and read:

$$\frac{\partial\rho_v}{\partial t} + \nabla\cdot\left[\rho_v\mathbf{v}_m\right] = 0\,, \tag{4.53a}$$

$$\begin{aligned}\frac{\partial(\rho_v\mathbf{v}_m)}{\partial t} + \nabla\cdot\left[\rho_v\mathbf{v}_m\otimes\mathbf{v}_m + p_v\underline{\boldsymbol{I}}\right] &= \nabla\cdot\left[\mu_v^{\text{eff}}\left(\nabla\mathbf{v}_m + (\nabla\mathbf{v}_m)^T - \frac{2}{3}(\nabla\cdot\mathbf{v}_m)\,\underline{\boldsymbol{I}}\right)\right]\\&+ \rho_v\mathbf{b}_m\,,\end{aligned} \tag{4.53b}$$

$$\begin{aligned}\frac{\partial(\rho_v E_m^*)}{\partial t} + \nabla\cdot\left[\rho_v H_m^*\mathbf{v}_m\right] &= \nabla\cdot\left[\mu_v^{\text{eff}}\left(\nabla\mathbf{v}_m + (\nabla\mathbf{v}_m)^T - \frac{2}{3}(\nabla\cdot\mathbf{v}_m)\,\underline{\boldsymbol{I}}\right)\right.\\&\left.\cdot\mathbf{v}_m\right]\\&+ \nabla\cdot\left[\lambda_v^{\text{eff}}\nabla T_v\right] + \rho_v\mathbf{b}_m\cdot\mathbf{v}_m\,.\end{aligned} \tag{4.53c}$$

The notable features of this model are described below.

Working variables The variables for which the system of equations is solved are the mixture pressure, velocities and temperature, i.e. the primitive variables set:

$$\mathbf{u} = \left[p_v,\ \mathbf{v}_m,\ T_v\right]\,.$$

Those variables, being part of the solution vector, are thus never assembled from the averaged phase variables. With this set of variables, the mixture kinetic energy, Equation 4.32, can be expressed as a function of $\mathbf{v}_m$ only,

$$e_{kin,m} = \frac{1}{2}\,\mathbf{v}_m\cdot\mathbf{v}_m\,.$$

Mixture quantities The mixture variables that are assembled from the phase variables are:

- From volume-averaged quantities:

$$\begin{aligned}\rho_v(p_v,T_v) &= \alpha_l(T_v)\,\overline{\rho_l}(p_v,T_v) + \alpha_s(T_v)\,\overline{\rho_s}(p_v,T_v)\,,\\ \mu_v^{\text{eff}}(T_v) &= \alpha_l(T_v)\,\overline{\mu_l} + \alpha_s(T_v)\,\overline{\mu_s}\,,\\ \lambda_v^{\text{eff}}(T_v) &= \alpha_l(T_v)\,\overline{\lambda_l} + \alpha_s(T_v)\,\overline{\lambda_s}\,.\end{aligned}$$

- From mass-averaged quantities:

$$\begin{aligned}C_{p,m}(p_v,T_v) &= f_l(p_v,T_v)\,\widetilde{C_{p,l}} + f_s(p_v,T_v)\,\widetilde{C_{p,s}}\,,\\ e_{int,m}(p_v,T_v) &= f_l(p_v,T_v)\,\widetilde{e_{int,l}}(p_v,T_v) + f_s(p_v,T_v)\,\widetilde{e_{int,s}}(p_v,T_v)\,,\\ h_m(p_v,T_v) &= f_l(p_v,T_v)\,\widetilde{h_l}(p_v,T_v) + f_s(p_v,T_v)\,\widetilde{h_s}(p_v,T_v)\,,\\ E_m(p_v,\mathbf{v}_m,T_v) &= e_{int,m}(p_v,T_v) + e_{kin,m}(\mathbf{v}_m)\,,\\ H_m(p_v,\mathbf{v}_m,T_v) &= h_m(p_v,T_v) + e_{kin,m}(\mathbf{v}_m)\,.\end{aligned}$$

- The mixture speed of sound is obtained either via the Wood's formula (see Wood (1930)),

$$\frac{1}{\rho_v\,c_m^2}(p_v,T_v) = \frac{\alpha_l}{\widetilde{\rho_l}(p_v,T_v)\,\overline{c_l}^2(p_v,T_v)} + \frac{\alpha_s}{\widetilde{\rho_s}(p_v,T_v)\,\overline{c_s}^2(p_v,T_v)}\,,$$

  or as a monotonic function,

$$c_m^2(p_v,T_v) = f_l(p_v,T_v)\,\overline{c_l}^2(p_v,T_v) + f_s(p_v,T_v)\,\overline{c_s}^2(p_v,T_v)\,.$$

Equations of state The EOS to describe each phase are allowed to be different. Typically, a Mie-Grüneisen type EOS described in Section 2.3.3 will be used for both phases. The parameters defining this kind of EOS, namely the thermal expansion coefficient, the isothermal compressibility, the reference density and the reference heat capacities, will be set different between the liquid and solid phases. The use of distinct EOS is for example involved in the computation of the mixture density where

$$\begin{aligned}&\overline{\rho_l}(p_v,T_v) \leftarrow \text{use of liquid EOS}\,,\\ &\overline{\rho_s}(p_v,T_v) \leftarrow \text{use of solid EOS}\,.\end{aligned}$$

Volume and mass fractions The expression for the liquid volume fraction is the one given by Equation 4.34, i.e.

$$\alpha_l(T_v) = \frac{1}{2}\left[\tanh\left(2\pi\,\frac{T_v - T_{\text{melt}}}{T_{\text{liq}} - T_{\text{sol}}}\right) + 1\right]\,,$$

with T_{melt}, the melting temperature, T_{liq}, the liquidus temperature and T_{sol}, the solidus temperature. These three temperatures are chosen depending on the material considered. However, it should be noted that for numerical reasons, if the properties jumps between the liquid and solid phases (including the latent heat) are important and the mesh refinement is not fine enough, it might be necessary to artificially widen the temperature range defining the width of the mushy zone. This is to avoid spurious oscillations around the interface in some fluid quantities. The solid volume fraction is simply obtained as

$$\alpha_s(T) = 1 - \alpha_l(T)\,,$$

while the mass fractions involve the phase densities and the mixture density,

$$f_l(p_v, T_v) = \alpha_l(T_v)\,\frac{\overline{\rho_l}(p_v, T_v)}{\rho_v(p_v, T_v)}\,,$$
$$f_s(p_v, T_v) = \alpha_s(T_v)\,\frac{\overline{\rho_s}(p_v, T_v)}{\rho_v(p_v, T_v)}\,.$$

Darcy term The Darcy-like term, Equation 4.47, to model the flow through a permeable solid matrix structure with constrained motion (i.e. moving with a known velocity or fixed) is included as a body forces such that

$$\rho_v\,\mathbf{b}_m \mathrel{-}= \frac{K_0\,(1-\alpha_l)^2}{\alpha_l^3 + \epsilon}\,(\mathbf{v}_m - \mathbf{v}_s^{\text{fix}})\,,$$

where $\mathbf{v}_s^{\text{fix}}$ is an imposed solid velocity, usually set to zero, ϵ is a small constant typically fixed to 10^{-3} in order to avoid division by a too small value, and K_0 is the Darcy factor commonly ranging from 10^4 to 10^{10}, depending on the situation. Note that the multiplication by the viscosity is not included here such that the amount of velocity penalization is only function of the value we select for K_0 and not the viscosity of the material under consideration. This term can be activated or not, depending on the targeted application.

Variable viscosity In addition to the Darcy term, the variable viscosity, Equation 4.37, modeling the behavior of an amorphous material and in a limited sense the presence of floating solid microstructures dispersed in the liquid, is given an expression taken from Nourgaliev et al. (2016):

$$\mu_v^{\text{eff}}(T_v) = \begin{cases} \overline{\mu_s}^* & \text{if } T_v < T^*\,, \\ \overline{\mu_l} f_\mu(T_v) & \text{if } T^* \le T_v \le T_{\text{liq}}\,, \\ \overline{\mu_l} & \text{otherwise}\,. \end{cases}$$

The function $f_\mu(T_v)$ smoothly varies from 1 to a large number, $\overline{\mu_s}^*/\overline{\mu_l}$, in the ranges of temperatures from the liquidus T_{liq} down to T^*. This model enables the effective mixture viscosity to smoothly transition from the dynamic viscosity

of the liquid in pure liquid regions (i.e. where $T_v > T_{\text{liq}}$) to a very large "solid-state" user-defined viscosity $\overline{\mu_s}^*$. The expression of $f_\mu(T_v)$ is rewritten here for convenience,

$$f_\mu(T_v) = 10^{\varphi(T_v)},$$

where:

$$\begin{aligned}
\varphi(T_v) &= -(a_0 - 4\,a_1)\,\psi(T_v) + 2(a_0 - 2\,a_1)\,\psi^2(T_v),\\
\psi(T_v) &= \frac{1}{2}\left[1 + \cos\left(\pi\left(b_0 + b_1\,T_v + b_2 T_v^2\right)\right)\right],\\
a_0 &= \log_{10}(\alpha\,\gamma_s),\\
a_1 &= \log_{10}(\gamma_s),\\
b_0 &= \frac{1}{d}\left[T^*\left(T_{\text{liq}}(T^* - T_{\text{liq}}) - 2\,T_{\text{sol}}(T^* - T_{\text{sol}})\right)\right],\\
b_1 &= \frac{1}{d}\left[(T^*)^2 + T_{\text{liq}}^2 - 2\,T_{\text{sol}}^2\right],\\
b_2 &= -\frac{1}{d}\left[T^* + T_{\text{liq}} - 2\,T_{\text{sol}}\right],\\
d &= 2\left(T^* - T_{\text{liq}}\right)\left(T^* - T_{\text{sol}}\right)\left(T_{\text{liq}} - T_{\text{sol}}\right),\\
T^* &= T_{\text{liq}} - \omega\left(T_{\text{liq}} - T_{\text{sol}}\right).
\end{aligned}$$

There are therefore three input parameters to this model: (i) γ_s, the value that the function $f_\mu(T_v)$ takes at the solidus temperature, (ii) the factor α that fixes the limiting value of the solid viscosity since

$$\overline{\mu_s}^* = \alpha\,\gamma_s\,\overline{\mu_l},$$

and (iii) the parameter ω influencing the width of the "creeping" zone, i.e. the zone over which the viscosity transitions. Note that a dimensionless temperature should be used in the above expressions. Similarly to the Darcy term, the user can decide to use or not the variable viscosity model.

Latent heat The absorption or release of latent heat during the liquid-solid phase transformation is not implemented as a source term like in Equation 4.52. It is instead incorporated during the assembly of the temporal and convective terms by modifying the liquid enthalpy contribution to the mixture enthalpy,

$$\begin{aligned}
h_m^*(p_v, T_v) = {} & f_l(p_v, T_v)\left[\widetilde{h_l}(p_v, T_v) + \widetilde{h_s}(p_v, T_{\text{melt}}) - \widetilde{h_l}(p_v, T_{\text{melt}}) + \Delta h_{\text{lat}}\right]\\
& + f_s(p_v, T_v)\,\widetilde{h_s}(p_v, T_v),
\end{aligned}$$

as explained before. The internal energy is directly obtained from this modified enthalpy,

$$e_{int,m}^*(p_v, T_v) = h_m^*(p_v, T_v) - \frac{p_v}{\rho_v},$$

and the total enthalpy H_m^* and energy E_m^* to be inserted in the temporal and convective terms are then readily obtained by adding the kinetic energy contri-

bution.

Link with the material type To close this chapter, the equations are further commented in the light of the models presented in Section 4.3. Among the three physical situations identified, it appears clearly that the model implemented in our solver coincides with the model for amorphous materials and dispersed floating solid microstructures. The only difference is the addition of a Darcy penalization source term. If the assumptions of mechanical equilibrium and equality of averaged velocities are not made, the equations inevitably become more complex. However, we have shown that by carefully postulating additional hypotheses for the two other situations, it is possible to arrive at momentum equations, Equation 4.42 and Equation 4.48, which strongly resemble the one we are using. The differences lie in some multiplicative factors in front of the viscous stress tensor or the pressure gradient. Concerning the energy equation, the model for the porous solid with constrained motion is very similar to the implemented energy equation. On the other hand, the energy equation describing the case of floating solid structures is more complex. The conclusion from this analysis is that although the implemented model is probably the simplest one we could come up with, it should still be able to describe the behavior of different liquid-solid phase transition situations, providing that they respect the underlying assumptions.

Some implementation aspects From an implementation point of view, no explicit filtering is performed in the code. Instead, the equations are written as presented in this chapter and the code thus solves the equations directly for the averaged quantities. More precisely, the working variables are chosen to be the mixture pressure, velocities and temperature and all the other mixture quantities are expressed based on them. This choice is the most convenient with respect to the way we incorporate the enthalpy of fusion into the model. It is also interesting to note that transport properties are temperature-dependent in those equations since they depend on the temperature-dependent volume fraction functions. Although the original Navier-Stokes equations for the gas consider constant properties, this is therefore not the case for the condensed phase. This implies that the Jacobians of the system have to be computed accordingly, taking into account the dependence of the transport properties to the solution vector.

Contribution 4.1: Models at the diffuse liquid–solid phase interface

The main contributions that can be identified from this chapter are:

- Starting from the most general volume-averaged Navier-Stokes equations, the thorough derivation of the homogeneous mixture model, including the statement of all the required hy-

potheses.
- The manipulation of the mixture equations to include specific models for the liquid-solid transition and the link with other existing equations.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- The extension of the class dealing with general EOS to define the liquid-solid mixture system, including latent heat release/absorption and two velocity penalization techniques in the mushy region.

This implementation in a discontinuous Galerkin framework constitutes an original contribution in the field.

Rest of this page intentionally left blank.

Part III

Numerical methods

The Argo DG platform

In this thesis, all the developments presented in this part III have been coded in the Argo platform, which is a proprietary code developed by the Belgian research center Cenaero. The following description of Argo is greatly inspired by Carton de Wiart (2014, Sec. 1.3). The development of the code, in its discontinuous Galerkin version, started in 2006 during the thesis of Koen Hillewaert (Hillewaert (2013)), under the supervision of Prof. Remacle. The core of the platform has been built as a substantial extension and generalization of the acoustics DG platform developed by the team of Prof. Remacle (Chevaugeon et al. (2007)), borrowing elements from the former finite volume version of Argo (Georges et al. (2008)) developed at Cenaero.
The solver is implemented in C++. It takes advantages of the full flexibility of polymorphism by handling generic conservation laws such that new physical models can be easily added. For these conservation laws, steady or unsteady approaches can be used. For unsteady computations, the code can solve both explicit and implicit temporal discretizations. The solver is mainly based on the discontinuous Galerkin method (DGM). When needed, the stabilization of the diffusive term is done using the interior penalty (IP) method. To deal with large cases, the solver relies on a hybrid parallelism between message passing interface (MPI) and open multiprocessing (OpenMP). The original DG discretization has been extended by Dr. Schrooyen at Cenaero (P. Schrooyen et al. (2017), P. Schrooyen et al. (2018)) toward the extended DG (XDG) method used in this work, exploiting class and functions templates offered by the C++ language. At this stage, the contributions of Pierre Schrooyen, Larbi Arbaoui and other of their coworkers at Cenaero to the original development of the DGImmersed module are gratefully acknowledged. It is indeed in this module that most of the developments were carried out during this thesis, in addition to the ones associated to the creation of the new GeneralEOS class containing the majority of physical models derived in this work.

Rest of this page intentionally left blank.

5

High-order spatial discretization with the discontinuous Galerkin method

Part of the contents of Section 5.3 was presented in Henneaux et al. (2019a) and published in Henneaux et al. (2019b). Some other material of this same section is the result of the master thesis of Claudio Miccoli (Miccoli (2021)) for which the present author was one of the advisors.

5.1 General overview of high-order methods for CFD 152
5.1.1 Weighted residual method: a unified framework to derive different classes of methods 152
5.1.2 Benefits, limitations and characteristics of DG schemes . 156
5.2 The discontinuous Galerkin method 160
5.2.1 Mesh and function space definitions 161
5.2.2 From the strong to the variational formulation of the Navier-Stokes equations 165
5.2.3 Ensuring stability of the discrete form through the definition of numerical fluxes 167
5.2.4 Assembly of the variational form 172
5.3 Implementation of an all-speed fluid solver 179
5.3.1 State-of-the-art overview and numerical challenges . . . 179
5.3.2 Versatile choice of working variables sets 183
5.3.3 Use of general equations of state 186
5.3.4 Treatment of low-Mach number flows by a density-based method 188

5.1 General overview of high-order methods for CFD

5.1.1 Weighted residual method: a unified framework to derive different classes of methods

The method of weighted residual (MWR) unifies many approximate methods for solving differential equations. The discontinuous Galerkin method is first introduced in this context to gain insight into its main features and to put it in perspective with other existing methods. The strong form of the differential problem we want to solve in this work, Equation 2.51, is first written under the following residual form: find $\tilde{\mathbf{u}} \in \tilde{\mathcal{U}} \subset C^2(\tilde{\Omega})$ such that

$$\tilde{\mathcal{R}}[\tilde{\mathbf{u}}] \triangleq \frac{\partial \tilde{\mathbf{u}}}{\partial t} + \tilde{\nabla} \cdot \underline{\boldsymbol{F}}(\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}}) - \mathbf{S}(\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}}) = \mathbf{0}\,, \qquad [5.1]$$

where

- The flux function $\underline{\boldsymbol{F}}(\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}}) = \underline{\boldsymbol{F}}^c(\tilde{\mathbf{u}}) - \underline{\boldsymbol{F}}^d(\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}})$ groups both convective and diffusive fluxes.
- $\tilde{\Omega} \subset \mathbb{R}^2$ is the two-dimensional open and bounded physical domain with smooth external boundaries $\partial\tilde{\Omega}$ that are one-dimensional manifold.
- $C^2(\tilde{\Omega})$ is the space of real-valued functions over $\tilde{\Omega}$ with continuous partial derivatives of order 2 and $\tilde{\mathcal{U}}$, called the solution or trial space, consists of functions in $C^2(\tilde{\Omega})$ that satisfy the boundary conditions of the PDE, i.e.

$$\tilde{\mathcal{U}} := \left\{ \tilde{\mathbf{u}} \in C^2(\tilde{\Omega}) : \tilde{\mathbf{u}} \text{ satisfies the BC on } \partial\tilde{\Omega} \right\}.$$

- $\tilde{\mathcal{R}} : C^2(\tilde{\Omega}) \to C^0(\tilde{\Omega})$ is the residual differential operator of order 2.
- $\tilde{\mathbf{u}}$ is the exact continuous and at least twice differentiable solution vector.
- $\tilde{\nabla}$ is the continuous gradient operator in two dimensions.

The continuous and discrete weighted residual formulation

Continuous form The strong form of the governing equations along with boundary conditions states the conditions that the solution must satisfy at every point over the domain. This formulation reaches its limits when non-smooth physical effects have to be represented (e.g. kinks in the material properties). It is therefore convenient to seek for an equivalent formulation of the problem that would impose conditions on the solution in an integral sense rather than in a pointwize manner. This is precisely what the weighted residual formulation achieves by proposing a weighted integral statement of the original differential

equations: find $\tilde{\mathbf{u}} \in \tilde{\mathcal{U}} \subset C^2(\tilde{\Omega})$ such that

$$\int_{\tilde{\Omega}} \tilde{\mathcal{R}}\left[\tilde{\mathbf{u}}\right] \, \tilde{\mathbf{w}} \, \mathrm{d}\mathbf{x} = \mathbf{0} \quad \forall \tilde{\mathbf{w}} \in C_c^\infty(\tilde{\Omega}) \,, \tag{5.2}$$

where $\tilde{\mathbf{w}}$ are the test functions belonging to the set of compactly supported and infinitely differentiable real-valued functions over $\tilde{\Omega}$, C_c^∞. By virtue of the fundamental lemma of variational calculus, Equation 5.1 and Equation 5.2 are equivalent. This is demonstrated by Matthew J. Zahr (2021, Sec. 3.3). It must however be noted that the weighted residual formulation does not incorporate any of the boundary condition of the original problem; those must be enforced strongly through the choice of the trial space.

Discretized form Up to now, both the trial and test functions are continuous and belong to infinite-dimensional function spaces. This is not appropriate for a numerical method. Two discretizations are therefore needed:

- *On the test functions*: we enforce the weighted residual equation on a finite-dimensional space $\mathcal{W}$. By doing so, the weighted residual formulation is no longer equivalent to the strong form, it becomes an approximation. The space $\mathcal{W}$ is constructed as the span of a linearly dependent set of functions $\{\mathbf{w}_1, \dots, \mathbf{w}_n, \dots, \mathbf{w}_{N_n}\}$,

  $$\mathcal{W} := \mathrm{span}\{\mathbf{w}_1, \dots, \mathbf{w}_n, \dots, \mathbf{w}_{N_n}\} \,.$$

  Note that the $\tilde{\bullet}$ notation is not used anymore to show that we are now considering a discrete space. As shown by Matthew J. Zahr (2021, Sec. 3.4), because the weight functions appear linearly in the residual formulation, Equation 5.2 holds for all $\mathbf{w}$ if and only if it holds for each $\mathbf{w}_n$, $n = 1, \dots N_n$:

  $$\int_{\tilde{\Omega}} \tilde{\mathcal{R}}\left[\tilde{\mathbf{u}}\right] \, \mathbf{w}_n \, \mathrm{d}\mathbf{x} = \mathbf{0} \quad \text{for } n = 1, \dots N_n \,.$$

- *On the trial functions*: we also need to restrict the trial space to a finite dimensional space, noted $\mathcal{U}$, to be able to perform the computations on a computer. Similarly to the test space, the basis of the trial space is composed by a set of linearly independent functions $\boldsymbol{\phi}_m$,

  $$\mathcal{U} := \mathrm{span}\{\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_m, \dots, \boldsymbol{\phi}_{N_m}\} \,.$$

  This second level of approximation with respect to the original strong form results in a residual which is not strictly equal to zero anymore,

  $$\mathcal{R}\left[\mathbf{u}\right] \neq \tilde{\mathcal{R}}\left[\tilde{\mathbf{u}}\right] = \mathbf{0} \,.$$

  The discretized form of the MWR thus seeks to minimize the residual, $\mathcal{R}$, by forcing a weighted integral of the strong form to zero for an appropriate

set of weighting functions:

$$\int_\Omega \mathcal{R}[\mathbf{u}]\ \mathbf{w}_n\ \mathrm{d}\mathbf{x} = \mathbf{0} \quad \text{for } n = 1, \dots N_n\,. \tag{5.3}$$

This amounts to an orthogonal projection of the residual of the equations on the space spanned by the test functions. Note that related to the discretization of the trial space, there is also the spatial discretization of the original continuous domain such that the computational domain becomes an approximation of the original one, $\Omega \approx \tilde{\Omega}$.

The choice of the trial and test functions

Both the trial and test functions belong to finite-dimensional spaces. There exists different ways of building such spaces, which leads to the derivation of distinct numerical methods. Some of them are reviewed below.

Trial functions There are two main ways of representing the solution with functions from a discrete space:

- *Global representation*: in a global method, the degrees of freedom over the entire domain contributes to the approximation of the solution at each computational node. This representation is at the basis of global spectral methods which present the most efficient approximation in terms of number of degrees invested, as noted by A. Beck (2015, Sec. 3.1).
- *Local representation*: in a local method, a solution representation in terms of a locally defined basis with compact support is adopted. The local ansatz functions can be either low order functions like the hat function in the "classical" FEM or higher order functions, on which spectral (high-order) FEM are built. Besides the choice of the basis, another distinguishing feature is the choice of the coupling between the subdomains. In continuous FE, the coupling is done by enforcing continuity of the local basis at the subdomains interfaces. This class of methods is labeled as conforming FEM because the discrete approximation space is a subset of the continuous one, $\mathcal{U} \subset \tilde{\mathcal{U}}$. In contrast to that, discontinuous methods such as DG are non-conforming, i.e. $\mathcal{U} \not\subset \tilde{\mathcal{U}}$, because the degrees of freedom are purely element-local, i.e. they are doubled-valued at the element faces.

The two most common approaches for selecting the nature of the trial functions are polynomial and trigonometric approximations.

Test functions The choice of the test functions, and thus of the way the expansion coefficients in the solution are computed from the weighted residual formulation, is the second fundamental ingredient in the derivation of a numerical method. The most common approaches are commented below.

- *Collocation*: in point collocation methods, the weight functions are selected in such a way that the residual is set to zero at N_n distinct points in the domain:
$$\mathbf{w}_n(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_n)\,, \quad \text{for } n = 1, \dots, N_n\,,$$
where $\mathbf{x}_n$ are selected collocation points throughout the domain and δ is the Dirac function. With a local solution approximation, the collocation method is related to the finite differences method. With a global approximation, it leads to pseudo-spectral methods.
- *Subdomain*: in the subdomain methods, each weight function is selected as unity over a specific subdomain and set equal to zero over the other parts:
$$\mathbf{w}_n(\mathbf{x}) = \begin{cases} 1\,, & \text{if } \mathbf{x}_n \le \mathbf{x} \le \mathbf{x}_{n+1}\,, \\ 0\,, & \text{otherwise}\,. \end{cases}$$
This choice implies that the average of the residual over each subdomain is forced to be zero. This is equivalent to the FVM if the solution representation is local and discontinuous.
- *Petrov-Galerkin*: if we select independently the test and trial basis,
$$\mathbf{u}(\mathbf{x}) = \sum_{m=1}^{N_m} \mathbf{U}_m\, \phi_m(\mathbf{x})\,,$$
$$\mathbf{w}_n(\mathbf{x}) = \psi_n(\mathbf{x})\,, \quad \text{for } n = 1, \dots N_n\,,$$
with ϕ_m and ψ_n being polynomial or trigonometric functions, we obtain methods belonging to the family of Petrov-Galerkin methods. A notable example is the Streamline-Upwind Petrov-Galerkin (SUPG) FEM.
- *(Bubnov)-Galerkin*: in the Galerkin or Bubnov-Galerkin methods, the trial and test functions are sampled from the same functional space,
$$\mathcal{U} = \mathcal{W},$$
such that
$$\psi_n(\mathbf{x}) = \phi_n(\mathbf{x})\,, \quad \text{for } n = 1, \dots N_n\,.$$

From the weighted residual to the weak formulation

The necessity of enforcing strongly the boundary conditions through the construction of the trial space in the weighted residual formulation makes the derivation of numerical methods not really practicable. The weak formulation is obtained from the weighted residual one by (i) moving derivatives from the PDE solution $\mathbf{u}$ to the test function $\mathbf{w}$ via integration-by-parts, and (ii) incorporating the natural Neumann boundary conditions into the flux boundary term

that arises from the integration-by-parts, such that

$$\int_\Omega \left[\frac{\partial \mathbf{u}}{\partial t} - \underline{\boldsymbol{F}}(\mathbf{u}, \nabla \mathbf{u}) \, \nabla \mathbf{w}_n - \mathbf{S}(\mathbf{u}, \nabla \mathbf{u}) \right] \mathbf{w}_n \, \mathrm{d}\mathbf{x} + \int_{\partial\Omega} \underbrace{(\underline{\boldsymbol{F}}(\mathbf{u}, \nabla \mathbf{u}) \cdot \mathbf{n}_{\partial\Omega})}_{\underline{\boldsymbol{F}}_{\text{Neu}}} \mathbf{w}_n \, \mathrm{d}s = \mathbf{0} \quad \text{for } n = 1, \dots N_n \,. \tag{5.4}$$

The weak formulation improves upon the weighted residual formulation in two aspects. The first one is that the regularity requirements on the trial space are weakened since solutions only need to be differentiable once[1]. The second one is that it extends the trial space because it does not have to be restricted to functions that satisfy the natural BC since they are directly weakly enforced into the weak form. As far as the essential Dirichlet BC are concerned, they can either be imposed in a strong or weak manner. For a strong imposition, the construction of a trial space restricted to functions satisfying the conditions is needed. This approach is typically followed in standard FEM where the essential BC are handled after the global stiffness matrix is assembled. For weak BC imposition, there is no restriction on the trial space but additional terms realizing the weak imposition need to be incorporated into the weak form. The DGM relies on this approach. In fact, discontinuous FEM not only exploit this weak condition enforcement at the domain boundaries but also at every element face to couple the solutions between adjacent elements.

Weak entropy solution Let us conclude this section by noting that although seeking for a weak solution of the original problem is required if the solution is not smooth (it extends the strong form to allow non-smooth solutions to exist in a distributional sense), there may exist in fact different weak solutions to the same problem. In order to single out the physically relevant solution, an extra admissibility criterion must be considered. For the Navier-Stokes system, this is the entropy inequality which can be exploited to ensure that the numerical approximation comply, in an integral sense, with the second law of thermodynamics.

5.1.2 Benefits, limitations and characteristics of DG schemes

The method on which this work is based is the DGM, which is characterized, from the discussion above, by (i) a local representation of the solution without continuity enforcement between elements, (ii) the use of identical trial and test functions spaces spanned by high-order polynomial functions, (iii) weak enforcement of boundary conditions. The benefits, limitations and distinguishing

[1] Having the possibility to use a linear representation of the solution for the discretization of second-order PDEs plays a particularly important role in the construction of classical low-order finite element methods since it enables the use of piecewise linear basis functions, which is by far the most widely used finite element solution space.

features between different DG formulations are reviewed below.

Benefits

- *High-order accuracy*: compared to standard low-order methods, high-order methods present two main favorable properties:
  - *Computational efficiency in terms of accuracy*: the most prominent factor for a numerical simulation in terms of memory, wall clock time and pre- and post-processing operations is the total number of degrees of freedom (DOF). A. Beck (2015, Sec. 3.1) introduces the concept of "points per wavelength" as the number of required grid points to resolve all the flow scales down to a given wavelength. The author shows that the recovered scales per invested DOF dramatically increase for higher order approximations. In other words, the accuracy per DOF of high-order methods is higher than for low order methods. However, it must be specified that in terms of operations count per DOF[2], higher-order methods are much more expensive. Since the number of operations typically scales as a fixed power of the number of DOF, there will be a point from which the gain in accuracy can outweigh the operation cost. That is the reason why high-order methods have the potential to make previously intractable problems accessible to simulation in an increasingly wide range of applications (Kronbichler and Per-Olof Persson (2021, Ch. 1)).
  - *Advantageous dispersion and dissipation properties*: high-order methods possess (much) smaller numerical dispersion and dissipation errors over a wider range of scales. This is a key feature to simulate phenomena involving small-scale and small-amplitude perturbations such as wave propagation, transition to turbulence and vortical structures in turbulent flows. The numerical dissipation at higher wavenumbers can also be exploited as a built-in implicit subgrid-scale model for under-resolved turbulence simulations.
  - *Proven optimality in L_2-norm for the Navier-Stokes equations*: some specific versions of the DG method, like the tensorial based SIPG version of Ralf Hartmann and Houston (2008) or the BR2 version of Bassi and Rebay (2000), have been proven to lead to optimal orders of convergence when the error is measured in terms of the L_2-norm for the set of compressible Navier-Stokes equations. This is yet another asset of the DG method for the applications we are targeting in this work.
- *Flexibility of element-based discretization*: compared to high-order global spectral methods, element-based discretizations are much more appropriate to solve problems in complex geometries by allowing non-periodic

[2] As noted by A. Beck (2015, Sec. 3.1), operation count is far from being the only determining factor influencing the numerical efficiency of a given implementation. But it gives a good indication in terms of wall-clock times.

basis functions, flexible domain discretization and efficient domain decomposition strategies for parallelization.

- *Adapted to arbitrarily shaped hybrid structured-unstructured and high-order meshes*: the locality and construction of the DG schemes based on mappings between physical and reference elements make them highly flexible to work on any type of elements, with or without curved boundaries. The discontinuous character of the approximation makes them also suited for non-conformal meshes. This in contrast to less flexible high-order finite differences methods for instance.
- *hp-adaptation*: compared to other high-order methods, DG methods are based on a strong mathematical foundation coming from the FE analysis that can be exploited for error estimation and adaptation purposes, both on the mesh (h) and the local interpolation order (p).
- *Local conservation*: an attractive feature that DGM share with FVM is the local conservation of physical properties, i.e. the guaranty that the net flux over each cell is in balance. This is not the case for standard FEM for which only global conservation is guaranteed.
- *Weak imposition of boundary conditions*: Benedikt Schott (2017, Sec. 4.2.2) emphasizes that the interest of weak constraint imposition is to relax the strength of strongly imposing the constraints to the benefit of improved accuracy, stability and robustness.
- *Stable discretization of convective operator*: in continuous formulations, some form of parameter-dependent upwinding, that introduces additional terms and decrease computational efficiency, are required. In contrast, discontinuous approximations like DG allow adding physics-informed mechanisms (the numerical fluxes) for the inter-element coupling which introduce numerical dissipation in a controlled way and provide robustness for transport-dominated processes.
- *Well-suited to parallelism and emerging computing architectures*: Abdi et al. (2019) point out the two main reasons for which element-based methods like DG are well suited for distributed computing. The first one is the localized memory access thanks to the compactness of the schemes which results in low communication overhead. The second reason is that high-order polynomial expansion of the solutions within each element leads to a large arithmetic intensity per DOF. These two properties ensures a high flop-to-communication ratio and make such methods ideal candidates for GPUs and many-core architectures (Fernández (2019, Sec. 1.4)).

Limitations and challenges

- *Computational cost*: by construction, the number of DOF in DGM can be significantly increased compared to their continuous counterparts because of the node duplication on the element faces. To alleviate this increased computational cost, different hybridization strategies have been proposed: hybrid DG, hybridisable DG (HDG) and hybrid high order

(HHO) methods. Giacomini et al. (2021) explain that the idea behind those new schemes is to eliminate the cellwize unknowns by expressing them as a function of DOF on the faces, similarly to static condensation in continuous FEM. This results in schemes having fewer globally coupled unknowns than standard DGM, thereby reducing the computational cost and memory footprint.

- *Robustness*: a silent assumption behind the advantages of the high-order methods mentioned above is the regularity of the solution to be discretized. That is the reason why special measures need to be taken to treat flow discontinuities like shocks, contacts or material interfaces (one of the objective of the present work). In a less obvious way, under-resolved flow features can also pose serious issues. As explained by A. Beck (2015, Sec. 4.1.1), insufficient grid resolution might be perceived as artificial roughness by the numerical scheme, which can degrade its theoretical error behavior. Associated to this decrease in order of accuracy, stability of high-order methods can be severely compromised by the Gibbs phenomena around discontinuous solutions and the aliasing errors that tend to accumulate energy in the higher under-resolved modes. Those instabilities can be addressed through various stabilization techniques such as artificial viscosity, filtering or slope limiting. However, as noted by Chan et al. (2019), these techniques often deteriorate locally the order of accuracy of the methods and it is not possible to prove that high-order schemes do not blow up even in the presence of stabilization. That is why, the design of high-order numerical schemes obeying one or both of the following properties, which are expected to combat the adverse effects mentioned above, has been a very active research area over the past decade:

  - *Entropy stability*: The discrete equivalent of the entropy inequality that is used to select the physically-relevant weak solution is referred to as entropy stability of the numerical scheme. This is a desired property for a scheme as the lack of entropy stability is a typical reason for convergence to nonphysical weak solutions and subsequent blow-up. However, Kuzmin (2021) notes that even if an entropy-stable scheme may exhibit an excellent convergence behavior in smooth regions, oscillations can still appear around high gradients regions. G. J. Gassner and Winters (2021) highlight that the role of entropy stability and how it fits into the question of provable nonlinear stability is still an active research area.
  - *Bound-preservation*: because entropy stability is not enough, a second mechanism ensuring the bounding or positivity of important physical quantities is critical for the sake of robustness of numerical computation, as noted by Xiangxiong Zhang (2017). This is often realized by means of appropriate limiters. Combined with this, it is important to consider strong stability preserving time-stepping schemes (SSP) that will preserve the maximum principle. Kuzmin

(2021) also remarks that a positivity-preserving scheme may converge to the wrong weak solutions if they are not entropy stable.

Distinguishing features of DGM

In addition to the characteristic features described above, there are a number of other specificities which lead to various formulations of the DGM:

- *Basis functions*: in the modal DGM, the shape functions are generic and independent of the element type. Those are typically orthogonal polynomials such as Legendre polynomials. In the nodal DGM, a set of independent control points associated to the element is used to fit interpolating polynomials; Lagrange interpolants being the most prevalent.
- *Quadrature rules*: to evaluate the integrals of the weak form, Gaussian quadrature rules are typically employed. The most common choices for the integration points are the Gauss-Legendre (GL) and Gauss-Lobatto-Legendre (GLL) points.
- *Control points for nodal basis*: for a given basis, different nodal interpolation points can be considered. Popular choices include equispaced points inside the element, or choosing the interpolation points as being collocated to the quadrature points, i.e. with the GL or GLL points. While the first choice leads to standard nodal DG, the second one is at the basis of DG Spectral Element Methods (DGSEM). A comparison between those different choices of interpolation points has been done by Bergmann et al. (2016).
- *Second order derivatives term*: to discretize second-order derivatives terms like the diffusive flux in the Navier-Stokes equations, Hillewaert (2023, Ch. 4) distinguishes two main families of methods: (i) the interior penalty methods (IPM) that can be seen as the application of FEM individually to each element, complemented by weak continuity coupling conditions at the faces, (ii) lifting methods such as Bassi-Rebay (BR) and local DG (LDG) methods where the equations are first rewritten into a first order system, similarly to mixed FE formulations, before applying the discontinuous Galerkin method on the system.

The temporal discretization, which is not directly related to the DG spatial discretization, introduces another level of possible choices.

5.2 The discontinuous Galerkin method

Based on the overview of the previous section, let's begin by specifying the DGM we are using in this work. It is a nodal DG using equidistant Lagrange basis and Gauss-Legendre quadrature rules. The elliptic terms are handled by the interior penalty method. We do not consider any improvements in terms of computational efficiency (such as hybridization) or robustness (such as entropy

preserving or limiting) to this standard DG scheme. This section presents the building elements of the DGM that were already implemented in the Argo code.

5.2.1 Mesh and function space definitions

Background mesh definition

The first step is the geometrical discretization of the continuous domain $\tilde{\Omega}$ on which the PDE is posed. Let $\mathtt{T}$ be a finite collection of N_T non-overlapping shape-regular simplices or quadrangles $\mathtt{T} = \{\mathtt{T}_m\}_{m=1,\dots,N_T}$ forming an approximate partition of $\tilde{\Omega}$:

$$\overline{\tilde{\Omega}} \approx \overline{\Omega} = \bigcup_{m=1}^{N_T} \overline{\mathtt{T}}_m\,, \quad \mathtt{T}_a \cap \mathtt{T}_b = \emptyset \text{ if } a \neq b\,. \tag{5.5}$$

The external boundary of $\mathtt{T}$ is noted $\partial\mathtt{T}$. Let $\mathtt{B}$ be the set of N_B boundary faces and $\mathtt{E}$ the set of N_E internal faces such that

$$\mathtt{B} = \left\{\{\mathtt{B}_b\}_{b=1,\dots,N_B} \mid \exists\, \mathtt{T}_m \in \mathtt{T} \text{ s.t. } \mathtt{B}_b = \partial\mathtt{T}_m \cap \partial\mathtt{T}\right\}\,,$$
$$\mathtt{E} = \left\{\{\mathtt{E}_e\}_{e=1,\dots,N_E} \mid \exists\, \mathtt{T}_{m1},\, \mathtt{T}_{m2} \in \mathtt{T} \text{ s.t. } \mathtt{E}_e = \partial\mathtt{T}_{m1} \cap \partial\mathtt{T}_{m2}\right\}\,.$$

For each $\mathtt{B}_b$ and $\mathtt{E}_e$, $\mathbf{n}_b$ and $\mathbf{n}_e$ are their respective unit outward normal vector. The notation $\mathbf{u}_e^+$ and $\mathbf{u}_e^-$ is used to distinguish traces of the discontinuous quantities on either side of a given $\mathtt{E}_e$, depending on the element from which the internal face is approached. The $\bullet^+$ sign corresponds to the element of which its exterior normal $\mathbf{n}_e$ corresponds to that of the oriented face, whilst $\bullet^-$ corresponds to the other. For a boundary face, $\mathbf{n}_b$ points outward to $\mathtt{T}$ and the traces taken from within the interior of the boundary element are noted $\mathbf{u}_b^+$.
Let us note that the use of high-order curved elements is possible in the solver.

Reference mapping

In order to simplify the operations on elements with arbitrary shapes, FEM standardized or parametric coordinates, which are defined per element type. It is assumed that each $\mathtt{T}_m \in \mathtt{T}$ in the coordinate system $\mathbf{x} = (x, y)$ is an image of a fixed reference element $\hat{\mathtt{T}}$ in parametric coordinates $\boldsymbol{\xi} = (\xi_1, \xi_2)$ through a smooth bijective mapping σ_m,

$$\begin{aligned} \sigma_m &: \mathbb{R}^d \to \mathbb{R}^d,\ \boldsymbol{\xi} \mapsto \mathbf{x}\,, \\ \sigma_m^{-1} &: \mathbb{R}^d \to \mathbb{R}^d,\ \mathbf{x} \mapsto \boldsymbol{\xi}\,, \end{aligned} \tag{5.6}$$

such that

$$\mathtt{T}_m = \sigma_m(\hat{\mathtt{T}})\,.$$

In this treatise, only two-dimensional parametric reference triangular and quadrilateral elements will be considered. They are defined as

$$\hat{\mathbb{T}} = \{0 \leq \xi_1 \leq 1,\ 0 \leq \xi_2 \leq 1,\ 0 \leq \xi_1 + \xi_2 \leq 1\} \qquad \text{if } \hat{\mathbb{T}} \text{ is a simplex ,}$$
$$\hat{\mathbb{T}} = \{-1 \leq \xi_1 \leq 1,\ -1 \leq \xi_2 \leq 1\} \qquad \text{if } \hat{\mathbb{T}} \text{ is a quadrangle .}$$

The Jacobian of the mapping is noted $\underline{\boldsymbol{J}}^{\sigma}$ and is given in two dimensions as

$$\underline{\boldsymbol{J}}^{\sigma} = \frac{\partial \mathbf{x}}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}) = \begin{bmatrix} \frac{\partial x}{\partial \xi_1} & \frac{\partial x}{\partial \xi_2} \\ \frac{\partial y}{\partial \xi_1} & \frac{\partial y}{\partial \xi_2} \end{bmatrix}. \tag{5.7}$$

The solutions and the test functions will be expressed in terms of the parametric coordinates. The physical gradient of a quantity f will then be obtained from

$$\frac{\partial f}{\partial \mathbf{x}} = \frac{\partial f}{\partial \boldsymbol{\xi}} \cdot \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}},$$

where the inverse of the gradient of the transformation,

$$\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} = \left(\underline{\boldsymbol{J}}^{\sigma}\right)^{-1},$$

is required and needs to be well-defined, which will be the case for convex triangles and quadrilaterals elements. Finally, the integration of a function over a physical element is transferred to the integration over the reference element through the ratio of elementary area,

$$\mathrm{d}\mathbf{x} = \left\| \left(\frac{\partial \mathbf{x}}{\partial \xi_1} \mathrm{d}\xi_1 \right) \times \left(\frac{\partial \mathbf{x}}{\partial \xi_2} \mathrm{d}\xi_2 \right) \right\| = \underbrace{\det\left(\underline{\boldsymbol{J}}^{\sigma}\right)}_{|\underline{\boldsymbol{J}}^{\sigma}|} \underbrace{\mathrm{d}\xi_1 \mathrm{d}\xi_2}_{\mathrm{d}\boldsymbol{\xi}},$$

which leads to

$$\int_{\mathbb{T}_m} f(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_{\hat{\mathbb{T}}} f(\boldsymbol{\xi})\, |\underline{\boldsymbol{J}}^{\sigma}|\, \mathrm{d}\boldsymbol{\xi}. \tag{5.8}$$

Standard function space

On the reference element $\hat{\mathbb{T}}$, the function space of degree $p \geq 0$ is defined as

$$\mathcal{D}_p(\hat{\mathbb{T}}) := \mathrm{span}\left(\prod_{i=1}^{d} \xi_i^{p_i} \mid 0 \leq \sum_{i=1}^{d} p_i \leq p \right) \qquad \text{if } \hat{\mathbb{T}} \text{ is a simplex element ,}$$
$$\mathcal{D}_p(\hat{\mathbb{T}}) := \mathrm{span}\left(\prod_{i=1}^{d} \xi_i^{p_i} \mid \forall i\colon 0 \leq p_i \leq p \right) \qquad \text{if } \hat{\mathbb{T}} \text{ is a tensor-product element .}$$

The broken vector-valued space $\mathcal{W}_p$ is then introduced. It contains L^2 functions on $\mathbb{T}$ whose restriction to each element $\mathbb{T}_m$ mapped to the reference element

belongs to the function space $\mathcal{P}_p$, i.e.

$$\mathcal{W}_p(\mathbb{T}) := \left\{ \mathbf{w} \in \left[L^2(\Omega)\right]^{N_f} : \mathbf{w}_{|_{\mathbb{T}_m}} \circ \sigma_m^{-1} \in \left[\mathcal{P}_p(\hat{\mathbb{T}})\right]^{N_f},\ \forall\, \mathbb{T}_m \in \mathbb{T} \right\}, \qquad [5.9]$$

where N_f is the number of fields in the solution vector, $\mathbf{w}_{|_{\mathbb{T}_m}}$ indicates a function whose support is restricted to element $\mathbb{T}_m$ and L^2 is a Hilbert space equipped with L^2-inner product $\int_\Omega fg \, \mathrm{d}\mathbf{x}$. The trial space $\mathcal{W}_p$ being a broken space, it can be composed by the weighted sum of functions which have a support restricted to a single element, i.e. it is the direct sum of element function spaces $\mathcal{W}_{p,m}(\mathbb{T}_m)$:

$$\mathcal{W}_p(\mathbb{T}) = \oplus_{m=1}^{N_T} \mathcal{W}_{p,m}. \qquad [5.10]$$

On each element $\mathbb{T}_m$, the space-dependent shape functions $\boldsymbol{\phi}_{m,-} \in \mathbb{R}^{N_n \times 1}$,

$$\boldsymbol{\phi}_{m,-}(\boldsymbol{\xi}) = \left[\phi_{m,1}, \dots, \phi_{m,n}, \dots, \phi_{m,N_n}\right]^T,$$

are then the chosen base for the element interpolation space $\mathcal{W}_{p,m}$,

$$\mathcal{W}_{p,m} = \mathrm{span}\left\{\phi_{m,n},\ n = 1, \dots, N_n\right\}. \qquad [5.11]$$

In this work where a nodal formulation is followed, the shape functions are chosen to be Lagrange interpolants of a given order p. Consider a set of N_n interpolation points defined on the reference element $\hat{\mathbb{T}}$,

$$\boldsymbol{\xi}_{\mathrm{IP},-} = \left\{\boldsymbol{\xi}_{\mathrm{IP},1}, \dots, \boldsymbol{\xi}_{\mathrm{IP},n}, \dots, \boldsymbol{\xi}_{\mathrm{IP},N_n}\right\}.$$

The set of Lagrange polynomials, $\boldsymbol{\phi}_{m,-}$, based on the space $\mathcal{W}_{p,m}$ and the set of points $\boldsymbol{\xi}_{\mathrm{IP}}$, is then defined as

$$\boldsymbol{\phi}_{m,-} = \left\{\phi_{m,n} \in \mathcal{W}_p : \phi_{m,n}(\boldsymbol{\xi}_{\mathrm{IP},n^*}) = \delta_{nn^*},\ n, n^* = 1, \dots, N_n\right\}.$$

In this work, only equidistant control points are considered as we do not use interpolation orders higher than four.

The time-dependent nodal degrees of $\underline{\boldsymbol{U}}_m(t) \in \mathbb{R}^{N_n \times N_f}$ are defined per element m, for each interpolation point n on this element and for each field f in the solution vector:

$$\underline{\boldsymbol{U}}_{m,-,-}(t) = \begin{bmatrix} U_{m,1,1} & \dots & U_{m,1,f} & \dots & U_{m,1,N_f} \\ & \ddots & \vdots & & \\ \vdots & & U_{m,n,f} & & \vdots \\ & & \vdots & \ddots & \\ U_{m,N_n,1} & \dots & U_{m,N_n,f} & \dots & U_{m,N_n,N_f} \end{bmatrix} = \begin{bmatrix} \mathbf{U}_{m,1,-} \\ \vdots \\ \mathbf{U}_{m,n,-} \\ \vdots \\ \mathbf{U}_{m,N_n,-} \end{bmatrix}.$$

The finite element approximate solution over the domain, $\mathbf{u}(\mathbf{x}, t) \in \mathcal{R}^{1 \times N_f}$, is

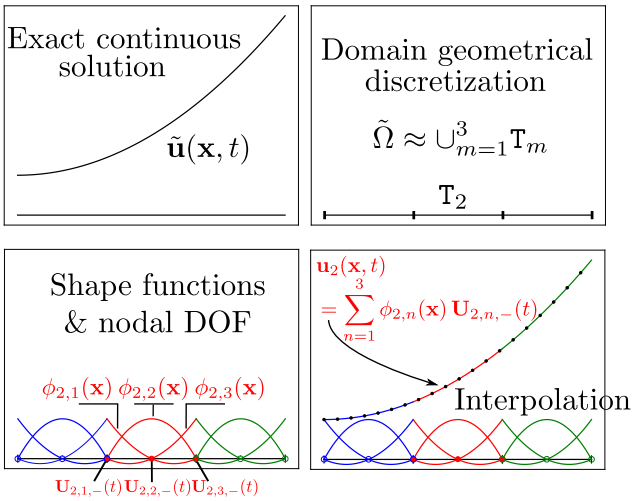


Figure 5.1 / Illustrative sketch of some elements involved in the DG discretization. Note that on the last sketch, the solution can be discontinuous across the element faces.

obtained as the sum of the solutions on each element,

$$\mathbf{u}(\mathbf{x}, t) = \sum_{m=1}^{N_T} \mathbf{u}_m(\mathbf{x}, t)\,, \tag{5.12}$$

with $\mathbf{u}_m$ expressed as a linear combination between the element shape functions and the nodal expansion weights, i.e.

$$\mathbf{u}_m(\mathbf{x}, t) = \sum_{n=1}^{N_n} \phi_{m,n}(\sigma_m(\boldsymbol{\xi}))\, \mathbf{U}_{m,n,-}(t) = \boldsymbol{\phi}_{m,-}^T(\mathbf{x}) \underline{\boldsymbol{U}}_{m,-,-}(t)\,,$$

The broken gradient operator $\nabla\colon H^1(\mathtt{T}) \to \left[L^2(\mathtt{T})\right]^d$ is defined by Di Pietro and Ern (2012, Sec. 1.2.5) as

$$(\nabla v)_{|_{\mathtt{T}_m}} := \nabla\left(v_{|_{\mathtt{T}_m}}\right)\,, \forall\, \mathtt{T}_m \in \mathtt{T}\,,$$

where $H^1(\mathtt{T})$ is the space of L^2 functions on $\mathtt{T}$ whose restriction to each element $\mathtt{T}_m$ belongs to the Sobolev space of order 1, $H^1(\mathtt{T}_m)$.

Figure 5.1 presents an illustrative sketch of some central elements discussed up to now involved in the DG discretization.

5.2.2 From the strong to the variational formulation of the Navier–Stokes equations

A preliminary definition necessary for the derivation of the variational formulation is about the homogeneity tensors $\underline{\boldsymbol{G}}_{kl}$ which are defined as the Jacobians of the diffusive fluxes, Equation 2.52, with respect to the solution gradients. More precisely,

$$\begin{aligned} \underline{\boldsymbol{G}}_{kl}(\mathbf{u}) &= \frac{\partial \underline{\boldsymbol{F}}_k^d}{\partial \nabla_l \mathbf{u}} \\ \Leftrightarrow \underline{\boldsymbol{F}}_k^d(\mathbf{u}, \nabla\mathbf{u}) &= \underline{\boldsymbol{G}}_{kl}(\mathbf{u})\, \nabla_l \mathbf{u} \end{aligned} \qquad [5.13]$$

is the homogeneity tensor of the diffusive fluxes along the kth-direction with respect to the gradient of the solution along the lth-direction. In two dimensions, $k = \{x, y\}$ and $l = \{x, y\}$, and the four tensors can be assembled into a global tensor as

$$\underline{\boldsymbol{G}} = \begin{bmatrix} \underline{\boldsymbol{G}}_{xx} & \underline{\boldsymbol{G}}_{xy} \\ \underline{\boldsymbol{G}}_{yx} & \underline{\boldsymbol{G}}_{yy} \end{bmatrix}. \qquad [5.14]$$

The expressions of the four homogeneity tensors for general equations of state can be found in Appendix A.

Going from the continuous strong form of the compressible Navier-Stokes system, Equation 2.51, to a discrete weak form suited to be implemented inside a solver requires several steps. In this work, the framework and formalism presented by Ralph Hartmann (2008) is followed and summarized here for convenience:

1. Rewrite the problem as a first-order system by using the homogeneity tensor, Equation 5.13.
2. Multiply the first part of the system by vector and the second part by tensor test functions, integrate over the elements of the domain and use integration by parts.
3. Replace the trial and test functions by discrete functions belonging to selected functional spaces.
4. In order to eliminate the extra unknowns, set the tensor test functions as equal to the broken gradient of the vector test functions, perform a second integration by parts and substitute the resulting second part of the system into the first part to obtain the element-based primal formulation.
5. Finally, transfer the element-based formulation into a face-based formulation by using mean and jump operators such that each interior face occurs only once.

Applying this procedure to the problem Equation 2.51, the semi-discretized weak form to be solved reads: find the unique function $\mathbf{u} \in \mathcal{U}_p$ such that, for all test functions $\mathbf{w} \in \mathcal{U}_p$, we have

$$\hat{\mathcal{N}}(\mathbf{u}, \mathbf{w}) = \mathcal{N}^{\text{tp}}(\mathbf{u}, \mathbf{w}) + \hat{\mathcal{N}}^{\text{sp}}(\mathbf{u}, \mathbf{w}) = 0 \quad \forall \mathbf{w} \in \mathcal{U}_p\,, \qquad [5.15]$$

with the test functions $\mathbf{w}$ taken from the same space as the trial functions $\mathbf{u}$ (since we consider a Galerkin method), i.e. they can also be expanded as

$$\mathbf{w}(\mathbf{x}) = \sum_{m=1}^{N_T} \mathbf{w}_m(\mathbf{x}) = \sum_{m=1}^{N_T} \sum_{n=1}^{N_n} \phi_{m,n}(\sigma_m(\boldsymbol{\zeta}))\mathbf{W}_{m,n,-} \,. \tag{5.16}$$

The contribution of the temporal term to the weak form is given by

$$\mathcal{N}^{\mathrm{tp}}(\mathbf{u}, \mathbf{w}) \equiv \sum_{m \in \mathbb{T}} \mathcal{T}_m(\mathbf{u}_m, \mathbf{w}_m) \,, \tag{5.17}$$

with

$$\mathcal{T}_m(\mathbf{u}_m, \mathbf{w}_m) = \int_{\mathbb{T}_m} \mathbf{w}_m \cdot \frac{\partial \mathbf{u}_m}{\partial t} \, \mathrm{d}\mathbf{x} \,. \tag{5.18}$$

The temporal discretization of this term is covered in Section 10.1. The DG discretization of the spatial fluxes of the Navier-Stokes equations, $\hat{\mathcal{N}}^{\mathrm{sp}}$, is expressed as

$$\begin{aligned} \hat{\mathcal{N}}^{\mathrm{sp}}(\mathbf{u}, \mathbf{w}) \equiv & \sum_{m \in \mathbb{T}} \mathcal{U}_m(\mathbf{u}_m, \mathbf{w}_m) \\ & + \sum_{e \in \mathbb{E}} \hat{\mathcal{E}}_e(\mathbf{u}^+_{m(e)}, \mathbf{u}^-_{m(e)}, \mathbf{w}^+_{m(e)}, \mathbf{w}^-_{m(e)}) \\ & + \sum_{b \in \mathbb{B}} \hat{\mathcal{B}}_b(\mathbf{u}^+_{m(b)}, \mathbf{w}^+_{m(b)}) \,. \end{aligned} \tag{5.19}$$

The different contributions to Equation 5.19 are given by:

$$\begin{aligned} \mathcal{U}_m(\mathbf{u}_m, \mathbf{w}_m) = & \int_{\mathbb{T}_m} \nabla \mathbf{w}_m : \left[-\underline{\boldsymbol{F}}^c(\mathbf{u}_m) + \underline{\boldsymbol{F}}^d(\mathbf{u}_m, \nabla \mathbf{u}_m) \right] \mathrm{d}\mathbf{x} \\ & - \int_{\mathbb{T}_m} \mathbf{w}_m \cdot \mathbf{S}(\mathbf{u}_m, \nabla \mathbf{u}_m) \, \mathrm{d}\mathbf{x} \,, \end{aligned} \tag{5.20}$$

$$\begin{aligned} \hat{\mathcal{E}}_e(\mathbf{u}^+_{m(e)}, \mathbf{u}^-_{m(e)}, \mathbf{w}^+_{m(e)}, \mathbf{w}^-_{m(e)}) = & \int_{\mathbb{E}_e} \Big[\Big(\hat{\underline{\boldsymbol{F}}}^c_{\mathbb{E}}(\mathbf{u}^+_e, \mathbf{u}^-_e) \\ & - \hat{\underline{\boldsymbol{F}}}^d_{\mathbb{E}}(\mathbf{u}^+_e, \nabla \mathbf{u}^+_e, \mathbf{u}^-_e, \nabla \mathbf{u}^-_e) \Big) \cdot \mathbf{n}_e \Big] \cdot (\mathbf{w}^+_e - \mathbf{w}^-_e) \, \mathrm{d}s \\ & + \int_{\mathbb{E}_e} (\hat{\mathbf{u}}_{\mathbb{E}}(\mathbf{u}^+_e, \mathbf{u}^-_e) - \mathbf{u}^+_e) \cdot \left[\mathbf{n}_e \cdot \underline{\boldsymbol{G}}^T(\mathbf{u}^+_e) \nabla \mathbf{w}^+_e \right] \\ & - (\hat{\mathbf{u}}_{\mathbb{E}}(\mathbf{u}^+_e, \mathbf{u}^-_e) - \mathbf{u}^-_e) \cdot \left[\mathbf{n}_e \cdot \underline{\boldsymbol{G}}^T(\mathbf{u}^-_e) \nabla \mathbf{w}^-_e \right] \mathrm{d}s \,, \end{aligned} \tag{5.21}$$

$$\begin{aligned} \hat{\mathcal{B}}_b(\mathbf{u}^+_{m(b)}, \mathbf{w}^+_{m(b)}) = & \int_{\mathbb{B}_b} \Big[\Big(\hat{\underline{\boldsymbol{F}}}^c_{\mathbb{B}}(\mathbf{u}^+_b, \mathbf{u}^{\mathrm{bnd}}_b) \\ & - \hat{\underline{\boldsymbol{F}}}^d_{\mathbb{B}}(\mathbf{u}^+_b, \nabla \mathbf{u}^+_b, \mathbf{u}^{\mathrm{bnd}}_b, \nabla \mathbf{u}^{\mathrm{bnd}}_b) \Big) \cdot \mathbf{n}_b \Big] \cdot \mathbf{w}^+_b \, \mathrm{d}s \\ & + \int_{\mathbb{B}_b} (\hat{\mathbf{u}}_{\mathbb{B}}(\mathbf{u}^+_b, \mathbf{u}^{\mathrm{bnd}}_b) - \mathbf{u}^+_b) \cdot \left[\mathbf{n}_b \cdot \underline{\boldsymbol{G}}^T(\mathbf{u}^+_b) \nabla \mathbf{w}^+_b \right] \mathrm{d}s \,. \end{aligned} \tag{5.22}$$

In Equation 5.21 and Equation 5.22, because of the discontinuous solution approximation, the flux terms are not uniquely defined across element faces

and boundaries and must therefore be replaced by numerical flux functions identified by $\hat{\bullet}$. Those fluxes ensure weak inter-elements coupling and should be specified such that the resulting method will satisfy three fundamental properties: conservativity, consistency and stability. The choice of the convective and diffusive numerical fluxes and their link with the above-mentioned properties is detailed in Section 5.2.3. Another important remark to make about the face (resp. boundary) contributions is that they not only depend on the traces of the solutions $\mathbf{u}_e^+$ and $\mathbf{u}_e^-$ (resp. $\mathbf{u}_b^+$) but on the solutions defined over the entire adjacent elements to the face, $\mathbf{u}_{m(e)}^+$ and $\mathbf{u}_{m(e)}^-$ (resp. $\mathbf{u}_{m(b)}^+$). This dependence comes from the presence of the diffusive fluxes. Indeed, contrary to the Lagrange functions, the restriction of their gradients on a face, $\nabla\mathbf{u}_e^+$ and $\nabla\mathbf{u}_e^-$ (resp. $\nabla\mathbf{u}_b^+$), depends on all the control points in the element, not only those on the face (resp. boundary).

5.2.3 Ensuring stability of the discrete form through the definition of numerical fluxes

Convective numerical fluxes

The hyperbolic part of the Navier-Stokes equations, commonly referred as the Euler equations, is characterized by an eigenstructure composed by real eigenvalues. Those eigenvalues are associated to eigenvectors defining characteristic directions along which information propagates. For the Euler equations, this "information" takes the form of differential relations specifying the constant physical quantities, called the Riemann invariants, carried along the characteristic directions[3]. From a numerical point of view, it makes sense to think about a numerical flux function that would be consistent with the physics and mathematics embodied in the PDE, i.e. such that the discretization is performed according to the direction of propagation of the information. This has led to the development of upwind schemes which have been shown to be conditionally stable, contrary to central or downwind schemes which are unconditionally unstable (Toro (2009, Sec. 5.1)). The first scheme of this kind designed for linear hyperbolic scalar equations is the *Courant-Isaacson-Rees* (CIR) scheme presented in Courant et al. (1952). The idea presented in Godounov (1959) to build a scheme for non-linear equations is to regard the discontinuous solution at each element faces and at each time level as the initial values of a one-dimensional Riemann problem along the face normal direction. Exploiting the waves structure, the Godunov's scheme achieves the required upwinding

[3]It is interesting to note that the non-linearities contained in the hyperbolic part of the equations is responsible for the apparition of shock waves and expansion fans by the processes of "wave steepening" and "wave-easing" described in Toro (2009, Sec. 2.4.2). The non-linearities make the characteristic speeds dependent on the solution itself, which means that some characteristics will travel faster than others. In particular, for a convex flux function, waves emanating from regions with higher initial values of the state variables will travel faster than the ones starting from regions with lower values. Since the characteristic waves carry the initial data, some flow area will tend to have narrower regions with increasing field values as time evolves, ultimately leading to the formation of a shock discontinuity.

criterion. Seeking for an exact solution of the Riemann problem might be not necessary because the details of the exact solution are not preserved after one time step or it might not even be possible because a similarity solution may not exist. That is why several approximation procedures for solving the Riemann problem have been proposed, e.g. by Roe (1981) (Roe), Osher and F. Solomon (1982) (Osher) or Einfeldt (1988) (HLLE). The main difference among these approaches lies in the way they approximate the exact wave structure by more tractable wave configurations. The numerical methods derived from this approach is often referred to as *Flux Difference Splitting* (FDS). Another useful class of convective numerical flux functions is known as *Flux Vector Splitting* (FVS), a category into which fall schemes like the ones of Van Leer (1982) (van Leer) or Steger and Warming (1981) (Steger-Warming). The basic idea of the FVS approach is that the flux at the element face collects contributions from both sides on the basis of upwinding, in the spirit of the CIR method, by using the eigenvalues of the system. The identification of the upwind directions is thus achieved with less effort than in Godunov-type methods. A downside of the FVS schemes is an increased numerical diffusion compared to FDS. The last family of upwind schemes, called the *Advection Upstream Splitting Method* (AUSM) and first proposed by Meng-Sing Liou and Steffen Jr (1993), combines the efficiency and robustness of FVS and the accuracy of FDS splittings, simultaneously eliminating their weaknesses. The underlying idea of the scheme is based on the recognition that the vector of convective fluxes consists of two physically distinct parts, namely the convective and the pressure terms. Two distinct splitting procedures of the Euler flux have been proposed, namely the Liou-Steffen (Meng-Sing Liou and Steffen Jr (1993)) and the Zha-Bilgen splittings (Zha and Bilgen (1993)), which consist in writing the normal convective flux as

$$\underline{\boldsymbol{F}}^c \cdot \mathbf{n}_e = \mathbf{F}_m^c + \mathbf{F}_p^c \,, \tag{5.23}$$

where, for the Liou-Steffen splitting,

$$\mathbf{F}_m^c = \rho\, v_n \begin{bmatrix} 1 \\ v_x \\ v_y \\ H \end{bmatrix}, \quad \mathbf{F}_p^c = \begin{bmatrix} 0 \\ p\, n_{e,x} \\ p\, n_{e,y} \\ 0 \end{bmatrix}, \tag{5.24}$$

while for the Zha-Bilgen splitting,

$$\mathbf{F}_m^c = \rho\, v_n \begin{bmatrix} 1 \\ v_x \\ v_y \\ E \end{bmatrix}, \quad \mathbf{F}_p^c = \begin{bmatrix} 0 \\ p\, n_{e,x} \\ p\, n_{e,y} \\ p\, v_n \end{bmatrix}. \tag{5.25}$$

In this work, if not stated otherwise, we will use AUSM-type schemes to specify the convective numerical flux function because of the above-mentioned advantages this approach offers compared to FDS and FVS and for the ease with which the schemes belonging to this category can be extended to general EOS (see Section 5.3.3) and low-Mach number regimes (see Section 5.3.4). We will also restrict ourselves to schemes based on the Liou-Steffen splitting even though Qu et al. (2014) show that AUSM-type schemes based on the Zha-Bilgen splitting could be at least as performant.

The numerical flux function involved in the internal face convective term will be noted $\mathcal{H}$ such that

$$\hat{\underline{\boldsymbol{F}}}^c_{\mathbb{E}} \cdot \mathbf{n}_e = \mathcal{H}(\mathbf{u}_e^+, \mathbf{u}_e^-, \mathbf{n}_e) \quad \text{on } \mathbb{E}. \tag{5.26}$$

The boundary conditions are imposed by providing an appropriate outside solution $\mathbf{u}^{\text{bnd}}(\mathbf{u}^+)$ (consistent with the prescribed Dirichlet boundary conditions) to the Riemann solver:

$$\hat{\underline{\boldsymbol{F}}}^c_{\mathbb{B}} \cdot \mathbf{n}_b = \mathcal{H}(\mathbf{u}_b^+, \mathbf{u}_b^{\text{bnd}}, \mathbf{n}_b) \quad \text{on } \mathbb{B}. \tag{5.27}$$

In addition to being consistent and conservative, the numerical flux must also be an E-flux, i.e. it must satisfy for a system of equations the following property presented by Hou and X.-D. Liu (2007):

$$(\mathcal{H}(\mathbf{u}_e^+, \mathbf{u}_e^-, \mathbf{n}_e) - \underline{\boldsymbol{F}}^c_{\mathbb{E}}(\mathbf{w}) \cdot \mathbf{n}_e) \cdot (\mathbf{u}_e^+ - \mathbf{u}_e^-) \geq 0 \quad \forall \mathbf{w} = \mathbf{u}_e^+ + \theta(\mathbf{u}_e^- - \mathbf{u}_e^+),\ 0 \leq \theta \leq 1.$$

This flux property allows to prove that DG schemes satisfy a discrete local cell entropy inequality which in turn is used to demonstrate the nonlinear stability in L_2 norm of DG schemes with arbitrary high order of accuracy. The proof for the scalar case can be found in Jiang and Shu (1994) and its subsequent extensions to systems in Hou and X.-D. Liu (2007). As remarked by Dumbser et al. (2014), this makes DG schemes by nature very robust and appropriate for the solution of nonlinear hyperbolic conservation laws. On the other hand, for hyperbolic conservation law systems, having the positivity-preserving property of some important quantities[4] might be a key feature of the scheme as this would make the scheme L_∞ stable for such quantities, avoiding any risks of nonlinear instabilities and blowups of the code because of the PDE is ill-conditioned. Unfortunately, most commonly used high order numerical schemes, including DG, do not in general satisfy such property automatically. That is why DG methods often needs some sort of nonlinear limiting to avoid the Gibbs phenomenon at shock waves or other discontinuities, as noted by Shu (2009). Those considerations go outside the scope of the present work in which E-flux for the convective numerical flux function will be employed

[4] For scalar problems, the convergence of the solution to the unique physically realistic weak solution satisfying the entropy inequality implies the design of schemes which comply with the maximum principle. As noted by Xiangxiong Zhang and Shu (2011), the entropy solutions in general do not satisfy the maximum principle for system of equations. The preservation of the positivity of some important physical quantities is considered instead.

but without any limiting strategies or strong stability preserving (SSP) time integration schemes. The interested reader can consult e.g. Xiangxiong Zhang (2017).

Finally, let us remark that allowing inter-element discontinuities in the solution is a key element of DG methods to handle convective effects. Indeed, the discontinuous solution representation offers the possibility to rely on upwind fluxes to weakly impose continuity across the element faces, which has a stabilizing effect on the resulting discretization. In comparison, the stabilization of continuous FE methods is made more complex by the need to explicit add diffusion in streamline direction in the weak formulation. In the popular Streamline Upwind Petrov-Galerkin (SUPG) method, this implies taking trial and test functions from different discrete function spaces.

Diffusive numerical fluxes

In contrast to the convective term for which the use of upwind schemes is the only reasonable solution, there exists several methods to discretize the diffusive terms. A unified analysis of the different possibilities is given by Arnold et al. (2002). The most simple choice would be to specify the numerical fluxes $\widehat{\underline{\boldsymbol{F}}}^d$ and $\hat{\mathbf{u}}$ as the mean between left and right states[5]. Although consistent, this formulation is known to be unstable. To stabilize the formulation, Hillewaert (2023) identifies two main classes of methods available:

- Interior penalty methods (IPM) can be seen as a generalization of continuous FEM applied to each element individually and in which the imposition of the Dirichlet boundary condition on the continuity of the solution between the elements is realized via Nitsche's method. The variants of IPM are the Symmetric (SIPG), the Non-symmetric (NIPG) and the Incomplete (NIPG) formulations.
- Lifting methods start from the mixed formulation to obtain a system of first-order differential equations and introduce an additional auxiliary variable to estimate the solution gradient. This auxiliary variable is then used in a second step to discretize the elliptic problem. A well-known lifting method is the Bassy-Rebay method (BR1 and BR2) by F. Bassi and S. Rebay (1997) and Francesco Bassi et al. (1997) and its generalization, the Local DG (LDG) method, by Cockburn and Shu (1998).

In this thesis, the Interior Penalty (IP) method is selected. The approach extends Nitsche's method for weak Dirichlet boundary conditions imposition to the weak enforcement of solution inter-element continuity. The idea is that instead of a rough and direct imposition, a penalization term $\underline{\boldsymbol{\delta}}$ is consistently added to the discretization to minimize the average interpolation errors between $\mathbf{u}^+$ and $\mathbf{u}^-$ across element faces and $\mathbf{u}^+$ and $\mathbf{u}^{\text{bnd}}$ on domain boundaries. The numerical

[5] Indeed, in contrast to convective effects which have a "preferred" direction and require upwinding, viscous effects are generally isotropic.

flux function $\hat{\underline{\boldsymbol{F}}}^d_{\mathbb{E}}$ on element faces is then expressed as

$$\hat{\underline{\boldsymbol{F}}}^d_{\mathbb{E}} = \frac{1}{2}\left(\underline{\boldsymbol{F}}^d(\mathbf{u}^+_e, \nabla\mathbf{u}^+_e) + \underline{\boldsymbol{F}}^d(\mathbf{u}^-_e, \nabla\mathbf{u}^-_e)\right) - \underline{\boldsymbol{\delta}}_{\mathbb{E}}(\mathbf{u}^+_e, \mathbf{u}^-_e) \quad \text{on } \mathbb{E}\,, \tag{5.28}$$

in which the penalization term is either scaled with a scalar diffusivity or with a tensorial one according to Ralf Hartmann and Houston (2008):

$$\underline{\boldsymbol{\delta}}_{\mathbb{E}}(\mathbf{u}^+_e, \mathbf{u}^-_e) = \gamma_{\mathbb{E}} \begin{cases} \max\left(\beta^+\,,\, \beta^-\right)(\mathbf{u}^+_e - \mathbf{u}^-_e) \otimes \mathbf{n}_e\,, & \text{if standard IP,} \\ 1/2\left[\underline{\boldsymbol{G}}(\mathbf{u}^+_e) + \underline{\boldsymbol{G}}(\mathbf{u}^-_e)\right](\mathbf{u}^+_e - \mathbf{u}^-_e) \otimes \mathbf{n}_e\,, & \text{if tensorial IP,} \end{cases} \tag{5.29}$$

where $\beta = (2\mu + \lambda/C_v)$. Although more complex to put in place than the scalar penalty, the use of the homogeneity tensor to scale the solution jumps allows taking into account the distinct physical nature of each process in the equations. This is not the case with the scalar penalty which penalizes all components in an equal manner. The value of the stabilization parameter $\gamma_{\mathbb{E}}$ determines the stability of the IP method. More precisely, a lower-bound on this parameter must be specified such that the resulting form will be coercive, hence ensuring inf-sup stability of the method. Selecting a too large value would however negatively affect the conditioning of the resulting system. An optimal choice of this parameter needs therefore to be derived, at least for the Symmetric (SIPG) and Incomplete (IIPG) methods. In Hillewaert (2013), bounds on γ for different element and mesh types are derived from trace inverse inequalities. For two-dimensional triangular and quadrilateral elements, those are given by

$$\begin{aligned} \gamma_{\mathbb{E}} &= \max\left(\frac{s \times C^+_{IP}}{h^+}\,,\, \frac{s \times C^-_{IP}}{h^-}\right), \\ C_{IP} &= \begin{cases} (p+1)(p+2)/2 & \text{for 2-D tri}\,, \\ (p+1)^2 & \text{for 2-D quad}\,, \end{cases} \\ h &= \frac{|\mathbb{T}_m|}{|\mathbb{E}_e|} = \frac{\text{“Element surface”}}{\text{“Face length”}}\,, \end{aligned} \tag{5.30}$$

with s, a user-defined safety factor. For the SIPG method (which is the by-default method used in this work), the numerical flux function on the solution at the faces $\hat{\mathbf{u}}_{\mathbb{E}}$ is taken as the mean value

$$\hat{\mathbf{u}}_{\mathbb{E}} = \frac{1}{2}(\mathbf{u}^+_e + \mathbf{u}^-_e) \quad \text{on } \mathbb{E}\,. \tag{5.31}$$

Concerning the boundary terms, Neumann boundary conditions are imposed by replacing the average diffusive flux in the first term of $\hat{\underline{\boldsymbol{F}}}^d_{\mathbb{B}}$ by the desired boundary fluxes $\underline{\boldsymbol{F}}^{d,\text{bnd}}$ consistent with the prescribed Neumann boundary conditions of the original problem, while Dirichlet boundary conditions are enforced by providing the appropriate outside state $\mathbf{u}^{\text{bnd}}$ to the penalty term

and replacing the mean solution flux by $\mathbf{u}^{\text{bnd}}$:

$$\hat{\underline{\boldsymbol{F}}}^{d}_{\mathbb{B}} = \underline{\boldsymbol{F}}^{d,\text{bnd}}(\mathbf{u}_b^+, \nabla\mathbf{u}_b^+) - \underline{\boldsymbol{\delta}}_{\mathbb{B}}(\mathbf{u}_b^+, \mathbf{u}_b^{\text{bnd}}) \quad \text{on } \mathbb{B}\,, \tag{5.32}$$

$$\hat{\mathbf{u}}_{\mathbb{B}} = \mathbf{u}_b^{\text{bnd}} \quad \text{on } \mathbb{B}\,. \tag{5.33}$$

The penalty parameter $\gamma_{\mathbb{B}}$ on the boundaries only depend on the interior state. In addition to provide stabilization of the scheme, it can be proved that the specified numerical fluxes are both consistent and conservative (see Arnold et al. (2002)), as desired.

5.2.4 Assembly of the variational form

Introduction of the shape functions into the weak form

Equation 5.15 to Equation 5.22 are not exactly the formulation implemented in practice. The unknowns of the discretized problem are the nodal degrees of freedom $\underline{\boldsymbol{U}}_{-,-,-}(t)$ at every time step. The nodal expansion weights of the test function, $\underline{\boldsymbol{W}}_{-,-,-}$, are not of interest and must be removed from the formulation. This is done by remarking that Equation 5.15 must be satisfied for any element and for any function $\mathbf{w}$. Since within each element, $\mathbf{w}_m$ is a linear combination of the shape functions, it suffices to test the residual for all shape functions on each element, i.e. Equation 5.15 becomes

$$\sum_{m\in\mathbb{T}} \int_{\mathbb{T}_m} \phi_{m,n} \cdot \frac{\partial}{\partial t}\left(\boldsymbol{\phi}_{m,-}^T \underline{\boldsymbol{U}}_{m,-,-}\right) \mathrm{d}\mathbf{x} + \mathcal{N}^{\text{sp}}(\underline{\boldsymbol{U}}_{-,-,-}) = 0 \quad \forall n \in \{1, \dots, N_n\}\,, \tag{5.34}$$

which is equivalent to the ordinary differential (ODE) system of $(N_T \times N_n) \times N_f$ equations,

$$\underbrace{\sum_{m\in\mathbb{T}} \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}\boldsymbol{\phi}_{m,-}^T \,\mathrm{d}\mathbf{x}}_{\underline{\boldsymbol{M}}} \frac{\mathrm{d}\underline{\boldsymbol{U}}_{-,-,-}}{\mathrm{d}t} + \mathcal{N}^{\text{sp}}(\underline{\boldsymbol{U}}_{-,-,-}) = 0\,, \tag{5.35}$$

where $\underline{\boldsymbol{M}}$ is the mass matrix. The proof of the equivalence between Equation 5.15 and Equation 5.34 is provided in Matthew J. Zahr (2021, Sec. 3.4). The assembly of the spatial residual, $\mathcal{N}^{\text{sp}}$, then follows three steps:

- The substitution of the test functions $\mathbf{w}$ in Equation 5.20, Equation 5.21 and Equation 5.22 by the shape functions $\boldsymbol{\phi}$ such that $\mathcal{N}^{\text{sp}}(\underline{\boldsymbol{U}}_{-,-,-})$ becomes a sole function of the unknown degrees of freedom.
- The replacement of the undefined face and boundary fluxes in Equation 5.21 and Equation 5.22 by the convective and diffusive numerical flux functions discussed in Section 5.2.3. Note that the spatial residual is here noted $\mathcal{N}^{\text{sp}}$ and not $\hat{\mathcal{N}}^{\text{sp}}$ to indicate this replacement.
- The spatial integration of the different terms. This is covered below.

Let us note that the derivation of the first term in Equation 5.35 assumes that the equations are solved for the set of conservative variables. The adaptation of this term to other set of solved-for variables is discussed in Chapter 10, along with the temporal discretization to transform Equation 5.35 from an ODE system to an algebraic system of equations.

Quadrature

All the terms present in the variational form, considered per element, per face or per boundary, could be put under the following form:

$$\int_S \underline{\boldsymbol{F}}(\mathbf{u}(\mathbf{x}))\, \mathrm{d}S\,, \tag{5.36}$$

where S is the entity on which to integrate (e.g. an element or a face of the mesh), $\mathbf{x}$ is the position vector in physical space and $\underline{\boldsymbol{F}} \in \mathbb{R}^{N_n \times N_f}$ is a function of the approximate solution vector $\mathbf{u}$ defined on all the interpolation points of the entity, for each physical field.

In order to evaluate Equation 5.36, we need to define a quadrature rule, i.e. find a set of weights and points within the domain of integration such that the integral can be approximated by

$$\int_S \underline{\boldsymbol{F}}(\mathbf{u}(\mathbf{x}))\, \mathrm{d}S \approx \sum_{q=1}^{N_q} \omega_q \underline{\boldsymbol{F}}(\mathbf{u}(\mathbf{x}_q))\,, \tag{5.37}$$

with the sets of N_q quadrature weights and nodes,

$$\begin{aligned} \boldsymbol{\omega} &= \{\omega_1, \dots, \omega_q, \dots, \omega_{N_q}\}\,, \\ \mathbf{x}^Q &= \{\mathbf{x}_1^Q, \dots, \mathbf{x}_q^Q, \dots, \mathbf{x}_{N_q}^Q\}\,. \end{aligned} \tag{5.38}$$

In this work, we rely on Gauss-Legendre quadrature rules providing an exact integration for polynomial functions of order $2N_q - 1$ and having an order $o_q = 2N_q$. The position of the nodes are expressed in the reference coordinates and found as the roots of Legendre polynomials of degree N_q, which are precomputed and stored. The integral on the physical entity is expressed in reference coordinates using the determinant of the Jacobian of the mapping, Equation 5.8. The quadrature order of accuracy is defined in function of the polynomial order of interpolation p. As a minimum, we select $o_q = 2p + 1$ such that the product of two functions of order p (which can be found in the variational form) are accurately integrated. Nevertheless, as noted in Hillewaert (2023, Sec. 2.7), the integrals of the weak form in our case will never be exact since the integrand is not a polynomial function, for two reasons: (i) the nonlinear dependence of the fluxes on the solution which is itself a polynomial of order p, (ii) the inverse of the Jacobian mapping which is a rational function in the most general case. Hopefully, Pazner and Haut (2021) note that the quadrature error itself is

high-order, and does not pose a barrier to the high-order convergence of the method.

Assembly of the residuals

Volume residuals To illustrate the assembly of the volume residuals, we focus on the diffusive flux. The assembly of the contribution of the convective flux is identical (even easier since the gradient of the solutions is not needed) while the one of the source term is similar to the assembly of the face and boundary residuals. In an element $\mathbb{T}_m$, the volume residual is computed as

$$\mathcal{V}_m(\mathbf{u}_m, \nabla\mathbf{u}_m) = \int_{\mathbb{T}_m} \nabla\boldsymbol{\phi}_{m,-} \cdot \underline{\boldsymbol{F}}(\mathbf{u}_m, \nabla\mathbf{u}_m)\, \mathrm{d}\mathbf{x}\,. \tag{5.39}$$

The integration through Equation 5.37 is performed in three steps, according to Hillewaert (2013, Sec. 5.3):

1. *Collocation*: the solution is interpolated to the positions of the quadrature points in the reference element, $\boldsymbol{\xi}^Q$, via

$$\mathbf{u}_m(\boldsymbol{\xi}^Q) = \begin{bmatrix} \phi_{m,1}(\boldsymbol{\xi}_1^Q) & \dots & \phi_{m,N_n}(\boldsymbol{\xi}_1^Q) \\ \vdots & & \vdots \\ \phi_{m,1}(\boldsymbol{\xi}_{N_q}^Q) & \dots & \phi_{m,N_n}(\boldsymbol{\xi}_{N_q}^Q) \end{bmatrix} \cdot \underbrace{\begin{bmatrix} U_{m,1,1} & \dots & U_{m,1,N_f} \\ \vdots & & \vdots \\ U_{m,N_n,1} & \dots & U_{m,N_n,N_f} \end{bmatrix}}_{\underline{\boldsymbol{U}}_{m,-,-}}. \tag{5.40}$$

   The parametric gradients of the shape functions in parametric coordinates, both not including and including the integration weights, are stored as

$$\nabla_{\boldsymbol{\xi}}\boldsymbol{\phi}_m = \left[\nabla_{\xi_1}\boldsymbol{\phi}_m, \nabla_{\xi_2}\boldsymbol{\phi}_m\right], \quad \nabla_{\boldsymbol{\xi}}\boldsymbol{\phi}_m^* = \left[\nabla_{\xi_1}\boldsymbol{\phi}_m^*, \nabla_{\xi_2}\boldsymbol{\phi}_m^*\right], \tag{5.41}$$

   with

$$\nabla_{\xi_i}\boldsymbol{\phi}_m = \begin{bmatrix} \frac{\partial\phi_{m,1}}{\partial\xi_i}(\boldsymbol{\xi}_1^Q) & \dots & \frac{\partial\phi_{m,1}}{\partial\xi_i}(\boldsymbol{\xi}_{N_q}^Q) \\ \vdots & & \vdots \\ \frac{\partial\phi_{m,N_n}}{\partial\xi_i}(\boldsymbol{\xi}_1^Q) & \dots & \frac{\partial\phi_{m,N_n}}{\partial\xi_i}(\boldsymbol{\xi}_{N_q}^Q) \end{bmatrix}, \tag{5.42}$$

$$\nabla_{\xi_i}\boldsymbol{\phi}_m^* = \begin{bmatrix} \frac{\partial\phi_{m,1}}{\partial\xi_i}(\boldsymbol{\xi}_1^Q)\,\omega_1\,|\underline{\boldsymbol{J}}^\sigma(\boldsymbol{\xi}_1^Q)| & \dots & \frac{\partial\phi_{m,1}}{\partial\xi_i}(\boldsymbol{\xi}_{N_q}^Q)\,\omega_{N_q}\,|\underline{\boldsymbol{J}}^\sigma(\boldsymbol{\xi}_{N_q}^Q)| \\ \vdots & & \vdots \\ \frac{\partial\phi_{m,N_n}}{\partial\xi_i}(\boldsymbol{\xi}_1^Q)\,\omega_1\,|\underline{\boldsymbol{J}}^\sigma(\boldsymbol{\xi}_1^Q)| & \dots & \frac{\partial\phi_{m,N_n}}{\partial\xi_i}(\boldsymbol{\xi}_{N_q}^Q)\,\omega_{N_q}\,|\underline{\boldsymbol{J}}^\sigma(\boldsymbol{\xi}_{N_q}^Q)| \end{bmatrix}. \tag{5.43}$$

   The solution parametric gradients at the quadrature points can then be computed by

$$\nabla_{\xi_i}\mathbf{u}_m(\boldsymbol{\xi}^Q) = \left(\nabla_{\xi_i}\boldsymbol{\phi}_m\right)^T \cdot \underline{\boldsymbol{U}}_{m,-,-}\,. \tag{5.44}$$

2. *Physical gradients*: the physical gradients of the solution vector evaluated at the quadrature points,

$$\nabla \mathbf{u}_m(\boldsymbol{\xi}^Q) = \left[\frac{\partial \mathbf{u}_m}{\partial x}, \frac{\partial \mathbf{u}_m}{\partial y} \right],$$

can be computed from the parametric gradient, Equation 5.44, and the inverse of the mapping Jacobian, Equation 5.7, as

$$\begin{aligned} \frac{\partial \mathbf{u}_m}{\partial x} &= \sum_{i=0}^{D} \frac{\partial \mathbf{u}_m}{\partial \xi_i} \frac{\partial \xi_i}{\partial x}(\boldsymbol{\xi}^Q) = \sum_{i=0}^{D} \nabla_{\xi_i} \mathbf{u}_m(\boldsymbol{\xi}^Q) \left(\underline{\boldsymbol{J}}^\sigma\right)^{-1}_{i,1}, \\ \frac{\partial \mathbf{u}_m}{\partial y} &= \sum_{i=0}^{D} \frac{\partial \mathbf{u}_m}{\partial \xi_i} \frac{\partial \xi_i}{\partial y}(\boldsymbol{\xi}^Q) = \sum_{i=0}^{D} \nabla_{\xi_i} \mathbf{u}_m(\boldsymbol{\xi}^Q) \left(\underline{\boldsymbol{J}}^\sigma\right)^{-1}_{i,2}. \end{aligned} \quad [5.45]$$

The physical gradients of the shape functions including the integration weights,

$$\nabla \boldsymbol{\phi}^*_{m,-}(\boldsymbol{\xi}^Q) = \left[\frac{\partial \boldsymbol{\phi}^*_{m,-}}{\partial x}, \frac{\partial \boldsymbol{\phi}^*_{m,-}}{\partial y} \right],$$

can be obtained in the same way, i.e.

$$\begin{aligned} \frac{\partial \boldsymbol{\phi}^*_{m,-}}{\partial x} &= \sum_{i=0}^{D} \frac{\partial \boldsymbol{\phi}^*_{m,-}}{\partial \xi_i} \frac{\partial \xi_i}{\partial x}(\boldsymbol{\xi}^Q) = \sum_{i=0}^{D} \nabla_{\xi_i} \boldsymbol{\phi}^*_{m,-}(\boldsymbol{\xi}^Q) \left(\underline{\boldsymbol{J}}^\sigma\right)^{-1}_{i,1}, \\ \frac{\partial \boldsymbol{\phi}^*_{m,-}}{\partial y} &= \sum_{i=0}^{D} \frac{\partial \boldsymbol{\phi}^*_{m,-}}{\partial \xi_i} \frac{\partial \xi_i}{\partial y}(\boldsymbol{\xi}^Q) = \sum_{i=0}^{D} \nabla_{\xi_i} \boldsymbol{\phi}^*_{m,-}(\boldsymbol{\xi}^Q) \left(\underline{\boldsymbol{J}}^\sigma\right)^{-1}_{i,2}. \end{aligned} \quad [5.46]$$

3. *Redistribution*: the last step is to fill in the fluxes based on the physical solution and its gradients evaluated at the quadrature points, i.e. Equation 5.40 and Equation 5.45, respectively. The fluxes are split based on the direction as

$$\underline{\boldsymbol{F}} = \left[\underline{\boldsymbol{F}}_x, \underline{\boldsymbol{F}}_y \right], \quad [5.47]$$

with

$$\underline{\boldsymbol{F}}_i = \begin{bmatrix} F_{i,1}\left(\mathbf{u}_m(\boldsymbol{\xi}^Q_1), \nabla \mathbf{u}_m(\boldsymbol{\xi}^Q_1)\right) & \dots & F_{i,N_f}\left(\mathbf{u}_m(\boldsymbol{\xi}^Q_1), \nabla \mathbf{u}_m(\boldsymbol{\xi}^Q_1)\right) \\ \vdots & & \vdots \\ F_{i,1}\left(\mathbf{u}_m(\boldsymbol{\xi}^Q_{N_q}), \nabla \mathbf{u}_m(\boldsymbol{\xi}^Q_{N_q})\right) & \dots & F_{i,N_f}\left(\mathbf{u}_m(\boldsymbol{\xi}^Q_{N_q}), \nabla \mathbf{u}_m(\boldsymbol{\xi}^Q_{N_q})\right) \end{bmatrix}. \quad [5.48]$$

The contribution of element $\mathbb{T}_m$ to the volume residual is finally given by

$$\mathcal{V}_m(\mathbf{u}_m, \nabla \mathbf{u}_m) \mathrel{+}= \nabla \boldsymbol{\phi}^*_{m,-}(\boldsymbol{\xi}^Q) \cdot \underline{\boldsymbol{F}}(\boldsymbol{\xi}^Q). \quad [5.49]$$

Face and boundary residuals We turn our attention to the assembly of the face and boundary residuals; more precisely to the convective fluxes and the direct and penalty parts of the diffusive fluxes which are all assembled in

the same way. On the other hand, the assembly of the transpose part of the diffusive fluxes differs and is therefore explained just after. We focus on the face contribution here but the boundary contribution is very similar, the main difference being the calls to some additional routines during the assembly to determine the boundary conditions to impose on the solution vector and its gradients. A general expression for the various residuals contributions on the faces is

$$\mathcal{E}_{e,1}(\mathbf{u}^+_{m(e)}, \mathbf{u}^-_{m(e)}) = \pm \int_{\mathbb{E}_e} \boldsymbol{\phi}^{\pm}_{e,-} \cdot (\underline{\boldsymbol{F}}_{\mathbb{E}}(\mathbf{u}^+_e, \nabla\mathbf{u}^+_e, \mathbf{u}^-_e, \nabla\mathbf{u}^-_e) \cdot \mathbf{n}_e)\ \mathrm{d}s\,, \qquad [5.50]$$

where $\boldsymbol{\phi}^{\pm}_{e,-}$ is the vector of shape functions on the face associated to the left or right element and evaluated at the quadrature points on the respective side of the face (the quadrature points might be different between the left and right sides). Thus, $\boldsymbol{\phi}^{\pm}_{e,-}$ is a matrix of size $N_{n,e} \times N_{q,e}$ where $N_{n,e}$ and $N_{q,e}$ are the numbers of interpolation and quadrature points on the face, respectively, making the distinction with the numbers in the element, N_n and N_q. The assembly steps are the same as for the volume residual:

1. *Collocation*: the interpolation of the solution on the face quadrature points is done similarly to Equation 5.40, except that only the shape functions and the nodal degrees of freedom on the face are needed here:

$$\mathbf{u}^{\pm}_e(\boldsymbol{\xi}^Q_e) = \underbrace{\begin{bmatrix} \phi^{\pm}_{e,1}(\boldsymbol{\xi}^Q_{e,1}) & \cdots & \phi^{\pm}_{e,N_{n,e}}(\boldsymbol{\xi}^Q_{e,1}) \\ \vdots & & \vdots \\ \phi^{\pm}_{e,1}(\boldsymbol{\xi}^Q_{e,N_{q,e}}) & \cdots & \phi^{\pm}_{e,N_{n,e}}(\boldsymbol{\xi}^Q_{e,N_{q,e}}) \end{bmatrix}}_{\boldsymbol{\phi}^{\pm}_{e,-}} \cdot \underbrace{\begin{bmatrix} U^{\pm}_{e,1,1} & \cdots & U^{\pm}_{e,1,N_f} \\ \vdots & & \vdots \\ U^{\pm}_{e,N_{n,e},1} & \cdots & U^{\pm}_{e,N_{n,e},N_f} \end{bmatrix}}_{\underline{\boldsymbol{U}}_{e,-,-}}. \qquad [5.51]$$

   To compute the solution parametric gradients at the face quadrature points, it is required to involve the nodal degrees of freedom on the entire left or right element of the face, hence the notation $\mathbf{u}^{\pm}_{m(e)}$ in Equation 5.50 to specify that the residual does not only depend on the solution restricted to the face. Thus, Equation 5.44 is used,

$$\nabla_{\xi_i}\mathbf{u}^{\pm}_e(\boldsymbol{\xi}^Q_e) = \left(\nabla_{\xi_i}\boldsymbol{\phi}_m(\boldsymbol{\xi}^Q_e)\right)^T \cdot \underline{\boldsymbol{U}}_{m,-,-}\,, \qquad [5.52]$$

   where the gradients of the shape functions are only evaluated at the position of the face quadrature points. Similarly to Equation 5.43 for the volume residual, a matrix containing the face shape functions evaluated at the face quadrature points and multiplied by the integration weights

and the determinant of the mapping Jacobian is formed:

$$\boldsymbol{\phi}_e^{*,\pm} = \begin{bmatrix} \phi_{e,1}^{\pm}(\boldsymbol{\xi}_{e,1}^{Q})\,\omega_{e,1}\,|\underline{\boldsymbol{J}}_e^{\sigma}(\boldsymbol{\xi}_{e,1}^{Q})| & \dots & \phi_{e,1}^{\pm}(\boldsymbol{\xi}_{e,N_{q,e}}^{Q})\,\omega_{e,N_{q,e}}\,|\underline{\boldsymbol{J}}_e^{\sigma}(\boldsymbol{\xi}_{e,N_{q,e}}^{Q})| \\ \vdots & & \vdots \\ \phi_{e,N_{n,e}}^{\pm}(\boldsymbol{\xi}_{e,1}^{Q})\,\omega_{e,1}\,|\underline{\boldsymbol{J}}_e^{\sigma}(\boldsymbol{\xi}_{e,1}^{Q})| & \dots & \phi_{e,N_{n,e}}^{\pm}(\boldsymbol{\xi}_{e,N_{q,e}}^{Q})\,\omega_{e,N_{q,e}}\,|\underline{\boldsymbol{J}}_e^{\sigma}(\boldsymbol{\xi}_{e,N_{q,e}}^{Q})| \end{bmatrix}, \tag{5.53}$$

where $\underline{\boldsymbol{J}}_e^{\sigma}$ is the mapping Jacobian between the physical and reference face entity.

2. *Physical gradients*: the computation of the physical gradients of the solution on the face follows exactly Equation 5.45, with the parametric gradients provided by Equation 5.52 and the inverse of the face mapping Jacobian.
3. *Redistribution*: the last step consists in using the solution and its physical gradients at the face quadrature points to compute the fluxes on the face, similarly to Equation 5.48, based on the numerical flux functions described in Section 5.2.3. The contribution of face $\mathbb{E}_e$ to the left and right face residuals is finally given by

$$\mathcal{E}_{e,1}(\mathbf{u}_{m(e)}^{+}, \mathbf{u}_{m(e)}^{-}) \mathrel{+}= \pm\boldsymbol{\phi}_e^{*,\pm}(\boldsymbol{\xi}_e^{Q}) \cdot (\underline{\boldsymbol{F}}_{\mathbb{E}} \cdot \mathbf{n}_e)\,(\boldsymbol{\xi}_e^{Q})\,. \tag{5.54}$$

Diffusive transpose face and boundary residuals The last remaining term in the spatial residual is the diffusive transpose term, present in the SIPG and NIPG discretizations. The general form for this term can be written as (here again using the notation for the face residual but the following can be equivalently applied to the boundary residual):

$$\mathcal{E}_{e,2}(\mathbf{u}_{m(e)}^{+}, \mathbf{u}_{m(e)}^{-}) = \pm \int_{\mathbb{E}_e} (\mathbf{u}_{\mathbb{E}}(\mathbf{u}_e^{+}, \mathbf{u}_e^{-}) - \mathbf{u}_e^{\pm}) \cdot \left[\mathbf{n}_{\mathbf{e}} \cdot \underline{\boldsymbol{G}}^{T}(\mathbf{u}_e^{\pm}) \nabla\boldsymbol{\phi}_{e,-}^{\pm}\right] \mathrm{d}s\,, \tag{5.55}$$

To make this expression more explicit, we decompose the integrand using Equation 5.14 as

$$\begin{aligned} &(\mathbf{u}_{\mathbb{E}}(\mathbf{u}_e^{+}, \mathbf{u}_e^{-}) - \mathbf{u}_e^{\pm}) \cdot \left[\mathbf{n}_{\mathbf{e}} \cdot \underline{\boldsymbol{G}}^{T}(\mathbf{u}_e^{\pm}) \nabla\boldsymbol{\phi}_{e,-}^{\pm}\right] \\ &\qquad = (\mathbf{u}_{\mathbb{E}}(\mathbf{u}_e^{+}, \mathbf{u}_e^{-}) - \mathbf{u}_e^{\pm}) \cdot \left[\mathbf{n}_{\mathbf{e}} \cdot \underline{\boldsymbol{G}}_{x,-}^{T}(\mathbf{u}_e^{\pm}),\ \mathbf{n}_{\mathbf{e}} \cdot \underline{\boldsymbol{G}}_{y,-}^{T}(\mathbf{u}_e^{\pm})\right] \cdot \begin{bmatrix} \nabla_x\boldsymbol{\phi}_{e,-}^{\pm} \\ \nabla_y\boldsymbol{\phi}_{e,-}^{\pm} \end{bmatrix}, \end{aligned} \tag{5.56}$$

where

$$\underline{\boldsymbol{G}}_{x,-}^{T} = \begin{bmatrix} \underline{\boldsymbol{G}}_{xx}^{T} \\ \underline{\boldsymbol{G}}_{xy}^{T} \end{bmatrix}, \quad \underline{\boldsymbol{G}}_{y,-}^{T} = \begin{bmatrix} \underline{\boldsymbol{G}}_{yx}^{T} \\ \underline{\boldsymbol{G}}_{yy}^{T} \end{bmatrix}. \tag{5.57}$$

Moreover, the physical gradients of the shape functions will be expressed as

$$\begin{aligned}\nabla_x \boldsymbol{\phi}^{\pm}_{e,-} &= \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_1}\frac{\partial \xi_1}{\partial x} + \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_2}\frac{\partial \xi_2}{\partial x}\,,\\ \nabla_y \boldsymbol{\phi}^{\pm}_{e,-} &= \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_1}\frac{\partial \xi_1}{\partial y} + \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_2}\frac{\partial \xi_2}{\partial y}\,.\end{aligned} \tag{5.58}$$

Combining Equation 5.57 and Equation 5.58, it is possible to rewrite the right-hand side of Equation 5.56 as

$$\begin{aligned}&(\mathbf{u}_{\mathrm{E}}(\mathbf{u}^{+}_{e}, \mathbf{u}^{-}_{e}) - \mathbf{u}^{\pm}_{e}) \cdot \left[\underline{\boldsymbol{G}}^{T}_{xx}\, n_{e,x} + \underline{\boldsymbol{G}}^{T}_{xy}\, n_{e,y},\, \underline{\boldsymbol{G}}^{T}_{yx}\, n_{e,x} + \underline{\boldsymbol{G}}^{T}_{yy}\, n_{e,y}\right]\\ &\qquad \cdot \begin{bmatrix} \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_1}\frac{\partial \xi_1}{\partial x} + \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_2}\frac{\partial \xi_2}{\partial x} \\ \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_1}\frac{\partial \xi_1}{\partial y} + \frac{\partial \boldsymbol{\phi}^{\pm}_{e,-}}{\partial \xi_2}\frac{\partial \xi_2}{\partial y} \end{bmatrix}\end{aligned} \tag{5.59}$$

The three-step process for the transpose term is then:

1. *Collocation*: the interpolation of the solution on the face quadrature points is done via Equation 5.51. An intermediate data structure noted $\underline{\boldsymbol{G}}^{*}$, with

$$\underline{\boldsymbol{G}}^{*} = \left[\underline{\boldsymbol{G}}^{*}_{x}, \underline{\boldsymbol{G}}^{*}_{y}\right], \tag{5.60}$$

   is initialized with $\underline{\boldsymbol{G}}^{*}_{x}$ and $\underline{\boldsymbol{G}}^{*}_{y}$ containing each $N_{q,e}$ rows and N_f columns. Then, for each quadrature point, the rows of $\underline{\boldsymbol{G}}^{*}_{x}$ and $\underline{\boldsymbol{G}}^{*}_{y}$ are filled by the product of the two first terms of Equation 5.59 including in addition the components of the inverse of the mapping Jacobian present in the last term and the integration weights, i.e.

$$\begin{aligned}\underline{\boldsymbol{G}}^{*}_{x}[q,:] &= \omega_{e,q}\,|\underline{\boldsymbol{J}}^{\sigma}(\boldsymbol{xi}^{Q}_{e,q})|\,\left[\mathbf{u}_{\mathrm{E}} - \mathbf{u}^{\pm}_{e}\right](\boldsymbol{\xi}^{Q}_{e,q})\\ &\quad\cdot\left[\left(\underline{\boldsymbol{G}}^{T}_{xx}\frac{\partial \xi_1}{\partial x} + \underline{\boldsymbol{G}}^{T}_{yx}\frac{\partial \xi_1}{\partial y}\right) n_{e,x} + \left(\underline{\boldsymbol{G}}^{T}_{xy}\frac{\partial \xi_1}{\partial x} + \underline{\boldsymbol{G}}^{T}_{yy}\frac{\partial \xi_1}{\partial y}\right) n_{y,x}\right](\boldsymbol{\xi}^{Q}_{e,q})\,,\\ \underline{\boldsymbol{G}}^{*}_{y}[q,:] &= \omega_{e,q}\,|\underline{\boldsymbol{J}}^{\sigma}(\boldsymbol{xi}^{Q}_{e,q})|\,\left[\mathbf{u}_{\mathrm{E}} - \mathbf{u}^{\pm}_{e}\right](\boldsymbol{\xi}^{Q}_{e,q})\\ &\quad\cdot\left[\left(\underline{\boldsymbol{G}}^{T}_{xx}\frac{\partial \xi_2}{\partial x} + \underline{\boldsymbol{G}}^{T}_{yx}\frac{\partial \xi_2}{\partial y}\right) n_{e,x} + \left(\underline{\boldsymbol{G}}^{T}_{xy}\frac{\partial \xi_2}{\partial x} + \underline{\boldsymbol{G}}^{T}_{yy}\frac{\partial \xi_2}{\partial y}\right) n_{y,x}\right](\boldsymbol{\xi}^{Q}_{e,q})\,.\end{aligned} \tag{5.61}$$

   Note that $\mathbf{u}_{\mathrm{E}}$ is a numerical flux function to specify, as described in Section 5.2.3.
2. *Physical gradients*: the physical gradients of the shape functions (last term in Equation 5.55) is not directly assembled because they will be obtained by multiplying the intermediate tensor $\underline{\boldsymbol{G}}^{*}$, which already contains the components of the inverse Jacobian mapping, with the parametric shape functions gradients. Those will be directly given by Equation 5.42, except that the evaluation will be done on the face quadrature points. Note that all the control points are involved for the shape functions gradients, not only the ones on the face.

3. *Redistribution*: the contribution of face $\mathbb{E}_e$ to the left and right face residuals is finally given by

$$\mathcal{E}_{e,2}(\mathbf{u}^+_{m(e)}, \mathbf{u}^-_{m(e)}) \mathrel{+}= \pm \nabla_{\tilde{\xi}_i} \boldsymbol{\phi}_m(\tilde{\boldsymbol{\xi}}^Q_e) \cdot \underline{\boldsymbol{G}}^*(\tilde{\boldsymbol{\xi}}^Q_e)\,. \tag{5.62}$$

An important remark to make to finish this section is that the contribution of $\mathcal{E}_{e,1}$ to the global residual, $\mathcal{N}^{sp}$, is such that only the values of the global residual corresponding to the interpolation points on the left and right sides of the face $\mathbb{E}_e$ are affected. On the other hand, the contribution of $\mathcal{E}_{e,2}$ is felt by all the interpolation points in the left and right elements, because of the multiplication by the gradient of the face shape functions which is not only dependent on the values of the shape functions restricted to the face.

5.3 Implementation of an all-speed fluid solver

The arguments in favor of modeling both the gas and the condensed phases by means of compressible equations were set out in Section 2.3.1. As pointed out in Denner et al. (2020), despite extensive research efforts over the past 50 years, the development of numerical methods and algorithms that are able to provide accurate and robust simulations of fluid flows with different compressibility and at all speed regimes has proven difficult and is still an active research area. The three main numerical challenges in this quest associated to density-based methods are here addressed to allow the development of an all-speed high-order solver. By "all-speed", we want to stress out that we are not including supersonic regimes and beyond. Although some results that will be presented after are in the low supersonic regime, the numerical treatment of strong moving shock waves in the context of a high-order scheme is beyond the scope of this work and is identified as a natural perspective in Section 13.2.

5.3.1 State-of-the-art overview and numerical challenges

Low-Mach / stiff compressible regime

The potentially high-temperature of the gas flow and the very stiff nature of the condensed phase imply important values of the speed of sound in both phases. Because the flow region targeted as part of this work is the one very closed to the body, the expected velocity of the gas flow at the edge of the boundary layer should be such that the value of the Mach number is at the boundary that traditionally delimits the incompressible and compressible regimes. For the liquid phase, the stiff compressible behavior of the flow is much more pronounced with an expected Mach number of the order of 10^{-4}, as predicted by

our analysis in Section 12.1. To tackle flows falling in this low-Mach regime, two main schools of thought exist: pressure-based and density-based methods[6] (see e.g. Keshtiban et al. (2004), C. Merkle et al. (1992), or Miettinen and Siikonen (2015) for a comparison). On the one hand, pressure-based algorithms like the well-known SIMPLE (Semi-Implicit Pressure Linked Equations) and PISO (Pressure-Implicit with Splitting of Operators) ones in the FVM community (see Barton (1998) for an overview) were originally developed for purely incompressible equations in order to handle, via a segregated approach, the velocity-pressure decoupling difficulty specific to incompressible models. Performance shortcomings of segregated algorithms were addressed by the development of coupled solution strategies, like in Darwish et al. (2009). In FE-type methods, the most prominent approach is the mixed formulation of the variational problem ensuring the satisfaction of the so-called Ladyzhenskaya-Babuška-Brezzi (LBB) condition in order to avoid spurious pressure oscillations (see e.g. Gelhard et al. (2005)). In this respect, it is valuable to mention the efficient implementation strategy of the mixed DG formulation available in `Argo` which is described in Hillewaert (2013, Sec. 3.2). On the other hand, density-based algorithms were conceived to solve in a coupled way the compressible equations where the acoustic perturbations propagate with a finite speed in the computational domain. Both families of methods have been extended to work in the critical nearly incompressible regime.

Pressure-based methods Pressure-based methods can achieve this by different means. One solution is to consider an asymptotic version of the compressible equations from which the pressure can be split into various contributions, each one being associated with different physical response, as done e.g. in Munz (2003). A segregated solution procedure is then applied. Another approach followed e.g. by C.-N. Xiao et al. (2017) is to have a strong coupling between the continuity and momentum by treating the pressure implicitly and expressing the density as a function of the pressure through an EOS. In the FEM world, one can also find methods using asymptotic decomposition of the equations, to be discretized by a mixed FE formulation as in Martinez and Gartling (2004) or by an IMEX splitting as done in Kaiser and Schütz (2017) for a DGM. Other methods exploiting the pressure-correction algorithms of FVM, as in Klein et al. (2016) and Hennink et al. (2021) in a DGM framework, have also been developed.

Density-based methods Regarding the density-based methods, two possibilities exist. The first one is to start from the incompressible equations and relax the incompressibility constraint by adding an artificial compressibility term to the continuity equation. The second option is to consider the compressible equa-

[6] The reason behind the names "pressure-based" and "density-based" is that generally, the methods for incompressible flows are characterized by the use of the pressure as a main dependent variable while in compressible flows, one solves for density and extract the pressure from an equation of state. However, as we will see after, the choice of primary variables is not fixed.

tions and overcome the numerical difficulties by rescaling the eigenvalues of the original system. Preconditioning is the most notable technique in this class of methods and the implementation in our solver is detailed in Section 5.3.4. As this last sentence suggests, we have opted in this work for a stabilized preconditioned density-based strategy rather than an extended pressure-based or artificial compressibility approach. The reasons are threefold:

- *Proven performance*: several studies have demonstrated the ability of high-order density-based DG schemes to simulate flows over a wide range of flow regimes: Renda et al. (2015), R. Nourgaliev et al. (2019) and Weston et al. (2019) to cite only a few. In the context of this thesis, the two last cited work are particularly pertinent. There, an all-speed reconstructed DG solver is shown to be able to perform accurate simulations of melt pool dynamics involving metals.
- *Ease of implementation*: contrary to density-based approaches relying on standard time-marching procedures, pressure-based methods are generally built around more complex algorithms (e.g. the pressure-correction strategies mentioned above). As we will see in Section 5.3.4, the treatment of unsteady equations combined with preconditioning by means of dual-time stepping does not bring any significant additional difficulties compared to the steady-state procedure.
- *Unification with hypersonic flow region*: finally, one of the perspectives of the current work is to simulate not only the flow region in the vicinity of the ablating body but also the upstream shock region. The use of the compressible Navier-Stokes equations in this flow region is customary. Hence, it is preferable to avoid using a different low-Mach version of the governing equations elsewhere in the computational domain. This is another motivation for choosing the preconditioned density-based approach over other ones based on asymptotic equations or artificial compressibility.

Departure from the standard perfect gas law

The use of a density-based solver goes hand in hand with the definition of an equation of state linking density, pressure and temperature. While the classical perfect gas EOS is appropriate for the gas phase, a more advanced EOS modeling the thermodynamic behavior of the liquid phase is required. The existing approaches are reviewed in Section 2.3.2 and the EOS implemented in the solver are compiled in Section 2.4. Comments about the implementation of the various EOS and the way they are used in the assembly of the governing equations are provided here below in Section 5.3.3.

The choice of solved-for variables

From the compressible Navier-Stokes equations written in Section 2.4, the set of conservative variables composed by the density, the linear momentum and

the total energy appears to be the vector of unknowns for which the equations need to be solved. This is however not the only valid choice because the governing equations can be reformulated in theory with respect to any other set of variables, provided that the set includes two independent thermodynamic properties. The choice between one set or another can be influenced by implementation aspects, mathematical properties and numerical considerations. From an implementation perspective, the imposition of boundary conditions most often does not apply to the conservative variables but rather to related quantities like temperature, pressure or velocity. Manipulating other variables than the conservative ones may therefore ease the implementation of certain boundary conditions in the solver. This is also the case to express the gradient of some quantities, whether for computing some fluxes (e.g. the temperature gradient for the heat flux) or for assembling Jacobian matrices. This ease of implementation is for example one of the reasons why the module of `Argo` dealing with high-temperature multispecies reactive flows has been implemented by Pierre Schrooyen (2015) based on the set of pressure, velocity and temperature variables.

Mathematically, the convergence of the compressible Navier-Stokes equations to the incompressible equations as the Mach number tends to zero has been proven rigorously by Hoff (1998), irrespective of the choice of working variables. However, conservative variables represent a poor choice when the Mach number becomes very small. The continuity equation becomes ineffective as a transport equation for the density which is almost independent of pressure variations ($\rho \approx$ const) and the total energy is dominated by internal energy ($0.5\mathbf{v}^2 \ll e_{int}$), as pointed out in R. Nourgaliev et al. (2019). This second observation is all the more pronounced in high-temperature conditions. This behavior translates into ill-conditioned Jacobian matrices of the system which are also difficult to precondition. Using conservative variables for low-Mach flows thus results, from a numerical standpoint, to schemes lacking robustness and in some situations leading to loss of accuracy. This has been highlighted by the detailed comparative study conduced by Hauke and Hughes (1998).

Each set of variables possesses different properties. Conservative variables provide a slightly better conditioned system than other sets at high speeds (R. Nourgaliev et al. (2015)) and more accurate solutions in the presence of shock waves (A. Nigro et al. (2013)). However, it leads to degenerate system of equations in the low-Mach regime, which requires switching to other variables sets. In order to avoid excessively large variations of pressure and internal energy due to small change in density, the natural choice is to replace the density by the pressure variable. To improve the solvability with respect to the total energy variable, it is common to substitute it by temperature, internal energy or enthalpy, as explained in R. Nourgaliev et al. (2015, Sec. 2.2). Although more complicated to be implemented, entropy variables, investigated in Hauke and Hughes (1998) and Chauchat et al. (2021), are also good candidates to achieve a well-posed problem, with the additional possibility to guarantee entropy conservation.

From the recent studies presented by Chauchat et al. (2021) and Carnevali et al. (2023) concerning DG schemes, it seems difficult to identify a set of variables that would be ideal in all respects and for all flow regimes. That is why we have designed our solver such that it can easily accommodate any user-defined set of variables. This is detailed in Section 5.3.2. As a final remark, Denner (2018) notes that choosing a set of solution variables different from the conservative quantities does not affect the conservative properties of the governing equations if a consistent discretization is applied.

5.3.2 Versatile choice of working variables sets

In addition to accommodate arbitrary EOS, the module of the code implementing the compressible Navier-Stokes equations is also designed to handle any user-defined set of working variables. To do so, the distinction is made between the implemented variables set, identified by the vector of unknowns $\mathbf{u}_{IVS}$ in the following, and the working variable set, identified by $\mathbf{u}_{WVS}$. Two routines centralizing the computation of any quantities or gradients of quantities are present, namely

$$\texttt{getStateVar}(\text{setType}, \text{varName}, \mathbf{u}, \textit{tagPhase}) \,,$$
$$\texttt{getGradStateVar}(\text{setType}, \text{varName}, \mathbf{u}, \nabla\mathbf{u}, \textit{tagPhase}) \,,$$

taking as arguments the type of variables set (e.g. conservative, "p–$\mathbf{v}$–T" primitive, etc.), the variable name (e.g. pressure, heat flux, etc.), the solution vector and its gradients, $\mathbf{u}$ and $\nabla\mathbf{u}$, and a tag identifying the phase. Two approaches to deal with general choice of working variables have been included in the code, in chronological order:

- *Approach 1*: the first implemented approach is the most generic one as it assumes that all the physical models in the module are expressed based on a single variable set type; the conservative variables in this case. This means that the two central routines `getStateVar` and `getGradStateVar` will always be called with "setType" identifying the set of conservative variables and $\mathbf{u} = \mathbf{u}_{IVS}$. Thus, the computation of any quantity depending on the solution vector or its gradients can be implemented with respect to a single variable set, facilitating the maintenance of the code and the addition of new variable sets. This comes at the expense of converting $\mathbf{u}_{WVS}$ to $\mathbf{u}_{IVS}$ before each call to `getStateVar`, as well as $\nabla\mathbf{u}_{WVS}$ to $\nabla\mathbf{u}_{IVS}$ before calling `getGradStateVar`. The conversion is performed in two other routines,

  $$\texttt{WVS2IVSVar}(\mathbf{u}_{WVS}) \rightarrow \mathbf{u}_{IVS} \,,$$
  $$\texttt{WVS2IVSGrad}(\mathbf{u}_{WVS}, \nabla\mathbf{u}_{WVS}) \rightarrow \nabla\mathbf{u}_{IVS} \,.$$

  The first one needs to express the conservative variables as a function of the working variables. The second one requires the definition of the

Jacobian matrix of the implemented variables with respect to the working variables such that the gradient of $\mathbf{u}_{IVS}$ can be obtained by the chain rule as

$$\nabla \mathbf{u}_{IVS} = \frac{\partial \mathbf{u}_{IVS}}{\partial \mathbf{u}_{WVS}} \nabla \mathbf{u}_{WVS} \,.$$

In order to add a new working variables set, the only thing to be done is to upgrade the routines `WVS2IVSVar` and `WVS2IVSGrad`. The Jacobian matrix $\frac{\partial \mathbf{u}_{IVS}}{\partial \mathbf{u}_{WVS}}$ may not even be coded, it will be computed by default via finite differences using the routine `WVS2IVSVar`. All the other quantities will be obtained from the conservative variable sets.

- *Approach 2*: the other approach was implemented in a second stage because of the implementation requirements imposed by the derived module featuring the mixture liquid-solid model. In this case, the routines `getStateVar` and `getGradStateVar` will be called with "setType" corresponding to the selected working variables type and $\mathbf{u} = \mathbf{u}_{WVS}$ and there is therefore no need to make use of the conversion routines of the first approach. The price to pay is the implementation of all the flow quantities as a function of the several working variables sets available and not based on a single reference implemented set.

Depending on the working variables set to be used, one approach or the other can be selected. Concerning the assembly of the Jacobians of the fluxes, either no analytical expression is possible in which case it is computed by finite differences, or it is implemented with respect to $\mathbf{u}_{IVS}$ and is converted to the working set using the same transformation Jacobian used in the approach 1:

$$\frac{\partial \underline{\underline{\mathbf{F}}}}{\partial \mathbf{u}_{WVS}} = \frac{\partial \underline{\underline{\mathbf{F}}}}{\partial \mathbf{u}_{IVS}} \frac{\partial \mathbf{u}_{IVS}}{\partial \mathbf{u}_{WVS}} \,.$$

An illustrative example for the assembly of the diffusive fluxes is given below.

Example 5.1: Versatile implementation of different set of variables

```
void diffusiveFlux(u_WVS, gradu_WVS, F_d, tagPhase) {
    // Conversion of working vars to implemented vars if needed
    if (CVS != WVS && approach_1) {
        u = WVS2IVSVar(u_WVS);
        gradu = WVS2IVSGrad(u_WVS, gradu_WVS);
        setType = "Conservative";
    }
    else {
        u = u_WVS;
        gradu = gradu_WVS;
        setType = "WorkingVar";
    }

    // Compute flow quantities required to assemble flux
```

```
    v = getStateVar(setType, "VelocityVector", u, tagPhase);
    tau = getGradStateVar(setType, "StressTensor", u, gradu, tagPhase)
    ↪ ;
    q = getGradStateVar(setType, "HeatFluxVector", u, gradu, tagPhase)
    ↪ ;

    // Assemble diffusive flux
    F_d[0][0] = F_d[0][1] = 0.0; // Continuity
    F_d[1][0] = tau[0][0],  F_d[0][1] = tau[0][1]; // Momentum-x
    F_d[2][0] = tau[1][0],  F_d[2][1] = tau[1][1]; // Momentum-y
    F_d[3][0] = v[0]*tau[0][0] + v[1]*tau[1][0] - q[0]; // Energy
    F_d[3][1] = v[0]*tau[0][1] + v[1]*tau[1][1] - q[1]; // Energy

}
```

Now that we have described how the code is structured to work with various working variables sets, we should pay attention to the continuous form of the governing equations that are originally a set of equations in the unknown conservative variables ρ–$\rho\mathbf{v}$–ρE contained in the vector $\mathbf{u}$. In order to express them in another variables set, noted $\mathbf{w}$, two main approaches can be followed:

- The first one is based on a mapping function between the conservative and the working variables set, $\mathbf{u}(\mathbf{w})$, such that the system in Equation 2.51 is written as
$$\frac{\partial \mathbf{u}(\mathbf{w})}{\partial t} + \mathcal{N}^{sp}(\mathbf{w}) = \mathbf{0}\,, \qquad [5.63]$$
where the spatial part of the Navier-Stokes equations, $\mathcal{N}^{sp}$, can directly be expressed from the working variables,
$$\mathcal{N}^{sp}(\mathbf{w}) \equiv \nabla \cdot \underline{\boldsymbol{F}}^{c}(\mathbf{w}) - \nabla \cdot \underline{\boldsymbol{F}}^{d}(\mathbf{w}, \nabla\mathbf{w}) - \mathbf{S}(\mathbf{w}, \nabla\mathbf{w})\,.$$
This formulation has been adopted e.g. in R. Nourgaliev et al. (2019) and Chauchat et al. (2021). This is also the approach that was already implemented in `Argo` and which is used in this work.
- The second option consists in applying the chain rule to the temporal derivative term, leading to
$$\frac{\partial \mathbf{u}}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial t} + \mathcal{N}^{sp}(\mathbf{w}) = \mathbf{0}\,, \qquad [5.64]$$
where the temporal term is now pre-multiplied by a transformation matrix. This strategy has been chosen by F. Bassi et al. (2015).

At the continuous level, both formulations are perfectly equivalent. However, some notable differences appear during the time discretization process. This is discussed in Section 10.1.3.

5.3.3 Use of general equations of state

The same governing equations are considered for the two phases but their closure are different since each of the phase has its own thermodynamic behavior. The definition of the various equations of state are handled by an internal library, named `LibEOS` separated from the module `GeneralEOS` specifying the governing equations. This internal library is divided in two main child classes : the `SimpleEOS` class implements short analytical EOS such as the perfect gas, the stiffened gas and the van der Waals EOS while the `MieGruneisen` class contains more involved expressions derived from the approach presented in Section 2.3.3. Each derived class needs to provide an implementation for a few abstract methods, namely the expressions of the pressure and internal energy as a function of the two independent density and temperature variables as well as some derivatives that are used to assemble the fluxes Jacobians and the transformation Jacobians to go from one set of variable to another. Those are the methods specific to each EOS to be implemented:

$$
\begin{aligned}
&e(\rho,T)\,,\ p(\rho,T)\,,\\
&\left.\frac{\partial e}{\partial T}\right|_{\rho}(\rho,T)\,,\ \left.\frac{\partial e}{\partial \rho}\right|_{T}(\rho,T)\,,\ \left.\frac{\partial p}{\partial T}\right|_{\rho}(\rho,T)\,,\ \left.\frac{\partial p}{\partial \rho}\right|_{T}(\rho,T)\,.
\end{aligned}
$$

To be noted here is that other derivatives can be obtained from the four derivatives above by using one or combining two of the following mathematical relations that can be found e.g. in Menikoff and Plohr (1989, App. A):

$$
\begin{aligned}
\left.\frac{\partial}{\partial z}\right|_{y} &= \left.\frac{\partial x}{\partial z}\right|_{y}\cdot\left.\frac{\partial}{\partial x}\right|_{y}\,, & \left.\frac{\partial x}{\partial y}\right|_{z}\cdot\left.\frac{\partial y}{\partial x}\right|_{z} &= 1\,,\\
\left.\frac{\partial}{\partial y}\right|_{z} &= \left.\frac{\partial x}{\partial y}\right|_{z}\cdot\left.\frac{\partial}{\partial x}\right|_{y}+\left.\frac{\partial}{\partial y}\right|_{x}\,, & \left.\frac{\partial x}{\partial y}\right|_{z}\cdot\left.\frac{\partial y}{\partial z}\right|_{x}\cdot\left.\frac{\partial z}{\partial x}\right|_{y} &= -1\,.
\end{aligned}
$$

In the mother class `LibEOS`, all the other useful methods are implemented based on the few methods specific to each EOS. For example, the temperature as a function of the density and internal energy is by default obtained from a non-linear equation on the internal energy,

$$
T(\rho,e) : e - e(\rho,T) = 0\,,
$$

which is solved by a Newton-Raphson procedure. Another example is the speed of sound which is computed, irrespective of the EOS, as

$$
c^2 = \left.\frac{\partial p}{\partial \rho}\right|_{e} + \frac{p}{\rho^2}\left.\frac{\partial p}{\partial e}\right|_{\rho}\,,
$$

with the partial derivatives of pressure expressed based on the four specified partial derivatives above, viz.

$$\left.\frac{\partial p}{\partial \rho}\right|_e = \left.\frac{\partial p}{\partial \rho}\right|_T - \left.\frac{\partial p}{\partial e}\right|_\rho \cdot \left.\frac{\partial e}{\partial \rho}\right|_T,$$
$$\left.\frac{\partial p}{\partial e}\right|_\rho = \left.\frac{\partial p}{\partial T}\right|_\rho \cdot \frac{1}{\left.\frac{\partial e}{\partial T}\right|_\rho}.$$

It should be mentioned that for EOS that allow it, it is preferable to override all the mother class methods involving the solution of a non-linear equation such that an explicit analytical expression is directly provided. Indeed, as the EOS routines are called everywhere in the code, avoiding the solution of non-linear equations can prevent some computational overhead, which is all the more marked for the most complex EOS.

For the diffusive fluxes and the Jacobian of the diffusive fluxes with respect to the solution gradients, the spatial gradient of the temperature needs to be evaluated. If the temperature is already part of the working variable set, this is straightforward. If it is not, like e.g. with the conservative variables set, it can be obtained as follows:

$$\mathbf{u} = \left(\rho,\, \rho v_x,\, \rho v_y,\, \rho E\right) = \left(u_1,\, u_2,\, u_3,\, u_4\right),$$
$$T = T(\rho, e) = T\left(u_1, \frac{u_4}{u_1} - 0.5\frac{u_2^2 + u_3^2}{u_1^2}\right) \triangleq T^*(\mathbf{u}),$$
$$\nabla T^* = \left.\frac{\partial T}{\partial \rho}\right|_e \nabla\rho + \left.\frac{\partial T}{\partial e}\right|_\rho \nabla e,$$
$$\nabla\rho = \nabla u_1, \quad \nabla e = \frac{1}{u_1}\left[\nabla u_4 - \left(\frac{u_4}{u_1} - \frac{u_2^2 + u_3^2}{u_1^2}\nabla u_1 - \nabla u_2 \frac{u_2}{u_1} - \nabla u_3 \frac{u_3}{u_1}\right)\right].$$

The derivation of the Jacobians of the convective and diffusive fluxes for both the conservative and primitive p–$\mathbf{v}$–T variables sets is detailed in Appendix A.

An example of how the internal EOS library is used in the module defining the governing equations is given below, detailing the routine `getStateVar` introduced in the previous section.

Example 5.2: Implementation of general EOS

```
// Constructor of the GeneralEOS class implementing the governing equations
GeneralEOS() {
    readInputFile(EOSParams1,EOSParams2);
     ↪  // Read user input file to get the parameters for each phase
    LibEOS** EOSList = new LibEOS*[2];
    EOSList[0] = new SimpleEOS(EOSParams1);
    EOSList[1] = new MieGruneisen(EOSParams2);
}
```

```
// A part of the convective flux assembly routine
void convectiveFlux(u_WVS, F_c, tagPhase) {
    // ...
    p = getStateVar(setType, "Pressure", u, tagPhase);
    // ...
}
// A part of the general routine to compute state variable
double getStateVar(setType,varTag,u,tagPhase) {
    // ...
    if (varTag == "Pressure") {
        if (setType == "Conservative") {
            rho = u[0];
            e_kin = 0.5*(u[1]*u[1]+u[2]*u[2])/(rho*rho);
            e = u[3]/rho - e_kin;
            T = EOSList[tagPhase]->T_rhoe(rho,e);
            p = EOSList[tagPhase]->p_rhoT(rho,T);
            return p;
        }
    }
    // ...
}
```

The idea of defining a list of EOS models that is accessed via a tag associated to the phase under consideration is also followed for the transport properties. Since we consider only constant transport properties, there is no need to build a separated library as it is done for the thermodynamic models.

5.3.4 Treatment of low-Mach number flows by a density-based method

The choice of a density-based solver for both the gas and the liquid phases was motivated in Section 5.3.1. We turn our attention to the practical implementation here. The first step is to further explore the numerical difficulties to be overcome before presenting the remedies.

Numerical difficulties

Convergence issues of density-based methods To reach the steady-state of the steady equations or of the unsteady equations at every time steps (by making use of dual-time stepping), a time-marching procedure relying on implicit time-stepping schemes is used in `Argo`. This is detailed in Section 10.1.2. It is explained in Venkateswaran and L. Merkle (1999) that during this time-marching procedure, the transient errors can be convected out of the domain by convective modes or damped within the domain by diffusive modes. Specifically, the convective (or inviscid) modes are represented by the eigenvalues of the convective flux Jacobian. Those can be determined to be $v_n - c$, v_n and $v_n + c$, where v_n is the local flow velocity normal to the surface across which the flow is considered. When $v_n \ll c$ in low-Mach regime, widely disparate time

scales are involved in the removal of the convective transient errors, resulting in very slow convergence, i.e. a lot of (pseudo-) time steps will be required to convect away the convective modes associated to the fluid velocity. Although implicit schemes are generally considered as unconditionally stable, in practice a (pseudo-) time step chosen too large will make the removal of the errors associated to the particles modes very inefficient. The case of explicit schemes is even worse: because of the time step restriction due to stability constraints, the particle CFL number at low Mach will become even smaller, such that the errors associated to the particle modes will not be removed at all. If the medium considered is such that the speed of sound is high, hence the CFL number more restrictive, the use of explicit schemes turns out to be impracticable, even if they are employed without dual-time stepping for unsteady simulations.

It must be emphasized that the viscous effects can also have an important role on the convergence of the schemes. Venkateswaran and L. Merkle (1999) show that for very small Reynolds numbers, $\mathrm{Re} \ll 1$, some modes of the system involve the viscous part, resulting in the condition number of the system matrix to be dependent on Re. It can be concluded that the flow conditions such that $\mathrm{Ma} \ll 1$ and $\mathrm{Re} \ll 1$ are the most critical in terms of convergence.

Accuracy issues of density-based methods The second issue of density-based methods with low Mach number flows is the accuracy of the discretization. From an asymptotic expansion of the Euler equations, the source of poor accuracy was identified by Guillard and Viozat (1999) to be due to an incorrect scaling of the pressure fluctuations in the discrete equations compared to their continuous counterpart. Guillard and Murrone (2004) show that this wrong scaling is inherent to the 1D Riemann problems considered in Godunov-type schemes. More precisely, it was found that first-order pressure fluctuations at the discrete level for upwind schemes scales with a Mach number one order of magnitude higher than the fluctuations at the continuous level, namely in $O(\mathrm{Ma})$ versus $O(\mathrm{Ma}^2)$. The wrong scaling at the discrete level alters the artificial dissipation of the upwind scheme, resulting in poor accuracy. Since the artificial dissipation is directly connected to the velocity jumps across the element faces, it was noted by Rieper (2011) that high-order methods should be less affected by those accuracy issues. However, as demonstrated by Francesco Bassi et al. (2009) for a DG method, accuracy issues remain a concern for high order schemes in low Mach regimes.

There are two main cures to the two issues identified above: time preconditioning and flux preconditioning.

Time preconditioning

Time preconditioning is meant to alleviate the convergence issues by changing the transient nature of the system in such a way that the disparity between the wave speeds gets significantly reduced. This translates into a reduction of the condition number of the resulting algebraic system, hence the name

"preconditioning" in analogy to the techniques used for iterative linear solvers. Time preconditioning is applied by (i) adding a pseudo-temporal derivative to the Navier-Stokes equations and (ii) by multiplying the pseudo-temporal term by a matrix such that the continuous system to be discretized becomes:

$$\underline{\boldsymbol{P}}_w(\mathbf{u})\frac{\partial \mathbf{w}}{\partial \tau} + \mathcal{N}(\mathbf{w}) = 0\,, \qquad [5.65]$$

where $\mathbf{w}$ is a certain variable set and $\mathcal{N}$ represents the Navier-Stokes system in Equation 2.51, i.e.

$$\mathcal{N}(\mathbf{w}) \equiv \frac{\partial \mathbf{u}(\mathbf{w})}{\partial t} + \nabla \cdot \underline{\boldsymbol{F}}^c(\mathbf{w}) - \nabla \cdot \underline{\boldsymbol{F}}^d(\mathbf{w}, \nabla \mathbf{w}) - \mathbf{S}(\mathbf{w}, \nabla \mathbf{w})\,.$$

Two comments can be made about Equation 5.65:

- The advantage of adding a pseudo-temporal term is that it offers the possibility to alter the transient process through its multiplication with the preconditioning matrix $\underline{\boldsymbol{P}}_w$, without modifying the final solution of the system. When a pseudo-time derivative is present in unsteady computations, as it is the case in the present solver, this is called dual-time stepping. The pseudo-time derivative will thus act on the removal of transient errors while the physical time derivative will ensure time accuracy. And there will be the possibility to control this removal of the transient errors by adjusting $\Delta\tau$ and selecting $\underline{\boldsymbol{P}}_w$. This is detailed in Section 10.1.2.
- Meyer (2022, Sec. 4.3) observes that a preconditioner can be built following different approaches and using different sets of variables. Hence, there are several families of preconditioning matrices, the only requirement is that the matrix is positive-definite to maintain the same physics of the problem, as explained in Turkel (1993). Preconditioning schemes are often derived using a set of primitive variables including the pressure, the velocity vector and another variable. The choice of pressure is preferable since it plays an important role in the low-Mach limit and it allows a direct treatment of the problematic pressure waves.

The preconditioner investigated in this work is the one of Weiss and Smith (Weiss and Smith (1995)) adapted to general EOS by taking inspiration from the work of Zong and Yang (2007). This preconditioner is built on the primitive p–$\mathbf{v}$–T variables set and its derivation starts by considering the steady/unsteady Navier-Stokes system with the pseudo-temporal term expressed in conservative variables, $\mathbf{u}$, and by pre-multiplying the pseudo-temporal term by the transformation Jacobian to express the equations in the p–$\mathbf{v}$–T variables, $\mathbf{w}$:

$$\frac{\partial \mathbf{u}}{\partial \tau} + \mathcal{N}(\mathbf{u}) = 0 \Rightarrow \frac{\partial \mathbf{u}}{\partial \mathbf{w}}\frac{\partial \mathbf{w}}{\partial \tau} + \mathcal{N}(\mathbf{w}) = 0\,,$$

where the transformation matrix from conservative to primitive p–$\mathbf{v}$–T variables, for general EOS, is given by

$$\frac{\partial \mathbf{u}}{\partial \mathbf{w}} = \begin{bmatrix} \left.\frac{\partial \rho}{\partial p}\right|_T & 0 & 0 & \left.\frac{\partial \rho}{\partial T}\right|_p \\ v_x \left.\frac{\partial \rho}{\partial p}\right|_T & \rho & 0 & v_x \left.\frac{\partial \rho}{\partial T}\right|_p \\ v_y \left.\frac{\partial \rho}{\partial p}\right|_T & 0 & \rho & v_y \left.\frac{\partial \rho}{\partial T}\right|_p \\ \rho \left.\frac{\partial e}{\partial p}\right|_T + E \left.\frac{\partial \rho}{\partial p}\right|_T & \rho v_x & \rho v_y & \rho \left.\frac{\partial e}{\partial T}\right|_p + E \left.\frac{\partial \rho}{\partial T}\right|_p \end{bmatrix}. \tag{5.66}$$

The second step is to realize that the term $\partial\rho/\partial p$ present in the first column of the matrix dictates the propagation speed of acoustic waves after the multiplication with the time derivative. Indeed, Menikoff and Plohr (1989) relate the ratio between the isentropic and isothermal compressibilities to the ratio of the heat capacities,

$$\frac{K_S}{\beta_T} = \frac{C_v}{C_p},$$

with

$$K_S = \frac{1}{\rho c^2}, \quad \beta_T = \frac{1}{\rho} \left.\frac{\partial \rho}{\partial p}\right|_T.$$

It then comes that

$$\left.\frac{\partial \rho}{\partial p}\right|_T = \frac{C_p}{C_v} \frac{1}{c^2} = \frac{\gamma}{c^2}, \tag{5.67}$$

where the speed of sound, c, appears. The preconditioning matrix is then established simply by replacing this critical term with a tunable scaling factor Θ:

$$\frac{\partial \mathbf{u}}{\partial \mathbf{w}} \Rightarrow \underline{\boldsymbol{P}}_q = \begin{bmatrix} \Theta & 0 & 0 & \left.\frac{\partial \rho}{\partial T}\right|_p \\ v_x \Theta & \rho & 0 & v_x \left.\frac{\partial \rho}{\partial T}\right|_p \\ v_y \Theta & 0 & \rho & v_y \left.\frac{\partial \rho}{\partial T}\right|_p \\ \rho \left.\frac{\partial e}{\partial p}\right|_T + E\Theta & \rho v_x & \rho v_y & \rho \left.\frac{\partial e}{\partial T}\right|_p + E \left.\frac{\partial \rho}{\partial T}\right|_p \end{bmatrix}. \tag{5.68}$$

From Equation 5.67, the scaling factor is defined as:

$$\left.\frac{\partial \rho}{\partial p}\right|_T \Rightarrow \Theta = \frac{\gamma}{c^2} + \frac{1/\epsilon^2 - 1}{c^2}, \tag{5.69}$$

with the preconditioning factor, $0 \leq \epsilon \leq 1$, including both inviscid and viscous effects following Weiss and Smith (1995), Buelow et al. (1994) and Zong and

Yang (2007), i.e.

$$\begin{aligned}
\epsilon &= \min\left(1, \max(\epsilon_{inv}, \epsilon_{vis})\right) \\
\epsilon_{inv} &= \min\left(1, \max(\mathrm{Ma}_l^2, \mathrm{Ma}^2)\right) \\
\epsilon_{vis} &= \max\left(\frac{v_x^2\delta_x(\delta_x-1)}{v_x^2\delta_x^2+c^2}, \frac{v_y^2\delta_y(\delta_y-1)}{v_y^2\delta_y^2+c^2}\right) \\
\delta_i &= \max\left(\nu, \frac{\nu}{\mathrm{Pr}}\right)\frac{1}{v_i}\frac{\mathrm{CFL}}{\mathrm{VNN}} \\
\mathrm{VNN} &= \frac{\nu}{\|\mathbf{v}\|R_{\Delta x}}\frac{\mathrm{CFL}}{\mathrm{Pr}}, \quad \mathrm{CFL} = \text{fixed}
\end{aligned} \tag{5.70}$$

where Ma_l is a lower value for the local Mach number to avoid singularities at solid walls, $\nu = \mu/\rho$ is the kinematic viscosity, CFL is the cell CFL number which is fixed per time step over the entire domain (the pseudo-time step is locally adapted accordingly), VNN is the cell von Neumann number and $R_{\Delta x}$ is the cell inscribed radius. For the inviscid fluxes, the definition of ϵ_{inv} makes such that the pressure fluctuations scales with Ma^2 as the Mach number approaches zero.

To see how the preconditioning effectively homogenize the eigenvalues of the system, the eigenvalues of the preconditioned system can be computed. Following Vigneron et al. (2006), this can be done by introducing a Fourier mode into the equations and by solving for the resulting dispersion equation. The eigenvalues for the inviscid part of the system are given by

$$\boldsymbol{\lambda} = \left(\lambda_1^*, \lambda_2^*, \|\mathbf{v}\|, \|\mathbf{v}\|\right),$$

with the two rescaled acoustic eigenvalues expressed as

$$\lambda_{1,2}^* = \frac{1}{2}\left[(1+\epsilon)\|\mathbf{v}\| \pm \sqrt{(1-\epsilon)^2\|\mathbf{v}\|^2 + 4\epsilon c^2}\right]. \tag{5.71}$$

When $\epsilon \to 1$ in the compressible regime, the original eigenvalues $\lambda_{1,2} = \|\mathbf{v}\| \pm c$ are retrieved while for $\epsilon \to 0$, $\lambda_{1,2}^* \to \|\mathbf{v}\|$, which indeed equalizes the various eigenvalues of the system. Equation 5.71 is used to compute the spectral radius of the convective operators to determine the pseudo-time step. Details can be found in Section 10.1.2 where the temporal discretization of the preconditioned system is also elaborated.

The last point to discuss is the way to transform the preconditioning matrix $\underline{\boldsymbol{P}}_w$ built for the primitive p–$\mathbf{v}$–T variables to any other variables set, say $\mathbf{w}'$, for which we want to solve the system. This is accomplished by multiplying the matrix by the transformation Jacobian:

$$\underline{\boldsymbol{P}}_{w'}\frac{\partial \mathbf{w}'}{\partial \tau} = \underline{\boldsymbol{P}}_w\frac{\partial \mathbf{w}}{\partial \tau} \Rightarrow \underline{\boldsymbol{P}}_{w'} = \underline{\boldsymbol{P}}_w\frac{\partial \mathbf{w}}{\partial \mathbf{w}'}. \tag{5.72}$$

It is noted by Hakimi (1997, Sec. 2.3) that the various preconditioned systems

that can be obtained depending on the choice of the working variables are strictly equivalent at the continuous level if Equation 5.72 is used.

Flux preconditioning

Flux preconditioning deals with the accuracy issues by adjusting the upwind-based discretization. For Godunov-type schemes, since time preconditioning is applied at the continuous level while flux preconditioning is done at the discrete one, the approximate Riemann solver should be based on the time-preconditioned system, as discussed by Guillard and Murrone (2004). For FVS and AUSM-type schemes, the "preconditioning" consists in modifying the artificial dissipation term. Sachdev et al. (2012) observes that AUSM-type schemes are particularly attractive in that regard since they provide a natural means to decouple the dissipation for the momentum and pressure flux thus allowing more flexibility to tailor the dissipation for acoustic and hydrodynamic instabilities in unsteady flows. Preconditioned methods effectively improve low-Mach accuracy but should also feature some kind of "switching" mechanism to smoothly transition from the critical low-Mach regime (in which the original Riemann solver is adapted) to the regular compressible regime (where the solver is left untouched). Such methods are referred as all-Mach. Based on the abundant recent literature about the design of AUSM-type all-Mach schemes, we present here a unifying framework that has been implemented in our solver with the goal of being easily extendable to new AUSM-type schemes.

The Liou-Steffen splitting, Equation 5.24, is considered:

$$\hat{\mathbf{F}}^c_m = \underbrace{\underbrace{\rho v_n}_{\hat{\dot{m}}} \underbrace{\begin{bmatrix} 1 \\ v_x \\ v_y \\ H \end{bmatrix}}_{\hat{\boldsymbol{\phi}}}}_{\hat{\dot{m}}^*} = \underbrace{\underbrace{v_n}_{\hat{\dot{m}}^*} \underbrace{\begin{bmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho H \end{bmatrix}}_{\hat{\boldsymbol{\phi}}^*}}_{\hat{\dot{m}}^*}, \quad \hat{\mathbf{F}}^c_p = \underbrace{\underbrace{p}_{\tilde{p}} \underbrace{\begin{bmatrix} 0 \\ n_{e,x} \\ n_{e,y} \\ 0 \end{bmatrix}}_{\mathbf{N}}}_{\tilde{p}}, \tag{5.73}$$

where the $\hat{\bullet}$ is used to identify the terms that are still undefined at this stage because they have to be evaluated at the element faces on which left and right solution vectors,$\mathbf{u}^+$ and $\mathbf{u}^-$, are available. The specification of the mass flux and pressure flux functions is done separately, which is one of the key features of AUSM-type schemes.

Mass flux Upwinding is first applied to the vector of quantities, $\hat{\boldsymbol{\phi}}$ or $\hat{\boldsymbol{\phi}}^*$, such that

$$\begin{aligned} \hat{\mathbf{F}}^c_m &= \hat{\dot{m}} \begin{cases} \boldsymbol{\phi}^+ & \text{if } \hat{\dot{m}} \geq 0 \\ \boldsymbol{\phi}^- & \text{if } \hat{\dot{m}} < 0 \end{cases}, \\ &= \hat{\dot{m}}^* \begin{cases} \boldsymbol{\phi}^{*,+} & \text{if } \hat{\dot{m}}^* \geq 0 \\ \boldsymbol{\phi}^{*,-} & \text{if } \hat{\dot{m}}^* < 0 \end{cases}, \end{aligned}$$

which can be rewritten as

$$\begin{aligned}\hat{\mathbf{F}}_m^c &= \frac{1}{2}\left(\hat{\dot{m}} + |\hat{\dot{m}}|\right)\boldsymbol{\phi}^+ + \frac{1}{2}\left(\hat{\dot{m}} - |\hat{\dot{m}}|\right)\boldsymbol{\phi}^- = \underbrace{\frac{1}{2}\left(\boldsymbol{\phi}^+ + \boldsymbol{\phi}^-\right)\hat{\dot{m}}}_{\text{Central difference}} + \underbrace{\frac{1}{2}\left(\boldsymbol{\phi}^+ - \boldsymbol{\phi}^-\right)|\hat{\dot{m}}|}_{\text{Upwinding dissipation}}\,, \\ &= \frac{1}{2}\left(\hat{\dot{m}}^* + |\hat{\dot{m}}^*|\right)\boldsymbol{\phi}^{*,+} + \frac{1}{2}\left(\hat{\dot{m}}^* - |\hat{\dot{m}}^*|\right)\boldsymbol{\phi}^{*,-} = \underbrace{\frac{1}{2}\left(\boldsymbol{\phi}^{*,+} + \boldsymbol{\phi}^{*,-}\right)\hat{\dot{m}}^*}_{\text{Central difference}} \\ &\qquad + \underbrace{\frac{1}{2}\left(\boldsymbol{\phi}^{*,+} - \boldsymbol{\phi}^{*,-}\right)|\hat{\dot{m}}^*|}_{\text{Upwinding dissipation}}\,.\end{aligned} \tag{5·74}$$

Then, the Mach number is introduced as a working variable instead of the velocity by writing

$$\hat{\dot{m}} = \hat{c}\cdot\widehat{\rho\,\mathrm{Ma}}\,, \quad \hat{\dot{m}}^* = \hat{c}\cdot\widehat{\mathrm{Ma}}\,.$$

The next step is to define the interface speed of sound, $\hat{c}$, and the interface Mach number multiplied by the density or not, $\widehat{\rho\,\mathrm{Ma}}$ and $\widehat{\mathrm{Ma}}$. It is really at this point that the various AUSM-type schemes will emerge. For the speed of sound, the most straightforward approach is to take the average between both states,

$$\hat{c} = \frac{1}{2}\left(c^+ + c^-\right)\,,$$

but other more involved possibilities exist, motivated e.g. by shock resolution capabilities, see Meng-Sing Liou (2006). Concerning the interface Mach number, the idea is to combine the waves traveling toward the interface from the two adjacent cells in a FVS spirit, i.e.

$$\widehat{\mathrm{Ma}} = \mathcal{M}_p(\mathrm{Ma}^+) + \mathcal{M}_m(\mathrm{Ma}^-)\,,$$

where various ways of defining the splitting, via $\mathcal{M}_{p/m}$, exist. A distinction has to be made at this point between the schemes belonging to the AUSM family and the ones to the "subfamily" of Simple Low-dissipation AUSM (SLAU) schemes. Schemes like AUSM (Meng-Sing Liou and Steffen Jr (1993)), AUSM-+ (Meng-Sing Liou (1996)), AUSM-+up(Meng-Sing Liou (2006)) rely on an improved version of the Van Leer's splitting (Van Leer (1982)) to split the interface Mach number:

$$\widehat{\mathrm{Ma}} = \underbrace{\mathcal{M}_p^{\mathrm{AUSM}}(\mathrm{Ma}^+,\beta) + \mathcal{M}_m^{\mathrm{AUSM}}(\mathrm{Ma}^-,\beta)}_{\text{Flux vector splitting of Ma}} + \underbrace{D_m^{\mathrm{AUSM}}(\Delta p,\hat{c})}_{\text{Pressure diffusion}}\,, \tag{5·75}$$

with the AUSM splitting functions depending on the parameter β and given by

$$\begin{aligned}\mathcal{M}_p^{\text{AUSM}}(\text{Ma},\beta) &= \begin{cases}\frac{1}{4}(\text{Ma}+1)^2+\beta(\text{Ma}^2-1)^2 & \text{if } |\text{Ma}|<1\\ \frac{1}{2}(\text{Ma}+|\text{Ma}|) & \text{if } |\text{Ma}|\geq 1\,,\end{cases}\\ \mathcal{M}_m^{\text{AUSM}}(\text{Ma},\beta) &= \begin{cases}-\frac{1}{4}(\text{Ma}-1)^2-\beta(\text{Ma}^2-1)^2 & \text{if } |\text{Ma}|<1\\ \frac{1}{2}(\text{Ma}-|\text{Ma}|) & \text{if } |\text{Ma}|\geq 1\,.\end{cases}\end{aligned} \tag{5.76}$$

and the pressure diffusion term $D_{\dot{m}}^{\text{AUSM}}$ to be commented below. On the other hand, schemes like SLAU (Shima and Kitamura (2009)) and SLAU2 (Kitamura and Shima (2013)) are based on the Courant–Isaacson–Rees Roe scheme's mass flux (Roe (1981)) which is modified by using arithmetic averaged values rather than Roe averaged ones. It is also enhanced to provide an accurate mass flux at all speeds by introducing a function g. The resulting splitting function is used to split the product of the density and the Mach number:

$$\widehat{\rho\,\text{Ma}} = \underbrace{\rho^+\,\mathcal{M}_p^{\text{SLAU}}(\text{Ma}^+,\text{Ma}^-)+\rho^-\,\mathcal{M}_m^{\text{SLAU}}(\text{Ma}^+,\text{Ma}^-)}_{\text{Flux vector splitting of }\rho\,\text{Ma}} + \underbrace{D_{\dot{m}}^{\text{SLAU}}(\Delta p,\hat{c})}_{\text{Pressure diffusion}}\,, \tag{5.77}$$

where the SLAU splitting functions are defined, following Yoo and Sung (2021), as

$$\begin{aligned}\mathcal{M}_p^{\text{SLAU}}(\text{Ma}^+,\text{Ma}^-) &= \frac{1}{2}\left(\text{Ma}^+ + \left[(1-g)|\text{Ma}^+| + g\overline{\text{Ma}}\right]\right)\\ \mathcal{M}_m^{\text{SLAU}}(\text{Ma}^+,\text{Ma}^-) &= \frac{1}{2}\left(\text{Ma}^- - \left[(1-g)|\text{Ma}^-| + g\overline{\text{Ma}}\right]\right)\\ \overline{\text{Ma}} &= \frac{\rho^+|\text{Ma}^+|+\rho^-|\text{Ma}^-|}{\rho^++\rho^-}\end{aligned} \tag{5.78}$$

In both Equation 5.75 and Equation 5.77, a last term is added to the splitting of the Mach number. This is a pressure diffusion term function of the pressure difference, $\Delta p = p^- - p^+$, and the interface speed of sound. Its expression depends on the scheme considered. The term is activated in the low-Mach regime to achieve accurate results by controlling the amount of dissipation which is added to that introduced by the upwinding term in Equation 5.74. Note also that the FVS flavor introduced in those schemes imply that the numerical fluxes are based on the eigenvalues of the system rather than the eigenstructure, which makes them particularly easy to implement.

Pressure flux Following Kitamura and Hashimoto (2016), the pressure flux part has the following general expression for AUSM and SLAU-type schemes:

$$\begin{aligned}\hat{p} = \underbrace{\frac{1}{2}(p^+ + p^-)}_{\text{Central difference}} + \underbrace{\frac{1}{2}(p^+ - p^-)(\mathcal{D}_p(\text{Ma},\alpha) - \mathcal{D}_m(\text{Ma},\alpha))}_{\text{Upwinding dissipation}}\\ + \underbrace{\frac{1}{2}\chi(p^+ + p^-)(\mathcal{D}_p(\text{Ma},\alpha) + \mathcal{D}_m(\text{Ma},\alpha) - 1)}_{\text{Additional dissipation}} + \underbrace{D_p(\Delta v_n)}_{\text{Additional dissipation}}\,,\end{aligned} \tag{5.79}$$

where the pressure splitting functions, also inspired by Van Leer FVS, are common for the different schemes and given by

$$\begin{aligned}\mathcal{P}_p(\mathrm{Ma},\alpha) &= \begin{cases}\frac{1}{4}(\mathrm{Ma}+1)^2(2-\mathrm{Ma})+\alpha\mathrm{Ma}(\mathrm{Ma}^2-1)^2 & \text{if } |\mathrm{Ma}|<1\\ \frac{1}{2}(1+\mathrm{sign}(\mathrm{Ma})) & \text{if } |\mathrm{Ma}|\geq 1\,,\end{cases}\\ \mathcal{P}_m(\mathrm{Ma},\alpha) &= \begin{cases}\frac{1}{4}(\mathrm{Ma}-1)^2(2+\mathrm{Ma})-\alpha\mathrm{Ma}(\mathrm{Ma}^2-1)^2 & \text{if } |\mathrm{Ma}|<1\\ \frac{1}{2}(1-\mathrm{sign}(\mathrm{Ma})) & \text{if } |\mathrm{Ma}|\geq 1\,.\end{cases}\end{aligned} \quad [5.80]$$

The scaling function χ and the additional dissipation term proportional to the normal velocity jump across the face interface, $v_n = v_n^- - v_n^+$, are schemes dependent.

The unifying formulation presented above has been implemented in the solver, exploiting the polymorphism of the language such that only a few methods need to be overloaded or added for each implemented scheme. Recent improved versions of AUSM-type schemes that have not been included in the code yet but should fit into the above formulation include AUSM+M (S.-s. Chen et al. (2020)), HAUS (Yoo and Sung (2021)), SLAU-M1 (Qu et al. (2021)), AUSMPW+-N (Hyeongjun Kim et al. (2018)) and AUSMPWM2 (Qu et al. (2017)). Other state-of-the-art all-speed Riemann solvers which are not of AUSM type have been recently developed and could be worth investigating, such as RoeMAS (Qu et al. (2015)), RoeM-N (Hyeongjun Kim et al. (2018)), HLLC+-lowMach (S. Chen et al. (2020)) or HLLE (Gogoi and Mandal (2022)).

Contribution 5.1: DG all-speed fluid solver for the bulk equations

The main contributions that can be identified from this chapter are:

- A presentation of the variational form following the formalism of Ralph Hartmann (2008), including the specification of the numerical fluxes, and a detailed explanation of its practical implementation (although the DG form was already implemented in `Argo`).
- A complete description of the ingredients needed for the discretization to be suited over a wide range of Mach numbers.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- The extension of the class dealing with general EOS by the versatile choice of working variables and by the generic implementation of AUSM-type flux preconditioning.

The developments provide the platform with a DG solver combining several versatile features to be able to tackle a wide range of flow conditions.

6

Spatial discretization of the level–set advection for interface motion

The material of Section 6.3, Section 6.4 and Section 6.5 was presented in Henneaux et al. (2021c) and published in Henneaux et al. (2021b).

6.1 An overview of the methods to follow interfaces and the choice of the level-set method . . . 198
6.1.1 State-of-the-art . . . 198
6.1.2 The choice of the level-set method . . . 200
6.2 The level-set method and its DG discretization . . . 204
6.2.1 Governing equations . . . 204
6.2.2 DG discretization . . . 208
6.2.3 Temporal discretization . . . 210
6.3 Reinitialization and computation of the closest points on the interface . . . 211
6.3.1 Some background . . . 211
6.3.2 Implementation . . . 212
6.3.3 General algorithm and additional remarks . . . 216
6.3.4 Verification test case . . . 217
6.4 Velocity extension . . . 219
6.4.1 Some background . . . 219
6.4.2 Implementation . . . 221
6.4.3 Verification test cases . . . 223
6.5 The narrow band strategy . . . 229
6.5.1 Some background . . . 229
6.5.2 Implementation . . . 229
6.6 Enforcement of continuous level-set iso-0 contour . . . 232
6.6.1 Two distinct level-set functions . . . 232
6.6.2 Four approaches to enforce continuity . . . 232
6.7 Organization and summary of the various methods . . . 238

We have seen in the previous chapter how the governing equations in the bulk of a phase are discretized by means of the DG method. Before investigating the way the sharp interface treatment is realized, an essential feature of the developed numerical framework concerns the representation and the tracking of moving interfaces. This is the object of this chapter and the next one.

6.1 An overview of the methods to follow interfaces and the choice of the level-set method

6.1.1 State-of-the-art

We start this chapter by giving a general overview of the existing numerical techniques to represent and follow a phase interface without direct link with compressible two-phase flows. Two main classes of methods can be identified.

Interface-fitting on evolving meshes Lagrangian *interface-tracking* algorithms are characterized by a moving computational mesh displaced with a velocity chosen as close as possible to the local fluid velocity. Doing so material interfaces can be properly identified and tracked. The main advantages of these methods is to reduce the numerical dissipation errors due to the nonlinear convective terms as those disappear from the discretization. They are however difficult to apply to problems involving large interface deformations as the mesh gets highly distorted, which may cause the breakdown of the computation.

This is the reason which led to the development of *Arbitrary Lagrangian-Eulerian* (ALE) methods where the grid nodes can be moved at an arbitrary velocity, independent of the local fluid velocity. Thanks to this flexibility, it is possible, from an initial mesh conforming with the interface, to constrain it to move with the interface velocity. Everywhere else in the domain, the mesh is free to move in any specified way. However, as the interface deforms significantly, the element quality deteriorates and some mesh regularization procedures followed by a remapping step are required to preserve validity of the grid. A complete overview of ALE methods with a focus on compressible multi-material flows is given in Barlow et al. (2016). The direct ALE approach avoids those remeshing plus remapping steps by building space-time control volumes which connect each element configuration at the current time step to the next one. An improvement of this approach allowing for topology changes has been recently proposed in Gaburro et al. (2020).

Another possible approach to ensure that mesh faces conform with the phase interface is to be found in the High-Order Implicit Shock Tracking (*HOIST*) by Matthew J Zahr and Persson (2018) and the Moving Discontinuous Galerkin Method with Interface Condition Enforcement (*MDG-ICE*) by Corrigan et al. (2019) methods. There the discontinuity tracking is done implicitly through the solution of an optimization problem. To achieve this, those methods treat the discrete grid geometry as an additional solver variable so that the solver simultaneously moves the discrete grid geometry to fit interfaces while resolving the flow field.

A very recent new mesh motion paradigm, named the extreme mesh deformation (X-MESH), was introduced in Moes et al. (2023). The distinguishing feature of this approach is that elements of the mesh can reach zero measure, based on a convergence condition for FEM established by Kučera (2016). This allows for an interface coinciding with mesh faces to be relayed from one node to another in a continuous fashion. Contrary to the above-mentioned methods, this approach can thus naturally handle complex topology changes.

Interface cutting fixed meshes The second class of methods relies on a fixed mesh on which the interface freely evolves by intersecting elements. To follow the interface in this context, there are two main categories. In the first one of *interface-tracking* methods, the interface is explicitly tracked by a discrete mesh lying on and evolving with the interface. The front-tracking method of Tryggvason et al. (2001) for multiphase flows computations is a notable

example. The use of marker particles in the context of compressible two-phase flows discretized by the DG method has been presented by H. Lu et al. (2016). With this representation, very high accuracy can be achieved. However, frequent reseeding of the markers/particles is required to keep an appropriate distribution, which becomes computationally intricate and expensive for large interface deformations. In the second category of Eulerian *interface-capturing* methods, the prominent methods are the volume-of-fluid (VOF), the level-set (LS) methods and their variants. A recent overview of the available methods can be found in Mirjalili et al. (2017) and a comparison between some of them applied to two-phase flows has recently been presented in Boniou et al. (2022a). The basic principle of those methods is to solve an advection equation for an auxiliary function defined over the entire domain from which the interface position can be determined. The need for remeshing is here avoided. In the VOF method, the advected quantity is the volume fraction of the fluid in each computational cell. For the level-set method, a scalar field embeds the interface as one of its iso-contours. Generally, compared to interface-fitting approaches, the major advantage of fixed mesh methods is to assign the task of interface motion to a method without touching the underlying grid. By avoiding the alignment of grid interfaces with flow interfaces, it becomes less onerous to simulate flows with unknown or dynamic interface topology. Moreover, the extension of an existing fixed grid Eulerian flow solver is made easier since the method to track the interface can be added separately from the one dealing with the discretization of fluid equations.

It is interesting to point out here that in diffuse interface methods like the multi-fluid or phase-field methods (introduced in Section 3.1.1), the interface position is not precisely defined as it is considered as a diffuse region rather than a sharp discontinuity. Instead, it is explicitly taken into account into the governing equations, e.g. in the multi-fluid approach through a volume fraction quantity that is used to derive the transport equations over the whole domain.

6.1.2 The choice of the level-set method

Based on the global overview above, we are now in a position to motivate and discuss the choice of the level-set method to represent and follow the interface in our two-phase flows applications.

Assets of the level-set method

First, the choice of a fixed underlying mesh with dedicated tracking methods over an interface-fitting moving/adapting mesh has been made mainly because of the complexity associated to the latter in the context of two-phase flows. Indeed, methods using interface-fitted moving meshes in an ALE formalism become particularly complex for significant interface deformations. They require either mesh regularization procedures followed by a remapping or for the direct ALE approach the construction of space-time control volumes. To alleviate

those difficulties, the X-MESH method is particularly promising and has been very recently applied to two-phase flow configurations by Quiriny et al. (2023). The method is however still in its infancy. Finally, optimization-based methods like HOIST and MDG-ICE would have required profound transformations of the existing solver with the need to solve overdetermined systems of nonlinear equations, which can be costly for an unsteady simulation. Those methods have also the potential to sharply capture flow interfaces but they are also very recent and have only been applied to shock discontinuities up to now (see e.g. Huang and Matthew J Zahr (2022) for HOIST and Kercher et al. (2021) for MDG-ICE) and not highly deforming phase interfaces.

Then, considering the available techniques using a fixed mesh, the choice of an implicit interface representation via an interface-capturing method appeared quite naturally for the inherent ability of such methods to handle complex interface deformations and topological changes like break-up and merging. The reasons that led to the choice of the level-set method among the interface-capturing methods can be stated as follows:

- *Strengths of the level-set method*: the first motivation is of course the advantages that the method presents. The level-set directly provides an implicit representation of the interface, hence also the possibility to readily obtain accurate estimations of secondary interface quantities like normal and curvature. In comparison, the interface in the VOF method is not directly available and must be retrieved through a reconstruction strategy, which may introduce some inaccuracies. Moreover, if the level-set function is chosen as a signed distance function, it is a smoother version than the volume fraction field, which facilitates its discretization and the computation of its derivatives. Finally, the level-set method can handle topology changes automatically because the function can undergo sign changes but the zero iso-contour, which forms the interface itself, remains continuous and ensures the preservation of the interface topology.
- *Suitability for high-order unfitted methods*: the implicit representation of the interface through an iso-contour allows to fully take advantage of the high-order features of the discretization scheme since the interface will be represented, like the flow variables, with a high-order polynomial function.
- *Availability in* `Argo`: the last reason is that the level-set method has been already implemented in the solver and proved its ability to simulate two-phase flows situations in Pochet et al. (2013). This gave us confidence in the possibility to build upon and improve the original implementation.

Difficulties of the level-set method

The level-set method, like any numerical method, has also its drawbacks.

Mass conservation The most important drawback concerns its mass conservation issues, which arise from two main sources:

- *Non-physical nature of the level-set*: because the level-set function does not represent any physical quantity, it does not enforce mass conservation by construction, unlike VOF method which is based on the physical volume fraction of one phase. If the level-set iso-contour representing the interface was perfectly advected such that it would correspond at any time to the exact physical interface, then there would not be any mass loss or gain, despite the fact that the interface is located by a non-physical scalar field. However, in practice, the numerical spatial and temporal discretizations will inevitably introduce inaccuracies in the interface position. This will result in an artificial mass gain or loss for both phases (e.g. when some fluid quantities are integrated on either side of the interface). Thus, as pointed out by Marchandise et al. (2006, Sec. 1.3), the poor mass conservation is more related to the discretization method applied to the level-set rather than the level-set formulation itself. The use of high-order schemes can significantly reduce this issue.
- *Reinitialization*: the second reason for the mass conservation issue has to do with the reinitialization procedure which is most of the time required during the level-set advection to preserve the signed distance property. The fact of adjusting the level-set function, including the interface region, to a signed-distance field will perturb the position of the interface and lead to a mass loss or gain. We will see that the use of an advection velocity corresponding to the extended velocity from the interface drastically reduces the frequency at which reinitialization should be applied.

In order to overcome the mass conservation issues, at least five strategies are possible:

- *Mass correction*: the first possibility is to correct the advection of the level-set such that mass conservation is enforced. This can be done e.g. by adding a volume-constraint to the level-set during the reinitialization, in the interface-preserving level-set redistancing algorithm of Sussman and Fatemi (1999), or by correcting the advection velocity to preserve the volume, in the interface-correction level-set of Ge et al. (2018).
- *Coupled level-set and VOF method* (*CLSVOF*): the idea behind this hybrid method, introduced in Sussman and Puckett (2000), is to combine the accuracy of the level-set method with the inherent mass conservation property of the VOF method. This is done by advecting both a level-set and volume fraction fields and by coupling the two via interface reconstruction and reinitialization.
- *Conservative level-set*: in an effort to add conservation properties to the level-set, a modified hyperbolic tangent function can be used in lieu of the standard signed distance function. This has been proposed by Olsson and Kreiss (2005) and adopted for instance by Owkes and Desjardins (2013) in a DG framework. This alternative definition mimics the volume fraction field (while keeping a smooth differentiable function) and therefore encompasses some physical meaning, which is beneficial for

conservation properties.

- *Reference map method*: in this method, introduced by Bellotti and Theillard (2019), the entire space on which the level-set function is defined (called the reference map) is deformed, instead of advecting the level-set itself. The level-set is then recovered as the image of its initial profile in the deformed space. Since the advected object, i.e. the reference map, is smoother than the level-set function, this approach grants higher precision and reduces conservation errors. The follow-up work of Theillard (2021) extended the approach by introducing an intermediate mapping ensuring volume preservation.
- *Particle-based level-set method*: this class of level-set methods contains itself several subcategories. In the particle level-set method of Enright et al. (2002), a set of Lagrangian particles seeded in the interface region is used to improve the Eulerian level-set solution. In the marker level-set method of Mihalef et al. (2007) and the marker redistancing level-set method of Robert Nourgaliev et al. (2010), the interface is advected by a set of unconnected markers, the continuous level-set function is recovered from those markers and the interface topology description is then delegated to the Eulerian level-set method. Finally, in the full Lagrangian method of Hieber and Koumoutsakos (2005) and Schulze et al. (2023), the level-set equation is described in the Lagrangian frame and the particles are used for its discretization through the domain. The common point of those methods, as highlighted by Robert Nourgaliev et al. (2010), is that Lagrangian marker tracking is the most effective way to carry under-resolved/subgrid interface topology and are thus effective to avoid mass conservation issues of level-set method which are maximal in such circumstances. As an additional benefit, the time step restriction of pure Eulerian level-set methods is removed.

Additional computational cost Another aspect of the level-set method that should be taken into consideration is the additional computational cost that the advection of a separated scalar field incurs. If an explicit scheme is used, the hyperbolic nature of the transport equation will impose a time constraint. On the other hand, the use of an implicit scheme will necessitate the solution of a non-linear equation at each time step. To significantly decrease the computational cost associated to the level-set operations, the narrow band technique is a common choice. It starts from the observation that only the interface region is actually meaningful in an interface problem and restricts all the operations to this region.

Low regularity regions Finally, a last important point to mention about the level-set method is the presence of possible ares of low regularity. Those can be found for instance at the edge of the narrow band, where the interface curvature is important or when level-set iso-0 are approaching to each other, e.g. in case of merging. With the presence of kinks or low regularity in the

level-set field, the high-order DG method might be prone to oscillations and instabilities. That is why some regularization strategies might be required, such as e.g. FV sub-cell limiting used in Fabian Föll et al. (2020) and Mossier et al. (n.d.). Furthermore, in the event of topological changes, the proximity of several iso-0 contours might lead to erroneous estimation of the interface normal and curvature because of the discontinuities in the level-set derivatives. Several approaches exist to handle this kind of situations, including the one of Salac and W. Lu (2008) in which several level-set functions are extracted from the global function such that they represent a single body and the LOLEX method by Ervik et al. (2014) where the construction of separated level-set fields remain local and temporary. Ngo et al. (2019) adapt and compare the two methods on unstructured meshes.

In this work, we do not explore any of the mass conservation improving techniques described above and rely on the conventional formulation of the level-set method. Nor do we implement any methods to deal with low regularity regions. We combine the classical level-set formulation with high-order reinitialization and extension procedures, as well as narrow band strategy. We believe that these enhancements, combined with the high-order property of the DG scheme, limit the shortcomings of the standard level-set method and make it appropriate for the targeted two-phase flows configurations.

6.2 The level-set method and its DG discretization

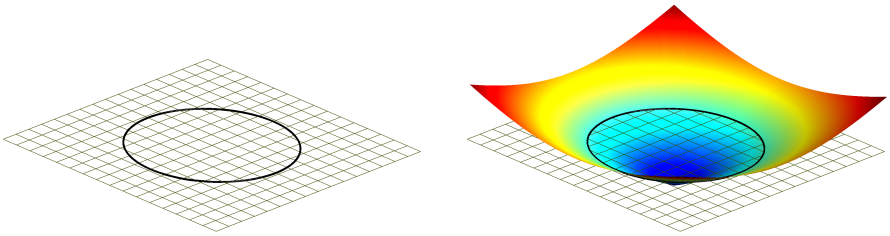
6.2.1 Governing equations

The level-set method, first proposed in Osher and James A Sethian (1988), relies on two central embeddings, as explained by J. A. Sethian and Smereka (2003): (i) the embedding of the $D-1$ dimensional interface Γ as the zero level-set of a D dimensional function φ defined over the entire domain Ω, (ii) the embedding (or extension) of the interface velocity $\mathbf{v}_\Gamma$ to this higher dimensional level-set function. This is illustrated in Figure 6.1 and Figure 6.2.

Interface representation

By definition, the continuous interface, $\tilde{\Gamma}$, is implicitly approximated by the zero iso-contour of the scalar discretized level-set function $\varphi : \Omega \to \mathbb{R}$ for all time t belonging to a finite dimensional function space $\mathcal{W}_p$ to be defined in Equation 6.12, viz.

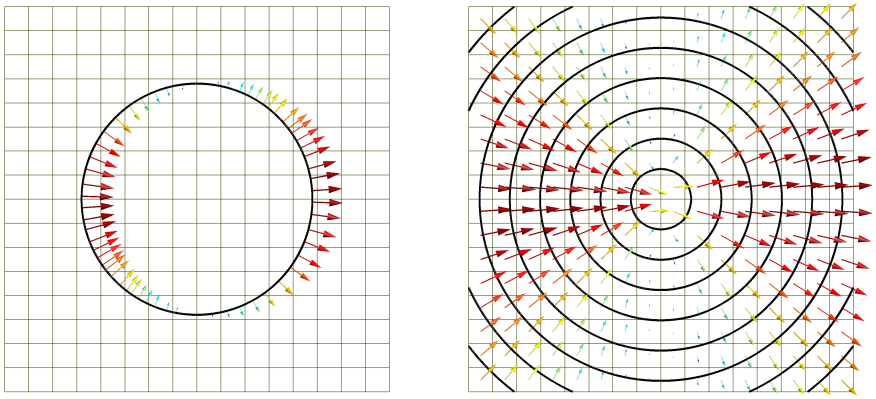
$$\tilde{\Gamma}(\mathbf{x},t) \approx \Gamma(\mathbf{x},t) := \{\mathbf{x} \in \Omega \,|\, \varphi(\mathbf{x},t) = 0\}\,, \quad \forall t\,. \tag{6.1}$$



a / Interface represented by a zero iso-contour.

b / Definition of the two-dimensional level-function as a signed distance function.

Figure 6.1 / Embedding of a one-dimension interface as the zero iso-contour of the two-dimensional level-set function.



a / Velocity defined on the level-set iso-0.

b / Extension of the interface velocity into the domain along the normal directions.

Figure 6.2 / Embedding (extension) of the interface velocity to the two-dimensional level-set function.

The normal vector $\mathbf{n}_\Gamma$ to the interface and the interface curvature κ_Γ can then be obtained as (see e.g. Goldman (2005))

$$\mathbf{n}_\Gamma(\mathbf{x},t) = \frac{\nabla\varphi(\mathbf{x},t)}{\|\nabla\varphi(\mathbf{x},t)\|}\,, \tag{6.2}$$

$$\kappa_\Gamma(\mathbf{x},t) = \nabla\cdot\left(\frac{\nabla\varphi(\mathbf{x},t)}{\|\nabla\varphi(\mathbf{x},t)\|}\right), \quad \forall\mathbf{x}\in\Gamma(\mathbf{x},t)\,. \tag{6.3}$$

Away from the interface front, the level-set function needs also to be specified. Any continuous function whose iso-0 contour corresponds to the interface could be a potential candidate. However, the common best practice is to select a level-set function defined as the signed distance to the interface, i.e.

$$\varphi(\mathbf{x},t) = \begin{cases} \operatorname{dist}(\mathbf{x},\Gamma(\varphi))\,, & \mathbf{x}\in\Omega_1\,,\\ 0\,, & \mathbf{x}\in\Gamma\,,\\ -\operatorname{dist}(\mathbf{x},\Gamma(\varphi))\,, & \mathbf{x}\in\Omega_2\,, \end{cases} \tag{6.4}$$

$$\text{with:}\quad \operatorname{dist}(\mathbf{x},\Gamma(\varphi)) = \min_{\mathbf{x}_{\Gamma,\mathrm{cp}}:\,\varphi(\mathbf{x}_{\Gamma,\mathrm{cp}})=0}\|\mathbf{x}-\mathbf{x}_{\Gamma,\mathrm{cp}}\|\,,$$

where $\mathbf{x}_{\Gamma,cp}(\mathbf{x},t)\in\Gamma(\mathbf{x},t)$ is the closest point belonging to the interface from the point $\mathbf{x}$.

Why a signed distance function? Defining the level-set as a signed distance function implies that it is everywhere differentiable (except at some singular points) and that its gradients satisfy the eikonal equation,

$$\|\nabla\varphi(\mathbf{x})\| = 1\,, \quad \mathbf{x}\in\Omega\,. \tag{6.5}$$

Those properties are important for three main reasons:

- Gomes and Faugeras (2000) explains that from the theoretical viewpoint, given any surface $\Gamma\in\mathbb{R}^{D-1}$, its signed distance function is uniquely defined. Conversely, any implicit function $\varphi\in\mathbb{R}^D$ satisfying Equation 6.5 is the signed distance function to a surface plus a constant (that can be taken as zero on the surface such that it is defined as the zero iso-contour of φ). This equivalence of descriptions is convenient as it allows transposing properties of the first one into properties of the second one and vice versa.
- From a practical point of view, using a signed distance function allows assigning positive and negative values to points on opposite side of the interface, which makes it straightforward to identify the non-overlapping domains from either side of the interface:

$$\Omega_1 := \{\mathbf{x}\in\Omega : \varphi(\mathbf{x}) < 0\}\,,$$
$$\Omega_2 := \{\mathbf{x}\in\Omega : \varphi(\mathbf{x}) > 0\}\,.$$

  Moreover, for some applications, the knowledge of the distance of any

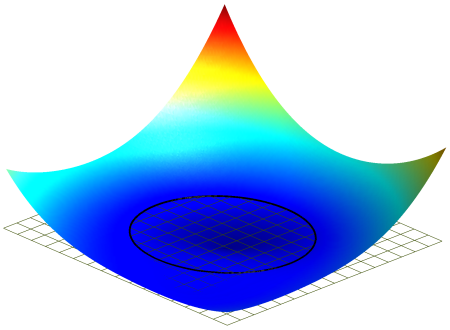


Figure 6.3 / Example of a level-set function with the same iso-0 contour as in Figure 6.1 but not satisfying the signed distance property.

point to the interface is required, and can be directly obtained from the construction of φ.

- From a numerical perspective, having a constant gradient of φ equal to unity everywhere in the domain, and especially around the interface, ensures accurate computation of secondary quantities like the normal and curvature. Without that property, regions of steep gradients could be present and compromise the accuracy of the normal, and *a fortiori* curvature, computation. This would in turn induces interface deformation if the normal and/or curvature are used to compute physical interfacial quantities (e.g. the fluxes in the jump conditions). On the contrary, the level-set function may become so flat in some regions that the determination of its zero iso-contour suffers a loss of accuracy. A function φ not satisfying the signed distance function and that could potentially lead to the two issues mentioned above is shown in Figure 6.3.

It can be concluded that initializing the level-set function as a signed distance function is a relevant choice. However, this property is not automatically maintained when the level-set is advected, for reasons that are elucidated just after. In such cases, it is common to apply a reinitilization procedure to restore the signed distance property. This is detailed in Section 6.3. At the same time, it is possible to (strongly) limit the deterioration of the signed distance property by carefully computing the velocity which all the level-set iso-contours are advected. This aspect is covered in Section 6.4.

It should be mentioned that reasons to not enforce a signed distance function exist as well, for example:

- If the evaluation of the derivatives of φ are not needed, using and advecting a level-set function which does not satisfy the signed distance property is totally valid. This approach has been followed by Marchandise and Remacle (2006) in which the need for reinitialization is thus avoided.

- In the case of the conservative level-set methods mentioned before, the level-set is on purpose not chosen to be a signed distance function in order to incorporate some physical meanings in its definition.

Interface motion

In order to update the position of the interface at every time step, the level-set advection equation needs to be solved:

$$\begin{aligned} \frac{\partial \tilde{\varphi}(\mathbf{x},t)}{\partial t} + \tilde{\mathbf{v}}_{\text{ext}}(\mathbf{x},t)\cdot\nabla\tilde{\varphi}(\mathbf{x},t) = 0\,, \quad \forall \mathbf{x}\in\tilde{\Omega}\,, \\ \tilde{\varphi}(\mathbf{x},t=0) = \tilde{\varphi}^0(\mathbf{x})\,, \end{aligned} \tag{6.6}$$

where $\tilde{\varphi}$ is the exact continuous level-set field and $\tilde{\varphi}^0$ its initial value. The continuous extended advection velocity, noted $\tilde{\mathbf{v}}_{\text{ext}}$ should, in the limit as one approaches the zero level-set, smoothly yields the velocity of the interface $\tilde{\mathbf{v}}_\Gamma$, i.e.

$$\lim_{\mathbf{x}\to\mathbf{x}_\Gamma} \tilde{\mathbf{v}}_{\text{ext}}(\mathbf{x}) = \tilde{\mathbf{v}}_\Gamma(\mathbf{x}_\Gamma)\,, \quad \mathbf{x}\in\tilde{\Omega}\,, \quad \mathbf{x}_\Gamma\in\tilde{\Gamma}\,. \tag{6.7}$$

Beyond that requirement, different approaches can be considered to extend the velocity to every points away from the interface. The approach used in this work is described in Section 6.4.

Why the level-set may lose its signed distance property? Even if the level-set field is initialized as a signed-distance function, it may progressively lose this property as it is advected. This will be surely the case if the extended velocity in the domain is arbitrary taken, for instance as the local fluid velocity. Indeed, in this case, the level-set function φ will evolve with a velocity which does not have any connection with the motion of the embedded surface Γ. There is therefore no chance that all the iso-contours of φ remain in a "signed distance arrangement" with respect to Γ. The careful choice of an extended velocity field can mitigate this potential issue. Those aspects are discussed in Section 6.4.1.

6.2.2 DG discretization

Equation 6.6 is first rewritten in conservative form:

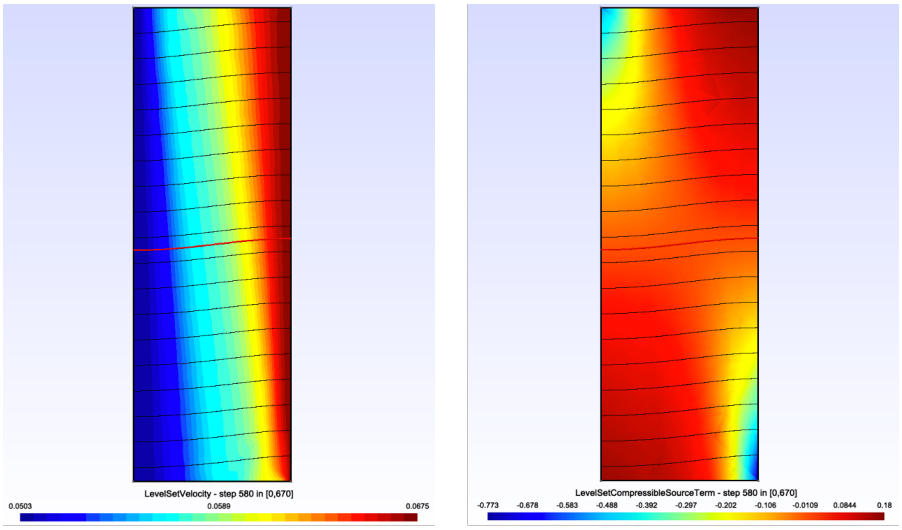
$$\frac{\partial\tilde{\varphi}}{\partial t} + \nabla\cdot(\tilde{\mathbf{v}}_{\text{ext}}\,\tilde{\varphi}) = \tilde{\varphi}\,\nabla\cdot\tilde{\mathbf{v}}_{\text{ext}}\,, \tag{6.8}$$

which defines the associated convective flux,

$$\mathbf{F}^c(\tilde{\varphi}) = \left[\tilde{v}_{\text{ext},x}\,\tilde{\varphi},\, \tilde{v}_{\text{ext},y}\,\tilde{\varphi}\right]\,, \tag{6.9}$$

and source term,

$$S(\tilde{\varphi}) = \tilde{\varphi}\,\nabla\cdot\tilde{\mathbf{v}}_{\text{ext}}\,. \tag{6.10}$$



a / Advection velocity of the level-set obtained by an extension from the iso-0.

b / Corresponding source term involving the divergence of the extended velocity.

Figure 6.4 / Illustration of the extended velocity and the associated source term in the conservative form of the level-set advection equation.

Since an extended velocity is used in the domain, this velocity field will not necessarily be divergence free and therefore the source term on the right-hand side of Equation 6.8 has to be taken into account. The presence of this non-zero source term is illustrated in Figure 6.4 on a basic configuration where the velocity in the domain is obtained by an extension from the interface.

Following the elements described in Section 5.2, the spatial discretization of the level-set governing equation is written as

$$\begin{aligned}
&\sum_{m\in\mathcal{T}} \int_{\mathcal{T}_m} w_m \frac{\partial \varphi_m}{\partial t} \,\mathrm{d}\mathbf{x} \\
&- \sum_{m\in\mathcal{T}} \int_{\mathcal{T}_m} \nabla w_m \cdot \mathbf{F}^c(\varphi_m) + w_m S(\varphi_m) \,\mathrm{d}\mathbf{x} \\
&+ \sum_{e\in\mathcal{E}} \int_{\mathcal{E}_e} \left[\hat{\mathbf{F}}^c_{\mathcal{E}}(\varphi_e^+, \varphi_e^-) \cdot \mathbf{n}_e\right] (w_e^+ - w_e^-) \,\mathrm{d}s \\
&+ \sum_{b\in\mathcal{B}} \int_{\mathcal{B}_b} \left[\hat{\mathbf{F}}^c_{\mathcal{B}}(\varphi_b^+, \varphi_b^{\mathrm{bnd}}) \cdot \mathbf{n}_b\right] w_b^+ \,\mathrm{d}s \\
&= 0 \quad \forall w \in \mathcal{W}_{p,\mathrm{LS}},
\end{aligned} \tag{6.11}$$

with the test functions w taken from the same space as the trial functions, i.e. the broken scalar-valued space given by

$$\mathcal{W}_{p,\mathrm{LS}}(\mathcal{T}) := \left\{ w \in \left[L^2(\Omega)\right] : w_{|_{\mathcal{T}_m}} \circ \sigma_m^{-1} \in \left[\mathcal{P}_p(\hat{\mathcal{T}})\right], \ \forall\, \mathcal{T}_m \in \mathcal{T} \right\}. \tag{6.12}$$

The approximate level-set solution on each element, φ_m, is expressed as a weighted sum of polynomial shape function, $\boldsymbol{\phi}_{m,-}$, whose weights are the unknown nodal degrees of freedom of the problem, $\boldsymbol{\Phi}_{m,-}$ such that

$$\boldsymbol{\Phi}_{m,-}(t) = \begin{bmatrix} \varphi_{m,1} \\ \vdots \\ \varphi_{m,n} \\ \vdots \\ \varphi_{m,N_n} \end{bmatrix}, \tag{6.13}$$

and

$$\begin{aligned} \varphi(\mathbf{x},t) &= \sum_{m=1}^{N_T} \varphi_m(\mathbf{x},t)\,, \\ \varphi_m(\mathbf{x},t) &= \sum_{n=1}^{N_n} \phi_{m,n}(\mathbf{x})\,\Phi_{m,n}(t)\,. \end{aligned} \tag{6.14}$$

The convective numerical flux in Equation 6.11 is a simple upwind flux for linear scalar advection law, i.e.

$$\hat{\mathbf{F}}^c_{\mathrm{E}}(\varphi_e^+,\varphi_e^-)\cdot\mathbf{n}_e = \mathcal{H}(\varphi_e^+,\varphi_e^-,\mathbf{n}_e) = \begin{cases} \varphi_e^-\, v_n & \text{if } v_n > 0\,, \\ \varphi_e^+\, v_n & \text{if } v_n \le 0\,, \end{cases} \quad \text{with } v_n = \mathbf{v}_{\text{ext}}\cdot\mathbf{n}_e \quad . \tag{6.15}$$

On the boundary, the same numerical flux is used, with the exterior trace of the solution set to the interior value, i.e.

$$\hat{\mathbf{F}}^c_{\mathrm{B}}(\varphi_b^+,\varphi_b^-)\cdot\mathbf{n}_b = \mathcal{H}(\varphi_b^+,\varphi_b^- = \varphi_b^+,\mathbf{n}_b)\,. \tag{6.16}$$

6.2.3 Temporal discretization

Although Chapter 10 is devoted to the temporal discretization, it is mainly focused on the fluid equations spatially discretized by the XDG method. That is why we briefly discuss here the temporal discretization of the spatially discretized DG level-set equation. In this work, we both use explicit and implicit time stepping schemes for the level-set equation. Since the implicit treatment of the fluid equations will be covered later, we choose to present the explicit treatment of the level-set equation by a means of a standard Runge-Kutta (RK) scheme. We start by recasting Equation 6.11 into the following more compact form representing a system of ODEs to be integrated in time to find the level-set nodal DOF, $\boldsymbol{\Phi}$, at the next time step:

$$\frac{\mathrm{d}\boldsymbol{\Phi}(t)}{\mathrm{d}t} + \underline{\boldsymbol{M}}^{-1}\,\mathcal{L}^{\mathrm{sp}}(\boldsymbol{\Phi}) = \mathbf{0}\,, \tag{6.17}$$

with the mass matrix $\underline{\boldsymbol{M}}$ defined in Equation 5.35 and the spatial part of the level-set residual defined as

$$\begin{aligned}\mathcal{L}^{\text{sp}} = &- \sum_{m\in\mathbb{T}} \int_{\mathbb{T}_m} \nabla\boldsymbol{\phi}_m \cdot \mathbf{F}^c(\varphi_m) + \boldsymbol{\phi}_m \, S(\varphi_m) \, \mathrm{d}\mathbf{x} \\ &\pm \sum_{e\in\mathbb{E}} \int_{\mathbb{E}_e} \boldsymbol{\phi}^{\pm}_{e,-} \cdot \mathcal{H}(\varphi^+_e, \varphi^-_e, \mathbf{n}_e) \, \mathrm{d}s \\ &+ \sum_{b\in\mathbb{B}} \int_{\mathbb{B}_b} \boldsymbol{\phi}_{b,-} \cdot \mathcal{H}(\varphi^+_b, \mathbf{n}_b) \, \mathrm{d}s \,,\end{aligned} \tag{6.18}$$

where the test functions w have been substituted by shape functions $\boldsymbol{\phi}$, following the procedure described in Section 5.2.4, and the $\hat{\bullet}$ notation has been removed from the numerical flux functions to indicate that they are now specified by Equation 6.15 and Equation 6.16. The temporal discretization of Equation 6.17 by a N_s-stage, N_s^{th}-order RK scheme from time step t_n to t_{n+1} can then be written as:

$$\begin{aligned}\bar{\boldsymbol{\Phi}}_1 &= \boldsymbol{\Phi}(t_n) \,, \\ \bar{\boldsymbol{\Phi}}_k &= \boldsymbol{\Phi}(t_n) + \sum_{i=1}^{k-1} \Delta t(t_n) \, a_{ki} \, \underline{\boldsymbol{M}}^{-1} \, \mathcal{L}^{\text{sp}}(\bar{\boldsymbol{\Phi}}_i) \,, \quad k = \{2, \dots, N_s\} \,, \\ \boldsymbol{\Phi}(t_{n+1}) &= \boldsymbol{\Phi}(t_n) + \sum_{s=1}^{N_s} \Delta t(t_n) \, \gamma_s \, \underline{\boldsymbol{M}}^{-1} \, \mathcal{L}^{\text{sp}}(\bar{\boldsymbol{\Phi}}_s) \,,\end{aligned} \tag{6.19}$$

where the time step is computed such as to comply with the CFL stability constraint,

$$\Delta t(t_n) = \min\left(\Delta t^{\text{stable}} \times \text{CFL}, \, \Delta t^{\text{user}}\right) \,, \tag{6.20}$$

with Δt^{user} the user-defined time step and Δt^{stable} the stable time step,

$$\Delta t^{\text{stable}} = \min_{\forall \mathbb{T}_m} \left[\frac{R_{\Delta x_m}}{(2p+1) \, \max_{\mathbb{T}_m} \|\mathbf{v}_{\text{ext}}\|} \right] \,,$$

with $R_{\Delta x_m}$ the inscribed element radius and p the interpolation order.

6.3 Reinitialization and computation of the closest points on the interface

6.3.1 Some background

Three main approaches exist in the literature to reinitialize the level-set function in order to recover its signed distance property:

- The first one is to solve the eikonal equation, Equation 6.5. Very efficient methods like the Fast Marching Method by J A Sethian (1996) and the

Fast Sweeping Method by Tsai et al. (2003) have been designed for that purpose.

- In the second approach, a PDE is formed from the eikonal condition to be satisfied. Three families of methods exist according to the nature of the PDE to be solved: (i) first-order hyperbolic PDE, introduced in Sussman et al. (1994), (ii) parabolic PDE, proposed by C. Li et al. (2010), and (iii) quasi-linear elliptic PDE, developed in Basting and Kuzmin (2013). In the framework of high-order methods, specifically DG methods, one can mention the work of Jibben and Herrmann (2017) proposing a hyperbolic reinitialization for the conservative LS formulation and J. Zhang and Yue (2019) solving the Hamilton-Jacobi equation with a DG method. T. Adams et al. (2019) extend the parabolic PDE-based reinitialization to DG. Finally, Utz et al. (2017) reformulate the eikonal equation as a minimization problem and solve an elliptic equation in DG.
- The last approach, proposed by David L. Chopp (2001), is to stand at each computational mesh point, find the corresponding closest point on the interface and assign to the level-set the signed distance between the mesh point and the closest point on the front. This geometrical approach directly uses the definition of the level-set function given by Equation 6.4. Anumolu and Mario F. Trujillo (2013) apply a hybrid reinitialization process by combining the closest point computation on the cells cut by the interface and the hyperbolic PDE approach in the rest of the domain. R. Saye (2014) extends the approach to high-order computation of closest points on implicit surfaces. This geometrical gradient-based high-order reinitialization process has recently gained attention with work such as the one of Ngo et al. (2022) that further develop the approach to unstructured meshes, the one of Schulze et al. (2023) that apply it in a fully Lagrangian context or the one of Henri et al. (2022a) that improve it by including a kink detection algorithm.

The closest point approach is the one adopted in this work for three main reasons: (i) the availability inside our solver of a first implementation based on this principle that could be directly exploited, (ii) the simplicity associated to the non-iterative (i.e. single-step) and geometrical features on which this approach is based, (iii) the fact that the velocity extension can be performed following exactly the same procedure.

6.3.2 Implementation

The reinitialization strategy that has been implemented is largely based on the work of Saye R. Saye (2014) in which a procedure for high-order closest points calculations to implicitly defined surfaces is presented. This new implementation extends the original one available in the `Argo` platform based on the work of Marchandise et al. (2007). The different steps involved in the reinitialization procedure are described below and illustrated in Figure 6.14.

1. A global set of sampling points approximating the interface identified by Equation 6.1 is first generated. This is accomplished by using either (i) the position of the quadrature points provided by the quadrature techniques for the iso-0 (see Section 8.2.4), or (ii) the recursive contouring algorithm presented in Remacle et al. (2007), which can be summarized as follows:

   - The elements crossed by the iso-0 of the level-set are divided recursively (with a chosen recursion level) into sub-elements and the edges of the subdivided elements are collected.
   - The values of the level-set at each vertex defining the edges are then obtained by interpolation.
   - If an edge is identified as being cut by the iso-0, i.e. if the level-set values at its vertices are of opposite sign, the approximate intersection point between the iso-0 and the edge is found by assuming a linear variation of the level-set solution in-between the two vertices:

$$\mathbf{x}_\Gamma^* = \mathbf{x}_{\text{vertex}}^1 - \varphi_{\text{vertex}}^1 \frac{\mathbf{x}_{\text{vertex}}^2 - \mathbf{x}_{\text{vertex}}^1}{\varphi_{\text{vertex}}^2 - \varphi_{\text{vertex}}^1} . \qquad [6.21]$$

   - Starting from the approximated intersection point $\mathbf{x}_\Gamma^*$, a simple Newton's procedure is finally applied to project it onto the iso-0:

$$\mathbf{x}_\Gamma \leftarrow \mathbf{x}_\Gamma - \frac{\varphi(\mathbf{x}_\Gamma)\nabla\varphi(\mathbf{x}_\Gamma)}{\|\nabla\varphi(\mathbf{x}_\Gamma)\|^2} . \qquad [6.22]$$

   The sampling point lying exactly on the iso-0 is then added to the global set of sampling points: `cloudPts` $=$ `cloudPts` $\cup\, \mathbf{x}_\Gamma$.
2. The sampling points are then stored in an Approximate Nearest Neighbor (ANN) search tree, proposed by Arya et al. (1998). It will be denoted by `searchTree` after.
3. For each computational mesh point $\mathbf{x}_q$ (interpolation or quadrature point, see after), a high-order computation of closest points on the interface is finally applied:

   - First, the previously created search tree is used to find efficiently the closest point $\mathbf{x}_{\Gamma,\text{cp}}^{\text{guess}}$ among the set of sampling points.
   - Then this point is used as a first guess to start a constrained opti-

mization problem stated as:

$$
\begin{aligned}
&\text{Starting from } \mathbf{x}_{\Gamma,\mathrm{cp}}^{\mathrm{guess}}\,,\\
&\text{Find } \mathbf{x}_{\Gamma,\mathrm{cp}} \in \mathbb{R}^d \text{ such that:}\\
&\qquad \mathbf{x}_{\Gamma,\mathrm{cp}} = \underset{\mathbf{x}\in\mathcal{F}}{\operatorname{argmin}}\, d(\mathbf{x})\\
&\text{with:}\\
&\qquad d(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{x}_q\|^2\\
&\mathcal{F} = \{\,\mathbf{x}\in\mathbb{R}^d \,|(1)\ \varphi(\mathbf{x}) = 0\,,\\
&\qquad\qquad (2)\ \boldsymbol{\xi}(\mathbf{x}) \in \hat{\mathbb{T}}\,\}
\end{aligned}
\qquad [6.23]
$$

Equation 6.23 corresponds to finding the point lying on the iso-0 of the level-set function (first constraint in the feasible region set $\mathcal{F}$) minimizing the distance (the function $d(\mathbf{x})$) between this point and the target $\mathbf{x}_q$. The second constraint in the feasible region $\mathcal{F}$ deserves some more attention. It ensures that the solution expressed in parametric coordinates (obtained with the isomorphism $\boldsymbol{\xi}(\mathbf{x})$) stays within the reference parametric element $\hat{\mathbb{T}}$. First, as it will be discussed after, this minimization problem is reformulated in parametric coordinates as all the operations in a FEM context. Second, as the level-set function is represented by elementwize polynomial functions, the process should ensure that the iterate stays within the element in which it started, otherwise the first constraint would not be valid anymore. Satisfying this second constraint, which translates into a set of inequality constraints, is not as straightforward as the first one. That is why two different approaches based on the method of Lagrange mulipliers were implemented. The first one, described below as the classical Lagrangian method, does not directly incorporate the inequality constraints into its formulation. On the other hand, the augmented Lagrangian method allows to ensure the satisfaction of all the constraints for each new iterate. Numerical experiments showed no significant improvements using the second approach and since it is more computationally expensive, the first approach is used in this work and described below.

Classical Lagrangian method with a single equality constraint In this first method, the second constraint in Equation 6.23 is not directly incorporated into the formulation, only the first one is. Thus, by virtue of the method of Lagrange, it is recast as:

$$
\nabla L(\mathbf{x},\lambda) = 0\,, \quad \text{with: } L(\mathbf{x},\lambda) = \frac{1}{2}\|\mathbf{x}-\mathbf{x}_q\|^2 + \lambda\varphi(\mathbf{x})\,,
$$

with the Lagrangian L and the Lagrange coefficient λ. The resulting non-linear system is then solved with an iterative Newton's method, starting from $\mathbf{x}_{\Gamma,\mathrm{cp}}^{\mathrm{guess}}$. An important point to highlight here is the fact that all the operations are

usually performed in the parametric coordinates in a FEM framework. The non-linear system to be solved is therefore expressed in terms of the derivatives with respect to the parametric coordinates $\boldsymbol{\xi}$ thanks to the inverse mapping, Equation 5.6, $\sigma_m^{-1}(\mathbf{x}) = \boldsymbol{\xi}$:

$$\nabla_{(\boldsymbol{\xi},\lambda)} L(\sigma_m^{-1}(\mathbf{x}), \lambda) = 0\,. \tag{6.24}$$

The linearization of this non-linear system around a certain point $(\boldsymbol{\xi}, \lambda)$ leads to the linear system giving the Newton's steps $(\Delta\boldsymbol{\xi}, \Delta\lambda)$ at each iteration:

$$\begin{bmatrix} \nabla^2_{\boldsymbol{\xi}\boldsymbol{\xi}} L(\boldsymbol{\xi},\lambda) & \nabla_{\boldsymbol{\xi}}\varphi(\boldsymbol{\xi}) \\ \nabla^T_{\boldsymbol{\xi}}\varphi(\boldsymbol{\xi}) & 0 \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{\xi} \\ \Delta\lambda \end{bmatrix} = - \begin{bmatrix} \nabla_{\boldsymbol{\xi}} L(\boldsymbol{\xi},\lambda) \\ \varphi(\boldsymbol{\xi}) \end{bmatrix}, \tag{6.25}$$

where the gradient of the Lagrangian with respect to $\boldsymbol{\xi}$ can be expressed based on the Jacobian matrix of the mapping between parametric and physical coordinates $\underline{\boldsymbol{J}}^{\sigma}$, Equation 5.7, i.e.

$$\nabla_{\boldsymbol{\xi}} L(\boldsymbol{\xi},\lambda) = \underline{\boldsymbol{J}}^{\sigma,T}(\mathbf{x}(\boldsymbol{\xi}) - \mathbf{x}_q) + \lambda \nabla_{\boldsymbol{\xi}}\varphi(\boldsymbol{\xi})\,. \tag{6.26}$$

The Hessian matrix of the Lagrangian with respect to $\boldsymbol{\xi}$ is computed by central finite differences for simplicity[1], viz.

$$\nabla^2_{\boldsymbol{\xi}\boldsymbol{\xi}} L(\boldsymbol{\xi},\lambda) = \frac{\nabla_{\boldsymbol{\xi}} L(\boldsymbol{\xi}+\epsilon,\lambda) - \nabla_{\boldsymbol{\xi}} L(\boldsymbol{\xi}-\epsilon,\lambda)}{2\epsilon}\,. \tag{6.27}$$

Assuming that $\nabla_{\boldsymbol{\xi}} L(\boldsymbol{\xi}^0, \lambda^0) \approx 0$, the initial value for the Lagrange multiplier is specified as:

$$\lambda^0 = -\frac{\underline{\boldsymbol{J}}^{\sigma,T}(\mathbf{x}^0 - \mathbf{x}_q) \cdot \nabla^T_{\boldsymbol{\xi}}\varphi(\boldsymbol{\xi}^0)}{\|\nabla_{\boldsymbol{\xi}}\varphi(\boldsymbol{\xi}^0)\|^2}\,. \tag{6.28}$$

Depending on the local shape of the level-set at which a given iterate is computed, the gradient and Hessian of the Lagrangian may be such that the Netwon's step leads to the next iterate being outside the reference element. This will of course be the case if the exact closest point can not be found inside the current element, in which case the process should be repeated on adjacent elements, as explained after. However, the exact solution could still be located inside the current element but be missed due to a too large Newton's step. Relaxation method is applied to limit the step in case the next iterate lies outside the reference element. Performing such simple damping does however not guarantee that the next iterate will exactly satisfy the equality constraint.

[1] An analytical expression of the Hessian matrix would involve the computation of second order derivatives of the mapping as well as the level-set function.

6.3.3 General algorithm and additional remarks

Algorithm 1 compiles the different steps of the reinitialization procedure. It is accompanied by the following additional remarks:

- *Considering level-set outside its parent element*: instead of enforcing the iterates to remain strictly inside the reference element, we may set a non-zero margin allowing some overlapping between adjacent elements. This is particularly useful when the exact solution is located just on the other side of an element edge. However, in high curvature regions, considering an extension of the level-set in a neighboring element, even small, may impact the accuracy.
- *Considering adjacent elements in case of non-convergence*: starting from a guess point inside a given element, the iterative process may not be able to find a solution inside this same element satisfying the prescribed tolerance. In this situation, the process is applied on the neighboring elements. One first needs to retrieve the sampling points defined on each adjacent elements, then create a local search tree and find a new guess located inside the adjacent element. The minimization method is then called starting from the guess and the loop over the adjacent elements is stopped as soon as the exact point is found.
- *Sampling elements nastily cut by the iso-0*: on elements containing a very short segment of the level-set iso-0, it may be that the only sampling points available are the intersections points between the iso-0 and the element edges. As starting the iterative process with a guess point located on an element boundary is not optimal, subdivision with a higher recursion level is done on those elements.
- *Interpolation and quadrature points*: the mesh points for which the closest point computation is performed can be selected as either the interpolation points or the quadrature points. If the interpolation points are chosen, the reinitialized level-set field is directly obtained since it is defined by the values at the nodal degrees of freedom. On the other hand, when the quadrature points are selected, the reinitialized field is obtained by an L^2-projection of the quadrature points values. More precisely, if $\mathbf{U}_{\mathrm{QP}} \in \mathbb{R}^{N_q \times 1}$ are the known values on the quadrature points of an element and $\mathbf{U}_{\mathrm{IP}} \in \mathbb{R}^{N_n \times 1}$ are the unknowns nodal values at the interpolation points, we want to solve the following overdetermined system of equations:

$$\mathbf{U}_{\mathrm{QP}} = \underbrace{\begin{bmatrix} \phi_1(\boldsymbol{\xi}_1^Q) & \cdots & \phi_{N_n}(\boldsymbol{\xi}_1^Q) \\ \vdots & & \vdots \\ \phi_1(\boldsymbol{\xi}_{N_q}^Q) & \cdots & \phi_{N_n}(\boldsymbol{\xi}_{N_q}^Q) \end{bmatrix}}_{\underline{\boldsymbol{A}}} \cdot \mathbf{U}_{\mathrm{IP}} \,.$$

This can be reformulated as a minimization problem in the DG L_2 norm,

$$\min_{\mathbf{U}_{\mathrm{IP}}} J(\mathbf{U}_{\mathrm{IP}}) \quad \text{with } J(\mathbf{U}_{\mathrm{IP}}) = \|\underline{\boldsymbol{A}}\,\mathbf{U}_{\mathrm{IP}} - \mathbf{U}_{\mathrm{QP}}\|^2_{L_2}\,.$$

This problem can be transformed into a system of linear equations by taking the gradient of the functional J with respect to $\mathbf{U}_{\mathrm{IP}}$ such that the unknowns $\mathbf{U}_{\mathrm{IP}}$ are given by

$$\underbrace{\int_{\mathrm{T}_m} \underline{\boldsymbol{A}}^T \underline{\boldsymbol{A}}\,\mathrm{d}\mathbf{x}}_{\underline{\boldsymbol{M}}_m}\,\mathbf{U}_{\mathrm{IP}} = \int_{\mathrm{T}_m} \underline{\boldsymbol{A}}^T\,\mathrm{d}\mathbf{x}\,\mathbf{U}_{\mathrm{QP}}\,. \tag{6.29}$$

6.3.4 Verification test case

A test case from Anumolu and Mario F. Trujillo (2013) is selected to illustrate the convergence rate of the reinitialization procedure. It consists of two circles of different radii with iso-contours initially specified by an exponential function,

$$\begin{aligned}\varphi_{\mathrm{init}}(\mathbf{x}) = \min\{\ &\exp((x-x_a)^2+(y-y_a)^2) - \exp(r_a^2),\\ &\exp((x-x_b)^2+(y-y_b)^2) - \exp(r_b^2)\ \}\,,\end{aligned}$$

with the centers and the radii of the circles set as $(x_a, y_a) = (-1, 1)$, $r_a = 0.5$, $(x_b, y_b) = (0.5, -0.5)$, $r_b = \pi/3$. The initial level-set function is shown in Figure 6.5a. The reinitialization procedure described above is applied a single time, with the interpolation points as "target" points. The level-set iso-contour field after the application of a single reinitialization cycle is shown in Figure 6.5b. The signed distance property is recovered on the narrow band region while the level-set values are set to a constant value outside (the narrow band concept is presented in Section 6.5). In Figure 6.6, we plot the L_2 and L_∞ errors between the reinitialized level-set field, φ, and the exact analytical expression, φ_{exact}, satisfying the signed distance property:

$$\varphi_{\mathrm{exact}}(\mathbf{x}) = \min\left\{\sqrt{(x-x_a)^2+(y-y_a)^2} - r_a,\ \sqrt{(x-x_b)^2+(y-y_b)^2} - r_b\right\}\,.$$

The errors are computed as follows:

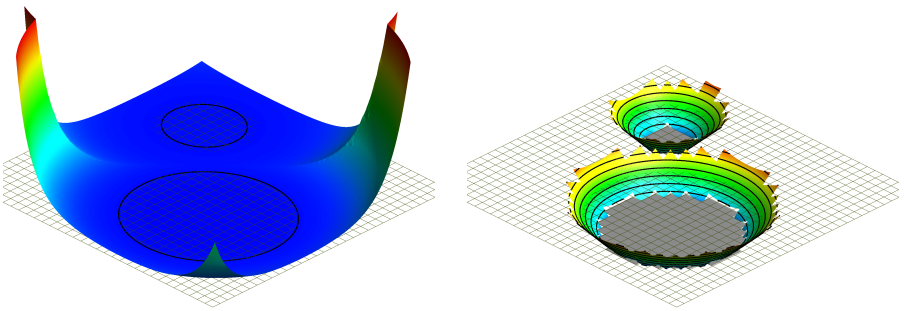
$$\|\varphi - \varphi_{\mathrm{exact}}\|^2_{L_2(\Omega)} = \sum_{m\in\mathrm{T}} \int_{\mathrm{T}_m} \left(\varphi_m(\mathbf{x}) - \varphi_{\mathrm{exact}}(\mathbf{x})\right)^2 \mathrm{d}\mathbf{x}\,, \tag{6.30}$$

$$\|\varphi - \varphi_{\mathrm{exact}}\|_{L_\infty(\Omega)} = \max_{\forall \mathrm{T}_m\in\mathrm{T}} \left(\max_{\forall \mathbf{x}_q\in\mathbf{x}^{\mathrm{Q}}} |\varphi_m(\mathbf{x}_q) - \varphi_{\mathrm{exact}}(\mathbf{x}_q)|\right)\,, \tag{6.31}$$

where $\mathbf{x}^{\mathrm{Q}}$ is the set of quadrature points defined in the element (see Equation 5.38). The results confirm the high-order behavior of the reinitialization procedure since the errors decrease with the expected theoretical rate of $p+1$

Algorithm 1 Reinitialization by high-order computation of the closest point, $\mathbf{x}_{\Gamma,\mathrm{cp}}$, on the iso-0 of the level-set, φ, for each target mesh point, $\mathbf{x}_q$. The process is initialized with the closest point in the cloud of sampling points `cloudPts` that is found by a search tree `searchTree`. The reinitialized level-set field $\varphi^{\mathrm{reinit}}$ as well as the set of exact closest points are returned.

procedure ComputeClosestPoint(φ, `cloudPts`, `searchTree`)
  for flagged elements $\mathbb{T}_m$ **do** ▷ Full domain or narrow band
    for target points $\mathbf{x}_q \in \mathbb{T}_m$ **do** ▷ Interpolation or quadrature points
      $\mathbf{x}_{\Gamma,\mathrm{cp}}^{\mathrm{guess}} \leftarrow$ `searchTree`$(\mathbf{x}_q,$ `cloudPts`$)$ ▷ Find closest point to $\mathbf{x}_q$ among `cloudPts`
      $\mathbf{x}_{\Gamma,\mathrm{cp}}^{*}$, Success $\leftarrow$ `ComputeCP`$(\mathbf{x}_q, \mathbf{x}_{\Gamma,\mathrm{cp}}^{\mathrm{guess}}, \varphi)$ ▷ Equation 6.24
      if Success **then** ▷ Convergence of method & $\mathbf{x}_{\Gamma,\mathrm{cp}}^{*} \in \hat{\mathbb{T}}$
        $\mathbf{x}_{\Gamma,\mathrm{cp}} \leftarrow \mathbf{x}_{\Gamma,\mathrm{cp}}^{*}$
      else ▷ Non-convergence of method or $\mathbf{x}_{\Gamma,\mathrm{cp}}^{*} \notin \hat{\mathbb{T}}$
        for cut adjacent elements of $\mathbb{T}_m$, i.e. $\{\mathbb{T}_m^{\mathrm{adj}}\}$ **do**
          `searchTree` $\leftarrow$ `CreateSearchTree(localCloudPts)` ▷ Cloud in $\mathbb{T}_m^{\mathrm{adj}}$
          $\mathbf{x}_{\Gamma,\mathrm{cp}}^{\mathrm{guess}} \leftarrow$ `searchTree`$(\mathbf{x}_q,$ `localCloudPts`$)$ ▷ Closest guess $\in \mathbb{T}_m^{\mathrm{adj}}$ to $\mathbf{x}_q$
          $\mathbf{x}_{\Gamma,\mathrm{cp}}^{*}$, Success2 $\leftarrow$ `ComputeCP`$(\mathbf{x}_q, \mathbf{x}_{\Gamma,\mathrm{cp}}^{\mathrm{guess}}, \varphi^{\mathrm{adj}})$ ▷ Equation 6.24
          if Success2 **then** ▷ Convergence of method & $\mathbf{x}_{\Gamma,\mathrm{cp}}^{*} \in \hat{\mathbb{T}}$
            $\mathbf{x}_{\Gamma,\mathrm{cp}} \leftarrow \mathbf{x}_{\Gamma,\mathrm{cp}}^{*}$
            break
          end if
        end for
      end if
      $\varphi^{\mathrm{reinit}} \leftarrow \mathrm{sign}(\varphi)\, \|\mathbf{x}_{\Gamma,\mathrm{cp}} - \mathbf{x}_q\|$ ▷ Equation 6.4
    end for
  end for
  return $\mathbf{x}_{\Gamma,\mathrm{cp}}$, $\varphi^{\mathrm{reinit}}$
end procedure



a / Initial field with the iso-0 contours.

b / Reinitialized field with some iso-contours on the narrow band.

Figure 6.5 / Dual-circle reinitialization test case - Level-set fields with the underlying grid.

(p being the polynomial interpolation order used to discretize the level-set).

6.4 Velocity extension

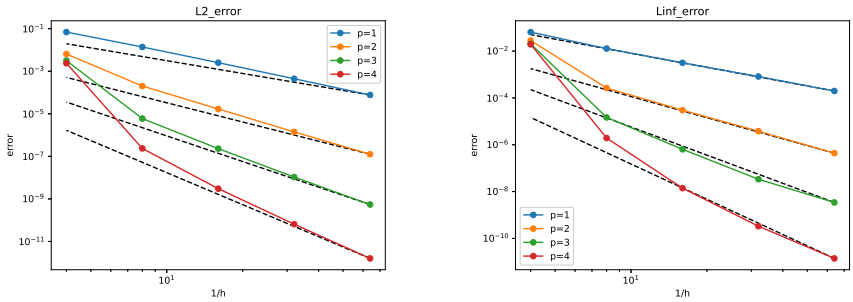
6.4.1 Some background

In order to advect the level-set function, which is a scalar field defined all over the computational domain, an advection velocity is required at every computational points (and not only on the interface). Peng et al. (1999) remark that for some applications for which the motion occurs with a constant speed or with a self-generated velocity like mean curvature flows, the advection velocity is defined globally or at least near the interface. On the other hand, for the kind of applications targeted in this work governed by externally generated velocity fields, the advection velocity has only a physical meaning on the interface and not away from it. Therefore, a choice on which value to give to $\mathbf{v}_{\text{ext}}$ in Equation 6.6 has to be made. The are then three possible options:

- The most direct approach to define the advection velocity away from the interface is to use the local fluid velocity, $\mathbf{v}_{\text{fluid}}$, evaluated at the same position at which the level-set field is evaluated, i.e.

$$\mathbf{v}_{\text{ext}}(\mathbf{x}, t) = \mathbf{v}_{\text{fluid}}(\mathbf{x}, t)\,. \qquad [6.32]$$

  This choice is often made in the literature in combination with a frequent reinitialization (every few time steps). Indeed, without reinitialization, the choice of this extended velocity field is very likely to not preserve the signed distance property of the level-set. More precisely, Mario F



a / L_2-norm error, Equation 6.30. **b /** L_∞-norm error, Equation 6.31.

Figure 6.6 / Dual-circle reinitialization test case - Spatial convergence study for different interpolation orders p and mesh sizes h. The error is computed from the analytical reinitialized level-set function. The rates of convergence tend to the theoretical rates $p+1$ (dashed lines).

Trujillo et al. (2017) show that this deviation depends on the nature of the flow field, regardless of the method used to advect the level-set. They demonstrate that any transported function will have its gradient and derivatives of higher order stretched under the presence of a flow with a non-zero strain rate tensor.

- The second approach starts from the requirements of (i) building an advection velocity which coincides with the true physical interface velocity on the interface, according to Equation 6.9, and (ii) at the same time preserving the signed distance property of the level-set field as it gets advected, i.e. satisfying Equation 6.5 in all points. Adalsteinsson and J. Sethian (1999) show that those requirements can be mathematically translated in

$$\begin{aligned} \nabla \mathbf{v}_{\text{ext}}(\mathbf{x},t) \cdot \nabla \varphi(\mathbf{x},t) &= 0 \qquad && \mathbf{x} \in \Omega\,, \\ \mathbf{v}_{\text{ext}}(\mathbf{x},t) &= \mathbf{v}_\Gamma(\mathbf{x},t) && \mathbf{x} \in \Gamma\,. \end{aligned} \tag{6.33}$$

The proof that building an extended velocity based on Equation 6.33 ensures conservation of the signed distance property can be found e.g. in Utz (2018, App. A). Equation 6.33 can be interpreted geometrically as the extended velocity field being extended away from the interface such that it remains constant along the normal to the interface directions. In other words, we can extrapolate the interface velocity in the domain such that at each grid point, the value of the interface velocity at the closest point on the front is used as the extension velocity at that point, viz.

$$\mathbf{v}_{\text{ext}}(\mathbf{x},t) = \mathbf{v}(\mathbf{x}_{\Gamma,\text{cp}},t)\,. \tag{6.34}$$

This choice was first adopted by Malladi et al. (1995). In conclusion, the iso-0 contour of the level-set function is driven by the physical interface

velocity of the problem at hand and its motion will define, by propagation, the evolution of all the other level-set contours. Hence, it avoids the non-physical meaning of the velocity away from the interface present in the first approach. Moreover, it preserves the signed distance property of the initial level-set field.

- The last possibility is to add a source term proportional to the level-set field in the advection equation. The expression of this source term is such that it is zero on the iso-0 and that it contributes to satisfy the eikonal equation during the advection away from the interface. This approach has been proposed by Ovsyannikov et al. (2012) and Sabelnikov et al. (2014).

In this thesis, the second approach has been followed for its aptitude to conserve the signed distance property (contrary to the first approach) and for its direct connection with the computation of the closest point performed during the reinitialization procedure. However, it must be said that even with this extended velocity field, numerical errors that can accumulate over time will tend to deteriorate the signed distance property and periodic reinitialization steps might be needed. This was pointed out by Smuda (2021, Sec. 3.6) in the context of a DG method. Probably even more important than the signed distance preservation is the accuracy with which the extended velocity is computed. Indeed, in the FEM framework we are using, there will be no velocity values directly stored on points lying on the interface. Instead, the level-set iso-0 contour will be advected by velocities at interpolation or quadrature points located inside the elements in the vicinity of the interface. And the velocities at those points will be obtained by the extension procedure. Any inaccuracy in the extension will therefore be directly felt in terms of mass conservation and spurious interface deformations. That is an additional reason motivating the use of the same high-order closest point computation for the velocity extension.

Finally, it should be mentioned that instead of relying on a closest point computation to extend the velocity, Equation 6.33 can also be solved. This has been done by the Fast Marching Method of Adalsteinsson and J. Sethian (1999) and David L. Chopp (2009). Similarly to the eikonal equation, Equation 6.33 can also be reformulated as a hyperbolic PDE. This has been done for instance in Peng et al. (1999). Utz and Kummer (2018) rely instead on an elliptic formulation of the problem (in a DG framework).

6.4.2 Implementation

Since the velocity is extended into the computational domain based on Equation 6.34, the positions of the closest points on the interface for every computational point need to be computed. The exact same procedure as Algorithm 1 is followed and to avoid doing the process twice, the extended velocity field is collected in the same loop as the reinitialized level-set field. While the reinitialized field can be directly obtained from the distance between the grid point and the

interface closest point, the extended velocity requires two special considerations (the two last sketches in Figure 6.14):

- The first one is the communication of the quantities computed by the fluid solver to the level-set solver, which are separated from each other and called in a segregated way. Further information about that coupling is provided in Section 10.3.2. We can just mention here that the fluid quantities required by the level-set are stored in a dedicated data structure at the end of every time step of the fluid solver and that those data are accessible by the level-set solver. Thus, we can assume that the fluid solution vectors in both phases, $\mathbf{u}_1$ and $\mathbf{u}_2$, are available at every interpolation points of the mesh during the extension procedure. After having determined the position of the closest point, the fluid solutions are then interpolated there by using the shape functions and the nodal values in the element to which this closest point belongs:

$$\mathbf{u}_{k,\Gamma} = \boldsymbol{\phi}_{m,-}^{T}(\mathbf{x}_{\Gamma,\mathrm{cp}})\underline{\boldsymbol{U}}_{m,-,k,-}\,, \tag{6.35}$$

  where $\underline{\boldsymbol{U}}_{m,-,k,-}$ contains the nodal values of the fluid solution in phase k over the element m which is assumed to be the one containing the closest point.
- The second point of attention concerns the way the interface velocity is evaluated based on the interpolated fluid quantities, in particular in a two-phase flow scenario. If there is no mass transfer across the interface, the interface velocity, $\mathbf{v}_{\mathrm{LS},\Gamma}$, is either obtained as an arithmetic average of the fluid velocities or as a density-weighted average:

$$\mathbf{v}_{\mathrm{LS},\Gamma} = \begin{cases} \dfrac{\mathbf{v}_{1,\Gamma}+\mathbf{v}_{2,\Gamma}}{2}\,, \\ \dfrac{\rho_{1,\Gamma}\mathbf{v}_{1,\Gamma}+\rho_{2,\Gamma}\mathbf{v}_{2,\Gamma}}{\rho_{1,\Gamma}+\rho_{2,\Gamma}}\,. \end{cases} \tag{6.36}$$

  As explained in Section 3.3, the absence of phase transition translates into the equality of the normal velocities from both sides of the interface. If there is no velocity slip phenomenon, all the fluid velocity components should be continuous at the interface. Smuda (2021, Sec. 3.6) notes that in practice, this condition is only weakly enforced in the fluid solver (see Section 9.2), motivating the use of an averaging between both sides. When phase transition takes place, there is a net mass flux across the interface and the velocities are not continuous anymore. Starting from the definition of the mass flux across the interface, Equation 3.16, and assuming the continuity of tangential velocities, the interface velocity computed from the kth-phase is given by

$$\mathbf{v}_{k,\Gamma}^{*} = \mathbf{v}_{k,\Gamma} - \frac{\dot{m}_{\Gamma}}{\rho_k}\mathbf{n}_{\Gamma}\,, \tag{6.37}$$

  which leads to the definition of the interface velocity, following the two

averaging above:

$$\mathbf{v}_{\mathrm{LS},\Gamma} = \begin{cases} \frac{\mathbf{v}^*_{1,\Gamma}+\mathbf{v}^*_{2,\Gamma}}{2}, \\ \frac{\rho_{1,\Gamma}\mathbf{v}^*_{1,\Gamma}+\rho_{2,\Gamma}\mathbf{v}^*_{2,\Gamma}}{\rho_{1,\Gamma}+\rho_{2,\Gamma}}. \end{cases} \quad [6.38]$$

The mass flux is evaluated by calling the evaporation model, Equation 3.32, i.e.

$$\dot{m}_\Gamma = \dot{m}^{\mathrm{model}}_\Gamma(\mathbf{u}_{1,\Gamma}, \mathbf{u}_{2,\Gamma}).$$

Similarly to the reinitialization, the extended velocity can be computed either at the interpolation points or at the quadrature points inside the domain. For the quadrature points, the velocity can directly be injected into the advection equation since the DG discretization needs the values at the quadrature points to perform the integrals of the variational formulation. If the interpolation points are chosen, the velocity field is interpolated to the volume and face quadrature points in a second step. Note that two possibilities are made available in our solver: (i) compute both the reinitialized level-set field and extended velocity at the same set of computational points (interpolation or quadrature points), (ii) compute the two fields on distinct sets of target points.

The several steps explained above are summarized in Algorithm 2. We recall here that those steps are in fact done along with the reinitialization process and not in a separated loop.

Algorithm 2 Extension of the velocity of the iso-0 of the level-set φ to each target mesh point $\mathbf{x}_q$. The velocity of a point in the domain is taken equal to the velocity at the closest point on the interface $\mathbf{x}_{\Gamma,\mathrm{cp}}$, determined from the provided fluid states $\mathbf{u}_k$ in each phase k. The extended velocity field over the domain $\mathbf{v}_{\mathrm{ext}}$ is returned.

procedure ComputeExtendedVelocity($\varphi, \mathbf{x}_{\Gamma,cp}, \underline{\boldsymbol{U}}^{\mathrm{fluid}}$)
  for flagged elements $\mathbb{T}_m$ **do** ▷ Full domain or narrow band
    for target points $\mathbf{x}_q$ **do** ▷ Interpolation or quadrature points
      $\mathbf{u}_{k,\Gamma} = \boldsymbol{\phi}^T_{m,-}(\mathbf{x}_{\Gamma,\mathrm{cp}})\underline{\boldsymbol{U}}^{\mathrm{fluid}}_{m,-,k,-}$ ▷ Interpolate fluid states to CP, Equation 6.35
      $\mathbf{v}_{\mathrm{ext}} \leftarrow$ `getVelocity`$(\mathbf{u}_{1,\Gamma}, \mathbf{u}_{2,\Gamma})$ ▷ Compute velocity from fluid states, Equation 6.36 or Equation 6.38
    end for
  end for
  return $\mathbf{v}_{\mathrm{ext}}$
end procedure

6.4.3 Verification test cases

For the verification test cases about the velocity extension, three error measures are introduced in addition to the L_2 error of Equation 6.30:

- Signed distance error: it measures the degree to which the numerical solution satisfies the eikonal equation, i.e. the signed distance property of the level-set field,

$$(E_{SD})^2 = \sum_{m \in \mathbb{T}} \int_{\mathbb{T}_m} (\|\nabla \varphi^h\| - 1)^2 \, \mathrm{d}\mathbf{x} \,. \qquad [6.39]$$

- Interface error: it measures the deviation of the numerical iso-0 level-set contour with respect to the analytical one and it is evaluated by integrating the computed solution along the analytical position of the iso-0,

$$(E_{Int})^2 = \sum_{m \in \mathbb{T}^\Gamma} \int_{\Gamma(\varphi_{\text{exact}})} \varphi^2 \, \mathrm{d}s \,, \qquad [6.40]$$

  where $\mathbb{T}^\Gamma$ designates the elements crossed by the iso-0 (see Section 8.2.1) and $\Gamma(\varphi_{\text{exact}})$ is the exact interface.
- Volume error: it measures the volume loss, in percentage, of the numerical level-set field from an analytical solution

$$E_{Vol} = \sum_{m \in \mathbb{T}^{1,*}} \frac{\left| \int_{T_m \cup \Omega_1(\varphi_{\text{exact}})} \mathrm{d}\mathbf{x} - \int_{T_m \cup \Omega_1(\varphi)} \mathrm{d}\mathbf{x} \right|}{\int_{T_m \cup \Omega_1(\varphi_{\text{exact}})} \mathrm{d}\mathbf{x}} \times 100 \,, \qquad [6.41]$$

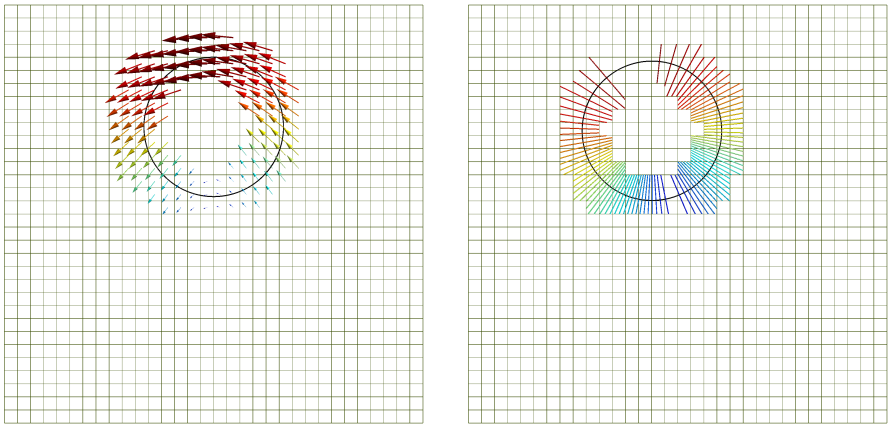
  where $\mathbb{T}^{1,*}$ designates the elements (cut or not) contained on the negative side of the interface, $T_m \cup \Omega_1(\varphi_{\text{exact}})$ is the element area contained in the negative side of the interface determined by the exact iso-0 contour.

Both the integrals over the iso-0 (defined by the exact analytical expression) in Equation 6.40 and the one in the cut elements in Equation 6.41 are evaluated based on dedicated quadrature techniques presented in Section 8.2.4. Three moving interface test cases where velocity extension is applied are considered. For all of them, the extended velocity field is computed on the interpolation points in the domain and then interpolated on the quadrature points for the evaluation of the integrals during the weak form assembly.

Rotating circle The first test case chosen here comes from Utz (2018, Sec.7.3.1). It is about the motion of a circular contour of radius 1 centered initially at $(0.0, 1.25)$ in a flow field which is rotating around the point $(0.0, 0.0)$ outside the circle. The velocity field has an angular velocity equal to 1 and is given in Cartesian coordinates by

$$\mathbf{v}(\mathbf{x}) = (-y, x) \,.$$

The problem is discretized on the domain $\Omega = (-3, 3)^2$ on structured meshes with different mesh sizes. The implicit multi-step ESDIRK43 time-stepping scheme is chosen with $\Delta t = 5e-4$ and the simulations are run until $t = 0.3$. The exact solution can be found using standard rotational transformation depending



a / Initial configuration with the prescribed velocity field on the narrow band.

b / Final configuration with the extended velocity iso-lines, perpendicular to the iso-0.

Figure 6.7 / Rotating circle test case – Extended velocity field, position of the iso-0 contour and underlying structured mesh.

on time:

$$\begin{bmatrix} x^* \\ y^* \end{bmatrix} = \begin{bmatrix} \cos(t) & \sin(t) \\ -\sin(t) & \cos(t) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\varphi_{\text{exact}}(\mathbf{x}, t) = \sqrt{(x^*)^2 + (y^* - 1.25)} - 1.0\,.$$

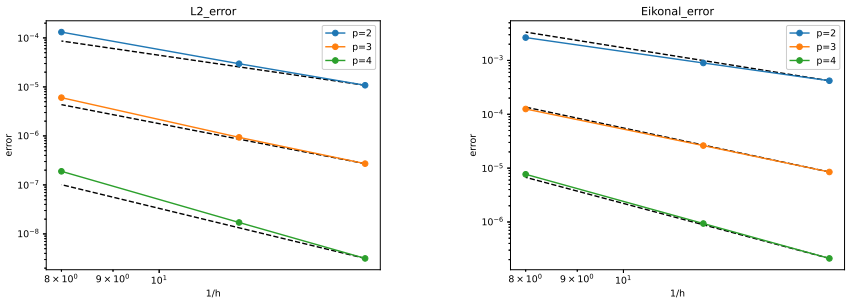
No reinitialization is performed to test the ability of the extension procedure to preserve the initial signed distance property of the level-set function. In Figure 6.7a, the initial velocity field and the position of the iso-0 are represented. In Figure 6.7b, the iso-contours of the level-set over the narrow band at the final time are drawn as well as the iso-contours of the extended velocity field. As it can be observed, the velocity iso-contours are straight and normal to the iso-0, indicating that the velocity of the iso-0 is correctly extended to the mesh points. This is confirmed on the convergence graphs in Figure 6.8 where the L_2 and the signed distance errors between the analytical expression of the level-set at the final time and the computed solution follow the expected trend.

Circle sheared by a channel flow This test case is inspired from Mousavi (2014, Sec. 4.3.4). It involves an initially circular interface placed in a channel with the equation

$$\varphi(\mathbf{x}, t = 0) = \sqrt{(x - x_c)^2 + (y - y_c)^2} - R_0\,,$$

with the center $(x_c, y_c) = (0.3, 0.0)$ and the initial radius $R_0 = 0.15$. Under the effect of a Poiseuille flow velocity profile,

$$v_x(y) = 1.5\,U_{\text{avg}}\,(1.0 - (4y^2)/H)\,,$$



a / L2-norm error measure, Equation 6.30.

b / Signed distance error measure, Equation 6.39.

Figure 6.8 / Rotating circle test case – Spatial convergence study for different interpolation orders p and mesh sizes h. The error is computed from the analytical level-set function at the final time $t = 0.3$ after 600 iterations. The rates of convergence tend to the theoretical rates $p + 1$ (dashed lines).

with $U_{\text{avg}} = 1.0\text{m/s}$ and the channel width $H = 0.5\text{m}$, the interface is convected and deformed by the external velocity field. The value of this velocity field is extended from the iso-0 to the mesh points such that the iso-contours maintain a signed distance arrangement. The analytical expression of the level-set as a function of time is derived in Mousavi (2014, App. A) and reads

$$\varphi_{\text{exact}}(\mathbf{x}, t) = \sqrt{(x - v_x\, t - x_c)^2 + (y - y_c)^2} - R_0\,. \qquad [6.42]$$

The rectangular domain $\Omega = [0, -0.25] \times [2.5, 0.25]$ is discretized by an unstructured mesh containing 3808 quadrilateral elements. The implicit multi-step ESDIRK43 time-stepping scheme is chosen with $\Delta t = 1.25e - 3$ and the simulations is run until $t = 1.25$. Reinitialization is applied every 100 time steps. An interpolation order of $p = 3$ is selected. The iso-0 level-set contours at different times along the simulation are drawn in Figure 6.9. We can observe that the iso-0 contours are globally undergoing the expected deformation. However, we can notice some small undesired instabilities appearing in the elongated regions, which impacts in turn the L_2 norm and the signed distance error measures listed in Table 6.1. The volume conservation remains however acceptably low. The cause of these spurious interface deformations is discussed at the end of the section.

Circle deformed by a vortex in a box This last test case, which is a standard benchmark, involves the deformation of an initially circular interface due to a prescribed velocity field corresponding to a periodic swirling flow. The

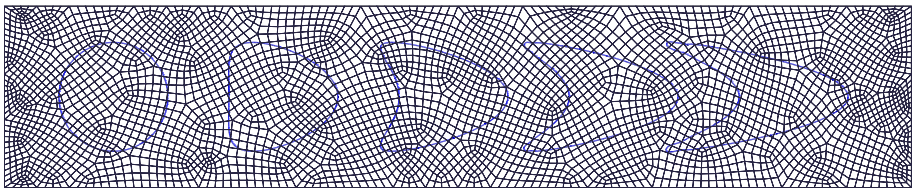


Figure 6.9 / Channel flow test case – snapshots of the iso-0 of the level-set at time $t = 0, 0.5, 0.75, 1, 1.25$s, being advected and deformed by a fully developed channel flow velocity profile.

Table 6.1 / Velocity extension test cases – values of the different error measures considered for the two last test cases.

	$\|.\|_{L_2}$	E_{SD}	E_{Int}	E_{Vol} (%)
Channel flow	$1.2 \cdot 10^{-2}$	$3.4 \cdot 10^{-2}$	$9.8 \cdot 10^{-4}$	0.69
Vortex-in-a-box	$4.0 \cdot 10^{-4}$	$7.6 \cdot 10^{-3}$	$1.2 \cdot 10^{-3}$	3.52

divergence free initial velocity field is defined as

$$v_x(x,y) = -\sin^2(\pi x)\sin(2\pi y)\,,$$
$$v_y(x,y) = \sin^2(\pi y)\sin(2\pi x)\,.$$

The interface is stretched by the vortical velocity field and develops filaments around the vortex center. The velocity field is reversed at $t = 1$ and it should recover its original shape at the final time $t = 2$. The $\Omega = [0,1] \times [0,1]$ square domain is discretized by an unstructured mesh containing 1560 quadrilateral elements. The implicit multi-step ESDIRK43 time-stepping scheme is chosen with $\Delta t = 1.0e - 3$. Reinitialization is applied every 100 time steps. An interpolation order of $p = 4$ is selected. The iso-0 level-set contours at different times along the simulation are drawn in Figure 6.10. The interface stretches and goes back to its initial shape as expected. However, some small undesired instabilities can be first observed in the filament section at maximum deformation just before the velocity field reverses. Those instabilities are then transported during the reversal phase and are still present on the final interface shape. A possible remedy to this problem is discussed just below. Looking at the error measures in Table 6.1, we can say that they remain relatively low, even if the volume conservation is not as good as in the previous test case.

Conclusions from the test cases As a conclusion to this series of test cases, we can say that the velocity extension procedure achieves its high order features on well-resolved and simple geometries like the rotating circle. It also allows convecting reasonably well the interface under important deformations, which is supported by the relatively low values of the four errors considered. However,

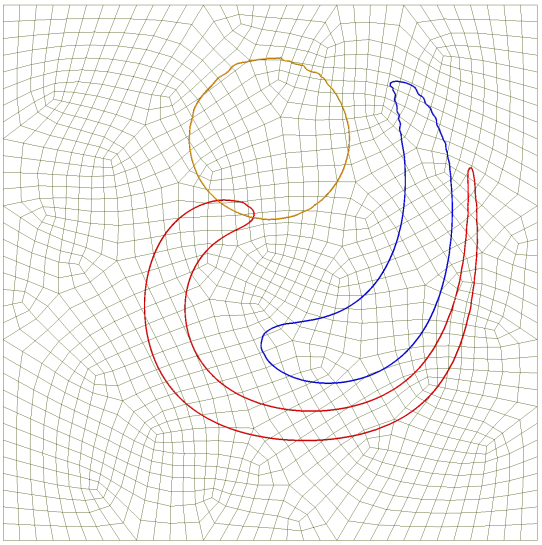


Figure 6.10 / Vortex-in-a-box test case – snapshots of the iso-0 of the level-set at time $t = 1.0$ (red), 1.5 (blue), and 2.0 (orange), being stretched by a reversing vortical velocity field.

as the two last test cases revealed, some spurious interface oscillations may appear in regions with high curvature or with a filament shape. We identify the two main causes of these inaccuracies as being:

- the under-resolved nature of the high curvature interface regions which leads to inaccurate extended velocity field and a subsequent deformation of the interface. Mario F Trujillo et al. (2017) note that for regions with high ratio of local radius of curvature to mesh spacing, reinitializing the level-set could even deteriorate the level-set shape.
- the presence of kinks in the level-set field, which are typically the most severe in the filaments regions of the two test cases studied above. Ervik et al. (2014) highlight the singularities of the level-set derivatives in the vicinity of those kinks. Henri et al. (2022a) identify them as being ill-defined regions affecting the accuracy of the gradient-based descent techniques to compute the closest points on the interface.

The two remedies to those issues would be to (i) include mesh adaptation driven by the interface resolution requirements or rely on a hybrid approach using Lagrangian markers to resolve the interface, (ii) detect and treat the kinks during the closest point computation. Those two potential remedies have not been investigated as part of this work and are spotted as important perspectives in Section 13.2.

6.5 The narrow band strategy

6.5.1 Some background

So far its was implicitly assumed that the advection, reinitialization and extension operations associated to the level-set were performed on the entire computational domain, in the same way as the fluid equations are solved. Even if by construction the level-set function is defined over the D-dimensional domain, its sole purpose (at least in our case) is to provide information about the $D-1$-dimensional interface to the fluid solver, namely its position, its speed and its geometrical quantities. Thus, advecting the level-set iso-contours in the entire domain represents an unnecessary effort, only those near the zero iso-contour corresponding to the interface need to be updated. This observation led to the development of narrow band level-set methods, first introduced in David Layne Chopp (1991) and studied extensively in David Adalsteinsson and James A Sethian (1995). A recent application of narrow-banding for level-set in a high-order parallel computational environment can e.g. be found in Appel et al. (2021).

The width of the narrow band is problem-dependent but the following remark formulated in T. Adams (2020, Sec. 3.2.2) holds in general. Whether for reasons of accuracy, stability or both, the interface motion during one time step is expected to remain confined in the elements cut by the interface at the current time and their close neighbors. Thus, the operations that determine the level-set evolution can be limited to these elements, forming a so-called narrow band around the iso-0 contour. As indicated also in T. Adams (2020), computational efficiency is not the only reason to adopt narrow-banding. Because of the signed-distance property of the level-set that needs to be maintained, there will always be some areas in the domain where the level-set function is singular. In such regions, the solution to the closest point problem seen before is not unique. Moreover, evaluating level-set derivatives (to obtain the normal e.g.) will produce oscillatory results that may lead to the failure of the method. If the mesh is sufficiently refined, it is possible to keep those singularities regions outside the narrow band and therefore avoid any issues.

6.5.2 Implementation

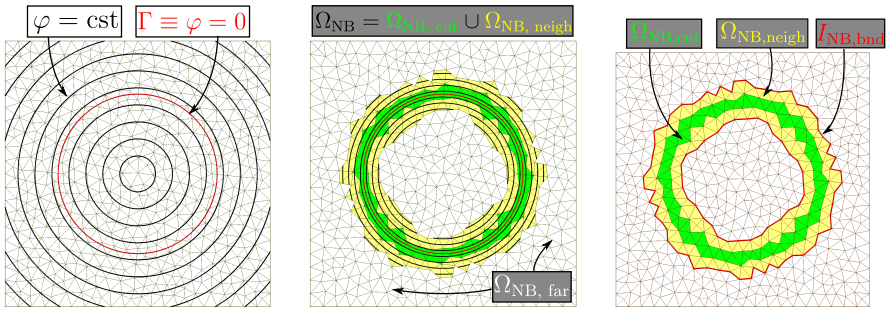
Figure 6.11 illustrates the narrow band concept and the notations used in the following. During the initialization and at the end of each time step, the building of the narrow band is carried out as follows:

1. *Flagging of cut elements* $\Omega_{\text{NB, cut}}$: a loop on all the elements is done and on each on them, the nodal degrees of freedom of the level-set solution are

collected. If the minimum and maximum values among the interpolation points values are of opposite sign, the element is flagged as cut.

2. *Flagging cut-neighbors elements* $\Omega_{\text{NB, neigh}}$: four kinds of neighbors elements can be flagged: (a) the elements sharing a vertex with one of the cut elements (single layer), (b) the ones sharing an edge with one of the cut elements (single layer), (c) the ones sharing a vertex with one of the elements belonging to category (a) (double layer), and (d) the ones sharing an edge with one of the elements belonging to category (b) (double layer). For categories (a) and (b), a loop on the elements flagged as cut is done and on each of them, either the vertices (a) or the edges (b) defining the element are collected. Based on that, the elements can be flagged. For the double layer categories (c) and (d), the elements satisfying the criteria of categories (a) or (b), respectively, are first collected. Then a loop is done on them and the same flagging process is applied (checking for each element that it has not been added yet to the list of course).
3. *Assembling of narrow band elements* Ω_{NB}: the narrow band elements are simply flagged by combining the cut elements and their neighbors: $\Omega_{\text{NB}} = \Omega_{\text{NB, cut}} \cup \Omega_{\text{NB, neigh}}$. Those elements will be the ones on which all the previously explained operations will be performed.
4. *Flagging of far elements* $\Omega_{\text{NB, far}}$: the elements outside the narrow band are directly obtained by subtracting the narrow band elements from the global element set: $\Omega_{\text{NB, far}} = \Omega \setminus \Omega_{\text{NB}}$.
5. *Flagging of appearing elements in the narrow band* $\Omega_{\text{NB, app}}$: the elements that appear in the narrow band at the new time step are obtained by storing the narrow band elements set over two time levels. They can then be readily determined by combining the two sets together: $\Omega_{\text{NB, app}} = \Omega^{t}_{\text{NB}} \setminus \Omega^{t-1}_{\text{NB}}$. As the level-set in those elements was kept unmodified during at least one time step, reinitialization must be applied on them to generate a level-set field consistent with the other elements inside the narrow band.
6. *Flagging of internal elements interfaces* $I_{\text{NB, int}}$: a loop on the narrow band elements interfaces is done. For each interface, the element indices on the right and the left are retrieved. If both elements belong to the previously built narrow band region, the interface is added to the set. It is then on this set of internal interfaces that the interfacial fluxes of the DGM will be integrated.
7. *Flagging of boundary elements interfaces* $I_{\text{NB, bnd}}$: the same approach as for internal interfaces is followed, except that for each interface, we check if one of the two elements on each side does not belong to the narrow band. If it is the case, the interface is added to the list. It is on this boundaries set that the homogeneous Neumann boundary conditions for the level-set, normally prescribed on the domain boundaries, will be imposed.

The building and update of this narrow band along a simulation requires therefore efficient dynamic data structures and the possibility of accessing those data anywhere in the code. An example of the narrow band application is shown in Figure 6.12.



a / Level-set defined and computed on the entire domain.

b / Restriction to a narrow band region Ω_{NB} around the iso-0. Elements in the far region $\Omega_{\text{NB, far}}$ are not considered.

c / narrow band composed of the elements cut by the iso-0 $\Omega_{\text{NB, cut}}$, their neighbors $\Omega_{\text{NB, neigh}}$, and the elements interfaces I_{NB} at the border of Ω_{NB}.

Figure 6.11 / Narrow band implementation – illustration of the notations.

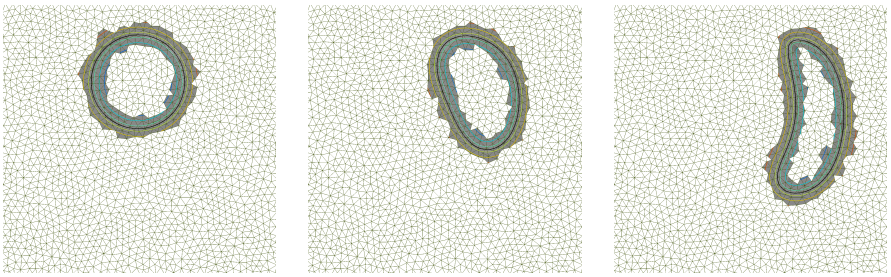


Figure 6.12 / Illustration of the narrow band concept on the vortex-in-a-box test case.

6.6 Enforcement of continuous level-set iso-0 contour

6.6.1 Two distinct level-set functions

In order to allow a high-order and sub-cell accurate interface representation, the level-set is approximated by a standard DG space (see Section 6.2.2). As it evolves, it is likely to develop discontinuities (of magnitude proportional to the spatial discretization error) across the element faces. The fact of having a piecewise continuous iso-0 contour is incompatible with the extended discontinuous Galerkin (XDG) discretization of the fluid equations which requires at least a level-set field $\in C^0(\Omega)$, i.e. being continuous across the element faces. The precise reasons for this requirement are elaborated in Section 8.2.3. The approach we follow, and which was also adopted by Smuda and Kummer (2022) and R. Saye (2017a), is to consider two distinct level-set fields,

$$\begin{aligned} \varphi(\mathbf{x},t) &\in \mathcal{W}_{p,\mathrm{LS}} && \text{Advected discontinuous level-set}, \\ \varphi_{C^0}(\mathbf{x},t) &\in \mathcal{W}^c_{p,\mathrm{LS}} && \text{XDG continuous level-set}, \end{aligned} \qquad [6.43]$$

where $\mathcal{W}^c_{p,\mathrm{LS}}$ is the space of continuous piecewise polynomials of degree p:

$$\mathcal{W}^c_{p,\mathrm{LS}}(\mathcal{T}) := \left\{ w \in C^0(\Omega) : w_{|_{\mathcal{T}_m}} \circ \sigma_m^{-1} \in \left[\mathcal{P}_p(\hat{\mathcal{T}}) \right], \ \forall\, \mathcal{T}_m \in \mathcal{T} \right\}. \qquad [6.44]$$

φ is the advected DG level-set field and φ_{C^0} is the continuous field obtained directly from φ at the end of every time step and which is used for discretization purposes in the XDG framework, i.e. for the numerical integration and possibly the geometrical quantities (normal and curvature). By considering two distinct fields and by letting φ unaltered, we prevent errors from cumulating over time levels.

6.6.2 Four approaches to enforce continuity

There are several ways in which the field φ_{C^0} can be obtained from φ. We list and compare below the approaches implemented and tested in this work.

Reinitialization on interpolation points

Based on what has been covered in this chapter, the most straightforward way is to assign the reinitialized level-set field to φ_{C^0}. More precisely, if the reinitialization is performed on the interpolation points, the minimum distance to the interface for two overlapping interpolation points lying on an element face must be the same. Hence, the reinitialization computed on the interpolation points automatically enforces the continuity. This is the approach followed

by R. Saye (2017a). From the conclusions stated at the end of Section 6.4.3, it appears that the closest point computation procedure might lead to some interface distortion in under-resolved regions or in the presence of kinks. That is why we will tend to avoid that first approach to impose the level-set continuity.

Averaging of face nodal degrees of freedom

The second possibility was implemented by Dr. Pierre Schrooyen in the solver. For every nodal interpolation point located on the mesh faces, the nodal values at this point associated to each adjacent element are averaged into a single value. This requires to accumulate, during the loop over the faces, the values at each nodal point location and average them at the end based on the number of collected values.

Constrained projection

The third approach was proposed by Smuda and Kummer (2022) and is the most rigorous one. It consists in solving a minimization problem with equality constraints for the unknown field φ_{C^0} over the domain or a portion of the domain (like the narrow band) such that

$$\begin{aligned} &\min_{\boldsymbol{\Phi}^*} \|\varphi_{C^0}(\boldsymbol{\Phi}^*) - \varphi\|_{L_2}^2 && \text{on } \mathbb{T}\,, \\ &\text{s.t. } \varphi^+_{C^0,e} = \varphi^-_{C^0,e} && \forall \mathbb{E}_e \in \mathbb{T}\,. \end{aligned} \qquad [6.45]$$

The goal is therefore to find the nodal coefficients, $\boldsymbol{\Phi}^*$, defining the φ_{C^0} polynomial over the mesh $\mathbb{T}$ such that the difference between φ_{C^0} and the original discontinuous level-set field φ is minimized, in the L_2 norm. Those unknown coefficients must be such that they ensure the continuity of the associated field across all the element faces, $\mathbb{E}_e$, present in the mesh.

Objective function The minimization statement can be further developed as

$$\min_{\boldsymbol{\Phi}^*} \|\varphi_{C^0}(\boldsymbol{\Phi}^*) - \varphi\|_{L_2}^2 = \min_{\boldsymbol{\Phi}^*} \left[\sum_{m \in \mathbb{T}} \int_{\mathbb{T}_m} \left(\boldsymbol{\phi}^T_{m,-}(\mathbf{x}) \boldsymbol{\Phi}^*_{m,-} - \boldsymbol{\phi}^T_{m,-}(\mathbf{x}) \boldsymbol{\Phi}_{m,-} \right)^2 \mathrm{d}\mathbf{x} \right], \qquad [6.46]$$

using the definition of the L_2 norm, Equation 6.30, and the interpolation by the shape function in each element $\mathbb{T}_m$, Equation 6.14. Developing and rearranging the terms in the integrand of Equation 6.46, we obtain

$$\min_{\boldsymbol{\Phi}^*} \left[\sum_{m \in \mathbb{T}} \left(\boldsymbol{\Phi}^{T,*}_{m,-} \underline{\boldsymbol{M}}_m \boldsymbol{\Phi}^*_{m,-} - 2 \boldsymbol{\Phi}^{T,*}_{m,-} \underline{\boldsymbol{M}}_m \boldsymbol{\Phi}_{m,-} + \boldsymbol{\Phi}^T_{m,-} \underline{\boldsymbol{M}}_m \boldsymbol{\Phi}_{m,-} \right) \right], \qquad [6.47]$$

where the mass matrix in element $\mathbb{T}_m$ is

$$\underline{\boldsymbol{M}}_m = \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x})\, \boldsymbol{\phi}^T_{m,-}(\mathbf{x})\, \mathrm{d}\mathbf{x}\,.$$

Constraints and redundancy Concerning the set of constraints, they can be expressed under a matrix-vector product of the form

$$\underline{\boldsymbol{A}}\, \boldsymbol{\Phi}^*_{-,-} = \mathbf{0}\,, \tag{6.48}$$

where the vector $\boldsymbol{\Phi}^*_{-,-}$ compiles all the nodal degrees of freedom of the domain region of interest,

$$\boldsymbol{\Phi}^*_{-,-} = \begin{bmatrix} \boldsymbol{\Phi}^*_{1,-} \\ \vdots \\ \boldsymbol{\Phi}^*_{m,-} \\ \vdots \\ \boldsymbol{\Phi}^*_{1,N_T} \end{bmatrix},$$

and the matrix of constraints, $\underline{\boldsymbol{A}}$, containing on each of its row a constraint of continuity at a nodal point. For example, for the ith-constraint at a face node with position $\mathbf{x}_e$ between elements $\mathbb{T}^+_m$ and $\mathbb{T}^-_m$, the corresponding row of $\underline{\boldsymbol{A}}$ would look like

$$\underline{\boldsymbol{A}}[i,:] = \left[\mathbf{0}^T, \dots, \boldsymbol{\phi}^{T,+}_m(\mathbf{x}_e), \dots \mathbf{0}^T, -\boldsymbol{\phi}^{T,-}_m(\mathbf{x}_e), \dots \mathbf{0}^T\right].$$

From the properties of the Lagrange basis used in this work, each row of $\underline{\boldsymbol{A}}$ simplifies to a vector full of zeroes, except two entries set to $+1$ and -1, corresponding to the index of the face nodal degrees of freedom seen by the left and right elements. An important detail pointed out by Smuda (2021, Sec. 3.6.1) concerns the number of constraints to be applied for the problem to be well posed. It turns out that for each vertex of the mesh which is not located on the domain boundary, we should impose one less constraint than the number of elements sharing this vertex. During the assembly of the constraints, we therefore check for each nodal degrees of freedom located on a vertex that the number of associated constraints is below the threshold before adding the corresponding line to $\underline{\boldsymbol{A}}$.

Assembly of the system to solve With the quadratic objective function, Equation 6.47, and the set of linear equality constraints, Equation 6.48, the problem to be solved is a quadratic programming (QP) optimization problem which can be formulated by means of Lagrange multipliers such that

$$\min_{\boldsymbol{\Phi}^*,\boldsymbol{\lambda}} L(\boldsymbol{\Phi}^*,\boldsymbol{\lambda}) = \min_{\boldsymbol{\Phi}^*,\boldsymbol{\lambda}} J(\boldsymbol{\Phi}^*) + \boldsymbol{\lambda}^T\, \underline{\boldsymbol{A}}\, \boldsymbol{\Phi}^*\,,$$

with

$$J(\boldsymbol{\Phi}^*) = \frac{1}{2}\boldsymbol{\Phi}^{T,*}_{-,-}\, \underline{\boldsymbol{M}}\, \boldsymbol{\Phi}^*_{-,-} - \boldsymbol{\Phi}^{T,*}_{-,-}\, \underline{\boldsymbol{M}}\, \boldsymbol{\Phi}_{-,-} + \frac{1}{2}\boldsymbol{\Phi}^T_{-,-}\, \underline{\boldsymbol{M}}\, \boldsymbol{\Phi}_{-,-}\,.$$

To find the solution, we impose

$$\begin{cases} \frac{\partial L}{\partial \boldsymbol{\Phi}^*} = \mathbf{0} & \Leftrightarrow \underline{\boldsymbol{M}}\, \boldsymbol{\Phi}^* - \underline{\boldsymbol{M}}\, \boldsymbol{\Phi} + \underline{\boldsymbol{A}}^T \boldsymbol{\lambda} = \mathbf{0}\,, \\ \frac{\partial L}{\partial \boldsymbol{\lambda}} = \mathbf{0} & \Leftrightarrow \underline{\boldsymbol{A}}\, \boldsymbol{\Phi}^* = \mathbf{0}\,. \end{cases} \tag{6.49}$$

Multiplying the first equation by $\underline{\boldsymbol{A}}\,\underline{\boldsymbol{M}}^{-1}$ and inserting the second equation in the result, we obtain the system to be solved for the Lagrange multipliers, i.e.

$$\underline{\boldsymbol{A}}\,\underline{\boldsymbol{M}}^{-1}\,\underline{\boldsymbol{A}}^{T}\,\boldsymbol{\lambda} = \underline{\boldsymbol{A}}\,\boldsymbol{\Phi}\,. \tag{6.50}$$

After having solved Equation 6.50, the first relation of Equation 6.49 is used to form the system determining the unknowns of the problem,

$$\underline{\boldsymbol{M}}\,\boldsymbol{\Phi}^{*} = \underline{\boldsymbol{M}}\,\boldsymbol{\Phi} - \underline{\boldsymbol{A}}^{T}\,\boldsymbol{\lambda}\,. \tag{6.51}$$

Partition of unity

The last approach was in fact first implemented for the purpose of curvature filtering (see Section 7.2.4) but it also allows to build a continuous function, which is not only C^0 but C^∞. This approach makes use of the partition of unity principle and follows the procedure presented in R. Saye (2017b, App. B). Given a point in the domain located at the position $\mathbf{x}^*$ inside the element $\mathbb{T}_{m,0}$, we proceed as follows:

1. We build a patch of elements surrounding the element $\mathbb{T}_0$, sharing common vertices or faces. This patch, excluding $\mathbb{T}_0$, is noted $\{\mathbb{T}_{m,j}\}$ with $j = 1, \dots, N_j$ where N_j is the number of elements in the patch.
2. For each element in the patch and the element $\mathbb{T}_{m,0}$, we determine the barycenter
$$\mathbf{x}^c_{m,j} = (x^c_{m,j},\, y^c_{m,j})\,.$$
We also need for each element the radius of the largest ball that can fit inside the patch associated to each element (i.e. each element in the patch of $\mathbb{T}_0$ has also a corresponding patch) and such that the ball is centered at $\mathbf{x}^c_{m,j}$. This radius, noted $r^c_{m,j}$, is estimated as the ratio between the area and the perimeter of the patch, with a multiplicative factor of two for quadrilateral elements.
3. The partition of unity principle then consists in computing the value of a function evaluated at $\mathbf{x}^*$, let's say $u(\mathbf{x}^*)$, as a normalized weighted linear combination over the patch such that
$$u(\mathbf{x}^*) = \frac{w_{m,0}(\mathbf{x}^*)\,\boldsymbol{\phi}^{T}_{m,0}(\mathbf{x}^*)\,\mathbf{U}_{m,0} + \sum_{j=1}^{N_j} w_{m,j}(\mathbf{x}^*)\,\boldsymbol{\phi}^{T,\mathrm{ext}}_{m,j}(\mathbf{x}^*)\,\mathbf{U}_{m,j}}{w_{m,0}(\mathbf{x}^*) + \sum_{j=1}^{N_j} w_{m,j}(\mathbf{x}^*)}\,, \tag{6.52}$$
where:
   - $w_{m,j}$ is the weight function for the element $\mathbb{T}_{m,j}$ in the patch given by
   $$w_{m,j} = \omega\left(\frac{x^* - x^c_{m,j}}{r^c_{m,j}}\right)\cdot\omega\left(\frac{y^* - y^c_{m,j}}{r^c_{m,j}}\right),$$
   with the one-dimensional bump function ω determining the blending

between the different elements,

$$\omega(x) = \begin{cases} \exp(C/(x^2-1)) & \text{if } |x| < 1\,, \\ 0 & \text{if } |x| \geq 1\,. \end{cases}$$

The constant C controls the width of the bump around its center and is fixed to $C = 32/9$ according to R. Saye (2017b).

- $\boldsymbol{\phi}^{\text{ext}}_{m,j}$ is the shape function defined on an element $\mathbb{T}_{m,j}$ of the patch (excluding $\mathbb{T}_{m,0}$) and which is evaluated outside its reference support since $\mathbf{x}^*$ is located in $\mathbb{T}_{m,0}$. Because the operations are performed in parametric coordinates, this is achieved in practice by using the inverse mapping function associated to element $\mathbb{T}_{m,j}$ to find the parametric coordinates corresponding to $\mathbf{x}^*$ at which the shape function $\boldsymbol{\phi}^{\text{ext}}_{m,j}$ needs to be evaluated:

$$\boldsymbol{\phi}^{\text{ext}}_{m,j}(\boldsymbol{\xi}^*) = \boldsymbol{\phi}^{\text{ext}}_{m,j}(\sigma^{-1}_{m,j}(\mathbf{x}^*))\,. \qquad [6.53]$$

This same kind of extension will be used for the cell-agglomeration procedure of the XDG method (see Section 8.2.5) as well as for the patch-recovery technique for curvature filtering (see Section 7.2.3).

Comparison of the different continuity enforcement methods

To conclude this section, we present a comparison of the different techniques described above. The situation considered here is the one of a starfish-shaped level-set field. The analytical expression of the level-set function is used to set the values at all the quadrature points in the domain. The L_2 projection of Equation 6.29 is then used to obtain the nodal values at the interpolation points. By initializing the level-set in this way, the continuity of its iso-contours is not enforced even before any advection step. The continuity projection is then performed on the level-set (without doing any advection) and the resulting continuous level-set field is extracted. The errors between the original broken level-set field and the continuous one for the four approaches are listed in Table 6.2. They are computed on a narrow band region around the interface. The magnitudes of the errors are basically comparable. As expected, the error in the L_2 norm is minimum for the constrained minimization method. Concerning the reinitialization, we did not list the L_2 and L_∞ errors, which are global errors, because the initial level-set field being not a signed distance function, the reinitialized field is quite different from it. Figure 6.13 presents a visual comparison by zooming on the iso-0 at two element faces in the domain. We observe that all four methods enforce indeed the continuity of the iso-0 across the faces. We could also say that the L_2 minimization approach seems to be the one minimizing the deformation with respect to the original iso-0. From the errors and the visual comparison, we decided to only rely on the face averaging approach as it is by far the fastest method and it allows to obtain an accurate continuous level-set field (at least provided that the resolution is sufficient).

Table 6.2 / Comparison of the different level-set continuity enforcement techniques based on several error measures. The errors are computed based on the original broken level-set field. The test case considered is a starfish-shape level-set on which a single continuity enforcement step is performed.

	$\|.\|_{L_2}$	$\|.\|_{L_\infty}$	E_{Int}	E_{Vol} (%)	CPU (ms)
Reinitialization	/	/	9.70e-05	3.42e-05	275.82
Face average	8.04e-06	4.56e-04	2.55e-05	2.71e-05	60.04
L_2 minimization	7.58e-06	4.63e-04	2.48e-05	2.26e-05	241.78
Partition of unity	9.99e-06	4.64e-04	2.78e-05	1.59e-05	4559.66

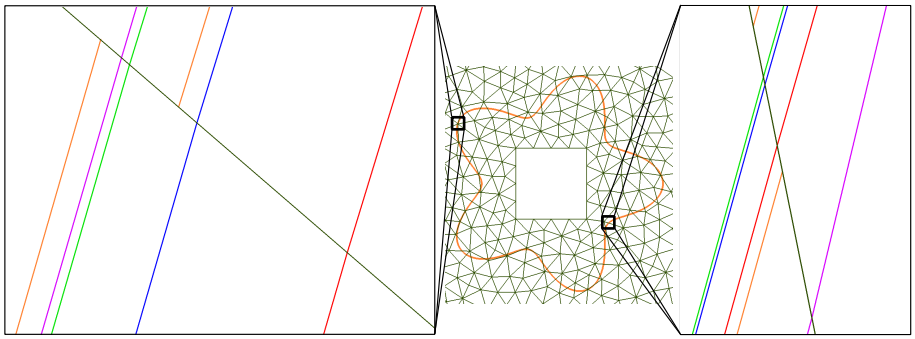


Figure 6.13 / Visual comparison of the different level-set continuity enforcement techniques with zooms on iso-0 at two element faces: original broken level-set (—), reinitialization (—), Face average (—), L_2 minimization (—), partition of unity (—).

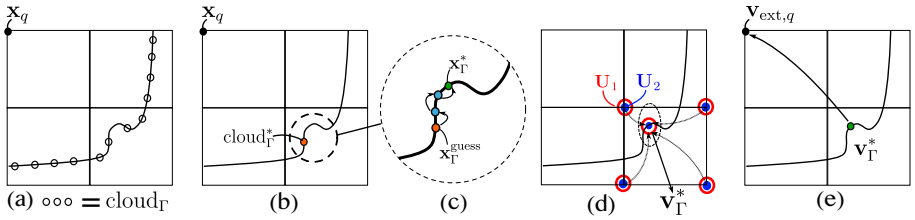


Figure 6.14 / Illustration of Algorithm 3. (a) sampling of level-set by cloud of points with recursive contouring algorithm or iso-0 quadrature rules, (b) closest point among the cloud w.r.t to a given target mesh point $\mathbf{x}_q$ found by search tree algorithm, (c) point used as a guess for a non-linear constrained minimization problem leading to the exact closest point $\mathbf{x}_\Gamma^*$ on the interface, (d) interpolation of the fluid states from both sides of the interface to $\mathbf{x}_\Gamma^*$ and computation of the interface velocity $\mathbf{v}_\Gamma^*$, (e) extension of the velocity to the mesh point.

6.7 Organization and summary of the various methods

In this last section, the various methods developed for the level-set in the previous sections are organized through two global algorithms. Algorithm 3 summarizes the main steps of the reinitialization and extension procedures and is depicted by a sequence of sketches in Figure 6.14. Algorithm 4 gives the complete overview of the steps involved in the advection of the level-set field over one time step and will be referred to in the second part. They are both accompanied by a list of comments. We can add the following comments

Algorithm 3 Main steps for the reinitialization and extension of the velocity based on a direct geometric approach. It takes as inputs the level-set field $\boldsymbol{\Phi}$ and the solution from the fluid solver $\underline{\boldsymbol{U}}^{\text{fluid}}$

1: **procedure** ReinitExtend($\boldsymbol{\Phi}$, $\underline{\boldsymbol{U}}^{\text{fluid}}$)
2: cloudPts ← `SampleLevelSet`($\boldsymbol{\Phi}$) ▷ (a)
3: searchTree ← `CreateSearchTree`(cloudPts) ▷ (b)
4: $\mathbf{x}_{\Gamma,\text{cp}}, \boldsymbol{\Phi}^{\text{reinit}}$ ← `ComputeClosestPoint`($\boldsymbol{\Phi}$, cloudPts, searchTree) ▷ (c)
5: $\mathbf{v}_{\text{ext}}$ ← `ComputeExtendedVelocity`($\boldsymbol{\Phi}, \mathbf{x}_{\Gamma,\text{cp}}, \underline{\boldsymbol{U}}^{\text{fluid}}$) ▷ (d)
6: **return** $\boldsymbol{\Phi}^{\text{reinit}}, \mathbf{v}_{\text{ext}}$
7: **end procedure**

concerning Algorithm 3:

- (*a*): generate cloud of points sampling the level-set, see Section 6.3.2.
- (*b*): create search tree from the cloud of points.
- (*c*): compute closest points for all the nodes and obtain from there the reinitialized level-set field following Algorithm 1.

- (d): compute the extended velocity field for all the nodes following Algorithm 2

Algorithm 4 Level-set solver for one time step taking as inputs the fluid solutions from the current time step $\underline{\boldsymbol{U}}^{\text{fluid}}(t_n)$ and a history of level-set solutions over previous time steps and computing the level-set at the next time step $\boldsymbol{\Phi}(t_{n+1})$ based on a spatial DG discretization and explicit temporal integration with N_s stages.

procedure LSOneTimeStep($\underline{\boldsymbol{U}}^{\text{fluid}}(t_n)$, $\{\boldsymbol{\Phi}(t_{n+i})\}_{i=-N_{\text{step}}+1,\dots,0}$) ▷ (a)
  $\bar{t} = t_n$ ▷ (b)
  $\bar{\boldsymbol{\Phi}}_s \leftarrow \boldsymbol{\Phi}(t_n)\ \forall s = 1, \dots, N_s$ ▷ (c)
  $\bar{\mathcal{L}}_s^{\text{sp}} \leftarrow \boldsymbol{0}\ \forall s = 1, \dots, N_s$ ▷ (d)
  $\Omega_{\text{NB}}(t_n) \leftarrow$ `UpdateNarrowBand`$(\boldsymbol{\Phi}(t_n))$ ▷ (e)
  $\Delta t(t_n) \leftarrow$ `GetStableTimeStep`$(\mathbf{v}_{\text{ext}})$ ▷ (f)
  for $s = 1, \dots, N_s$ **do** ▷ (g)
    $\bar{\boldsymbol{\Phi}}_s \mathrel{+}= \sum_{i=1}^{s-1} \Delta t(t_n)\, a_{si}\, \bar{\mathcal{L}}_s^{\text{sp}}$ ▷ (h)
    on $\Omega_{\text{NB}}(t_n)$ **do**: ▷ (i)
      $\mathbf{v}_{\text{ext}}(\bar{t}) \leftarrow$ `ReinitExtend`$(\bar{\boldsymbol{\Phi}}_s, \underline{\boldsymbol{U}}^{\text{fluid}}(t_n))$ ▷ (j)
      $\bar{\mathcal{L}}_s^{\text{sp}} = \underline{\boldsymbol{M}}^{-1}\, \mathcal{L}^{\text{sp}}(\bar{\boldsymbol{\Phi}}_s)$ ▷ (k)
    $\bar{t} \mathrel{+}= t_n + \alpha_s\, \Delta t(t_n)$
  end for
  $\boldsymbol{\Phi}(t_{n+1}) = \boldsymbol{\Phi}(t_n) + \sum_{s=1}^{N_s} \Delta t(t_n)\, \gamma_s\, \bar{\mathcal{L}}_s^{\text{sp}}$
  $\Omega_{\text{NB}}(t_{n+1}) \leftarrow$ `UpdateNarrowBand`$(\boldsymbol{\Phi}(t_{n+1}))$
  on $\Omega_{\text{NB}}(t_{n+1})$ **do**:
    $\boldsymbol{\Phi}^{\text{reinit}}(t_{n+1}) \leftarrow$ `ReinitExtend`$(\boldsymbol{\Phi}(t_{n+1}), \underline{\boldsymbol{U}}^{\text{fluid}}(t_n))$ ▷ (l)
  on $\Omega_{\text{NB,app}}(t_{n+1})$ **do**:
    $\boldsymbol{\Phi}(t_{n+1}) \leftarrow \boldsymbol{\Phi}^{\text{reinit}}(t_{n+1})$ ▷ (m)
  if Reinitilization needed **then**
    $\boldsymbol{\Phi}(t_{n+1}) \leftarrow \boldsymbol{\Phi}^{\text{reinit}}(t_{n+1})$ ▷ (n)
  end if
  return $\boldsymbol{\Phi}(t_{n+1})$
end procedure

We formulate the following comments about Algorithm 4:

- (a): because of the Lie operator splitting employed in this work (see Section 10.3.2), the fluid solution used by the level-set is the one at the previous time step t_n and is not updated during the level-set advection. In the case of a single-step multi-stage time stepping scheme as a Runge-Kutta scheme presented in Section 6.2.3, $N_{\text{step}} = 1$ and we only need to store the level-set solution from the previous time step.
- (b): $\bar{t}$ is the time updated at every stage of the temporal scheme.
- (c): initialize the solution container for each stage with the previous solution.
- (d): initialize the spatial residual container for each stage.

- (e): update the narrow band elements, see Section 6.5.2.
- (f): compute the time step compatible with the imposed CFL number from Equation 6.20. Note that this algorithm can therefore be repeated several times per time step if the stable time step is smaller than the user-defined time step.
- (g): loop on the stages of the time stepping scheme.
- (h): obtain solution at s^{th}-stage based on Equation 6.19.
- (i): perform the operations only on the elements contained in the narrow band.
- (j): apply Algorithm 3 to generate the extended velocity field over the narrow band at the sub-time $\bar{t}$.
- (k): assemble the spatial residual of the level-set by Equation 6.18.
- (l): apply Algorithm 3 to compute the reinitialized level-set field over the new narrow band from the new level-set solution.
- (m): assign the reinitialized level-set field to the level-set solution on the elements entering the narrow band.
- (n): if the frequency of reinitialization is reached, replace the new level-set solution by the reinitialized level-set field.

Contribution 6.1: High-order level-set framework

The main contributions that can be identified from this chapter are:

- The design, inspired by R. Saye (2014), of high-order single-step reinitialization and velocity extension procedures for the level-set method.
- The comparison of various strategies to enforce continuity of the level-set iso-0 across element faces.
- The organization of the various techniques inside a global algorithm to advect the interface over one time step.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- A new class extending the capabilities of the original one by implementing the described exact closest point computation.
- The possibility of handling two distinct level-set fields, one of them being enforced to be continuous.
- The narrow band strategy.

The resulting level-set solver features the first of its kind exact closest point computation strategy for high-order level-set reinitialization and extension on general (unstructured) meshes.

7

Curvature filtering strategies for high-order sharp interface surface tension treatment

The material of this chapter was submitted in Henneaux et al. (2023d). Part of the contents of this chapter is the result of the master thesis of Ward Haegeman (Haegeman (2022)) for which the present author was one of the advisors.

7.1 Overview of techniques to handle surface tension forces 242
7.1.1 Modeling and overview of methods to treat surface tension forces 242
7.1.2 Curvature filtering strategies in high-order FE-based methods 243
7.1.3 Objectives of the analysis 245
7.2 Description of the implemented filtering framework 246
7.2.1 Global flow chart 246
7.2.2 The artificial viscosity method 250
7.2.3 The patch-recovery method 250
7.2.4 The partition of unity method 252
7.3 Assessment of the various strategies 253

7.1 Overview of techniques to handle surface tension forces

7.1.1 Modeling and overview of methods to treat surface tension forces

Because of the imbalance of intermolecular attractive forces that appear at the interface between a liquid and another medium, a resulting localized surface force is present. This force can act both normal and tangential to a fluid interface and is responsible for the phenomenon known as surface tension. Surface tension plays a crucial role in immiscible multiphase flows. It has a determinant effect on interfacial stability, droplet coalescence and break up, surfactant behavior and wetting. Those effects, confined at the interface, can also have a profound impact on the dynamic behavior of the flow in the bulk of each fluid, even when the interface region is small compared to the whole flow. Predicting how surface tension influences the dynamic of a multi-fluid system is therefore very important for many practical applications in different fields. In particular, as we have seen in Section 1.2, surface tension effects are determinant in the way liquid particles are released from the melt layer, hence it can influence the ablating process of a re-entering space debris. When it comes to the modeling of surface tension, a momentum balance across the interface is performed from which it can be extracted that the stresses from the bulk acting on the interface are balanced by a pressure force depending on the surface tension coefficient and the interface curvature (neglecting Marangoni effects). The way of accurately incorporating this force into a numerical framework has been the topic of extensive research over the past decades (see e.g. the review of Popinet (2018)). Three main challenges make this task particularly delicate: (i) the surface tension force is a singular force acting on an interface, (ii) this interface is most of the time moving and can exhibit complex shapes, (iii) the direct computation of the interface curvature often leads to inaccurate estimations affecting the accuracy, stability and robustness of the global method. The first difficulty will be treated in the two following chapters, Chapter 8 and Chapter 9. The second aspect is the subject of the previous chapter, Chapter 6. The last challenge is the reason of this chapter.

Let us start by mentioning some methods that have been proposed to deal with the first difficulty, specifically in the context of surface tension driven flows. The common distinction between diffuse and sharp interface approaches is here followed, as already done in previous chapters and for which a further discussion is provided in Section 8.1. In the diffuse interface methods, a numerical smearing of the interface is allowed. The most common approach dealing with surface tension for incompressible flows relies on the Continuous Surface Force (CSF) approach first proposed by Brackbill et al. (1992) and its variants where the surface force is converted to a volumetric force active over grid points in the

vicinity of the interface in a region that is a few cells thick. For compressible flows, multi-fluid models derived from averaging procedures of the one-fluid Euler or Navier-Stokes equations are considered. The capillary forces are inserted into the model through additional source terms that need to be carefully built to preserve the thermodynamically consistent and hyperbolic nature of the original system (Schmidmayer et al. (2017), Cordesse et al. (2019)). These methods have proven to be successful thanks to their simplicity of implementation, adaptation to modeling a wide variety of physics and robustness. However, the inherent smoothing of interfacial forces can lower modeling fidelity. In case of highly intricate physics, such as interfacial instabilities and breakup in a shear-dominated environment, a rigorous and exact imposition of the jump conditions turns out to be necessary, as it was pointed out by R.R. Nourgaliev et al. (2008) and Robert I. Saye and James A. Sethian (2020). For the sharp interface methods, the interfacial force due to surface tension is captured as a discontinuity. In the context of Finite Volume Method (FVM), this is commonly realized through the Ghost Fluid Method (GFM) of Fedkiw et al. (1999) or the Sharp Surface Force (SSF) of Francois et al. (2006). In Finite Element (FE) based methods, the discretization of the surface force for incompressible flows is generally done by adding a term in the weak formulation that integrates on the interface the force along the normal, like in Kummer and Warburton (2016b) and Hashemi et al. (2020). For compressible flows on the other hand, the pressure jump induced by surface tension should be incorporated inside a multiphase Riemann solver to correctly take into account the modification of the wave structure due to the presence of surface tension, hence properly providing stabilizing upwinding, as done e.g. in Chang et al. (2013) and in Stefan Fechter et al. (2018). While improving the modeling fidelity, the sharp enforcement of the surface forces entails more complicated implementation and narrows the scope of applications compared to their diffuse interface counterparts. Francois et al. (2006) note that whatever the method used, it is important to ensure a discrete consistent balance between the surface tension force and the associated momentum jump, otherwise spurious currents will develop even if the exact interface curvature is prescribed.

7.1.2 Curvature filtering strategies in high-order FE-based methods

The challenge concerning interface curvature computation is the central subject of this chapter. It can be addressed in various ways that depend on the type of discretization, the method to treat the singular interface force and the approach selected for interface motion. For standard low-order FVM, since the double differentiation is required to estimate curvature, a stencil made up of neighboring elements is considered in order to be able to approximate the curvature with sufficient accuracy. Because the curvature that needs to be computed is the one at the interface location while the flows variables are manipulated at nodes in the computational domain, a strategy to distribute it to the cells in the

vicinity of the interface is required. The height-function method for VOF and the closest point method for the level-set are some popular examples. As the interest here is on sharp interface high-order FE-based methods relying on a level-set interface-capturing method, one shall focus the review of the available methods on this category of approaches. Before reviewing them, it is worth explaining the reason why a straightforward computation of the interface curvature is problematic. In high-order FE-based methods employing an implicit interface representation, the interface is approximated by a high-order polynomial. Contrary to standard low-order methods that require a stencil of several neighboring elements to be able to reach sufficient accuracy, the curvature in a given cell could in theory be directly computed anywhere in the cell (in particular at the interface) from the level-set interpolant derivatives. However, this will certainly produce an inaccurate and oscillatory curvature field which will in turn generate pressure oscillations inducing spurious currents. The root cause of this issue is that the higher the polynomial order, the higher the frequencies contained in the level-set interpolant (for the most general case in which the interface shape cannot be exactly represented by a polynomial). The non-linear combination of the level-set function derivatives to assemble the curvature will magnify the high-order modes contained in the level-set signal and result in a noisy and inaccurate curvature field. This is noted by Marchandise et al. (2007) Kummer and Warburton (2016b) and R. Saye (2017b)[1].

Two solutions are then possible. The first one is to avoid having to compute the curvature. This can be achieved invoking a theorem from differential geometry linking the product of the interface curvature and normal with the Laplacian of the identify mapping on the interface. After integration by parts, the original term modeling surface tension forces and involving the curvature is replaced by two terms where the curvature is absent. This Laplace-Beltrami formulation has been followed for FEM in S. Hysing (2006), Gross and Arnold Reusken (2007) and Cheng and Fries (2012) and for discontinuous Galerkin (DG) method in Smuda and Kummer (2022). Another interesting approach is to augment the traditional level-set advection equation with an additional equation to explicitly track the interface curvature, thus avoiding the need to compute level-set derivatives. This is proposed in Vogl (2016). The second solution is to apply some sort of filtering during the computation of the curvature from the level-set equation with the aim of damping the high-order modes when evaluating the derivatives. This can be carried out by different means. Marchandise et al. (2007) propose a least squares method to reconstruct the level-set gradients. For each computational node, the level-set is assumed to behave linearly or quadratically over the adjacent elements. This allows to evaluate the level-set at the center of gravity of those elements and build an overdetermined system for the unknown level-set gradient and Hessian at the considered node. The system is finally solved in a least-squares sense. The curvature approximation

[1] It should be remarked here that this problem is expected both for continuous and discontinuous FE approximations. It is therefore not the discontinuous character of the approximation that is the primary cause of the oscillatory curvature, as noted by Marchandise et al. (2007).

is shown to be of second order. S. Fechter and C.-D. Munz (2015) apply the $P_N P_M$-reconstruction proposed by Dumbser (2010) to reconstruct the level-set polynomial. Starting from a function defined over a given element, the idea is to consider a stencil of neighboring elements and perform a L_2-projection of an unknown function belonging to a higher order polynomial space on each of the elements in the stencil. This results in an overdetermined system for the unknown reconstructed nodal degrees of freedom. The approach is therefore related to the first one except that instead of a nodal projection, a more accurate L_2-projection is considered. In the same vein, Kummer and Warburton (2016b) propose the patch-recovery operator that can easily be applied at any level of the process of curvature computation (i.e. on the level-set, on the gradients of the level-set and/or on the final curvature field). It shares some similarities with the $P_N P_M$-reconstruction with the main difference that the L_2-projection is here considered on a composite cell composed by all the adjacent elements on which a smoothly extended polynomial space is defined, rather than on each element separately. There is therefore no need to solve an overdetermined system if the new function space is chosen to be of higher order than the original one. R. Saye (2017b) use a similar patch-recovery operator in combination with a partition of unity method. For each element, the best polynomial, in the sense of the L_2 norm, is found matching the original function in the neighborhood of this element. In a second step, those polynomials which can now be evaluated outside their original element, are linearly combined with weight functions with compact support in order to obtain a C^∞ smooth function. Thus, this procedure involves two steps of smoothing. What those four methods have in common is the way they filter out the small amplitude high frequency modes contained in the level-set by combining neighboring element information. This is effective because those modes tend to be uncorrelated on adjacent elements and the combination between the elements will favor the accuracy of low-order modes, as explained by R. Saye (2017b). Finally, a last possible approach, employed by Tornberg and Engquist (2000) and Felix Heimann (2013), is to solve a diffusion equation for the level-set or curvature, hence introducing some artificial viscosity on the original fields to reduce the spurious peaks in the fields.

7.1.3 Objectives of the analysis

It is important to remark at this stage that because of the compressible models considered, the Laplace-Beltrami reformulation of the surface tension forces mentioned earlier does not seem appropriate, at least under its original form. Indeed, as it will be explained in Chapter 9, the pressure jump induced by the surface tension must be incorporated inside a multiphase Riemann solver. The separate treatment of the surface tension term in the variational form is thus not adapted. Therefore, the surface tension forces in this work are computed by assembling the curvature from the level-set field. The purpose of this study is to compare several methodologies designed for accurate curvature estimation and

assess their ability to provide robust and stable surface tension computations. This objective is in line with previous analyses comparing different approaches such as the ones conducted by Marschall et al. (2014) and Abadie et al. (2015). The novelty lies in the fact that here state-of-the-art approaches tailored to high-order methods are scrutinized, namely the patch-recovery, partition of unity and artificial viscosity methods. To this end, a general flow chart composed of decisive steps in the curvature estimation process is established, in addition to the choice of the filtering strategy: the projection or not of the level-set function on a continuous polynomial space, the use of discrete or lifted derivatives, the interpolation and assembly of the curvature on the interface, and the choice between the Bonnet's formula and the divergence of the normal.

7.2 Description of the implemented filtering framework

7.2.1 Global flow chart

As introduced in section Section 7.1.2, the existing strategies to be explored for curvature computation depend on several elements. Those elements determine different paths along the curvature estimation process which is itself composed of several steps. This is illustrated by the flow chart in Figure 7.1, the items of which are described here:

- *Level-set Type*: the discretized level-set field, noted φ, is in general discontinuous across element edges. However, to assemble the XDG variational formulation, a C^0 level-set field is required, as explained in Section 8.2.3. To impose the continuity between the elements, an averaging of the nodal values located on the edges is performed, as detailed in Section 6.6. Two level-set fields are then manipulated, the *discontinuous* field φ_{br} which is advected, and the *continuous* one φ_{C^0} which is provided to the XDG method. At the beginning of the curvature computation process, one can select which level-set field is used, by taking *Cont* = *None* for the original one, and *Cont* = *FaceAverage* for the continuous one.
- *Curvature Formula*: the curvature can be obtained by either taking the *divergence of the normal* field (*Div.* in the flowchart),

$$\kappa = -\nabla \cdot \mathbf{n}_\Gamma \,, \tag{7.1}$$

or by using the so-called *Bonnet*'s formula,

$$\kappa = \frac{\nabla\varphi \cdot \mathrm{H}(\varphi)\nabla\varphi - \|\nabla\varphi\|_2^2 \mathrm{tr}(\mathrm{H}(\varphi))}{\|\nabla\varphi\|_2^3} \,, \tag{7.2}$$

where $\mathrm{H}(\varphi)$ is the Hessian matrix of the level-set function and $\mathrm{tr}(\mathrm{H}(\varphi))$

its trace (i.e. the Laplacian). The unit normal vector is computed from the gradient of the level-set,

$$\mathbf{n}_\Gamma = \frac{\nabla\varphi}{\|\nabla\varphi\|_2} \,. \tag{7.3}$$

Although strictly equivalent for exact continuous function, Equation 7.1 and Equation 7.2 become different when the level-set field is a function belonging to a discrete functional space. This is the reason why the two formulae are considered here. Note that to apply Equation 7.1, the normal field first need to be assembled with Equation 7.3, which corresponds to the upper branch on the flow chart. For Equation 7.2, both the gradient and the Hessian matrix of the level-set function need to be computed.

- *Derivatives Choice*: in DG methods, a discretized function f is part of a discontinuous polynomial space, i.e. that the restriction of the function to each element Ω_m, noted $f|_{\Omega_m}$, belongs to a function space of a given order defined on a reference parametric element. There is no continuity constraint across the element edges. Similarly, the *broken* gradient, noted ∇_{br}, is obtained as the cellwize differentiation of the function:

$$(\nabla_{\text{br}} f)|_{\Omega_m} := \nabla\left(f|_{\Omega_m}\right) \quad \forall\, \Omega_m. \tag{7.4}$$

This way of computing the derivatives can also be applied to the divergence and Hessian operators. In order to take into account inter-element jumps during derivatives computation, the *lifted* approach can be followed. In this case, the gradient of a function, noted ∇_{DG}, is obtained by:

$$\int_{\Omega_m} (\nabla_{\text{DG}} f)\, w \,\mathrm{d}V = -\int_{\Omega_m} f \nabla w \,\mathrm{d}V + \int_{\partial\Omega_m} \langle f \rangle\, w\, \mathbf{n}_{\partial\Omega_m} \,\mathrm{d}S \quad \forall\, \Omega_m \,, \tag{7.5}$$

where $\langle f \rangle$ is the average of the traces of f coming from both sides of the edges $\partial\Omega_m$ and w is the test function. When applied to the continuous level-set field, Equation 7.4 and Equation 7.5 produce the same result. But for the gradient of the discontinuous level-set field, the divergence of the normal and the Hessian of the level-set, these two ways of evaluating the derivatives will lead to different results.

- *Interpolation on level-set iso-0*: after assembling the curvature field on the nodal degrees of freedom (based on the divergence of the normal or the Bonnet's formula, using broken or lifted derivatives), it must be interpolated to the level-set iso-0. Indeed, to evaluate the interface integrals of the XDG formulation, the curvature is required at every quadrature point located on the interface. The position of the interface is at this stage represented by the zero iso-contour of the continuous level-set field. A specificity of the Bonnet's formula is that one has the choice between assembling the curvature on the nodal degrees of freedom before doing the interpolation on the interface (*Bonnet IP*) and assembling it directly on the interface by interpolating the level-set gradient and Hessian on the iso-0

instead (*Bonnet Isoo*). This second path corresponds to the four arrows in the lower right part of the flow chart. To perform this interpolation, the *classical* approach using the cellwize shape functions is the first possibility. The second one is to involve neighboring elements information via a *partition of unity* evaluation such that the curvature is continuous across the element edges along the level-set iso-0.

- *Filtering Procedure*: finally, in order to reduce the spurious oscillations in the curvature field, several filtering techniques are available (*Filt* on the figure): *artificial viscosity*, *patch-recovery*, *partition of unity*, *patch-recovery followed by partition of unity*. Those can be applied to the level-set field, the normal field (in case the curvature is obtained via the divergence of the normal), the level-set gradient and Hessian (in case the Bonnet's formula is chosen) or the curvature field itself. The three filtering methods are described in the next sections.

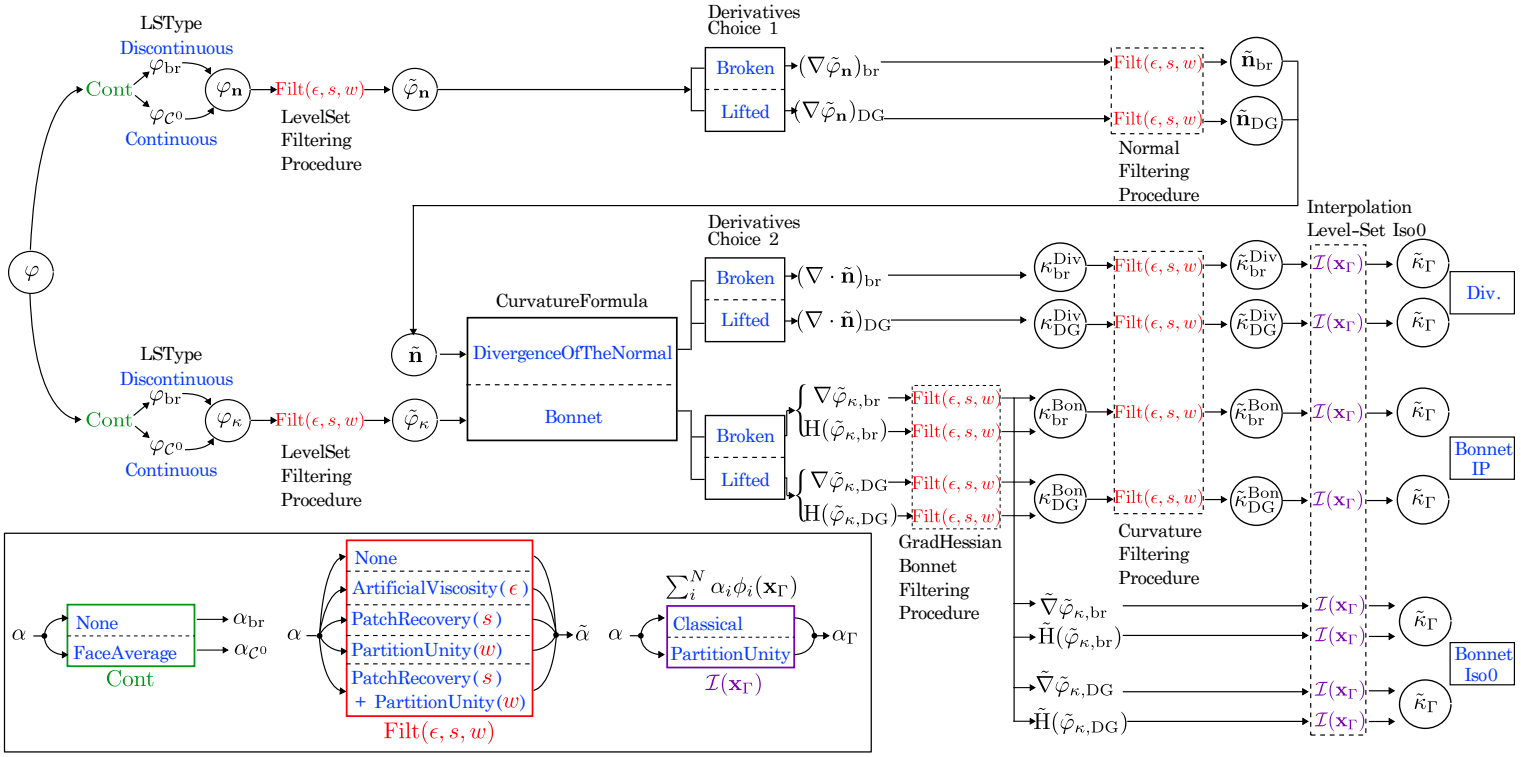


Figure 7.1 / Flowchart showing the organization of the various elements involved in the curvature computation and filtering procedure.

7.2.2 The artificial viscosity method

The first approach is a PDE-based filter that has been used in Tornberg and Engquist (2000), Felix Heimann (2013) and Hashemi et al. (2020). From an input function α, the filter solves a diffusive equation with Neumann boundary equation for the filtered function $\tilde{\alpha}$:

$$\begin{cases} \tilde{\alpha} - \epsilon \Delta \tilde{\alpha} = \alpha & \text{in } \Omega\,, \\ \frac{\partial \tilde{\alpha}}{\partial n} = 0 & \text{on } \partial\Omega\,, \end{cases} \tag{7.6}$$

where the artificial viscosity coefficient ϵ scales with the square of the local mesh size h through a user-defined parameter ϵ_0: $\epsilon = \epsilon_0\, h^2$. Equation 7.6 can be understood as a low-pass filter with the artificial viscosity being related to the inverse of the spatial cut-off frequency. Indeed, assuming $\Omega = \mathbb{R}^d$, the Fourier transforms $\hat{\tilde{\alpha}}$ and $\hat{\alpha}$ of $\tilde{\alpha}$ and α, respectively, verify

$$\hat{\tilde{\alpha}}(\xi) = \frac{1}{1 + \epsilon |\xi|^2} \hat{\alpha}(\xi)\,,$$

which motivates the choice $\epsilon = \epsilon_0\, h^2$. This filter has the property to preserve the signal mean while reducing its energy in the sense that $\int_\Omega \tilde{\alpha}\, \mathrm{d}V = \int_\Omega \alpha\, \mathrm{d}V$ while $\int_\Omega \tilde{\alpha}^2\, \mathrm{d}V \leq \int_\Omega \alpha^2\, \mathrm{d}V$. Equation 7.6 is discretized by means of the Symmetric Interior Penalty (SIP) DG method (see Section 5.2.3). The constant matrices resulting from the discretization are assembled once at the beginning of the simulation and are then used to solve the linear system by LU-factorization at every time step. More precisely, `Argo` being a solver for generic non-linear systems of conservation laws, it relies on a Newton method. This involves the construction of the Jacobian matrix of the residual with respect to the state-vector. Since for a linear conservation law with homogeneous Neumann boundary conditions, the matrix of the discrete system equals the Jacobian matrix of its residual, we used the pre-existing framework to simplify the implementation of the matrix assembly procedure.

7.2.3 The patch-recovery method

Consider an element $\mathbb{T}_m$, and denote by $\boldsymbol{\phi}_{m,-}(\mathbf{x})$ the column vector of its shape functions such that the function to filter on this element, α_m, writes

$$\alpha_m(\mathbf{x}) = \boldsymbol{\phi}_{m,-}^T(\mathbf{x})\, \mathbf{A}_{m,-}\,,$$

with $\mathbf{A}_{m,-}$ the column vector containing its nodal degrees of freedom. Associated with each of the elements $\mathbb{T}_m$, a patch of vertex-based neighboring elements

$$\mathbb{T}_{m,-}^{\text{patch}} = \{\mathbb{T}_{m,1}^{\text{patch}}, \dots, \mathbb{T}_{m,j}^{\text{patch}}, \dots, \mathbb{T}_{m,N_m}^{\text{patch}}\}$$

is built, containing N_m elements with the first one being the element $\mathbb{T}_m$ itself, i.e. $\mathbb{T}^{\text{patch}}_{m,1} = \mathbb{T}_m$. The patch-recovery technique of Kummer and Warburton (2016a) consists in computing for each $\mathbb{T}_m$ the projection of α on a polynomial space defined on the entire patch, i.e. extending over several adjacent elements. This new polynomial space is spanned by the shape functions $\boldsymbol{\phi}^{\text{patch}}_{m,-}$. Following the ideas of R. Saye (2017b), weighting factors $C^{\text{patch}}_{m,-}$ giving more importance to the central element are introduced in the projection operator. In practice, $C^{\text{patch}}_{m,1} = 1.0$ is set and $C^{\text{patch}}_{m,j} = 0.5$ for $j = \{2, \dots, N_m\}$. The L_2 projection then reads: for each $\mathbb{T}_m$, find α^{patch}_m such that

$$\sum_{j=\{1,\dots,N_m\}} C^{\text{patch}}_{m,j} \int_{\mathbb{T}^{\text{patch}}_{m,j}} \left(\alpha^{\text{patch}}_m - \alpha_j\right) \boldsymbol{\phi}^{\text{patch}}_{m,-} \, \mathrm{d}V = 0 \,. \tag{7.7}$$

The filtered field $\tilde{\alpha}_m$ is then obtained by projecting the restriction of α^{patch}_m to $\mathbb{T}_m$, i.e. $\alpha|_{\mathbb{T}^{\text{patch}}_{m,-}}$, on the polynomial space spanned by $\boldsymbol{\phi}_{m,-}$. In practice, several sweeps of the patch-recovery procedure are performed. This implies that at the beginning of each new sweep, the functions α_j in Equation 7.7 are replaced by the filtered functions computed from the previous sweep.
It should be noted that in this work, the degree of the polynomial space on which the projection is performed is chosen to be identical to the original polynomial space. As such, the construction of the patch-based shape functions $\boldsymbol{\phi}^{\text{patch}}_{m,-}$ is greatly simplified as it is done by extending the shape functions $\boldsymbol{\phi}_{m,-}$ to the neighboring cells $\mathbb{T}^{\text{patch}}_{m,j}$ for $j = \{2, \dots, N_m\}$. Example of such extension are illustrated in Figure 8.6 and Figure 8.7. This extension is encompassed in a matrix-vector multiplication involving the local shape functions $\boldsymbol{\phi}_{j,-}$ on $\mathbb{T}^{\text{patch}}_{m,j}$ and a coupling matrix $\underline{\boldsymbol{Q}}_{m,j}$ such that

$$\boldsymbol{\phi}^{\text{patch}}_{m,-}\Big|_{\mathbb{T}^{\text{patch}}_{m,j}} = \underline{\boldsymbol{Q}}_{m,j} \, \boldsymbol{\phi}_{j,-} \,. \tag{7.8}$$

The coupling matrix embodies the projection of the smooth shape function extension onto the considered element in the patch. This idea is taken from the work of Muller et al. (2016) about the cell-agglomeration procedure and is further detailed in Section 8.2.5. The expression of this coupling matrix is given here for completeness:

$$\begin{aligned} \underline{\boldsymbol{Q}}^{*}_{m,j} &= \int_{\mathbb{T}^{\text{patch}}_{m,j}} \boldsymbol{\phi}^{\text{patch}}_{m,-}\Big|_{\mathbb{T}^{\text{patch}}_{m,j}} \boldsymbol{\phi}^{T}_{j,-} \, \mathrm{d}V \,, \\ \underline{\boldsymbol{Q}}_{m,j} &= \underline{\boldsymbol{Q}}^{*}_{m,j} \, \underline{\boldsymbol{M}}^{-1}_{j} \,, \end{aligned} \tag{7.9}$$

with $\underline{\boldsymbol{M}}_j$ the local mass matrix of element $\mathbb{T}^{\text{patch}}_{m,j}$ to which shape functions are extended. Notice that $\underline{\boldsymbol{Q}}_{i,i}$ is the identity matrix and therefore

$$\tilde{\alpha}_m = \alpha^{\text{patch}}_m\Big|_{\mathbb{T}_m} = \boldsymbol{\phi}^T_{m,-}\,\mathbf{A}^{\text{patch}}_{m,-}\,, \tag{7.10}$$

with $\mathbf{A}^{\text{patch}}_{m,-}$ the degrees of freedom of α^{patch}_m in basis $\boldsymbol{\phi}^{\text{patch}}_{m,-}$ which are obtained by solving Equation 7.7. It is convenient to manipulate $\underline{\boldsymbol{Q}}_{m,j}$ because they can be assembled in a pre-processing step and re-used during the simulation every time the extension over an adjacent element is required. In the same way, the other expensive steps that are the construction of the patches and the assembly of the inverse of the extended mass matrices and the matrices for the right-hand-side of Equation 7.7 are performed only once at the beginning of the simulation.

7.2.4 The partition of unity method

The last filter considered in this work relies on the concept of partition of unity and comes from the work of R. Saye (2017b). It was already introduced in Section 6.6.2. Consider the same neighboring patch elements construction as for the patch-recovery technique. The filtered field $\tilde{\alpha}_m(\mathbf{x})$ at a position $\mathbf{x}$ located in element $\mathbb{T}_m$ is obtained by a normalized linear combination over the local patch between some weights functions $g_j(\mathbf{x})$ and the functions $\alpha^{\text{patch}}_j(\mathbf{x})$,

$$\tilde{\alpha}_m(\mathbf{x}) = \frac{\sum_{j=\{1,\dots,N_m\}} g_j(\mathbf{x})\alpha^{\text{patch}}_j(\mathbf{x})}{\sum_{j=\{1,\dots,N_m\}} g_j(\mathbf{x})}\,, \tag{7.11}$$

where $\alpha^{\text{patch}}_j(\mathbf{x})$ for $j \neq 1$ are functions evaluated outside their reference element $\mathbb{T}^{\text{patch}}_{m,j} \in \mathbb{T}^{\text{patch}}_{m,-}$ thanks to a smooth extension of the local shape functions into $\mathbb{T}_m$ through the coupling matrices, Equation 7.9. The weights functions are defined as

$$g_j(\mathbf{x}) = \omega((x - x^c_j)/r_j)\,\omega((y - y^c_j)/r_j)\,,$$

with r_j the radius of the largest ball that can fit inside the patch associated to element $\mathbb{T}^{\text{patch}}_{m,j}$ centered at (x^c_j, y^c_j). The one-dimensional local bump function ω is defined as

$$\omega(\theta) = \begin{cases} \exp\left(C\frac{1}{\theta^2-1}\right) & \text{if } |\theta| < 1\,, \\ 0 & \text{if } |\theta| \geq 1\,, \end{cases}$$

where the constant C control the width of the function. Its value is fixed at $C = 32/9$ according to R. Saye (2017b).

When the patch-recovery method is used in combination with the partition of unity approach, the nodal expansion weights of the functions evaluated over the patch, α^{patch}_j, are obtained through a patch-recovery procedure before being

combined with their associated smoothly extended shape functions. Similarly to the patch-recovery, the neighbors search is done once when starting the simulation, as well as the computation of the inscribed balls.

Closing remark As a closing remark for this section, one can notice that each of the three filtering approaches studied here depends on user-defined parameters which are *a priori* not obvious to set. The artificial viscosity requires a value for the viscosity coefficient, the patch-recovery depends on the number of sweeps applied, the choice of the neighboring elements and the weighting factors over the patch, and the partition-of-unity is based on the expression of the bump function and its extent. This makes the design of a completely generic filtering strategy a challenging task.

7.3 Assessment of the various strategies

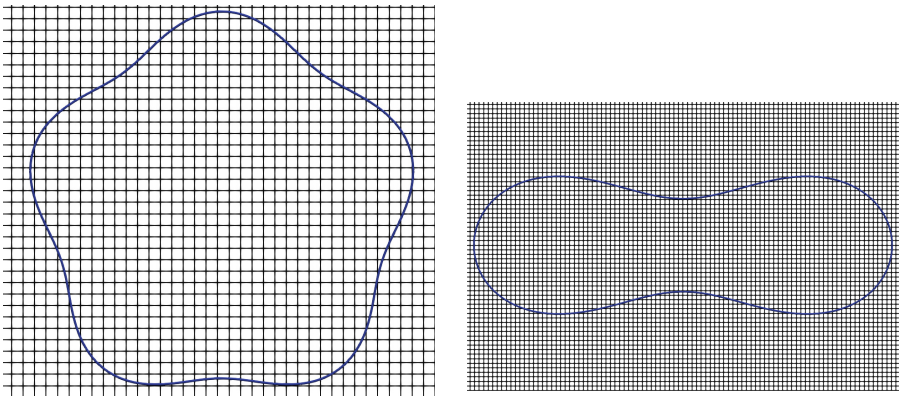
Although surface tension computation is only relevant for unsteady test cases, the investigation here focuses on steady test cases. As pointed out by Coquerelle and Glockner (2016), the accurate computation of surface tension forces is dictated by two main criteria about interface curvature[2]: (i) a fast convergence of the numerical curvature, (ii) minimized oscillations of the curvature along the interface. Because of the huge number of possible combinations the flowchart in Figure 7.1 contains, the objective of this first set of test cases is to extract the best features among all the possibilities based on the two identified criteria. We can then proceed with unsteady test cases in Section 11.6 by using the most promising combinations. To assess the first criterion, the L_2 and L_∞ errors on the elements cut by the interface, $\mathbb{T}^\Gamma$, between the computed and the exact curvatures, κ^h and κ^{ex}, are considered:

$$E_{L_2} = \sqrt{\sum_{\mathbb{T}_m \in \mathbb{T}^\Gamma} \int_{\mathbb{T}_m} (\kappa^h - \kappa^{\text{ex}})^2 \, \mathrm{d}V}, \qquad [7.12]$$

$$E_{L_\infty} = \max_{\mathbb{T}_m \in \mathbb{T}^\Gamma} |\kappa^h_{\text{QP}} - \kappa^{\text{ex}}_{\text{QP}}|, \qquad [7.13]$$

where κ^h_{QP} and $\kappa^{\text{ex}}_{\text{QP}}$ are the values of the curvature evaluated at the quadrature points over the element. The order of convergence on the curvature field is then evaluated based on those two error measures. To evaluate the oscillations of the curvature along the interface, the L_2 measure is obtained by integrating the error along the interface, $\mathbb{I}$, while the L_∞ measure identifies the maximum error

[2]The third criterion mentioned in Coquerelle and Glockner (2016) is related to the variation of the curvature along the interface normal direction, which is only important when a diffuse interface treatment is adopted.



a / Starfish geometry. **b /** Peanut geometry.

Figure 7.2 / Two of the interface geometries considered for the static geometrical analysis of the curvature computation, with the underlying finest grid considered.

on the interface quadrature points:

$$E_{L_2,\Gamma} = \sqrt{\sum_{\mathrm{I}_m \in \mathrm{I}} \int_{\mathrm{I}_m} (\kappa^h - \kappa^{\mathrm{ex}})^2 \, \mathrm{d}S}\,, \qquad [7.14]$$

$$E_{L_\infty,\Gamma} = \max_{\mathrm{I}_m \in \mathrm{I}} |\kappa^h_{\mathrm{QP}} - \kappa^{\mathrm{ex}}_{\mathrm{QP}}|\,. \qquad [7.15]$$

The two-dimensional geometries that are selected are a circular droplet, a starfish shape and a peanut shape, defined respectively by

$$\begin{aligned}
\varphi_{\mathrm{drop}} &\equiv \sqrt{(x^2+y^2)} = 0.2 && \text{circle,}\\
\varphi_{\mathrm{star}} &\equiv \sqrt{(x^2+y^2)} - 0.02\sin(5.0\,\mathrm{atan2}(y,x)) = 0.25 && \text{starfish,}\\
\varphi_{\mathrm{pean}} &\equiv \sqrt{((x+1)^2+y^2)} + \sqrt{((x-1)^2+y^2)} + 0.9\cos(x) = 3.0 && \text{peanut.}
\end{aligned}$$

The starfish shape is inspired from T. Chen et al. (2004) while the peanut shape is the one used in Kummer and Warburton (2016a). The iso-0 contours of the starfish and peanut geometries are shown in Figure 7.2, with the underlying finest mesh on which the study is conducted. The analytical expressions of the curvature for both shapes are obtained via symbolic computation using the *Python Sympy* package. Inspiration is taken from the exhaustive study conducted by Kummer and Warburton (2016a) by considering all the possible combinations of the flowchart. More precisely, on a given mesh for a given polynomial order, 1200 simulations are run for the cases where the curvature is obtained from the divergence of the normal (*Div.*), 600 simulations for the Bonnet's formula applied to the interpolation points before interpolation on the iso-0 (*Bonnet IP*) and 120 simulations for the Bonnet's formula directly assembled on the iso-0 from the level-set gradient and Hessian matrix (*Bonnet*

Iso0), for a total of 1920 runs. The user-defined parameters of the various approaches are at this stage not varied and are fixed based on some preliminary investigations. The artificial viscosity on the level-set field is set at $\epsilon_0 = 0.02$ while the one on the level-set derivatives and the curvature is chosen as $\epsilon_0 = 0.05$. The number of patch-recovery sweeps at every stage of the process is taken equal to 2. One can also note that only two distinct filtering methods on the level-set are applied (contrary to four elsewhere), namely artificial viscosity and patch-recovery. The study is here limited to a single polynomial order $p = 3$ and the orders of convergence are computed based on only two meshes, the finest one being the mesh for which the values of the errors are reported. The circle and starfish geometries are considered in a square domain $[-0.5, 0.5]^2$ while the peanut geometry is defined in the rectangular domain $[-2, 2] \times [-1, 1]$, the three of them discretized by a Cartesian grid. The two grids for the droplet test case contain 16×16 and 32×32 elements, the ones for the starfish test case 32×32 and 64×64 elements and the ones for the peanut test case 64×32 and 128×64. The results are presented in Figure 7.3a, Figure 7.3b, Figure 7.3c and Table 7.1, Table 7.2, Table 7.3. From a general point of view, it appears that:

- Compared to the base case scenario, the filtering approaches reduce all the considered errors by one to several orders of magnitude (note the logarithmic scaling in figures Figure 7.3a, Figure 7.3b, Figure 7.3c).
- While the convergence rates for the curvature in terms of the L_2 and L_∞ norms for the base case scenario are in line with the theoretical $p - 1$ values, the ones for the filtered situations are significantly increased.
- Not surprisingly, the use of filtering techniques increases the CPU time. This must be nuanced by the fact that only the initialization step of the solver is performed here, which involves all the most computational expensive operations related to the filtering methods.

The results also allow drawing up a first list of conclusions regarding the choice of the various methods and parameters in the global flowchart:

- *Curvature formula*: the curvature formula is separated in three approaches for the analysis (see Figure 7.1). For the circle geometry, the divergence of the normal seems to perform globally as good or even slightly better than the Bonnet's formula applied to the interpolation points. For the starfish and peanut shapes, both approaches based on Bonnet's formula outperform the divergence of the normal. This seems to suggest that for general complex geometries, the use of Bonnet's formula is more accurate because it involves one less interpolation step.
- *Level-set Type*: the use of the *continuous* or *discontinuous* level-set field to start the process does not have an appreciable effect on the efficiency of the strategies. Since the continuity enforcement is realized though a simple face averaging and that the interface shape is pretty well approximated by the third order polynomial used here, the deviation between the two level-set fields is only minor.

- *Derivatives choice*: the choice of the lifted derivatives clearly offers the most effective filtering. This looks consistent since taking into account the inter-elements jumps and ensuring a global coupling of the derivatives calculations should favor the stability of the scheme.
- *Fields to filter*: it is manifest that the level-set field itself should not be filtered, or at least not with the parameters and the methods considered here. This indeed perturbs too much the interface shape and leads to inaccurate curvature estimations. For the divergence of the normal, it looks like no filtering on the normal field is required. It turns out that filtering should be applied on the level-set gradients and Hessian matrix (for Bonnet's formula) and/or on the final curvature field (for Bonnet's formula on IP and divergence of the normal). Bonnet's formula on IP, applying two successive filters (possibly different) on the gradient and Hessian and then on the curvature looks promising. Those observations seem to be in contradiction with the ones in Kummer and Warburton (2016a) in which is it advocated to only filter the level-set field. This could be explained by the different parameters selected and also by the fact that they project their level-set field on a higher-order polynomial space before starting the process. When they used a continuous level-set belonging to the original space, they find that filtering the gradient and Hessian of the level-set was optimal.
- *Filtering methods*: the most effective filtering technique among the four considered appears to be the succession of the patch-recovery and partition of unity filters. As already suggested by R. Saye (2017b), those two techniques provide two complementary filtering effects. It must be highlighted that the artificial viscosity filter, which is barely present here, could potentially be as successful as the others by choosing the optimal viscosity coefficient (which is not *a priori* obvious).
- *Interpolation on level-set iso-0*: interestingly, common to the different curvature formula, is the benefit in most of the cases of applying a last filtering step through the partition of unity technique during the interpolation from the interpolation points to the quadrature points located on the iso-0.

Table 7.1 / The 8 best identified curvature computation combinations for φ_{drop} test case, based on Figure 7.3a. The 53th-combination is the best among the ones using the Bonnet formula on iso-0. The 1255th-combination is the reference one, i.e. without any filtering and broken derivatives. The columns name refer to Figure 7.1. AV = Artificial Viscosity, PR = Patch-Recovery, PU = Partition of Unity, CL = classical .

Ranking	κ formula	LS type	Deriv. choice 1	Deriv. choice 2	Filt. LS	Filt. normal	Filt. grad-Hessian	Filt. curvature	Interp. iso-0
1	Div.	Discont.	Lifted	Lifted	None	None	N/A	PR+PU	PU
2	Bonnet IP	Discont.	N/A	Lifted	None	N/A	None	PR+PU	PU
3	Div.	Discont.	Broken	Lifted	None	None	N/A	PR+PU	PU
4	Div.	Cont.	Lifted	Lifted	None	None	N/A	PR+PU	PU
5	Div.	Cont.	Broken	Lifted	None	None	N/A	PR+PU	PU
6	Bonnet IP	Cont.	N/A	Lifted	None	N/A	None	PR+PU	PU
7	Bonnet IP	Discont.	N/A	Lifted	None	N/A	AV	PR+PU	PU
8	Bonnet IP	Cont.	N/A	Lifted	None	N/A	AV	PR+PU	PU
53	Bonnet Isoo	Cont.	N/A	Lifted	AV	N/A	AV	N/A	CL
1255	Bonnet IP	Cont.	N/A	Broken	None	N/A	None	None	CL

Table 7.2 / The 8 best identified curvature computation combinations for φ_{star} test case, based on Figure 7.3b. The 29th-combination is the best among the ones using the divergence of the normal formula. The 987th-combination is the reference one, i.e. without any filtering and broken derivatives. The columns name refer to Figure 7.1. AV = Artificial Viscosity, PR = Patch-Recovery, PU = Partition of Unity, CL = classical .

Ranking	κ formula	LS type	Deriv. choice 1	Deriv. choice 2	Filt. LS	Filt. normal	Filt. grad-Hessian	Filt. curvature	Interp. iso-0
1	Bonnet Isoo	Discont.	N/A	Lifted	None	N/A	PR+PU	N/A	PU

2	Bonnet Isoo	Cont.	N/A	Lifted	None	N/A	PR+PU	N/A	PU
3	Bonnet IP	Dis-cont.	N/A	Lifted	None	N/A	PR+PU	None	PU
4	Bonnet IP	Cont.	N/A	Lifted	None	N/A	PR+PU	None	PU
5	Bonnet IP	Dis-cont.	N/A	Lifted	None	N/A	PR	PU	PU
6	Bonnet IP	Cont.	N/A	Lifted	None	N/A	PR	PU	PU
7	Bonnet IP	Dis-cont.	N/A	Lifted	None	N/A	PR+PU	PU	CL
8	Bonnet IP	Cont.	N/A	Lifted	None	N/A	PR+PU	PU	CL
29	Div.	Cont.	Lifted	Lifted	None	None	N/A	PR	PU
987	Bonnet IP	Cont.	N/A	Broken	None	N/A	None	None	CL

Table 7.3 / The 8 best identified curvature computation combinations for φ_{pean} test case, based on Figure 7.3c. The 27th-combination is the best among the ones using the divergence of the normal formula. The 1272th-combination is the reference one, i.e. without any filtering and broken derivatives. The columns name refer to Figure 7.1. AV = Artificial Viscosity, PR = Patch-Recovery, PU = Partition of Unity, CL = classical .

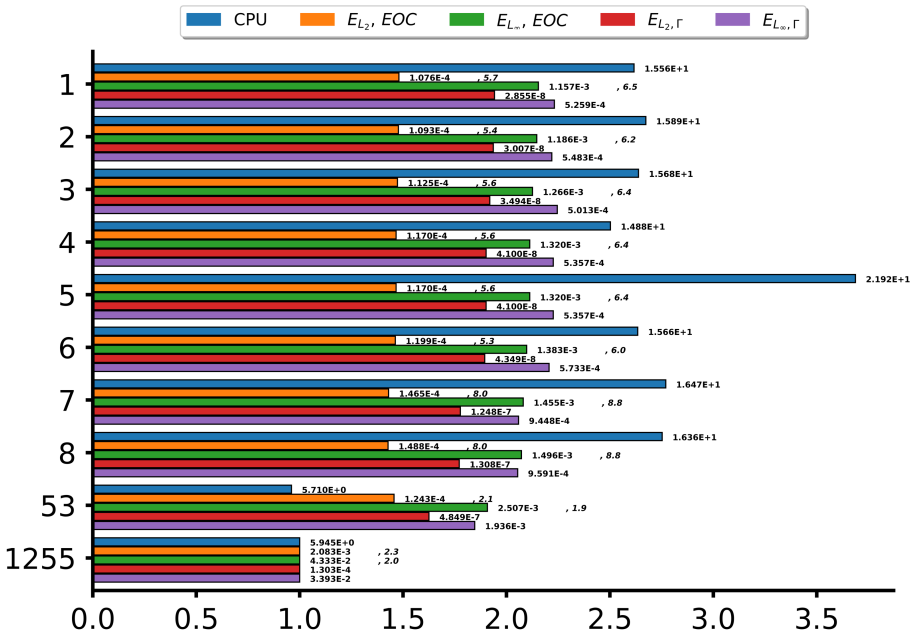
Ranking	κ formula	LS type	Deriv. choice 1	Deriv. choice 2	Filt. LS	Filt. normal	Filt. grad-Hessian	Filt. curvature	Interp. iso-0
1	Bonnet IP	Dis-cont.	N/A	Lifted	None	N/A	PR+PU	None	CL
2	Bonnet IP	Cont.	N/A	Lifted	None	N/A	PR+PU	None	CL
3	Bonnet Isoo	Dis-cont.	N/A	Lifted	None	N/A	PR+PU	N/A	CL
4	Bonnet Isoo	Cont.	N/A	Lifted	None	N/A	PR+PU	N/A	CL

5	Bonnet Isoo	Cont.	N/A	Lifted	None	N/A	PR+PU	N/A	PU
6	Bonnet Isoo	Dis-cont.	N/A	Lifted	None	N/A	PR+PU	N/A	PU
7	Bonnet Isoo	Cont.	N/A	Lifted	None	N/A	PR	N/A	PU
8	Bonnet Isoo	Dis-cont.	N/A	Lifted	None	N/A	PR	N/A	PU
27	Div.	Cont.	Broken	Lifted	None	None	N/A	PR+PU	PU
1272	Bonnet IP	Cont.	N/A	Broken	None	N/A	None	None	CL

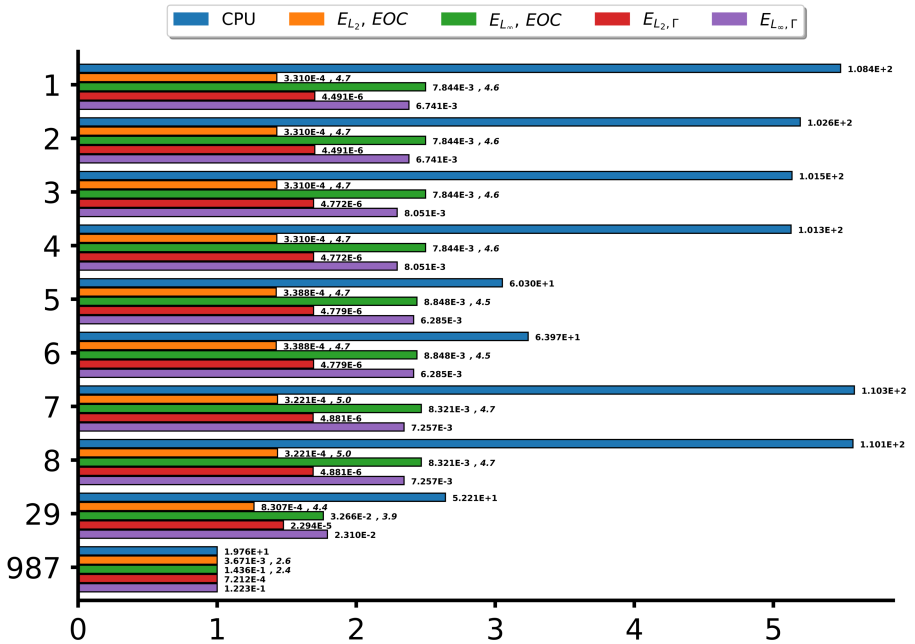
This geometrical study demonstrates the ability of the filtering procedures considered as part of this work to substantially improve the curvature accuracy estimation and decrease its oscillations along the interface; two factors that are expected to play a crucial role in the simulation of surface-tension driven flows. It also allows highlighting, among a huge set of parameters, the best options in the global procedure for the curvature computation and filtering.

To close this section, the beneficial effects of the filtering procedure on the circular droplet configuration is illustrated, still with a polynomial order $p = 3$ on the finest mesh 32×32. In Figure 7.4, the curvature field is extracted along the iso-0 representing the droplet, at the initial time. The left figure shows the highly oscillatory curvature field that is obtained without any filtering, albeit the interface has a basic shape. The curvature along the interface after filtering, visible on the right figure, is on the contrary very smooth and very close to the exact curvature value. From the non-filtered curvature, one can expect the development in time of an oscillatory induced pressure jump, resulting in the generation of non-negligible parasitic velocities[3]. That is precisely what can be seen in Figure 7.5a which corresponds to the situation after three time steps. The simulation crashed at the next time step. Contrastingly in Figure 7.5b, the pressure field with the filtered curvature is free of spurious oscillations and the induced velocities remain small. It must be emphasized that the situation depicted in Figure 7.5b is after 100 time steps (the simulation was voluntary stopped after). This already indicates the importance of curvature filtering to ensure a stable unsteady simulation. This observation will be confirmed by the moving interface test cases in Section 11.6.

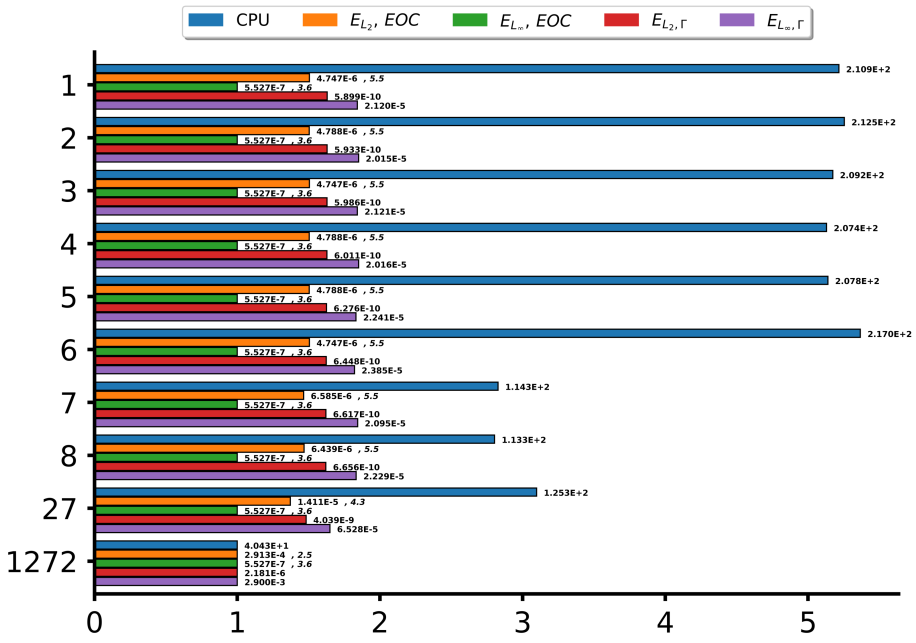
[3]The initial pressure jump to start the simulation is prescribed to the exact value corresponding to the analytical curvature of the circle. Therefore, any induced velocities can be considered as parasitic.



a / Results for φ_{drop} test case.

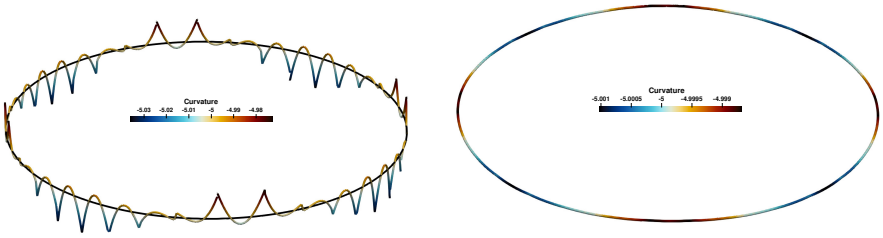


b / Results for φ_{star} test case.



c / Results for φ_{drop} test case.

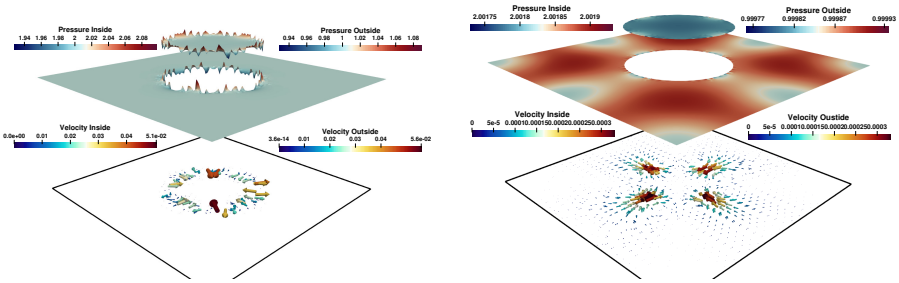
Figure 7.3b / Bar chart presenting the results of the three geometrical curvature accuracy test cases. The five quantities gathered for each situation are the CPU time and the four error measures, Equation 7.12, Equation 7.13, Equation 7.14, Equation 7.15. The vertical axis lists the best curvature computation strategies in terms of the $E_{L_2,\Gamma}$ error (Equation 7.14), with the labels being the rank of each strategy among the 1920 possible combinations. The last one on the bottom is the reference simulation used to plot the five quantities in a dimensionless form. The CPU time is simply divided by the reference one while the errors are scaled in a logarithmic way, i.e. $\overline{E} = \log_{10}(E)/\log_{10}(E_{\text{ref}})$. The horizontal axis labels are then the ratios between the best combinations quantities and the reference ones. The numbers on the right of each bar are the absolute dimensional values. The estimated order of convergence (EOC) for the two errors E_{L_2} and E_{L_∞} are written on the same line after the comma.



a / Curvature without filtering (oscillatory) with the exact constant curvature in black for reference. Computed from continuous level-set, with Bonnet formula and broken derivatives.

b / Curvature with patch-recovery (2 sweeps) + partition of unity filtering. Computed from continuous level-set, with Bonnet formula and lifted derivatives.

Figure 7.4 / Sampling of the curvature along the level-set iso-0 of the circular droplet configuration at the initial time, with a polynomial order $p = 3$. The same elevation factor is used for both images.



a / Without filtering after 3 time steps.

b / With filtering after 100 time steps.

Figure 7.5 / Velocity and elevated pressure fields for the circular droplet test case with time step $\Delta t = 0.02$s.

Contribution 7.1: Analysis of curvature filtering strategies

The main contributions that can be identified from this chapter are:

- The benchmarking of state-of-the-art curvature filtering based on reference static interface test cases. To the best of the author's knowledge, this is the first time that such comparison in the context of a high-order compressible two-phase flow solver and inside a unique computational framework is conducted.
- From a broader perspective, the proposition of some general guidelines about the estimation of interface curvature for surface tension computation based on sharp interface FE type methods.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- Three general filtering techniques that could be applied to any solution fields.
- The arrangement of the different blocks such that all the paths of the flowchart can be easily explored.

Rest of this page intentionally left blank.

8

Sharp interface spatial discretization with the eXtended DG method

The material of this chapter supports the content of the two following chapters. The material of Section 8.2 constitutes a part of the work published in Henneaux et al. (2020b) and Henneaux et al. (2023f) and presented in Henneaux et al. (2020c), Henneaux et al. (2021d), Henneaux et al. (2022b) and Henneaux et al. (2022c).

8.1 Overview of discretization approaches for discontinuous fields across interfaces and the choice of the XDG method 266
8.1.1 General overview about discontinuities handling 266
8.1.2 Overview of Finite Element type geometrically unfitted methods 269
8.1.3 Main challenges of unfitted methods 271
8.1.4 Advantages of high-order unfitted methods and the choice of the XDG method 274
8.2 The eXtended discontinuous Galerkin method 277
8.2.1 Cut mesh and extended function space definitions 277
8.2.2 Modifications of the discrete variational form 281
8.2.3 Enforcement of continuous level-set iso-0 surface 285
8.2.4 Integration over implicitly-defined surfaces 285
8.2.5 Cell agglomeration: treatment of small cut elements . . . 288
8.3 Implementation aspects of the XDG framework 291
8.3.1 Assembly of agglomerated residuals and jacobians . . . 291
8.3.2 Artificial viscosity technique in cut elements 303
8.3.3 Mesh-to-mesh projection 306

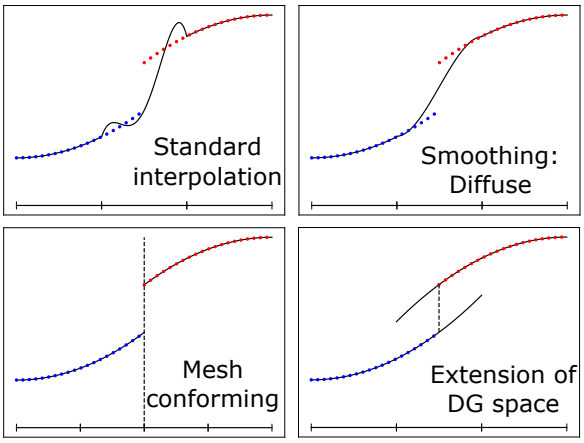


Figure 8.1 / Illustrative sketch of the main approaches to discretize a discontinuous solution. The dots represent the exact solution, the continuous black lines the discretized one.

The spatial discretization of the governing equations in the bulk of each phase was covered in Chapter 5 and the way to represent and follow a moving interface in the domain was addressed in Chapter 6. The question now is to see how discontinuous fields across moving interfaces can be sharply discretized in a high-order manner.

8.1 Overview of discretization approaches for discontinuous fields across interfaces and the choice of the XDG method

8.1.1 General overview about discontinuities handling

The way the interface is represented and followed is of course strongly related to the way the numerical method deals with the field discontinuities across the phase interface. Most often in the literature, reviews about existing methods to treat discontinuities are organized based on the various techniques to follow the interfaces (e.g. level-set versus VOF methods). However, our feeling is that discontinuities representation and interface tracking remain two distinct features inside a numerical solver and this is the reason why an overview about the main approaches to deal with discontinuous fields is provided here, separated from the overview given in Section 6.1. Figure 8.1 already gives an illustrative comparison between the different ways of handling a discontinuity in the solution to be discretized.

Interface fitting the mesh faces If the numerical method intrinsically allows for solutions jumps between elements, the discontinuities at the interface can naturally and sharply be captured if the interface is represented by mesh faces at all times (which can be achieved by means of any methods reviewed in the first paragraph of Section 6.1). There is no need to locally modify the discretization. This has been done for two-phase compressible flows in the context of FVM in Chalons et al. (2017) and DGM in Nguyen et al. (2010). For FEM where the solution is by construction continuous between elements, two options can be considered. The first one is to consider discontinuous basis functions only at the interface while keeping C^0 finite element basis functions elsewhere like in Yu Zhang et al. (2019). The other possibility consists in adding degrees of freedom on the interface to be able to decouple the solutions from both phases. This has been achieved with the dummy-node technique in Potghan and Jog (2022).

Interface cutting across the mesh cells If the background mesh is not adapted such as to fit the interface, two approaches already mentioned before are possible to handle solution singularities: sharp interface and diffuse interface methods. In the framework of *sharp interface methods* (*SIM*), the interface is treated as a discontinuity without thickness. Thus, the general benefit is that the singular nature of real, immiscible, fluid-fluid interfaces (at the continuum scale) is exactly resolved. There are three main strategies to accomplish this:

- *Extension of the approximation space*: one way to obtain a sharp representation is to locally modify the approximation space of the scheme. By considering two distinct sets of solutions variables on cells intersected by the interface, it is then possible to couple them together such that the interface jumps are exactly captured. In the FVM community, the ghost-fluid method (GFM) is the most used: the solution from one side of the interface is extrapolated to the other side to populate a layer of ghost cells such that numerical fluxes associated to a given phase in the cut cells can be computed as if there was only a single phase (Fedkiw et al. (1999)). The interface position needs of course to be taken into account during the integration of the fluxes on the cut cells. Concerning FE-based methods, the central idea is to enrich/extend locally the original approximation space so that the linear combination between the solutions nodal degrees of freedom and the new basis functions spanning the space will result in a discontinuous field. Section 8.1.2 is devoted to a more detailed review about this kind of methods.
- *Local cut-cell construction*: based on the seminal work of Hu et al. (2006), numerous cut-cells methods have been developed over the past decade. The essence of these methods can be summarized in the following steps: (i) on the cells that are cut by the interface (usually identified by a level-set on a Cartesian mesh), the interface is reconstructed with piecewise linear or higher-order polynomials, (ii) volume fractions of both phases in each cut cell are computed, (iii) cut cells with too small volume fractions are

merged with neighboring ones to avoid numerical issues, which creates an unstructured mesh around the interface region, (iv) the solutions from both phases are projected onto the unstructured mesh using the volume fractions, (v) the numerical fluxes at the non-interface cells faces and the ones across the reconstructed interface are assembled, (vi) the discretized governing equations are advanced in time based on the numerical fluxes to obtain the solutions at the next time step, (vii) finally the solutions in the interface region are redistributed to the original Cartesian mesh in a conservative manner. The discontinuities in the solution fields are thus handled by inserting cell faces coinciding with the reconstructed interface such that they can naturally be accommodated by the numerical scheme employed. The interface exchanges between the phases are realized through the interface fluxes. To reduce the inaccuracies introduced by the cut-cells construction, those methods usually make use of mesh adaptation around the interface like the Adaptive Mesh Refinement (AMR). In the context of compressible two-phase flows, the work of Chang et al. (2013) and Lin et al. (2017) based on FVM and the one of X.-L. Deng and M. Li (2018) based on DGM can be mentioned. The main challenges of the cut-cell methods are the complexity of small cell merging process and topology changes handling. Embedding a complex-shaped interface on a fixed background mesh to form a composite mesh made of standard cells and cells with curved faces is also an approach followed in the FE community, like e.g. in Johansson and Larson (2013). More details about the application of this type of methods to two-phase flows are given in Section 8.1.2.

- *Use of surrogate boundary*: the last strategy seeks to preserve the favorable properties of the original discretization scheme in the absence of discontinuity by shifting the location of the material interface to a surrogate boundary, typically a mesh face. The jumps can then be naturally handled by the discontinuous representation of the solution between elements, in the case of FVM or discontinuous FE-type methods. To preserve the order of accuracy, some corrections are applied during the computation of the fluxes at the mesh face accommodating the solution discontinuity. We can mention the FIVER approach in the FV framework (Farhat et al. (2012)) and the Shifted Boundary Method (SBM) in FE (Main and Scovazzi (2018)).

Resolving and explicitly tracking interfaces in a sharp interface context make it possible to produce high-fidelity simulations of interfacial phenomena. This level of accuracy comes however at the expense of a great flexibility for industrial-scale simulations of multiphase flows. When interfaces inside the flow span a wide range of length scales, it is often necessary to abandon the explicit tracking and shift to a diffuse interface philosophy.

The principle of the *diffuse interface methods* (*DIM*) is to consider that the interface is rather a region of non-zero but finite thickness over which the flow variables evolve rapidly but continuously. Two main approaches rely

on that concept: the phase-field and the multi-fluid approaches. Because the phase-field approach considers physically diffuse interfaces at a microscale level, it is particularly well-suited to the simulation of phase transition in very small systems, therefore going beyond the scope of application of this paper. The multi-fluid methods on the other hand are designed to solve a single set of equations over the whole domain derived from averaging procedures of the one-fluid Euler or Navier-Stokes equations. Thanks to the averaging (usually a volume average), fluid variables exhibit a smooth transition in a mixture region around the interface. Hence, the difficulties of representing the discontinuous interface is avoided. Instead, they are shifted to the treatment of the non-conservative products terms that appear in the governing equations to discretize. Those must be modeled such as to ensure hyperbolicity of the system. To cite just a few references, those difficulties have been addressed for FVM in Furfaro and Saurel (2015) and for DGM in Franquet and Perrier (2012). The other advantage of this setting is that is has an intrinsic handling of the interface, avoiding any need for interface tracking or reconstruction, as well as the capability to dynamically create interfaces that are not present initially. The inherited drawback of the DIMs is the time-increasing interface smearing which may incur excessive artificial diffusion. Different strategies have recently been developed to limit the interface diffusion, for example, introducing diffusion-sharpening terms like in Jain et al. (2020) or using the tangent of hyperbola for interface capturing (THINC) approach proposed by Shyue and F. Xiao (2014). Recent reviews about this vast area of multi-fluid models and methods can be found in Maltsev et al. (2022) and Saurel and Pantano (2018).

8.1.2 Overview of Finite Element type geometrically unfitted methods

The more focused review below is intended to motivate the choice of the extended discontinuous Galerkin method and situate it among the other available methods. Several denominations can be found in the literature to identify one or the other geometry independent FE discretization. Here the term "geometrically unfitted FEM" embraces all of those "extended/ cut-cell/ unfitted/ embedded/ immersed" methods. Distinguishing features of some of them are presented hereafter. The central idea of the geometrically unfitted FE-type methods is to embed a complex geometry into a geometrically simple domain that can easily be meshed. To recover the original problem, there are then two possibilities (see e.g. Engwer (2009, Sec. 3.3), Westerheide (2018, Sec. 1.3)) that are discussed below.

Embedding domain techniques The first approach under the embedding domain techniques umbrella consists in extending the original PDE to the simpler domain in such a way that the solution of this new PDE matches the one of the original equation on the original complex domain. Those methods are

also known as fictitious domain methods (P. Sun (2019)). A classical FEM is applied to discretize the problem on the simplified domain and the conditions on the original boundaries are imposed as constraints via the technique of Lagrange multipliers. The finite cell methods are also part of this class of methods but it avoids the introduction of any constraints by penalizing the material parameters of the equation of interest (Schillinger et al. (2013)). The numerical complications brought by the constraints imposition and the need for specific adjustments of the methods depending on the problem do not make the above methods ideal candidates for the targeted applications. The recent shifted boundary method (SBM) by Main and Scovazzi (2018) leverages a surrogate boundary strategy and is more flexible than the previously mentioned approaches. The key idea is to shift the location of the boundary where conditions are applied and to appropriately modify the shifted boundary conditions in order to maintain the convergence rates of the overall formulation (Atallah et al. (2022)). The main advantage is that the small cut-cell issue of some unfitted discretizations is naturally avoided. Another general and versatile embedding domain technique, sharing some features with the cut-cells methods mentioned earlier, has been presented in Johansson and Larson (2013) and R. Saye (2017a) for DGM. There the original PDEs are not modified. Instead, a new mesh is assembled from the intersection between a background Cartesian mesh and the interface. The problem is discretized on this new mesh composed of standard elements and elements with curved faces, keeping the same polynomial basis everywhere.

Modified basis functions In the second class of methods, the basis functions of the standard FE approximation space are modified according to the complex original geometry of interest. The first subcategory includes the eXtended FEM (XFEM) (Thomas-Peter Fries and Belytschko (2010)) and its derivatives. The principle is to enrich/extend the discrete approximation space by additional basis functions with compact support which enable to accurately represent desired features within finite elements in the interface zone. For example, modeling jumps across the interface can be realized by multiplying the additional basis functions with an enrichment function which involves a piecewise constant step function. Adding to the original interpolation the contribution of the linear combination between the additional degrees of freedom and the new basis functions allows to perfectly represent a discontinuous field. The introduction of additional unknowns and the dependency of the enrichment to the problem considered limit the applicability of XFEM. The second subcategory comprises the more flexible unfitted FEM by A. Hansbo and P. Hansbo (2002). There the classical FE approximation space is cut off at the interface, which would amount to multiply the original space by a Heaviside enrichment function in a XFEM context as pointed out by Benedikt Schott (2017, Sec. 2.2). Essential interface boundary conditions are imposed in a weak sense using Nitsche's method (Nitsche (1971)). Recent improvements of the unfitted FEM, which have been coined under the name of cut FEM (Burman et al. (2015)), consider

additional stabilizing terms in the weak form, known as ghost penalties.

8.1.3 Main challenges of unfitted methods

To take advantage of the potential benefits offered by geometrically unfitted FE-type methods, three computational challenges in comparison to standard mesh-fitting FEM must be overcome (see e.g. de Prenter et al. (2022), Benedikt Schott (2017, Sec. 1.2.1)).

Critical intersections Because the mesh is not fitted to the interface, unfortunate cuts resulting in elements with very small intersections or so-called sliver elements can cause stability and conditioning issues and sometimes affect accuracy. To be more precise, according to de Prenter et al. (2022) there are two distinct, albeit strongly related, facets to the small-cut-cell problem:

- *Conditioning and accuracy*: in small cut cells, the support over which the basis functions are defined is truncated. For some of them, the magnitude of the volume and boundary integrals that remain within the small support will be much lower than for the others. This causes the apparition of eigenvalues of the system matrix which can be arbitrary close to zero. As explained in de Prenter et al. (2022), it is the lack of a lower bound to the eigenvalues that generally causes the ill-conditioning of the system. To remedy this source of tiny eigenvalues, a possibility is to re-scale the basis functions, as proposed in Prenter et al. (2017). Another issue associated to the trimming operation in the small cut cells is the lost of linear dependency between some of the local basis functions. This results in the near linear dependence of row and columns of the matrix and therefore to an ill-conditioned system. This problem cannot be solved by rescaling and appropriate preconditioning is then required, as detailed in Prenter et al. (2017). In this work, we mostly rely on the use of direct linear solvers for reasons which are elucidated later. It might seem at first sight that contrary to iterative techniques whose convergence is plagued by ill-conditioning, direct solvers should not experience particular difficulties. However, the severity of the small cut cells on the conditioning is such that direct solvers (as well as iterative ones) might be affected by accuracy issues. Backward stable methods such as Gaussian elimination with partial or complete pivoting, which are guaranteed to produce small residuals[1], will not be able to ensure a small level of errors in the presence of very important condition number, κ. Indeed, it is demonstrated by Watkins (2004, Ch. 2.7) that the loss of precision with respect to condition number associated to floating-point arithmetic scales as $\log_{10}(\kappa)$.

[1] If it was possible to do infinite-precision arithmetic, direct linear solvers would be able to solve exactly any linear systems. In practice however, because of round-off errors, the returned solution will not be exact and will lead to a residual measuring how close the numerical solution is not satisfying the original equation.

- *Stability*: to impose essential boundary conditions, many methods including ours (through the interior penalty method) rely on some version the classical Nitsche's method. This method requires stabilization to ensure the coercivity of the variational form. The corresponding stabilization parameter is typically chosen as an elementwize constant proportional to the ratio between the perimeter and the area of an element (see Equation 5.30). In cut elements, this ratio must be expressed based on the active part of the element and can therefore become arbitrary large. A straightforward application of the Nitsche's method to such scenario would therefore require an unbounded penalty parameter, which would deteriorate the stability estimate, the convergence rate and cause ill-conditioning. An important remark to make at this point is that not only small cut elements but also the so-called sliver-like cut configurations where the physical active volume vanishes but not the physical boundary (e.g. for interfaces parallel and close to an element face) can pose stability issues, as explained in Benedikt Schott (2017, Sec. 3.4.1) and Prenter et al. (2018). Strategies to overcome unbounded penalty parameters include the use of the alternative non-symmetric interior penalty method which only requires the penalty parameter γ in Equation 5.30 to be greater than zero, at the expense of a non-symmetric form.

In addition to the specific techniques mentioned earlier, methods have been developed in the field of unfitted discretization to cope with the different issues all at once. In FEM, the most widely used strategy is the ghost-penalty method of Burman (2010): weakly consistent stabilization operators are added in the interface region to control the solution outside the physical domain while increasing the contribution of basis function with small supports within the physical domain. Recent geometrically unfitted FEM resorting to this stabilization strategy are usually referred as *CutFEM* (Burman et al. (2015)). Instead of altering the variational form, the recently proposed aggregated FEM (*AgFEM*) of Badia et al. (2018) modifies the finite element space in troubled cells by constructing a discrete extension operator between well-posed and ill-posed degrees of freedom. The main difficulty is to build this operator in such a way that the continuity between adjacent elements is maintained. In the context of DGM, this kind of extensions is very natural as the inter-element continuity is enforced only weakly, thus each local basis function can be chosen independently. This has lead to the development of the unfitted DGM (*UDG*) by Engwer (2009) and F. Heimann et al. (2013) in which problematic cut cells are removed from the discretization and merged with incident ones. Similarly, the eXtended DGM (*XDGM*) developed by Kummer (2012), Kummer (2017), and Muller et al. (2016) uses cell-agglomeration by injecting the solution on an agglomerated basis thanks to cell-local extension operations. Ghost-penalty stabilization has also been applied to DGM under the name of *CutDG* by Gürkan and Massing (2019).

Integration on cut-cells Central to the construction of any FE discretization is the approximation of the integrals in the weak form. While this can be achieved with standard quadrature rules on non-cut elements, evaluating integrals on arbitrary shaped cut elements defined by surfaces that do not have an explicit parametrization requires dedicated quadrature techniques. As the accuracy of the numerical integration is one of the factors directly influencing the accuracy of the resulting FE scheme, it is essential to have an accurate integration technique. The major challenge is to develop schemes which are robust in their construction, ensure a sufficiently accurate approximation and at the same time minimize the number of quadrature points to retain efficiency. There are several major groups of methods to perform numerical integration over implicitly-defined geometries; an overview of some them can be found in Robert I. Saye (2022). The choice of one method or another will determine the position of the quadrature points (strictly inside the domain of integration or not) and the quadrature weights (strictly positive or not).

Imposition of interface jumps To preserve the stability of the standard FEM or DGM and thus ensure well-posedness of the final discretization, the imposition of the jump conditions (in the context of two-phase flows here) on the unfitted interface must be carefully carried out, both for diffusive and convective effects:

- *Diffusive effects*: for FEM, the unfitted discretization precludes the strong imposition of Dirichlet boundary conditions as in the boundary-fitted case. A weak imposition of the interface jumps is therefore usually followed, which requires special measures to control induced instabilities. Among the different strategies available (see e.g. Chouly (2022) for an overview), Nitsche's method is particularly well suited for interface conditions imposition because it is easy to implement and it ensures fundamental properties like inf-sup stability and optimal *a priori* estimates (Burman and Zunino (2011)). The dependence of the stability parameter to cut cells scenarios appearing in Nitsche's method can be accommodated with the techniques for critical intersections described above. Likewise, UDGM, XDGM and CutDGM also use penalty interface terms to impose jump conditions related to elliptic operators.
- *Convective effects*: interface instabilities which emanate from the non-linear convective terms need also to be dealt with. In FEM, they can be balanced by consistently adding stabilizing terms allowing to control the inflow from one phase to the other depending on the flow direction as in B. Schott et al. (2016). In a DG context, the naturally inherited upwind flux term can simply be adapted on the interface to couple both phases, as it is done in Kummer (2017) (XDGM). When ghost-penalty is used (CutDGM), the additional stabilization terms take into account flow direction (Engwer et al. (2020) and Gürkan et al. (2020)).

An additional challenge in the design of an unfitted method to simulate interface

problems is therefore to consistently add to the variational form the stabilizing interface terms and to specify them in such a way that the interface jump conditions of the physical situation considered are weakly enforced. As this is not a trivial task, the whole Chapter 9 is devoted to this specific aspect.

8.1.4 Advantages of high-order unfitted methods and the choice of the XDG method

At this stage, the three main advantages that an approach based on a geometrically unfitted FE-type method can potentially offer for simulating separated immiscible two-phase flows are highlighted.

High-order in the bulk and at the interface The approximation of the solution in a discrete function space constructed from the mesh provide FE-type methods with the ability to naturally construct high-order accurate numerical schemes on unstructured meshes. In comparison to equivalent standard low-order methods, the well-known advantages of high-order methods to discretize the equations in the bulk of a phase are at least as valuable when it comes to discretize flows with sharp interfaces, in particular involving compressible effects. As pointed out in Karni (1996) and Glimm et al. (2001), local approximation errors across material discontinuities are critical because they will propagate into the rest of the flow field in a cumulative manner, hence impacting the predictive capability of the simulation tool (contrary to non-linear discontinuities like shocks waves which have a natural self-sharpening mechanism and tend to mitigate errors propagation). To meet the stringent resolution requirements of many applications, capturing interfacial phenomena with a high-order accuracy is therefore a key feature in the development of a numerical method.

Mesh independent of interface location The second asset of unfitted methods is to relieve the coupling between the meshing process and the discretization of the equations. For complex-shaped evolving interfaces, this is particularly appealing as deforming the mesh with the interface can quickly become impracticable. Although recent ALE techniques like direct ALE are able to maintain interface-fitted meshes and optimal convergence orders, those require significant effort to implement (like the construction of space–time control volumes) and the use of high-order meshes in order to accurately represent the interface shape. By avoiding the burden of mesh motion, unfitted methods enjoy high versatility with regard to moving interface problems like two-phase flows. This flexibility inevitably entails some difficulties on other aspects, as explained in the previous section. But unfitted FE-type methods now rely on solid established theoretical foundations which allow to consider their applications to more and more complex physical situations.

Possibility to achieve sharp interface representation Compared to diffusive interface models, the general benefits of using a sharp interface treatment are that (i) the singular nature of immiscible, fluid-fluid interfaces (at the continuum scale) is exactly resolved, (ii) the treatment of non-conservative products and the modeling of interfacial source terms are avoided, and (iii) arbitrary high discontinuities can be captured without the need to rely on any complex limiting and interface-sharpening procedures.

On the choice of the XDG method

At this point, the reasons to rely on a high-order unfitted method for our targeted applications should appear clearly. It now remains to justify the choice of the XDG method among the high-order unfitted methods reviewed before. As far as the choice of the discretization technique is concerned, it turns out that discontinuous FEM are particularly well suited compared to their continuous counterparts for the reasons outlined in Section 5.1.2. Our choice fell on the DG method but other methods like the hybridized DG (HDG) method would have been equally valid candidates. In fact, the HDG method has already been applied to interface problems under the name of eXtended HDG in Gürkan et al. (2019), unfitted HDG in Burman et al. (2021) or cut HDG in Schoeder et al. (2020). It is interesting to note that there does not seem to be a common naming convention since only the last approach mentioned here, called "cut" HDG, is based on the cell-agglomeration procedure while for DGM relying on this same approach, the term "extended" seems to be preferred. Two other DG-based methods that fall outside the scope of geometrically unfitted methods are still important to mention here as they both achieve a sharp interface treatment of viscous compressible two-phase flows. The first one proposed by Robert Nourgaliev et al. (2017) relies on a reconstructed DG (rDG) discretization and takes advantage of the reconstruction procedure to enforce the interface jump conditions via a least-squares approach. The second work presented in Jöns et al. (2023) makes use of a discontinuous Galerkin spectral element method (DGSEM) combined with a local FVM in subdivided cut cells and imposes the jump conditions by means of a diffusive generalized Riemann problem.
The goal of the present work is to build a geometrically unfitted DG solver for two-phase flows. In this regard, four notable methods designed for two-phase flows are here identified. All of them make use of a level-set method to follow the interface:

- *Space-time unfitted DGM* by W.E.H. Sollie et al. (2011) and WEH Sollie et al. (2006): the method proposed in this work relies on a Cartesian space-time background mesh subdivided into time slabs. For a given time slab, all the cells crossed by the interface are subdivided into a tesselation of non-overlapping space-time elements based on a linear approximation of the level-set iso-0. The tessellations define the numerical interface as a collection of straight finite element faces. In every space-time element containing now only one set of flow variables, a local basis

function is defined using a mapping from a reference element. Because the basis functions are defined locally in every element, the space-time flow solution is allowed to be discontinuous across the phase interface. To deal with potential small cut-cells issues, a cell merging procedure is applied during the construction of the tesselations. Because the merged elements tend to have complex shapes, defining a local basis function may be difficult. A simply shaped bounding box element is defined around the merged elements to alleviate that problem. Assembly of the weak form is done by summing the integrations over every element of the tesselation. *Inviscid* compressible two-phase test cases are presented for which a convective interface numerical fluxes coupling the two phases based on an approximate Riemann solver is designed.

- *Unfitted DGM* by Engwer and Felix Heimann (2012) and F. Heimann et al. (2013): this method shares strong methodological similarities with the previous one. A fixed Cartesian background mesh on which the equations are discretized is first defined. On top of it, a refined mesh is built from a uniform refinement. This mesh is solely used to construct a piecewise linear approximation of the interface from which the cut cells are assembled. A cell merging procedure is also used to avoid small cut cells problems. Local basis functions with their support restricted to each cut cell are assembled and scaled according to a bounding box. Finally, the quadrature is realized from a cut-cell sub-triangulation based on the refined mesh to increase accuracy.
- *DG with implicitly defined mesh* by R. Saye (2017a): this approach has some flavor of the cut-cell/embedded domain methods mentioned earlier. However, the noticeable difference is that the cut-cells are here implicitly defined by a level-set without any reconstruction. Hence, not only the field variables but also the interface itself are approximated with high-order polynomials. By evaluating the integrals of the weak formulation with high-order accurate quadrature algorithms developed by the same author in R. I. Saye (2015), this approach allows prescribed interfacial jump conditions to be captured with high-order accuracy. During the construction of the implicitly defined mesh, small cells are naturally merged with nearby cells to ultimately form the interface conforming mesh on which the equations will be discretized.
- *Extended DGM* by Kummer (2017) and Kummer et al. (2021): this last method can be classified in the category of unfitted/cut FEM (despite its name of "extended"). Similarly to the previous implicit defined mesh DGM, the interface is represented with high-order polynomials by the zero iso-contour of the level-set. In the same vein as the two first approaches, the sharp interface treatment is achieved by considering two independent sets of degrees of freedom on the cut cells and truncating the solution at the interface. The major difference here is that there is no need to locally map the reference function space to each cut cell. The original DG space on a cut cell is kept the same and is simply "extended"

by adding a second one. Performing the integration on the respective side of the interface automatically cut off the standard function space. High-order quadrature, presented in B. Müller et al. (2013), which is based on Hierarchical Moment Fitting (HMF) is used. To face the troubles posed by small cut cells, a cell agglomeration procedure is proposed by Muller et al. (2016). It differs from the cell merging technique of the other approaches in the sense that the topology of the original mesh is kept unchanged, hence the assembly of the system can be done without changing the underlying data structures. The weak formulation is assembled on every element as if there was no problematic small cuts. It is only during the assembly of the global system that the integrals over the small cut cells are disregarded. Their contributions are taken into account by a coupling with neighboring elements from which a smooth extension of the shape functions is computed. It is interesting to note that this cell agglomeration technique can also be considered in the context of multigrid methods without the usual hierarchy of meshes. This is explored in Kummer et al. (2021).

Among the previously mentioned UDGM, CutDGM and XDGM, our work builds upon the XDGM because (i) it offers a way to both discretize bulk equations and enforce interfacial jumps with high-order accuracy (see Chapter 9) by using implicitly defined interface and high-order quadrature rules (Section 8.2.4), (ii) it provides a simple strategy to extend existing DG solver to two-phase sharp interface discretization by considering two separated sets of degrees of freedom on cut cells (Section 8.2.1) without the need to locally adapt the mesh topology (contrary to cut-cell type methods), (iii) it makes use of a convenient and efficient agglomeration procedure whose implementation is rather non-intrusive (Section 8.2.5), contrary to the other widespread ghost-penalty stabilization strategy.

8.2 The eXtended discontinuous Galerkin method

8.2.1 Cut mesh and extended function space definitions

Let $\tilde{\Omega} \subset \mathbb{R}^2$ be the two-dimensional open and bounded physical domain with smooth external boundaries $\partial\tilde{\Omega}$ that are one-dimensional manifold. Let $\tilde{\Gamma} \subset \mathbb{R}$ be a one-dimensional smooth internal interface partitioning $\tilde{\Omega}$ into two non-overlapping subdomains $\tilde{\Omega}_1$ and $\tilde{\Omega}_2$ such that

$$\tilde{\Omega} := \tilde{\Omega}_1 \cup \tilde{\Omega}_2 \cup \tilde{\Gamma}\,, \quad \text{with } \tilde{\Gamma} := \overline{\tilde{\Omega}}_1 \cap \overline{\tilde{\Omega}}_2\,.$$

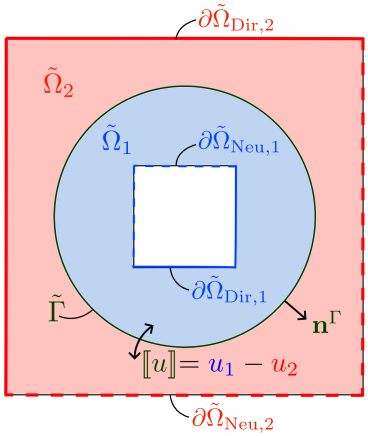


Figure 8.2 / Setting of the two-phase interface problem to be discretized.

Each subdomain $\tilde{\Omega}_k$ is occupied by phase k. The domain boundary $\partial\tilde{\Omega}$ is decomposed into two disjoint subsets $\partial\tilde{\Omega}_{\mathrm{Dir}}$ and $\partial\tilde{\Omega}_{\mathrm{Neu}}$ on which Dirichlet and Neumann boundary conditions, respectively, will be prescribed

$$\partial\tilde{\Omega}_{\mathrm{Dir}} \cup \partial\tilde{\Omega}_{\mathrm{Neu}} = \partial\tilde{\Omega} \,.$$

Subdomains boundaries associated to phase k, $\partial\tilde{\Omega}_k^*$, is the union of the external boundaries $\partial\tilde{\Omega}_k$ and the interface

$$\partial\tilde{\Omega}_k^* := \partial\tilde{\Omega}_k \cup \tilde{\Gamma} \,, \quad \partial\tilde{\Omega}_k \subset \partial\tilde{\Omega} \,.$$

Outward-pointing unit normal vectors defined on the external boundaries are denoted $\mathbf{n}$ while $\mathbf{n}_\Gamma$ is the interface normal pointing outwards from phase $k = 1$ into phase $k = 2$. This setting is sketched on Figure 8.2.

Cut mesh definition

In this work, as explained in Section 6.2.1, the continuous phase interface $\tilde{\Gamma}$ is implicitly represented by the zero iso-0 contour of a scalar level-set function $\varphi \in \mathcal{W}_p$ separating the domain into two non-overlapping regions such that

$$\begin{aligned} \tilde{\Gamma} \approx \Gamma &:= \{\mathbf{x} \in \Omega : \varphi(\mathbf{x}) = 0\} \,, \\ \Omega_1 &:= \{\mathbf{x} \in \Omega : \varphi(\mathbf{x}) < 0\} \,, \\ \Omega_2 &:= \{\mathbf{x} \in \Omega : \varphi(\mathbf{x}) > 0\} \,. \end{aligned}$$

From this definition, the set of elements intersected by the interface is identified by

$$\mathbb{T}^\Gamma := \left\{ \mathbb{T}_m : \mathbb{T}_m \cap \Gamma \neq \emptyset \,, \forall\, \mathbb{T}_m \in \mathbb{T} \right\} \,,$$

as well as the active elements for each phase k as

$$\mathcal{T}^{k,*} := \left\{ T_m : T_m \cap \Omega_k \neq \emptyset \,, \forall\, T_m \in \mathcal{T} \right\} .$$

The set of elements containing only a single phase k can then be defined as

$$\mathcal{T}^{k} := \mathcal{T}^{k,*} \setminus \mathcal{T}^{\Gamma} ,$$

such that

$$\mathcal{T} = \mathcal{T}^{1} \cup \mathcal{T}^{2} \cup \mathcal{T}^{\Gamma} .$$

Similarly, the sets of internal faces and boundary faces intersected by the interface are defined by

$$\mathcal{E}^{\Gamma} := \left\{ E_e : E_e \cap \Gamma \neq \emptyset \,, \forall\, E_e \in \mathcal{E} \right\} ,$$
$$\mathcal{B}^{\Gamma} := \left\{ B_b : B_b \cap \Gamma \neq \emptyset \,, \forall\, B_b \in \mathcal{B} \right\} ,$$

and the ones contained strictly in a single phase k by

$$\mathcal{E}^{k} := \left\{ E_e : E_e \cap \Omega_k \neq \emptyset \,, \forall\, E_e \in \mathcal{E} \right\} \setminus \mathcal{E}^{\Gamma} ,$$
$$\mathcal{B}^{k} := \left\{ B_b : B_b \cap \Omega_k \neq \emptyset \,, \forall\, B_b \in \mathcal{B} \right\} \setminus \mathcal{B}^{\Gamma} .$$

Finally, the set of interfaces per element is defined by

$$\mathcal{I} := \left\{ I_m : I_m = T_m \cap \Gamma \,, \forall\, T_m \in \mathcal{T}^{\Gamma} \right\} ,$$

with the associated interface normals calculated by:

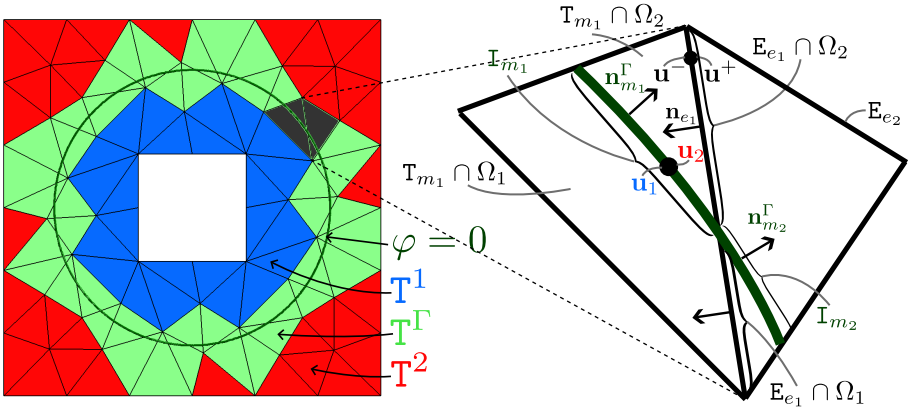
$$\mathbf{n}_{\Gamma,m} = \frac{\nabla \varphi}{\|\nabla \varphi\|} ,$$

where ∇ is the broken gradient operator. The various notations used for the discretization are gathered in Figure 8.3.

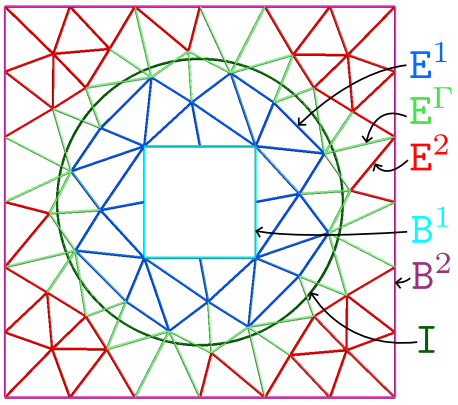
Discontinuities representation: extension of the function space

In order to enable independent solution approximations on two sides of the interface within one cut finite element, hence allowing arbitrary discontinuities to be represented in an unfitted manner, it is necessary to consider independent sets of degrees of freedom on those elements. To do so, the active finite element space for each phase k, $\mathcal{W}_{p,k}$, is first defined as being the standard space in Equation 5.9 defined on the corresponding active elements $\mathcal{T}^{k,*}$,

$$\mathcal{W}_{p,k}(\varphi, \mathcal{T}) := \left\{ \mathbf{w} \in \left[L^2(\Omega) \right]^4 : \mathbf{w}_{|_{T_m}} \circ \sigma_m^{-1} \in \left[\mathcal{P}_p(\hat{T}) \right]^4 , \forall\, T_m \in \mathcal{T}^{k,*} \right\} . \quad [8.1]$$



a / Illustration of the different sets of elements $\mathtt{T}$ (left) and zoom on two cut elements showing the definition of local components (right).



b / Illustration of the different sets of internal faces $\mathtt{E}$, domain boundaries $\mathtt{B}$ and phase interface $\mathtt{I}$.

Figure 8.3 / Geometrical discretization of the continuous domain in Figure 8.2.

The extended DG space is then introduced as

$$\mathcal{W}_p^X(\varphi, \mathcal{T}) = \mathcal{W}_{p,1}(\varphi, \mathcal{T}) \times \mathcal{W}_{p,2}(\varphi, \mathcal{T}) \,. \tag{8.2}$$

This space depends on the level-set field φ as it is composed of two spaces defined on the respective active meshes $\mathcal{T}^{k,*}$ that depend on the position of the level-set iso-0. Moreover, since the cut elements are shared by both active meshes, the degrees of freedom are doubled on those elements. Hence, in the following, $\mathbf{u} \in \mathcal{W}_p^X$ will mean that on cut elements $\mathbf{u} = (\mathbf{u}_1, \mathbf{u}_2)$ with $\mathbf{u}_k \in \mathcal{W}_{p,k}$ such that for a given element $\mathcal{T}_m \in \mathcal{T}^\Gamma$

$$\mathbf{u}_m(\mathbf{x}, t) = \sum_{k=1}^{2} \mathbf{u}_{m,k}(\mathbf{x}, t) = \sum_{k=1}^{2} \sum_{n=1}^{N_n} \phi_{m,n}(\mathbf{x}) \mathbf{U}_{m,n,k}(t) \,, \tag{8.3}$$

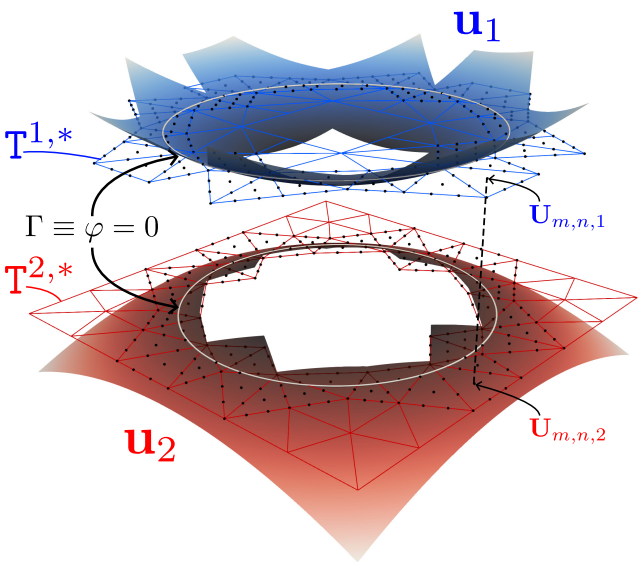
where $\mathbf{U}_{m,n,k} \in \mathbb{R}^{1 \times N_f}$ is the nodal solution unknown in element m corresponding to the nth-degree of freedom associated to phase k. This is illustrated in Figure 8.4a. It is worth mentioning here that the final solution of the coupled interface problem, noted $\mathbf{u}_m^{\mathrm{F}}$, is in fact given by the sum of the solutions in each phase restricted to their corresponding subdomains

$$\begin{aligned} \mathbf{u}_m^{\mathrm{F}}(\mathbf{x}, t) &= \sum_{k=1}^{2} \mathbf{u}_{m,k}(\mathbf{x}, t)\Big|_{\Omega_k} \,, \\ &= \mathbf{u}_{m,1}\, \mathrm{H}(-\varphi) + \mathbf{u}_{m,2}\, \mathrm{H}(\varphi) \,, \end{aligned}$$

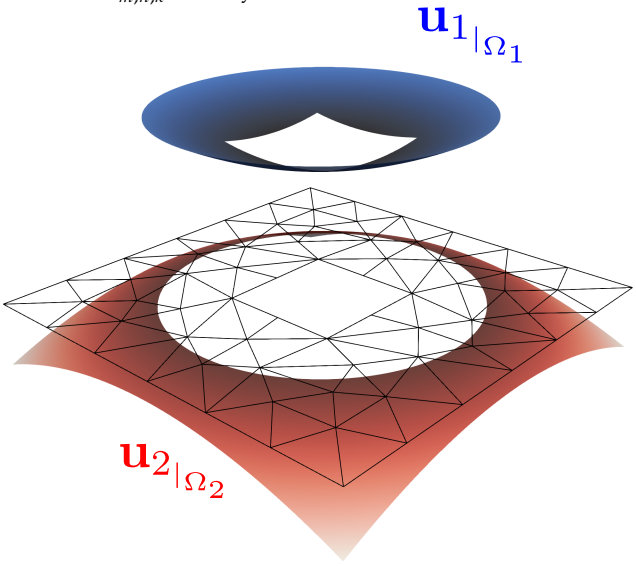
with the Heaviside function $\mathrm{H}(\varphi)$ truncating the solution at the level-set iso-0, as shown in Figure 8.4b. From an implementation point of view however, the extended function space on the cut elements does not contain truncated functions on either side of the interface. The only modification compared to non-cut elements is thus the extension of the standard DG space by the addition of another decoupled one, without altering the reference shape functions spanning the original spaces. The cut-off at the interface is realized during the assembly of the variational form by integrating on the respective subdomains. To evaluate those integrals, $\mathbf{u}_k(\mathbf{x})$ only needs to be computed at locations $\mathbf{x} \in \Omega_k$, so that the solution is never evaluated outside its corresponding subdomain. For visualization purposes, the resulting solutions in each phase can then be truncated at the interface in a post-processing step to obtain $\mathbf{u}^{\mathrm{F}}$.

8.2.2 Modifications of the discrete variational form

Applying the same 5-steps procedure described in Section 5.2.2 to the strong formulation, Equation 9.4, the non-linearly coupled weak form to be solved



a / The interface Γ is located by the zero iso-contour of a level-set function φ. The sets of active elements for phase k are $\mathcal{T}^{k,*}$. The nth nodal DOF (identified by the black dots) in element m for phase k is noted $\mathbf{U}_{m,n,k}$ and they are doubled on cut elements.



b / The restriction of the solutions in their respective domain Ω_k, $\mathbf{u}_{k_{|\Omega_k}}$, yields the sharp interface representation.

Figure 8.4 / Illustration of the way discontinuous fields are represented in XDGM based on the example of Figure 8.3.

reads: find $\mathbf{u} \in \mathcal{U}_p^X$ such that

$$\begin{aligned}\mathcal{N}(\mathbf{u},\mathbf{w}) = \mathcal{N}^1(\mathbf{u}_1,\mathbf{w}) + \mathcal{N}^2(\mathbf{u}_2,\mathbf{w}) \\ + \hat{\mathcal{N}}^\Gamma(\mathbf{u}_1,\mathbf{u}_2,\mathbf{w}) = 0 \quad \forall \mathbf{w} \in \mathcal{U}_p^X ,\end{aligned} \tag{8.4}$$

with the contributions from the bulk of each phase k unchanged compared to the classical DGM, i.e.

$$\begin{aligned}\mathcal{N}^k(\mathbf{u}_k,\mathbf{w}) \equiv & \sum_{m\in\mathbb{T}^k} \mathcal{T}_{m,k}(\mathbf{u}_{m,k},\mathbf{w}_{m,k}) \\ & + \sum_{m\in\mathbb{T}^k} \mathcal{V}_{m,k}(\mathbf{u}_{m,k},\mathbf{w}_{m,k}) \\ & + \sum_{e\in\mathbb{E}^k} \mathcal{E}_{e,k}(\mathbf{u}^+_{m(e),k},\mathbf{u}^-_{m(e),k},\mathbf{w}^+_{m(e),k},\mathbf{w}^-_{m(e),k}) \\ & + \sum_{b\in\mathbb{B}^k} \mathcal{B}_{b,k}(\mathbf{u}^+_{m(b),k},\mathbf{w}^+_{m(b),k}) ,\end{aligned} \tag{8.5}$$

and the contributions from the cut elements, cut faces, cut boundaries and the interface given by

$$\begin{aligned}\hat{\mathcal{N}}^\Gamma(\mathbf{u}_1,\mathbf{u}_2,\mathbf{w}) \equiv \sum_{k\in\{1,2\}} \{ & \sum_{m\in\mathbb{T}^\Gamma} \mathcal{T}_{m,k}(\mathbf{u}_{m,k},\mathbf{w}_{m,k}) \\ & + \sum_{m\in\mathbb{T}^\Gamma} \mathcal{V}_{m,k}(\mathbf{u}_{m,k},\mathbf{w}_{m,k}) \\ & + \sum_{e\in\mathbb{E}^\Gamma} \mathcal{E}_{e,k}(\mathbf{u}^+_{m(e),k},\mathbf{u}^-_{m(e),k},\mathbf{w}^+_{m(e),k},\mathbf{w}^-_{m(e),k}) \\ & + \sum_{b\in\mathbb{B}^\Gamma} \mathcal{B}_{b,k}(\mathbf{u}^+_{m(b),k},\mathbf{w}^+_{m(b),k}) \\ & + \sum_{m\in\mathbb{I}} \hat{\mathcal{I}}_{m,k}(\mathbf{u}_{m,1},\mathbf{u}_{m,2},\mathbf{w}_{m,k}) \} ,\end{aligned} \tag{8.6}$$

with the test functions $\mathbf{w}$ taken from the same space as the trial functions $\mathbf{u}$. A few comments are in order at this stage:

- *Sum over two phases*: the contributions from the cut elements to the global weak form involve a sum over the two phases because, as explained in Section 8.2.1, the degrees of freedom are doubled on the cut entities.
- *Change of integration domains*: we have removed the $\hat{\bullet}$ notation from $\mathcal{N}^k$, $\mathcal{E}$ and $\mathcal{B}$ because the specification of the numerical flux functions in those terms is the same as in the bulk, i.e. Equation 5.28 and Equation 5.31 for $\mathcal{E}$, and Equation 5.32 and Equation 5.33 for $\mathcal{B}$. The assembly of the temporal, volume, face and boundary contributions, $\mathcal{T}$, $\mathcal{V}$, $\mathcal{E}$, $\mathcal{B}$ respectively, in the cut and non-cut elements, faces or boundaries differs only by the

integration domain:

$$\begin{aligned}
\mathcal{T}_{m,k} &: \int_{\mathtt{T}_m} \to \int_{\mathtt{T}_m \cap \Omega_k}, \\
\mathcal{V}_{m,k} &: \int_{\mathtt{T}_m} \to \int_{\mathtt{T}_m \cap \Omega_k}, \\
\mathcal{E}_{e,k} &: \int_{\mathtt{E}_e} \to \int_{\mathtt{E}_e \cap \Omega_k}, \\
\mathcal{B}_{b,k} &: \int_{\mathtt{B}_b} \to \int_{\mathtt{B}_b \cap \Omega_k}.
\end{aligned} \tag{8.7}$$

Let us remark that these modifications in the integration domains concern in practice only the cut entities (since e.g. on a non-cut element $\mathtt{T}_m \cap \Omega_k = \mathtt{T}_m$).

- *Special quadrature techniques*: on non-cut entities, classical Gaussian quadrature rules can be applied to evaluate the integrals over $\mathtt{T}_m$, $\mathtt{E}_e$ and $\mathtt{B}_b$. On the cut faces and boundaries, the one-dimensional integrals (we restrict our attention to two dimensions here) on $\mathtt{E}_e \cap \Omega_k$ and $\mathtt{B}_b \cap \Omega_k$, respectively, can easily be recast as classical one-dimensional Gaussian quadratures on reference segments after having computed the intersection between the level-set iso-0 and the segment. On the other hand, dedicated quadrature techniques are required to evaluate the integrals over the implicitly-defined two-dimensional surfaces $\mathtt{T}_m \cap \Omega_k$. This is covered in Section 8.2.4.
- *Special treatment of temporal term*: concerning the temporal term, we can already realize at this point that the temporal discretization of the term $\partial \mathbf{u}_m / \partial t$ in $\mathcal{T}_{m,k}$ will require special consideration because of the time dependence of the integration domain $\mathtt{T}_m \cap \Omega_k$. This is treated in Section 10.2. It is worth mentioning that because the solutions in cut and non-cut elements may be combined over several time levels in case where the cut topology changes, the separated loop on cut and non-cut elements suggested by the equations above do not reflect the actual implementation. The temporal residual of the XDG form is assembled by looping on all the elements at the same time, contrary to the other contributions.
- *Addition of interface term*: finally, compared to the standard DG form, we highlight the apparition of the interface term $\hat{\mathcal{J}}$ in the XDG discretization on the elements crossed by the level-set iso-0. Its expression is given by

$$\begin{aligned}
&\hat{\mathcal{J}}_{m,k}(\mathbf{u}_{m,1}, \mathbf{u}_{m,2}, \mathbf{w}_{m,k}) \\
&\qquad = \int_{\mathtt{I}_m} \Big[\Big(\hat{\underline{\mathbf{F}}}^c_{\mathtt{I},k}(\mathbf{u}_{m,1}, \mathbf{u}_{m,2}) \\
&\qquad\qquad - \hat{\underline{\mathbf{F}}}^d_{\mathtt{I},k}(\mathbf{u}_{m,1}, \nabla \mathbf{u}_{m,1}, \mathbf{u}_{m,2}, \nabla \mathbf{u}_{m,2}) \Big) \cdot \mathbf{n}_{\Gamma,m,k} \Big] \cdot \mathbf{w}_{m,k} \, \mathrm{d}s \\
&\qquad + \int_{\mathtt{I}_m} (\hat{\mathbf{u}}_{\mathtt{I},k}(\mathbf{u}_{m,1}, \mathbf{u}_{m,2}) - \mathbf{u}_{m,k}) \cdot \left[\mathbf{n}_{\Gamma,m,k} \cdot \underline{\mathbf{G}}^T_k(\mathbf{u}_{m,k}) \nabla \mathbf{w}_{m,k} \right] \mathrm{d}s \, .
\end{aligned} \tag{8.8}$$

In the same way as numerical fluxes across faces couple the solution between elements inside a given phase, the coupling at the interface between the phases is achieved by the addition of the terms $\hat{\mathcal{J}}$. Those terms include

numerical fluxes whose role is to impose the interface jump conditions. The specification of those fluxes (which are let unspecified at this stage, hence the use of the same notation $\hat{\bullet}$) is explained in Section 9.2. The numerical integration over the interface I_m is also realized by dedicated high-order quadrature rules discussed in Section 8.2.4.

The development of all the terms present in the XDG formulation is done in Section 9.3.1 and Section 9.3.3.

8.2.3 Enforcement of continuous level-set iso-0 surface

As a result of the advection via a DG method, discontinuities in the level-set iso-contours will inevitably appear. In particular, the discontinuity of the zero iso-contour which identifies the position of the interface for the XDG discretization leads to three possible scenarios:

- *Keep the discontinuities*: the first option is to allow discontinuities of the phase interface across element faces and just leave the gaps as they are. R. Saye (2017a) notes however that, although the magnitude of those jumps should be small because proportional to the spatial discretization error, their presence might compromise the high-order accuracy of the solver.
- *Consider the discontinuities as new interfaces*: the second possible remedy for bridging the gaps proposed by R. Saye (2017a) is to introduce additional small iso-0 faces aligned with the boundaries of the cells. This would however lead to an overly complicated and intricate procedure, especially in three dimensions.
- *Remove the discontinuities*: the last approach, which is the one adopted in general, is to remove the discontinuities in order to provide a well-defined implicit description of each phase and the interface.

The last approach is the one selected in this work and several techniques were implemented to obtain a continuous level-set field from the original DG one. Those are described in Section 6.6.2. We should emphasize again here that the continuous level-set function does not replace the original level-set field. It is solely used to define the quadrature rules and identify the cut entities for the XDG method. It is computed from the original level-set field at each time step, which is left unaltered.

8.2.4 Integration over implicitly-defined surfaces

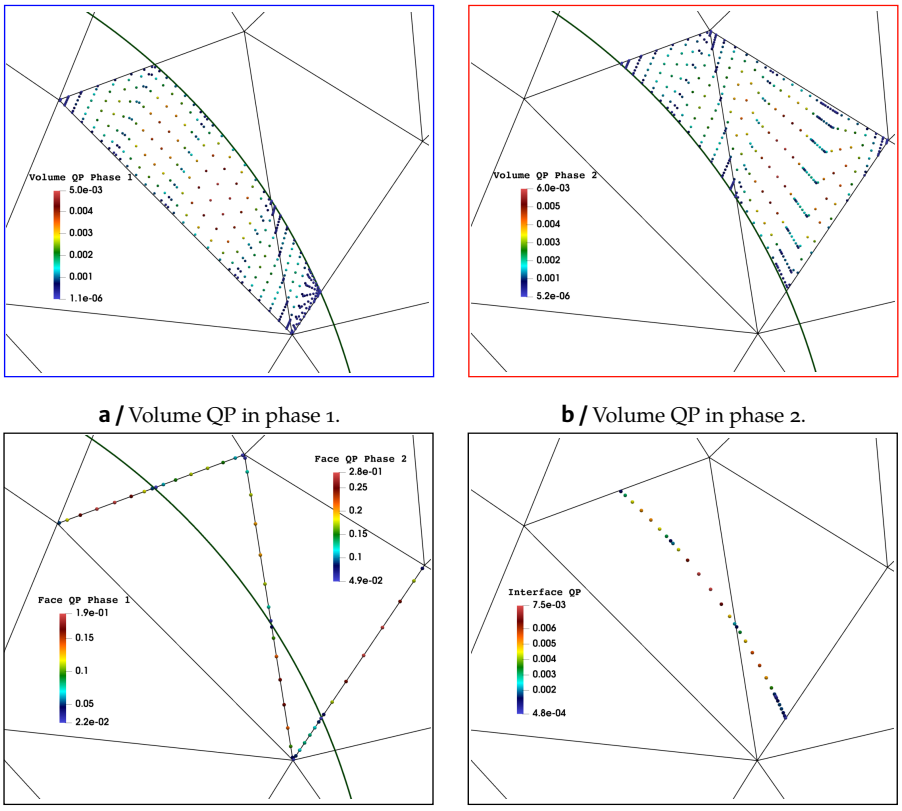
The capacity of the XDGM of reaching high-order accuracy is directly linked to the accuracy with which the integrals in Equation 8.7 and Equation 8.8 can be computed. Contrary to non-cut elements on which standard quadrature

rules can be applied, for a cut element the integrals over the volume and the iso-0 need to be computed over surfaces which are defined implicitly via a fixed iso-surface of a smooth scalar function. A special procedure for numerical integration is therefore requested which will build a quadrature for each cut element, i.e. a set of quadrature nodes and weights. Three different approaches have been included in the solver:

- *HMF*: a hierarchical moment-fitting (HMF) method proposed in B. Müller et al. (2013) and implemented into the solver by Pierre Schrooyen (P. Schrooyen et al. (2017)).
- `Algoim`: a high-order quadrature method for domains implicitly-defined by multivariate polynomials developed by Robert I. Saye (2022), included through the external library `Algoim` (R. Saye (n.d.)). This method is an extension of the prior work of the same author, R. I. Saye (2015), that is based on recasting the geometry as the graph of an implicitly defined height function relying on the implicit function theorem.
- `PHG`: high-order quadrature techniques for triangles and rectangles, through the interfacing with the Parallel Hierarchical Grid (`PHG`) toolbox (T. Cui et al. (n.d.)). The method presented in Tao Cui et al. (2020), which can be seen as an extension to simplices of the method proposed in R. I. Saye (2015), is also based on the concept of height function that allows a proper choice of integration directions along which 1-D Gaussian quadrature rule can be applied.

While the HMF method uses quadrature points on the entire element (for both $\int_{\mathrm{T}_m \cap \Omega_k} \mathrm{d}V$ and $\int_{\mathrm{I}_m} \mathrm{d}S$), the methods in `Algoim` and `PHG` define them strictly inside their respective phase (for $\int_{\mathrm{T}_m \cap \Omega_k} \mathrm{d}V$) or on the zero iso-contour of the level-set (for $\int_{\mathrm{I}_m} \mathrm{d}S$). An example of volume quadrature points generated by `PHG` is shown in Figure 8.5. An intuitive explanation of the design of quadrature rules based on the recasting of the zero iso-contour as the graph of an implicit height function can be found in L. Beck and Kummer (2023). For two-phase flows, and in particular for compressible flows, evaluating the field variables at positions strictly inside their respective phase is preferable to avoid extrapolation of thermodynamic quantities that could lead to non-physical values. If not specified otherwise, the `PHG` library is used in this work because it seamlessly provides quadrature rules for both quad and simplex elements.

All the integrals present in the weak form will be evaluated by building the quadrature on a reference element or face. In other words, what is provided as inputs to the quadrature generator are the reference element and the nodal values of the level-set interpolant. Similarly to what was explained in Section 5.2.4, the outputs are a set of N_Q quadrature nodes $\boldsymbol{\xi}^Q = \{\boldsymbol{\xi}_1^Q, \dots, \boldsymbol{\xi}_q^Q, \dots, \boldsymbol{\xi}_{N_Q}^Q\}$ (expressed in parametric coordinates) and associated weights $\boldsymbol{\omega} = \{\omega_1, \dots, \omega_q, \dots, \omega_{N_Q}\}$ to perform the integration on the reference element. To be able to use them to evaluate the integrals in the weak form, a rescaling is needed. For the one-dimensional integrals on faces and boundaries, a simple rescaling based on the coordinates of the segment end points allows to convert the integrals in the



a / Volume QP in phase 1.

b / Volume QP in phase 2.

c / Element faces QP for both phases.

d / Phase interface QP (common for both phases).

Figure 8.5 / Positions of the quadrature points colored by their weights on the two selected cut elements of Figure 8.3. Quadrature generated by PHG library (with 14th 1D Gaussian quadrature order).

parametric to the physical space. For the integrals on the elements surfaces, the rescaling involves the determinant of the Jacobian of the mapping between parametric and physical coordinates, $\underline{\boldsymbol{J}}^{\sigma}$:

$$\int_{\mathbb{T}_m \cap \Omega_k} \mathbf{f}(\mathbf{x})\, \mathrm{d}\mathbf{x} = \sum_{q=1}^{N_Q^{\mathbb{T}}} \omega_q^{\mathbb{T}}\, \mathbf{f}(\boldsymbol{\xi}_q^{Q,\mathbb{T}})\, \det(\underline{\boldsymbol{J}}^{\sigma}(\boldsymbol{\xi}_q^{Q,\mathbb{T}}))\,, \tag{8.9}$$

with $\underline{\boldsymbol{J}}^{\sigma}$ defined in Equation 5.7. Finally, for the rescaling of the infinitesimal line element $\mathrm{d}s$ in the integrals over the interface which crosses across elements, a scaling is needed in addition to the one provided by the determinant of $\underline{\boldsymbol{J}}^{\sigma}$. Indeed, in order to correctly take into account the curvature of the one-dimensional interface inside the two-dimensional element, it is essential to scale by the ratio between the norms of the physical and parametric level-set gradients, as explained in B. Müller et al. (2013):

$$\int_{\mathbb{I}_m} \mathbf{f}(\mathbf{x})\, \mathrm{d}s = \sum_{i=1}^{N_Q^{\mathbb{I}}} \omega_q^{\mathbb{I}}\, \mathbf{f}(\boldsymbol{\xi}_q^{Q,\mathbb{I}})\, \det(\underline{\boldsymbol{J}}^{\sigma}(\boldsymbol{\xi}_q^{Q,\mathbb{I}}))\, \frac{\|\nabla_{\mathbf{x}}\varphi\|}{\|\nabla_{\boldsymbol{\xi}}\varphi\|}(\boldsymbol{\xi}_q^{Q,\mathbb{I}})\,, \tag{8.10}$$

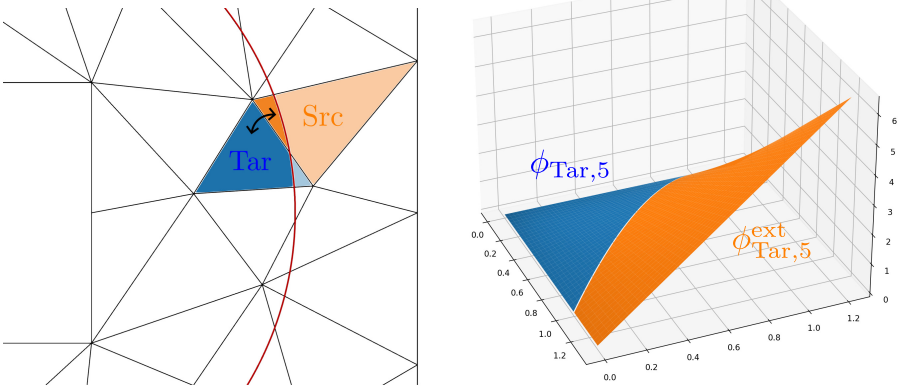
where $\nabla_{\mathbf{x}}\varphi$ and $\nabla_{\boldsymbol{\xi}}\varphi$ are the physical and parametric gradients of the level-set function, respectively.

8.2.5 Cell agglomeration: treatment of small cut elements

Since the interface is free to cross the elements of the underlying mesh in any manner, there exist pathological intersection scenarios in which one of the phase occupies only a small volume fraction of the cut element. This results in conditioning issues for the resulting system, intersection-dependent instabilities and suboptimal error estimates, as noted by Benedikt Schott (2017, Sec. 3.4). In the XDGM, a cell agglomeration technique is applied to overcome those issues. It is based on the work of Kummer (2017) and Muller et al. (2016) and its implementation in the present solver was first presented in P. Schrooyen et al. (2017) and P. Schrooyen et al. (2018). The principle of the approach is to agglomerate a non-problematic neighboring element to the one identified as critical. The threshold on the ratio between the cut volume fraction and the full cell volume from which it is decided if an element has to be agglomerated or not is noted δ_{agg} such that:

$$\begin{aligned} \text{If} \quad \mathrm{fr}_k(T_m) := \frac{|\mathbb{T}_m \cap \Omega_k|}{|\mathbb{T}_m|} &= \frac{\text{“Volume fraction of phase } k\text{”}}{\text{“Volume of the cell”}} < \delta_{\text{agg}}\,, \\ &\Rightarrow \text{ agglomeration needed.} \end{aligned} \tag{8.11}$$

Based on this criterion, a set of agglomerated “target”-“source” pairs cells is formed: each cell such that $\mathrm{fr}_k(\mathbb{T}_m) < \delta_{\text{agg}}$, called a source cell $\mathbb{T}_{m1}^{\text{Src}}$, is associated



a / Target-Source pair creation in phase 1 based on the volume fraction (dark orange region) below a fixed threshold.

b / Example of the smooth extension $\phi^{\text{ext}}_{\text{Tar},5}$ of the 5th quadratic target Lagrange shape function $\phi_{\text{Tar},5}$ onto the associated source.

Figure 8.6 / Illustration of the cell agglomeration procedure on a triangular mesh for a chosen target-source pair in phase 1 taken from the situation in Figure 8.4.

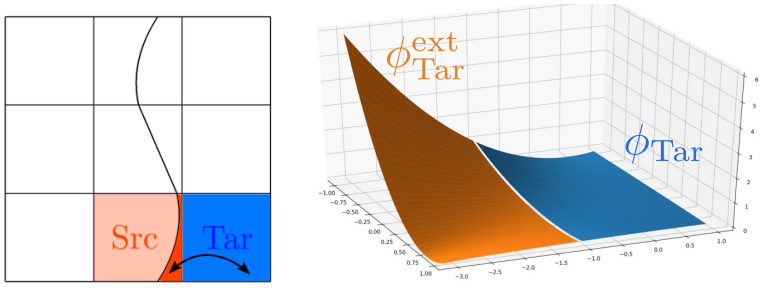
to the edge-neighbor cell containing the largest volume fraction of the considered phase, called the target cell $\mathrm{T}^{\text{Tar}}_{m2}$, with $m1$ and $m2$ being two distinct mesh element indices. This agglomeration set per phase k is written as

$$\begin{aligned}\text{Agg}_k := \big\{(\mathrm{T}^{\text{Src}}_{m1}, \mathrm{T}^{\text{Tar}}_{m2}) :\ & \partial\mathrm{T}^{\text{Src}}_{m1} \cap \partial\mathrm{T}^{\text{Tar}}_{m2} \neq \emptyset\,,\ \text{fr}_k(\mathrm{T}^{\text{Src}}_{m1}) < \delta_{\text{agg}}\,,\\ & \text{fr}_k(\mathrm{T}^{\text{Tar}}_{m2}) \text{ is max } \forall\, \mathrm{T}^{\text{Src}}_{m1}, \mathrm{T}^{\text{Tar}}_{m2} \in \mathcal{T}^{k,*}\big\}\ .\end{aligned} \tag{8.12}$$

Because the conditioning issues are caused by the small physical support of the basis functions associated to the source cells, those are removed from the global function space and the shape functions from the target cells are extended beyond their original support into the source ones, as illustrated in Figure 8.6 and Figure 8.7. This is done in practice by applying the inverse parametric-to-physical mapping function, σ^{-1}, of the target element to the physical coordinates of the interpolation points in the source element, as presented in Equation 6.53. This modifies the active finite element space for each phase k defined in Equation 8.1 in such a way that

$$\begin{aligned}\mathcal{W}^{\text{agg}}_{p,k}(\varphi, \mathcal{T}, \text{Agg}_k) := \Big\{ & \mathbf{w} \in \big[L^2(\Omega)\big]^4 :\\ & \forall\, \mathrm{T}^{\text{Src}}_{m1}, \mathrm{T}^{\text{Tar}}_{m2} \in \mathcal{T}^{k,*} \text{ s.t. } (\mathrm{T}^{\text{Src}}_{m1}, \mathrm{T}^{\text{Tar}}_{m2}) \in \text{Agg}_k : \mathbf{w}_{|_{\mathrm{T}^{\text{Src}}_{m1} \cup \mathrm{T}^{\text{Tar}}_{m2}}} \circ \sigma^{-1,\text{Tar}}_{m2} \in \big[\mathcal{P}_p(\hat{\mathrm{T}})\big]^4\,,\\ & \forall\, \mathrm{T}_m \in \mathcal{T}^{k,*} \text{ s.t. } \mathrm{T}_m \notin \text{Agg}_k : \mathbf{w}_{|_{\mathrm{T}_m}} \circ \sigma^{-1}_m \in \big[\mathcal{P}_p(\hat{\mathrm{T}})\big]^4 \Big\}\ ,\end{aligned} \tag{8.13}$$

where it can be noted that the mapping of the target cell, σ^{Tar}_{m2}, is used to map the function $\mathbf{w}_{|_{\mathrm{T}^{\text{Src}}_{m1} \cup \mathrm{T}^{\text{Tar}}_{m2}}}$ restricted to the aggregation between the source and target



a / Target-Source pair creation based on the volume fraction (dark orange region) below a fixed threshold.

b / Example of the smooth extension $\phi^{\text{ext}}_{\text{Tar}}$ of one of the quadratic target Lagrange shape functions ϕ_{Tar} onto the associated source.

Figure 8.7 / Illustration of the cell agglomeration procedure on a quadrilateral mesh.

cells to the reference element. The corresponding extended agglomerated functional space from which the test and trial functions are sampled is then given by

$$\mathcal{W}_p^{X,\text{agg}}(\varphi, \mathbb{T}, \text{Agg}) = \mathcal{W}_{p,1}^{\text{agg}}(\varphi, \mathbb{T}, \text{Agg}_1) \times \mathcal{W}_{p,2}^{\text{agg}}(\varphi, \mathbb{T}, \text{Agg}_2) \,. \tag{8.14}$$

This agglomerated space is a subspace of the original XDG space, i.e. $\mathcal{W}_p^{X,\text{agg}} \subseteq \mathcal{W}_p^X$. When agglomeration is active (which should be almost always the case), the solution $\mathbf{u}$ and the test function $\mathbf{w}$ are taken from $\mathcal{W}_p^{X,\text{agg}}$ instead of $\mathcal{W}_p^X$ in the weak formulation Equation 8.4. Two remarks are given to close this section:

- *Preservation of the topology*: as already mentioned in Section 8.1.4, in practice the source cells are not explicitly removed from the discretization, hence the original data structure implemented for the XDG discretization without agglomeration is kept unchanged. Instead, coupling matrices containing the projections of the extended target basis onto the source cells are assembled and all the agglomerated operations can be performed by cell-local matrix multiplications. A step-by-step description of the application of the cell-agglomeration procedure to the assembly of the XDG residual is provided in Section 8.3.1.
- *Agglomeration chain*: during the building of the agglomeration map, it is possible that none of the neighboring cells of a source cell are suited to become the corresponding target cell. This will happen if all the adjacent cells are already marked as "source" and/or if no appropriate cell containing the same phase as the source cell can be found. As noted by Smuda and Kummer (2021), in this case a chain of agglomerated cells must be formed. In this work, we avoid the construction of such chains for ease of implementation and because it is expected that the occurrence of chains is limited and that they will be short. Instead, we allow the extension

of target cells basis functions into source cells which do not share any common face. It is therefore possible that a source element does not share any edge with its target, such that the definition of the agglomerated map, Equation 8.12, is not exactly respected. An example of such a scenario is given Figure 8.8.

To close this section, it is useful to recapitulate how the cell agglomeration procedure allows to tackle all at once the important issues associated with the presence of small or sliver cut elements, already described at the beginning of Section 8.1.3:

- *Conditioning and accuracy*: To tame the condition number of the system, the basis functions of the target element are considered on the whole agglomerated element while the ones on the source element are not included, thus effectively removing the issues associated to the small truncated support. The control of the conditioning ensures that the solution of the system will not be affected by unacceptable round-off errors or convergence issues. Since we are talking about conditioning, it is already useful to note here that another source of system ill-conditioning in our case is the stiffness of the physical models considered, especially in the condensed phase. As explained in Section 10.1.2, because of the absence of appropriate preconditioner in our code, we favor the use of a direct solver over an iterative one (and this choice is completely decoupled from the presence of small cut cells).
- *Stability*: To ensure coercivity of the discretized form while keeping a bounded penalty factor, the latter is expressed based on the agglomerated mesh information, as proposed by Kummer (2017). This is further discussed in Section 9.2.3 and illustrated by Figure 9.2.

8.3 Implementation aspects of the XDG framework

8.3.1 Assembly of agglomerated residuals and jacobians

The cell agglomeration implemented in the solver comprises several steps which are described below and illustrated by various sketches.

Flagging of target–source pairs

The first step is to identify the elements that need to be agglomerated, i.e. the "sources" noted with the $\bullet^{\mathrm{Src}}$ superscript, and the candidates from which the basis will be extended outside their original support, i.e. the "targets" with

the $\bullet^{\text{Tar}}$ superscript. As already explained in Section 8.2.5, the goal of the agglomeration procedure is to overcome the critical issues posed by the presence of small cut cells. Actually, we take advantage of the procedure for an equally fundamental task, which is to ensure a constant cut topology between different time levels, as done e.g. Smuda and Kummer (2021). This is crucial to be able to obtain a consistent time discretization for moving interface problems. This aspect is covered in details in Section 10.2 but is already mentioned at this stage because the flagging of the target-source pairs needs to take it into account.
In order to simplify the notations, we will refer to the part of element $\mathbb{T}_m$ contained in phase k as $\mathbb{T}_{m,k}$ i.e.

$$\mathbb{T}_{m,k} \triangleq \mathbb{T}_m \cap \Omega_k \,.$$

This can represent a part of the original element if it is crossed by the interface or the whole element if it is not cut. The global agglomeration map we want to construct is subdivided per phase and per time step. Another important remark to make here is about the choice of the time integration strategy selected, namely "splitting" or "moving interface". For reasons which are elucidated in Section 10.2.2, the former approach only needs a map defined over the current time level t while the latter requires a map defined over several time levels:

$$\begin{aligned} \text{Agg} &= \left\{\text{Agg}_1^t,\ \text{Agg}_2^t\right\} \quad \text{if splitting}\,, \\ \text{Agg} &= \left\{\left\{\text{Agg}_1^{t-N_t},\ \text{Agg}_2^{t-N_t}\right\}, \dots, \left\{\text{Agg}_1^t,\ \text{Agg}_2^t\right\}\right\} \quad \text{if moving interface}\,, \end{aligned} \tag{8.15}$$

where N_t is the number of time steps required by the time-stepping scheme. For instance, $N_t = 1$ for the first-order Euler scheme. At the time level t for phase k, the agglomeration map is composed by a list of target element indices, each one linked to the list of their associated source elements:

$$\text{Agg}_k^t = \left\{\dots, \mathbb{T}_{m1,k}^{\text{Tar},t} : \left[\mathbb{T}_{m2,k}^{\text{Src},t}\right], \dots, \mathbb{T}_{m3,k}^{\text{Tar},t} : \left[\mathbb{T}_{m4,k}^{\text{Src},t}, \mathbb{T}_{m5,k}^{\text{Src},t}\right], \dots\right\} .$$

Let us note that we also use the $\bullet^t$ notation to make clear at which time step we are considering the elements since those can vary depending on the interface position. In addition to this agglomeration map Agg, a second useful map, called the source map SrcM, is also assembled, directly from Agg. It is composed by subsets referencing per time level and per phase the source elements and their associated targets:

$$\text{SrcM}_k^t = \left\{\dots, \mathbb{T}_{m1,k}^{\text{Src},t} : \mathbb{T}_{m2,k}^{\text{Tar},t}, \dots, \mathbb{T}_{m3,k}^{\text{Src},t} : \mathbb{T}_{m4,k}^{\text{Tar},t}, \dots\right\} .$$

A given cell at time t, $\mathbb{T}_{m,k}^t$, is marked as a source cell if at least one of the following three criteria is met:

1. *Volume fraction*: this is the condition of Equation 8.11 which identifies the element as being a source if its volume fraction, fr_k, at time t is below a certain threshold.

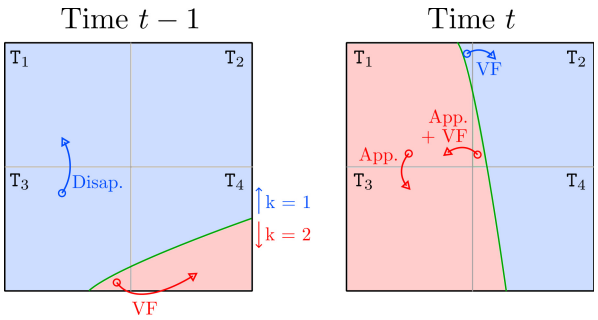


Figure 8.8 / Illustrative example of the agglomeration flagging procedure. Creation of target-source pair based on volume fraction (VF), appearing (App.) or disappearing (Disap.) elements. The arrows go from the source to the target elements.

2. *Appearing element*: for both the splitting and moving interface time integration approaches, we need to mark cells which appear at the beginning of the new time step, i.e. in which phase k was not present at the previous time $t-1$:

$$|\mathbb{T}_{m,k}^{t-1}| = 0 \,\&\, |\mathbb{T}_{m,k}^{t}| > 0\,.$$

   If more than two time levels are considered because a higher time stepping scheme is used, the flagging needs to be performed over all the time levels for both approaches[2]. The maps of all the levels are combined at the end: into a single one defined at the current time for the splitting approach, and into separated maps for each level for the moving interface strategy.
3. *Disappearing element*: this condition is only applied for the moving interface approach which requires cells that disappear from the current topology to be flagged as source:

$$|\mathbb{T}_{m,k}^{t-1}| > 0 \,\&\, |\mathbb{T}_{m,k}^{t}| = 0\,.$$

Additional steps after the flagging based on the above criteria has been realized include the removal of agglomeration chains and for moving interface, the combination of the agglomeration maps over the different time levels. The flagging and the additional steps are illustrated on the example below.

[2]Indeed, let's take the example where an element at time t would be flagged as a target because one of its neighboring elements appeared between $t-1$ and t. If the time-stepping scheme requires combining three time levels, it is not guaranteed that this target element will exist for the considered phase at $t-2$. Hence, it must be changed by another element which is valid over all the time levels.

Example 8.1: Flagging procedure in the cell-agglomeration technique

The example is based on Figure 8.8. We assume that the moving interface approach with an Euler time-stepping scheme is used here such that disappearing elements need to be taken into account.

1. *Flagging based on the criteria*: the first step is to loop over the time levels, over each phase and over each element, identifying the source elements based on the three criteria listed above and their associated targets. The result for this illustrative situation is given in the table below.

	Time $t-1$	
	$k=1$	$k=2$
VF	/	$\mathrm{T}_{4,2}^{\mathrm{Tar},t-1} : \left[\mathrm{T}_{3,2}^{\mathrm{Src},t-1}\right]$
App.	/	/
Disap.	$\mathrm{T}_{1,1}^{\mathrm{Tar},t-1} : \left[\mathrm{T}_{3,1}^{\mathrm{Src},t-1}\right]$	/
	Time t	
	$k=1$	$k=2$
VF	$\mathrm{T}_{2,1}^{\mathrm{Tar},t} : \left[\mathrm{T}_{1,1}^{\mathrm{Src},t}\right]$	$\mathrm{T}_{1,2}^{\mathrm{Tar},t} : \left[\mathrm{T}_{2,2}^{\mathrm{Src},t}\right]$
App.	/	$\mathrm{T}_{1,2}^{\mathrm{Tar},t} : \left[\mathrm{T}_{2,2}^{\mathrm{Src},t}\right]$ $\mathrm{T}_{3,2}^{\mathrm{Tar},t} : \left[\mathrm{T}_{1,2}^{\mathrm{Src},t}\right]$
Disap.	/	/

2. *Breaking of agglomeration chains*: the second step is to remove, for each time level separately, any potential agglomeration chains. There is one present in this case which is broken as follows:

$$\mathrm{T}_{3,2}^{\mathrm{Tar},t} : \left[\mathrm{T}_{1,2}^{\mathrm{Src},t}\right] \,\&\, \mathrm{T}_{1,2}^{\mathrm{Tar},t} : \left[\mathrm{T}_{2,2}^{\mathrm{Src},t}\right] \Rightarrow \mathrm{T}_{3,2}^{\mathrm{Tar},t} : \left[\mathrm{T}_{1,2}^{\mathrm{Src},t}, \mathrm{T}_{2,2}^{\mathrm{Src},t}\right] .$$

   This illustrates the comment made above about the fact that agglomeration between elements not sharing any common faces is possible.
3. *Redistribution of source elements*: an additional intermediate step consists in trying to redistribute more equitably some of the source elements among the non-problematic elements. Indeed, a target element may be associated to several source elements because it was the best candidate in a certain neigh-

borhood. But it is possible that another nearby element could also become an equally valid target element (containing only a slightly smaller volume fraction than the best identified target). To be able to operate this redistribution, we collect for each source element during the first step the ordered list of all the possible associated target elements. We can then pick for example the second-best target element of the list for a given source element if the first one is already associated to several other source elements. This redistribution contributes to limit the extent of the agglomerated cells. In practice, the user can set the maximum number of source elements per target element and the situation stops if this threshold is exceeded.

4. *Combination over time levels (for moving interface time integration)*: finally, in case the moving interface approach is used, a last step combining the agglomerations maps from the different time levels is performed such that a common topology exists between all the time levels. It is done one level at a time.

   - From t to $t-1$: we add to the agglomeration map already constructed at the first step (see table above) the contribution from time level t:

$$\begin{aligned}
\mathrm{Agg}_1^{t-1} &\leftarrow \mathrm{Ind}^{t\to t-1}(\mathbb{T}_{2,1}^{\mathrm{Tar},t}) : \left[\mathrm{Ind}^{t\to t-1}(\mathbb{T}_{1,1}^{\mathrm{Src},t})\right] \\
&= \mathbb{T}_{2,1}^{\mathrm{Tar},t-1} : \left[\mathbb{T}_{1,1}^{\mathrm{Src},t-1}\right], \\
\mathrm{Agg}_2^{t-1} &\leftarrow \mathrm{Ind}^{t\to t-1}(\mathbb{T}_{3,2}^{\mathrm{Tar},t}) : \left[\mathrm{Ind}^{t\to t-1}(\mathbb{T}_{1,2}^{\mathrm{Src},t}),\ \mathrm{Ind}^{t\to t-1}(\mathbb{T}_{2,2}^{\mathrm{Src},t})\right] \\
&= \mathbb{T}_{3,2}^{\mathrm{Tar},t-1} : [\mathrm{N/D},\ \mathrm{N/D}] = \emptyset .
\end{aligned}$$

     What to pay attention to here is the conversion of the element indices from t to $t-1$, $\mathrm{Ind}^{t\to t-1}$, that needs to be operated such that the combination is possible. Indeed, on a cut element, to identify one or the other phase, distinct element indices are used and this indexation will evolve in time. While for phase 1, the source element from t has an equivalent at $t-1$, it is not the case for phase 2. In this latter case, the agglomeration data from t will not change the ones at $t-1$. After the indices conversion and the addition to the map at $t-1$, the removal of agglomeration chain procedure is applied. In this case, it leads to:

$$\begin{aligned}
&\mathbb{T}_{2,1}^{\mathrm{Tar},t-1} : \left[\mathbb{T}_{1,1}^{\mathrm{Src},t-1}\right] \ \&\ \mathbb{T}_{1,1}^{\mathrm{Tar},t-1} : \left[\mathbb{T}_{3,1}^{\mathrm{Src},t-1}\right] \\
&\qquad \Rightarrow \mathbb{T}_{2,1}^{\mathrm{Tar},t-1} : \left[\mathbb{T}_{1,1}^{\mathrm{Src},t-1},\ \mathbb{T}_{3,1}^{\mathrm{Src},t-1}\right] .
\end{aligned}$$

   - From $t-1$ to t: note that we keep a copy of the unmodified

map at $t-1$ to operate the combination in this second step:

$$\begin{aligned}
\mathrm{Agg}_1^t &\leftarrow \mathrm{Ind}^{t-1\to t}(\mathbb{T}_{1,1}^{\mathrm{Tar},t-1}) : \left[\mathrm{Ind}^{t-1\to t}(\mathbb{T}_{3,1}^{\mathrm{Src},t-1})\right] \\
&= \mathbb{T}_{1,1}^{\mathrm{Tar},t} : [\mathrm{N/D}] = \emptyset\,, \\
\mathrm{Agg}_2^t &\leftarrow \mathrm{Ind}^{t-1\to t}(\mathbb{T}_{4,2}^{\mathrm{Tar},t-1}) : \left[\mathrm{Ind}^{t-1\to t}(\mathbb{T}_{3,2}^{\mathrm{Src},t-1})\right] \\
&= \mathbb{T}_{4,2}^{\mathrm{Tar},t} : \left[\mathbb{T}_{3,2}^{\mathrm{Src},t}\right]\,.
\end{aligned}$$

Again, the potential agglomeration chains are removed after the modifications of the agglomeration maps, which gives:

$$\begin{aligned}
&\mathbb{T}_{4,2}^{\mathrm{Tar},t} : \left[\mathbb{T}_{3,2}^{\mathrm{Src},t}\right] \,\&\, \mathbb{T}_{3,2}^{\mathrm{Tar},t} : \left[\mathbb{T}_{1,2}^{\mathrm{Src},t}, \mathbb{T}_{2,2}^{\mathrm{Src},t}\right] \\
&\qquad \Rightarrow \mathbb{T}_{4,2}^{\mathrm{Tar},t} : \left[\mathbb{T}_{1,2}^{\mathrm{Src},t}, \mathbb{T}_{2,2}^{\mathrm{Src},t}, \mathbb{T}_{3,2}^{\mathrm{Src},t}\right]\,.
\end{aligned}$$

At the end, the resulting agglomeration map that will be used for all the subsequent operations is given by

$$\begin{aligned}
\mathrm{Agg} &= \left\{\left\{\mathrm{Agg}_1^{t-1}, \mathrm{Agg}_2^{t-1}\right\}, \left\{\mathrm{Agg}_1^t, \mathrm{Agg}_2^t\right\}\right\}\,, \\
\mathrm{Agg}_1^{t-1} &= \left\{\mathbb{T}_{2,1}^{\mathrm{Tar},t-1} : \left[\mathbb{T}_{1,1}^{\mathrm{Src},t-1}, \mathbb{T}_{3,1}^{\mathrm{Src},t-1}\right]\right\}\,, \\
\mathrm{Agg}_2^{t-1} &= \left\{\mathbb{T}_{4,2}^{\mathrm{Tar},t-1} : \left[\mathbb{T}_{3,2}^{\mathrm{Src},t-1}\right]\right\}\,, \\
\mathrm{Agg}_1^{t} &= \left\{\mathbb{T}_{2,1}^{\mathrm{Tar},t} : \left[\mathbb{T}_{1,1}^{\mathrm{Src},t}\right]\right\}\,, \\
\mathrm{Agg}_2^{t} &= \left\{\mathbb{T}_{4,2}^{\mathrm{Tar},t} : \left[\mathbb{T}_{1,2}^{\mathrm{Src},t}, \mathbb{T}_{2,2}^{\mathrm{Src},t}, \mathbb{T}_{3,2}^{\mathrm{Src},t}\right]\right\}\,.
\end{aligned}$$

A common agglomerated topology over the time levels is indeed ensured because the target elements at both levels are the same.
Agglomeration loop: let us note that if $\mathbb{T}_{4,2}^t$ was identified as a source (because having a volume fraction below the threshold), we would end up with an agglomeration loop between $\mathbb{T}_{3,2}$ and $\mathbb{T}_{4,2}$ and the forced simulation stop. This situation is however very unlikely to occur.

Construction of coupling and agglomerated mass matrices

As already introduced in Section 7.2.3, a coupling matrix for every target-source pair is assembled based on the work of Muller et al. (2016). The idea is to incorporate in this matrix the projection of the extended target basis onto the source cell such that all the agglomeration operations can be performed based on it. Let $\boldsymbol{\phi}_{m1,-}^{\mathrm{Tar}}$ be the shape functions on the target element $\mathbb{T}_{m1}^{\mathrm{Tar}}$, $\boldsymbol{\phi}_{m2,-}^{\mathrm{Src}}$ the shape functions on the associated source element $\mathbb{T}_{m2}^{\mathrm{Src}}$ and $\boldsymbol{\phi}_{m1,-}^{\mathrm{Tar},*}$ the shape functions from the target element extended beyond their original support into the source element. By construction, the coupling matrix between both elements,

$\underline{\boldsymbol{Q}}^{m2\to m1} \in \mathbb{R}^{N_n \times N_n}$ is thus given by

$$\underline{\boldsymbol{Q}}^{m2\to m1} = (\underline{\boldsymbol{M}}^{\mathrm{Src}}_{m2})^{-1} \int_{\mathsf{T}^{\mathrm{Src}}_{m2}} \boldsymbol{\phi}^{\mathrm{Src}}_{m2,-}\, \boldsymbol{\phi}^{\mathrm{Tar},*,T}_{m1,-}\, \mathrm{d}\mathbf{x}\,, \tag{8.16}$$

where the mass matrix of the source element is

$$\underline{\boldsymbol{M}}^{\mathrm{Src}}_{m2} = \int_{\mathsf{T}^{\mathrm{Src}}_{m2}} \boldsymbol{\phi}^{\mathrm{Src}}_{m2}\, \boldsymbol{\phi}^{\mathrm{Src},T}_{m2}\, \mathrm{d}\mathbf{x}\,.$$

Thanks to this matrix, the shape functions in the source element are readily obtained as

$$\boldsymbol{\phi}^{\mathrm{Tar},*}_{m1,-} = \underline{\boldsymbol{Q}}^{m2\to m1,T}\, \boldsymbol{\phi}^{\mathrm{Src}}_{m2,-}\,. \tag{8.17}$$

Note that the extension of the target shape function and the construction of the coupling matrix are done independently of the cut topology of the elements, i.e. the entirety of the elements is considered, even if they are cut. On the other hand, the mass matrix of the agglomerated cell needs to consider the intersection of the interface with the mesh. For the agglomerated cell made up of $\mathsf{T}^{\mathrm{Tar}}_{m1,k}$ and $\mathsf{T}^{\mathrm{Src}}_{m2,k}$, viz.

$$\mathsf{T}^{\mathrm{Agg}}_{m1,k} = \mathsf{T}^{\mathrm{Tar}}_{m1,k} \cup \mathsf{T}^{\mathrm{Src}}_{m2,k}\,,$$

associated to phase k, the agglomerated mass matrix can be computed as

$$\begin{aligned}
\underline{\boldsymbol{M}}^{\mathrm{Agg}}_{m1,k} &= \int_{\mathsf{T}^{\mathrm{Agg}}_{m1,k}} \boldsymbol{\phi}^{\mathrm{Agg}}_{m1,-}\, \boldsymbol{\phi}^{\mathrm{Agg},T}_{m1,-}\, \mathrm{d}\mathbf{x}\,, \\
&= \int_{\mathsf{T}^{\mathrm{Tar}}_{m1,k}} \boldsymbol{\phi}^{\mathrm{Tar}}_{m1,-}\, \boldsymbol{\phi}^{\mathrm{Tar},T}_{m1,-}\, \mathrm{d}\mathbf{x} + \int_{\mathsf{T}^{\mathrm{Src}}_{m2,k}} \underbrace{\boldsymbol{\phi}^{\mathrm{Tar},*}_{m1,-}}_{\underline{\boldsymbol{Q}}^{m2\to m1,T}\, \boldsymbol{\phi}^{\mathrm{Src}}_{m2,-}} \underbrace{\boldsymbol{\phi}^{\mathrm{Tar},*,T}_{m1,-}}_{\boldsymbol{\phi}^{\mathrm{Src},T}_{m2,-}\, \underline{\boldsymbol{Q}}^{m2\to m1}} \mathrm{d}\mathbf{x}\,, \\
&= \underline{\boldsymbol{M}}^{\mathrm{Tar}}_{m1,k} + \underline{\boldsymbol{Q}}^{m2\to m1,T}\, \underline{\boldsymbol{M}}^{\mathrm{Src}}_{m2,k}\, \underline{\boldsymbol{Q}}^{m2\to m1}\,,
\end{aligned}$$

where the mass matrices over the target and source elements are obtained by an integration over the cut volume, if the element is crossed by the interface. This integration is performed with the dedicated quadrature rules described in Section 8.2.4. Note that the quantities of the agglomerated cell will always be identified by the index of the target element, i.e. $m1$ here. Finally, it must be made clear that if a target element possesses more than one associated source, the agglomerated mass matrix will be computed by summing over the source elements,

$$\underline{\boldsymbol{M}}^{\mathrm{Agg}}_{m1,k} = \begin{cases} \underline{\boldsymbol{M}}^{\mathrm{Tar}}_{m1,k} + \sum_{i=2}^{N_{\mathrm{Src}}} \underline{\boldsymbol{Q}}^{mi\to m1,T}\, \underline{\boldsymbol{M}}^{\mathrm{Src}}_{mi,k}\, \underline{\boldsymbol{Q}}^{mi\to m1}\,, & \text{if } \mathsf{T}_{m1,k} \text{ is target}\,, \\ 0\,, & \text{if } \mathsf{T}_{m1,k} \text{ is source}\,, \\ \underline{\boldsymbol{M}}_{m1,k}\,, & \text{otherwise}\,. \end{cases} \tag{8.18}$$

where N_{Src} is the number of source elements for this particular target element.

Injection of agglomerated solution into original mesh

At the beginning of a new time step, we start by flagging the target and source elements and then we assemble the coupling and agglomerated mass matrices. We also need, before starting the assembly of the residual, to inject the agglomerated solution into the non-agglomerated mesh because as we will see just after the spatial residual is assembled from operations on the original mesh. To be able to perform this injection, we first need to determine the agglomerated solution in element $m1$ for phase k,

$$\mathbf{u}^{\text{Agg}}_{m1,k}(\mathbf{x}) = \boldsymbol{\phi}^{\text{Agg},T}_{m1,-}(\mathbf{x})\, \underline{\boldsymbol{U}}^{\text{Agg}}_{m1,-,k,-}\,,$$

where $\underline{\boldsymbol{U}}^{\text{Agg}}_{m1,-,k,-} \in \mathbb{R}^{N_n \times N_f}$ are the agglomerated nodal degrees of freedom of phase k on the N_n nodes in the element $\mathbb{T}_{m1}$, with N_f field variables on each node. We obtain those agglomerated DOF by a projection of the original solution $\mathbf{u}_{m1,k} + \mathbf{u}_{m2,k}$ on the agglomerated cell, i.e.

$$\int_{\mathbb{T}^{\text{Agg}}_{m1,k}} \left[\mathbf{u}^{\text{Agg}}_{m1,k} - \left(\mathbf{u}_{m1,k} + \mathbf{u}_{m2,k}\right)\right] \boldsymbol{\phi}^{\text{Agg}}_{m1,-}\, \mathrm{d}\mathbf{x} = 0\,,$$

which leads to

$$\begin{aligned}
\underline{\boldsymbol{M}}^{\text{Agg}}_{m1,k}\, \underline{\boldsymbol{U}}^{\text{Agg}}_{m1,-,k,-} &= \int_{\mathbb{T}^{\text{Tar}}_{m1,k}} \underbrace{\mathbf{u}_{m1,k}}_{\boldsymbol{\phi}^{\text{Tar},T}_{m1,-}\underline{\boldsymbol{U}}_{m1,-,k,-}} \boldsymbol{\phi}^{\text{Tar}}_{m1,-}\, \mathrm{d}\mathbf{x} \\
&+ \int_{\mathbb{T}^{\text{Src}}_{m2,k}} \underbrace{\mathbf{u}_{m2,k}}_{\boldsymbol{\phi}^{\text{Src},T}_{m2,-}\underline{\boldsymbol{U}}_{m2,-,k,-}} \underbrace{\boldsymbol{\phi}^{\text{Tar},*}_{m1,-}}_{\underline{\boldsymbol{Q}}^{m2\to m1,T}} \boldsymbol{\phi}^{\text{Src}}_{m2,-}\, \mathrm{d}\mathbf{x} \\
&= \underline{\boldsymbol{M}}^{\text{Tar}}_{m1,k}\, \underline{\boldsymbol{U}}_{m1,-,k,-} + \underline{\boldsymbol{Q}}^{m2\to m1,T}\, \underline{\boldsymbol{M}}^{\text{Src}}_{m2,k}\, \underline{\boldsymbol{U}}_{m2,-,k,-}\,.
\end{aligned}$$

Similarly to Equation 8.18, if several source elements are present for a given target element, the formula to obtain the agglomerated DOF becomes

$$\underline{\boldsymbol{U}}^{\text{Agg}}_{m1,-,k,-} = \left(\underline{\boldsymbol{M}}^{\text{Agg}}_{m1,k}\right)^{-1} \left[\underline{\boldsymbol{M}}^{\text{Tar}}_{m1,k}\, \underline{\boldsymbol{U}}_{m1,-,k,-} + \sum_{i=2}^{N_{\text{Src}}} \underline{\boldsymbol{Q}}^{mi\to m1,T}\, \underline{\boldsymbol{M}}^{\text{Src}}_{mi,k}\, \underline{\boldsymbol{U}}_{mi,-,k,-}\right]. \tag{8.19}$$

One subtlety needs to be mentioned here. Since we need to apply Equation 8.19 at the beginning of the time step, if there are elements flagged as source because they appeared at this new time step, we do not include them in the source contributions to the agglomerated DOF. Indeed, the data structure for those elements has been initialized but it is filled with zero at this stage.

The injection of the agglomerated solution on the non-agglomerated mesh can then be done with the L_2 projections below, still following the idea of Muller

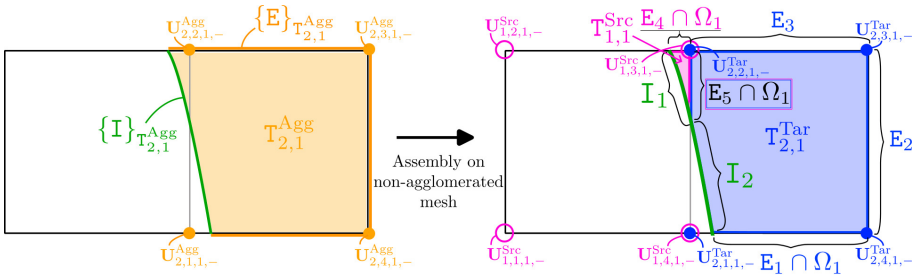


Figure 8.9 / Illustrative example of the assembly of the agglomerated residual. All the operations are performed on the non-agglomerated mesh such that the data structure and the original assembly of the residual can be exploited.

et al. (2016):

$$\int_{\mathrm{T}^{\mathrm{Tar}}_{m1,k}} \left[\mathbf{u}^{\mathrm{Tar}}_{m1,k} - \mathbf{u}^{\mathrm{Agg}}_{m1,k} \right] \boldsymbol{\phi}^{\mathrm{Tar}}_{m1,-} \, \mathrm{d}\mathbf{x} = 0 \,,$$

$$\int_{\mathrm{T}^{\mathrm{Src}}_{m2,k}} \left[\mathbf{u}^{\mathrm{Src}}_{m2,k} - \mathbf{u}^{\mathrm{Agg}}_{m1,k} \right] \boldsymbol{\phi}^{\mathrm{Src}}_{m2,-} \, \mathrm{d}\mathbf{x} = 0 \,,$$

which leads to the expressions for the DOF on the target and source elements in terms of the agglomerated DOF,

$$\underline{\boldsymbol{U}}^{\mathrm{Tar}}_{m1,-,k,-} = \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m1,-,k,-} \,, \qquad [8.20]$$

$$\underline{\boldsymbol{U}}^{\mathrm{Src}}_{m2,-,k,-} = \underline{\boldsymbol{Q}}^{m2 \to m1} \, \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m1,-,k,-} \,. \qquad [8.21]$$

An important remark about the moving interface time integration approach is that the injection procedure described above has to be done at every time levels. This requires to recompute the coupling and agglomerated matrices over all the time levels. Indeed, we cannot just store them from the previous time steps since the agglomeration map, which depends on several time levels, might evolve from one time step to the other.

Assembly of the agglomerated residual

We are now ready to see how the spatial residual is assembled in the XDG method in the presence of cell-agglomeration. Contrary to the spatial residual which is assembled by combining contributions from the non-agglomerated mesh, the assembly of the temporal residual is done directly based on the agglomerated solution. This is covered in Section 10.2.3. We use again the previous illustrative situation to explain the procedure.

Example 8.2: Assembly of the agglomerated residual

Figure 8.9 sketches the situation we are considering. Here we focus on time level t (since we only consider the spatial part of the residual)

and the residual in phase $k = 1$ associated to the target-source pair $\mathtt{T}^{\mathrm{Tar}}_{2,1} - \mathtt{T}^{\mathrm{Src}}_{1,1}$. To assemble the agglomerated residual, we should consider the left part of Figure 8.9, i.e. where a single new element has replaced the original target and source elements. Based on Equation 5.39, Equation 5.50, Equation 8.7 and Equation 8.8, the expression for the residual associated to this new element would be[a]

$$\begin{aligned}\mathcal{N}^{\mathrm{sp,Agg}}_{2,-,1,-} = &\int_{\mathtt{T}^{\mathrm{Agg}}_{2,1}} \nabla\boldsymbol{\phi}^{\mathrm{Agg}}_{2,-} \cdot \underline{\boldsymbol{F}}(\mathbf{u}^{\mathrm{Agg}}_{2,1})\,\mathrm{d}\mathbf{x} \\ &+ \sum_{e\in\{\mathtt{E}\}_{\mathtt{T}^{\mathrm{Agg}}_{2,1}}} \int_{\mathtt{E}_e} \boldsymbol{\phi}^{\mathrm{Agg}}_{2,-}\Big|_{\mathtt{E}_e} \cdot \left(\underline{\boldsymbol{F}}_{\mathtt{E}}(\mathbf{u}^{\mathrm{Agg}}_{2,1}\Big|_{\mathtt{E}_e}, \mathbf{u}^{\mathrm{Agg}}_{?,1}\Big|_{\mathtt{E}_e}) \cdot \mathbf{n}_e\right) \mathrm{d}s \\ &+ \sum_{m\in\{\mathtt{I}\}_{\mathtt{T}^{\mathrm{Agg}}_{2,1}}} \int_{\mathtt{E}_e} \boldsymbol{\phi}^{\mathrm{Agg}}_{2,-}\Big|_{\mathtt{I}_m} \cdot \left(\underline{\boldsymbol{F}}_{\mathtt{E},1}(\mathbf{u}^{\mathrm{Agg}}_{2,1}\Big|_{\mathtt{I}_m}, \mathbf{u}^{\mathrm{Agg}}_{?,1}\Big|_{\mathtt{I}_m}) \cdot \mathbf{n}_{\Gamma,m,1}\right) \mathrm{d}s\,.\end{aligned} \tag{8.22}$$

Computing the agglomerated residual in this way would require to modify the parametric shape functions for each agglomerated cell, to adapt the quadrature rule to the entire agglomerated cell and to remove the DOF associated to the source element from the data structures. In practice, based on the ingenious idea proposed by Muller et al. (2016), we avoid all those modifications by directly exploiting the original operations on the non-agglomerated mesh. To see how it is done, we start by decomposing Equation 8.22 into integrals over the original target and source elements, as illustrated in the right part of Figure 8.9. Before doing so, we recall that

$$\begin{aligned}\mathtt{T}^{\mathrm{Agg}}_{2,1} &= \underbrace{(\mathtt{T}_1 \cap \Omega_1)}_{\mathtt{T}^{\mathrm{Src}}_{1,1}} \cup \underbrace{(\mathtt{T}_2 \cap \Omega_1)}_{\mathtt{T}^{\mathrm{Tar}}_{2,1}}, \\ \{\mathtt{E}\}_{\mathtt{T}^{\mathrm{Agg}}_{2,1}} &= (\mathtt{E}_1 \cap \Omega_1) \cup \mathtt{E}_2 \cup \mathtt{E}_3 \cup (\mathtt{E}_4 \cap \Omega_1)\,, \\ \{\mathtt{I}\}_{\mathtt{T}^{\mathrm{Agg}}_{2,1}} &= \mathtt{I}_1 \cup \mathtt{I}_2\,, \\ \boldsymbol{\phi}^{\mathrm{Tar},*}_{2,-} &= \underline{\boldsymbol{Q}}^{1\to 2,T}\boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\,, \quad \nabla\boldsymbol{\phi}^{\mathrm{Tar},*}_{2,-} = \underline{\boldsymbol{Q}}^{1\to 2,T}\nabla\boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\,.\end{aligned}$$

Equation 8.22 is then rewritten as follows:

$$
\begin{aligned}
\mathcal{N}^{\mathrm{sp,Agg}}_{2,-,1,-} = & \int_{\mathrm{T}^{\mathrm{Src}}_{1,1}} \underline{\boldsymbol{Q}}^{1\to 2,T} \nabla \boldsymbol{\phi}^{\mathrm{Src}}_{1,-} \cdot \underline{\boldsymbol{F}}(\mathbf{u}^{\mathrm{Src}}_{1,1})\, \mathrm{d}\mathbf{x} \\
& + \int_{\mathrm{T}^{\mathrm{Tar}}_{2,1}} \nabla \boldsymbol{\phi}^{\mathrm{Tar}}_{2,-} \cdot \underline{\boldsymbol{F}}(\mathbf{u}^{\mathrm{Tar}}_{2,1})\, \mathrm{d}\mathbf{x} \\
& + \sum_{e \in \{\mathrm{E}_4 \cap \Omega_1\}} \int_{\mathrm{E}_e} \underline{\boldsymbol{Q}}^{1\to 2,T} \boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\Big|_{\mathrm{E}_e} \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{E}_e}, \mathbf{u}_{?,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}_e \right) \mathrm{d}s \\
& + \sum_{e \in \{\mathrm{E}_1 \cap \Omega_1, \mathrm{E}_2, \mathrm{E}_3\}} \int_{\mathrm{E}_e} \boldsymbol{\phi}^{\mathrm{Tar}}_{2,-}\Big|_{\mathrm{E}_e} \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{E}_e}, \mathbf{u}_{?,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}_e \right) \mathrm{d}s \\
& + \sum_{m \in \{\mathrm{I}_1\}} \int_{\mathrm{I}_m} \underline{\boldsymbol{Q}}^{1\to 2,T} \boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\Big|_{\mathrm{I}_m} \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{I},1}(\mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{I}_m}, \mathbf{u}_{?,2}\Big|_{\mathrm{I}_m}) \cdot \mathbf{n}_{\Gamma,m,1} \right) \mathrm{d}s \\
& + \sum_{m \in \{\mathrm{I}_2\}} \int_{\mathrm{I}_m} \boldsymbol{\phi}^{\mathrm{Tar}}_{2,-}\Big|_{\mathrm{I}_m} \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{I},1}(\mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{I}_m}, \mathbf{u}_{?,2}\Big|_{\mathrm{I}_m}) \cdot \mathbf{n}_{\Gamma,m,1} \right) \mathrm{d}s \\
& + \sum_{e \in \{\mathrm{E}_5 \cap \Omega_1\}} \int_{\mathrm{E}_e} \underline{\boldsymbol{Q}}^{1\to 2,T} \boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\Big|_{\mathrm{E}_e} \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{E}_e}, \mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}^{\mathrm{Src}}_e \right) \mathrm{d}s \\
& + \sum_{e \in \{\mathrm{E}_5 \cap \Omega_1\}} \int_{\mathrm{E}_e} \boldsymbol{\phi}^{\mathrm{Tar}}_{2,-}\Big|_{\mathrm{E}_e} \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{E}_e}, \mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}^{\mathrm{Tar}}_e \right) \mathrm{d}s\,.
\end{aligned}
\tag{8.23}
$$

The last two terms have been artificially added and concern the contributions from the face separating the target and the source elements. Those two terms are needed to retrieve the expression of the original residuals but they can only be added if their sum cancel out. This is the case because by conservativity of the numerical flux function, we have

$$
\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{E}_e}, \mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}^{\mathrm{Tar}}_e = -\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{E}_e}, \mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}^{\mathrm{Tar}}_e\,,
$$

which allows us to write the two last terms into a single one,

$$
+ \sum_{e \in \{\mathrm{E}_5 \cap \Omega_1\}} \int_{\mathrm{E}_e} \left(\boldsymbol{\phi}^{\mathrm{Tar}}_{2,-}\Big|_{\mathrm{E}_e} - \underline{\boldsymbol{Q}}^{1\to 2,T} \boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\Big|_{\mathrm{E}_e} \right) \cdot \left(\underline{\boldsymbol{F}}_{\mathrm{E}}(\mathbf{u}^{\mathrm{Tar}}_{2,1}\Big|_{\mathrm{E}_e}, \mathbf{u}^{\mathrm{Src}}_{1,1}\Big|_{\mathrm{E}_e}) \cdot \mathbf{n}^{\mathrm{Tar}}_e \right) \mathrm{d}s\,.
$$

Now, it can be shown that by construction, the third and fourth rows of the coupling matrix in this situation are

$$
\underline{\boldsymbol{Q}}^{1\to 2,T}[:,3] = [0,\,,1,\,0,\,,0]^T\,, \underline{\boldsymbol{Q}}^{1\to 2,T}[:,4] = [1,\,,0,\,0,\,,0]^T\,,
$$

such that

$$
\boldsymbol{\phi}^{\mathrm{Tar}}_{2,-}\Big|_{\mathrm{E}_e} - \underline{\boldsymbol{Q}}^{1\to 2,T} \boldsymbol{\phi}^{\mathrm{Src}}_{1,-}\Big|_{\mathrm{E}_e} = \mathbf{0}\,, \tag{8.24}
$$

which confirms that the addition of the two last terms was consistent. Grouping the terms associated to the target and the source elements in Equation 8.23, we obtain that

$$\mathcal{N}^{\text{sp,Agg}}_{2,-,1,-} = \mathcal{N}^{\text{sp,Tar}}_{2,-,1,-} + \underline{\boldsymbol{Q}}^{1\to 2,T}\, \mathcal{N}^{\text{sp,Src}}_{1,-,1,-}\,,$$

which shows that the spatial part of the agglomerated residual can be assembled from the original residual contributions from the target and source elements, provided a multiplication of the source contribution by the coupling matrix. We could arrive at this elegant result because the target and source solutions, $\mathbf{u}^{\text{Tar}}_{2,1}$ and $\mathbf{u}^{\text{Src}}_{1,1}$, that appear in the equation above come from the agglomerated solution injected onto the non-agglomerated mesh, i.e.

$$\mathbf{u}^{\text{Tar}}_{2,1} = \boldsymbol{\phi}^{\text{Tar},T}_{2,-}\, \underline{\boldsymbol{U}}^{\text{Tar}}_{2,-,1,-}\,,$$
$$\mathbf{u}^{\text{Src}}_{1,1} = \boldsymbol{\phi}^{\text{Src},T}_{1,-}\, \underline{\boldsymbol{U}}^{\text{Src}}_{1,-,1,-}\,,$$

with $\underline{\boldsymbol{U}}^{\text{Tar}}_{2,-,1,-}$ and $\underline{\boldsymbol{U}}^{\text{Src}}_{1,-,1,-}$ given respectively by Equation 8.20 and Equation 8.21.

[a] To lighten the notations, we only write the first part of the face and interface residuals and disregard the second part linked to the diffusive transpose term. We also do not consider any source term. Finally, we assume that there is no contribution from the domain boundaries.

The example above allows us to extract a general formula to assemble the spatial part of the agglomerated residual in element $\mathbb{T}_{m1}$ and associated to phase k:

$$\mathcal{N}^{\text{sp,Agg}}_{m1,-,k,-} = \begin{cases} \mathcal{N}^{\text{sp,Tar}}_{m1,-,k,-} + \sum_{i=2}^{N_{\text{Src}}} \underline{\boldsymbol{Q}}^{mi\to m1,T}\, \mathcal{N}^{\text{sp,Src}}_{mi,-,k,-}\,, & \text{if } \mathbb{T}_{m1,k} \text{ is target}\,, \\ 0\,, & \text{if } \mathbb{T}_{m1,k} \text{ is source}\,, \\ \mathcal{N}^{\text{sp}}_{m1,-,k,-}\,, & \text{otherwise}\,, \end{cases} \qquad [8.25]$$

with the non-agglomerated XDG spatial residuals given by Equation 8.5 in non-cut elements and by Equation 8.6 in cut ones, removing the temporal contribution $\widetilde{\mathcal{J}}$. It is worth noting that the residual is not filled for the DOF associated to the source elements since we want to solve for the agglomerated solution. The solution in the source element can then be obtained via Equation 8.21, while the solution in the target element is simply the agglomerated solution itself given Equation 8.20.

Masking and faces contributions To conclude this subsection, we should discuss the contributions of the faces located inside the agglomerated cell, like the face $\mathbb{E}_5 \cap \Omega_1$ in Figure 8.9. We have seen with the example above that for a face separating the target element from one of its source element, the extension

of the target basis function should be such that the coupling matrix ensures a net-zero contribution from that face to the agglomerated residual when summing the target and source contributions (see Equation 8.24). However, because we are breaking agglomeration chains and allowing target-source pairs to exist without sharing a common face, this property may no longer hold in certain situations. That is why a mask function forces the contributions from those potentially problematic faces to zero.

Expression of the agglomerated Jacobian matrix

As already mentioned, the non-linear system will be solved for the agglomerated solution. To this end, we need to obtain the Jacobian of the agglomerated residual with respect to the agglomerated solution. Based on Equation 8.25, we will therefore need to assemble, for each target element,

$$\frac{\partial \mathcal{N}^{\mathrm{sp,Agg}}_{m1,-,k,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{-,-,-,-}} = \frac{\partial \mathcal{N}^{\mathrm{sp,Tar}}_{m1,-,k,-}}{\partial \underline{\boldsymbol{U}}_{-,-,-,-}} \underbrace{\frac{\partial \underline{\boldsymbol{U}}_{-,-,-,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{-,-,-,-}}}_{\underline{\boldsymbol{I}}} + \sum_{i=2}^{N_{\mathrm{Src}}} \left[\frac{\partial \mathcal{N}^{\mathrm{sp,Tar}}_{m1,-,k,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Src}}_{mi,-,-,-}} \underbrace{\frac{\partial \underline{\boldsymbol{U}}^{\mathrm{Src}}_{mi,-,-,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m1,-,-,-}}}_{\underline{\boldsymbol{Q}}^{mi\to m1}} \right]$$

$$+ \sum_{i=2}^{N_{\mathrm{Src}}} \left[\underline{\boldsymbol{Q}}^{mi\to m1,T} \frac{\partial \mathcal{N}^{\mathrm{sp,Src}}_{mi,-,k,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Src}}_{mi,-,-,-}} \underbrace{\frac{\partial \underline{\boldsymbol{U}}^{\mathrm{Src}}_{mi,-,-,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m1,-,-,-}}}_{\underline{\boldsymbol{Q}}^{mi\to m1}} + \underline{\boldsymbol{Q}}^{mi\to m1,T} \frac{\partial \mathcal{N}^{\mathrm{sp,Src}}_{mi,-,k,-}}{\partial \underline{\boldsymbol{U}}_{-,-,-,-}} \underbrace{\frac{\partial \underline{\boldsymbol{U}}_{-,-,-,-}}{\partial \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{-,-,-,-}}}_{\underline{\boldsymbol{I}}} \right], \tag{8.26}$$

In this equation, we use $\underline{\boldsymbol{U}}_{-,-,-,-}$ to refer to the global solution vector in the classical and target elements but not in the source elements. Thus, the first term is the original Jacobian (w.r.t. normal and target solution) for which we skip all the source contributions. The remaining terms come from the dependence of the agglomerated residuals to the source residuals. Even if the Jacobian assembly is a bit more complicated than the residual one, we can once again re-use all the operations on the non-agglomerated mesh.

8.3.2 Artificial viscosity technique in cut elements

As compressible regimes are considered for some test cases treated in this work, a special technique is required to remove spurious oscillations that might appear in the vicinity of regions with high gradients of flow quantities. We rely here on the widespread shock-capturing localized Laplacian artificial viscosity technique presented by Per-Olof Persson and Peraire (2006) for DG methods and the subsequent improvements proposed by Barter and Darmofal (2010) and Vandenhoeck and Lani (2019). This choice has been made because of the availability of this technique in `Argo` for standard DG discretization, scrutinized by Bordes (2019), and its ease of adaptation to the XDG method. It is interesting

to note that the XDG method has also been applied in a shock-fitting framework in the work of Geisenhofer et al. (2020) and Vandergrift and Kummer (2023). This was however not investigated as part of this work.

Detection of shock location by a sensor

Contrary to material interfaces that are tracked by a technique like the level-set, we need to identify the *a priori* unknown position of the shock by an appropriate detector. The detector of Per-Olof Persson and Peraire (2006) works on the principle that polynomial expansions of the solutions of two different orders will give similar values in regions where the solution is smooth but will be different when it contains discontinuities. In each element $\mathbb{T}_m$, the polynomial expansion of order p, $\mathbf{u}_m$ and its projection on a function space of order $p-1$, $\hat{\mathbf{u}}_m$, are considered. Discontinuities can then be detected by comparing the two expansions of different orders for a particular solution field. Several choices are possible depending on the applications. In this work, we mostly rely on the density field such that the cell-local sensor is obtained by

$$S_m = \frac{\int_{\mathbb{T}_m} (\rho_m - \hat{\rho}_m) \cdot (\rho_m - \hat{\rho}_m) \, \mathrm{d}V}{\int_{\mathbb{T}_m} (\rho_m \cdot \rho_m) \, \mathrm{d}V} \,. \tag{8.27}$$

Following Geisenhofer et al. (2019), the sensor is reformulated on cells cut by the interface. In the kth phase on the mth element, the sensor is simply computed by

$$S_{m,k} = \frac{\int_{\mathbb{T}_{m,k}} (\rho_{m,k} - \hat{\rho}_{m,k}) \cdot (\rho_{m,k} - \hat{\rho}_{m,k}) \, \mathrm{d}V}{\int_{\mathbb{T}_{m,k}} (\rho_{m,k} \cdot \rho_{m,k}) \, \mathrm{d}V} \,, \tag{8.28}$$

using the available quadrature rules on cut cells. Note that we do not consider agglomeration in Equation 8.28. The elementwize shock sensor is finally expressed as

$$s_{m,k} = \log_{10}(S_{m,k}) \,. \tag{8.29}$$

Smoothing of the shock by the addition of artificial viscosity

When a troubled cell has been detected, artificial viscosity has to be added depending on the strength of the shock through the artificial viscosity parameter $\epsilon_{m,k}$ given as the product between a maximum viscosity to reach per element, $\epsilon^*_{m,k}$, and a smoothed detector function, $d_{m,k}$,

$$\epsilon_{m,k} = \epsilon^*_{m,k} \, d_{m,k} \,. \tag{8.30}$$

According to Vandenhoeck and Lani (2019), the maximum value of the viscosity is expressed based on the reasoning that a higher artificial viscosity is needed at higher speeds and in bigger elements (in order to spread out the shock over

the element) such that

$$\epsilon^*_{m,k} = f \frac{\Delta x_m}{p} \|\lambda_{m,k}\|_{\max},$$

where p is the interpolation order, Δx_m the characteristic length of the element and $\|\lambda_{m,k}\|_{\max}$ the maximum eigenvalue of the convective fluxes on the element. To avoid the need for fine-tuning the parameters, which can be quite cumbersome in many circumstances, Vandenhoeck and Lani (2019) derive an approximate expression for the function f based on the number of cells over which the shock should be spread,

$$f = \alpha \, \delta^*_s \, \frac{\mathrm{Ma}_\infty - 1}{\mathrm{Ma}_\infty},$$

where $\alpha = 0.027$ based on some calibration tests, δ^*_s is a characteristic shock thickness that is a measure for the number of elements over which the shock is spread out and Ma_∞ is the freestream Mach number of the problem under consideration.
The detector function in Equation 8.30 is such that the artificial viscosity is only active in problematic regions,

$$d_{m,k} = \begin{cases} 0 & \text{if } s_{m,k} < s_0 - \kappa, \\ \frac{1}{2}\left[1 + \sin\left(\frac{\pi(s_{m,k} - s_0)}{2\kappa}\right)\right] & \text{if } s_0 - \kappa \leq s_{m,k} \leq s_0 + \kappa, \\ 1 & \text{if } s_{m,k} > s_0 + \kappa, \end{cases}$$

where $s_{m,k}$ is the sensor given by Equation 8.28 and Equation 8.29, s_0 the sensor threshold and κ a parameter to control the working spectrum of the artificial viscosity, i.e. the interval over which the artificial viscosity increases smoothly from zero to its maximum value. The reference smoothness is set to

$$s_0 = -3 \, \log_{10}(p),$$

according to Park et al. (2014).
As pointed out by Barter and Darmofal (2010), it is essential to enforce $C0$-continuity for the artificial viscosity field in order to obtain a robust and reliable method. Thus, a loop on the faces is performed and the maximum trace values on the faces are collected before being reinterpolated into the elements to obtain a continuous field.
The artificial viscosity term is finally injected in the equations in the form of a Laplacian term in the diffusive fluxes:

$$\underline{\boldsymbol{F}}^d_{m,k}(\mathbf{u}, \nabla\mathbf{u}) \mathrel{+}= \epsilon_{m,k} \nabla \mathbf{u}_{m,k}, \qquad [8.31]$$

where $\nabla\mathbf{u}$ is the gradient of the conservative variables.
An important remark to end this section is about the fact that for the cut elements,

the artificial viscosity is not included in the phase interface term $\mathtt{I}_{m,k}$, but only in the volume and the element face terms. The reason comes from the observation that adding this smoothing term leads to a small smearing of the discontinuity at the interface, which should be avoided.

8.3.3 Mesh-to-mesh projection

The need to interpolate the discrete solution from one mesh to another arises frequently in computational physics, such as in the case of adaptive remeshing or the restart of the simulation on a new mesh. This interpolation step inevitably entails some discretization errors that should be kept as low as possible in order not to disrupt the accuracy of the scheme. In `Argo`, this mesh-to-mesh procedure was already available and follows four main steps:

- *Loading old solution and topology*: the old solution is loaded from the intermediate data that are saved during the course of the simulation.
- *Creating search tree on old topology*: a search tree is prepared based on the vertices of the old mesh.
- *Listing target points on new mesh*: the points on the new mesh, either interpolation or quadrature points (see below), are gathered along with their positions.
- *Interpolating on the target points*: the search tree is used for each target point to identify the best element on the old mesh from which the nodal degrees of freedom and the shape functions will be combined to interpolate the old solution at the position of the target point.

What is done in this work is to extend this procedure to the case of two-phase flows separated by an interface cutting across the elements. This necessitates the following modifications:

- *Loading old solution and topology*: the intermediate saving process needs to also consider the extended degrees of freedom on the cut elements.
- *Flagging of old topology*: an additional step is inserted in order to flag the the elements that are cut by the interface, both on the old and the new meshes, over the number of time levels required by the time-stepping scheme.
- *Creating search tree on old topology*: the search tree is extended such that it does not only return the best element on the old mesh based on the position of a target point, but rather a list of best elements candidates. This is done because some elements in the neighborhood of the target point might not be suited for the interpolation. This will be the case if the element contains a single phase, which is different from the one associated to the target point.
- *Listing target points on new mesh*: on the cut elements, we add the possibility to collect the quadrature points used for the integrals on implicitly defined domains.

The last step, namely *interpolating on the target points*, is described below and the distinction is made between two types of interpolation.

Direct interpolation

For the direct interpolation approach, the interpolation points are the targets. In this situation, we might encounter the following issue for an unfitted discretization. It might happen that the position of the target point on the old mesh is such that no element in a relatively close neighborhood is suitable, i.e. they would all contain a single phase that is not the one of the target point. In this case, we need to extend the list of elements candidates so that we can find an appropriate element. This however will results in the introduction of more important interpolation errors.

L_2 projection

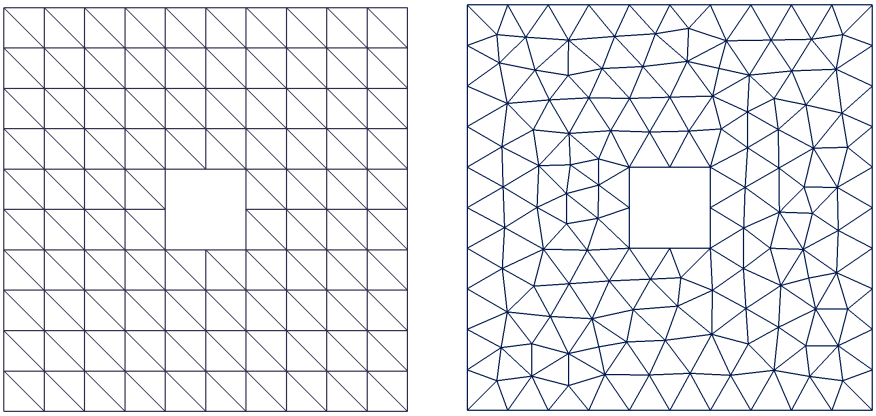
As recalled by Thompson (2015), the orthogonal projection must be optimal, i.e. leading to the lowest interpolation errors, in the L_2 norm. It therefore seems logical to apply it here in order to limit as much as possible the interpolation errors associated to the solution transfer from one mesh to the other. In this situation, the target points for the interpolation on the new mesh are the standard quadrature points on non-cut elements and the specific quadrature points on the cut entities. Let $\mathcal{W}_p^X(\varphi, \mathbb{T}^{\text{old}})$ and $\mathcal{W}_p^X(\varphi, \mathbb{T}^{\text{new}})$ be the finite dimensional spaces associated to the old and new meshes, respectively, and spanned by the basis functions $\boldsymbol{\phi}_{-,-}^{\text{old}}$ and $\boldsymbol{\phi}_{-,-}^{\text{new}}$. Let $\mathbf{u}^{\text{old}}(\mathbf{x})$ and $\mathbf{u}^{\text{new}}(\mathbf{x})$ be the approximate solution on the old and new meshes. Then, $\mathbf{u}^{\text{new}}$ must satisfy the following orthogonality condition

$$\int_\Omega \left(\mathbf{u}^{\text{new}} - \mathbf{u}^{\text{old}}\right) \mathbf{w} \, \mathrm{d}\mathbf{x} = 0\,, \quad \forall \mathbf{w} \in \mathcal{W}_p^X(\varphi, \mathbb{T}^{\text{new}})\,,$$

which is equivalent to the requirement that the error function $\mathbf{u}^{\text{new}} - \mathbf{u}^{\text{old}}$ be orthogonal to each basis function in the space $\mathcal{W}_p^X(\varphi, \mathbb{T}^{\text{new}})$, i.e.

$$\sum_{m \in \mathbb{T}} \int_{\mathbb{T}_m} \left(\boldsymbol{\phi}_{m,-}^{\text{new},T}(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}^{\text{new}} - \boldsymbol{\phi}_{m,-}^{\text{old},T}(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}^{\text{old}}\right) \phi_{m,n}^{\text{new}}(\mathbf{x}) \, \mathrm{d}\mathbf{x} = 0\,, \quad \forall n \in \{1, \dots, N_n\}\,. \tag{8.32}$$

The shape functions and the nodal degrees of freedom on the old mesh in this equation, i.e. $\boldsymbol{\phi}_{m,-}^{\text{old},T}$ and $\underline{\boldsymbol{U}}_{m,-,-}^{\text{old}}$, are obtained via the same five-step procedure as described above. The main difference with the direct interpolation at this stage is that the target points on which the interpolation needs to be performed are guaranteed to be located strictly on one side of the interface. This avoids the problem that can arise for the interpolation points.
It has been observed that assembling Equation 8.32 as it is on every element, no matter their cut configuration, leads to oscillatory projected solutions in



a / Structured mesh on which the intermediate solution is computed.

b / Unstructured mesh on which the intermediate solution is interpolated.

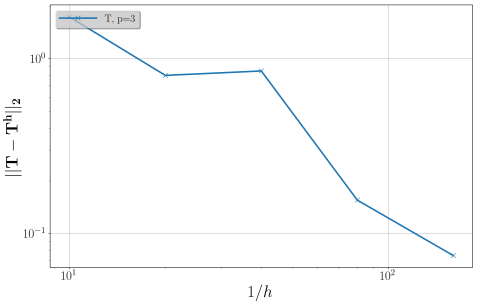
Figure 8.10 / Mesh-to-mesh projection test case – one pair of meshes.

tiny cut elements. Thus we reformulate this L_2-projection by including agglomeration, similarly to what is done in Equation 8.19, except that we do not use the agglomerated target and source mass matrices here since a product between old and new shape functions is present. An important implementation aspect to emphasize is that this projection not only needs to be solved for the current restarting time, but also over possibly several previous time levels. This requires, in addition to the flagging of cut elements, the construction of quadrature rules and the computation of agglomerated data over several time levels. Those features have also to be added for the moving interface time integration approach (see Section 10.2.3). Hence, their implementation serve two purpose.

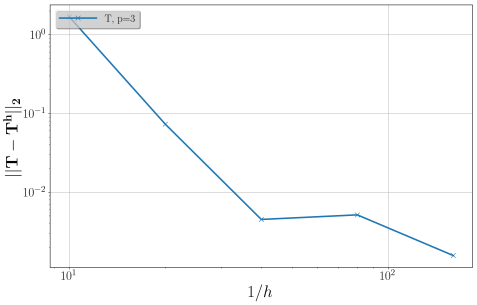
Assessment of the projections

In order to assess the projection strategies described above, we consider a series of meshes pairs with different resolution, one of them being shown in Figure 8.10. The test case we address here is the spinning bubble test case described in Section 11.3.3. On the structured meshes, we initialize the solution with the exact reference manufactured expression and we perform a single iteration at the end of which we save an intermediate solution. We then use this intermediate solution, project it on the corresponding unstructured mesh (which has a comparable number of elements) and evaluate the errors between the projected solution and the manufactured one. The results for the direct projection and the L_2-projection without and with agglomeration are shown in Figure 8.11. It can be noticed that only the L_2-projection combined with agglomeration allows retrieving the theoretical convergence rate.

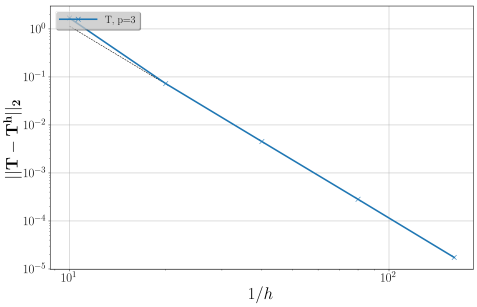
To close this section, we should point out that the interpolation strategies presented above do not ensure conservativity of the solution. Indeed, for the



a / Direct projection.



b / L_2 projection without agglomeration.



c / L_2 projection with agglomeration.

Figure 8.11 / Mesh-to-mesh projection test case – spatial convergence study at $p = 3$ of the projected temperature field based on a manufactured solution.

direct interpolation approach, there is no guaranty that the integral of the output field will be the same as the integral of the input field. For the L_2-projection on the other hand, this integral equality should be respected in theory. However, Farrell and Maddison (2011) note that this is not strictly the case in practice because the basis functions of the donor mesh are (in general) discontinuous piecewise polynomials over any given element of the target mesh. This results in the fact that the quadrature accuracy of the mixed product between the basis functions in the second term of Equation 8.32 will be deteriorated. This introduced error causes the loss of conservation. To remediate this issue, Farrell and Maddison (2011) propose to construct a "supermesh", which is formed by the intersections of the target and donor elements. On each element of this supermesh, the basis functions of both old and new meshes are, by construction, continuous polynomials, allowing high accuracy evaluation of the projection integrals. This technique has been tested by Levý and G. May (2023) in the context of a HDG solver but not implemented in the present work.

Contribution 8.1: XDG method for sharp interface two-phase flows

The main contributions that can be identified from this chapter are:

- A broad overview of the existing numerical approaches to deal with discontinuous solutions and the fact that the XDG method is put into perspective in relation to others geometrically unfitted FE-type methods.
- A description of the XDG method relying on the same formalism as in Chapter 5, which allows to highlight the addition of an interface term in the variational form on the cut elements. This is the first time that the XDG form is presented in this very general way which is valid for generic advection-diffusion systems.
- A thorough explanation, supported by an illustrative example, of the assembly of the spatial residual with cell-agglomeration.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform are about the extension of the original XDG implementation performed by Pierre Schrooyen and its Cenaero's coworkers to two-phase flows. Those include:

- The interfacing with two external libraries providing quadrature rules over implicitly defined geometries.
- The extension of dynamic data allocation based on the cut topology to be compatible with two-phase configurations.
- The adaptation of output routines to correctly export the solutions in both phases in cut elements.

- The improvement of cut element and cut face detection based on Bernstein basis.
- The extension of the mesh-to-mesh projection algorithms to the situation where the mesh is cut by an interface.

Rest of this page intentionally left blank.

9

Enforcement of jump conditions between two compressible viscous phases

The material of this chapter is the core of the work published in Henneaux et al. (2023f) and presented in Henneaux et al. (2022b) and Henneaux et al. (2022c).

9.1 Overview of interface coupling approaches for compressible two-phase flows . . . 314
9.1.1 Monolithic versus partitioned approaches . . . 314
9.1.2 Enforcement of the jumps with a monolithic approach for two-phase flows . . . 315
9.2 Enforcement of the jump conditions into the variational form . . . 318
9.2.1 Continuous strong form of the equations . . . 319
9.2.2 Interface convective numerical fluxes . . . 320
9.2.3 Interface diffusive numerical fluxes . . . 326
9.3 Resulting discretization implemented and its analysis . . . 330
9.3.1 Resulting discretization . . . 330
9.3.2 Jacobians of interface terms . . . 332
9.3.3 Weak consistency . . . 333

9.1 Overview of interface coupling approaches for compressible two-phase flows

9.1.1 Monolithic versus partitioned approaches

As already introduced in Section 3.1.2, the use of the local instant formulation to describe the two-phase systems we are considering in this work leads to a multi-domain problem, which can additionally be multiphysics if the two phases involved behave differently (e.g. gas and liquid). To address such problems, coupling strategies, both in space and time if the problem is unsteady, need to be considered. We focus in this section on the existing approaches to couple spatially sub-problems defined on separated parts of the computational domain. In our case, it corresponds to taking a snapshot of the two phases separated by a fixed interface at a given time and to looking at how the two sets of equations can be spatially discretized. The temporal coupling between the interface motion and the fluid equations is elaborated in Section 10.3.1. Two principal approaches can be identified: monolithic and partitioned approaches. As the aim here is to give an overview of the two main strategies without going into too much detail, they will be examined in relation to each other. In practice, the distinction is a little more subtle, as there is a whole range of mixed methods that fall between these two categories. This is summarized in Uekermann (2016, Sec. 1.2).

Partitioned approach In the partitioned approach, each physical system is discretized by a dedicated method and the "two-way" jump conditions become "one-way" boundary conditions allowing to exchange the information between the separated solvers at specified instants and/or spatial locations. The obvious benefit of this approach is the possibility to use tailored numerical schemes in every subdomain, an aspect that becomes all the more important if time and spatial scales between the domains differ significantly. On the one hand, in the weak/explicit coupling procedure, a fixed amount of solvers calls is performed during one time step and the separated problems are solved sequentially. On the other hand, strong/implicit coupling schemes consider subiterations between the problems such that some prescribed interface conditions are satisfied up to a tolerance level before going to the next time step. The partitioned approach was investigated by Pierre Schrooyen et al. (2018) for ablation problems, in which a comparison between explicit and implicit loosely coupled strategies is conducted.

Monolithic approach In the monolithic approach, the coupled problem is discretized at each time step into a single large non-linear system. For our case, it means that the equations on either side of the interface are included in the same system, with the jump conditions acting as coupling constraints which

are parts of the system as well. This approach thus requires the development of a dedicated solver, which is the topic of this chapter and the previous one.

Choice of the monolithic approach The main reason for which our choice fell on the monolithic approach has to do with the very important stiffness and non-linear nature of the coupled problem investigated. The use of an explicit segregated approach would certainly be prone to severe instabilities. To control them, a solution of equal quality than a monolithic solution has to be recovered in every time step. The use of an implicit coupling presents however important numerical challenges. First, the non-linear equations made of the coupling conditions between the systems have to be assembled and solved. The solution process usually requires stabilization and convergence improvement methods, for instance through quasi-Newton solvers such as in Delaissé et al. (2023). The other delicate aspect is the formulation of appropriate convergence criteria to decide at which point the coupling is accurate enough. This is a critical part as it can highly influence the stability and accuracy of the process. We could then argue that implementation difficulties posed by the monolithic approach are also present in the implicit segregated approach. Moreover, the iterative process required at every time step of the latter approach is likely to make it at least as costly as the monolithic one. We therefore opted for the monolithic approach for its incorporated robustness, temporal stability and conservativity properties, which we believe is essential for the targeted applications. It may be noted, however, that although this choice is probably the most robust and efficient for the present applications, it is not the most flexible in terms of reusability and extension to other physical problems. Uekermann (2016) talks about the concept of "time-to-solution" which is the overall time spent from the algorithm development through the code implementation to the final simulation. To keep this time-to-solution at a reasonable level, the only viable strategy to adopt in the development of a numerical platform accommodating a wide range of different physical models is the segregated approach. That is not the purpose of this work. We can add a nuance to this remark by highlighting the general nature of the fluid dynamics equations implemented in this work which should allow to treat a quite large class of multiphase fluid dynamics problems and could be extended with more complex models.

9.1.2 Enforcement of the jumps with a monolithic approach for two-phase flows

The review below follows the ones about the techniques to compute interface motion in Section 6.1 and the methods to represent discontinuous fields in Section 8.1. This review is about the last piece of our two-phase flow solver which concerns the strategy to enforce the physical interface jump conditions. This strategy depends on the way the discontinuity is treated.

Sharp interface The jump conditions to be incorporated into the numerical scheme are the ones directly obtained from the local instant formulation of the two-phase flow configuration presented in Section 3.2.2, i.e. the coupling conditions between the standard differential balance equations in the bulk of each phase. Whether the sharp representation is achieved by interface-fitting mesh (ALE or cut-cell) or with extension of the approximation space, the steps to be taken to impose the jumps are the same: (i) the two separated sets of solutions variables (and their gradients), first have to be obtained on each side of the interface, (ii) then numerical fluxes need to be assembled taking into account the jump conditions on the solution fields (and their gradients), (iii) lastly the fluxes are inserted into the discretization (the strong integral form for the FVM, the weak variational form for FE-based methods) and integrated over the interface, which enforces the jumps. Different methodologies following those common steps have been designed in the context of compressible two-phase flows:

- *Ghost-fluid methods* (*GFM*): the most widespread approach is based on the idea of assigning to ghost fluid states in the vicinity of the interface solutions values such that, when combined with their real counterparts, produce numerical interface flux satisfying the jump conditions. There exists many variants of the GFM which differ in the way in which the ghost fluid states are populated. The original GFM (OGFM) of Fedkiw et al. (1999) can be mentioned. It uses the local real fluid velocity and pressure to define the corresponding ghost fluid states. In order to take into account the influence of both the material properties and wave interaction with the interface, a fundamental improvement of the OGFM consists in solving a local multi-medium Riemann problem to define ghost fluid states. This has resulted in the modified GFM (MGFM) by T.G. Liu et al. (2003) first and subsequently to several variants. A recent extension of the MGFM to treat inviscid multispecies compressible phases is presented in Xu et al. (2022). When viscous effects are considered, solving an interfacial Riemann problem including the diffusive components of the jumps is not an easy task as this would introduce solutions gradients modifying the original hyperbolic nature of the Riemann problem, as pointed out in Rohde (2018, Sec. 1.1). To circumvent this, it is possible to decouple the hyperbolic and parabolic steps like in Das and Udaykumar (2020), Houim and Kuo (2013), and Majidi and Afshari (2016). Worth mentioning is the recent work in Tiegang Liu et al. (2019) and Xiaotao Zhang et al. (2022) shedding the light on the first-order time error of the previously mentioned GFMs which can lead to time-increasing errors for accelerating interfaces. They propose to overcome this issue by solving a generalized Riemann problem at the interface. Another caveat of the previous GFM versions is that the ghost states assignment is not related to the exact location of the interface inside the cut-cells. A weighted GFM version is proposed in Y.-L. Liu et al. (2021) to alleviate that aspect. However, an unavoidable pitfall of all those GFM are the lack of exact discrete

conservation, which has been investigated in Xu and Tiegang Liu (2011).

- *Conservative sharp interface*: to ensure strict conservativity at the interface, various work have exploited the idea of solving a two-phase Riemann problem at the interface, here not with an interface cutting across the cells like in GFM but rather in the framework of cut-cell and interface-fitting methods. Solving the interface Riemann problem (exactly or not) provides the interface convective numerical fluxes as well as the interface speed for the mesh motion or the interface-capturing method. Recent work include Lin et al. (2017) and Shen et al. (2020) (FV cut-cell method), X.-L. Deng and M. Li (2018) (DG cut-cell method) or Nguyen et al. (2010) (ALE DGM). Including viscous effects in the coupling of two compressible phases pose the same difficulties as with the GFM. Assuming continuity of the viscous stresses at the interface, an effective viscosity approach has been proposed in Luo et al. (2015) and re-used in Hoppe et al. (2022) and Long et al. (2021b). Another noticeable work taking into account viscous effects is the one in Chang et al. (2013) where viscous contributions are frozen in the interface Riemann solver to compute the convective fluxes. For the computation of the interface diffusive fluxes, sided finite difference formula are considered. Worth mentioning as well are the work of Bai and X. Deng (2017) (FVM) and Bai and M. Li (2022) (DGM) combining the advantages of cut-cells and ghost-fluid methods.
- *Constrained reconstruction*: in the cut-cell based method introduced in R.R. Nourgaliev et al. (2008), jump conditions are directly imposed on the interface by a constrained least-squares reconstruction of fluid variables and their gradients on interface nodal points. The advantage of this approach is that there is no splitting between the convective and diffusive parts of the jumps during their imposition. On the other hand, an overdetermined system has to be solved at each iteration. This approach has been adopted in H. Kim and M.-S. Liou (2019) and adapted to a reconstructed DG framework in Robert Nourgaliev et al. (2017).
- *Weak imposition*: the last strategy to impose jump conditions in a sharp manner is to be found in FE-based methods. The general idea is to consistently add to the variational form some terms integrated on the interface whose role is to "test" the jump conditions (written in terms of convective and diffusive fluxes normal to the interface), thus weakly enforcing them. The vast majority of FE-based methods solving compressible equations consider inviscid flow models so that the only term responsible for jumps imposition in the variational form is the numerical flux provided by the multiphase Riemann solver. Examples of work dealing with *viscous* compressible two-phase flows in a FE-based sharp interface context are Potghan and Jog (2022) and Yu Zhang et al. (2019) (the work of Robert Nourgaliev et al. (2017) mentioned above also does it but in a different way). Although substituting the jumps into the weak form is quite common in the incompressible multiphase community (as can be seen from the review in Section 8.1.4), this is much less so for compressible

two-phase flows.

Diffuse interface Compared to the sharp interface approach where the main challenge is to design a numerical strategy able to strongly couple the two sets of equations, the averaged equations of the multi-fluid framework pose particular difficulties on the modeling and mathematical sides. Indeed, in order to close the system of equations, the interfacial source terms coming from the averaging process need to be properly modeled. First, they need to encompass the interfacial exchange phenomena of the physical situation considered so that the jump conditions are fulfilled within the artificial mixture zone through different relaxation operators. Then, they should be carefully chosen in such a way that numerical fluctuations are avoided across the contact discontinuity thanks to suitable thermodynamic mixture laws. Depending on the assumptions made for the closure, a variety of models have been developed, from the seven equations fully out-of-equilibrium model (Saurel and Abgrall (1999)) to several reduced models assuming equilibrium of various quantities like pressure, velocity and/or temperature. Because of the homogenization of the interface region, not all the characteristics inherent to the local instant formulation can be brought into the multi-fluid models.

In the context of this thesis exploiting the XDG method, the strategy developed to enforce the jump conditions is part of the sharp interface weak imposition category listed above.

9.2 Enforcement of the jump conditions into the variational form

This section addresses the computation of the still unspecified interfacial numerical fluxes of Equation 8.8 in order to couple through complex jump conditions (from Section 3.4) two general sets of governing equations involving inviscid and viscous effects (from Section 2.4). Capitalizing on the way the fluxes are specified in the bulk of each phase (see Section 5.2.3), the interfacial jumps are weakly imposed into the coupled variational formulation. The key idea of weak constraint imposition is to relax the strength of imposing the constraints to the benefit of improved accuracy, stability and robustness, as noted by Benedikt Schott (2017, Sec. 4.2.2). These advantages are all the more marked at a phase interface across which huge contrasts in material properties potentially exist. Before detailing the computation of the convective and then diffusive fluxes, some notations are first introduced. The normal to the interface contained in element $\mathbb{T}_m$, $\mathbf{n}_{\Gamma,m}$, has the same direction as the normal vector emanating from phase 1, $\mathbf{n}_{\Gamma,m,1}$, and is opposite to the normal associated to phase 2:

$$\mathbf{n}_{\Gamma,m} = \mathbf{n}_{\Gamma,m,1} = -\mathbf{n}_{\Gamma,m,2}\,. \tag{9.1}$$

The 'swapped' phase index k^* is simply defined as

$$k^* = \begin{cases} 1 & \text{if } k = 2\,, \\ 2 & \text{if } k = 1\,. \end{cases} \tag{9.2}$$

Finally, a tag t_k is associated to each phase such that

$$t_k = \begin{cases} 1 & \text{if } k = 1\,, \\ -1 & \text{if } k = 2\,. \end{cases} \tag{9.3}$$

9.2.1 Continuous strong form of the equations

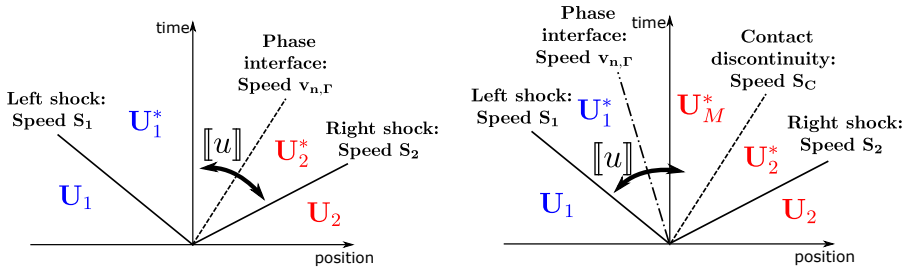
The first step is to state here the complete continuous strong formulation of the non-linear coupled boundary value problem we want to discretize, based on notations introduced in Figure 8.2: find the conservative variable $\tilde{\mathbf{u}}(\mathbf{x}) : \tilde{\Omega} \to \mathbb{R}^4$, with the restrictions to the subdomains $\tilde{\mathbf{u}}_k = \tilde{\mathbf{u}}_{|\tilde{\Omega}_k}$ such that for $k \in \{1, 2\}$

$$\begin{aligned}
\frac{\partial \tilde{\mathbf{u}}_k}{\partial t} + \tilde{\nabla} \cdot \left(\underline{\tilde{\boldsymbol{F}}}^c_k(\tilde{\mathbf{u}}_k) - \underline{\tilde{\boldsymbol{F}}}^d_k(\tilde{\mathbf{u}}_k, \tilde{\nabla}\tilde{\mathbf{u}}_k) \right) &= \mathbf{S}_k(\tilde{\mathbf{u}}_k, \tilde{\nabla}\tilde{\mathbf{u}}_k) && \forall \mathbf{x} \in \tilde{\Omega}_k\,, \\
\mathbf{BC}_{\mathrm{Dir},k}(\tilde{\mathbf{u}}_k) &= 0 && \forall \mathbf{x} \in \partial\tilde{\Omega}_{\mathrm{Dir},k}\,, \\
\mathbf{BC}_{\mathrm{Neu},k}(\tilde{\mathbf{u}}_k, \tilde{\nabla}\tilde{\mathbf{u}}_k) &= 0 && \forall \mathbf{x} \in \partial\tilde{\Omega}_{\mathrm{Neu},k}\,, \\
-\mathbf{v}_{n,\Gamma}[\![\tilde{\mathbf{u}}]\!] + [\![\underline{\tilde{\boldsymbol{F}}}^c(\tilde{\mathbf{u}}) - \underline{\tilde{\boldsymbol{F}}}^d(\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}})]\!] \cdot \mathbf{n}_\Gamma &= \mathbf{F}_\Gamma && \forall \mathbf{x} \in \tilde{\Gamma}\,, \\
[\![\tilde{v}_t]\!] &= \Delta_\Gamma v_t && \forall \mathbf{x} \in \tilde{\Gamma}\,, \\
[\![\tilde{T}]\!] &= \Delta_\Gamma T && \forall \mathbf{x} \in \tilde{\Gamma}\,, \\
\dot{m}_\Gamma := \tilde{\rho}_k(\tilde{v}_{n,k} - \tilde{v}_{n,\Gamma}) &= \dot{m}^{\mathrm{model}}(\tilde{u}_1, \tilde{u}_2) && \forall \mathbf{x} \in \tilde{\Gamma}\,,
\end{aligned} \tag{9.4}$$

where:

- $\mathbf{S}_k$ is a bulk source term,
- $\mathbf{BC}_{\mathrm{Dir},k}$ and $\mathbf{BC}_{\mathrm{Neu},k}$ represent the Dirichlet and Neumann boundaries conditions on the respective Dirichlet and Neumann parts, $\partial\tilde{\Omega}_{\mathrm{Dir},k}$ and $\partial\tilde{\Omega}_{\mathrm{Neu},k}$, of the subdomain boundary $\partial\tilde{\Omega}_k$,
- $\mathbf{F}_\Gamma$ includes the contributions from both the divergence of the line flux along the interface and the interface source

$$\mathbf{F}_\Gamma = \begin{bmatrix} 0 \\ \Delta_\Gamma p\, \mathbf{n}_\Gamma + \Delta_\Gamma \tau_{nt}\, \mathbf{t}_\Gamma \\ \Delta_\Gamma p\, \tilde{v}_{n,\Gamma} + \Delta_\Gamma \tau_{nt}\, \tilde{v}_{t,\Gamma} + \dot{q}_\Gamma \end{bmatrix}. \tag{9.5}$$



a / HLLC. **b /** HLLCP.

Figure 9.1 / Waves structure of the approximate two-phase Riemann solvers considered.

9.2.2 Interface convective numerical fluxes

The specification of the convective fluxes across the interface relies on the concept of multiphase Riemann solvers and reads

$$\widehat{\underline{\mathbf{F}}}^{c}_{\mathrm{I},k} \cdot \mathbf{n}_{\Gamma,m,k} = \mathcal{H}_{\mathrm{I},k}(\mathbf{u}_k, \mathbf{u}_{k^*}, \mathbf{n}_{\Gamma,m,k}) \quad \text{on } \mathrm{I}_m \,. \tag{9.6}$$

The difference with the Riemann solvers used at elements faces is that the multiphase solver considered here returns two distinct fluxes across the iso-0, i.e. $\mathcal{H}_{\mathrm{I},1} \neq \mathcal{H}_{\mathrm{I},2}$, such that the jumps are incorporated into the resulting fluxes. At the same time, the waves propagation structure associated to the hyperbolic part of the system is taken into account such as to provide upwinding to stabilize the non-linear convective effects. More precisely, two different approximate multiphase Riemann solvers, the HLLC and HLLCP, are considered here. Those have been shown to produce accurate results, with the advantages of being much faster and more robust than exact Riemann solvers, as pointed out by S. Fechter et al. (2013). They are both based on the extension of the classical single-phase HLLC solver. The waves structure from which they are built are sketched in Figure 9.1. Since the goal of the solvers is to solve (approximately) the one-dimensional Riemann problem along the normal to the interface, the jump conditions to consider are given by Equation 3.40, without the tangential momentum jump. An important remark should be made at this point. The jumps on the viscous stresses and heat fluxes across the interface are not taken into account for solving the Riemann problem. Indeed, including their contributions would lead to a system of equations that would not only depend on the states on both sides of the interface but also on their gradients. This would change the hyperbolic nature of the problem and would require to solve a generalized Riemann problem including both convective and viscous fluxes (see e.g. G. Gassner et al. (2007), Jöns and Claus-Dieter Munz (2020), and Jöns et al. (2023)). The approach followed here is thus similar to the one in Das and Udaykumar (2020), Houim and Kuo (2013), and Majidi and Afshari (2016) where the normal momentum jump is split into one pressure jump and one

Table 9.1 / The two strategies considered to split the jump conditions between convective and diffusive interface numerical fluxes.

	Splitting #1	Splitting #2
Convective fluxes	$[\![p]\!] = \Delta_\Gamma p - \dot{m}_\Gamma [\![v_n]\!]$ /	$[\![p]\!] = \Delta_\Gamma p - \dot{m}_\Gamma [\![v_n]\!]$ $\dot{m}_\Gamma [\![E]\!] + [\![p\, v_n]\!] = \Delta_\Gamma p\, v_{n,\Gamma}$
Diffusive fluxes	$[\![\tau_{nn}]\!] = 0$ $[\![q_n]\!] - [\![\tau_{nn}\, v_n]\!] - [\![\tau_{nt}\, v_t]\!]$ $= -\dot{m}_\Gamma [\![E]\!] - [\![p\, v_n]\!]$ $+ \Delta_\Gamma p\, v_{n,\Gamma} + \Delta_\Gamma \tau_{nt}\, v_{t,\Gamma} + \dot{q}_\Gamma$	$[\![\tau_{nn}]\!] = 0$ $[\![q_n]\!] - [\![\tau_{nn}\, v_n]\!] - [\![\tau_{nt}\, v_t]\!]$ $= \Delta_\Gamma \tau_{nt}\, v_{t,\Gamma} + \dot{q}_\Gamma$

jump on the normal viscous stress:

$$\dot{m}_\Gamma [\![v_n]\!] + [\![p]\!] - \Delta_\Gamma p = [\![\tau_{nn}]\!] \Rightarrow \begin{cases} [\![p]\!] = \Delta_\Gamma p - \dot{m}_\Gamma [\![v_n]\!]\,, \\ [\![\tau_{nn}]\!] = 0\,. \end{cases} \tag{9.7}$$

Moreover, if $\dot{m}_\Gamma \neq 0$, the energy jump without the diffusive contributions is not equivalent to the normal momentum jump and it could therefore be envisaged that this jump is also taken into account in the Riemann problem by splitting the energy jump in the following way:

$$\begin{aligned} \dot{m}_\Gamma [\![E]\!] + [\![p\, v_n]\!] - \Delta_\Gamma p\, v_{n,\Gamma} - \Delta_\Gamma \tau_{nt}\, v_{t,\Gamma} - \dot{q}_\Gamma = [\![\tau_{nn}\, v_n]\!] + [\![\tau_{nt}\, v_t]\!] - [\![q_n]\!] \\ \Rightarrow \begin{cases} \dot{m}_\Gamma [\![E]\!] + [\![p\, v_n]\!] = \Delta_\Gamma p\, v_{n,\Gamma}\,, \\ [\![q_n]\!] - [\![\tau_{nn}\, v_n]\!] - [\![\tau_{nt}\, v_t]\!] = \Delta_\Gamma \tau_{nt}\, v_{t,\Gamma} + \dot{q}_\Gamma\,, \end{cases} \end{aligned} \tag{9.8}$$

with the first condition handled by the Riemann solver and the second by the diffusive flux functions. Because the choice of the Riemann solver (HLLC VS HLLCP) determines the type of splitting, thus also the conditions to be imposed by the diffusive numerical flux functions (discussed in Section 9.2.3), Table 9.1 compiles the two possibilities considered in this work. Both solvers follow the same global steps to compute the numerical fluxes. The procedure is basically the same as the one described in Stefan Fechter et al. (2018) and it is here restated for completeness. Given $\mathbf{u}_1(\boldsymbol{\xi}_{\mathrm{QP}}) = \mathbf{u}_1$ and $\mathbf{u}_2(\boldsymbol{\xi}_{\mathrm{QP}}) = \mathbf{u}_2$, the solution vectors from both phases interpolated at a given quadrature point with parametric coordinates $\boldsymbol{\xi}_{\mathrm{QP}}$ on the iso-0 contour of the level-set, the normal $\mathbf{n}^\Gamma(\boldsymbol{\xi}_{\mathrm{QP}}) = \mathbf{n}^\Gamma$ and the tangent $\mathbf{t}^\Gamma(\boldsymbol{\xi}_{\mathrm{QP}}) = \mathbf{t}^\Gamma$ vectors computed from the level-set, we follow those steps:

- *Step 1*: Project $\mathbf{u}_1$ and $\mathbf{u}_2$ onto the interface-attached frame and assemble

the vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ as:

$$\mathbf{V}_k = \left(\rho_k\,,\, v_{n,k}\,,\, e_k\,,\, p_k\right)^T\,,$$

with $v_{n,k} = \mathbf{v}_k \cdot \mathbf{n}_\Gamma$.

- *Step 2*: Compute an approximation of the two external shock wave speeds, S_1 and S_2, based on the linearization of the Lax curves as proposed by S. Fechter et al. (2013). The tangent to the wave curves in terms of the variables $\tilde{\mathbf{V}}_k = (\tilde{\rho}_k, \tilde{v}_{n,k}, \tilde{p}_k)$ can be expressed as

$$\tilde{\mathbf{V}}_1 = \begin{bmatrix} \rho_1 \\ v_{n,1} \\ p_1 \end{bmatrix} + \eta_1 \begin{bmatrix} 1 \\ -c_1/\rho_1 \\ c_1^2 \end{bmatrix}\,, \quad \tilde{\mathbf{V}}_2 = \begin{bmatrix} \rho_2 \\ v_{n,2} \\ p_2 \end{bmatrix} + \eta_2 \begin{bmatrix} 1 \\ c_2/\rho_2 \\ c_2^2 \end{bmatrix}\,, \quad [9.9]$$

where $c_k = c_k(\mathbf{V}_k)$ is the speed of sound in phase k obtained from an appropriate thermodynamic model and η_k are the slopes given by

$$\begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix} = \frac{1}{c_1 c_2 (c_2/\rho_1 + c_1/\rho_2)} \begin{bmatrix} c_2^2 & -c_2/\rho_2 \\ c_1^2 & c_1/\rho_1 \end{bmatrix} \begin{bmatrix} v_{n,1} - v_{n,2} - \Delta_\Gamma v_n \\ p_1 - p_2 - \Delta_\Gamma p \end{bmatrix}, \quad [9.10]$$

with the interface jumps on normal velocity given by Equation 3.41a, $\Delta_\Gamma v_n$, and pressure, $\Delta_\Gamma p$. The estimation of the shock wave speeds is finally obtained by

$$\begin{aligned} S_0 &= \min\left(v_{n,1} - c_1, \tilde{v}_{n,1} - \tilde{c}_1\right)\,, \\ S_1 &= \max\left(v_{n,2} + c_2, \tilde{v}_{n,2} + \tilde{c}_2\right)\,, \end{aligned} \quad [9.11]$$

with $\tilde{c}_k = c_k(\tilde{\mathbf{V}}_k)$.

- *Step 3*: Solve the Riemann problem to obtain the solution states $\mathbf{V}_1^*$ and $\mathbf{V}_2^*$, and the interface normal velocity $v_{n,\Gamma}$:
  - *HLLC*: Solve Equation 9.13.
  - *HLLCP*: Solve Equation 9.14, including Equation 9.15 when $\dot{m}_\Gamma = 0$.

- *Step 4*: Re-project the interface aligned solution states to the reference frame and assemble the set of conservative variables $\mathbf{u}_1^*$ and $\mathbf{u}_2^*$:

$$\begin{aligned} \mathbf{v}_k^* &= v_{n,k}^* \mathbf{n}_\Gamma + v_{t,k}^* \mathbf{t}^\Gamma\,, \quad v_{t,k}^* = \mathbf{v}_k \cdot \mathbf{t}^\Gamma\,, \\ \mathbf{u}_k^* &= \rho_k^* \left(1\,,\, \mathbf{v}_k^*\,,\, \mathbf{e}_k^* + 1/2 \mathbf{v}_k^* \cdot \mathbf{v}_k^*\right)^T\,. \end{aligned}$$

An important remark is that the tangential component of the velocity, $v_{t,k}^*$, is set equal to the tangential velocity from the external state. No jump is therefore imposed on the tangential velocity through the interface convective fluxes, it is the role of the interface diffusive fluxes covered in the next section.

- *Step 5*: Compute the numerical fluxes normal to the interface:

$$\mathcal{H}_{\Gamma,1} = \begin{cases} \underline{\boldsymbol{F}}_1^c(\mathbf{u}_1)\cdot\mathbf{n}_\Gamma & S_1 > 0 \\ \underline{\boldsymbol{F}}_1^c(\mathbf{u}_1)\cdot\mathbf{n}_\Gamma + S_1(\mathbf{u}_1^* - \mathbf{u}_1) & S_1 \leq 0 \;\;\&\;\; v_{n,\Gamma} > 0 \\ \underline{\boldsymbol{F}}_1^c(\mathbf{u}_1)\cdot\mathbf{n}_\Gamma + S_1(\mathbf{u}_1^* - \mathbf{u}_1) & S_2 > 0 \;\;\&\;\; v_{n,\Gamma} \leq 0 \\ \underline{\boldsymbol{F}}_1^c(\mathbf{u}_1)\cdot\mathbf{n}_\Gamma + S_1(\mathbf{u}_1^* - \mathbf{u}_1) & S_2 \leq 0\,, \end{cases}$$

$$\mathcal{H}_{\Gamma,2} = \begin{cases} \underline{\boldsymbol{F}}_2^c(\mathbf{u}_2)\cdot\mathbf{n}_\Gamma + S_2(\mathbf{u}_2^* - \mathbf{u}_2) & S_1 > 0 \\ \underline{\boldsymbol{F}}_2^c(\mathbf{u}_2)\cdot\mathbf{n}_\Gamma + S_2(\mathbf{u}_2^* - \mathbf{u}_2) & S_1 \leq 0 \;\;\&\;\; v_{n,\Gamma} > 0 \\ \underline{\boldsymbol{F}}_2^c(\mathbf{u}_2)\cdot\mathbf{n}_\Gamma + S_2(\mathbf{u}_2^* - \mathbf{u}_2) & S_2 > 0 \;\;\&\;\; v_{n,\Gamma} \leq 0 \\ \underline{\boldsymbol{F}}_2^c(\mathbf{u}_2)\cdot\mathbf{n}_\Gamma & S_2 \leq 0\,. \end{cases}$$

- *Step 6* (*optional*): in case the "moving interface" time integration procedure (see Section 10.2.2) is selected, the convective flux due to the motion of the interface is taken into account as

$$\begin{aligned} \mathcal{H}_{\Gamma,1} &-= v_{n,\Gamma}\,\mathbf{u}_1^*\,, \\ \mathcal{H}_{\Gamma,2} &-= v_{n,\Gamma}\,\mathbf{u}_2^*\,. \end{aligned} \qquad [9.12]$$

HLLC multiphase Riemann solver

The two-phase HLLC multiphase solver implemented in the solver is based on the one presented in Stefan Fechter (2015, Sec. 6.3.2.1). It is here extended by including the possibility to consider non-zero mass flux across the interface like in Jöns and Claus-Dieter Munz (2022), i.e. considering a discontinuity in the interface normal velocities and the recoil pressure effect in the pressure jump. Because the structure of the HLLC solver has three waves, there are too few equations to be able to simultaneously impose the jumps on the normal momentum and total energy (if $\dot{m}_\Gamma \neq 0$). This solver is therefore part of the first splitting strategy (Table 9.1). Combining the Rankine-Hugoniot relations across the two external shocks with the interface jumps, the system to solve reads:

$$\begin{aligned} S_1(\rho_1^* - \rho_1) &= \rho_1^* v_{n,1}^* - \rho_1 v_{n,1} \\ S_1(\rho_1^* v_{n,1}^* - \rho_1 v_{n,1}) &= \rho_1^* v_{n,1}^{*,2} - \rho_1 v_{n,1}^2 + p_1^* - p_1 \\ S_1(\rho_1^* E_1^* - \rho_1 E_1) &= \rho_1^* v_{n,1}^* E_1^* - \rho_1 v_{n,1} E_1 + p_1^* v_{n,1}^* - p_1 v_{n,1} \\ S_2(\rho_2^* - \rho_2) &= \rho_2^* v_{n,2}^* - \rho_2 v_{n,2} \\ S_2(\rho_2^* v_{n,2}^* - \rho_2 v_{n,2}) &= \rho_2^* v_{n,2}^{*,2} - \rho_2 v_{n,2}^2 + p_2^* - p_2 \\ S_2(\rho_2^* E_2^* - \rho_2 E_2) &= \rho_2^* v_{n,2}^* E_2^* - \rho_2 v_{n,2} E_2 + p_2^* v_{n,2}^* - p_2 v_{n,2} \\ v_{n,1}^* - v_{n,2}^* &= \dot{m}_\Gamma \left(1/\rho_1^* - 1/\rho_2^*\right) \\ p_1^* - p_2^* &= \Delta_\Gamma p - \dot{m}_\Gamma^2 \left(1/\rho_1^* - 1/\rho_2^*\right) \\ \dot{m}_\Gamma &= \dot{m}_\Gamma^{\text{model}}(\mathbf{V}_1^*, \mathbf{V}_2^*) \end{aligned} \qquad [9.13]$$

The last equation corresponds to Equation 3.41d for the closure of the interface mass flux. The strategy to reduce this system to a single non-linear equation is as follows. The quantities ρ_k^*, p_k^* and e_k^* are first expressed as a function of $v_{n,k}^*$ through the Rankine-Hugoniot relations (the first six equations). They are then plugged into the mass and normal momentum jumps to express $v_{n,1}^*$ and $v_{n,2}^*$ as a function of $\dot{m}_\Gamma$ and the known $\mathbf{V}_k$ states. The last equation for the mass flux model is then solved by a Newton's method with the $\mathbf{V}_k^*$ states now expressed entirely based on $\dot{m}_\Gamma$.

HLLCP multiphase Riemann solver

This second solver considers, in addition to the contact discontinuity and the two shock waves, a fourth non-classical wave modeling the phase interface. The distinction between the contact (C) and phase (P) discontinuities has physical roots: during phase transition, a Knudsen layer containing the newly created vapor is formed, separating the liquid and gas equilibrium regions. This is considered in Stefan Fechter (2015, Sec. 6.4.6.2) and Long et al. (2021a). With the structure of the HLLCP solver, it is possible to take into account the energy balance across the phase interface. The use of this solver is therefore linked to the second splitting strategy (Table 9.1). If it is assumed, as in Figure 9.1b, that the phase interface moving with a normal velocity $v_{n,\Gamma}$ is located between the left star and the middle states, $\mathbf{V}_1^*$ and $\mathbf{V}_M^*$, while the contact discontinuity with speed S_C separates the middle and the right state $\mathbf{V}_2^*$, then the equations to solve are:

$$S_1(\rho_1^* - \rho_1) = \rho_1^* v_{n,1}^* - \rho_1 v_{n,1} \tag{9.14a}$$
$$S_1(\rho_1^* v_{n,1}^* - \rho_1 v_{n,1}) = \rho_1^* v_{n,1}^{*,2} - \rho_1 v_{n,1}^2 + p_1^* - p_1 \tag{9.14b}$$
$$S_1(\rho_1^* E_1^* - \rho_1 E_1) = \rho_1^* v_{n,1}^* E_1^* - \rho_1 v_{n,1} E_1 + p_1^* v_{n,1}^* - p_1 v_{n,1} \tag{9.14c}$$
$$S_2(\rho_2^* - \rho_2) = \rho_2^* v_{n,2}^* - \rho_2 v_{n,2} \tag{9.14d}$$
$$S_2(\rho_2^* v_{n,2}^* - \rho_2 v_{n,2}) = \rho_2^* v_{n,2}^{*,2} - \rho_2 v_{n,2}^2 + p_2^* - p_2 \tag{9.14e}$$
$$S_2(\rho_2^* E_2^* - \rho_2 E_2) = \rho_2^* v_{n,2}^* E_2^* - \rho_2 v_{n,2} E_2 + p_2^* v_{n,2}^* - p_2 v_{n,2} \tag{9.14f}$$
$$v_{n,M}^* = v_{n,2}^* \tag{9.14g}$$
$$p_M^* = p_2^* \tag{9.14h}$$
$$\rho_1^*(v_{n,1}^* - v_{n,\Gamma}) = \rho_M^*(v_{n,M}^* - v_{n,\Gamma}) \tag{9.14i}$$
$$p_1^* - p_M^* - \Delta_\Gamma p = -\dot{m}_\Gamma(v_{n,1}^* - v_{n,M}^*) \tag{9.14j}$$
$$p_1^* v_{n,1}^* - p_M^* v_{n,M}^* - \Delta_\Gamma p v_{n,\Gamma} = -\dot{m}_\Gamma \left[e_1^* + 1/2\,(v_{n,1}^{*,2} + v_{t,1}^2) - e_M^* - 1/2\,(v_{M,1}^{*,2} + v_{t,2}^2) \right] \tag{9.14k}$$
$$S_C = v_{n,M}^* \tag{9.14l}$$
$$\dot{m}_\Gamma = \rho_1^*(v_{n,1}^* - v_{n,\Gamma}) \tag{9.14m}$$
$$\dot{m}_\Gamma = \dot{m}_\Gamma^{\text{model}}(\mathbf{V}_1^*, \mathbf{V}_M^*) \tag{9.14n}$$

$$\frac{\rho_1^*}{\rho_M^*} \approx \frac{\rho_1}{\rho_2} \tag{9.140}$$

Equation 9.14a to Equation 9.14f are the two Rankine-Hugoniot relations across the two external shock waves, Equation 9.14g and Equation 9.14h are the equality of normal velocity and pressure across the contact discontinuity, Equation 9.14i to Equation 9.14k are the interface jump conditions (without the diffusive contributions), Equation 9.14l is a direct consequence of the equality of normal velocity across the contact discontinuity, Equation 9.14m is the definition of the interface mass flux and Equation 9.14n corresponds to Equation 3.41d for the closure for the interface mass flux. This represents 14 equations for 15 unknowns: $\mathbf{V}_1^*\,(4)$, $\mathbf{V}_2^*\,(4)$, $\mathbf{V}_M^*\,(4)$, $v_{n,\Gamma}$ and S_C. There are two possibilities at this stage. The first one is to solve this non-linear underdetermined system of equations with an iterative procedure, which can be quite costly. The second one is to introduce an additional equation, which will in any case come from an assumption as all the physical relations have already been used in the 14 equations. This is the choice made here with the introduction of Equation 9.140 based on Long et al. (2021a). It can be noticed that the implementation in the solver allows also to treat the situation where the phase interface and the contact discontinuity are permuted by checking the sign of $\dot{m}_\Gamma$. The procedure described in Long et al. (2021a, App. A) to reduce the system to a single non-linear equation is followed: all the unknowns are expressed as a function of $\dot{m}_\Gamma$ and the non-linear equation to solve is given by Equation 9.14n. At the end of the procedure, $\mathbf{V}_2^* = \mathbf{V}_M^*$ if $\dot{m}_\Gamma \geq 0$ or $\mathbf{V}_1^* = \mathbf{V}_M^*$ if $\dot{m}_\Gamma < 0$ are set. The interface convective fluxes can then be assembled at the step 5 explained above.

Although this solver is mostly designed to deal with phase transition problems, the situation of vanishing $\dot{m}_\Gamma$ can also be treated. In this case, the phase interface and contact discontinuity coincide. For such cases, a modification to the HLLCP solver of Long et al. (2021a) is proposed here. When $\dot{m}_\Gamma^{\text{model}} = 0$, the mass jump Equation 9.14i and the energy jump Equation 9.14k become redundant. The idea is to replace those two equations by two other ones. Those new equations should ensure that the temperature returned by the solver satisfies the kinematic condition Equation 3.41c. Indeed, the HLLC solver and the original HLLCP solver described above do not take into account this condition on the temperature jump. This means that the interface convective fluxes that are assembled at the end do not incorporate it either. As this condition is anyway enforced as a Dirichlet jump condition by the interface diffusive fluxes, the global procedure should still converge toward a solution that satisfies the interface temperature jump. But to improve the consistency between the convective and diffusive fluxes, a strategy to impose the temperature jump inside the Riemann solver is pursued. This is achieved by replacing Equation 9.14i and Equation 9.14k by a 2×2 internal non-linear system to be solved for the unknowns ρ_M^* and e_M^* such

that the jumps on temperature and pressure are simultaneously satisfied:

$$\begin{cases} T_1^*(e_1^*, p_1^*) - T_M^*(e_M^*, p_M^*) = \Delta_\Gamma T\,, \\ p_1^*(\rho_1^*, e_1^*) - p_M^*(\rho_M^*, e_M^*) = \Delta_\Gamma p\,, \end{cases} \tag{9.15}$$

where thermodynamic relations specific to each phase are used to compute the temperatures and pressures as a function of other quantities.

9.2.3 Interface diffusive numerical fluxes

The way to incorporate the Dirichlet jumps on the solutions and the Robin jumps on the diffusive fluxes is inspired by the strategies developed for incompressible two-phase flows, e.g. in Benedikt Schott (2017, Sec. 4.2) and R. Saye (2020, App. A): when considering the assembling of the interface term for a given element, the solution, resp. the flux, on the other side of the interface is compensated by the jump condition to correctly account for the intended discontinuity in the solution, resp. the flux. In addition, those convex combinations between both sides are weighted by a parameter α defined by

$$\alpha_k = \frac{(q_k)^a}{(q_1)^a + (q_2)^a}\,, \quad \alpha_{k^*} = 1 - \alpha_k\,, \tag{9.16}$$

where a is a user-defined power ($a = 1$ gives the well-known harmonic weighting) and q is a quantity related to the diffusion process. Here we set $q_k = 2\mu_k + \lambda_k / C_{v,k}$. Biasing the numerical flux toward one phase or the other, depending on the rate of diffusion of the interface phenomena involved, may both improve solution accuracy and conditioning of the system according to Burman and Zunino (2011).

Numerical flux function for the direct term

The numerical flux function $\underline{\hat{\boldsymbol{F}}}^d_{\mathrm{I},k}$ in Equation 8.8 is specified based on the Interior Penalty (IP) method as

$$\underline{\hat{\boldsymbol{F}}}^d_{\mathrm{I},k}(\mathbf{u}_k, \nabla\mathbf{u}_k, \mathbf{u}_{k^*}, \nabla\mathbf{u}_{k^*}) = \alpha_{k^*}\underline{\boldsymbol{F}}^d_k + \alpha_k\left(\underline{\boldsymbol{F}}^d_{k^*} + \mathbf{J}^{d,\mathrm{Rob}} \otimes \mathbf{n}_{\Gamma,k}\right) - \underline{\boldsymbol{\delta}}_{\mathrm{I},k}\,, \tag{9.17}$$

where $\mathbf{J}^{d,\mathrm{Rob}}$ is a vector incorporating the Robin jump conditions on the fluxes whose explicit expression is given just after. The focus here is on the two first terms, the third term (penalty term) is discussed in Section 9.2.3. The role of two first terms is to take into account via a multivalued interfacial flux the physical jumps prescribed on the diffusive fluxes. It can be noticed the multiplication of $\mathbf{J}^{d,\mathrm{Rob}}$ by the normal $\mathbf{n}_{\Gamma,k}$ associated to phase k such that the jumps values are either added to or subtracted from the diffusive fluxes in the other phase. The term $\mathbf{J}^{d,\mathrm{Rob}}$ embodies the non-linear Robin jump conditions to be imposed and its expression depends on the splitting strategy (Table 9.1) adopted. In both

cases, the condition $[\![\tau_{nn}]\!] = 0$ is taken into account to reformulate the x- and y-momentum jumps of Equation 3.38. Depending on the splitting, the energy jump differs so that:

$$\mathbf{J}^{d,\mathrm{Rob}}(\mathbf{u}_1,\mathbf{u}_2) = -\begin{cases} \begin{bmatrix} 0 \\ \dot{m}_\Gamma\,[\![v_t]\!]\,t_x^\Gamma - \Delta_\Gamma \tau_{nt}\,t_x^\Gamma \\ \dot{m}_\Gamma\,[\![v_t]\!]\,t_y^\Gamma - \Delta_\Gamma \tau_{nt}\,t_y^\Gamma \\ \dot{m}_\Gamma\,[\![E]\!] + [\![p\,v_n]\!] - \Delta_\Gamma p\,v_{n,\Gamma} - \Delta_\Gamma \tau_{nt}\,v_{t,\Gamma} - \dot{q}_\Gamma \end{bmatrix} & \text{if \#1}\,, \\ \begin{bmatrix} 0 \\ \dot{m}_\Gamma\,[\![v_t]\!]\,t_x^\Gamma - \Delta_\Gamma \tau_{nt}\,t_x^\Gamma \\ \dot{m}_\Gamma\,[\![v_t]\!]\,t_y^\Gamma - \Delta_\Gamma \tau_{nt}\,t_y^\Gamma \\ -\Delta_\Gamma \tau_{nt}\,v_{t,\Gamma} - \dot{q}_\Gamma \end{bmatrix} & \text{if \#2}\,. \end{cases} \tag{9.18}$$

The negative sign in front of the bracket stands from the fact that the jumps need to be imposed on the opposite of the diffusive fluxes (see Table 9.1).
In the most general case, the jumps on the diffusive fluxes could depend both on the solutions and their gradients. This is the case for example if a jump on the heat flux has to be prescribed. In this situation, the function $\mathbf{J}^{d,\mathrm{Rob}}$ must compensate in the energy balance the contributions from the viscous dissipation terms since they are included in the diffusive fluxes $\underline{\boldsymbol{F}}_k^d$ by definition but the goal in this example is to impose a jump solely on the heat fluxes. This yields

$$J^{d,\mathrm{Rob}}_{\mathrm{energy}}(\mathbf{u}_1,\nabla\mathbf{u}_1,\mathbf{u}_2,\nabla\mathbf{u}_2) = -\begin{cases} -[\![\underline{\boldsymbol{\tau}}\cdot\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma + \dot{m}_\Gamma\,[\![E]\!] + [\![p\,v_n]\!] \\ \qquad -\Delta_\Gamma p\,v_{n,\Gamma} - \Delta_\Gamma \tau_{nt}\,v_{t,\Gamma} - \dot{q}_\Gamma \end{cases} \text{if \#1}\,, \\ -[\![\underline{\boldsymbol{\tau}}\cdot\mathbf{v}]\!]\cdot\mathbf{n}_\Gamma - \Delta\tau_{nt}\,v_{t,\Gamma} - \dot{q}_\Gamma \qquad \text{if \#2}\,. \tag{9.19}$$

The solver is designed to treat the most general case of $\mathbf{J}^{d,\mathrm{Rob}}$ depending both on the solutions and their gradients coming from both phases.

Numerical flux function for the transpose term

The numerical flux function $\hat{\mathbf{u}}_{\mathrm{I},k}$ in Equation 8.8 is specified as

$$\hat{\mathbf{u}}_{\mathrm{I},k}(\mathbf{u}_k,\mathbf{u}_{k^*}) = (1-\theta\alpha_{k^*})\mathbf{u}_k + \theta\alpha_{k^*}\mathbf{u}\mathbf{J}_k\,, \tag{9.20}$$

with

$$\mathbf{u}\mathbf{J}_k(\mathbf{u}_k,\mathbf{u}_{k^*}) := \left(\mathbf{u}_{k^*} + t_k\mathbf{J}^{d,\mathrm{Dir}}\right)\,, \tag{9.21}$$

where t_k is given in Equation 9.3, the choice of θ selects the symmetric (SIPG) or non-symmetric (NIPG) interior penalty method ($\theta = 1$ for SIPG, $\theta = -1$ for NIPG) and $\mathbf{J}^{d,\mathrm{Dir}}$ is a vector incorporating the Dirichlet jump conditions on the solutions whose explicit expression is given just after. The flux function is also multivalued such that when the trace of the solution on the other side of the interface is evaluated, this trace is compensated by the jump data, either posi-

tively or negatively depending on the sign of t_k, which results in the imposition of the prescribed jumps on the solutions. The term $\mathbf{J}^{d,\mathrm{Dir}}$ should enclose the jumps on the solutions. The distinction should be made here between schemes that employ the primitive p–$\mathbf{v}$–T variables set, denoted $\mathbf{u}_k^{\mathrm{prim}}$, and the ones that solve for another variables set, e.g. the conservative variables $\mathbf{u}_k$, which will be labelled as $\mathbf{u}_k^{\mathrm{cons}}$ for the sake of clarity. There exists a correspondence between both sets through transformations $u_k^{\mathrm{p}\nrightarrow\mathrm{c}}$ and $u_k^{\mathrm{c}\nrightarrow\mathrm{p}}$ based on the equations of state associated to each phase k as explained in Section 5.3.2:

$$\begin{aligned} \mathbf{u}_k^{\mathrm{prim}} &= u_k^{\mathrm{c}\nrightarrow\mathrm{p}}(\mathbf{u}_k^{\mathrm{cons}}) \qquad \text{Conservative to primitive sets}\,, \\ \mathbf{u}_k^{\mathrm{cons}} &= u_k^{\mathrm{p}\nrightarrow\mathrm{c}}(\mathbf{u}_k^{\mathrm{prim}}) \qquad \text{Primitive to conservative sets}\,. \end{aligned} \tag{9.22}$$

When the solver solves for $\mathbf{u}_k^{\mathrm{prim}}$, the term $\mathbf{J}^{d,\mathrm{Dir}}$ can readily be assembled by combining Equation 9.7, Equation 3.39a, Equation 3.39a and Equation 3.39c:

$$\mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{prim}}(\mathbf{u}_1^{\mathrm{prim}}, \mathbf{u}_2^{\mathrm{prim}}) = \begin{bmatrix} \Delta_\Gamma p - \dot{m}_\Gamma [\![v_n]\!] \\ \dot{m}_\Gamma \, [\![1/\rho]\!] \, n_{x,\Gamma} + \Delta_\Gamma v_t \, t_{x,\Gamma} \\ \dot{m}_\Gamma \, [\![1/\rho]\!] \, n_{y,\Gamma} + \Delta_\Gamma v_t \, t_{y,\Gamma} \\ \Delta_\Gamma T \end{bmatrix} \text{ if primitive variables set}\,. \tag{9.23}$$

When a set of variables different from p–$\mathbf{v}$–T is used, there exists no explicit and unique expression to be assigned to $\mathbf{J}^{d,\mathrm{Dir}}$ because since both phases can be governed by different thermodynamic models, it is not possible to convert $\mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{prim}}$ with a transformation $u_k^{\mathrm{p}\nrightarrow\mathrm{c}}$. The approach chosen here is to convert the conservative sets to p–$\mathbf{v}$–T primitive ones for both phases, add or subtract from those vectors the known $\mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{prim}}$ vector, convert the results back to conservative variables and set $\mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{cons}}$ as being equal to the jumps between those compensated conservative sets:

$$\begin{aligned} \mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{cons}} &= u_1^{\mathrm{p}\nrightarrow\mathrm{c}}(\mathbf{u}_1^{\mathrm{prim}}) - u_2^{\mathrm{p}\nrightarrow\mathrm{c}}(\mathbf{u}_2^{\mathrm{prim}}) \\ &= u_1^{\mathrm{p}\nrightarrow\mathrm{c}}(\mathbf{u}_2^{\mathrm{prim}} + \mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{prim}}) - u_2^{\mathrm{p}\nrightarrow\mathrm{c}}(\mathbf{u}_1^{\mathrm{prim}} - \mathbf{J}^{d,\mathrm{Dir}}_{\mathrm{prim}}) \quad \text{if conservative var. set}\,. \end{aligned} \tag{9.24}$$

Both the conservative and p–$\mathbf{v}$–T primitive variables sets are tested in this work.

Nitsche's penalization term

The last numerical flux function that remains to be specified is the penalty term in Equation 9.17. The idea of the IP method is that instead of a rough and direct imposition, this term is consistently added to the discretization to minimize the average interpolation errors between the solutions on either sides of the interface. Similarly to the penalty in the bulk, the penalization term is either scaled with a scalar diffusivity or with a tensorial one according to Ralf

Hartmann and Houston (2008):

$$\underline{\boldsymbol{\delta}}_{\mathrm{I},k}(\mathbf{u}_k, \mathbf{u}_{k^*}) = \gamma_{\mathrm{I}} \begin{cases} \beta_{\mathrm{I}} \left[\mathbf{u}_k - \mathbf{u}\mathbf{J}_k\right] \otimes \mathbf{n}_{\Gamma,k}\,, & \text{standard IP,} \\ 0.5 \left[\underline{\boldsymbol{G}}_k(\mathbf{u}_k) + \underline{\boldsymbol{G}}_{k^*}(\mathbf{u}_{k^*})\right] \left[\mathbf{u}_k - \mathbf{u}\mathbf{J}_k\right] \otimes \mathbf{n}_{\Gamma,k}\,, & \text{tensorial IP,} \end{cases} \tag{9.25}$$

where $\mathbf{u}\mathbf{J}_k$ is given by Equation 9.21 and the diffusivity scaling β_{I} in the scalar case is computed from a weighted average, effectively scaling the penalty parameters with the minimum diffusion coefficient between the two phases $\beta_{\mathrm{I}} = \alpha_2\,\beta_1 + \alpha_1\,\beta_2$. The expression of the stabilization parameter γ_{I} is crucial to ensure the well-posedness of the discrete problem. The parameter at the interface takes a similar expression as in the bulk,

$$\gamma_{\mathrm{I}} = \max\left(\frac{s \times C_{IP}}{h_1}\,,\ \frac{s \times C_{IP}}{h_2}\right), \tag{9.26a}$$

$$C_{IP} = \begin{cases} (2p+1)(2p+2)/2 & \text{for 2-D tri}\,, \\ (2p+1)^2 & \text{for 2-D quad}\,, \end{cases} \tag{9.26b}$$

$$h_k = \frac{|\breve{\mathrm{T}}_{m,k}|}{|\partial\breve{\mathrm{T}}_{m,k}|} = \frac{\text{“Agglomerated element surface in phase k”}}{\text{“Agglomerated face length in phase k”}}\,. \tag{9.26c}$$

When tensorial IP is used, a safety factor of $s = 2$ was found to produce convergent results in all the different test cases considered as part of this work. When scalar penalty is used, this factor has to be increased. Rather than relying on the heuristic approach adopted here, we can mention the recent work by Levaux et al. (2023) presenting a rigorous framework to study the trace inverse inequality for cut-cells. Compared to bulk penalty, the fundamental difference is on the choice of the geometric factor h_k which follows the reasoning in Kummer (2017). To avoid stability issues with small cut cells, the ratio between element area and perimeter is here computed from the agglomerated topology, i.e. by using the ratio between agglomerated element area in phase k, noted $|\breve{\mathrm{T}}_{m,k}|$, and its perimeter, $|\partial\breve{\mathrm{T}}_{m,k}|$. For an element T_m which is a target element associated to a set of source elements $\mathrm{T}_m^{\mathrm{src}}$, the agglomerated area in phase k is computed as

$$|\breve{\mathrm{T}}_{m,k}| = |\mathrm{T}_m \cap \Omega_k| + \sum_{j \in \mathrm{T}_m^{\mathrm{src}}} |\mathrm{T}_j \cap \Omega_k|\,. \tag{9.27}$$

The perimeter is the sum of the cut edges lengths (contained in phase k) defining the target and the associated source elements and the lengths of the interface segments crossing the target and the sources. From this sum, the contributions of the set of cut edges which are shared between the target and the sources, noted as $\mathrm{E}_m^{\mathrm{shared}}$, are removed to obtain the perimeter of the agglomerated element:

$$|\partial\breve{\mathrm{T}}_{m,k}| = |\partial\mathrm{T}_m \cap \Omega_k| + |\mathrm{I}_m| + \sum_{j \in \mathrm{T}_m^{\mathrm{src}}} \left(|\partial\mathrm{T}_j \cap \Omega_k| + |\mathrm{I}_j|\right) - \sum_{l \in \mathrm{E}_m^{\mathrm{shared}}} |\mathrm{E}_l \cap \Omega_k|\,. \tag{9.28}$$

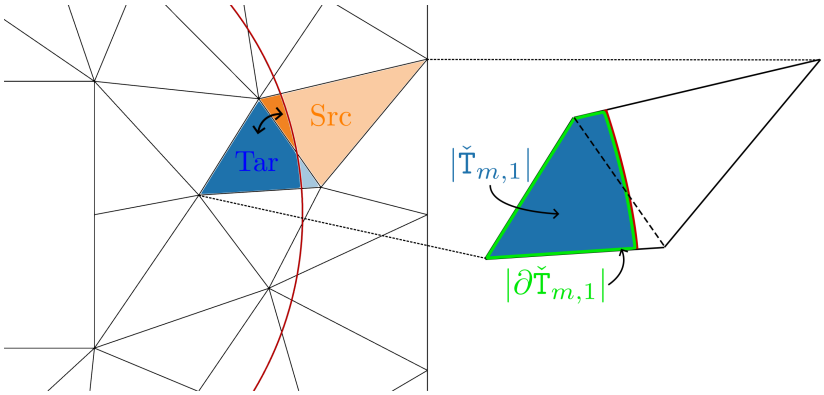


Figure 9.2 / Illustration of the computation of the geometric factor h of the penalization term for an agglomerated cells pair in phase 1.

This is illustrated in Figure 9.2. It can be noticed that the quadrature rules to compute the cut areas and the lengths of the interface are available from the construction of the cut-cell quadratures.

9.3 Resulting discretization implemented and its analysis

9.3.1 Resulting discretization

Now that we have seen the specification of the numerical flux functions for all the terms involved in the XDG weak form, we gather here the expressions of the face, boundary and interface terms in order to analyze the final form of the discretization. First, the commonly used mean and jump operators for vector-valued functions $\mathbf{q}$ and tensor-valued functions $\underline{\boldsymbol{q}}$ are introduced, starting by the element faces:

$$\begin{aligned} \{\{\underline{\boldsymbol{q}}\}\} &= \frac{1}{2}(\underline{\boldsymbol{q}}^+ + \underline{\boldsymbol{q}}^-) && \text{on } \mathrm{E}, \\ \underline{[\![\mathbf{q}]\!]} &= \mathbf{q}^+ \otimes \mathbf{n}_e^+ + \mathbf{q}^- \otimes \mathbf{n}_e^- && \text{on } \mathrm{E}. \end{aligned} \qquad [9.29]$$

These notations follow the ones in Ralf Hartmann and Houston (2008). On the phase interface, specific means operators which include the weighting of the

diffusive fluxes in Equation 9.16 are also defined:

$$\begin{aligned}
\{\{\mathbf{q}\}\}_{\alpha} &= \alpha_1\,\mathbf{q}_1 + \alpha_2\,\mathbf{q}_2 && \text{on } \mathtt{I}\,,\\
\{\{\underline{\boldsymbol{q}}\}\}_{\alpha^*} &= \alpha_2\,\underline{\boldsymbol{q}}_1 + \alpha_1\,\underline{\boldsymbol{q}}_2 && \text{on } \mathtt{I}\,,\\
\{\{\underline{\boldsymbol{q}}\}\} &= \frac{1}{2}(\underline{\boldsymbol{q}}_1 + \underline{\boldsymbol{q}}_2) && \text{on } \mathtt{I}\,,\\
\underline{[\![\mathbf{q}]\!]} &= \mathbf{q}_1 \otimes \mathbf{n}_{\Gamma,m,1} + \mathbf{q}_2 \otimes \mathbf{n}_{\Gamma,m,2} && \text{on } \mathtt{I}\,.
\end{aligned} \tag{9.30}$$

Face term The face term Equation 5.21 is entirely specified[1] by combining it with Equation 5.26, Equation 5.28, Equation 5.29 and Equation 5.31 to obtain, for the tensorial penalty version:

$$\begin{aligned}
\mathcal{E}_{e,k}(\mathbf{u}^+_{m(e),k}, \mathbf{u}^-_{m(e),k}, \mathbf{w}^+_{m(e),k}, \mathbf{w}^-_{m(e),k}) = \int_{\mathtt{E}_e\cap\Omega_k} & \mathcal{H}_k(\mathbf{u}^+_{e,k}, \mathbf{u}^-_{e,k}, \mathbf{n}_e)\underline{[\![\mathbf{w}_{e,k}]\!]}\cdot\mathbf{n}_e\\
& - \{\{\underline{\boldsymbol{F}}^d_k(\mathbf{u}_{e,k}, \nabla\mathbf{u}_{e,k})\}\} : \underline{[\![\mathbf{w}_{e,k}]\!]}\\
& + \gamma_{\mathtt{E}}\,\underline{[\![\mathbf{u}_{e,k}]\!]}\{\{\underline{\boldsymbol{G}}^T_k(\mathbf{u}_{e,k})\}\} : \underline{[\![\mathbf{w}_{e,k}]\!]}\,\mathrm{d}s\\
& - \theta\int_{\mathtt{E}_e\cap\Omega_k} \underline{[\![\mathbf{u}_{e,k}]\!]} : \{\{\underline{\boldsymbol{G}}^T_k(\mathbf{u}_{e,k})\nabla\mathbf{w}_{e,k}\}\}\,\mathrm{d}s\,.
\end{aligned} \tag{9.31}$$

Note that the notation $\mathbf{u}_{e,k}$ identifies the trace of the solution from phase k on the element face $\mathtt{E}_e$ and $\mathbf{u}_{m(e),k}$ represents the solution defined over one of the whole element adjacent to the face (see Section 5.2.2). The θ parameter determines if the form is symmetric (SIPG), non-symmetric (NIPG) or incomplete (IIPG):

$$\begin{aligned}
\theta = 1 \quad &\rightarrow \text{SIPG}\,,\\
\theta = -1 \quad &\rightarrow \text{NIPG}\,,\\
\theta = 0 \quad &\rightarrow \text{IIPG}\,.
\end{aligned} \tag{9.32}$$

Boundary term In the same way, the boundary term Equation 5.22 combined with Equation 5.27, Equation 5.32 and Equation 5.33 becomes

$$\begin{aligned}
\mathcal{B}_{b,k}(\mathbf{u}^+_{m(b),k}, \mathbf{w}^+_{m(b),k}) = \int_{\mathtt{B}_b\cap\Omega_k} & \mathcal{H}_k(\mathbf{u}^+_{b,k}, \mathbf{u}^{\mathrm{bnd}}_{b,k}, \mathbf{n}_b)\cdot\mathbf{w}^+_{b,k}\\
& - (\underline{\boldsymbol{F}}^{d,\mathrm{bnd}}_k(\mathbf{u}^+_{b,k}, \nabla\mathbf{u}^+_{b,k})\cdot\mathbf{n}_b)\cdot\mathbf{w}^+_{b,k}\\
& + \gamma_{\mathtt{B}}\,(\mathbf{u}^+_{b,k} - \mathbf{u}^{\mathrm{bnd}}_{b,k})(\mathbf{n}_b\cdot\underline{\boldsymbol{G}}^T_k(\mathbf{u}^+_{b,k})\cdot\mathbf{n}_b)\cdot\mathbf{w}^+_{b,k}\,\mathrm{d}s\\
& - \theta\int_{\mathtt{B}_b\cap\Omega_k} (\mathbf{u}^+_{b,k} - \mathbf{u}^{\mathrm{bnd}}_{b,k})\cdot(\mathbf{n}_b\cdot\underline{\boldsymbol{G}}^T_k(\mathbf{u}^+_{b,k})\nabla\mathbf{w}^+_{b,k})\,\mathrm{d}s\,.
\end{aligned} \tag{9.33}$$

Interface term The contribution of the interface term for phase k in an element m is first written by using the specifications of the numerical fluxes in

[1] To make the distinction with the numerical fluxes that were still unspecified in Section 5.2.2, the $\hat{\bullet}$ notation is removed from the fluxes here.

Equation 9.6, Equation 9.17, Equation 9.20 and Equation 9.25 (for the tensorial penalty version):

$$\begin{aligned}
\mathtt{I}_{m,k}(\mathbf{u}_{m,k},\mathbf{u}_{m,k^*},\mathbf{w}_{m,k}) &= \int_{\mathtt{I}_m} \left[\mathcal{H}_{\mathtt{I},k}(\mathbf{u}_{\Gamma_m,k},\mathbf{u}_{\Gamma_m,k^*},\mathbf{n}_{\Gamma,m,k})\right]\cdot\mathbf{w}_{m,k} \\
&- \left[\{\{\underline{\boldsymbol{F}}^d(\mathbf{u}_{\Gamma_m},\nabla\mathbf{u}_{\Gamma_m})\}\}_{\alpha^*}\cdot\mathbf{n}_{\Gamma,m,k} + \alpha_k\,\mathbf{J}^{d,\mathrm{Rob}}(\mathbf{u}_{\Gamma_m,k},\nabla\mathbf{u}_{\Gamma_m,k},\mathbf{u}_{\Gamma_m,k^*},\nabla\mathbf{u}_{\Gamma_m,k^*})\right]\cdot\mathbf{w}_{m,k} \\
&+ \gamma_{\mathtt{I}}\left(\mathbf{u}_{\Gamma_m,1}-\mathbf{u}_{\Gamma_m,2}-\mathbf{J}^{d,\mathrm{Dir}}(\mathbf{u}_{\Gamma_m,k},\mathbf{u}_{\Gamma_m,k^*})\right)\left(\mathbf{n}_{\Gamma,m}\cdot\{\{\underline{\boldsymbol{G}}^T(\mathbf{u}_{\Gamma_m})\}\}\cdot\mathbf{n}_{\Gamma,m,k}\right)\cdot\mathbf{w}_{m,k}\mathrm{d}s \\
&- \theta\int_{\mathtt{I}_m}\alpha_{k^*}\left(\mathbf{u}_{\Gamma_m,1}-\mathbf{u}_{\Gamma_m,2}-\mathbf{J}^{d,\mathrm{Dir}}(\mathbf{u}_{\Gamma_m,k},\mathbf{u}_{\Gamma_m,k^*})\right)\cdot\left(\mathbf{n}_{\Gamma,m}\cdot\underline{\boldsymbol{G}}_k^T(\mathbf{u}_{\Gamma_m,k})\nabla\mathbf{w}_{m,k}\right)\,\mathrm{d}s\,.
\end{aligned} \tag{9.34}$$

It is then instructive to write the interface term in a similar fashion as the face term. To do this, Equation 9.34 is summed over both phases such that:

$$\begin{aligned}
\sum_{k\in\{1,2\}}\mathtt{I}_{m,k}(\mathbf{u}_{m,k},\mathbf{u}_{m,k^*},\mathbf{w}_{m,k}) &= \int_{\mathtt{I}_m}\mathcal{H}_{\mathtt{I},1}(\mathbf{u}_{\Gamma_m,1},\mathbf{u}_{\Gamma_m,2},\mathbf{n}_{\Gamma,m})\cdot\mathbf{w}_{m,1} \\
&\qquad - \mathcal{H}_{\mathtt{I},2}(\mathbf{u}_{\Gamma_m,1},\mathbf{u}_{\Gamma_m,2},\mathbf{n}_{\Gamma,m})\cdot\mathbf{w}_{m,2} \\
&- \{\{\underline{\boldsymbol{F}}^d(\mathbf{u}_{\Gamma_m},\nabla\mathbf{u}_{\Gamma_m})\}\}_{\alpha^*} : \underline{[\![\mathbf{w}_m]\!]} - \mathbf{J}^{d,\mathrm{Rob}}(\mathbf{u}_{\Gamma_m,1},\nabla\mathbf{u}_{\Gamma_m,1},\mathbf{u}_{\Gamma_m,2},\nabla\mathbf{u}_{\Gamma_m,2})\cdot\{\{\mathbf{w}_m\}\}_{\alpha} \\
&+ \gamma_{\mathtt{I}}\Big(\underline{[\![\mathbf{u}_{\Gamma_m}]\!]}\{\{\underline{\boldsymbol{G}}^T(\mathbf{u}_{\Gamma_m})\}\} : \underline{[\![\mathbf{w}_m]\!]} \\
&\qquad - \mathbf{J}^{d,\mathrm{Dir}}(\mathbf{u}_{\Gamma_m,1},\mathbf{u}_{\Gamma_m,2})(\mathbf{n}_{\Gamma,m}\cdot\{\{\underline{\boldsymbol{G}}^T(\mathbf{u}_{\Gamma_m})\}\}) : \underline{[\![\mathbf{w}_m]\!]}\Big)\,\mathrm{d}s \\
&- \theta\int_{\mathtt{I}_m}\underline{[\![\mathbf{u}_{\Gamma_m}]\!]} : \{\{\underline{\boldsymbol{G}}^T(\mathbf{u}_{\Gamma_m})\nabla\mathbf{w}_m\}\}_{\alpha^*} \\
&\qquad - \mathbf{J}^{d,\mathrm{Dir}}(\mathbf{u}_{\Gamma_m,1},\mathbf{u}_{\Gamma_m,2})\cdot(\mathbf{n}_{\Gamma,m}\cdot\{\{\underline{\boldsymbol{G}}^T(\mathbf{u}_{\Gamma_m})\nabla\mathbf{w}_m\}\}_{\alpha^*})\,\mathrm{d}s\,.
\end{aligned} \tag{9.35}$$

Let us note that the notation $\mathbf{u}_{\Gamma_m,k}$ identifies the solution from phase k interpolated on the interface $\mathtt{I}_m$ contained in element $\mathtt{T}_m$. The shape functions $\mathbf{w}_{m,k}$ are also written per phase to be exact even though the current implementation in our solver does not allow considering distinct function spaces between the phases.

9.3.2 Jacobians of interface terms

In addition to the appropriate specification of the interface numerical flux functions, a key ingredient of the solver is the computation of the Jacobians of those fluxes. Indeed, as explained in Section 10.1.2, the non-linear procedure to solve the resulting system of equations require the Jacobian of the global residual and the contributions of the interface fluxes to this global Jacobian are therefore needed. In order to identify the Jacobians that need to be assembled, Equation 9.34 is considered. The vector $\mathtt{I}_{m,k}$ will be added to the global residual vector at positions corresponding to the nodal degrees of freedom associated to phase k in element m. The corresponding contributions to the global Jacobian

matrix requires the computation of the local matrices

$$\frac{\partial \mathrm{I}_{m,k}}{\partial \mathbf{u}_1} \text{ and } \frac{\partial \mathrm{I}_{m,k}}{\partial \mathbf{u}_2},$$

since the interface fluxes for phase k depend on the solution in both phases. Given the expression in Equation 9.34, the following local Jacobian matrices thus need to be assembled (still for the tensorial penalty version):

Convective fluxes	$\frac{\partial \mathcal{H}_{\mathrm{I},k}}{\partial \mathbf{u}_1}$ for $k = 1, 2$	[9.36]
Diffusive direct term	$\frac{\partial(\underline{\boldsymbol{F}}_k^d \cdot \mathbf{n}_{\Gamma,k})}{\partial \mathbf{u}_k}$ and $\frac{\partial \mathbf{J}^{d,\mathrm{Rob}}}{\partial \mathbf{u}_k}$ for $k = 1, 2$	[9.37]
	$\frac{\partial(\underline{\boldsymbol{F}}_k^d \cdot \mathbf{n}_{\Gamma,k})}{\partial(\nabla \mathbf{u}_k)}$ and $\frac{\partial \mathbf{J}^{d,\mathrm{Rob}}}{\partial(\nabla \mathbf{u}_k)}$ for $k = 1, 2$	[9.38]
Diffusive penalty term	$\frac{\partial(\mathbf{n}_\Gamma \cdot \underline{\boldsymbol{G}}_k^T \cdot \mathbf{n}_{\Gamma,k})}{\partial \mathbf{u}_k}$ and $\frac{\partial \mathbf{J}^{d,\mathrm{Dir}}}{\partial \mathbf{u}_k}$ for $k = 1, 2$	[9.39]
Diffusive transpose term	$\frac{\partial(\mathbf{n}_\Gamma \cdot \underline{\boldsymbol{G}}_k^T \nabla \mathbf{w})}{\partial \mathbf{u}_k}$	[9.40]
	$\frac{\partial \mathbf{J}^{d,\mathrm{Dir}}}{\partial \mathbf{u}_k}$ for $k = 1, 2$	[9.41]

All the above Jacobians are computed via forward finite differences formula by perturbing the fluxes with respect to the solutions or the gradients of the solutions.

9.3.3 Weak consistency

The weak consistency of the discrete formulation is then discussed. It must be shown that the solution $\tilde{\mathbf{u}}$ of the continuous problem Equation 9.4 makes the discrete problem Equation 8.4 converges as the mesh spacing h tends to zero:

$$\lim_{h \to 0} \mathcal{N}(\tilde{\mathbf{u}}, \mathbf{w}) = 0 \quad \forall\, \mathbf{w} \in \mathcal{U}_p^X, \tag{9.42}$$

where $\mathcal{N}$ is given by Equation 8.4. Since the contributions $\mathcal{N}^k$ ($k = \{1, 2\}$) in Equation 8.5 result from a standard DG discretization, the focus here is put on the contribution from the cut entities $\mathcal{N}^\Gamma$ defined in Equation 8.6. In Section 9.2.2, the splitting of the interface jump conditions in two separated convective and diffusive parts is discussed. It is inherent to the way the convective and diffusive fluxes are treated by the numerical approach considered. The effect of splitting the jumps is to slightly modify the original strong form of the problem to be discretized. Taking as example the splitting #2 in Table 9.1,

the jumps in the corresponding strong form should be rewritten as

$$\begin{aligned} -\tilde{v}_{n,\Gamma}[\![\tilde{\mathbf{u}}]\!] + [\![\underline{\boldsymbol{F}}^c(\tilde{\mathbf{u}})]\!] \cdot \mathbf{n}_\Gamma &= \mathbf{F}_\Gamma^c \qquad \forall \mathbf{x} \in \tilde{\Gamma}\,, \\ -[\![\underline{\boldsymbol{F}}^d(\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}})]\!] \cdot \mathbf{n}_\Gamma &= \mathbf{F}_\Gamma^d \qquad \forall \mathbf{x} \in \tilde{\Gamma}\,, \end{aligned} \tag{9.43}$$

where the right-and-side surface terms associated to the convective and diffusive parts are given by:

$$\mathbf{F}_\Gamma^c = \begin{bmatrix} 0 \\ \Delta_\Gamma p\, \mathbf{n}_\Gamma \\ \Delta_\Gamma p\, \tilde{v}_{n,\Gamma} \end{bmatrix}, \tag{9.44}$$

$$\mathbf{F}_\Gamma^d = \begin{bmatrix} 0 \\ (\Delta_\Gamma \tau_{nt} - \dot{m}_\Gamma [\![\tilde{v}_t]\!])\, \mathbf{t}_\Gamma \\ \Delta_\Gamma \tau_{nt}\, \tilde{v}_{t,\Gamma} + \dot{q}_\Gamma \end{bmatrix}. \tag{9.45}$$

Thus, the consistency of the scheme shall be demonstrated based on the continuous problem Equation 9.4 equipped with the modified jumps, Equation 9.43. Following Ralph Hartmann (2008, Sec. 9.3.2), the first step is to apply integration by parts on the volume term Equation 5.20 to reformulate it as

$$\begin{aligned} \mathcal{U}_{m,k}(\mathbf{u}, \mathbf{w}) = &\int_{\mathbb{T}_m \cap \Omega_k} \nabla \cdot \left(\underline{\boldsymbol{F}}_k^c(\mathbf{u}) - \underline{\boldsymbol{F}}_k^d(\mathbf{u}, \nabla\mathbf{u}) \right) \cdot \mathbf{w}\, \mathrm{d}\mathbf{x} \\ &- \int_{\overline{\mathbb{T}_m \cap \Omega_k}} \left(\underline{\boldsymbol{F}}_k^c(\mathbf{u}) - \underline{\boldsymbol{F}}_k^d(\mathbf{u}, \nabla\mathbf{u}) \right) : \mathbf{w} \otimes \mathbf{n}\, \mathrm{d}s\,, \end{aligned} \tag{9.46}$$

where the integrals along the edges of the cut element $\mathbb{T}_m \cap \Omega_k$, noted $\overline{\mathbb{T}_m \cap \Omega_k}$, can be split into integrals over the element faces contained in phase k, the integrals over the domain boundaries in phase k and the integrals over the phase interface. The contribution from a given phase k to the discrete form on cut entities is first considered, i.e. the term noted $\mathcal{N}_k^\Gamma$ such that $\mathcal{N}^\Gamma = \mathcal{N}_1^\Gamma + \mathcal{N}_2^\Gamma$. This contribution is rewritten in an element-based form, i.e. the integrals over each interior face appear twice in the global sum. To make the distinction with the face-based form where the sum is performed on the set of mesh faces $\mathbb{E}^\Gamma$, the notation $\partial(T_m \cap \Omega_k)$ will be used to identify the edges of a given element T_m contained in phase k, which are shared with neighboring elements with opposite normals. It has to be remarked that $\partial(T_m \cap \Omega_k)$ does not include domain boundaries nor phase interfaces, which will still be identified by $\mathbb{B}_b \cap \Omega_k$

and I_m respectively. Injecting Equation 9.46 into $\mathcal{N}_k^\Gamma$ yields:

$$
\begin{aligned}
\mathcal{N}_k^\Gamma(\mathbf{u}_k, \mathbf{u}_{k^*}, \mathbf{w}) = & \sum_{m\in\mathbb{T}^\Gamma} \int_{\mathrm{T}_m\cap\Omega_k} \left[\frac{\partial \mathbf{u}_k}{\partial t} + \nabla\cdot\left(\underline{\boldsymbol{F}}_k^c(\mathbf{u}_k) - \underline{\boldsymbol{F}}_k^d(\mathbf{u}_k, \nabla\mathbf{u}_k)\right) \right. \\
& \qquad \left. - \mathbf{S}_k(\mathbf{u}_k, \nabla\mathbf{u}_k) \right] \cdot \mathbf{w}\,\mathrm{d}\mathbf{x} \\
& + \sum_{m\in\mathbb{T}^\Gamma} \{ \int_{\partial(T_m\cap\Omega_k)} \left[\mathcal{H}_k(\mathbf{u}_k^+, \mathbf{u}_k^-, \mathbf{n}^+) - \underline{\boldsymbol{F}}_k^c(\mathbf{u}_k^+)\cdot\mathbf{n}^+ \right] \cdot \mathbf{w}^+ \\
& \qquad - \left[\{\{\underline{\boldsymbol{F}}_k^d(\mathbf{u}_k, \nabla\mathbf{u}_k)\}\}\cdot\mathbf{n}^+ - \underline{\boldsymbol{F}}_k^d(\mathbf{u}_k^+, \nabla\mathbf{u}_k^+)\cdot\mathbf{n}^+ \right] \cdot \mathbf{w}^+ \\
& \qquad + \gamma_{\mathrm{E}}\,(\mathbf{u}_k^+ - \mathbf{u}_k^-)(\mathbf{n}^+ \cdot \{\{\underline{\boldsymbol{G}}_k^T(\mathbf{u}_k)\}\}\cdot\mathbf{n}^+)\cdot\mathbf{w}^+\,\mathrm{d}s \\
& \qquad - \theta \int_{\partial(T_m\cap\Omega_k)} \frac{1}{2}(\mathbf{u}_k^+ - \mathbf{u}_k^-)\cdot(\mathbf{n}^+\cdot\underline{\boldsymbol{G}}_k^T(\mathbf{u}_k^+)\nabla\mathbf{w}^+)\,\mathrm{d}s\ \} \\
& + \sum_{b\in\mathbb{B}^\Gamma} \{ \int_{\mathrm{B}_b\cap\Omega_k} \left[\mathcal{H}_k(\mathbf{u}_k^+, \mathbf{u}_k^{\mathrm{bnd}}, \mathbf{n}_b) - \underline{\boldsymbol{F}}_k^c(\mathbf{u}_k^+)\cdot\mathbf{n}_b^+ \right] \cdot \mathbf{w}^+ \\
& \qquad - \left[\underline{\boldsymbol{F}}_k^{d,\mathrm{bnd}}(\mathbf{u}_k^+, \nabla\mathbf{u}_k^+)\cdot\mathbf{n}_b^+ - \underline{\boldsymbol{F}}_k^d(\mathbf{u}_k^+, \nabla\mathbf{u}_k^+)\cdot\mathbf{n}_b^+ \right] \cdot \mathbf{w}^+ \\
& \qquad + \gamma_{\mathrm{B}}\,(\mathbf{u}_k^+ - \mathbf{u}_k^{\mathrm{bnd}})(\mathbf{n}_b^+\cdot\underline{\boldsymbol{G}}_k^T(\mathbf{u}_k^+)\cdot\mathbf{n}_b^+)\cdot\mathbf{w}^+\,\mathrm{d}s \\
& \qquad - \theta \int_{\mathrm{B}_b\cap\Omega_k} (\mathbf{u}_k^+ - \mathbf{u}_k^{\mathrm{bnd}})\cdot(\mathbf{n}_b^+\cdot\underline{\boldsymbol{G}}_k^T(\mathbf{u}_k^+)\nabla\mathbf{w}^+)\,\mathrm{d}s\ \} \\
& + \sum_{m\in\mathbb{I}} \{ \int_{\mathrm{I}_m} \left[\mathcal{H}_{\mathrm{I},k}(\mathbf{u}_{k,\Gamma}, \mathbf{u}_{k^*,\Gamma}, \mathbf{n}_\Gamma) - \underline{\boldsymbol{F}}_k^c(\mathbf{u}_{k,\Gamma})\cdot\mathbf{n}_{\Gamma,k} \right] \cdot \mathbf{w} \\
& \qquad - \left[\{\{\underline{\boldsymbol{F}}^d(\mathbf{u}_\Gamma, \nabla\mathbf{u}_\Gamma)\}\}_{\alpha^*}\cdot\mathbf{n}_{\Gamma,k} - \underline{\boldsymbol{F}}_k^d(\mathbf{u}_{k,\Gamma}, \nabla\mathbf{u}_{k,\Gamma})\cdot\mathbf{n}_{\Gamma,k} + \alpha_k\mathbf{J}^{d,\mathrm{Rob}} \right] \cdot \mathbf{w} \\
& \qquad + \gamma_{\mathrm{I}} \left(\mathbf{u}_{1,\Gamma} - \mathbf{u}_{2,\Gamma} - \mathbf{J}^{d,\mathrm{Dir}}(\mathbf{u}_{k,\Gamma}, \mathbf{u}_{k^*,\Gamma})\right)\left(\mathbf{n}_\Gamma\cdot\{\{\underline{\boldsymbol{G}}^T(\mathbf{u}_\Gamma)\}\}\cdot\mathbf{n}_{\Gamma,k}\right)\cdot\mathbf{w}\,\mathrm{d}s \\
& \qquad - \theta \int_{\mathrm{I}_m} \alpha_{k^*}\left(\mathbf{u}_{1,\Gamma} - \mathbf{u}_{2,\Gamma} - \mathbf{J}^{d,\mathrm{Dir}}(\mathbf{u}_{k,\Gamma}, \mathbf{u}_{k^*,\Gamma})\right)\cdot\left(\mathbf{n}_\Gamma\cdot\underline{\boldsymbol{G}}_k^T(\mathbf{u}_{k,\Gamma})\nabla\mathbf{w}\right)\,\mathrm{d}s\ \}
\end{aligned}
\qquad [9.47]
$$

Injecting the exact solutions in phase k and k^*, $\tilde{\mathbf{u}}_k$ and $\tilde{\mathbf{u}}_{k^*}$, into $\mathcal{N}_k^\Gamma(\mathbf{u}_k, \mathbf{u}_{k^*}, \mathbf{w})$, it comes out that all the contributions tend to zero because of the following reasons:

- Volume: $\tilde{\mathbf{u}}_k$ must satisfy the strong form of the equations inside the domain:
$$\frac{\partial\tilde{\mathbf{u}}_k}{\partial t} + \nabla\cdot\left(\underline{\boldsymbol{F}}_k^c(\tilde{\mathbf{u}}_k) - \underline{\boldsymbol{F}}_k^d(\tilde{\mathbf{u}}_k, \nabla\tilde{\mathbf{u}}_k)\right) = \mathbf{S}_k(\tilde{\mathbf{u}}_k, \nabla\tilde{\mathbf{u}}_k)\,.$$
- Internal faces: because the numerical flux provided by the Riemann solver in the bulk is consistent, this gives
$$\mathcal{H}_k(\tilde{\mathbf{u}}_k, \tilde{\mathbf{u}}_k, \mathbf{n}) - \underline{\boldsymbol{F}}_k^c(\tilde{\mathbf{u}}_k)\cdot\mathbf{n} = 0\,.$$
Then, because of the continuity of $\tilde{\mathbf{u}}_k$, the jump in diffusive fluxes disappear
$$\{\{\underline{\boldsymbol{F}}_k^d(\tilde{\mathbf{u}}_k, \nabla\tilde{\mathbf{u}}_k)\}\} - \underline{\boldsymbol{F}}_k^d(\tilde{\mathbf{u}}_k, \nabla\tilde{\mathbf{u}}_k) = 0\,,$$
as well as the consistently added transpose and penalty terms that involve

the inter-element solution jumps

$$\underline{[\![\tilde{\mathbf{u}}_k]\!]} = 0 \,.$$

- Boundaries: because the continuous solution satisfies the Dirichlet and Neumann boundary conditions,

$$\mathbf{u}_k^{\mathrm{bnd}}(\tilde{\mathbf{u}}_k) = \tilde{\mathbf{u}}_k \,,\ \underline{\boldsymbol{F}}_k^{d,\mathrm{bnd}}(\tilde{\mathbf{u}}_k, \nabla\tilde{\mathbf{u}}_k) = \underline{\boldsymbol{F}}_k^{d}(\tilde{\mathbf{u}}_k, \nabla\tilde{\mathbf{u}}_k) \,,$$

  all the terms in the boundary contributions converge to zero.
- Interfaces: the multiphase Riemann solver provides a two-valued interface numerical flux function that is consistent because it is constructed from a standard HLLC structure. Moreover, each $\mathcal{H}_{\mathrm{I},k}$ function computes the convective fluxes for phase k from "star" states $\mathbf{u}_k^*$ that are obtained such as to satisfy the prescribed convective jumps at the interface. This leads to the equality of this numerical convective flux with the normal convective flux evaluated at the interface when the exact solution is considered:

$$\mathcal{H}_{\mathrm{I},k}(\tilde{\mathbf{u}}_{k,\Gamma}, \tilde{\mathbf{u}}_{k^*,\Gamma}, \mathbf{n}_\Gamma) - \underline{\boldsymbol{F}}_k^{c}(\tilde{\mathbf{u}}_{k,\Gamma}) \cdot \mathbf{n}_{\Gamma,k} = 0 \,.$$

  For the direct diffusive flux, simplifications can be made to obtain

$$\begin{aligned}
&\left[\left(\{\{\underline{\boldsymbol{F}}^{d}(\mathbf{u}_\Gamma, \nabla\mathbf{u}_\Gamma)\}\}_{\alpha^*} - \underline{\boldsymbol{F}}_k^{d}(\mathbf{u}_{k,\Gamma}, \nabla\mathbf{u}_{k,\Gamma})\right) \cdot \mathbf{n}_{\Gamma,k} + \alpha_k \mathbf{J}^{d,\mathrm{Rob}}\right] \\
&= \alpha_k \Big[\left(\underline{\boldsymbol{F}}_1^{d}(\mathbf{u}_{1,\Gamma}, \nabla\mathbf{u}_{1,\Gamma}) - \underline{\boldsymbol{F}}_2^{d}(\mathbf{u}_{2,\Gamma}, \nabla\mathbf{u}_{2,\Gamma})\right) \cdot \mathbf{n}_\Gamma \\
&\qquad - \mathbf{J}^{d,\mathrm{Rob}}(\mathbf{u}_{1,\Gamma}, \nabla\mathbf{u}_{1,\Gamma}, \mathbf{u}_{2,\Gamma}, \nabla\mathbf{u}_{2,\Gamma})\Big] \,,
\end{aligned}$$

  which tends to zero since by construction, $\mathbf{J}^{d,\mathrm{Rob}}$ is such that

$$\begin{aligned}
&\left[\underline{\boldsymbol{F}}_1^{d}(\tilde{\mathbf{u}}_{1,\Gamma}, \nabla\tilde{\mathbf{u}}_{1,\Gamma}) - \underline{\boldsymbol{F}}_2^{d}(\tilde{\mathbf{u}}_{2,\Gamma}, \nabla\tilde{\mathbf{u}}_{2,\Gamma})\right] \cdot \mathbf{n}_\Gamma \\
&= \mathbf{J}^{d,\mathrm{Rob}}(\tilde{\mathbf{u}}_{1,\Gamma}, \nabla\tilde{\mathbf{u}}_{1,\Gamma}, \tilde{\mathbf{u}}_{2,\Gamma}, \nabla\tilde{\mathbf{u}}_{2,\Gamma}) \,.
\end{aligned}$$

  The additional transpose and stabilizing penalty terms also both tend to zero since by construction of $\mathbf{J}^{d,\mathrm{Dir}}$:

$$\tilde{\mathbf{u}}_1 - \tilde{\mathbf{u}}_2 = \mathbf{J}^{d,\mathrm{Dir}}(\tilde{\mathbf{u}}_1, \tilde{\mathbf{u}}_2) \,.$$

This shows that the formulation is consistent with respect to the continuous problem in which the jump conditions are adjusted to the ones imposed by the numerical splitting selected.

Contribution 9.1: Enforcement of jump conditions between two compressible viscous phases

The main contributions that can be identified from this chapter are:

- A detailed overview of the ways to enforce jump conditions in

a monolithic sharp interface two-phase solver.
- A novel strategy to enforce non-linear jump conditions for viscous compressible flows in a monolithic sharp interface DG method.
- An analysis of the resulting discretization in terms of consistency.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- Two classes to provide the values of the jumps on the convective and diffusive fluxes to be inserted into the weak form, depending on the splitting.
- The new assembly of the interface contributions to the global residual and Jacobian matrix to be aligned with the proposed strategy.

The resulting two-phase solver is, to the author's best knowledge, the first of its kind high-order geometrically unfitted (i.e. based on the extension of the approximation space and not cut-cell construction) FE solver for viscous compressible flows.

Rest of this page intentionally left blank.

10

Temporal discretization, system solving and coupling between interface and fluid dynamics

Part of the contents of Section 10.2 and Section 10.3 was published in Henneaux et al. (2021b) and presented in Henneaux et al. (2021c).

10.1 Implicit time integration applied to standard DG discretization 340
10.1.1 Overview of time stepping schemes 340
10.1.2 Dual time-stepping and time preconditioning 345
10.1.3 Use of non-conservative sets of variables 351
10.2 Time integration with moving interfaces for the XDG method . 353
10.2.1 Overview of methods to handle evolving cut-mesh topologies 353
10.2.2 Splitting and moving interface approaches to integrate over time-varying domains 354
10.2.3 Cell-agglomeration to ensure constant cut topology over time levels 359
10.2.4 Assembly of the temporal residual for general sets of variables 364
10.2.5 General procedure 366
10.3 Coupling strategy between the fluid and level-set solvers 370
10.3.1 Overview of possible coupling strategies 370
10.3.2 Algorithm of the implemented segregated approach . . 371

10.1 Implicit time integration applied to standard DG discretization

10.1.1 Overview of time stepping schemes

Instead of using full discretization scheme such as space-time methods, the temporal and spatial discretizations are usually considered separately. In CFD, the most common approach is the method of lines consisting in performing first the spatial discretization before the temporal one. In this case, the goal of the temporal discretization is to solve a system of ODEs. Two strategies are possible: explicit and implicit. They are discussed below along with the reasons why an implicit time stepping scheme is preferable in this work and what are the associated challenges.

Choice of implicit schemes

Characteristic of the ODEs to integrate The system of ODEs resulting from the spatial discretization of the governing equations considered in this work contain several physical time scales associated with acoustic waves, viscous diffusion, material advection, phase transition, heat flux and surface tension. Most of the time, one is not interested in resolving all those time scales. For example, selecting a time step that captures the acoustic waves propagation is likely to be not compatible with the time frame of interest to simulate. Thus, we would like to have the possibility to step over some of those time scales. However, the presence of the under-resolved temporal scales results in numerical systems that develop strong hyperbolic and parabolic stiffness, as noted by Weston et al. (2019).

On stability properties The concept of stability for an initial value problem such as the one we are interested here refers to the behavior of the numerical solution in response to small perturbations. Those are often caused by numerical errors. While stable time integration schemes will dampen those oscillations, unstable methods will tend to amplify them, resulting in erroneous values that grow with each time step. Holst et al. (2020) categorize the major categories of stability as follows:

- *Linear stability*: linear stability studies solutions to linear ODE such as the canonical $y' = \lambda\, y(x)$.
  - *Zero-stability*: zero-stability is concerned with the behavior of the solution when the time step tends to zero, $\Delta t \to 0$, at a fixed time instant, $t =$ constant.

  - *A-stability*: absolute-stability is about the solution behavior when time goes to infinity, $t \to \infty$, for a fixed time step, Δt = constant.
  - *L-stability*: linear-stability is a special case of A-stability for which the stability function of the method tends to zero as the corresponding complex number with respect to which it is expressed tends to infinity. This ensures that the method, in addition of being stable for long times, will also reflect the asymptotic behavior of the solution of the original problem.

  From a practical point of view, we are generally interested in the absolute stability of the schemes as it will determine the convergence of a particular scheme with a fixed Δt as time grows. The schemes possessing this property are said to be A-stable. For stiff problems, the L-stability property is desirable because they can dampen the irrelevant high-frequency components of the solutions, which make them more efficient than A-stable methods for this class of ODEs.
- *Non-linear stability*: nonlinear stability studies solutions to nonlinear ODE with the general form $y' = f(x, y)$. Similar concepts as for the linear stability exist, such as algebraic-, B-, BN-, or AN-stability. The interested reader can consult Wanner and Hairer (1996, Sec. IV.12) for more details.

Since the Navier-Stokes equations lead to a nonlinear system of ODEs, it would seem logical to analyze the time-stepping schemes within the purview of nonlinear stability. However, Kennedy and Carpenter (2016, Sec. 2.4), citing some other work, seem to advocate the use of L-stable methods over non-linearly (algebraically) stable methods because of their generally greater practical value (at least in the context of implicit Runge-Kutta methods). There seems to be no consensus on which stability criterion a method should meet as this choice is problem-dependent. From a general point of view, in the high-order CFD community, the most common methods are A-stable or L-stable. Bijl et al. (2002) suggest that L-stability is preferable such that the eigenvalues in the far left-hand side of the complex plane are damped. Fidkowski (2023) points out that for high spatial-order discretizations of convection-dominated flows, this is even more important since the spatial discretization may lead to a combination of eigenvalues with large-magnitude negative real parts and ones close to the imaginary axis.

Choice of implicit over explicit schemes Explicit schemes are attractive for their efficiency and ease of implementation. However, they are conditionally stable and they can place burdensome requirements on the time step size, especially in the presence of fast (stiff) time scales. This constraint is exacerbated by high spatial order. That is why for many applications, *a fortiori* the ones described by a stiff model, the stability offered by implicit schemes outweighs their drawbacks related to implementation difficulties and storage requirements (Holst et al. (2020)). The alternative to choosing a purely explicit or implicit scheme is to be found in the implicit-explicit (IMEX) methods, such as in

Zeifang and A. Beck (2023). The idea behind these methods is to use operator splitting such that the parts of the model associated to fast time scales are treated by an implicit scheme while the slower time scales are handled by an explicit scheme. Another variant proposed in Per-Olof Persson (2011) consists in applying implicit time integration on small elements where the time step constraint would be critical and explicit integration elsewhere. This type of schemes is not investigated in this work and a fully-implicit time discretization is adopted.

Characteristics of implicit schemes

There is a wide variety of implicit schemes but the most commonly applied to the compressible Navier-Stokes equations can be categorized as follows:

- *Linear multi-step methods*: the principle of those methods is to linearly combine solution values from the previous and the new time steps. They are computationally efficient because they do not require intermediate time steps. They do require storing solution from previous time steps but this is in general not an issue. However, A-stable linear multi-step methods are limited to second-order, which can be a significant drawback. The most popular class of implicit linear multi-step methods are the Backward Differentiation Formulae (BDF). Bijl et al. (2002) note that practical experience indicates that large-scale simulations are seldom stable with BDF4 while the BDF3 is often stable but may diverge under certain circumstances.
- *Multi-stage Runge-Kutta methods*: another way to obtain higher-order accuracy is to perform multiple sub-time steps per time step. The implicit Runge-Kutta (IRK) methods are particularly exploited for high-order compressible CFD computations because of their favorable stability properties. In this large family of RK methods, each method can be identified by their coefficients in the Butcher tableau. Holst et al. (2020) describe that Diagonally implicit RK (DIRK) are easy to implement because each stage can be solved sequentially due to the fact that all coefficients above the main diagonal in the tableau are zero, and that Singly DIRK (SDIRK), with their constant coefficient along the main diagonal, have the added benefit of having a constant Jacobian over all stages. However, Kennedy and Carpenter (2016, Sec. 1) point out that the stage-order of DIRK methods may not exceed two. Bijl et al. (2002) highlight that the reduced stage order may give rise to order-reduction phenomena in the presence of substantial stiffness. If a stage-order of three or greater is desired, fully implicit RK (FIRK) methods may be considered but are substantially more expensive to implement. Other methods in this category worth mentioning are the Explicit first stage SDIRK (ESDIRK) methods that reduce by one the number of non-linear systems to be solved per time step, and the linearly implicit Rosenbrock that linearize a DIRK scheme to reduce the computational effort, at the expense of sacrificing some

stability properties and accuracy (Blom et al. (2016)).
- *Hybrid multi-step multi-stage methods*: hybrid methods blend the two previous classes of methods with the goal of reducing the number of intermediate stages while maintaining A- or L-stability. This includes the Modified Extended BDF (MEBDF) schemes (Jeff R Cash (1983)) which are A-stable up to order four and the Two-Implicit Advanced Step-point (TIAS) schemes (Psihoyios and J. Cash (1998)) which are A-stable up to order six.

A comparison between some of the methods mentioned above can be found e.g. in Francesco Bassi et al. (2014), Alessandra Nigro et al. (2014) and Morozova et al. (2023).

Non-linear iterations Applied to the unsteady compressible Navier-Stokes equations, all implicit schemes lead to a nonlinear system of equations to be solved to obtain the solution at the next time step. The common choice in numerically solving these nonlinear systems is to use Newton-like iterative methods. At each iteration, a linear system is formed and needs to be solved.

Linear system solving To solve the linear systems, we have the choice between exact direct solvers or approximate iterative ones. Direct solvers often involve two successive steps: (i) decomposition of the system matrix, (ii) substitution to find the exact solution of the system. Although the efficiency of direct solvers can be considerably improved by renumbering techniques, their memory and CPU requirements still make their use inappropriate for very large systems, like in three-dimensional settings. Thanks to the sparsity of the system matrix, iterative solvers turn out to be a much more efficient choice for large systems. Iterative techniques construct a sequence of solutions which is expected to converge toward the exact solution, within a prescribed tolerance level. Among the variety of iterative linear solvers, Krylov subspace methods are nowadays the most commonly used for their better robustness and convergence properties compared to classical iterative schemes like Gauss-Seidel or successive over relaxation (SOR) methods (see e.g. Saad (2003, Sec. 6)).

Efficiency, accuracy and robustness As noted by Yan (2021, Sec. 1.2), many researchers turn to the explicit time integration methods for unsteady simulations. This can likely be associated to the complexity of implicit schemes in terms of the numerous parameters to adjust, such as the order of the scheme, the time step size, the Newton iteration convergence tolerance, the linear iteration convergence and other numerical parameters associated to the linear solver. These parameters are usually highly problem-dependent and can have a strong influence on the efficiency, accuracy and robustness of the solver. Intensive research is still underway to improve each of these aspects. Some directions being explored are summarized below:

- *Efficiency*: although Krylov subspaces methods are attractive, they are all likely to suffer from slow convergence and lack of robustness for problems in fluid dynamics. This is even more true for linear systems arising from stiff multiphysics problems like the one we are considering in this work because they tend to be highly ill-conditioned. To improve the convergence, to keep the number of iterations relatively small and to prevent the storage and CPU time from becoming prohibitive, preconditioning is applied. In fact, Saad (2003, Sec. 9.1) notes that the reliability of iterative techniques "depends much more on the quality of the preconditioner than on the particular Krylov subspace accelerators used". However, Pan (2022, Sec. 1.4.1) highlights that there is not an optimal preconditioner for general problems.
- *Accuracy*: the local discretization errors introduced during the advancement of the solution from t_n to t_{n+1} consist of spatial, temporal and iterative errors. Spatial errors can be estimated and reduced through mesh- or order-adaptive techniques. An adaptive time-stepping strategy combined with an adaptive Newton tolerance would also be desirable to automatically set a time step and a nonlinear tolerance such that the temporal and iterative errors are smaller than the spatial error. Otherwise, the benefits of high-order spatial accuracy can be drowned out. Such balanced time step adaptation strategies taking into account spatial discretization errors have been recently proposed by Pan et al. (2022) and Fidkowski (2023). As far as the iterative errors linked to the linear system are concerned, Yan (2021, Sec. 1.2) indicates that they have a much larger influence on high-order simulations, which can lead to much more nonlinear iterations.
- *Robustness*: although implicit scheme may be unconditionally stable, this does not mean that the solution will not diverge. One of the reasons why this may happen is that local Newton's methods do not guarantee the convergence of the non-linear iterations. Thus, although the underlying implicit scheme will not be responsible for the failure of the computation, whatever the time step chosen, the non-linear iterative process may diverge due to a too large time step inducing an initial guess which is not "sufficiently good". The choice of the time step size based solely on the flow physics of interest rather than the grid size (which is in theory the strength of implicit schemes compared to explicit ones) is therefore not always valid. To overcome these robustness issues, globalization of standard local Newton's solvers, through backtracking or trust-region methods, have been applied in the CFD community, such as e.g. in Gutiérrez-Jorquera and Kummer (2022). A complete overview can be found in Pawlowski et al. (2006). It is interesting to relay here the remark made by Bijl et al. (2002) about the fact that L-stable scheme properties should facilitate convergence of the nonlinear equation solvers.

In the `Argo` platform, BDF, ESDIRK, MEBDF and TIAS schemes are available. In this work, we mostly rely on the BDF schemes for (i) its low computational cost, (ii) its single-stage nature and its order of accuracy which are compatible

with the implementation of the coupling between the level-set and XDG solvers, as discussed in Section 10.3.2.

10.1.2 Dual time-stepping and time preconditioning

Since we are using the method of lines for the temporal discretization of the equations, we start by considering the spatially discretized form given in Equation 5.15. This represents a system of non-linear ordinary differential equations (ODEs) that needs to be now discretized in time.

Temporal residual in terms of general variables

First, the semi-discretized elementwize temporal residual, Equation 5.18, is rewritten as

$$\begin{aligned} \mathcal{T}_m(\underline{\boldsymbol{U}}_{m,-,-}) &= \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \frac{\partial \mathbf{u}^{\text{cons}}\left(\mathbf{u}_m(\mathbf{x},t)\right)}{\partial t} \, \mathrm{d}\mathbf{x} \\ &= \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \frac{\partial \mathbf{u}^{\text{cons}}\left(\boldsymbol{\phi}_{m,-}^T(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}(t)\right)}{\partial t} \, \mathrm{d}\mathbf{x}, \end{aligned} \tag{10.1}$$

in which the notation $\mathbf{u}^{\text{cons}}(\mathbf{u})$ is used to indicate that the governing equations are formulated with respect to the conservative variables sets, ρ–$\rho\mathbf{v}$–ρE , but the system can be solved for any other valid set of variables (the validity of this choice is discussed in Section 5.3.1), here simply noted $\mathbf{u}$ to stay as general as possible. In other words, the conversion from the selected working variables set to the conservative one has to be done. This corresponds to the first approach described at the end of Section 5.3.2, the second one consisting in applying the chain rule to the temporal term. The partial derivative appearing in Equation 10.1 then needs to be discretized. Because we will only rely on single-stage multi-step schemes for the temporal discretization of the XDG discretized fluid system (for reasons explained in Section 10.3.2), we write the contribution of the now fully discretized temporal residual in element $\mathbb{T}_m$ as

$$\mathcal{T}_m^{\Delta t}(\underline{\boldsymbol{U}}_{m,-,-}) = \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\sum_{i=-N_{\text{step}}+1}^{1} \alpha_i \, \mathbf{u}^{\text{cons}}\left(\boldsymbol{\phi}_{m,-}^T(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}(t_{n+i})\right) \right] \mathrm{d}\mathbf{x}, \tag{10.2}$$

where the notation $\mathcal{T}^{\Delta t}$ is used to indicate that the temporal discretization has been applied and α_i are the coefficients of the multi-step time-stepping scheme. We can remark that in the case where $\mathbf{u} = \mathbf{u}^{\text{cons}}$, we retrieve the classical mass matrix,

$$\mathcal{T}_m^{\Delta t}(\underline{\boldsymbol{U}}_{m,-,-}) = \underline{\boldsymbol{M}}_m \left[\sum_{i=-N_{\text{step}}+1}^{1} \alpha_i \, \underline{\boldsymbol{U}}_{m,-,-}(t_{n+i}) \right]. \tag{10.3}$$

System to be solved

Based on the assembly of the volume, interface and boundary residual terms covered in Section 5.2.4 and Equation 10.2, the problem to be solved can be stated as follows, making the distinction between steady and unsteady simulations:

- *Steady*: find the unknown nodal degrees of freedom $\underline{\boldsymbol{U}}_{-,-,-}(t_\infty)$ such that

$$\mathcal{N}(\underline{\boldsymbol{U}}(t_\infty)) = \mathcal{N}^{\text{sp}}(\underline{\boldsymbol{U}}(t_\infty)) = \boldsymbol{0}\,. \tag{10.4}$$

- *Unsteady*: for every new time level t_{n+1}, find the unknown nodal degrees of freedom $\underline{\boldsymbol{U}}_{-,-,-}(t_{n+1})$ such that

$$\mathcal{N}(\underline{\boldsymbol{U}}(t_{n+1})) = \mathcal{N}^{\text{tp},\Delta t}(\underline{\boldsymbol{U}}(t_{n+1})) + \mathcal{N}^{\text{sp}}(\underline{\boldsymbol{U}}(t_{n+1})) = \boldsymbol{0}\,. \tag{10.5}$$

The temporal part of the global residual is given by Equation 5.17,

$$\mathcal{N}^{\text{tp},\Delta t}(\underline{\boldsymbol{U}}) = \sum_{m\in\mathbb{T}} \mathcal{T}_m^{\Delta t}(\underline{\boldsymbol{U}}_{m,-,-})\,,$$

with $\mathcal{T}_m^{\Delta t}$ given by Equation 10.2, and the spatial part by Equation 5.19,

$$\begin{aligned}\mathcal{N}^{\text{sp}}(\underline{\boldsymbol{U}}) = &\sum_{m\in\mathbb{T}} \mathcal{V}_m(\underline{\boldsymbol{U}}_{m,-,-}) \\ &+ \sum_{e\in\mathbb{E}} \mathcal{E}_e(\underline{\boldsymbol{U}}^+_{m(e),-,-}, \underline{\boldsymbol{U}}^-_{m(e),-,-}) \\ &+ \sum_{b\in\mathbb{B}} \mathcal{B}_b(\underline{\boldsymbol{U}}^+_{m(b),-,-})\,,\end{aligned}$$

with $\mathcal{V}_m$ given by Equation 5.39, and the assembly of $\mathcal{E}_e$ and $\mathcal{B}_b$ following Equation 5.50 and Equation 5.55. Note that we replace $\underline{\boldsymbol{U}}_{-,-,-}$ by $\underline{\boldsymbol{U}}$ to simplify the notations. This generic way of assembling the residual is convenient as both steady and unsteady simulations can be performed following the same steps.

Addition of a pseudo-temporal term

Equation 10.4 and Equation 10.5 represent a non-linear system of equations to be solved at each time step. There are two possibilities to do so: (i) solve directly this equation by an appropriate non-linear solver, (ii) transform the system into a transient problem and search for the steady-state solution by time-marching. In `Argo`, the second option was already implemented and a dual-time stepping strategy is thus followed. This is justified by the fact that solving directly the non-linear system by an iterative non-linear solver can be costly and prone to divergence. Instead, it can be solved by treating it as a steady state problem by introducing this pseudo-temporal derivative and by *time-marching* the solution using local pseudo-time steps in pseudo-time τ. In other words, we wish to find the steady-state solution $\mathbf{u}^* = \lim_{\tau\to\infty} \mathbf{u}(\tau)$ starting from some initial condition $\mathbf{u}(\tau = 0)$. Since we are not concerned by the time accuracy in pseudo-time, the

best strategy is to select a low-order implicit time-stepping scheme to discretize it (typically backward Euler) so that arbitrary large time steps $\Delta\tau$ can be taken and dynamically adjusted during the sequence of pseudo-time iterations. As noted by Nishikawa (2021), the addition of the pseudo-temporal term amounts to add a diagonal matrix to the Jacobian matrix of the linear system that would be obtained from the linearization of the original non-linear system. This has the effect of improving the solver robustness and has the potential to accelerate the convergence[1].
Equation 10.4 and Equation 10.5 are therefore augmented by a pseudo-temporal term spatially discretized by the DGM, viz.

$$\mathcal{N}^{\text{pseudo}}(\underline{\boldsymbol{U}}) = \sum_{m\in\mathbb{T}} \mathcal{J}_m^{\text{pseudo}}(\underline{\boldsymbol{U}}_{m,-,-}) \,,$$

with

$$\begin{aligned}\mathcal{J}_m^{\text{pseudo}}(\underline{\boldsymbol{U}}_{m,-,-}) &= \sum_{m\in\mathbb{T}} \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(\underline{\boldsymbol{P}}_{\mathbf{u}}(\mathbf{u}_m) \frac{\partial \mathbf{u}_m}{\partial \tau} \right) \mathrm{d}\mathbf{x} \\ &= \sum_{m\in\mathbb{T}} \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(\underline{\boldsymbol{P}}_{\mathbf{u}} \left(\boldsymbol{\phi}_{m,-}^T(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}(t) \right) \frac{\partial \left(\boldsymbol{\phi}_{m,-}^T(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}(t) \right)}{\partial \tau} \right) \mathrm{d}\mathbf{x} \,.\end{aligned}$$

The role of the preconditioning matrix $\underline{\boldsymbol{P}}_{\mathbf{u}}(\mathbf{u})$ is to alter the transient process leading to steady-state solution in pseudo-time τ, as discussed in Section 5.3.4. We can also remark that this pseudo-temporal term is directly expressed based on the working variables set and not the conservative one since $\underline{\boldsymbol{P}}_{\mathbf{u}}$ depends on the variables choice. Since we are not interested in a time accurate solution with respect to the dual time, an implicit Euler scheme is employed to discretize the pseudo-temporal derivative,

$$\begin{aligned}\mathcal{J}_m^{\text{pseudo},\Delta\tau}(\underline{\boldsymbol{U}}_{m,-,-}) = &\underbrace{\int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x})\, \underline{\boldsymbol{P}}_{\mathbf{u}} \left(\boldsymbol{\phi}_{m,-}^T(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}^{\tau_j}(t_{n+1}) \right) \boldsymbol{\phi}_{m,-}^T \,\mathrm{d}\mathbf{x}}_{\underline{\boldsymbol{M}}_{p,m}(\underline{\boldsymbol{U}}_{m,-,-}^{\tau_j}(t_{n+1}))} \\ &\cdot \frac{1}{\Delta\tau_{m,j}} \left[\underline{\boldsymbol{U}}_{m,-,-}^{\tau_{j+1}}(t_{n+1}) - \underline{\boldsymbol{U}}_{m,-,-}^{\tau_j}(t_{n+1}) \right] \,,\end{aligned} \tag{10.6}$$

where

- $\underline{\boldsymbol{U}}_{m,-,-}^{\tau_j}(t_{n+1})$ are the nodal DOF in element $\mathbb{T}_m$, at the physical time t_{n+1} and the pseudo-temporal time τ_j.
- $\underline{\boldsymbol{M}}_{p,m}$ is the modified mass matrix including the preconditioning matrix evaluated at the pseudo-temporal time τ_j.
- $\Delta\tau_{m,j}$ is the elementwize pseudo-time step controlling the convergence toward the steady-state solution in pseudo-time.

[1] Kollermeier (2012, Sec. 3.5.2) explains that, for a linear system, if the diagonal values of the matrix are larger than the sum of the non-diagonal entries in the same row, this matrix is diagonally dominant. Thus adding a diagonal matrix via the addition of the pseudo-temporal term yields better conditioning numbers and improves the solvability of iterative linear solvers.

The problem to be solved for an unsteady simulation becomes:

1: **for** each physical time t_{n+1} **do**
2: | **for** each pseudo-time τ_{j+1} **do**
3: | | Solve for $\underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1})$

$$\mathcal{N}^{\text{pseudo},\Delta\tau}(\underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1})) + \mathcal{N}(\underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1})) = \mathbf{0} \qquad [10.7]$$

4: | | $\tau_{j+1} = \tau_j + \Delta\tau_j$
5: | **end for**
6: | Set $\underline{\boldsymbol{U}}((t_{n+1})) \leftarrow \underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1})$
7: | $t_{n+1} = t_n + \Delta t$
8: **end for**

For a steady simulation, only the inner loop in pseudo-time is kept and $t_{n+1} = t_\infty$. Equation 10.7 is a non-linear system in the unknowns $\underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1})$ that can be rewritten as: find $\underline{\boldsymbol{U}}^*$ such that

$$\mathcal{R}(\underline{\boldsymbol{U}}^*) = \mathbf{0}\,,$$

with

$$\mathcal{R}(\underline{\boldsymbol{U}}^*) = \frac{1}{\Delta\tau_j}\left[\underline{\boldsymbol{U}}^* - \underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right] + \underline{\boldsymbol{M}}_p^{-1}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\ \mathcal{N}(\underline{\boldsymbol{U}}^*)\,.$$

Linearization procedure

Solving the non-linear system would in theory necessitate an iterative process to obtain an accurate solution. However, that is not our goal here. What we want to achieve is a fast convergence to the next physical solution $\underline{\boldsymbol{U}}(t_{n+1})$ that is theoretically given by

$$\underline{\boldsymbol{U}}(t_{n+1}) = \lim_{j\to\infty} \underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1})\,,$$

with $\tau_{j+1} = \tau_j + \Delta\tau_j$. We are therefore not interested in getting an accurate solution. What we do is to linearize the system and obtain the solution at the next pseudo-time step by solving the following linear system:

$$\frac{\partial\mathcal{R}}{\partial\underline{\boldsymbol{U}}^*}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\left[\underline{\boldsymbol{U}}^* - \underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right] = -\mathcal{R}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\,.$$

This can be seen as performing only a single Newton iteration per pseudo-time step, with as output $\underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1}) = \underline{\boldsymbol{U}}^*$. The Jacobian matrix is obtained as

$$\frac{\partial\mathcal{R}}{\partial\underline{\boldsymbol{U}}^*}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right) = \frac{\underline{\boldsymbol{I}}}{\Delta\tau_j} + \underline{\boldsymbol{M}}_p^{-1}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\frac{\partial\mathcal{N}}{\partial\underline{\boldsymbol{U}}^*}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\,,$$

in which $\partial\mathcal{N}/\partial\underline{\boldsymbol{U}}^*$ is the Jacobian of the residual. At the end, the linear system that needs to be solved at each pseudo-time iteration can be written as

$$\left[\frac{\underline{\boldsymbol{I}}}{\Delta\tau_j} + \underline{\boldsymbol{M}}_p^{-1}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\ \frac{\partial\mathcal{N}}{\partial\underline{\boldsymbol{U}}}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\right]\Delta\underline{\boldsymbol{U}}^{\tau_{j+1}} = -\underline{\boldsymbol{M}}_p^{-1}\left(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})\right)\ \mathcal{N}(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}))\,, \tag{10.8}$$

such that the next iterate is obtained as

$$\underline{\boldsymbol{U}}^{\tau_{j+1}}(t_{n+1}) = \underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}) + \Delta\underline{\boldsymbol{U}}^{\tau_{j+1}}\,. \tag{10.9}$$

Computation of the residual Jacobian The `Argo` code has been structured in such a way that as many contributions to the global Jacobian matrix as possible are expressed analytically. The other contributions for which an analytical expression is not available, typically from the Riemann solver fluxes (similar to Equation 9.36 for XDG iso-o term) or the tensorial penalty term (similar to Equation 9.39 for XDG iso-o term), are computed based on finite differences. In addition to those contributions, in the class dealing with general EOS conservation laws implemented in this work, we also compute via FD the derivatives of the diffusive transpose term with respect to the solution vector (similar to Equation 9.40 for XDG iso-o term) and the ones of the diffusive flux (similar to Equation 9.37 for XDG iso-o term) in case variables others than the primitive p–$\mathbf{v}$–T set are used. Although not detailed in this manuscript, the assembly steps of the local Jacobians are essentially the same as the ones for the residual detailed in Section 5.2.4.

Linear solver

In this work, the linear system at each pseudo time step is most of the time solved via a direct Gauss solver with partial pivoting relying on a lower-upper (LU) factorization, implemented directly inside `Argo` by Koen Hillewaert. For a few cases, an iterative GMRES solver combined with an incomplete LU preconditioning, described in the work of Hillewaert (2013, Ch. 4.1), is also considered. Because this solver was seldom used and that the improvement of linear solvers was not in the scope of this thesis, it is not detailed here. The interested reader can consult Saad (2003) for more information about Krylov subspace methods and preconditioning techniques. There are two reasons why the majority of the two-phase unsteady simulations presented in this work and involving gas-liquid phases have been performed with the direct solver. The first one is that we are considering relatively small and coarse computational mesh in this work, which allows the use of a direct solver, which would of course not be scalable to bigger systems. The second reason is that we have observed in many situations that the robustness of the direct solver was necessary to ensure the convergence. The use of the GMRES solver often led to the divergence of the computations. The most probable reason for this is the highly

ill-conditioned nature of the global linear system, which should be addressed by a proper preconditioner to ensure the convergence of the Newton-Krylov solver, as noted by Weston et al. (2019). Although tested on a few test cases, it seems that the time-preconditioning introduced through the matrix $\underline{\boldsymbol{M}}_p$ associated to the pseudo-time term (combined with the use of primitive variables) is not sufficient. As pointed out by Nourgaliev et al. (2016), finding a good preconditioner such that the eigenvalues of the iteration matrix are efficiently clustered "is often a combination of art, science, and intuition". The design of such preconditioner is beyond the scope of this work.

Choice of pseudo-time step

Solving directly Equation 10.5 for the unknown $\underline{\boldsymbol{U}}(t_{n+1})$ via Newton's iterations would lead to a very similar result. The only differences are the addition of the term $\underline{\boldsymbol{I}}/\Delta\tau_j$ to the Jacobian of the system and the multiplication of the residual by the inverse of the preconditioning mass matrix. Those modifications enhance the condition number of the system by making the Jacobian more diagonally dominant and improves the robustness of the Newton's method by underrelaxing the correction $\Delta\underline{\boldsymbol{U}}^* = \underline{\boldsymbol{U}}^* - \underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})$. On the downside, it may slow down the convergence of the non-linear iterations because the Jacobian matrix is perturbed compared to its exact expression. But this perturbation is controlled by adjusting $\Delta\tau_j$ at every inner iteration. This is achieved by computing elementwize the pseudo-time step, at the jth-iteration, such that a constant CFL number is maintained across the domain, as explained by Hillewaert (2013, Sec. 4.1):

$$\Delta\tau_{m,j} = \frac{\mathrm{CFL}_j}{\lambda^c_{m,j} + \lambda^d_{m,j}}\,, \qquad [10.10]$$

where λ^c and λ^d are the estimated spectral radii of the convective and diffusive operators that depend on the local flow solution, the interpolation order and the element shape. The convective spectral radius is computed from Equation 5.71 when time preconditioning is applied to rescale the eigenvalues of the system. The choice of the CFL number depends on whether the simulation is unsteady or steady:

- *Unsteady*: in unsteady simulations, the CFL is set to a high constant value, typically
$$\mathrm{CFL}_j = 1\mathrm{e}10\,.$$
- *Steady*: in steady iterations, the pseudo-time step is adjusted in an adaptive manner such that it is reduced when the solver encounters difficulties and increased when the solver experiences no problem. To recuperate convergence, the CFL number is progressively increased from a fixed lower bound CFL_{min} to an upper bound CFL_{max} as the residual is decreased

according to

$$\mathrm{CFL}_j = \min\left\{\mathrm{CFL}_{max}\,,\ \max\left[\mathrm{CFL}_{min}\,,\ \mathrm{CFL}_{min} \times \left(\frac{\|\mathcal{R}(\underline{\boldsymbol{U}}^{\tau_0})\|}{\|\mathcal{R}(\underline{\boldsymbol{U}}^{\tau_j})\|}\right)^{\omega}\right]\right\}\,,$$

where the exponent ω is usually chosen between 1/2 and 1.

Note that we retrieve the standard Newton's method for $\Delta\tau_j \to \infty$.

10.1.3 Use of non-conservative sets of variables

Mapping versus transformation matrix formulations

In Section 5.3.2, we discussed the two ways of expressing the governing equations in terms of another set of variables than the conservative one, namely (i) the use of a mapping function from the conservative set to another set, keeping the original equation unchanged, (ii) the multiplication of the temporal term by a transformation matrix via the chain rule. At the continuous level, those two formulations are perfectly equivalent. At the discrete level however, the assembly of the temporal residual differs. The first approach is the one implemented in the solver and the corresponding temporal residual expression is given by Equation 10.2. The second approach corresponds in fact to the discretization of the pseudo-temporal term, Equation 10.6, with the preconditioning matrix replaced by the transformation matrix, i.e.

$$\underline{\boldsymbol{P}}_{\mathbf{u}}\left(\boldsymbol{\phi}^T_{m,-}(\mathbf{x})\,\underline{\boldsymbol{U}}_{m,-,-}(t)\right) = \frac{\partial \mathbf{u}^{\mathrm{cons}}}{\partial \mathbf{u}}\left(\boldsymbol{\phi}^T_{m,-}(\mathbf{x})\,\underline{\boldsymbol{U}}_{m,-,-}(t)\right)\,.$$

Thus, the second approach requires the assembly of a transformation mass matrix, that depends on the time-dependent unknown DOF. Contrary to the preconditioning mass matrix $\underline{\boldsymbol{M}}_p$ which can be taken as dependent on the previous pseudo-time iterate, the transformation matrix should depend on the current non-linear iterate. This implies that we would need to take into account its dependence to the solution vector in the Jacobian expression, as done e.g. in F. Bassi et al. (2015). The added simplicity of the first approach is preferred here. Moreover, since the residual is still written in the conservative form even at the discrete level, the first formulation ensures that the conservation will be satisfied to the chosen tolerance level upon convergence of the non-linear iterations, as noted by Nourgaliev et al. (2016).

Assembly of the temporal residual and its Jacobian

While the assembly of the spatial residual is not affected by a change of working variables (we should of course make sure that the fluxes are computed based on the selected variables set, as described in Section 5.3.2), the assembly of the temporal residual differs between conservative variables and others. For

conservative variables, once the mass matrix has been assembled, it is straightforward to compute the elementwize temporal residual using Equation 10.3. For other variables sets, the assembly of Equation 10.2 closely follows the one of the face and boundary residuals detailed in Section 5.2.4:

1. *Collocation*: the solution for each time level involved in the time stepping scheme is first interpolated to the positions of the quadrature points in the reference element, $\boldsymbol{\xi}^Q$, via

$$\mathbf{u}_m(t_{n+i}, \boldsymbol{\xi}^Q) = \begin{bmatrix} \phi_{m,1}(\boldsymbol{\xi}^Q_1) & \dots & \phi_{m,N_n}(\boldsymbol{\xi}^Q_1) \\ \vdots & & \vdots \\ \phi_{m,1}(\boldsymbol{\xi}^Q_{N_q}) & \dots & \phi_{m,N_n}(\boldsymbol{\xi}^Q_{N_q}) \end{bmatrix} \cdot \underbrace{\begin{bmatrix} U_{m,1,1}(t_{n+i}) & \dots & U_{m,1,N_f}(t_{n+i}) \\ \vdots & & \vdots \\ U_{m,N_n,1}(t_{n+i}) & \dots & U_{m,N_n,N_f}(t_{n+i}) \end{bmatrix}}_{\underline{\boldsymbol{U}}_{m,-,-}} .$$

2. *Mapping*: the solved-for solution at the QP, $\mathbf{u}_m(t_{n+i}, \boldsymbol{\xi}^Q)$, is then converted to the conservative variables by applying the mapping $\mathbf{u}^{\text{cons}}(\mathbf{u}_m(t_{n+i}, \boldsymbol{\xi}^Q))$. This is done for all the time steps to obtain the resulting matrix $\underline{\boldsymbol{U}}^{\text{cons}}_{m,Q}$:

$$\underline{\boldsymbol{U}}^{\text{cons}}_{m,Q} \mathrel{+}= \frac{\alpha_i}{\Delta t}\, \mathbf{u}^{\text{cons}}(\mathbf{u}_m(t_{n+i}, \boldsymbol{\xi}^Q)) \quad \text{for } i = \{-N_{\text{step}} + 1, \dots, 1\} .$$

3. *Redistribution*: finally, the integral in Equation 10.2 is evaluated by an efficient multiplication between a pre-assembled redistribution matrix and the matrix containing the mapped solution at the QP:

$$\mathcal{J}^{\Delta t}_m \mathrel{+}= \begin{bmatrix} \phi_{m,1}(\boldsymbol{\xi}^Q_{m,1})\, \omega_{m,1}\, |\underline{\boldsymbol{J}}^{\sigma}_m(\boldsymbol{\xi}^Q_{m,1})| & \dots & \phi_{m,1}(\boldsymbol{\xi}^Q_{m,N_{q,m}})\, \omega_{m,N_{q,m}}\, |\underline{\boldsymbol{J}}^{\sigma}_m(\boldsymbol{\xi}^Q_{m,N_{q,m}})| \\ \vdots & & \vdots \\ \phi_{m,N_{n,m}}(\boldsymbol{\xi}^Q_{m,1})\, \omega_{m,1}\, |\underline{\boldsymbol{J}}^{\sigma}_e(\boldsymbol{\xi}^Q_{m,1})| & \dots & \phi_{m,N_{n,m}}(\boldsymbol{\xi}^Q_{m,N_{q,m}})\, \omega_{m,N_{q,m}}\, |\underline{\boldsymbol{J}}^{\sigma}_e(\boldsymbol{\xi}^Q_{m,N_{q,m}})| \end{bmatrix} \underline{\boldsymbol{U}}^{\text{cons}}_{m,Q} .$$

Concerning the contribution of $\mathcal{J}^{\Delta t}_m$ to the global Jacobian matrix Jacobian, it can be obtained as

$$\frac{\partial \mathcal{J}^{\Delta t}_m}{\partial \underline{\boldsymbol{U}}_{m,-,-}(t_{n+1})} = \frac{\alpha_1}{\Delta t} \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \underbrace{\frac{\partial \mathbf{u}^{\text{cons}}\left(\boldsymbol{\phi}^T_{m,-}(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}(t_{n+i})\right)}{\partial \underline{\boldsymbol{U}}_{m,-,-}(t_{n+1})}}_{\partial \mathbf{u}^{\text{cons}}/\partial \mathbf{u}} \underbrace{\frac{\partial \boldsymbol{\phi}^T_{m,-}(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,-}(t_{n+1})}{\partial \underline{\boldsymbol{U}}_{m,-,-}(t_{n+1})}}_{=\boldsymbol{\phi}^T_{m,-}(\mathbf{x})} \mathrm{d}\mathbf{x} , \tag{10.11}$$

where $\partial \mathbf{u}^{\text{cons}} / \partial \mathbf{u}$ is the transformation matrix from conservative to another set of variables, already introduced in Equation 5.66.

Let's conclude this section by indicating that we do not present here an algorithm listing all the steps involved in a time step of the spatial DG + temporal discretization since the extended version of this algorithm for the XDG discretization can be found in Section 10.2.5.

10.2 Time integration with moving interfaces for the XDG method

10.2.1 Overview of methods to handle evolving cut-mesh topologies

As explained in Utz (2018, Sec. 3.5.2), although the underlying mesh remains fixed in the XDGM, the domains occupied by each phase are time-dependent. This poses some conceptual issues for the time discretization of the spatially discretized equations with classical time-stepping schemes. This observation holds true for any unfitted methods applied to transient problems and relying on an Eulerian reference frame for the description of the fluid phases. Indeed, classical time-stepping schemes aim at calculating the solution at the next time step based on the solution(s) previously computed. Because the topology of the cut-mesh is evolving with time and that the spatial approximation is adapted on cut elements, combining the solutions from different time levels may become problematic. This is exemplified in the next section. In this context, several approaches have been designed:

- *Space-time domain*: instead of applying separated space and time discretizations, time is considered as an additional dimension. A simultaneous discretization of spatial and temporal variables is done using basis functions depending on both space and time. This approach provides the most accurate results at the cost of involving much larger systems and a complex implementation in an existing FE or DG code framework.
- *Rothe method*: this technique consists in discretizing the equations first in time and then in space. When solving for a discrete approximation to the new time step, all terms are then evaluated at the corresponding new time level, as described in Benedikt Schott (2017, Sec. 3.6.3), which avoids the compatibility issues when combining solutions on different cut-cell topologies. This is however not easy to apply because the integration of functions defined at different time levels must be handled, as explained in Pierre Schrooyen et al. (2013).
- *Splitting approach*: sticking to the more traditional approach of first discretizing in space and then using a common time-stepping scheme, the easiest approach is to split the interface evolution from the governing equations. We can therefore integrate in time the equations considering

a fixed interface and then update the interface position in a second step. This however does not solve the issue of combining solutions on evolving cut-mesh topologies. That is why interpolation between time levels is needed to compensate the mismatch between the approximation spaces. In this work, we follow the approach proposed by Kummer et al. (2018) that exploits the cell-agglomeration technique described earlier. This is explained in the next section.

- *Moving interface approach*: because of its simplicity, the splitting approach is appealing. However, it limits the time accuracy of the global coupled problem, as discussed in Kummer et al. (2018). A method that allows to regain the theoretical temporal convergence rate without the difficulties of the two first approaches is proposed in Kummer et al. (2018) and is also described in the next section.

The development of strategies for moving interface problems in an Eulerian framework is still a very active research area, as shown by recent publications such as Lehrenfeld and M. Olshanskii (2019), Burman et al. (2022) or M. A. Olshanskii et al. (2023). We can remark that interface-fitted meshes ALE-based formulations do not suffer from these difficulties, at least as long as the mesh topology is kept unchanged. Indeed, as noted by Thomas-Peter Fries and Zilian (2009), in ALE the time dependence of the shape functions is shifted to a so-called mesh velocity and all the shape functions are evaluated at the current time. A cumbersome evaluation of integrals involving time-varying functional spaces is therefore avoided.

10.2.2 Splitting and moving interface approaches to integrate over time-varying domains

In Section 8.2.2, we discussed the modifications that the XDG discretization brings to the standard DG discretization, focusing on the spatial part of the residual. We turn our attention now to the discretization of the temporal residual, Equation 5.17, in the most general case of a moving interface problem. The extension of the function spaces on the cut elements to be able to capture discontinuous solutions poses two main challenges for the temporal discretization: (i) the time-dependence of the domains of integration in the weak form, (ii) the evolution of the cut topology and the time-dependence of the functional space. The first one is addressed in this section, the second one in the next section. As pointed out by Benedikt Schott (2017, Sec. 3.6.3.1), in contrast to space-time methods, in our approach the time-dependent change of the cut finite element approximation spaces is not tracked continuously in time. This is an advantage in terms of ease of implementation and size of the system to solve. However, this poses also some difficulties because the extended discrete finite dimensional spaces do no match in general between different time levels since the domains of integration in the cut elements evolve with the interface motion. We restrict our attention to the case where the interface motion remains

constrained within a single element; the situation where it changes element is linked to the second challenge. The two approaches we implemented to handle the first challenge are discussed below.

Splitting/projection approach

In the splitting approach, we split the motion of the interface from the time integration of the fluid equations (first interface, second fluid) such that the temporal residual must be evaluated at the current time $t = t_n$ as if the interface would be fixed. Let's consider the spatially discretized temporal residual in a cut element $\mathbb{T}_m$ for the phase k. If we assume that we are manipulating the conservative variables, it comes:

$$\begin{aligned}\mathcal{T}_{m,k} &= \int_{\mathbb{T}_{m,k}(t_n)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \frac{\partial \mathbf{u}_k(\mathbf{x},t)}{\partial t}\, \mathrm{d}\mathbf{x} \\ &= \int_{\mathbb{T}_{m,k}(t_n)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \frac{\partial}{\partial t}\left[\boldsymbol{\phi}^T_{m,-}(\mathbf{x}) \underline{\boldsymbol{U}}_{m,-,k,-}(t)\right]\, \mathrm{d}\mathbf{x} \\ &= \int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x})\, \mathrm{H}(\mathbf{x},\varphi(t_n)) \cdot \frac{\partial}{\partial t}\left[\boldsymbol{\phi}^T_{m,-}(\mathbf{x})\, \mathrm{H}(\mathbf{x},\varphi(t))\, \underline{\boldsymbol{U}}_{m,-,k,-}(t)\right]\, \mathrm{d}\mathbf{x}\,,\end{aligned}$$

where the last equality transfers the integration over the cut volume into the truncation of the shape functions by Heaviside function $\mathrm{H}(\varphi)$ depending on the interface location, as already introduced at the end of Section 8.2.1. If an Euler time stepping scheme is chosen, this yields:

$$\begin{aligned}\mathcal{T}^{\Delta t}_{m,k} = \int_{\mathbb{T}_m} &\boldsymbol{\phi}_{m,-}(\mathbf{x})\, \boldsymbol{\phi}^T_{m,-}(\mathbf{x}) \mathrm{H}(\mathbf{x},\varphi(t_n)) \\ &\frac{1}{\Delta t}\left[\underline{\boldsymbol{U}}_{m,-,k,-}(t_n)\, \mathrm{H}(\mathbf{x},\varphi(t_n)) - \underline{\boldsymbol{U}}_{m,-,k,-}(t_{n-1})\, \mathrm{H}(\mathbf{x},\varphi(t_{n-1}))\right]\, \mathrm{d}\mathbf{x}\,.\end{aligned}$$

That is where we notice that the integral above involves the product of functions at two distinct time levels:

$$-\int_{\mathbb{T}_m} \boldsymbol{\phi}_{m,-}(\mathbf{x})\, \boldsymbol{\phi}^T_{m,-}(\mathbf{x}) \mathrm{H}(\mathbf{x},\varphi(t_n))\, \mathrm{H}(\mathbf{x},\varphi(t_{n-1}))\, \mathrm{d}\mathbf{x}\, \underline{\boldsymbol{U}}_{m,-,k,-}(t_{n-1})\,.$$

As explained by Thomas-Peter Fries and Zilian (2009), there are two possibilities to handle this situation. The first one is to consider in the quadrature the two distinct locations of the discontinuities inside the domain. The numerical integration thus becomes more elaborate. The second one is to extrapolate (resp. interpolate), depending on whether the support of the basis function at t_{n-1} is smaller (resp. bigger) than the one at t_n, the solution at t_{n-1} onto the cut volume at t_n, which amounts to set $\mathrm{H}(\mathbf{x},\varphi(t_{n-1})) = \mathrm{H}(\mathbf{x},\varphi(t_n))$ in the above integral. This can be done in practice by simply expressing the solution $\mathbf{u}_k(\mathbf{x},t_{n-1})$ by keeping the same DG coefficients $\underline{\boldsymbol{U}}_{m,-,k,-}(t_{n-1})$ but replacing the truncated basis functions with the ones at the current time. This is this second simplified approach that is followed in this work such that the temporal residual will take

an expression similar to the following one (the exact expression is determined by the choice of the time integration scheme and the working variables):

$$\mathcal{J}_{m,k}^{\Delta t} = \underline{\boldsymbol{M}}_{m,k}(t_n)\frac{1}{\Delta t}\left[\underline{\boldsymbol{U}}_{m,-,k,-}(t_n) - \underline{\boldsymbol{U}}_{m,-,k,-}(t_{n-1})\right]. \qquad [10.12]$$

Moving interface/simplified space-time approach

The second approach comes directly from the ingenious idea presented in Kummer et al. (2018). With the goal of restoring the accuracy lost in the first approach due to the splitting of space and time and the simplified extrapolation between time levels, we start here by considering the possibility of performing space and time discretizations together. This implies to treat the spatial and temporal dimensions equivalently and build a space-time domain

$$\Omega^{\text{st}} = [0,T]\times\Omega\,,$$
$$\Omega_k^{\text{st}} = [0,T]\times\Omega_k(t)\,,$$

discretized, in all the dimensions, by space-time elements based on Equation 5.5,

$$\Omega^{\text{st}} \approx \left[t_0,\dots,t_n\dots,t_{N_{\Delta t}}\right]\times\bigcup_{m=1}^{N_T}\mathbb{T}_m = \bigcup_{m^*=1}^{N_T\times N_{\Delta t}}\mathbb{T}_{m^*}^{\text{st}}\,.$$

We extend this definition to consider space-time elements strictly contained in a given phase k:

$$\mathbb{T}_{m^*,k}^{\text{st}} = \left\{\mathbf{x}^{\text{st}} = (t,\mathbf{x})\in\Omega^{\text{st}} \mid t_{n-1} < t < t_n,\ \mathbf{x}\in\mathbb{T}_m\cap\Omega_k^{\text{st}}(t)\right\}.$$

In order to derive the weak formulation, we should start again from the prototype governing equation we want to discretize in each phase, i.e.

$$\frac{\partial\mathbf{u}_k}{\partial t} + \nabla\cdot\underline{\boldsymbol{F}}_k(\mathbf{u}_k) = \mathbf{0}\,,$$

and rewrite it under a form suited for space-time domain,

$$\begin{bmatrix}\partial/\partial t\\ \nabla\end{bmatrix}\cdot\begin{bmatrix}\mathbf{u}_k\\ \underline{\boldsymbol{F}}_k(\mathbf{u}_k)\end{bmatrix} = \nabla^{\text{st}}\cdot\underline{\boldsymbol{F}}_k^{\text{st}}(\mathbf{u}_k) = \mathbf{0}\,.$$

The procedure is then identical to the standard DG method: we multiply the equation by a set of test functions, integrate over the space-time domain and apply integration-by-parts. Utz (2018, Sec. 3.5.2) points out that the key point here is that contrary to classical space-time methods where space-time test functions are considered, here we keep the standard DG test functions (which are time-independent) such that in a given space-time element, the contribution

to the weak form writes

$$\int_{\mathbb{T}^{\mathrm{st}}_{m^*,k}} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(\nabla^{\mathrm{st}} \cdot \underline{\boldsymbol{F}}^{\mathrm{st}}_k(\mathbf{u}_k)\right) \,\mathrm{d}\mathbf{x}\,,$$

which can be transformed by integration-by-parts as

$$\oint_{\partial\mathbb{T}^{\mathrm{st}}_{m^*,k}} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(\underline{\boldsymbol{F}}^{\mathrm{st}}_k(\mathbf{u}_k) \cdot \mathbf{n}^{\mathrm{st}}\right) \,\mathrm{d}s^{\mathrm{st}} - \int_{\mathbb{T}^{\mathrm{st}}_{m^*,k}} \nabla^{\mathrm{st}} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \underline{\boldsymbol{F}}^{\mathrm{st}}_k(\mathbf{u}_k) \,\mathrm{d}\mathbf{x}^{\mathrm{st}}\,. \quad [10.13]$$

Let's remark that because the shape functions are only dependent on the spatial dimension, the second integrand becomes

$$\nabla^{\mathrm{st}} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \underline{\boldsymbol{F}}^{\mathrm{st}}_k(\mathbf{u}_k) = \nabla \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \underline{\boldsymbol{F}}_k(\mathbf{u}_k)\,. \quad [10.14]$$

We can write the space-time outer normals to the different faces of the space-time element in the coordinate system $\mathbf{x}^{\mathrm{st}} = (t, \mathbf{x})$ as:

$$\mathbf{n}^{\mathrm{st}} = \begin{cases} \frac{1}{\sqrt{\|\mathbf{n}_{\Gamma,m,k}\|^2 + \mathbf{v}^2_{\Gamma_m,n}}} (-v_{\Gamma_m,n}, \mathbf{n}_{\Gamma,m,k})^T & \text{on } \Gamma^{\mathrm{st}}\,, \\ (-1, \mathbf{0})^T & \text{on } \mathbb{T}_{m,k}(t_{n-1})\,, \\ (1, \mathbf{0})^T & \text{on } \mathbb{T}_{m,k}(t_n)\,, \\ (0, \mathbf{n}_{\partial\mathbb{T}_{m,k}})^T & \text{on } \partial\mathbb{T}^{\mathrm{st}}_{m^*,k} \setminus \{\Gamma^{\mathrm{st}}, \mathbb{T}_{m,k}(t_{n-1}), \mathbb{T}_{m,k}(t_n)\}\,, \end{cases}$$

where in the first equation, we have used the definition of the interface normal and velocity,

$$v_{\Gamma_m,n} = \frac{\partial \Gamma^{\mathrm{st}}_m(\mathbf{x}, t)}{\partial t}\,, \quad \mathbf{n}_{\Gamma,m,k} = \frac{\partial \Gamma^{\mathrm{st}}_m(\mathbf{x}, t)}{\partial \mathbf{x}}\,,$$

and the multiplying factor is there to obtain a unit normal vector. For a vector $\mathbf{q} = (q_t, \mathbf{q}_{\mathbf{x}})$, the next step is to decompose the space-time integrals of the equations above as done by Kummer et al. (2018):

$$\begin{aligned} &\int_{\mathbb{T}^{\mathrm{st}}_{m^*,k}} \mathbf{q} \,\mathrm{d}\mathbf{x}^{\mathrm{st}} = \int_{t_{n-1}}^{t_n} \int_{\mathbb{T}_{m,k}(t)} \mathbf{q} \,\mathrm{d}\mathbf{x} \,\mathrm{d}t\,, \\ &\oint_{\partial\mathbb{T}^{\mathrm{st}}_{m^*,k}} \mathbf{q} \cdot \mathbf{n}^{\mathrm{st}} \,\mathrm{d}s^{\mathrm{st}} \\ &= \int_{t_{n-1}}^{t_n} \int_{\Gamma_m(t)} \frac{1}{\sqrt{1 + v^2_{\Gamma_m,n}}} \left[-\mathbf{q}_t \, v_{\Gamma_m,n} + \mathbf{q}_{\mathbf{x}} \cdot \mathbf{n}_{\Gamma,m,k}\right] \sqrt{1 + v^2_{\Gamma_m,n}} \,\mathrm{d}s \,\mathrm{d}t \\ &+ \int_{t_{n-1}}^{t_n} \int_{\partial\mathbb{T}_{m,k}(t) \setminus \Gamma_m(t)} \mathbf{q}_{\mathbf{x}} \cdot \mathbf{n}_{\partial\mathbb{T}_{m,k}} \,\mathrm{d}s \,\mathrm{d}t \\ &+ \int_{\mathbb{T}_{m,k}(t_n)} \mathbf{q}_t \,\mathrm{d}\mathbf{x} - \int_{\mathbb{T}_{m,k}(t_{n-1})} \mathbf{q}_t \,\mathrm{d}\mathbf{x}\,, \end{aligned}$$

where the factor $\sqrt{1 + v^2_{\Gamma_m,n}}$ on the first line of the surface integral scales the elemental surface area in order to integrate on the inclined curve defined by $\Gamma_m(t)$

in the space-time domain. Injecting those integral identities in Equation 10.13 and using Equation 10.14, we get

$$\begin{aligned}
&\int_{t_{n-1}}^{t_n} \Bigg[\int_{\Gamma(t)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(-\mathbf{u}_k \, v_{\Gamma_m,n} + \underline{\boldsymbol{F}}_k(\mathbf{u}_k) \cdot \mathbf{n}_{\Gamma,m,k} \right) \, \mathrm{d}s \\
&+ \int_{\partial \mathcal{T}_{m,k}(t) \setminus \Gamma_m(t)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(\underline{\boldsymbol{F}}_k(\mathbf{u}_k) \cdot \mathbf{n}_{\partial \mathcal{T}_{m,k}} \right) \, \mathrm{d}s \Bigg] \, \mathrm{d}t \\
&+ \int_{\mathcal{T}_{m,k}(t_n)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \mathbf{u}_k(\mathbf{x}, t_n) \, \mathrm{d}\mathbf{x} - \int_{\mathcal{T}_{m,k}(t_{n-1})} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \mathbf{u}_k(\mathbf{x}, t_{n-1}) \, \mathrm{d}\mathbf{x} \\
&- \int_{t_{n-1}}^{t_n} \int_{\mathcal{T}_{m,k}(t)} \nabla \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \underline{\boldsymbol{F}}_k(\mathbf{u}_k) \, \mathrm{d}\mathbf{x} \, \mathrm{d}t
\end{aligned} \tag{10.15}$$

In order to obtain a form suited for the XDG spatial discretization that can be discretized in time in a second step following the method of lines, we take the temporal derivative of Equation 10.15 and let $t_n \to t_{n-1}$ to get rid of the specific time integration scheme. This yields

$$\begin{aligned}
&\frac{\partial}{\partial t} \int_{\mathcal{T}_{m,k}(t)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \mathbf{u}_k(\mathbf{x}, t) \, \mathrm{d}\mathbf{x} - \int_{\mathcal{T}_{m,k}(t)} \nabla \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \underline{\boldsymbol{F}}_k(\mathbf{u}_k) \, \mathrm{d}\mathbf{x} \\
&+ \int_{\Gamma(t)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(-\mathbf{u}_k \, v_{\Gamma_m,n} + \underline{\boldsymbol{F}}_k(\mathbf{u}_k) \cdot \mathbf{n}_{\Gamma,m,k} \right) \, \mathrm{d}s \\
&+ \int_{\partial \mathcal{T}_{m,k}(t) \setminus \Gamma_m(t)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \left(\underline{\boldsymbol{F}}_k(\mathbf{u}_k) \cdot \mathbf{n}_{\partial \mathcal{T}_{m,k}} \right) \, \mathrm{d}s \, .
\end{aligned} \tag{10.16}$$

When transferring this local element-based formulation to a global face-based formulation, we obtain an XDG discretization identical to the one given in Equation 8.4, Equation 8.5 and Equation 8.6, with two notable exceptions:

- *Time-dependent mass matrix*: the temporal term in Equation 10.16, namely the first term, involves the temporal derivative of a time-dependent domain. This implies that all the ingredients to assemble the temporal residual such as the mass matrix or the quadrature rules will need to be computed over several time levels when time discretization will be applied. To illustrate that, consider again an Euler scheme with conservative working variables such that the temporal residual can be written as

$$\mathcal{J}_{m,k}^{\Delta t} = \frac{1}{\Delta t} \left[\underline{\boldsymbol{M}}_{m,k}(t_n) \, \underline{\boldsymbol{U}}_{m,-,k,-}(t_n) - \underline{\boldsymbol{M}}_{m,k}(t_{n-1}) \, \underline{\boldsymbol{U}}_{m,-,k,-}(t_{n-1}) \right] . \tag{10.17}$$

  This expression can be compared to the one for the splitting approach, Equation 10.12, to realize that a mass matrix has to be computed per time level here.
- *Corrected convective numerical flux across the interface*: as before, the specification of numerical flux functions is necessary to define a unique flux across the space-time element faces, including mesh faces and the moving interface. Across the mesh faces, nothing changes compared to the XDG discretization presented previously. However, on the moving interface,

we should now specify a convective numerical flux function of the form

$$\hat{\underline{\boldsymbol{F}}}_{\Gamma,k}^{c,\mathrm{st}} \cdot \mathbf{n}_{\Gamma,k} = \left[\mathbf{u}_k \, v_{\Gamma,n} + \underline{\boldsymbol{F}}_k^c(\mathbf{u}_k) \cdot \mathbf{n}_{\Gamma,k}\right]^{\hat{\bullet}} ,$$

where the notation $[]^{\hat{\bullet}}$ is used to indicate that it is the complete expression as a whole that needs to be replaced by a flux function. The additional first term on the right-hand side can be seen as an ALE-type contribution that appears because we consider the motion of the interface in the space-time formulation, contrary to the splitting approach for which the interface is assumed to be fixed when discretizing the fluid equations. Kummer et al. (2018) explain that there are two ways to specify this numerical flux function. The first one is to consider the standard static flux of the splitting approach on the interface (see Equation 9.6) and correct it by the addition of the flux due to interface motion, i.e.

$$\hat{\underline{\boldsymbol{F}}}_{\Gamma,k}^{c,\mathrm{st}} \cdot \mathbf{n}_{\Gamma,k} = \hat{\underline{\boldsymbol{F}}}_{\Gamma,k}^{c} \cdot \mathbf{n}_{\Gamma,k} - \hat{\mathbf{u}}_k \, v_{\Gamma,n} \, .$$

The resulting expression in the framework of a multiphase Riemann solver is given in Equation 9.12. The second possibility to define the numerical flux function is to consider the flux in a moving frame. This has to be done specifically for the equation of interest. We adopted here the first approach for its simplicity and generality.

On the accuracy of the time integration scheme An important remark to formulate here is that the *a priori* error estimate analysis of the DG method typically assumes exact integration of the volumetric and surface integrals that appear in the weak formulation, as noted by Pazner and Haut (2021). In practice, it is rarely the case that this condition can be met but as long as the quadrature errors are high-order, this should not compromise the theoretical order of accuracy of the scheme. This consideration should also apply to the space-time formulation in Equation 10.13. Since we first perform the spatial integration before the temporal one, Kummer et al. (2018) finds out that the temporal integration scheme should ideally be of order $2p$, with p the spatial interpolation degree, in order to prevent the quadrature errors from spoiling the global convergence. This poses serious constraints on the choice of time stepping scheme and may limit the applicability of the moving interface approach.

10.2.3 Cell-agglomeration to ensure constant cut topology over time levels

In Section 8.3.1, we described the flagging procedure of the agglomerated cells for which we described three criteria to determine if an element should be flagged as a source. Two of them are based on the concepts of appearing and disappearing elements. Those two criteria are added only to deal with evolving cut topology and we shall now motivate them based on the following example.

Since the requirements in terms of topology preservation differ between the splitting and moving interface approaches introduced before, we analyze them separately.

Example 10.1: Assembly of the temporal residual with changing cut topologies

The example is based on Figure 10.1 and Figure 10.2.

Inconsistencies On the first sketch, in the first row, let's say we want to assemble the temporal residual for the phase $k = 2$ in the element T_3 at the current time step t_n. If we assume that we are manipulating the conservative variables, that we are using an implicit Euler scheme and that the splitting approach described earlier is used, the temporal residual at Equation 10.12 becomes:

$$\mathcal{J}_{3,2}^{\Delta t} = \underline{\boldsymbol{M}}_{3,2}(t_n)\frac{1}{\Delta t}\left[\underline{\boldsymbol{U}}_{3,-,2,-}(t_n) - \underline{\boldsymbol{U}}_{3,-,2,-}(t_{n-1})\right].$$

We directly notice that there is an inconsistency: the element T_3 at time t_{n-1} does not carry any DOF associated to phase $k = 2$. It is therefore not possible to assemble the temporal residual in this element by combining solutions from different time levels.

Splitting approach As explained by Benedikt Schott et al. (2019), directly copying the DOF from previous configurations to the current one is valid in the following two situations: (i) the interface motion remains within the same cut element over the time period (which was the situation considered in Section 10.2.3), (ii) the interface leaves the cut element such that the DOF associated to the considered phase get deactivated (what we call "disappearing" elements). However, for newly activated DOF ("appearing" elements), something has to be done, otherwise we will end up with the inconsistencies explained above. That is where the agglomeration procedure comes into play. In the present case, we overcome the inconsistency by flagging $\mathrm{T}_{3,2}$ as a source at t_n, hence avoiding the assembly of the temporal residual in this element. Instead, we will consider the solution in its associated target element, $\mathrm{T}_{2,2}$. The role of the agglomeration can therefore be seen as extending the extrapolation/interpolation operators within a single cut element to the cases where topology changes occur: the DOF of the previous time levels are extrapolated onto the agglomerated topology and the solutions are expressed based on the agglomerated shape functions at the current time. This is illustrated by the top right sketch in Figure 10.2.

The middle row of Figure 10.1 illustrates the situation where $\mathbb{T}_{2,2}$ would also need to be agglomerated because not present at time t_{n-2}, also involved in the time discretization. This would be the case for instance if a BDF2 scheme was used, which would lead to a temporal residual of the form

$$\mathcal{J}_{1,2}^{\mathrm{Agg},\Delta t} = \underline{\boldsymbol{M}}_{1,2}^{\mathrm{Agg}}(t_n)\frac{1}{\Delta t}\Big[3/2\,\underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_n) - 2\,\underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n-1}) + 1/2\,\underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n-2})\Big]. \tag{10.18}$$

Moving interface approach In addition to appearing cells, the disappearing ones also need to be identified as source in the moving interface approach, as illustrated in the bottom row of Figure 10.1 with the agglomerated elements in phase $k = 1$. The reason is that we want to ensure that the space-time domains considered in Section 10.2.3 to derive the weak formulation are free of topology changes. If we take the same example as the one for the splitting approach above, the expression of the temporal residual can be obtained from Equation 10.17 as

$$\mathcal{J}_{1,2}^{\mathrm{Agg},\Delta t} = \frac{1}{\Delta t}\Big[3/2\,\underline{\boldsymbol{M}}_{1,2}^{\mathrm{Agg}}(t_n)\,\underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_n) - 2\,\underline{\boldsymbol{M}}_{1,2}^{\mathrm{Agg}}(t_{n-1})\,\underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n-1}) + 1/2\,\underline{\boldsymbol{M}}_{1,2}^{\mathrm{Agg}}(t_{n-2})\,\underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n-2})\Big]. \tag{10.19}$$

The corresponding space-time domain from which this expression can be derived is sketched at the bottom right of Figure 10.2.

This example highlights the similarities and differences between the splitting and moving interface approaches. Among the similarities, we can mention the fact that the building of the agglomeration map has to be realized by combining the information from the different time levels. The notable difference between the two approaches is that the splitting approach only needs agglomeration data (mass and coupling matrices) at the current time while for the moving interface approach, those data must be computed and stored for each time level, as Equation 10.19 shows it. Another important remark to make concerns the dissimilarity between the assembly of the agglomerated spatial and temporal residuals. For the spatial residual, Equation 8.25, the agglomerated residual is obtained by combining pre-assembled residuals in the target and source elements. As we have seen, we cannot follow the same principle for the temporal residual since we expressly want to avoid the assembly of the residual on the source elements. That is why we need to compute the temporal residual directly on the agglomerated mesh, based on the following general formula (to be

compared to Equation 8.25 for the spatial part). For the splitting approach, we have

$$\mathcal{N}^{\mathrm{tp,Agg}}_{m,-,k,-} = \begin{cases} \mathcal{J}^{\mathrm{tp,Agg}}_{m,-,k,-} = \int_{\mathbb{T}^{\mathrm{Agg}}_{m,k}} \boldsymbol{\phi}^{\mathrm{Agg}}_{m,-}(\mathbf{x}) \cdot \frac{\partial \mathbf{u}^{\mathrm{cons}}\left(\boldsymbol{\phi}^{\mathrm{Agg},T}_{m,-}(\mathbf{x})\, \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m,-,k,-}(t)\right)}{\partial t}\, \mathrm{d}\mathbf{x}, & \text{if } \mathbb{T}_{m,k} \text{ is target}, \\ 0, & \text{if } \mathbb{T}_{m,k} \text{ is source}, \\ \mathcal{J}^{\mathrm{tp}}_{m,-,k,-} = \int_{\mathbb{T}_{m,k}} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \frac{\partial \mathbf{u}^{\mathrm{cons}}\left(\boldsymbol{\phi}^{T}_{m,-}(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,k,-}(t)\right)}{\partial t}\, \mathrm{d}\mathbf{x}, & \text{otherwise}, \end{cases} \quad [10.20]$$

where the same notation $\mathbf{u}^{\mathrm{cons}}(\mathbf{u})$ as in Equation 10.1 is used such that the formula is valid for any set of working variables. For the moving interface approach,

$$\mathcal{N}^{\mathrm{tp,Agg}}_{m,-,k,-} = \begin{cases} \mathcal{J}^{\mathrm{tp,Agg}}_{m,-,k,-} = \frac{\partial}{\partial t} \int_{\mathbb{T}^{\mathrm{Agg}}_{m,k}(t)} \boldsymbol{\phi}^{\mathrm{Agg}}_{m,-}(\mathbf{x}) \cdot \mathbf{u}^{\mathrm{cons}}\left(\boldsymbol{\phi}^{\mathrm{Agg},T}_{m,-}(\mathbf{x})\, \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m,-,k,-}(t)\right) \mathrm{d}\mathbf{x}, & \text{if } \mathbb{T}_{m,k} \text{ tar}, \\ 0, & \text{if } \mathbb{T}_{m,k} \text{ src}, \\ \mathcal{J}^{\mathrm{tp}}_{m,-,k,-} = \frac{\partial}{\partial t} \int_{\mathbb{T}_{m,k}(t)} \boldsymbol{\phi}_{m,-}(\mathbf{x}) \cdot \mathbf{u}^{\mathrm{cons}}\left(\boldsymbol{\phi}^{T}_{m,-}(\mathbf{x})\, \underline{\boldsymbol{U}}_{m,-,k,-}(t)\right) \mathrm{d}\mathbf{x}, & \text{otherwise}, \end{cases} \quad [10.21]$$

For conservative variables, the above formula are conveniently simplified by relying on the agglomerated mass matrices, as shown by Equation 10.18 and Equation 10.19. The case of general working variables is a bit more involved and discussed in Section 10.2.4.

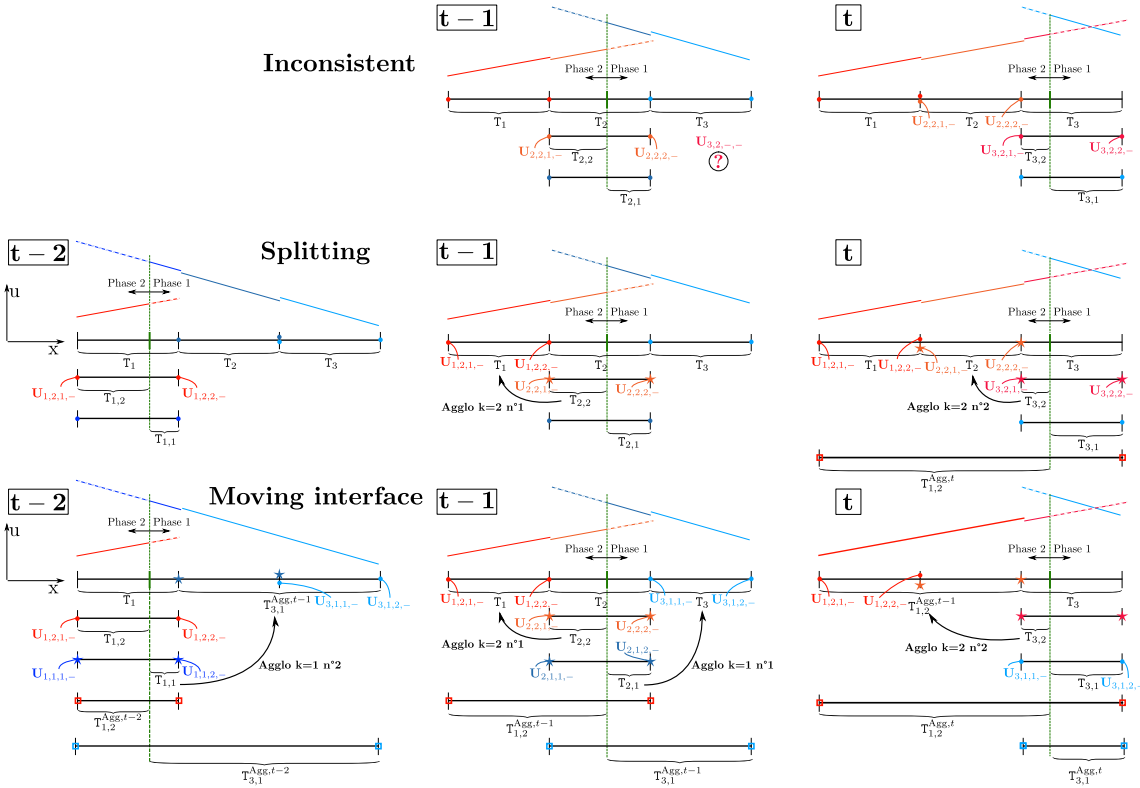


Figure 10.1 / Illustrative example of the inconsistencies that can appear during the temporal integration with moving interface problem and the two strategies solving those issues, the splitting and moving interface approaches. Dots represent DOF of normal or target elements, stars the ones of source elements.

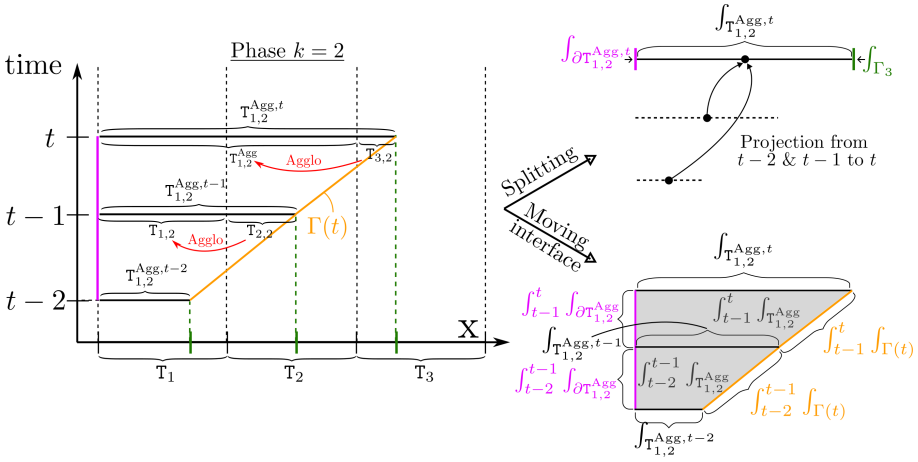


Figure 10.2 / Illustrative example from Figure 10.1 of the two approaches to discretize the space-time domain formed by the intersection between agglomerated mesh and the moving interface.

10.2.4 Assembly of the temporal residual for general sets of variables

Starting from Equation 10.20 and Equation 10.21, we discretize in time, by means of a single-stage multi-step scheme, the elementwize temporal residual in an agglomerated element associated to phase k, $\mathrm{T}^{\mathrm{Agg}}_{m1,k}$, similarly to Equation 10.2. For the splitting approach, we get

$$\begin{aligned} &\mathcal{J}^{\mathrm{tp,Agg},\Delta t}_{m,-,k,-} \\ &= \int_{\mathrm{T}^{\mathrm{Agg}}_{m,k}(t_{n+1})} \boldsymbol{\phi}^{\mathrm{Agg}}_{m,-}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\sum_{i=-N_{\mathrm{step}}+1}^{1} \alpha_i \, \mathbf{u}^{\mathrm{cons}} \left(\boldsymbol{\phi}^{\mathrm{Agg},T}_{m,-}(\mathbf{x}) \, \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m,-,k,-}(t_{n+i}) \right) \right] \mathrm{d}\mathbf{x}, \end{aligned} \tag{10.22}$$

while for the moving interface approach, we have

$$\begin{aligned} &\mathcal{J}^{\mathrm{tp,Agg},\Delta t}_{m,-,k,-} \\ &= \sum_{i=-N_{\mathrm{step}}+1}^{1} \int_{\mathrm{T}^{\mathrm{Agg}}_{m,k}(t_{n+i})} \boldsymbol{\phi}^{\mathrm{Agg}}_{m,-}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\alpha_i \, \mathbf{u}^{\mathrm{cons}} \left(\boldsymbol{\phi}^{\mathrm{Agg},T}_{m,-}(\mathbf{x}) \, \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{m,-,k,-}(t_{n+i}) \right) \right] \mathrm{d}\mathbf{x}. \end{aligned} \tag{10.23}$$

We then illustrate in practice those formulas on the example of the previous section.

Example 10.2: Assembly of the temporal residual with general sets of variables

Splitting At time t_{n+1}, the agglomeration map for phase $k+2$ is given by

$$\mathrm{Agg}_2^{t_{n+1}} = \left\{ {}^{\mathrm{Tar},t_{n+1}}_{\mathbb{T}_{1,2}} \ : \ \left[{}^{\mathrm{Src},t_{n+1}}_{\mathbb{T}_{2,2}}, \ {}^{\mathrm{Src},t_{n+1}}_{\mathbb{T}_{3,2}} \right] \right\},$$

such that the temporal residual in Equation 10.22 applied to this situation can be decomposed as

$$\begin{aligned}
&\mathcal{J}^{\mathrm{tp,Agg},\Delta t}_{1,-,2,-} \\
&= \int_{\mathbb{T}^{\mathrm{Tar}}_{1,2}(t_{n+1})} \underbrace{\boldsymbol{\phi}^{\mathrm{Agg}}_{1,-}(\mathbf{x})}_{\boldsymbol{\phi}^{\mathrm{Tar}}_{1,-}} \\
&\qquad \cdot \frac{1}{\Delta t} \left[\sum_{i=-N_{\mathrm{step}}+1}^{1} \alpha_i \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}^{\mathrm{Agg},T}_{1,-}(\mathbf{x})}_{\boldsymbol{\phi}^{\mathrm{Tar},T}_{1,-}} \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{1,-,2,-}(t_{n+i}) \right) \right] \mathrm{d}\mathbf{x} \\
&+ \sum_{j=2}^{3} \int_{\mathbb{T}^{\mathrm{Src}}_{j,2}(t_{n+1})} \underbrace{\boldsymbol{\phi}^{\mathrm{Agg}}_{j,-}(\mathbf{x})}_{\underline{\boldsymbol{Q}}^{j\to 1,T}\boldsymbol{\phi}^{\mathrm{Src}}_{j,-}} \\
&\qquad \cdot \frac{1}{\Delta t} \left[\sum_{i=-N_{\mathrm{step}}+1}^{1} \alpha_i \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}^{\mathrm{Agg},T}_{j,-}(\mathbf{x})}_{\boldsymbol{\phi}^{\mathrm{Src},T}_{j,-}\underline{\boldsymbol{Q}}^{j\to 1}} \underline{\boldsymbol{U}}^{\mathrm{Agg}}_{1,-,2,-}(t_{n+i}) \right) \right] \mathrm{d}\mathbf{x},
\end{aligned}$$

where Equation 8.17 was used to express the agglomerated shape functions inside the source elements supports. A crucial point to note here is the fact that the agglomerated DOF are manipulated and not the ones on the non-agglomerated mesh as it is the case for the spatial residual. This is required to avoid any type of inconsistency in the assembly of the temporal residual, as pointed out previously. Since the injection of the agglomerated solution on the non-agglomerated mesh, given by Equation 8.20 and Equation 8.21, takes place before the assembly of the residual, the agglomerated DOF are simply the ones corresponding to the target element, i.e.

$$\underline{\boldsymbol{U}}^{\mathrm{Agg}}_{1,-,2,-}(t_{n+i}) = \underline{\boldsymbol{U}}^{\mathrm{Tar}}_{1,-,2,-}(t_{n+i}) \, .$$

Moving interface For the moving interface, we need to consider the agglomeration maps at each time level (as explained in Exam-

ple 8.1):

$$\begin{aligned}\mathrm{Agg}_2^{t_{n+1}} &= \left\{ \mathrm{T}_{1,2}^{\mathrm{Tar},t_{n+1}} \ : \ \left[\mathrm{T}_{2,2}^{\mathrm{Src},t_{n+1}}, \ \mathrm{T}_{3,2}^{\mathrm{Src},t_{n+1}} \right] \right\} , \\ \mathrm{Agg}_2^{t_n} &= \left\{ \mathrm{T}_{1,2}^{\mathrm{Tar},t_n} \ : \ \left[\mathrm{T}_{2,2}^{\mathrm{Src},t_n} \right] \right\} , \\ \mathrm{Agg}_2^{t_{n-1}} &= \{/\} \ .\end{aligned}$$

Following the same idea as for the splitting approach, we develop Equation 10.23 as

$$\begin{aligned}
&\mathcal{J}_{1,-,2,-}^{\mathrm{tp,Agg},\Delta t} \\
&= \int_{\mathrm{T}_{1,2}^{\mathrm{Tar}}(t_{n+1})} \underbrace{\boldsymbol{\phi}_{1,-}^{\mathrm{Agg}}}_{\boldsymbol{\phi}_{1,-}^{\mathrm{Tar}}}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\alpha_1 \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}_{1,-}^{\mathrm{Agg},T}}_{\boldsymbol{\phi}_{1,-}^{\mathrm{Tar},T}}(\mathbf{x}) \, \underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n+1}) \right) \right] \mathrm{d}\mathbf{x} \\
&+ \sum_{j=2}^{3} \int_{\mathrm{T}_{j,2}^{\mathrm{Src}}(t_{n+1})} \underbrace{\boldsymbol{\phi}_{j,-}^{\mathrm{Agg}}}_{\underline{\boldsymbol{Q}}^{j\to1,T}\boldsymbol{\phi}_{j,-}^{\mathrm{Src}}}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\alpha_1 \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}_{j,-}^{\mathrm{Agg},T}}_{\boldsymbol{\phi}_{j,-}^{\mathrm{Src},T}\underline{\boldsymbol{Q}}^{j\to1}}(\mathbf{x}) \, \underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n+1}) \right) \right] \mathrm{d}\mathbf{x} \\
&+ \int_{\mathrm{T}_{1,2}^{\mathrm{Tar}}(t_n)} \underbrace{\boldsymbol{\phi}_{1,-}^{\mathrm{Agg}}}_{\boldsymbol{\phi}_{1,-}^{\mathrm{Tar}}}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\alpha_0 \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}_{1,-}^{\mathrm{Agg},T}}_{\boldsymbol{\phi}_{1,-}^{\mathrm{Tar},T}}(\mathbf{x}) \, \underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_n) \right) \right] \mathrm{d}\mathbf{x} \\
&+ \int_{\mathrm{T}_{2,2}^{\mathrm{Src}}(t_n)} \underbrace{\boldsymbol{\phi}_{j,-}^{\mathrm{Agg}}}_{\underline{\boldsymbol{Q}}^{2\to1,T}\boldsymbol{\phi}_{2,-}^{\mathrm{Src}}}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\alpha_0 \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}_{2,-}^{\mathrm{Agg},T}}_{\boldsymbol{\phi}_{2,-}^{\mathrm{Src},T}\underline{\boldsymbol{Q}}^{j\to1}}(\mathbf{x}) \, \underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_n) \right) \right] \mathrm{d}\mathbf{x} \\
&+ \int_{\mathrm{T}_{1,2}^{\mathrm{Tar}}(t_{n-1})} \underbrace{\boldsymbol{\phi}_{1,-}^{\mathrm{Agg}}}_{\boldsymbol{\phi}_{1,-}^{\mathrm{Tar}}}(\mathbf{x}) \cdot \frac{1}{\Delta t} \left[\alpha_{-1} \, \mathbf{u}^{\mathrm{cons}} \left(\underbrace{\boldsymbol{\phi}_{1,-}^{\mathrm{Agg},T}}_{\boldsymbol{\phi}_{1,-}^{\mathrm{Tar},T}}(\mathbf{x}) \, \underline{\boldsymbol{U}}_{1,-,2,-}^{\mathrm{Agg}}(t_{n-1}) \right) \right] \mathrm{d}\mathbf{x} \, .
\end{aligned}$$

In this case, the coupling matrices and the agglomerated DOF have to be computed and stored over the different time steps.

10.2.5 General procedure

Algorithm 5 summarizes the main elements covered in Chapter 5, Chapter 8 and Chapter 10 and organizes them into a sequence of steps. Each one of them is commented below.

- (a): the time history of the topologies and the quadrature rules is only needed for moving interface approach (or to restart the simulation). To simplify the notations, we use $\{\bullet\}_i$ to designate a quantity that may need to be collected over the current time level t_{n+1} down to the time $t_{n-N_{\mathrm{step}}+1}$ (with $N_{\mathrm{step}} = 0$ for the splitting approach), and $\{\bullet\}_{i^*}$ for a quantity defined

Algorithm 5 XDG fluid solver for one time step taking as inputs a history of topologies $\{\texttt{Topo}^{t_{n+i}}\}_{i=-N_{\text{step}}+1,\dots,1}$, quadratures $\{\texttt{Quad}^{t_{n+i}}\}_{i=-N_{\text{step}}+1,\dots,1}$ and solutions as well as information about the interface at the newt time step $\Gamma_{\text{info}}^{t_{n+1}}$, and computing the solution at the next time step $\underline{\boldsymbol{U}}^{t_{n+1}}$ based on a spatial XDG discretization combined with cell-agglomeration and single-stage multi-step implicit temporal integration.

procedure XDGOneTimeStep($\{\texttt{Topo}^{t_{n+i}}\}_i$, $\{\texttt{Quad}^{t_{n+i}}\}_i$, $\Gamma_{\text{info}}^{t_{n+1}}$, $\{\underline{\boldsymbol{U}}(t_{n+i})\}_{i^*}$) ▷ (a)

$t_{n+1} \leftarrow t_n + \Delta t$, $\tau_{j=0} = 0$

$\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}) \leftarrow \texttt{ShiftAndResizeData}(\{\texttt{Topo}^{t_{n+i}}\}_i)$ ▷ (b)

$\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}) \leftarrow \underline{\boldsymbol{U}}(t_n)$ ▷ (c)

$\text{Agg} = \{\dots\} \leftarrow \texttt{BuildAgglomMap}(\{\texttt{Topo}^{t_{n+i}}\}_i)$ ▷ (d)

$\{\underline{\boldsymbol{Q}}^{t_{n+i}}\}_i$, $\{\underline{\boldsymbol{M}}^{\text{Agg},t_{n+i}}\}_i \leftarrow \texttt{ComputeAgglomData}(\text{Agg}, \{\texttt{Quad}^{t_{n+i}}\}_i)$ ▷ (e)

$\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1})$, $\{\underline{\boldsymbol{U}}^{\text{Agg}}(t_{n+i})\}_{i^*} \leftarrow \texttt{ProjectSol}(\text{Agg}, \dots$
$\dots \{\underline{\boldsymbol{Q}}^{t_{n+i}}\}_i$, $\{\underline{\boldsymbol{M}}^{\text{Agg},t_{n+i}}\}_i$, $\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})$, $\{\underline{\boldsymbol{U}}(t_{n+i})\}_{i^*})$ ▷ (f)

$\underline{\boldsymbol{U}}^{\text{Tar},\tau_j}(t_{n+1})$, $\underline{\boldsymbol{U}}^{\text{Src},\tau_j}(t_{n+1})$, $\leftarrow \texttt{InjectAgglomSol}(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}) \dots$
$\dots \text{Agg}, \underline{\boldsymbol{Q}}^{t_{n+1}})$ ▷ (g)

while $\left(j \le j_{\max} \;\&\; \left\|\underline{\boldsymbol{M}}_p^{\text{Agg},-1}(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}))\mathcal{N}^{\text{Agg}}(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}))\right\| > \text{tol}\right)$ **do** ▷ (h)

$\mathcal{N}^{\text{tp, Agg}} \leftarrow \texttt{GetAgglomTempRes}(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}) \dots$
$\dots \{\underline{\boldsymbol{U}}^{\text{Agg}}(t_{n+i})\}_{i^*}$, Agg, $\{\underline{\boldsymbol{Q}}^{t_{n+i}}\}_i$, $\{\underline{\boldsymbol{M}}^{\text{Agg},t_{n+i}}\}_i)$ ▷ (i)

$\mathcal{N}^{\text{sp}} \leftarrow \texttt{GetSpatialRes}(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}), \dots$
$\dots \underline{\boldsymbol{U}}^{\text{Tar},\tau_j}(t_{n+1})$, $\underline{\boldsymbol{U}}^{\text{Src},\tau_j}(t_{n+1})$, $\texttt{Quad}^{t_{n+1}})$ ▷ (j)

$\mathcal{N}^{\text{sp, Agg}} \leftarrow \texttt{ProjectAgglomSpatialRes}(\mathcal{N}^{\text{sp}}, \underline{\boldsymbol{Q}}^{t_{n+1}}, \text{Agg})$ ▷ (k)

$\partial\mathcal{N}^{\text{tp, Agg}}/\partial\underline{\boldsymbol{U}}^{\text{Agg}} \leftarrow \texttt{GetAgglomTempJaco}(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}), \dots$
$\dots \text{Agg}, \underline{\boldsymbol{Q}}^{t_{n+1}}, \underline{\boldsymbol{M}}^{\text{Agg},t_{n+1}})$ ▷ (l)

$\partial\mathcal{N}^{\text{sp}}/\partial\underline{\boldsymbol{U}} \leftarrow \texttt{GetSpatialJaco}(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}), \dots$
$\dots \underline{\boldsymbol{U}}^{\text{Tar},\tau_j}(t_{n+1})$, $\underline{\boldsymbol{U}}^{\text{Src},\tau_j}(t_{n+1})$, $\texttt{Quad}^{t_{n+1}})$ ▷ (m)

$\partial\mathcal{N}^{\text{sp, Agg}}/\partial\underline{\boldsymbol{U}}^{\text{Agg}} \leftarrow \texttt{GetAgglomSpatialJaco}(\partial\mathcal{N}^{\text{sp}}/\partial\underline{\boldsymbol{U}}, \dots$
$\dots \underline{\boldsymbol{Q}}^{t_{n+1}}, \text{Agg})$ ▷ (n)

$\Delta\tau_j \leftarrow \texttt{GetPseudoTimeStep}(\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1}), \dots$
$\dots \underline{\boldsymbol{U}}^{\text{Tar},\tau_j}(t_{n+1})$, $\underline{\boldsymbol{U}}^{\text{Src},\tau_j}(t_{n+1}))$ ▷ (o)

$\Delta\underline{\boldsymbol{U}}^{\text{Agg},\tau_{j+1}} \leftarrow \texttt{SolveLinearSystem}(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}), \dots$
$\dots \partial\mathcal{N}^{\text{Agg}}/\partial\underline{\boldsymbol{U}}^{\text{Agg}})$ ▷ (p)

$\underline{\boldsymbol{U}}^{\text{Agg},\tau_{j+1}}(t_{n+1}) = \underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}) + \Delta\underline{\boldsymbol{U}}^{\text{Agg},\tau_{j+1}}$

```
    |
    |   |  U^{Src,τ_{j+1}}(t_{n+1}) ← InjectAgglomSol(U^{Agg,τ_{j+1}}(t_{n+1}), ...
    |   |     ... Agg, Q^{t_{n+1}})                                          ▷ (q)
    |  end while
    |  U(t_{n+1}) ← U^{Agg,τ_{j+1}}(t_{n+1}), U^{Src}(t_{n+1}) ← U^{Src,τ_{j+1}}(t_{n+1})
    |  return U(t_{n+1})
end procedure
```

from the previous time level t_{n+1} down to $t_{n-N_{\text{step}}+1}$. The $\Gamma_{\text{info}}^{t_{n+1}}$ variable represents all the useful quantities that the XDG solver needs from the level-set solver, in particular the normal (and curvature) to the interface. It is not explicitly used in the following steps to simplify the notations but it will be needed for the assembly of the residual on the iso-0.

- (b): shift in time the previous data (such as the cut quadrature rules e.g. so that they don't need to be recomputed over all time levels) and resize the current data structures based on the new topology.
- (c): initialize the solution iterate with the solution from the previous time level. If there are disappearing elements, simply set the solution to zero (in any case those elements will be flagged as source).
- (d): build the agglomeration map by flagging target and source elements based on the topology (Equation 8.15). This is done over several time levels for the moving interface approach and only at the new time level for the splitting one.
- (e): compute, over a single or several time levels, the agglomerated data, i.e. the coupling matrices $\underline{\boldsymbol{Q}}$, Equation 8.16, and the agglomerated mass matrices $\underline{\boldsymbol{M}}^{\text{Agg}}$, Equation 8.18.
- (f): project the initial solution iterate over the agglomerated mesh via Equation 8.19. This might need to be done over previous time steps as well.
- (g): inject the agglomerated solution of the new time step onto the non-agglomerated mesh to obtain the corresponding target and source DOF, using Equation 8.20 and Equation 8.21. Note that in the elements not concerned by agglomeration, the solution remains $\underline{\boldsymbol{U}}^{\tau_j}(t_{n+1})$.
- (h): pseudo-time iteration. The agglomerated time preconditioning mass matrix $\underline{\boldsymbol{M}}_p^{\text{Agg}}$ is either equal to the standard agglomerated mass matrix, $\underline{\boldsymbol{M}}^{\text{Agg}}$, or it is obtained from its definition on non-agglomerated mesh, Equation 10.6, by combining target and source contributions following Equation 8.18. The agglomerated residual is the sum of the temporal and spatial parts, in both phases,

$$\mathcal{N}^{\text{Agg}} = \sum_{k=1}^{2} \left[\mathcal{N}_{-,-,k,-}^{\text{sp, Agg}} + \mathcal{N}_{-,-,k,-}^{\text{tp, Agg}} \right].$$

- (i): assemble the agglomerated temporal residual $\mathcal{N}^{\text{tp, Agg}}$ directly from the agglomerated solution via Equation 10.22 (splitting) or Equation 10.23 (moving interface).
- (j): assemble the non-agglomerated XDG spatial residual $\mathcal{N}^{\text{sp}}$ given by Equation 8.5 in non-cut elements and by Equation 8.6 in cut ones, removing the temporal contribution $\widetilde{\mathcal{J}}$. It depends on the normal XDG solutions and the ones in target and source elements injected on the non-agglomerated mesh.
- (k): assemble the agglomerated spatial residual $\mathcal{N}^{\text{sp, Agg}}$ from the non-agglomerated residual based on Equation 8.25.
- (l): assemble the Jacobian of the agglomerated temporal residual with respect to the agglomerated solution, based on Equation 10.11.
- (m): assemble the non-agglomerated XDG spatial Jacobian $\partial\mathcal{N}^{\text{sp}}/\partial\underline{\boldsymbol{U}}$. It depends on the normal XDG solutions and the ones in target and source elements injected on the non-agglomerated mesh. Some parts of the Jacobian are computed by finite differences, others analytically, as discussed in Section 10.1.2.
- (n): assemble the Jacobian of the agglomerated spatial residual with respect to the agglomerated solution based on the non-agglomerated Jacobians, using Equation 8.26.
- (o): compute the pseudo-time step $\Delta\tau_j$ per element based on Equation 10.10.
- (p): solve Equation 10.8 on the agglomerated mesh to find the increment $\Delta\underline{\boldsymbol{U}}^{\text{Agg},\tau_{j+1}}$:

$$\left[\frac{\underline{\boldsymbol{I}}}{\Delta\tau_j} + \underline{\boldsymbol{M}}_p^{\text{Agg},-1}\left(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1})\right)\frac{\partial\mathcal{N}^{\text{Agg}}}{\partial\underline{\boldsymbol{U}}^{\text{Agg}}}\left(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1})\right)\right]\Delta\underline{\boldsymbol{U}}^{\text{Agg},\tau_{j+1}} = -\underline{\boldsymbol{M}}_p^{\text{Agg},-1}\left(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1})\right)\mathcal{N}^{\text{Agg}}(\underline{\boldsymbol{U}}^{\text{Agg},\tau_j}(t_{n+1}))\,, \tag{10.24}$$

  where we insist on the fact that we need to compute the Jacobian of the agglomerated residual with respect to the agglomerated solution, that is obtained from the Jacobians of the temporal and spatial parts,

$$\frac{\partial\mathcal{N}^{\text{Agg}}}{\partial\underline{\boldsymbol{U}}^{\text{Agg}}} = \frac{\partial\mathcal{N}^{\text{tp, Agg}}}{\partial\underline{\boldsymbol{U}}^{\text{Agg}}} + \frac{\partial\mathcal{N}^{\text{sp, Agg}}}{\partial\underline{\boldsymbol{U}}^{\text{Agg}}}\,.$$

- (q): inject the agglomerated solution on the source elements via Equation 8.21. It is required to do so because the agglomerated spatial residual is assembled based on the non-agglomerated residual, which depends on the source DOF. By construction, the target DOF are equal to the agglomerated DOF (see Equation 8.20), so nothing has to be done.

10.3 Coupling strategy between the fluid and level-set solvers

10.3.1 Overview of possible coupling strategies

For the coupling in space (i.e. between the two phases), we rely on a fully-coupled monolithic approach through the discretization of the problem with the XDG method (see Section 8.2) combined with the enforcement of the jump conditions linking both domains (see Section 9.2). Following the same categorization made before, there are several strategies for the temporal coupling between the fluid and level-set solvers:

- *Operator splitting (segregated) with explicit coupling*: the two main strategies in this category are the Lie and the Strang splittings. With the Lie splitting, the interface position is only updated once from time t to $t+1$ using the fluid velocity at time t before solving the governing equations considering the interface position at time $t+1$. In the case of the Strang splitting, one first performs a half-time step to advect the level-set to time $t+1/2$ with the fluid velocity at time t. One then solves the governing equations with a full time step from t to $t+1$ taking the interface position at $t+1/2$. Finally, one performs a second half-time step on the level-set to obtain the level-set at time $t+1$ from the fluid velocity at time $t+1$ (F. Heimann et al. (2013)). During the resolution of the governing equations, the interface can be assumed totally fixed, in which case the "splitting" approach (see Section 10.2) for temporal integration is selected. The other possibility is to take into account the effect of the interface motion during the time integration of the fluid equations, without explicitly updating the interface position at every iteration of the fluid solver. This is done via the "moving interface approach" described before.
- *Operator splitting (segregated) with implicit coupling*: the second scenario consists in updating the interface position by the new fluid velocity field at time $t+1$. This requires the update of the level-set during the iterations of the fluid solver. In Smuda (2021, Sec. 3.9), the interface update is not performed in every iteration, but only when the fluid residual has converged, before starting a new iteration loop until the tolerances on both the fluid and the level-set residuals decrease below a prescribed threshold. Depending on the time step selected, this approach may produce more accurate results than the explicit coupling, with a higher computational cost.
- *Fully coupled (strong) approach*: this last option avoids any operator splitting by building a residual and a Jacobian matrix incorporating both the fluid and the interface dynamics variables. In this way, the interface dynamics and fluid motion are fully-non-linearly-coupled within a Newton iterator. This approach may be necessary to ensure convergence, maintain a strict

conservation and meet the stringent resolution requirements when dealing with strongly coupled interfacial phenomena having very small time scales, as motivated by Robert Nourgaliev et al. (2008). This approach was also explored in Pochet et al. (2013). Tempting as it is, this coupling strategy seems nonetheless complicated to be included into an existent XDG–level-set framework, especially with the implicit time advancement we are favoring in this work. One of the reasons for this is that the dependence of the fluid variables on the level-set is strongly non-linear, and vice-versa. A particularly critical point would be the linearization of the fluid variables with respect to the level-set which implicitly defines the quadrature rules involved in the fluid residual.

In this work, the simplest approach among the ones discussed above is followed, namely an explicit first-order Lie operator splitting. When this approach is combined with the splitting temporal integration (which is the case for most of the test cases considered in this work), the partitioned advancement of the level-set and the fluid leads to a loss of conservation, as further discussed below. Whether this may lead to the development of some instabilities is still open as no precise investigation has been carried out on this subject in the context of this work.

10.3.2 Algorithm of the implemented segregated approach

The explicit operator splitting between the XDG fluid and level-set solvers implemented in this work is summarized in the following algorithm, exploiting the elements covered previously. A brief comment accompanies each step.

Algorithm 6 Coupling of level-set and XDG fluid solvers over one time step by an explicit first-order Lie operator splitting.

procedure FullSolverOneTimeStep($\{\boldsymbol{\Phi}(t_{n+i})\}_i$, $\{\underline{\boldsymbol{U}}(t_{n+i})\}_i$) ▷ (a)
  $\boldsymbol{\Phi}(t_{n+1}) \leftarrow$ `LSOneTimeStep`($\underline{\boldsymbol{U}}(t_n)$, $\{\boldsymbol{\Phi}(t_{n+i})\}_{i=-N_{\text{step}}+1,\dots,0}$) ▷ (b)
  $\boldsymbol{\Phi}_{C^0}(t_{n+1}) \leftarrow$ `EnforceContinuousLS`($\boldsymbol{\Phi}(t_{n+1})$) ▷ (c)
  $\{\text{Topo}^{t_{n+i}}\}_i \leftarrow$ `UpdateTopology`($\{\boldsymbol{\Phi}_{C^0}(t_{n+i})\}_i$) ▷ (d)
  $\{\text{Quad}^{t_{n+i}}\}_i \leftarrow$ `UpdateQuadrature`($\{\text{Topo}^{t_{n+i}}\}_i$, $\{\boldsymbol{\Phi}_{C^0}(t_{n+i})\}_i$) ▷ (e)
  $\Gamma_{\text{info}}^{t_{n+1}} \leftarrow$ `GetInterfaceInfo`($\boldsymbol{\Phi}_{C^0}(t_{n+1})$) ▷ (f)
  $\underline{\boldsymbol{U}}(t_{n+1}) \leftarrow$ `XDGOneTimeStep`($\{\text{Topo}^{t_{n+i}}\}_i$, $\{\text{Quad}^{t_{n+i}}\}_i$, …
    $\Gamma_{\text{info}}^{t_{n+1}}$, $\{\underline{\boldsymbol{U}}(t_{n+i})\}_{i=-N_{\text{step}}+1,\dots,0}$) ▷ (g)
  return $\boldsymbol{\Phi}(t_{n+1})$, $\underline{\boldsymbol{U}}(t_{n+1})$
end procedure

- (a): depending on the selected time stepping scheme, the level-set and fluid solutions need to be stored over a variable number of time levels.
- (b): advance the level-set field one time step, see Algorithm 4.
- (c): enforce the continuity of the advected level-set field, see Section 6.6.
- (d): update the cut topology information (cut elements, faces and boundaries) based on the continuous level-set field at the new time level.
- (e): compute the new quadrature rules based on the new position of the continuous level-set field.
- (f): collect all the useful quantities that the XDG solver needs from the level-set solver, in particular the normal (and curvature) to the interface.
- (g): advance the Navier-Stokes solutions one time step, see Algorithm 5.

On conservation properties It must be pointed out that the described moving interface solver is not fully conservative. In other words, if the governing physical equations conserve some quantity such as mass, the same quantity in the discrete setting may not be exactly conserved. R. Saye (2017a) identifies two main causes for this. The first one is associated to the fact that neither the advection of the level-set nor the projection of the advected level-set to a continuous field are expected to conserve exactly the volume of each phase. The second cause applies only to the temporal splitting strategy: owing to the extrapolation of the solution from previous time levels to the current one, strict conservation cannot be guaranteed. On the other hand, the moving interface approach, by allowing the preservation of the spatial discretization between the time levels, leads to a conservative scheme in time, as noted by Smuda and Kummer (2022) (but it does not avoid the first cause of conservation loss). R. Saye (2017a) points out, however, that the discrepancies in conservation generally relate to high-order spatial truncation errors and that they are confined to regions of the mesh near the interface, the rest of the domain keeping the usual conservation properties of DGM.

On the use of multi-stage and high-order temporal schemes As mentioned at the beginning of the chapter, the preferred temporal integration scheme in this work is the BDF scheme, in particular BDF2. In addition to its low computational cost, there are also two reasons related to the coupling strategy described above for which this type of scheme is favored. The first one is the first-order accuracy of the Lie coupling which limits anyway the whole process to first accuracy in time. Employing a high-order time-stepping scheme for the fluid equations might therefore not pay off. This same reasoning is followed by Rieckmann et al. (2023). The second reason is that it is unclear if the position of the interface would need to be updated at every substage of multistage schemes to preserve the designed order of accuracy. No investigation has been carried out on this subject, which is why we prefer to stick to the use of single-stage methods.

Contribution 10.1: Temporal discretization for moving interface problems

The main contributions that can be identified from this chapter are:

- The description of the dual time-stepping strategy with the associated change of working variables, the assembly of the temporal residual and the linearization procedure (although those elements were already present in `Argo`).
- A thorough explanation, supported by an illustrative example, of the assembly of the temporal residual combining cell-agglomeration and change of working variables.
- The complete algorithm listing all the steps during one time level of the XDG solver.

From a code development point of view, the elements associated to this chapter that have been implemented in the `Argo` platform can be summarized as:

- The splitting/projection and moving interface/simplified space-time approaches for consistent time integration over moving domains.
- The refactoring and extension of the cell-agglomeration procedure implemented by Pierre Schrooyen and its Cenaero's coworkers such that the data are collected over several time levels.

The developments required for time integration on evolving cut-mesh topology represent the first implementation of the cell-agglomeration strategy to the implicit time integration of the Navier-Stokes system written under a general convective-diffuse fluxes form with the possibility of solving for any set of working variables.

Rest of this page intentionally left blank.

Part IV

Verification and experimental testing

11

Verification based on benchmark test cases

Most of the test cases presented in this chapter have been published or presented by the author during conferences. The generation of the reference solutions in Section 11.3 and Section 11.7.1 was submitted in Henneaux et al. (2023b) and presented in Henneaux et al. (2023e).

11.1 Overview of the test cases 378
11.2 Verification of the general EOS models on single-phase problems 379
11.2.1 One-dimensional Riemann 379
11.2.2 Steady flow around a cylinder 380
11.3 Spatial convergence study of the XDG solver on steady configurations 382
11.3.1 Compressible Couette 386
11.3.2 Manufactured inclined interface 387
11.3.3 Manufactured spinning bubble 393
11.4 Benchmark two-phase test cases in the incompressible regime . 402
11.4.1 Rising bubble 403
11.4.2 Rayleigh-Plesset oscillating bubble 404
11.5 Benchmark two-phase test cases in the compressible regime . . 405
11.5.1 Sod Riemann test case 407
11.5.2 Gas-liquid Riemann test case 408
11.5.3 Shock-bubble interaction 409
11.6 Benchmark two-phase test cases with surface tension 412
11.6.1 Oscillation of a deformed liquid iron droplet 414
11.6.2 Buoyancy-driven rise of a bubble 416
11.6.3 Standing capillary waves 419
11.7 Test cases with phase transformation 422
11.7.1 Mass transfer at the gas-liquid interface 422
11.7.2 Phase transformation at the liquid-solid interface 427

11.1 Overview of the test cases

This chapter starts by Section 11.2 in which the first verification test cases performed in this thesis are presented. Those test cases, which only consider a single phase, are meant to verify the implementation of the new module dealing with general EOS, i.e. the models described in Section 2.4. Section 11.3 then analyzes the spatial convergence behavior of our XDG two-phase flow solver, based on steady complex manufactured solutions satisfying prescribed interfacial jump conditions (see Appendix C). This set of test cases is focused on the verification of the strategy covered in Section 9.2 to enforce the jump conditions. Next, the solver is applied to two unsteady benchmark test cases extracted from the incompressible two-phase flows literature in Section 11.4 to bring into play both the spatial and temporal discretizations in two-phase moving interface situations. The other objective is to assess the capacity of the solver, relying on compressible models, to deal with incompressible regime. Section 11.5 logically follows by addressing reference two-phase compressible test cases, which includes some artificial viscosity to deal with the shocks. In Section 11.6, the application of the curvature smoothing techniques developed in Section 7.2 is evaluated based on standard incompressible test cases. Finally, mass transfer effects are considered in the first part of Section 11.7 while in the second part, melting phase transformation is included in the simulated problems.

All the meshes used in the following test cases have been generated by the open source mesh and visualization tool `Gmsh` presented in Geuzaine and Remacle (2009). To visualize the solution and extract useful quantities, the open source `Paraview` software has been used (Ahrens et al. (2005)). In particular, the Gmsh Reader plugin of Paraview developed by Rasquin et al. (2019) has been used to visualize high-order solutions. It is based on the internal method of `Gmsh` relying on recursive h-refinements of the initial mesh and proposed in Remacle et al. (2007).

Concerning the general presentation of the following test cases, we can remark that for each batch of test cases, a table listing the most important physical and numerical parameters is first presented. Then, for each test case separately, the reference solution with respect to which we evaluate the quality of our numerical solution is described before showing the results and commenting them.

11.2 Verification of the general EOS models on single-phase problems

Two test cases are considered to verify the implementation of the general EOS models into the code. The parameters for each of them can be found in Table 11.1.

Table 11.1 / Numerical and physical parameters for the general EOS single-phase test cases.

	Parameters	VdW Riemann	VdW steady cylinder flow
Spatial discretization	Cartesian grid	1000×1	unstruct.
	Cell size Δx (m)	4e-4	unstruct.
	Interp. order	$p = 0$	$p = 4$
	Riemann solver	HLLC & SLAU	SLAU
Temporal discretization	Time step Δt (s)	1.0e-7	steady
	Scheme	BDF2	steady
Physical parameters	EOS	van der Waals	van der Waals
	R (J/(kg K))	231.11	288.20
	C_v (J/(kg K))	577.80	746.32
	A(SI)	0.5	5.0
	B(SI)	0.005	0.5
	Nondim. numbers	/	$\mathrm{Re} = 20$, $\mathrm{Pr} = 1$

11.2.1 One-dimensional Riemann

Setup and reference solution

The one-dimensional Riemann problem gives an exact solution to the complex non-linear Euler equations and it is therefore very common to consider it in order to test the accuracy of computational fluid codes. Initially, two fluids at rest having different thermodynamic states are separated by a diaphragm. At time $t = 0$, the sudden breakdown of the diaphragm generates a high-speed flow, which propagates in the tube along with some characteristic waves that can be of different nature. Here, the well-known shock-tube problem is treated which consists initially in a high-pressure fluid in the left part of the tube

and a low-pressure one in the right part. In this configuration, the fluid at high pressure expands through an expansion wave and pushes the fluid of the right part. The compression of the low-pressure fluid generates a shock wave propagating to the right. Finally, the expanded fluid is separated from the compressed fluid by a contact discontinuity. This situation is solved for the van der Waals EOS which write

$$p = \frac{RT}{1/\rho - B} - A\rho^2 \,,$$
$$e = e_T = C_v T - A\rho \,,$$

where $R = (\gamma - 1)C_v$. The numerical solution is compared to the exact semi-analytical solution that can be computed as detailed in Appendix D. The initial conditions,

$$\text{Left state:} \begin{cases} \rho = 100 \,, \\ v_x = 0 \,, \\ T = 5000 \,, \end{cases} \qquad \text{Right state:} \begin{cases} \rho = 10 \,, \\ v_x = 0 \,, \\ T = 3000 \,, \end{cases}$$

are taken from Saurel et al. (1994). The SLAU and HLLC approximate Riemann solvers are selected here. The order of the polynomial interpolation of the DGM solver inside each element is kept equal to 0, which corresponds to a classical first-order FVM scheme. In the other Riemann problems presented after, this order is increased and artificial viscosity is added to avoid spurious oscillations in the vicinity of discontinuities.

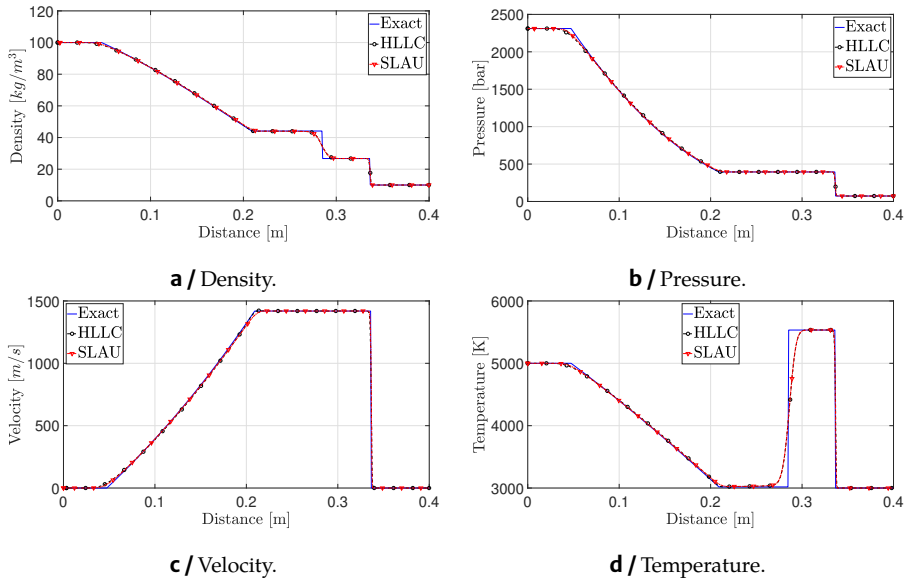
Results

The excellent agreement between the numerical profiles superposed to the exact solutions in Figure 11.1 show that the use of the van der Waals EOS in the new module is working as expected.

11.2.2 Steady flow around a cylinder

Setup and reference solution

In order to include viscous effects, the classical situation of a flow around a fixed cylinder in a channel is studied. The Reynolds number is chosen such that the flow is steady. The SU2 open-source CFD software, which was extended by Rinaldi et al. (2014) to work with non-ideal EOS such as the van der Waals EOS, is used to compare the results obtained with Argo, with the van der Waals EOS. SU2 relies on a finite volume discretization and we use MUSCL reconstruction to reach a second order of convergence. The Riemann solver selected is Roe. At the inlet of the domain, a velocity of $v_x = 30\text{m/s}$ and a temperature of $T = 293.15\text{K}$ are imposed. At the outlet, the pressure is set to $p = 101320\text{Pa}$. A



a / Density. **b /** Pressure.

c / Velocity. **d /** Temperature.

Figure 11.1 / Comparison between the numerical *van der Waals Riemann* test case solutions obtained with the HLLC and SLAU Riemann solvers and the analytical solution in plain blue line at time $t = 6 \cdot 10^{-5}$s.

no-slip isothermal boundary condition, with $T = 293.15$K, is imposed on the cylinder surface.

Results

The comparison of the meshes used with the two codes in Figure 11.2 highlights the important difference in terms of refinement level. This simple test case already brings out the benefit of using high-order methods such as the DGM: for a number of degrees of freedom which is lower, `Argo` is able to produce results with the same level of accuracy than the one of `SU2`, with a shorter CPU time on this configuration. In Figure 11.3, a qualitative comparison between the two codes based on the iso-contours of the Mach number is shown. A good matching can be observed. A quantitative comparison on the values of the drag coefficients has also been considered with the following results:

$$C_D = \begin{cases} 2.44363 & \text{for } \texttt{Argo}\,, \\ 2.46465 & \text{for } \texttt{SU2}\,. \end{cases}$$

With a relative error of 0.85%, we can affirm that both codes are in agreement. Note that we use curved elements to represent more accurately the cylinder surface with `Argo` but could not do the same with `SU2`.
We took advantage of the steady cylinder configuration to also perform a preliminary verification on the coupling between the module providing general

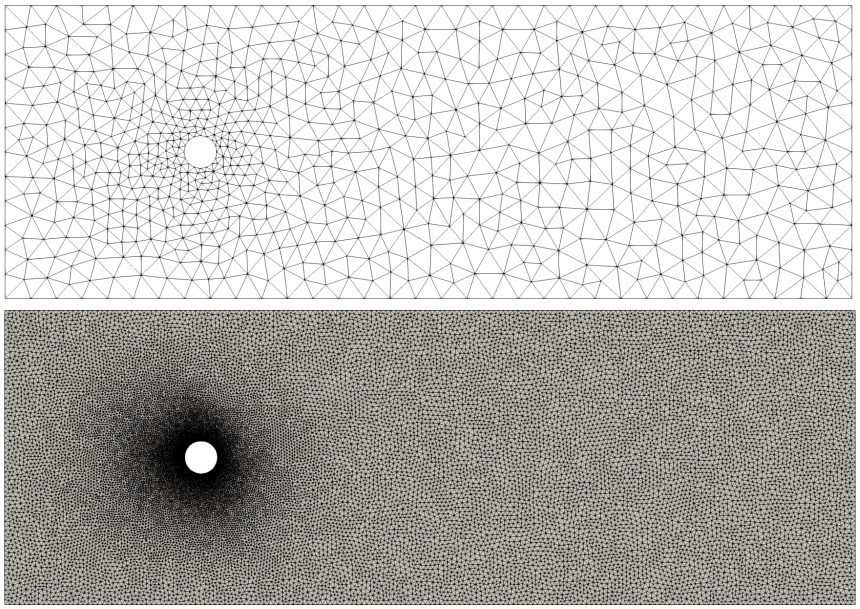


Figure 11.2 / Meshes for the *van der Waals steady cylinder flow*. The top mesh is used by `Argo` with 21570 DOF (fifth order of convergence), the bottom mesh by `SU2` with 46034 DOF (second order of convergence) .

EOS and the XDG discretization routines. We solved the same problem with the surface of the cylinder cutting across the underlying mesh and treated in an immersed way by the XDG method. As Figure 11.4 shows, the velocity fields obtained by the standard DG conforming discretization and the XDG one look the same. This is confirmed by the values of the drag coefficient computed in both cases:

$$C_D = \begin{cases} 2.44363 & \text{for } \texttt{Argo} \text{ DG conforming}, \\ 2.44453 & \text{for } \texttt{Argo} \text{ XDG immersed}. \end{cases}$$

11.3 Spatial convergence study of the XDG solver on steady configurations

Three test cases with a stationary interface are considered to verify the implemented solver: a compressible Couette test case (Section 11.3.1), a manufactured solution inclined interface test case (Section 11.3.2) and a manufactured solution spinning bubble test case (Section 11.3.3). An overview of the various parameters selected for each of them is provided in Table 11.2. It should be pointed out at this stage that the second test case cannot be considered as a high contrast interface problem, unlike the other two. This is done on purpose so that

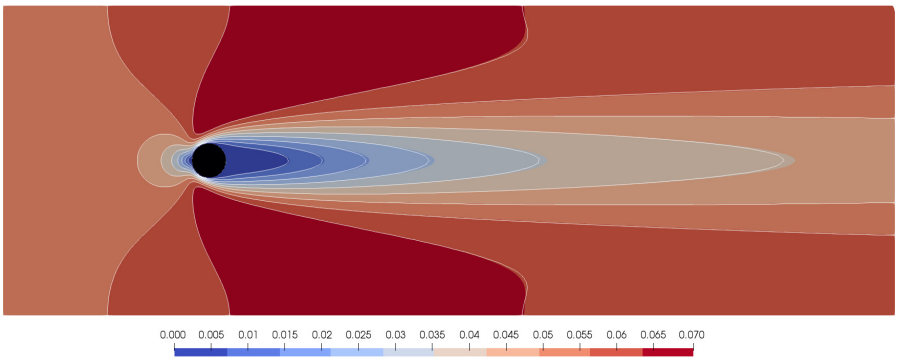


Figure 11.3 / Mach number distribution for the *van der Waals steady cylinder flow*. White iso-contours are the SU2 solution, filled contours the Argo solution, with the same scale.

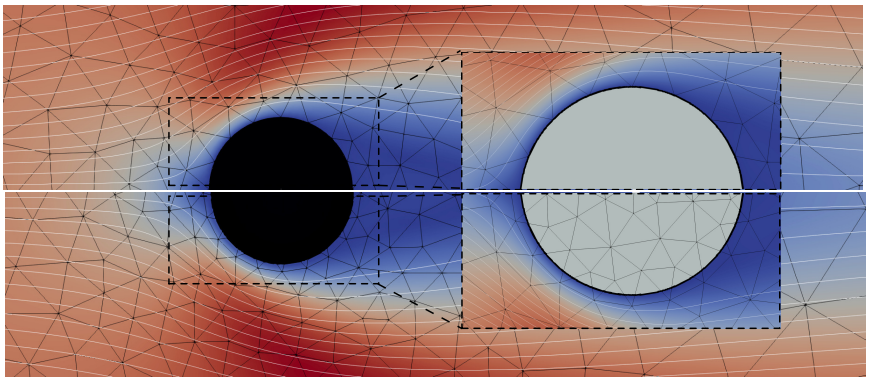


Figure 11.4 / Velocity distribution and streamlines for the *van der Waals steady cylinder flow*. Comparison between the solution obtained by Argo with a boundary conforming mesh (top) and an immersed boundary treated by the XDGM (bottom).

a sensibility analysis of various numerical parameters can be conducted without being influenced by potential effects associated to high properties jumps. The two other test cases are there to complement the analysis. The symbolic `Python` software that has been developed for generating complex manufactured solutions satisfying prescribed conditions is described in Appendix C. It must be noted that for each test case, the initial solutions given to the solver are slightly perturbed compared to the exact ones so that the initial value of the residual is not small and the capacity of the solver to bring it to a prescribed tolerance level can be evaluated. For each of the test case, an order-of-accuracy study is conducted by evaluating the experimental order of convergence (EOC), which depends both on the grid level of refinement h and the polynomial order p, as

$$\mathrm{EOC}(h,p) = \frac{\log\left(E^p_{h_{i-1}}/E^p_{h_i}\right)}{\log\left(h_{i-1}/h_i\right)}, \qquad [11.1]$$

where h_{i-1}/h_i is the ratio of grid sizes between two consecutive grids and $E^p_{h_i} = \left\|\mathbf{u}^p_{h_i} - \tilde{\mathbf{u}}\right\|$ denotes the error of the numerical approximation of degree p on the grid with spacing h_i, noted $\mathbf{u}^p_{h_i}$, with respect to the reference analytical solution, $\tilde{\mathbf{u}}$. The errors used in the following are measured in terms of two different norms:

$$\|\mathbf{u} - \tilde{\mathbf{u}}\|^2_{L_2(\Omega)} = \sum_{k\in\{1,2\}} \sum_{m\in\mathbb{T}^{k,*}} \int_{\mathbb{T}_m\cap\Omega_k} (\mathbf{u}_k - \tilde{\mathbf{u}}_k)^2 \,\mathrm{d}\mathbf{x}, \qquad [11.2]$$

$$\|\mathbf{u} - \tilde{\mathbf{u}}\|_{L_\infty(\Omega)} = \max_{\forall k\in\{1,2\}} \left(\max_{\forall q\in Q_k} |\mathbf{u}_k(\mathbf{x}_q) - \tilde{\mathbf{u}}_k(\mathbf{x}_q)| \right), \qquad [11.3]$$

where the integrals in Equation 11.2 use the special volume quadrature rules in the cut elements to integrate on both sides of the interface and Q_k in Equation 11.3 labels the set of volume quadrature points belonging to phase k with their corresponding coordinates $\mathbf{x}_q$.

Table 11.2 / Parameters of the test cases used for the grid convergence study. Some parameters of the *Inclined interface* and *Spinning bubble* test cases are modified for some sensitivity analysis study afterward, hence the "V1" denomination here.

Parameters		Compressible Couette	Inclined interface (V1)	Spinning bubble (V1)
Bulk properties (Section 2.4)	EOS	Stiffened	PG	Stiffened
	γ_1 , γ_2	3.0 , 1.4	1.4 , 1.6	1.911 , 1.290
	$C_{v,1}$, $C_{v,2}$	3.229 , 0.708	2.5 , 2.0	977.704, 991.20
	$p_{\infty,1}$, $p_{\infty,2}$	1.0×10^5 , 0.0	0.0 , 0.0	7.16×10^9 , 0.0
	μ_1 , μ_2	10.0 , 1.0	1.0 , 0.25	4.45e-3 , 6.98e-5

	λ_1 , λ_2	10.0 , 1.376	1.0 , 0.25	35.24 , 0.127
Jumps values (Table 3.2)	$\dot{m}_\Gamma$	0.0	0.0	-0.001
	$\Delta_\Gamma p$	0.0	-0.2	200.0
	$\Delta_\Gamma \tau_{nt}$	0.0	0.2	0.2
	$\Delta_\Gamma q$	1000.0	1.0	1000.0
	$\Delta_\Gamma T$	0.0	0.5	100.0
	$\Delta_\Gamma v_t$	0.0	-0.5	-50.0
Numerics - General	Variables sets	$\rho - \rho\mathbf{v} - \rho E$	$p - \mathbf{v} - T$	$p - \mathbf{v} - T$
	Iterative solver	Newton +Gauss	Newton +Gauss	Newton +GMRES (ILU)
Numerics - Convective	Riemann bulk	SLAU2	HLLC	SLAU2
	Riemann iso-0 (Section 9.2.2)	HLLC	HLLCP	HLLC
Numerics - Diffusive	IP	SIPG	SIPG	SIPG
	Penalty (Equation 9.25)	Tensorial	Tensorial	Tensorial
Numerics - XDG	Splitting (Table 9.1)	# 1 (jump on heat flux!)	# 2	# 1
	δ_{agg} (Section 8.2.5)	0.2	0.3	0.3
	Quadrature	*Algoim*	*PHG*	*PHG*
	Exponent a (Equation 9.16)	0.0	1.0	1.0

On the suitability of the manufactured solutions An appropriate manufactured solution should exercise as many of the terms in the discretized form as possible. Given the complexity of the equations considered here, the expressions of the forcing source terms that are added are also very complex and

lengthy (up to several thousands terms). The question then arises whether the source term, having a much more complex expression than the other terms, will dominate the overall verification process. We do not investigate and quantify this possibility. Nonetheless, we can point out that the manufactured solutions involve trigonometric (for the inclined interface) or rational functions (for the spinning bubble). Hence, the polynomial interpolation accuracy of our DG method is inevitably put to the test, whether for the source or the other terms in the discretization.

11.3.1 Compressible Couette

This test case is taken from S. Peluchon et al. (2022) and is slightly modified. A horizontal interface separates a stiff fluid on the top and a gas-like fluid on the bottom inside an infinite two-dimensional channel, as represented in Figure 11.5. At the lower channel boundary, a no-slip condition and a temperature of 310 K are imposed while the upper boundary moves at a speed of 10 m/s and has a temperature of 330 K. All the interface jumps are set to zero except a heat flux source equal to $\Delta_\Gamma q = 1000.0\,W/m^2$. A jump on the heat flux (and not on the difference between the heat flux and the viscous dissipation) is imposed here, i.e. Equation 9.19 is considered for the energy jump condition. Since the solution is assumed to be independent of the horizontal coordinate, it is possible to compute an analytical solution. Because the analytical solution is composed of second-order polynomials for the temperature and velocity fields, this test case is solved in terms of the conservative variables such that higher order terms are involved and a convergence analysis for $p > 1$ is possible. Two of the structured quadrilateral grids used for the convergence study are shown in Figure 11.6. In Table 11.3, the errors on the pressure and horizontal velocity fields are listed for several interpolation orders and grid sizes. For $p = 4$, the refinement stops earlier than for the lower polynomial orders because the absolute values of the errors quickly become negligible. Overall, the data indicate that the solver produces optimally convergent results in the sense that the orders of convergence measured in L_2 and L_∞ error norms behave roughly as $p + 1$. However, some deviations from this theoretical rate can be observed which might be explained by the use of the conservative variables set rather than the p–$\mathbf{v}$–T set. This is further illustrated in Section 11.3.3.

Table 11.3 / *Compressible Couette* test case (Table 11.2) – Discretization errors and experimental orders of convergence (EOC) in L_2 and L_∞ error norms for polynomial degrees p on *structured quadrilateral* grids of $3 \times n$ elements.

p	n	P				v_x			
		L_2	EOC	L_∞	EOC	L_2	EOC	L_∞	EOC
1	20	2.41e-03	**1.6**	1.10e-01	**1.6**	4.18e-05	**2.0**	1.27e-03	**2.6**

	40	6.97e-04	**1.8**	3.34e-02	**1.7**	1.07e-05	**2.0**	3.67e-04	**1.8**
	80	2.06e-04	**1.8**	9.02e-03	**1.9**	2.70e-06	**2.0**	1.12e-04	**1.7**
	160	5.95e-05	**1.8**	2.48e-03	**1.9**	6.82e-07	**2.0**	3.74e-05	**1.6**
2	20	8.87e-06	**2.5**	5.82e-04	**2.1**	3.40e-07	**3.6**	2.13e-05	**2.9**
	40	1.03e-06	**3.1**	6.74e-05	**3.1**	4.88e-08	**2.8**	2.92e-06	**2.9**
	80	1.55e-07	**2.7**	1.94e-05	**1.8**	7.27e-09	**2.7**	4.32e-07	**2.8**
	160	2.80e-08	**2.5**	5.68e-06	**1.8**	1.03e-09	**2.8**	6.87e-08	**2.7**
3	20	6.37e-08	**3.4**	4.67e-06	**3.1**	3.28e-09	**4.0**	1.35e-07	**3.7**
	40	3.18e-09	**4.3**	2.56e-07	**4.2**	2.10e-10	**4.0**	1.02e-08	**3.7**
	80	1.60e-10	**4.3**	1.84e-08	**3.8**	1.33e-11	**4.0**	8.04e-10	**3.7**
	160	1.09e-11	**3.9**	1.52e-09	**3.6**	8.49e-13	**4.0**	5.99e-11	**3.7**
4	10	9.60e-08	**-**	9.90e-06	**-**	5.36e-09	**-**	8.18e-07	**-**
	20	6.59e-10	**7.2**	9.56e-08	**6.7**	1.36e-11	**8.6**	1.33e-09	**9.3**
	40	2.50e-11	**4.7**	4.02e-09	**4.6**	3.94e-13	**5.1**	4.91e-11	**4.8**

11.3.2 Manufactured inclined interface

This second test case is built from the method of manufactured solutions. The manufactured solution is first found for a horizontal interface and is then rotated by 15° to obtain the inclined interface configuration in Figure 11.7. Two fluids governed by the perfect gas EOS with different heat capacity ratios and heat capacities at constant volume and ratios of dynamic viscosities and thermal conductivities equal to four are considered. To test the ability of the solver to impose all the different jumps treated in this work, those are all given a value different from zero, except the interface mass flux which is imposed to zero. The HLLCP multiphase Riemann solver with splitting strategy 2 is used.

Grid convergence study

Two of the Cartesian quadrilateral grids used for the convergence study are shown in Figure 11.8. Table 11.4 presents the discretization orders of accuracy on primitive variables. The theoretical $p + 1$ convergence rates are here also reached. Note that for $p = 4$, the refinement stops earlier than for the lower polynomial orders because the absolute values of the errors become negligible more quickly. Because the position of the inclined interface is kept unchanged on the various grids considered, it can be concluded that the discrete errors are insensitive to the way the interface intersects the elements on this particular

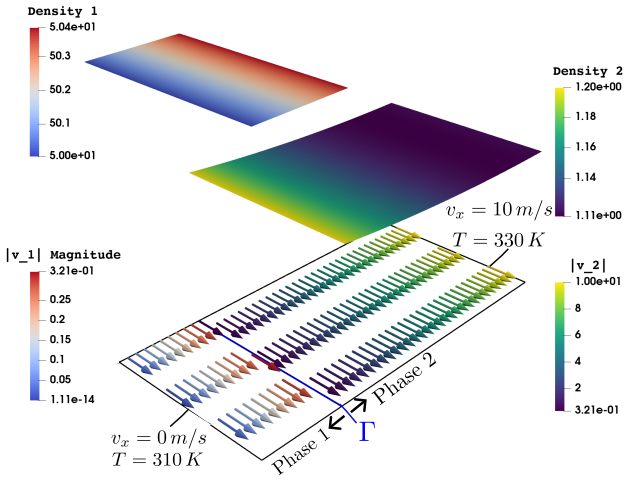


Figure 11.5 / Domain of the *compressible Couette* test case with the position of the phase interface Γ. Arrows represent direction of the velocity field for both phases. The elevated density field (not to scale) is also drawn.

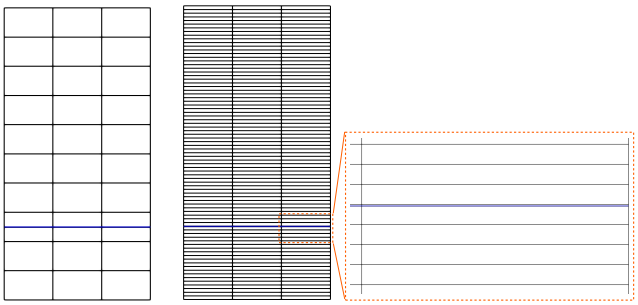


Figure 11.6 / Grids 3×10 (left) and 3×80 (middle) used for the *compressible Couette* test case. The zoom on the interface region (right) shows the proximity of the interface with the element edge.

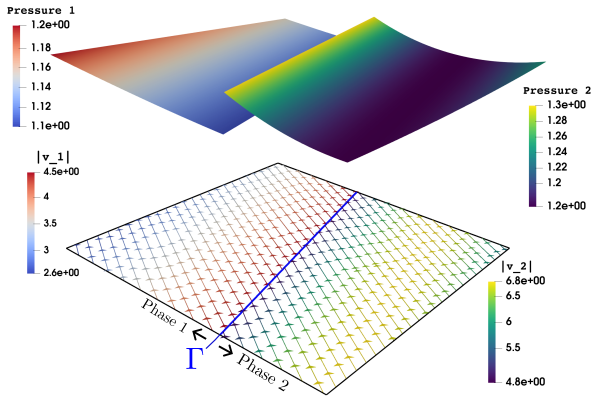


Figure 11.7 / Domain of the *inclined interface* test case with the position of the phase interface Γ. Arrows represent direction of the velocity field for both phases. The elevated pressure field (not to scale) is also drawn.

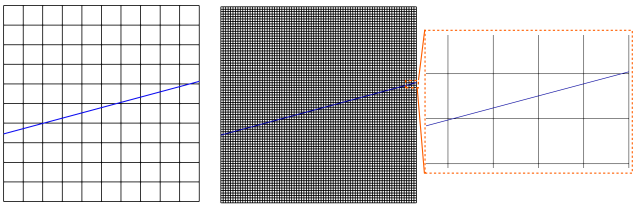


Figure 11.8 / Grids 10 × 10 (left) and 80 × 80 (middle) used for the *inclined interface* test case. The zoom on the interface region (right) shows the variety of intersections between the interface and the grid: from normal cuts to (very) small ones.

configuration. It shall be demonstrated with the remaining test cases that this observation holds also true for more complex interface geometries.

Table 11.4 / *Inclined interface* test case, *V1* (Table 11.2). Discretization errors and experimental orders of convergence (EOC) in L_2 and L_∞ error norms for polynomial degrees p on *Cartesian quadrilateral* grids of $n \times n$ elements.

p	n	$\boxed{P}$ L_2	EOC	L_∞	EOC	$\boxed{T}$ L_2	EOC	L_∞	EOC
2	10	1.12e-04	-	1.08e-03	-	1.79e-04	-	1.08e-03	-
	20	1.41e-05	**3.0**	1.48e-04	**2.9**	2.04e-05	**3.1**	9.53e-05	**3.5**
	40	1.77e-06	**3.0**	1.91e-05	**3.0**	2.49e-06	**3.0**	1.02e-05	**3.2**
	80	2.18e-07	**3.0**	2.26e-06	**3.1**	3.09e-07	**3.0**	1.24e-06	**3.0**
3	10	4.97e-06	-	4.21e-05	-	1.03e-05	-	7.01e-05	-
	20	3.57e-07	**3.8**	5.77e-06	**2.9**	7.22e-07	**3.8**	7.55e-06	**3.2**
	40	2.43e-08	**3.9**	5.31e-07	**3.4**	4.74e-08	**3.9**	6.44e-07	**3.5**
	80	1.47e-09	**4.0**	3.75e-08	**3.8**	2.86e-09	**4.1**	4.39e-08	**3.9**
4	10	3.00e-07	-	7.04e-06	-	6.17e-07	-	6.90e-06	-
	20	9.52e-09	**5.0**	3.15e-07	**4.5**	1.81e-08	**5.1**	2.89e-07	**4.6**
	40	3.06e-10	**5.0**	1.23e-08	**4.7**	5.47e-10	**5.0**	1.08e-08	**4.7**

p	n	$\boxed{v_x}$ L_2	EOC	L_∞	EOC	$\boxed{v_y}$ L_2	EOC	L_∞	EOC
2	10	1.40e-04	-	9.48e-04	-	2.16e-04	-	1.56e-03	-
	20	1.69e-05	**3.1**	1.57e-04	**2.6**	2.60e-05	**3.1**	2.30e-04	**2.8**
	40	2.07e-06	**3.0**	2.03e-05	**2.9**	3.18e-06	**3.0**	3.26e-05	**2.8**
	80	2.48e-07	**3.1**	2.60e-06	**3.0**	3.81e-07	**3.1**	4.17e-06	**3.0**
3	10	5.80e-06	-	1.89e-05	-	8.81e-06	-	2.92e-05	-
	20	3.65e-07	**4.0**	1.13e-06	**4.1**	5.54e-07	**4.0**	1.74e-06	**4.1**
	40	2.29e-08	**4.0**	7.01e-08	**4.0**	3.48e-08	**4.0**	1.06e-07	**4.0**
	80	1.43e-09	**4.0**	4.38e-09	**4.0**	2.17e-09	**4.0**	6.64e-09	**4.0**
4	10	7.36e-08	-	7.60e-07	-	1.12e-07	-	1.21e-06	-
	20	2.18e-09	**5.1**	3.14e-08	**4.6**	3.28e-09	**5.1**	4.97e-08	**4.6**
	40	6.66e-11	**5.0**	1.19e-09	**4.7**	1.00e-10	**5.0**	1.86e-09	**4.7**

Sensitivity analysis

In this second part, some numerical parameters listed in Table 11.2 are modified to conduct an analysis of their influence on the convergence behavior of the solver.

Influence of the interior penalty method choice Table 11.5 compares the results obtained with the three different interior penalty methods available. As expected, the *NIPG* and *IIPG* methods exhibits a reduced p convergence rates with respect to the L2-norm for even polynomial orders Rivière (2008, Sec. 1.5). The *SIPG* also allows reaching slightly lower errors on the finest grids.

Table 11.5 / *Inclined interface* test case with same parameters as *V1* (Table 11.2) except the choice of Interior Penalty (IP) method. Discretization errors and experimental orders of convergence (EOC) in L_2 error norms for polynomial degrees p on *Cartesian quadrilateral* grids of $n \times n$ elements for the *pressure* field. Comparison between the three different interior penalty methods considered: *SIPG*, *NIPG*, *IIPG*.

p	n	*SIPG*		*NIPG*		*IIPG*	
		L_2	EOC	L_2	EOC	L_2	EOC
2	10	1.12e-04	-	4.13e-04	-	2.70e-04	-
	20	1.41e-05	**3.0**	1.13e-04	**1.9**	6.89e-05	**2.0**
	40	1.77e-06	**3.0**	3.00e-05	**1.9**	1.78e-05	**2.0**
	80	2.18e-07	**3.0**	7.74e-06	**2.0**	4.54e-06	**2.0**
3	10	4.97e-06	-	7.18e-06	-	6.20e-06	-
	20	3.57e-07	**3.8**	4.91e-07	**3.9**	4.32e-07	**3.8**
	40	2.43e-08	**3.9**	3.26e-08	**3.9**	2.89e-08	**3.9**
	80	1.47e-09	**4.0**	1.98e-09	**4.0**	1.76e-09	**4.0**
4	10	3.00e-07	-	4.70e-07	-	3.90e-07	-
	20	9.52e-09	**5.0**	2.46e-08	**4.3**	1.76e-08	**4.5**
	40	3.06e-10	**5.0**	1.49e-09	**4.0**	9.76e-10	**4.2**

Comparison between scalar and tensorial penalty The second comparison is about the use of scalar and tensorial penalty forms for the discretization of the diffusive fluxes, both in the bulk and at the phase interface. From Table 11.6, it can be noticed that a sub-optimal p convergence rate is obtained for even polynomial orders for the scalar penalty form. This observation was also made

in Ralf Hartmann and Houston (2008, Sec. 5.1). Moreover, the errors values are higher with the scalar penalty. Finally, it is important to remark that the safety factor s of the penalty term, Equation 9.26, had to be increased by a factor of five to ten compared to the tensorial penalty to obtain convergent results.

Table 11.6 / *Inclined interface* test case with same parameters as *V1* (Table 11.2) except the choice of the penalty form. Discretization errors and experimental orders of convergence (EOC) in L_2 error norms for polynomial degrees p on *Cartesian quadrilateral* grids of $n \times n$ elements for the *temperature* field. Comparison between the two different forms of penalty terms considered: scalar and tensorial penalty (Equation 9.25).

p	n	*Tensorial*		*Scalar*	
		L_2	EOC	L_2	EOC
2	10	1.79e-04	-	1.42e-03	-
	20	2.04e-05	**3.1**	7.45e-04	**0.9**
	40	2.49e-06	**3.0**	2.15e-04	**1.8**
	80	3.09e-07	**3.0**	4.48e-05	**2.3**
3	10	1.03e-05	-	2.08e-05	-
	20	7.22e-07	**3.8**	8.35e-07	**4.6**
	40	4.74e-08	**3.9**	5.43e-08	**3.9**
	80	2.86e-09	**4.1**	5.61e-09	**3.3**
4	10	6.17e-07	-	2.83e-06	-
	20	1.81e-08	**5.1**	2.32e-07	**3.6**
	40	5.47e-10	**5.0**	1.51e-08	**3.9**

Influence of the variables set choice The comparison between the use of primitive and conservative variables is presented in Table 11.7. For this mild configuration, both choices lead to comparable results, with the primitive variables producing slightly lower errors.

Table 11.7 / *Inclined interface* test case with same parameters as *V1* (Table 11.2) except the choice of the solved-for variables sets. Discretization errors and experimental orders of convergence (EOC) in L_2 error norms for polynomial degrees p on *Cartesian quadrilateral* grids of $n \times n$ elements for the *pressure* field. Comparison between the use of primitive and conservative variables sets.

p	n	p-$\mathbf{v}$-T				ρ-$\rho\mathbf{v}$-ρE			
		L_2	EOC	L_∞	EOC	L_2	EOC	L_∞	EOC
2	10	1.12e-04	-	1.08e-03	-	3.83e-04	-	3.58e-03	-
	20	1.41e-05	**3.0**	1.48e-04	**2.9**	4.70e-05	**3.0**	7.01e-04	**2.4**
	40	1.77e-06	**3.0**	1.91e-05	**3.0**	5.77e-06	**3.0**	8.07e-05	**3.1**
	80	2.18e-07	**3.0**	2.26e-06	**3.1**	6.55e-07	**3.1**	1.12e-05	**2.8**
3	10	4.97e-06	-	4.21e-05	-	2.54e-05	-	2.81e-04	-
	20	3.57e-07	**3.8**	5.77e-06	**2.9**	1.66e-06	**3.9**	3.25e-05	**3.1**
	40	2.43e-08	**3.9**	5.31e-07	**3.4**	1.06e-07	**4.0**	2.77e-06	**3.6**
	80	1.47e-09	**4.0**	3.75e-08	**3.8**	5.70e-09	**4.2**	2.02e-07	**3.8**
4	10	3.00e-07	-	7.04e-06	-	2.18e-06	-	2.82e-05	-
	20	9.52e-09	**5.0**	3.15e-07	**4.5**	7.98e-08	**4.8**	2.19e-06	**3.7**
	40	3.06e-10	**5.0**	1.23e-08	**4.7**	2.64e-09	**4.9**	1.01e-07	**4.4**

11.3.3 Manufactured spinning bubble

This last test case is inspired from Robert Nourgaliev et al. (2017). A static fluid bubble is placed at the middle of a squared domain. A manufactured solution is assembled in cylindrical coordinates and then converted to Cartesian coordinates to be passed to the solver. Here real physical conditions are selected that could correspond to a bubble of molten liquid titanium interacting with surrounding air heated around 2000 K (high-temperature effects are however not taken into account). The liquid titanium is almost incompressible with stiffened gas EOS parameters chosen such that the Mach number inside the bubble is of the order of 5×10^{-4}. There are high jumps in properties between the two phases with ratios of densities $\rho_1/\rho_2 \sim 10^5$, of dynamic viscosities $\mu_1/\mu_2 \sim 10^2$ and of thermal conductivities $\lambda_1/\lambda_2 \sim 10^2$. All the jumps are considered different from zero, including the interface mass flux. It can be noted that at the center of the domain, a small empty squared region is added to avoid singularities of the manufactured source terms which is converted from cylindrical to Cartesian coordinates before being fed to the solver. This is shown in Figure 11.9.

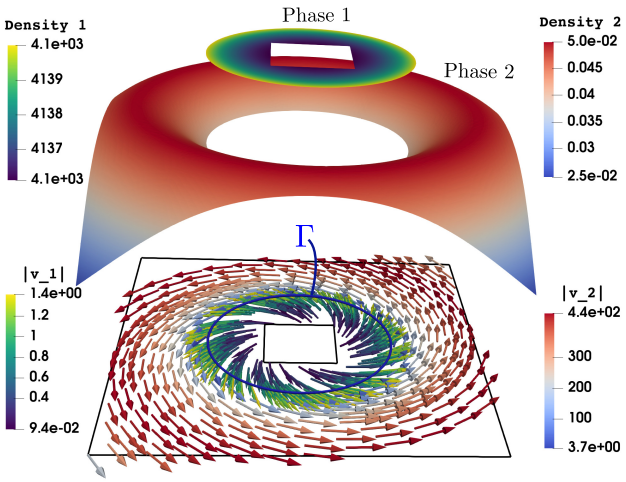


Figure 11.9 / Domain of the *spinning bubble* test case with the position of the phase interface Γ. Arrows represent direction of the velocity field for both phases. The elevated density field (not to scale) is also drawn.

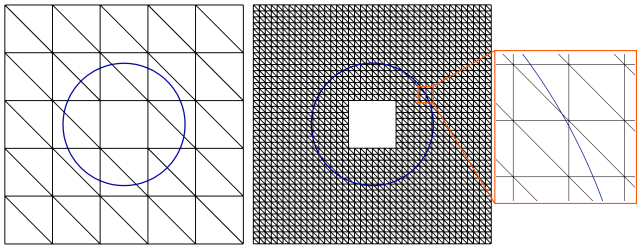


Figure 11.10 / Grids 10 × 10 (left) and 80 × 80 (middle) used for the *spinning bubble* test case. The zoom on the interface region (right) shows the variety of intersections between the interface and the grid: from normal cuts to (very) small ones.

Grid convergence study

Two of the Cartesian triangular grids used for the convergence study are shown in Figure 11.10. Table 11.8 provides the results obtained on Cartesian triangular grids. Due to the geometry considered, the real physical parameters and the stiffness of the models involved, the values of the errors are higher than for the two first test cases. Overall, in the L_2 norm, the targeted $p+1$ convergence rates for the density and vertical velocity and p rates for the heat flux and vorticity are achieved. In the L_∞ norm, this trend is a bit less marked with some rates below or above the expected ones. This is probably due to the stiffness of the models which impacts the accuracy with which the solver can discretize some quantities in different ways. In particular, the treatment of the convective fluxes is very sensitive and critical, as discussed in the first paragraph of Section 11.3.3.

Table 11.8 / *Spinning bubble* test case, *V1* (Table 11.2). Discretization errors and experimental orders of convergence (EOC) in L_2 and L_∞ error norms for polynomial degrees p on *Cartesian triangular* grids of $n \times n$ elements. Results for density, y-component of velocity, x-component of heat flux and vorticity ω fields.

p	n	$\boxed{\rho}$ L_2	EOC	L_∞	EOC	$\boxed{v_y}$ L_2	EOC	L_∞	EOC
2	10	4.48e-02	-	5.13e-01	-	2.49e+00	-	1.52e+01	-
	20	6.01e-03	**2.9**	1.44e-01	**1.8**	3.17e-01	**3.0**	7.22e+00	**1.1**
	40	5.18e-04	**3.5**	1.23e-02	**3.6**	4.58e-02	**2.8**	7.59e-01	**3.3**
	80	6.11e-05	**3.1**	1.52e-03	**3.0**	5.22e-03	**3.1**	1.65e-01	**2.2**
	160	7.41e-06	**3.0**	2.17e-04	**2.8**	5.83e-04	**3.2**	2.24e-02	**2.9**
3	10	5.28e-03	-	6.74e-02	-	5.07e-01	-	1.04e+01	-
	20	6.19e-04	**3.1**	1.94e-02	**1.8**	5.21e-02	**3.3**	2.22e+00	**2.2**
	40	1.61e-05	**5.3**	7.37e-04	**4.7**	2.57e-03	**4.3**	7.03e-02	**5.0**
	80	8.22e-07	**4.3**	4.69e-05	**4.0**	1.84e-04	**3.8**	1.36e-02	**2.4**
	160	4.74e-08	**4.1**	7.72e-06	**2.6**	8.49e-06	**4.4**	8.25e-04	**4.0**
4	10	1.28e+00	-	3.29e+01	-	1.05e-01	-	2.50e+00	-
	20	1.59e-04	**13.0**	6.23e-03	**12.4**	6.53e-03	**4.0**	2.01e-01	**3.6**
	40	6.10e-07	**8.0**	3.75e-05	**7.4**	1.83e-04	**5.2**	7.57e-03	**4.7**
	80	1.48e-08	**5.4**	1.04e-06	**5.2**	6.24e-06	**4.9**	5.19e-04	**3.9**
	160	4.58e-10	**5.0**	3.29e-08	**5.0**	1.36e-07	**5.5**	1.23e-05	**5.4**

p	n	$\boxed{q_x}$ L_2	EOC	L_∞	EOC	$\boxed{\omega}$ L_2	EOC	L_∞	EOC
2	10	3.98e+01	-	3.52e+02	-	1.41e+02	-	6.76e+02	-
	20	1.06e+01	**1.9**	1.02e+02	**1.8**	3.68e+01	**1.9**	2.77e+02	**1.3**
	40	2.86e+00	**1.9**	4.77e+01	**1.1**	1.04e+01	**1.8**	1.57e+02	**0.8**
	80	6.91e-01	**2.0**	1.31e+01	**1.9**	2.56e+00	**2.0**	4.62e+01	**1.8**
	160	1.61e-01	**2.1**	3.34e+00	**2.0**	6.12e-01	**2.1**	1.14e+01	**2.0**
3	10	7.43e+00	-	9.17e+01	-	4.10e+01	-	6.16e+02	-
	20	1.40e+00	**2.4**	2.71e+01	**1.8**	6.79e+00	**2.6**	1.77e+02	**1.8**
	40	1.43e-01	**3.3**	3.15e+00	**3.1**	7.79e-01	**3.1**	1.75e+01	**3.3**
	80	1.85e-02	**3.0**	1.20e+00	**1.4**	1.20e-01	**2.7**	5.43e+00	**1.7**

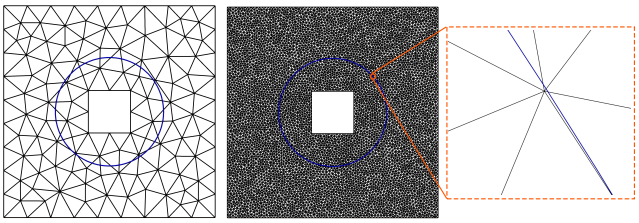


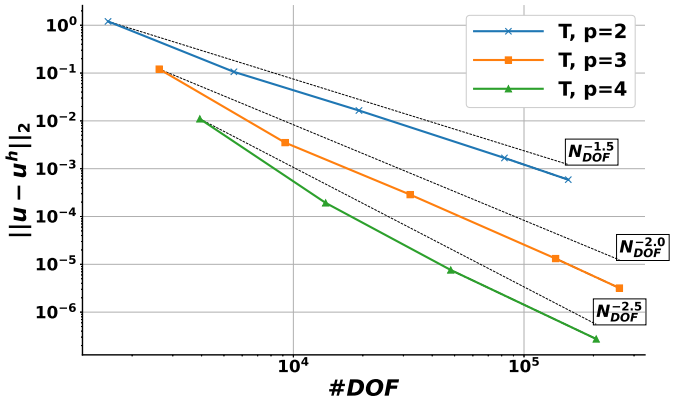
Figure 11.11 / Coarsest (left) and second finest (middle) unstructured triangular grids used for the *spinning bubble* test case. The zoom on the interface region (right) shows critical small cuts intersections.

	160	1.94e-03	**3.3**	1.51e-01	**3.0**	1.21e-02	**3.3**	5.46e-01	**3.3**
4	10	2.21e+00	-	5.72e+01	-	2.37e+01	-	1.51e+03	-
	20	1.98e-01	**3.5**	1.06e+01	**2.4**	1.20e+00	**4.3**	6.57e+01	**4.5**
	40	9.15e-03	**4.4**	4.46e-01	**4.6**	7.02e-02	**4.1**	2.93e+00	**4.5**
	80	7.12e-04	**3.7**	7.78e-02	**2.5**	4.84e-03	**3.9**	2.99e-01	**3.3**
	160	3.57e-05	**4.3**	5.86e-03	**3.7**	2.51e-04	**4.3**	1.58e-02	**4.2**

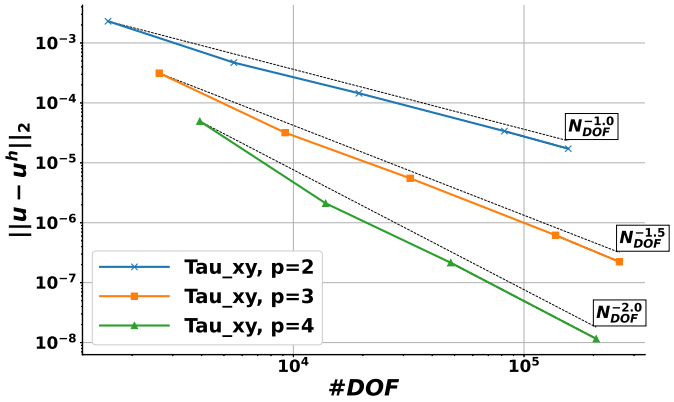
To close this grid convergence study section, the spinning bubble test case is repeated with the same physical and numerical parameters on a set of unstructured triangular grids. Two of them are shown in Figure 11.11. Figure 11.12 shows the L_2 convergence graphs for the temperature field and the xy-component of the shear stress tensor in terms of the number of degrees of freedom (N_{DOF}) on the mesh. The results are in agreement with the theoretical rate $\sim N_{\text{DOF}}^{-(p+1)/2}$ and $\sim N_{\text{DOF}}^{-p/2}$ for the pressure and shear stress tensor, respectively (Rivière (2008)). Those graphs emphasize also the benefits of high-order methods: for the same number of degrees of freedom, the errors are becoming smaller for increasing polynomial orders.

Sensitivity analysis

Influence of the bulk Riemann solver choice The very low Mach number regime of this situation brings additional numerical challenges linked to the disparity between the convective and acoustic time scales. Thus, using a standard bulk Riemann solver like HLLC (Toro (2019)) on this spinning bubble test case (V1) leads to errors curves that saturate. This behavior is all the more marked for increasing polynomial interpolation orders and finer grids. Saturation is observed from mesh 160×160 for $p = 2$ and from mesh 80×80 for $p = 3$ and $p = 4$. That is why the SLAU2 Riemann solver, introduced in Section 5.3.4, is selected here for its ability to provide accurate solution over a wide range of Mach numbers.



a / Temperature field.



b / *xy*-component of the shear stress tensor $\underline{\boldsymbol{\tau}}$.

Figure 11.12 / *Spinning bubble* test case, *V1* (Table 11.2) on *unstructured triangular* grids. Discretization errors in L_2 error norms for different polynomial degrees p as a function of the number of degrees of freedom (DOF). Dashed lines indicate theoretical order of convergence.

Influence of the variables set choice Solving the spinning bubble test case in terms of conservative variables was found to be impossible as the solver could not converge (keeping the same numerical parameters). This reflects the fact that for this particular case involving a very stiff phase, the Jacobian of the residual computed with respect to the conservative variables yields a system to be solved which is extremely ill-conditioned. To be able to still perform the analysis, a less stiff second version (without mass transfer) of the spinning bubble test case is considered in Table 11.9. In this case, the solver is able to make the residual converge to the prescribed tolerance, except for $p = 4$ on the finest grid for which it saturates. As Table 11.10 reveals, the convergence and the level of accuracy are negatively affected by this new choice of variable. This behavior can be associated to the imposition of the solution jumps by the transpose (Section 9.2.3) and penalty terms (Section 9.2.3). Since there is no explicit conversion to convert the physical jumps on primitive variables to another variables set, the jumps on the solution vector enforced through conservative variables are only an approximation of the exact prescribed jumps. This discrepancy does not seem to impact too much the solutions accuracy for the first "toy" inclined interface problem but has a non-negligible effect when harsh physical parameters are considered.

Table 11.9 / Changes in parameters for the *spinning bubble* test case *V2* compared to V1 in Table 11.2.

		Spinning bubble (V2)
Bulk properties	γ_1	1.0
	$p_{\infty,1}$	1×10^6
Jumps values	$\dot{m}_\Gamma$	0.0
	Δq	100.0
	ΔT	10.0
	Δv_t	-0.5

Table 11.10 / *Spinning bubble V2* (Table 11.9) with different choices of solved-for variables sets. Discretization errors and experimental orders of convergence (EOC) in L_2 error norms for polynomial degrees p on *Cartesian triangular* grids of $n \times n$ elements for the *pressure* field. Comparison between the use of primitive and conservative variables sets.

p	n	p-$\mathbf{v}$-T	ρ-$\rho\mathbf{v}$-ρE

		L_2	EOC	L_∞	EOC	L_2	EOC	L_∞	EOC
2	10	6.59e+01	-	9.02e+02	-	2.53e+02	-	3.45e+03	-
	20	1.13e+01	**2.5**	2.37e+02	**1.9**	2.78e+01	**3.2**	5.55e+02	**2.6**
	40	7.29e-01	**4.0**	1.70e+01	**3.8**	2.23e+00	**3.6**	7.34e+01	**2.9**
	80	8.35e-02	**3.1**	3.00e+00	**2.5**	1.01e+00	**1.1**	1.37e+02	**0.9**
3	10	5.18e+00	-	5.48e+01	-	2.24e+02	-	1.85e+03	-
	20	3.51e-01	**3.9**	9.81e+00	**2.5**	2.55e+00	**6.5**	3.36e+01	**5.8**
	40	1.72e-02	**4.3**	3.58e-01	**4.8**	8.85e-02	**4.8**	6.38e+00	**2.4**
	80	7.91e-04	**4.4**	3.09e-02	**3.5**	1.26e-02	**2.8**	1.93e+00	**1.7**
4	10	9.37e-01	-	1.09e+01	-	1.63e+03	-	1.67e+04	-
	20	4.58e-02	**4.4**	7.21e-01	**3.9**	2.14e-01	**12.9**	1.16e+01	**10.5**
	40	9.72e-04	**5.6**	2.02e-02	**5.2**	4.90e-03	**5.4**	2.62e-01	**5.5**
	80	2.09e-05	**5.5**	8.20e-04	**4.6**	5.07e-04	**3.3**	1.15e-01	**1.2**

Flower-like geometry

The spinning bubble test case is finally adapted to a more complex interface geometry, namely a flower (or starfish) shape represented in Figure 11.13 and whose equation is given by

$$\varphi_0 \equiv \sqrt{(x-1)^2 + (y-1)^2} - 0.06\cos\left(5.0\,\mathrm{atan2}(y-1, x-1)\right) - 0.25625 = 0\,. \quad [11.4]$$

The properties selected are the same as the ones in Table 11.2. The analytical solutions and source terms provided by the method of manufactured solution (MMS) are also identical. However, it is important to note that the imposition of the jumps into our solver is adapted for this configuration. Instead of giving as inputs the values of the jumps parameters (Table 3.2) used during the MMS process, we instead directly evaluate the analytical solution and its gradients along the interface to set the jumps we want to impose on the pressure, the velocities, the temperature and the diffusive fluxes. This means that the physical jump conditions, Equation 3.38 and Equation 3.40, are not respected on the flower-shaped interface, they are strictly respected only on the circular interface. The reason is that we could not manufacture a solution which would satisfy the complex jump conditions on an interface with an arbitrary shape. But that does not make this test any less interesting since the ability of the solver to impose prescribed jumps, through the multiphase Riemann solver and the interior penalty method, is still exercised. In order to present the results from a slightly different point of view, the errors are here measured by means of the L_2 norm

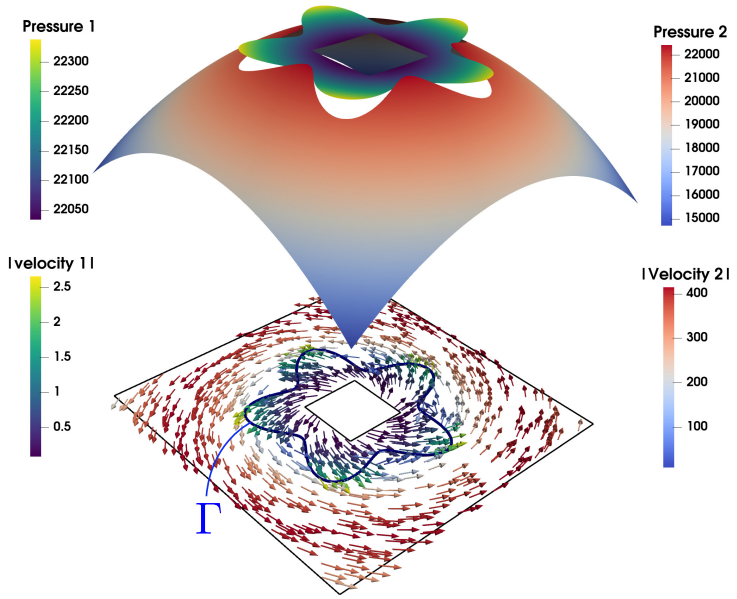


Figure 11.13 / Domain of the *flower shape* test case with the position of the phase interface Γ. Arrows represent direction of the velocity field for both phases. The elevated pressure field (not to scale) is also drawn.

restricted to the level-set iso-0, with an error for each phase, i.e.

$$\|\mathbf{u}_k - \tilde{\mathbf{u}}_k\|^2_{L_{2,\Gamma(\Omega)}} = \sum_{m \in \mathbb{I}} \int_{\mathbb{I}_m} (\mathbf{u}_k - \tilde{\mathbf{u}}_k)^2 \, \mathrm{d}s \,. \qquad [11.5]$$

Moreover, instead of evaluating the accuracy based on solution variables, the normal projections of the convective and diffusive fluxes along the interface are here selected as error measures. The errors are listed in Table 11.11. Overall, the errors are decreasing as expected with the refinement of the grids and the superiority of higher order discretizations for comparable number of degrees of freedom manifests itself again. Irregularities in the values of the convergence rates can nonetheless be noticed. Those probably come from the use of the specific $L_{2,\Gamma}$ error norm (Equation 11.5). It was checked that in terms of the L_2 norm (Equation 11.2), the theoretical convergence rates of $N_{\mathrm{DOF}}^{-(p+1)/2}$ on the primitive variables were indeed retrieved as in the case of the circular interface. The high values of the absolute errors on the normal energy convective fluxes can be explained by the high value of the product between the density and the total enthalpy. This last test case emphasizes the ability of the solver to reach high-order accuracy on non-trivial interface shapes for quantities involving products of solutions variables or their gradients.

Table 11.11 / *Flower shape* test case. Discretization errors and experimental orders of convergence (EOC) in the $L_{2,\Gamma}$ error norm for polynomial degrees p on *unstructured triangular* meshes containing an increasing number of degrees of freedom (N_{DOF}). Results per phase k for the interface normal projection of the x-momentum ($\mathbf{F}_{k,2}$) and total energy ($\mathbf{F}_{k,4}$) components of the convective and diffusive fluxes.

p	N_{DOF}	$\boxed{\mathbf{F}^c_{k,2}\cdot\mathbf{n}_\Gamma}$				$\boxed{\mathbf{F}^c_{k,4}\cdot\mathbf{n}_\Gamma}$			
		$k=1$	EOC	$k=2$	EOC	$k=1$	EOC	$k=2$	EOC
3	2620	9.34e+00	-	1.92e+00	-	9.05e+06	-	1.76e+04	-
	9200	2.05e-01	**2.2**	1.11e-01	**2.2**	8.42e+05	**2.4**	6.80e+02	**2.2**
	32080	2.02e-02	**1.8**	7.96e-03	**1.7**	6.71e+04	**1.8**	5.59e+01	**2.2**
	137080	1.43e-03	**1.9**	6.55e-04	**2.1**	4.82e+03	**2.0**	2.15e+00	**2.0**
	259200	3.42e-04	**3.0**	1.61e-04	**2.3**	1.07e+03	**1.9**	5.28e-01	**2.6**
4	3930	3.14e+00	-	4.64e-01	-	2.23e+06	-	4.95e+03	-
	13800	1.73e-02	**2.4**	7.55e-03	**2.2**	8.00e+04	**2.5**	5.52e+01	**2.7**
	48120	1.10e-03	**2.2**	4.38e-04	**2.3**	3.83e+03	**2.4**	1.98e+00	**2.7**
	205620	3.48e-05	**4.1**	1.71e-05	**3.3**	9.79e+01	**2.7**	4.16e-02	**3.6**

p	N_{DOF}	$\boxed{\mathbf{F}^d_{k,2}\cdot\mathbf{n}_\Gamma}$				$\boxed{\mathbf{F}^d_{k,4}\cdot\mathbf{n}_\Gamma}$			
		$k=1$	EOC	$k=2$	EOC	$k=1$	EOC	$k=2$	EOC
3	2620	3.87e-04	-	3.42e-03	-	8.57e+00	-	1.81e+01	-
	9200	6.54e-05	**1.4**	4.24e-04	**1.3**	4.10e+00	**1.9**	6.50e+00	**1.1**
	32080	1.29e-05	**1.2**	6.34e-05	**1.7**	3.06e-01	**1.0**	5.97e-01	**2.1**
	137080	2.20e-06	**1.3**	5.41e-06	**1.5**	6.76e-02	**2.1**	2.83e-02	**1.9**
	259200	8.95e-07	**1.4**	2.38e-06	**1.7**	1.98e-02	**0.6**	1.38e-02	**0.8**
4	3930	2.02e-04	-	1.98e-03	-	5.99e+00	-	8.63e+00	-
	13800	1.01e-05	**2.1**	4.69e-05	**2.1**	5.59e-01	**1.9**	9.98e-01	**2.4**
	48120	1.52e-06	**1.5**	3.13e-06	**2.2**	3.25e-02	**2.3**	3.11e-02	**2.8**
	205620	7.37e-08	**2.4**	1.43e-07	**3.0**	2.03e-03	**1.9**	8.96e-04	**1.7**

11.4 Benchmark two-phase test cases in the incompressible regime

In this section, two unsteady test cases picked in the two-phase flows incompressible literature are considered. The goal of this section and the next one is to demonstrate the ability of the solver to handle a wide range of flow conditions. The numerical and physical parameters used for the two test cases are grouped in Table 11.12.

Table 11.12 / Numerical and physical parameters for the two-phase flows test cases in the *incompressible regime*.

	Parameters	Rising bubble	Oscillating bubble
Spatial discretization	Cartesian grid	37×62	32×32
	Cell size Δx (m)	4.0e-3	3.13e-7
	Interp. order	$p = 2$	$p = 2$
	Riemann solver	AUSM+-up	AUSM+-up
Temporal discretization	Time step Δt (s)	2.5e-4	1e-9
	Schemes (LS/XDG)	BDF2/BDF2	BDF2/BDF2
Physical parameters	EOS	Stiffened	Stiffened
	γ_1, γ_2 (-)	1.4, 7.14	1.4, 4.4
	$C_{v,1}$, $C_{v,2}$ (J/(kg K))	852.8, 184.0	696.2, 604.0
	$p_{\infty,1}$, $p_{\infty,2}$ (N/m^2)	0.0, 3.3e8	0.0,, 6.0e8
	μ_1, μ_2 (kg/(m s))	4.5e-3, 3.5e-2	1.8e-5 , 8.9e-4
	λ_1, λ_2 (W/(m K))	0.0, 0.0	0.0, 0.0
	p_1, p_2 $(t = 0)$ (N/m^2)	1.0e5, 1.0e5	1.0e5, 1.0e5
	T_1, T_2 $(t = 0)$ (K)	293.15, 293.15	293.15, 293.15
	ρ_1, ρ_2 $(t = 0)$ (kg/m^3)	1.0, 1000.0	1.2, 997.0
	Nondim. numbers	Re = 1000.0 Bnd = ∞	$\omega/\omega_{\text{ref}} = 1.38$

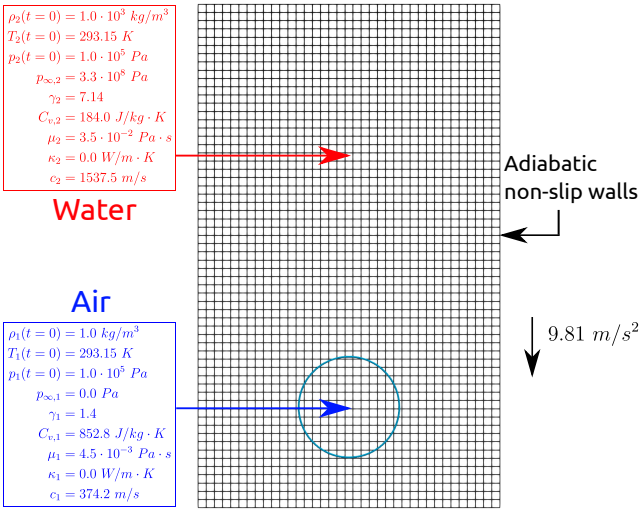


Figure 11.14 / Initial configuration of the *rising bubble* test case with the physical parameters in each phase.

11.4.1 Rising bubble

Setup

The first test case is about the rise of an air bubble submerged in a water column initially at rest under the sole influence of the buoyancy force. The initial configuration is shown in Figure 11.14. The parameters are directly taken from Zou (2021, Sec. 7.4), except that surface tension is neglected here. The initial radius of the bubble is $R_0 = 0.025$m from which the Reynolds and Bound numbers can be defined as

$$\mathrm{Re} = \frac{\rho_{l,0}\, v_{\mathrm{ref}}\, D_0}{\mu_l},$$
$$\mathrm{Bnd} = \frac{4\, \rho_{l,0}\, g\, R_0^2}{\sigma},$$

with the reference velocity $v_{\mathrm{ref}} = \sqrt{2\, g\, R_0}$. Because no surface tension is taken into account here, $\sigma = 0$N/m which leads to $\mathrm{Bnd} = \infty$. We can also estimate the representative Mach number of the problem based on the value of speed of sound in the water and v_{ref}, yielding a value of the order of $4 \cdot 10^{-4}$, which represents clearly a challenging regime for a compressible solver.

Results

The velocity fields and streamlines in both phases are depicted in Figure 11.15 where we can observe their symmetry and their vortex-like structure corre-

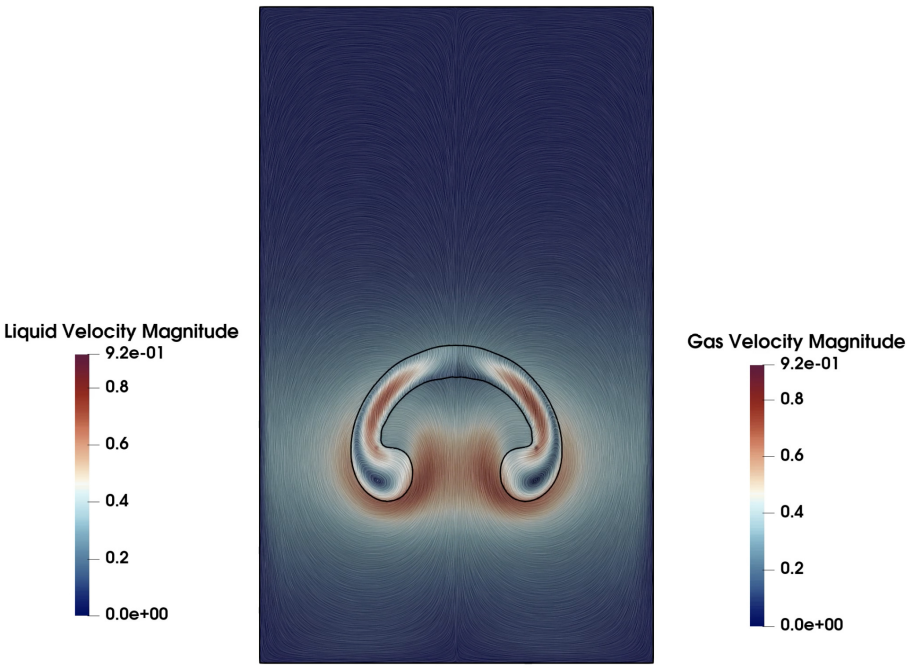


Figure 11.15 / Bubble shape, velocity and streamlines of the *rising bubble* test case at time $t = 0.19$s.

sponding to the skirted shape of the deformed bubble. A qualitative comparison with the reference work of Sussman et al. (1994) is presented in Figure 11.16. A good quality match in the bubble shapes at different nondimensional times can be noticed.

11.4.2 Rayleigh–Plesset oscillating bubble

Setup and reference solution

For this second test case, we simulate the oscillations of an air bubble immersed in a water domain with boundaries on which a time fluctuating pressure function is prescribed as

$$p_{\mathrm{BC}}(t) = 10^5 \ [1 + p_0 \sin(\omega t)] \ ,$$

with p_0 the amplitude of the pressure fluctuations and ω its angular frequency. The initial radius of the bubble is $R_0 = 2 \cdot 10^{-6}$m and the half-length of the domain is $R_\infty = 5$e-6m. The initial configuration is depicted in Figure 11.17. Inspired by Jain et al. (2020), we set $p_0 = 10^4$ and $\omega = 10208967.75$. The natural

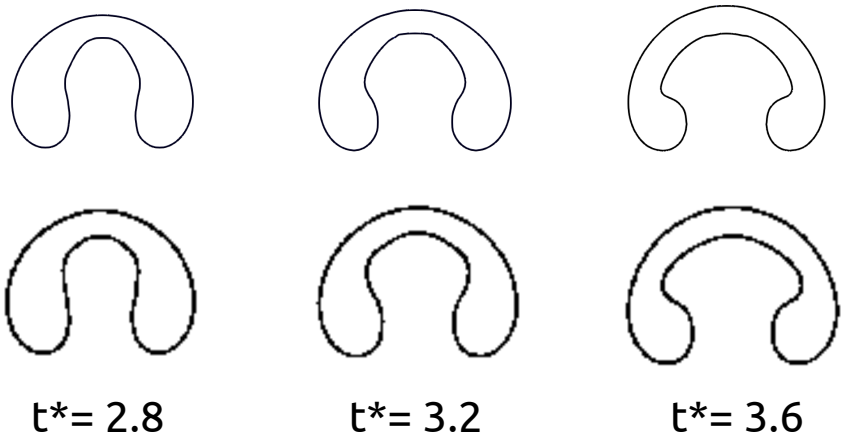


Figure 11.16 / Bubble shape comparison at different non-dimensional times, $t^* = \sqrt{R_0/g}$, for the *rising bubble* test case at time. Top row – our computations, bottow row – results from Sussman et al. (1994, Fig. 14).

frequency of the problem can be estimated by the formula

$$\omega_{\text{ref}}^2 = \frac{2\,p_0}{\rho_{l,0}\,R_0^2\log\left(R_\infty/R_0\right)},$$

which yields to a ration $\omega/\omega_{\text{ref}} = 1.38$. The semi-analytical of the Rayleigh-Plesset problem in two dimensions is derived in Appendix D.

Results

The temporal profiles of the computed and analytical normalized radius are plotted in Figure 11.18. There is a fair agreement between the two even though we can notice that they tend to deviate more a more as time goes by. One reason to explain this is directly related to the assumptions on which the Rayleigh-Plesset semi-analytical solution lies, namely that it considers a truly incompressible liquid phase. The pressure waves propagation is thus inevitably slightly different compared to a compressible model, even by modeling the liquid with very stiff parameters, as we do here.

11.5 Benchmark two-phase test cases in the compressible regime

We turn now our attention to three test cases for which compressibility effects are not negligible. The two first ones are one-dimensional Riemann problems that are treated here as two-phase flows, contrary to the first Riemann test case

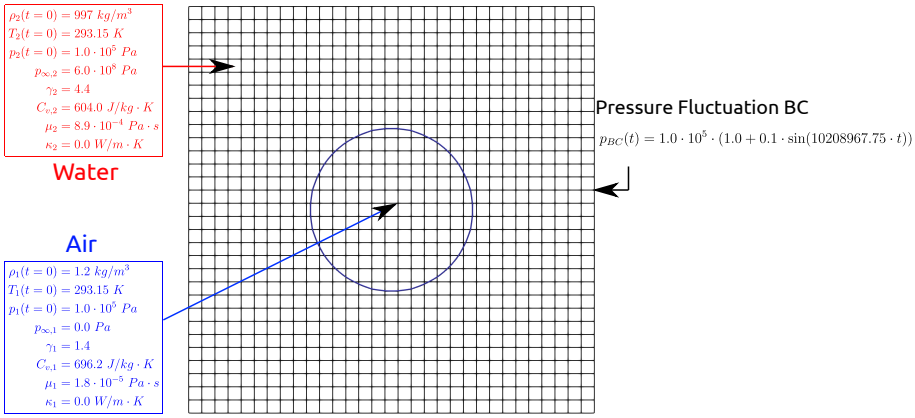


Figure 11.17 / Initial configuration of the *Rayleigh-Plesset oscillating bubble* test case with the physical parameters in each phase.

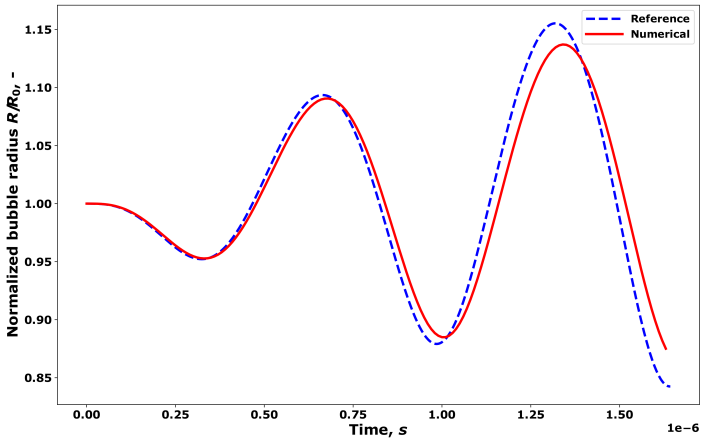


Figure 11.18 / Time evolution of the radius of the *Rayleigh-Plesset oscillating bubble*. Comparison between numerical and semi-analytical reference solution.

considered in Section 11.2. The parameters are gathered in Table 11.13.

Table 11.13 / Numerical and physical parameters for the two-phase flows test cases in the *compressible regime*.

Parameters		Sod Riemann	Gas-liquid Riemann	Shock-bubble interaction
Spatial discretization	Cartesian grid	200×1	200×1	unstruct.
	Cell size Δx (m)	5.0e-3	4.0e-2	unstruct.
	Interp. order	$p = 2$	$p = 2$	$p = 2$
	Riemann solver	HLLC	SLAU	SLAU
Temporal discretization	Time step Δt (s)	1.0e-4	2.5e-6	1e-7
	Schemes (LS/XDG)	RK44/BDF2	RK44/BDF2	BDF2/BDF2
Physical parameters	EOS	PG	Stiffened	PG
	γ_1, γ_2 (-)	1.4, 1.4	1.4, 5.5	1.4, 1.648
	$C_{v,1}$, $C_{v,2}$ (J/(kg K))	2.5, 2.5	747.0, 4000.0	720.0, 2443.3
	$p_{\infty,1}$, $p_{\infty,2}$ (N/m^2)	0.0, 0.0	0.0, 4.9e8	0.0, 0.0
	p_1, p_2 ($t = 0$) (N/m^2)	1.0, 0.1	1.0e7, 1.0e5	1.0e5/1.57e5, 1.0e5
	ρ_1, ρ_2 ($t = 0$) (kg/m^3)	1.0, 0.125	10.0, 1000.0	1.0/1.38, 0.18

11.5.1 Sod Riemann test case

Setup and reference solution

The original Sod test case is modified by imposing a pressure jump $\Delta p_\sigma = 0.2$ across the contact discontinuity, as it is done in S. Fechter et al. (2013) and sketched in Figure 11.19. In this situation, the apparition of the rarefaction fan on the left and especially the presence of the shock propagating to the right require the use of artificial viscosity to damp the oscillations near the strong gradients regions. At the beginning of the simulation, the shock starts to develop in the cells cut by the interface and it is therefore important to have a correct way of estimating the artificial viscosity. The derivation of the semi-analytical solution is explained in Appendix D.

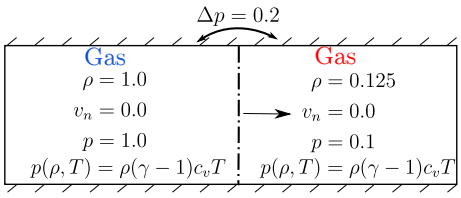


Figure 11.19 / Initial configuration of the *Sod Riemann* test case with the physical parameters in each phase.

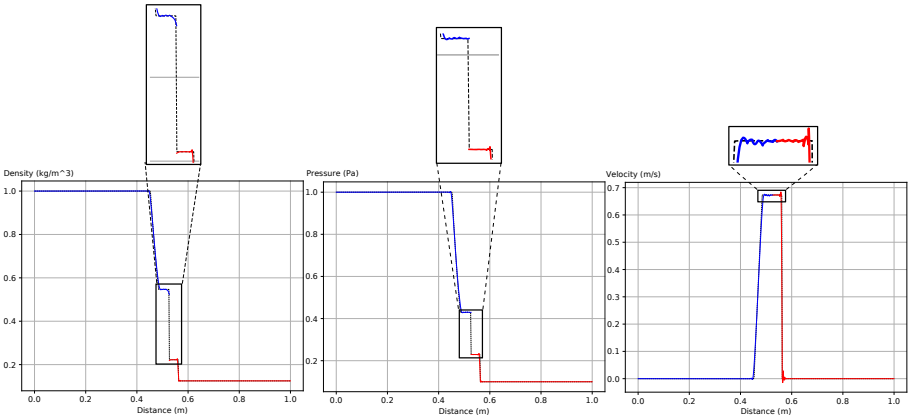


Figure 11.20 / Comparison of the numerical *Sod Riemann* test case solution with the analytical one (dashed lines) at time $t = 0.02$s.

Results

The results in Figure 11.20 are found to be in very good agreement with the exact solution. The jump in pressure at the contact discontinuity is perfectly imposed, as well as the continuity of the velocity, in accordance with the prescribed jumps. There are slight inclinations in the density profiles near the interface, that could be caused by the addition of the artificial viscosity. This effect is similarly visible in the temperature plot. But it is fair to say that the contact discontinuity is still successfully kept sharp and that its movement is accurately predicted.

11.5.2 Gas-liquid Riemann test case

Setup and reference solution

The purpose of this test case is to demonstrate the performance of the method in a two-phase setting under challenging conditions and is inspired from the one proposed in Rouy et al. (2004). It deals with a gas-liquid flow with initial ratio of pressure and density equal to 100. The gas is represented by a perfect gas EOS while the liquid obeys a stiffened gas EOS. In addition, a pressure jump between both phases is imposed, $\Delta p = 10$ bar, as represented in Figure 11.21.

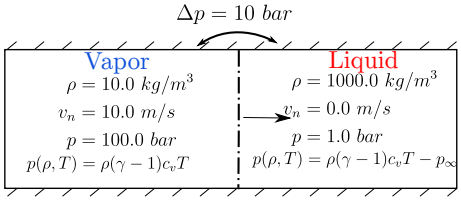


Figure 11.21 / Initial configuration of the *gas-liquid Riemann* test case with the physical parameters in each phase.

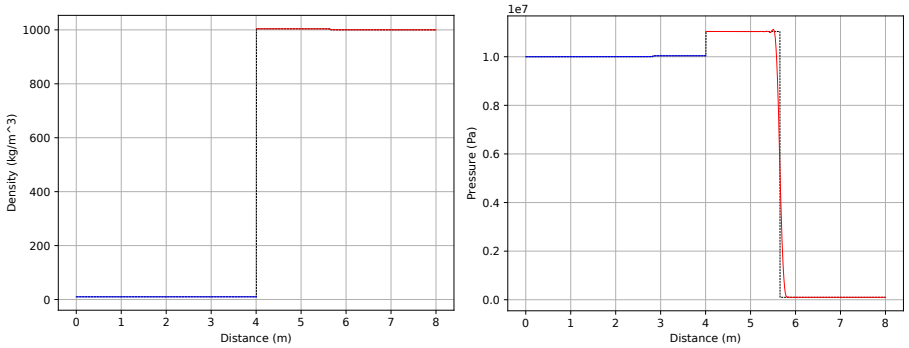


Figure 11.22 / Comparison of the numerical *gas-liquid Riemann* test case solution with the analytical one (dashed lines) at time $t = 0.001$s.

Results

Figure 11.22 shows that the important discontinuity in the density field across the phase interface is perfectly captured by the numerical scheme. The imposed pressure jump is respected while the normal velocity is continuous between both phases. The effect of the artificial viscosity is illustrated in Figure 11.23 on the velocity field where the important spurious oscillations are removed near the shock and rarefaction waves.

11.5.3 Shock-bubble interaction

Setup and reference solution

This test case is a classic in the compressible two-phase flows community. It consists in a Helium bubble of initial radius 0.025m embedded in air. A Mach 1.22 shock wave is inserted into the gas freestream. When it impacts the bubble, a reflected rarefaction wave rebounds in the opposite shock direction while a refracted wave is transmitted inside the bubble and moves faster than the incident shock. The initial configuration is shown in Figure 11.24. A recent detailed numerical study about this problem can be found e.g. in Kundu (2021).

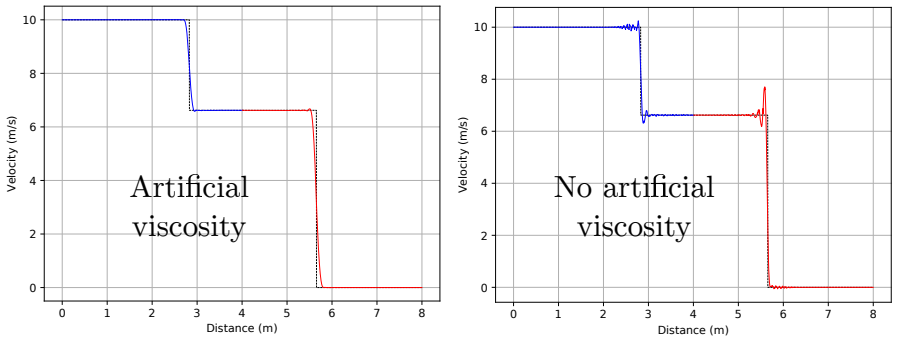


Figure 11.23 / Illustration of the effects of artificial viscosity on the velocity profile of the *gas-liquid Riemann* test case.

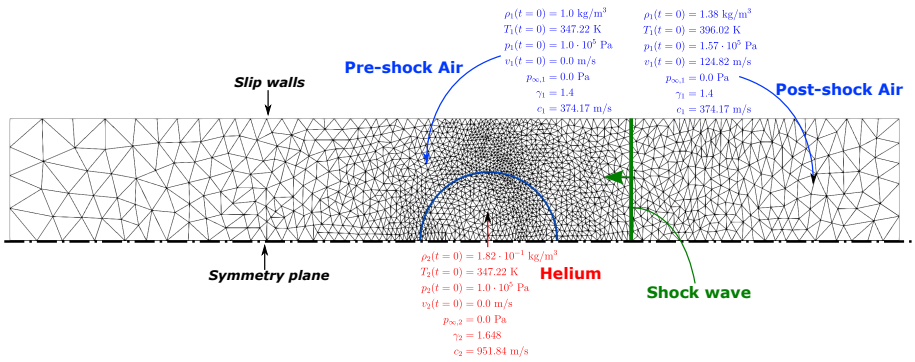


Figure 11.24 / Initial configuration of the *shock-bubble interaction* test case with the physical parameters in each phase.

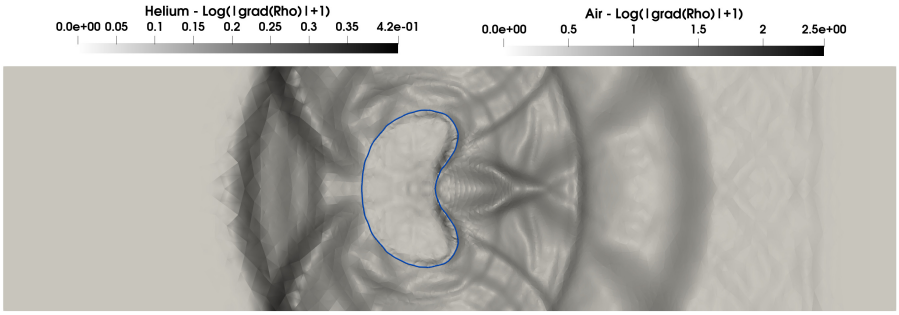


Figure 11.25 / Numerical Schlieren visualization of the *shock-bubble interaction* test case at time t = 2.5e-4s.

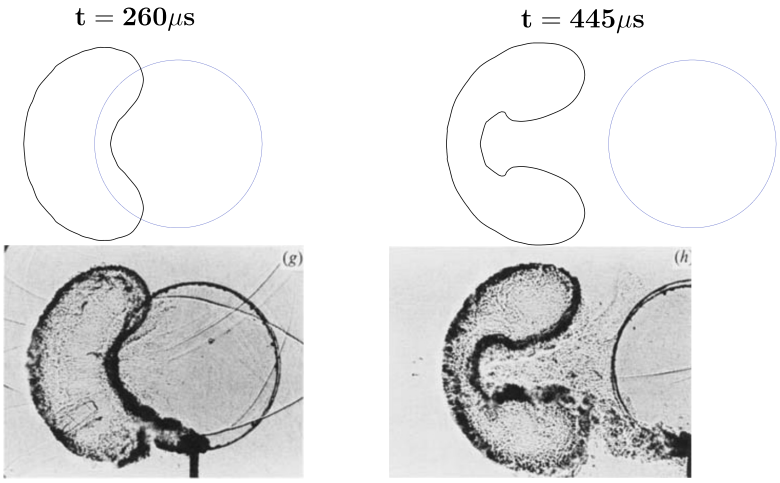


Figure 11.26 / Comparison of the numerical *shock-bubble interaction* test case results (top row) with experimental results from Haas and Sturtevant (1987, Fig. 7) (bottom row).

Results

Figure 11.25 presents a numerical Schlieren snapshot on which we can observe all the wave patterns resulting from the interactions between the reflected and refracted shocks on the bubble and the shocks reflected on the lateral boundaries. It is apparent from this figure that the numerical solution is severely under-resolved in some flow regions. A finer mesh and/or adaptation techniques would be required to obtain results of better quality. Despite this, the overall shape of the Helium bubble is well captured by our solver, as the comparison with the reference experimental pictures in Figure 11.26 testifies.

11.6 Benchmark two-phase test cases with surface tension

After having identified the curvature filtering strategies offering the best accuracy and the minimal oscillations along the interface in Section 7.3, three moving interface test cases are now studied. The objective is to assess how well the curvature filtering procedure allows to obtain a stable, robust and accurate two-phase flow solver in the presence of non-negligible surface tension forces. While the analysis of the previous section was restricted to the curvature computation alone, the results presented below involve the highly non-linear coupling that exists between the interface curvature computation, the interface advection and the flow solver. This coupling manifests itself by the presence of the curvature dependent term in the jump conditions linking the governing equations across the interface (see Section 3.4) and the level-set advection velocity that is determined from the resulting fluid velocities satisfying the jumps. An overview of the numerical parameters selected for each test case is given in table Table 11.14. The nondimensional numbers of interest are defined by

$$\begin{aligned} \mathrm{La} &= \frac{\sigma \rho L}{\mu^2} && \text{Laplace number,} \\ \mathrm{Eo} &= \frac{\rho V^2 L}{\sigma} && \text{Eötvös number,} \\ \mathrm{Re} &= \frac{\rho V L}{\mu} && \text{Reynolds number.} \end{aligned}$$

It should be mentioned that the spatial resolution is (very) coarse for all the test cases. The main reason for this choice is related to the quite important computational overhead induced by the use of the XDG method and the curvature filtering techniques combined with the limitation of this part of the code to run only in a shared memory parallel environment (OpenMP) and not in a distributed one.

Choice of curvature filtering

For the three test cases, the level-set from which the curvature is computed is the one enforced to be continuous across the element faces because it is the most consistent with the XDG fluid solver that relies on the continuous level-set field and because it was observed in Section 7.3 that the choice of the original discontinuous versus continuous level-set does not have a significant influence on the curvature computation. Moreover, the derivatives of the level-set field are computed based on the lifting approach for all the test cases as they seem to clearly offer better results than the broken ones. For the two first test cases, strategies among the most promising ones identified before are selected. This translates into the use of the Bonnet formula on the interpolation point (IP) or directly on the iso-0 quadrature points and the use of combinations of patch-recovery and

partition of unity filtering steps, either on the curvature or on the gradients and Hessian matrix. On the contrary, for the third test case, seemingly sub-optimal options are selected. There are two reasons for this choice. The first motivation is to show that filtering procedures which are not at the top of the rankings established in section Section 7.3 based on some specific criteria can still be considered. This applies particularly for the artificial viscosity filtering whose performance strongly depends on the choice of the viscosity coefficient which may not have been optimally set for the static test cases analyzed before. The second reason is related to the particular configuration of the standing waves test case. It was observed that the patch-recovery and partition of unity techniques do not perform as well near the domain boundaries. This is due to the fact that the information collected from the neighboring elements is truncated, which seems to perturb the filtering process. In such a situation, it appears that the artificial viscosity is better suited, given the current implementation of the two other filters.

Table 11.14 / Numerical and physical parameters for the surface-tension driven test cases.

	Parameters	Oscillating droplet	Rising bubble	Standing waves
Curvature filtering	Formula	Bonnet IP	Bonnet Isoo	Div. normal
	LS Type	Continuous	Continuous	Continuous
	Deriv. choice 1	N/A	N/A	Lifted
	Deriv. choice 2	Lifted	Lifted	Lifted
	Filt. LS	None	None	None
	Filt. normal	N/A	N/A	None
	Filt. grad-Hessian	None	PR2+PU	N/A
	Filt. curvature	PR2	None	AV 0.05
	Interp. iso-0	PU	None	None
Spatial discretization	Cartesian grid	16×16	21×42	11×33
	Cell size Δx (m)	7.50e-4	4.76e-2	9.09e-2
	Interp. order	$p = 3$	$p = 3$	$p = 3$
	Riemann solver	SLAU2	SLAU2	AUSM+-up
Temporal discretization	Time step Δt (s)	1e-5	1e-3	1e-3
	Schemes (LS/XDG)	RK44/BDF2	BDF2/BDF2	BDF2/BDF2

Physical parameters	EOS	Stiffened	Stiffened	Stiffened
	γ_1, γ_2 (-)	7.0, 1.4	2.1, 1.4	7.14, 7.14
	$C_{v,1}$, $C_{v,2}$ (J/(kg K))	65.76, 740.68	60.87, 717.50	977.52, 9775.24
	$p_{\infty,1}$, $p_{\infty,2}$ (N/m^2)	5.0e9, 0.0	2.0e7, 0.0	6.0e4, 6.0e4
	μ_1, μ_2 (kg/(m s))	5.9e-3, 5.8e-5	1.0e1, 1.0e-1	1.0e-1, 1.0e-2
	λ_1, λ_2 (W/(m K))	0.0, 0.0	0.0, 0.0	0.0, 0.0
	p_1, p_2 $(t = 0)$ (N/m^2)	1.01e5, 1.01e5	8.61e4, 8.61e4	20.0, 20.0
	T_1, T_2 $(t = 0)$ (K)	1800.0, 1800.0	300.0, 300.0	1.0, 1.0
	ρ_1, ρ_2 $(t = 0)$ (kg/m^3)	7040.0, 0.19	1000.0, 1.0	10.0, 1.0
	σ (N/m)	0.935	1.96	1.0
	Nondim. numbers	La = 5.77e5	Eo = 125 Re = 35	La = 1.0e3
	Ratio ρ_1/ρ_2	37053	1000	10
	Ratio μ_1/μ_2	101	100	10

11.6.1 Oscillation of a deformed liquid iron droplet

Setup and reference solution

The first test case consists in the oscillation of an initially deformed droplet due to the only effects of the surface tension forces. An initial modal radial displacement perturbation, $R^*_{n,0}$, is added to the original circular droplet equation of radius R_0, centered at $(x, y) = (0, 0)$, such that the initial equation of the interface is

$$\varphi_0(x, y) \equiv \sqrt{(x^2 + y^2)} = R_0 + R^*_{n,0}(\theta)\,,$$

with

$$R^*_{n,0}(\theta) = R_0 \left(\epsilon_{n,0} \cos(n\theta) - \frac{1}{4}\epsilon^2_{n,0} \right).$$

Here $\epsilon_{n,0}$ is the initial normalized amplitude of the perturbation, n the index of the perturbation mode and θ the angular coordinate in the polar reference frame, i.e. $\theta = \text{atan2}(y, x)$. An analytical solution in terms of the angular frequency ω_n and the damping parameter λ_n of the nth mode was derived by Rayleigh et al. (1879) and Lamb (1932) and recently improved by Aalilija et al. (2020), accounting for general viscous forces in the bulk of the fluid. In the

underdamped regime, the temporal evolution of the normalized modal radius displacement in two dimensions is given by Aalilija et al. (2020) as

$$R_n^*(\theta, t) = R_0 \left(\epsilon_n(t) \cos(n\theta) - \frac{1}{4}\epsilon_n^2(t) \right), \qquad [11.6]$$

with

$$\begin{aligned} \epsilon_n(t) &= \epsilon_{n,0}\, e^{-\lambda_n t} \cos\left(\omega_n t\right), \\ \omega_n &= \sqrt{\omega_{n,0}^2 - \lambda_n^2}, \\ \omega_{n,0}^2 &= n(n-1)(n+1)\frac{\sigma}{\rho_l R_0^3}, \\ \lambda_n &= 2n(n-1)\frac{\mu_l}{\rho_l R_0^2}, \end{aligned}$$

with the liquid density ρ_l and dynamic viscosity μ_l. The setup of the test case is the same as the first one proposed in Aalilija et al. (2020). The undeformed radius is $R_0 = 3 \cdot 10^{-3}$m and the initial perturbation $\epsilon_{n,0} = 0.02$. The fifth mode of deformation, $n = 5$, is here selected. The period of oscillation of the droplet can be obtained as

$$T_n = 2\pi/\omega_n = 8.17818 \cdot 10^{-3}\,\mathrm{s}.$$

The droplet lies in the middle of a square domain $[-2R_0, 2R_0] \times [-2R_0, 2R_0]$. The other parameters can be found in Table 11.14. We ca highlight the very high contrast in the properties (assumed constant!) between the liquid and the gas phases which correspond to liquid iron and air at a temperature of 1800K. The liquid phase is here extremely stiff, which puts the all-speed capabilities of the solver through its paces.

Results

Figure 11.27 depicts the velocity and elevated pressure fields of the oscillating droplet at time $t = 7T_n/8$. The direction of the velocity inside the bubble is in accordance with the variations of pressure induced by the surface tension forces. One can point out the smoothness at the interface with which the pressure jump is imposed (no spurious oscillations). From a quantitative point of view, Figure 11.28 presents the comparison between the analytical solution, Equation 11.6, and the numerical solutions, both with and without curvature filtering. One can observe that without filtering, the radius quickly exhibits an oscillating time behavior before the simulation crashes. With filtering, the simulation proceeds until the end. Although the agreement with the reference solution is good over most part of the oscillation period, a deviation appears near the end. The slight overshoot at the maximum contraction of the droplet and the damped oscillation after one period were also reported in Colomés et al. (2021) on their coarsest meshes. Thus, they can certainly be attributed to the numerical errors coming from the coarse mesh resolution considered here which is not compensated by the third interpolation order with which the

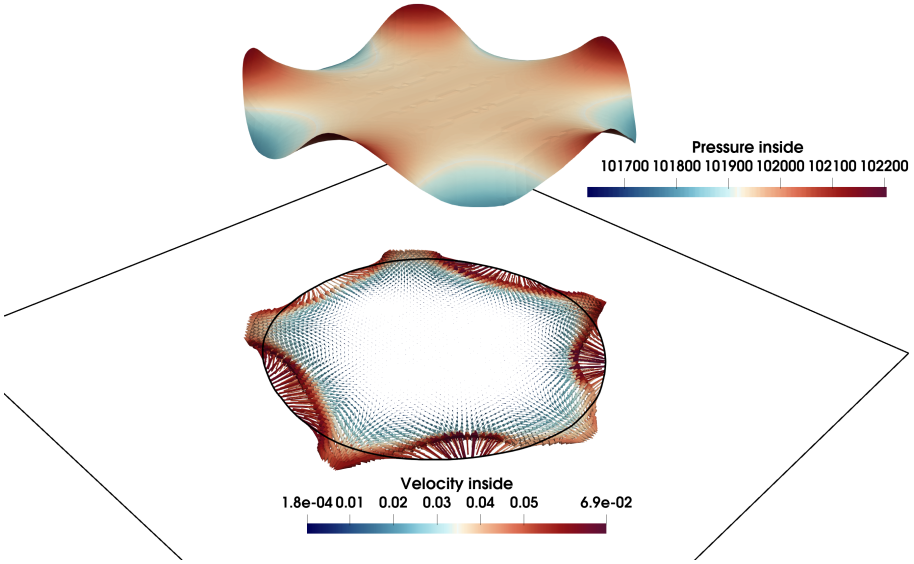


Figure 11.27 / *Oscillating droplet* – Velocity and elevated pressure fields at time $t = 7T_n/8$.

solution is discretized.

11.6.2 Buoyancy-driven rise of a bubble

Setup and reference solution

This test case is inspired by the benchmark two-dimensional incompressible test cases established by Hysing et al. (2009). The corresponding parameters are presented in Table 11.14. An initially circular gas bubble of radius $R_0 = 0.25\text{m}$ is placed such that its center is located at $(x, y) = (0.0, 0.5)$ in a rectangular column of liquid $[-2R_0, 2R_0]\times[0.0, 8R_0]$. The gravity acceleration set to $g = -0.98\text{m/s}^2$ induces a buoyant force which drives the bubble upwards. The deformation of the bubble during its rise is then influenced by its rise velocity (determined by the gravitational force), the viscous effects and the surface tension forces.

Results

The two quantities used to compare the results with the reference ones are the vertical position of the center of mass,

$$y_c(t) = \frac{\int_{\Omega_2(t)} y \, \mathrm{d}V}{\int_{\Omega_2(t)} 1 \, \mathrm{d}V},$$

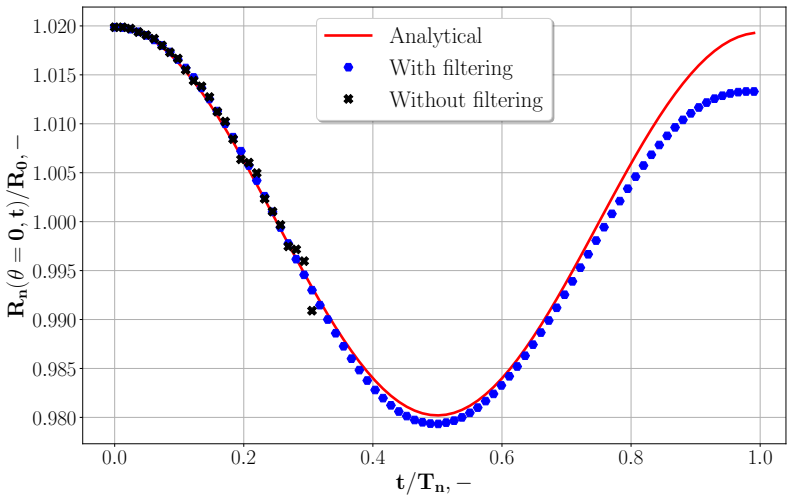


Figure 11.28 / *Oscillating droplet* – Temporal evolution of droplet radius along $\theta = 0°$ direction. Comparison between analytical and numerical solutions, both with and without filtering.

where $\Omega_2(t)$ is the time-dependent region occupied by the bubble, and the degree of circularity of the bubble defined as

$$c(t) = \frac{\text{Perimeter of initial circular bubble}}{\text{Perimeter of the deformed bubble}} = \frac{2\pi R_0}{P_b(t)}.$$

One can remark that the integrals involved in the two quantities above are evaluated with the dedicated quadrature rules for implicitly defined geometries, also employed for the assembly of the XDG residual. The characteristic timescale with respect to which the results are plotted is given by $T_r = 2R_0/V_r$, with the reference gravitational velocity $V_r = \sqrt{|g|\,2R_0}$. Figure 11.29 shows the deformed bubble at the end of the simulation with the velocity fields superposed. The skirted shape is in agreement with previous reference experimental and numerical studies. Figure 11.30 and Figure 11.31 present a quantitative comparison based on the center of mass and circularity benchmark quantities. For both of them, the numerical results follow the trend of the reference solutions with a fair level of errors although a time increasing discrepancy is noticeable. These errors most likely originate from the combination of a low spatial resolution, the use of second order time stepping schemes and the first-order explicit loose coupling between the level-set and XDG solvers.

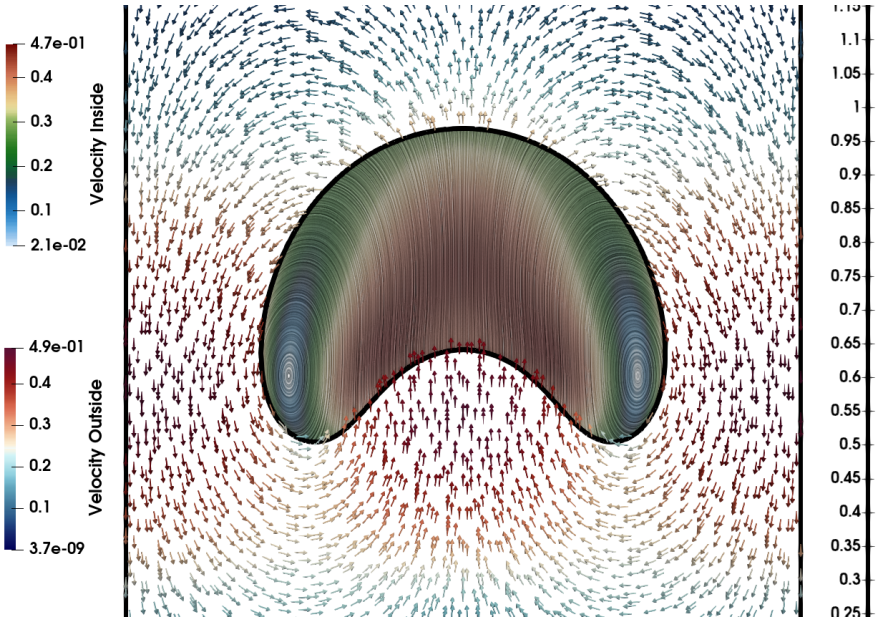


Figure 11.29 / *Rising bubble* – Velocity fields inside and outside the bubble at time $t = 1.95$s.

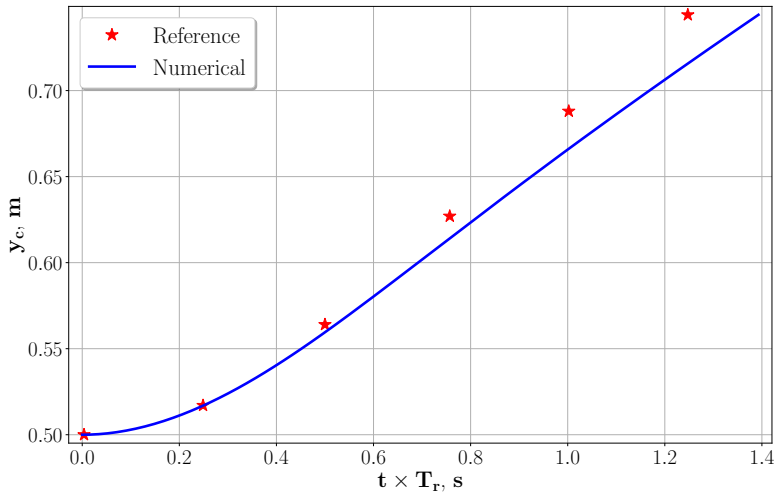


Figure 11.30 / *Rising bubble* – Temporal evolution of the center of mass. Comparison between the reference solution sampled from Hysing et al. (2009) and the numerical solution.

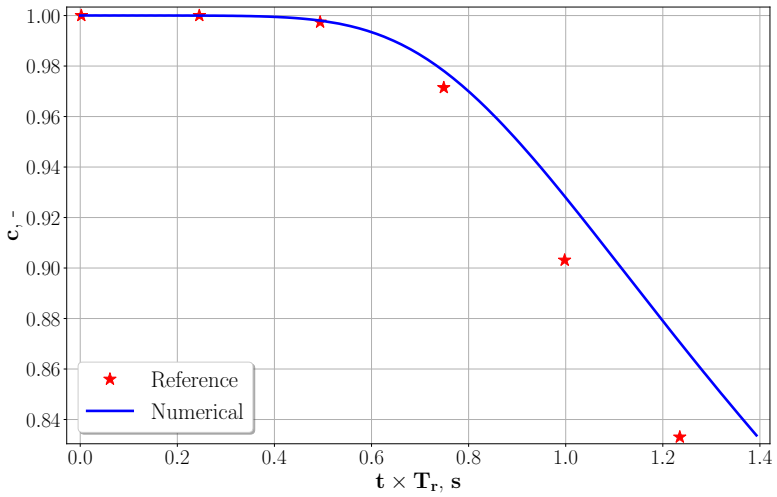


Figure 11.31 / *Rising bubble* – Temporal evolution of the degree of circularity. Comparison between the reference solution sampled from Hysing et al. (2009) and the numerical solution.

11.6.3 Standing capillary waves

Setup and reference solution

The last test case of this section is about the damped oscillations of a small-amplitude capillary wave. A planar interface is disturbed by a sinusoidal perturbation of amplitude a_0 such that the initial level-set field is defined by

$$\varphi_0(x,y) \equiv y = a_0 \cos(kx) \ ,$$

with the wave number $k = 2\pi/\lambda$ and λ is the wavelength of the perturbation. The values selected here are $a_0 = 0.05$ and $\lambda = W$, where $W = 1$ is the width of the rectangular domain defined by $[0.0,\ W]\times[-3/2W,\ 3/2W]$. Gravity is absent here and only the surface tension forces makes the interface oscillate to restore its equilibrium state. Those oscillations are damped by viscous dissipation in the bulk of the phases. In the limit of a small-amplitude initial perturbation, $a_0 \ll W$, and in the special case where both fluids have equal kinematic viscosities, $\nu_1 = \nu_2 = \nu$, an analytical solution for the time-evolution of the interface

amplitude, derived by Andrea Prosperetti (1981), is available:

$$\frac{a(\tau)}{a_0} = \frac{4(1-4\beta)\epsilon^2}{8(1-4\beta)\epsilon^2+1}\operatorname{erfc}\left(\sqrt{\epsilon\,\tau}\right) + \sum_{i=1}^{4}\frac{z_i}{Z_i}\frac{\omega_0^2}{z_i^2-\epsilon\omega_0}\exp\left((z_i^2-\epsilon\omega_0)\frac{\tau}{\omega_0}\right)\operatorname{erfc}\left(z_i\sqrt{\frac{\tau}{\omega_0}}\right),$$

where:

$$\omega_0^2 = \frac{\sigma k^3}{\rho_1+\rho_2},$$
$$\epsilon = \frac{\nu k^2}{\omega_0},$$
$$\tau = \omega_0 t,$$
$$\beta = \frac{\rho_1\rho_2}{(\rho_1+\rho_2)^2},$$
$$Z_i = \prod_{j\in\{1,2,3,4\}i}(z_j - z_i),$$

and with the z_i being the four roots of the algebraic equation

$$z^4 - 4\beta(\epsilon\omega_0)^{1/2}z^3 + 2(1-6\beta)\epsilon\omega_0 z^2 + 4(1-3\beta)(\epsilon\omega_0)^{3/2}z + (1-4\beta)(\epsilon\omega_0)^2 + \omega_0^2 = 0.$$

Results

Figure 11.32 displays the velocity and elevated pressure fields at the nondimensional time $\tau = 6.65$. One can emphasize the sharpness with which the pressure jumps are enforced at the interface. In Figure 11.33, the extracted numerical interface oscillation amplitude is compared to the analytical solution. One can say that a fair agreement can be observed considering the coarse mesh resolution.

Closing remarks Finally, as a last comment concerning all the results presented in this section, it is important to remember that the reference solutions come from incompressible models while the solver presented in this work solves the compressible Navier-Stokes equations. The high values of the stiffened gas EOS parameter p_∞ selected here for the liquid phases should mimic the incompressible behavior, but slight density and pressure variations are still possible. Moreover, the gas phases for the two first test cases is modeled as a perfect gas. Even though compressibility effects are expected to be very small in the flow regimes considered here, a potential influence of the compressible model on the results cannot be totally excluded.

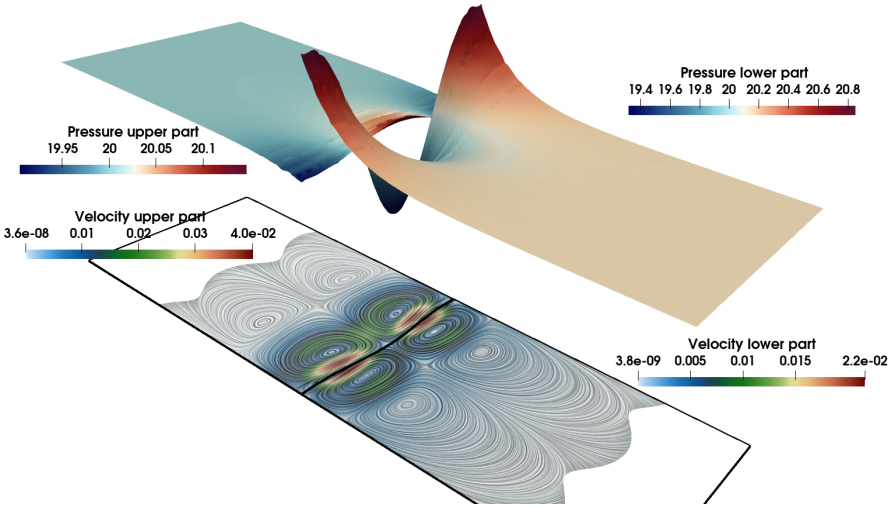


Figure 11.32 / *Standing waves* – Velocity and elevated pressure fields at time $t = 1.4$s. The velocity magnitude in the white regions is below the visualization threshold.

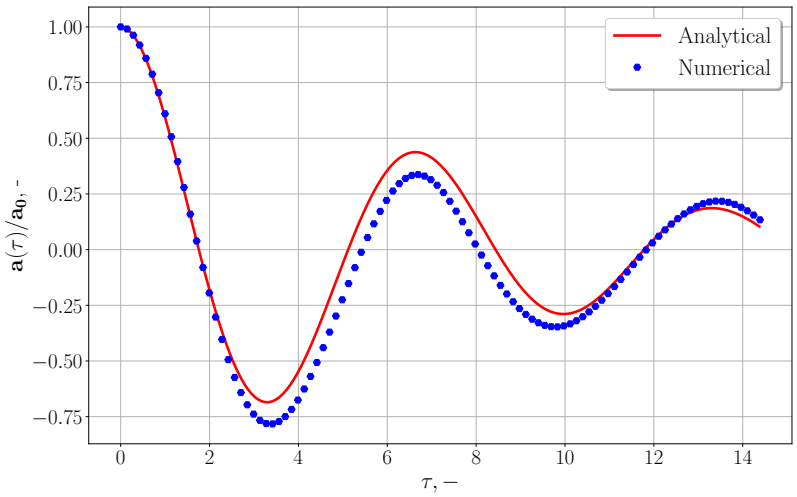


Figure 11.33 / *Standing waves* – Temporal evolution of the interface amplitude. Comparison between analytical and numerical solutions.

11.7 Test cases with phase transformation

To close this verification chapter, we increase the complexity of the equations to be solved by considering mass transfer effects (i.e. evaporation/condensation) in the first two test cases and liquid-solid phase transformation in the two last ones.

11.7.1 Mass transfer at the gas-liquid interface

Sor far we have only considered mass transfer effects in the steady spinning bubble test case in Section 11.3.3. It is time now to do it also on unsteady configurations. We should point out that for the two test cases treated below, we prescribe an expression for the mass flux and do not consider any physically motivated model discussed in Section 3.3.1. The reason is that we focus at this stage on the verification of the numerical tool and considering complex kinetic-based models for the mass flux would have made obtaining a reference solution to compare to particularly difficult, if not impossible. Another worth mentioning point concerns the use of the first splitting strategy of Table 9.1 associated to the multiphase HLLC solver for the iso-0 fluxes. Based on preliminary tests, it appeared that the second splitting strategy seems less robust and more computationally expensive than the first one.

Table 11.15 / Numerical and physical parameters for the two-phase flows test cases with *mass transfer*.

Parameters		1D manuf. sol	Rayleigh-Plesset with evaporation
Spatial discretization	Interp. order	$p = 2\&3$	$p = 3$
	Riemann solver	SLAU2	SLAU2
Temporal discretization	Time step Δt (s)	2e-3 - 1e-4	5e-3
	Schemes (LS/XDG)	BDF4/ BDF3&BF4	BDF2/BDF2
	Variables sets	$p - \mathbf{v} - T$	$p - \mathbf{v} - T$
Numerics - XDG	Riemann iso-0 (Section 9.2.2)	HLLC	HLLC
	Splitting (Table 9.1)	# 1	# 1
	δ_{agg} (Section 8.2.5)	0.2	0.3
	Quadrature	PHG	PHG

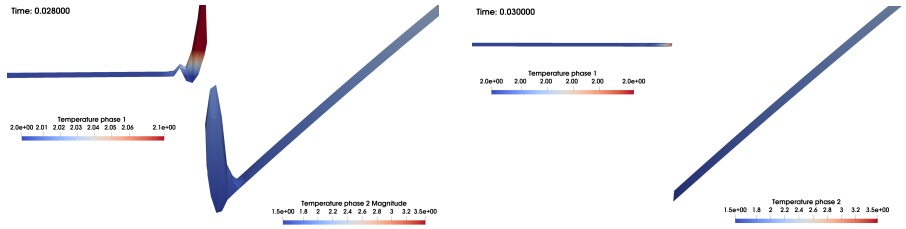
	EOS	PG	Stiffened
	γ_1, γ_2 (-)	1.4, 1.6	7.14, 1.4
Physical	$C_{v,1}$, $C_{v,2}$ (J/(kg K))	2.5, 2.0	0.17, 0.85
parameters	$p_{\infty,1}$, $p_{\infty,2}$ (N/m^2)	/	3.0e4, 0.0
	μ_1, μ_2 (kg/(m s))	1.0, 0.25	0.0, 0.0
	λ_1, λ_2 (W/(m K))	1.0, 0.25	0.0, 0.0
	$\dot{m}_\Gamma$	1.0	$\sin(4.0\pi\, t)$
	$\Delta_\Gamma p$	-0.2	0.0
Jumps values	$\Delta_\Gamma \tau_{nt}$	0.0	/
(Table 3.2)	$\Delta_\Gamma q$	1.0	/
	$\Delta_\Gamma T$	0.5	/
	$\Delta_\Gamma v_t$	0.0	/

One-dimensional unsteady manufactured solution with mass transfer

Setup and reference solution We extend the applicability of the manufactured solutions with prescribed jump conditions used in Section 11.3 to the case of moving interface problems with a non-zero interfacial mass flux. This is detailed in Appendix C. Moreover, we take advantage of this non-trivial situation to confront the splitting and moving interface temporal integration approaches described in Section 10.2.2. Considering a simple one-dimensional domain is enough to fulfill the purpose of evaluating their impact on the temporal accuracy and stability of the whole solver. The values of the jumps prescribed on the moving interface at every instant, including the mass flux, are listed in Table 11.15. We point out that the velocity with which the interface is advected is not imposed by the reference manufactured solution but obtained from the fluid velocities, as explained in Section 6.4.2.

Results The first notable observation to be made is that in this particular configuration, the splitting temporal integration approach leads to simulations that diverge before reaching the final time set to $t = 0.03$. This happens irrespective of the choice of the time-stepping scheme (BDF3 or BDF4), the spatial level of refinement Δx or the magnitude of the time step Δt. Instabilities quickly appear in the interface region as time passes until the magnitude of those is such that the solver diverges. This is illustrated on the temperature field in Figure 11.34 where we can notice the important spurious oscillations appearing before the end of the simulation for the splitting case. On the other hand, the moving interface approach preserves a perfectly clean interface jump. Because of this result, we will focus on a temporal convergence analysis for the moving interface approach only. It must be noted that according to the results presented

in Kummer et al. (2018), the splitting approach should in any case be limited to first order of convergence, except at very small (impractical) spatial resolution where the temporal convergence could be restored.
The temporal convergence study is conducted on pseudo two-dimensional domains (i.e. with a single element along one direction) of length 1 subdivided in 64, 128 and 256 elements. The time steps considered are, in decreasing order, 2e-3, 1e-3, 5e-4, 2.5e-4 and 1e-4. Both the BDF3 and BDF4 time-stepping schemes are used to integrate spatially discretized equations with orders $p = 2$ and $p = 3$. The results are plotted in Figure 11.35. Points that are not drawn on those two graphs correspond to simulations that crashed before reaching the final time. As a general comment, none of the scenario follows perfectly the theoretical order of convergence of the time-stepping schemes. For the BDF3 scheme at $p = 2$, the theoretical order tends to be achieved for the bigger time steps on the coarser meshes before the error reaches a plateau, which reflects maybe the spatial discretization errors since those same plateau can also be observed for BDF4. On the finest mesh for BDF3 with $p = 2$, the theoretical order is only recovered with the smallest time steps. When increasing the spatial interpolation order to $p = 3$, the errors are significantly increased and does not seem to converge at all. This is attributed to the fact that, as discussed at the end of Section 10.2.2, the order of the time stepping scheme should in theory not be chosen independently of the spatial order since the space-time formulation requires integrating in time an expression of degree $2p$. The degeneracy of the errors behavior for $p = 3$ thus can be related to the quadrature errors spoiling the global convergence. This observation according to which an increase in spatial resolution can increase the total error if the temporal resolution is not adjusted properly was also made by Kummer et al. (2018). Turning our attention now to the case of BDF4, we can observe more or less the same behavior of the errors for $p = 2$, even though a suboptimal convergence rate is noticeable in the parts of the curves where the errors decrease. At $p = 3$ for the lager time steps, the errors follow the same trend as for the BDF3 scheme, i.e. a higher spatial interpolation order leads to higher discretization errors and even divergence of the simulation, as the absence of points attests it. For the two smaller time steps on the other hand, the theoretical order of convergence of four (for BDF4) is recovered, even though for the finest mesh, the errors are two order of magnitude higher. This seems to corroborate one of the conclusions made in Kummer et al. (2018) according to which the desired temporal convergence order can be obtained if the spatial resolution is sufficient.
Before closing this section, we want to emphasize that preliminary tests on the exact same configuration but without mass transfer effects across the interface seem to lead to an opposite situation where only the splitting approach would be able to maintain a stable computation. This would require further investigations but we can already indicate at this stage with a reasonable certainty that the temporal integration of the XDG system is a critical point of the solver.



a / Splitting approach at $t = 0.028$.

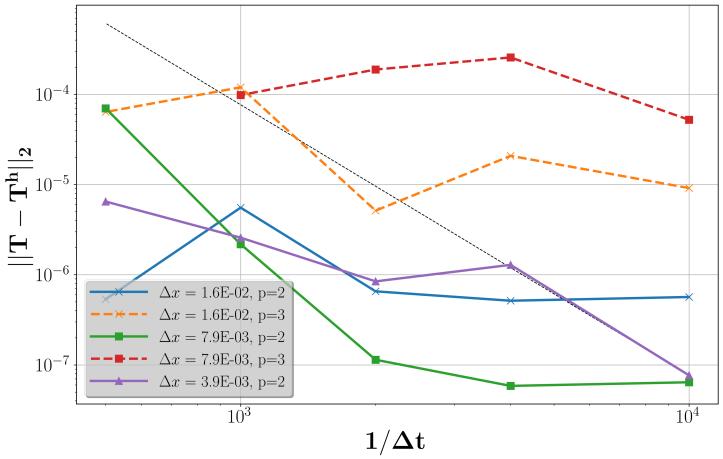
b / Moving interface approach at $t = 0.03$.

Figure 11.34 / *1D manufactured solution test case with mass transfer* – Elevated temperature field on a mesh composed by 128 elements at an interpolation order $p = 2$ with a time step $\Delta t =$ 1e-4.

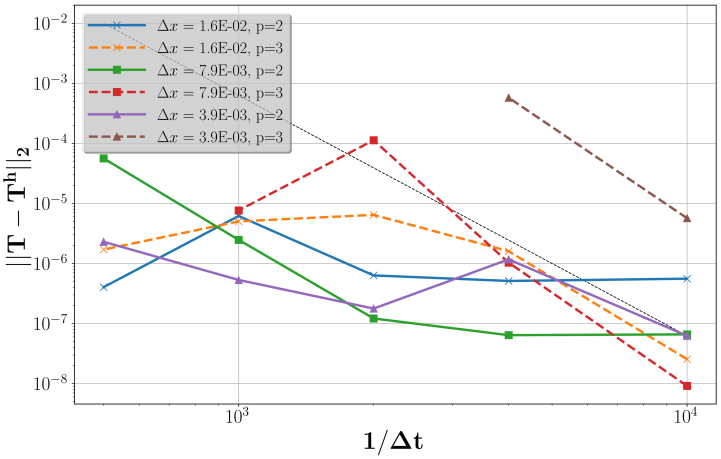
Rayleigh–Plesset collapse bubble with mass transfer

Setup and reference solution We consider once again the Rayleigh-Plesset test case configuration but this time without any pressure fluctuations imposed on the domain boundaries. Instead, a constant pressure of $p = 200$ is set at the liquid external boundary and the initial pressure field is initialized such that it smoothly decreases toward a value of $p = 100$ at the gas-liquid interface. The vapor bubble of initial radius $R_0 = 1$ is initialized with a constant pressure of $p = 100$. We use here an inviscid model, contrary to the first Rayleigh-Plesset test case. This configuration is sometimes called the Rayleigh-Plesset collapse bubble test case. Under the influence of the pressure, the compressible vapor bubble will first start shrinking. Then, when its internal pressure reaches a value that compensates the external liquid pressure, the bubble will expand. The novelty we bring here is that we prescribe a time-dependent mass flux across the interface that will influence the bubble shrinkage and growth. The new semi-analytical solution we derived for this test case is presented in Appendix D and the initial configuration with the mesh are given in Figure 11.36.

Results Figure 11.37 shows the pressure and velocity fields at a given instant. Concerning the pressure, we can note the reversed profile in the liquid phase compared to the initial situation with a pressure reaching its maximum at the interface and which decreases toward the domain extremities. The pressure inside the bubble is also at its peak at the interface across which a jump between the two phases is visible, which is the manifestation of the recoil pressure due to mass transfer. About the velocities, it is interesting to observe that at this selected instant, the jump in the normal velocity components induced by the non-zero mass flux leads to velocity fields in the vapor and liquid with opposite directions. The normalized bubble radius evolution is plotted in Figure 11.38, along with the semi-analytical solution. The two profiles follow a similar trend, especially at the beginning of the shrinkage phase and when the bubble is already well advanced in its expansion phase near the end of the simulation. However, there are significant discrepancies in between the two phases, around



a / BDF_3 time-stepping scheme.



b / BDF_4 time-stepping scheme.

Figure 11.35 / *1D manufactured solution test case with mass transfer* – Discretization errors in L_2 error norms on the temperature field as a function of the inverse of the time step. Dashed lines indicate theoretical order of convergence of the time-stepping scheme.

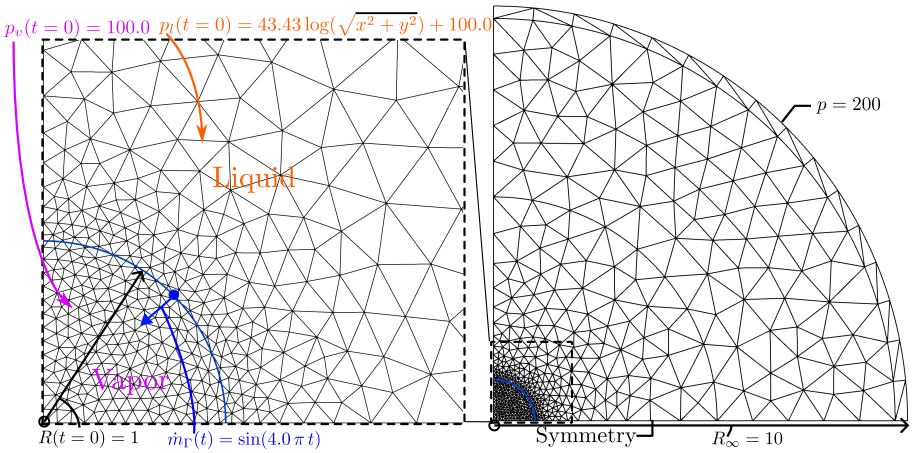


Figure 11.36 / *Rayleigh-Plesset collapse bubble with mass transfer* – Initial configuration.

the maximal bubble contraction. The same justifications as the ones formulated in Section 11.4.2 can be invoked, namely that the inherent compressibility in the liquid phase model leads to a different evolution of the pressure and density inside the liquid phase. In this situation, we think however that the main source of divergence is to be attributed to the assumptions concerning the gas phase on which relies the Rayleigh-Plesset model. Indeed, the bubble is assumed to be a uniform compressible medium, i.e. with pressure, temperature and density that can evolve depending on the bubble volume but that are assumed to be constant inside it. This is actually not the case, as evidenced by the pressure profile in Figure 11.37. Given these differences, we can therefore estimate the comparison between the numerical and theoretical solutions favorable.

11.7.2 Phase transformation at the liquid–solid interface

Up to now, none of the test cases treated before included the volume-averaged equations presented in Section 4.4 to describe both the liquid and solid phases by a single set of equations. The aim of the two following test cases is to verify this aspect of the solver. More precisely, we want to test the phase transformation model included in those equations as well as the use of the generic Mie-Gruneisen EOS derived in Section 2.3.3. As before, the parameters for the two test cases are first presented in Table 11.16.

Table 11.16 / Numerical and physical parameters for the two-phase flows test cases with *melting*.

Parameters		Stefan problem	Solidifying cylinder
Spatial discretization	Interp. order	$p = 3$	$p = 3$
	Riemann solver	AUSM+	SLAU2
Temporal discretization	Time step Δt (s)	2.0e+0	2.5e-1
	Schemes (LS/XDG)	BDF2/BDF2	BDF2/BDF2
Liquid/solid parameters	EOS	Mie-Gruneisen	Mie-Gruneisen
	$\alpha_{p,l}$, $\alpha_{p,s}$ (1/K)	1.17e-11, 1.17e-11	1.17e-4, 1.17e-11
	$\beta_{T,l}$, $\beta_{T,s}$ (1/Pa)	2.38e-11, 2.38e-11	2.38e-11, 2.38e-11
	μ_l, μ_s (kg/(m s))	1.2e-3, 1.2e-3	1.2e-3, 1.2e-3
	λ_l, λ_s (W/(m K))	90.0, 210.0	90.0, 210.0
	$C_{v,l}$, $C_{v,s}$ (J/(kg K))	970.0, 970.0	970.0, 970.0
	$\rho_{0,l}$, $\rho_{0,s}$ (kg/m^3)	2368, 2557	2368, 2557
	$p_{0,l}$, $p_{0,s}$ (N/m^2)	1.0e5, 1.0e5	1.01e5, 1.01e5
	$T_{0,l}$, $T_{0,s}$ (K)	933.0, 933.0	933.0, 933.0
Gas parameters	EOS	/	PG
	γ_g (-)	/	1.4
	$C_{v,g}$ (J/(kg K))	/	927.89
	μ_g (kg/(m s))	/	8.0e-5
	λ_g (W/(m K))	/	0.30
Phase change parameters	T_{melt} (K)	933.0	933.0
	T_{liq}, T_{sol} (K)	941.0, 925.0	956.0, 910.0
	Δh_{lat} (J/kg)	3.97e5	0.0
	K_0 (kg/(m^3 s))	1.0e6	1.0e5

One-dimensional Stefan problem

Setup and reference solution The Stefan problem is a standard test case to verify phase transformation models. As shown in Figure 11.39, a slab of liquid-solid material is considered with an imposed temperature at the top, higher than the melting point of the material under consideration. Here, this temperature condition is imposed on an immersed boundary with the XDG method but only a single phase (the condensed phase) is solved. In order to speed up the computation, we initialized the numerical solution with the semi-analytical

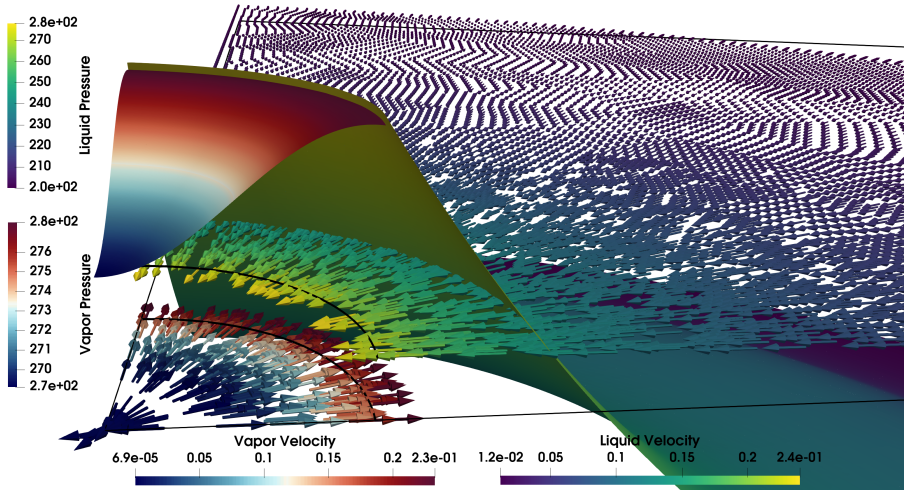


Figure 11.37 / *Rayleigh-Plesset collapse bubble with mass transfer* – Velocity and elevated pressure fields at time $t = 1.375$s.

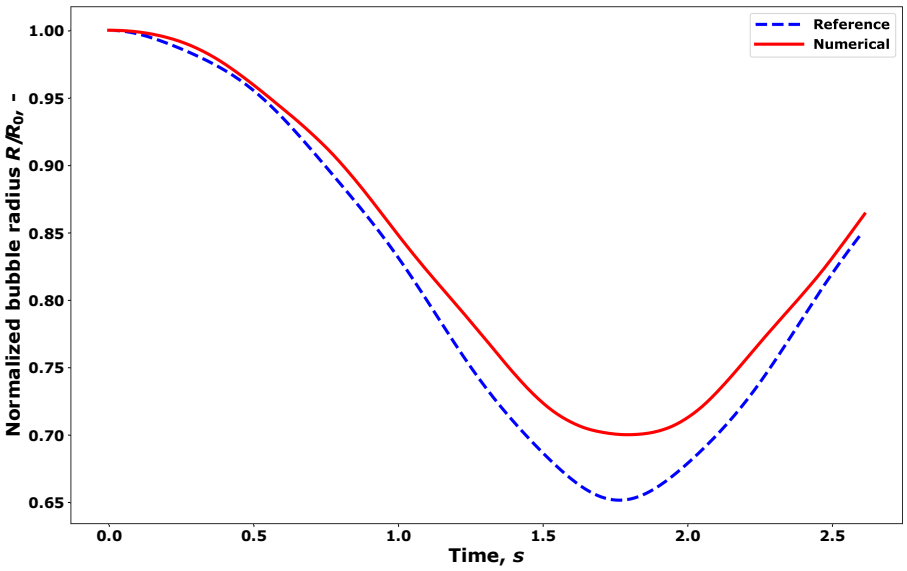


Figure 11.38 / *Rayleigh-Plesset collapse bubble with mass transfer* – Temporal evolution of the normalized vapor bubble radius. Comparison between numerical and semi-analytical solutions .

solution at time $t = 50$s. That is why a part of the slab is already liquid at the start of the simulation. This semi-analytical solution for the temperature profile can be found in Carslaw and Jaeger (1959, Sec. 11.2.III) as

$$T(x,t) = \begin{cases} T_l(x,t) = T_h - \frac{(T_h - T_{\text{melt}})}{\text{erf}(\zeta)} \text{erf}\left(\frac{x}{2\sqrt{\alpha_l t}}\right) & \text{for} \quad x < x^*(t)\,, \\ T_s(x,t) = T_c + \frac{(T_{\text{melt}} - T_c)}{\text{erfc}\left(\zeta\sqrt{\alpha_l/\alpha_s}\right)} \text{erfc}\left(\frac{x}{2\sqrt{\alpha_s t}}\right) & \text{for} \quad x > x^*(t)\,. \end{cases} \tag{11.7}$$

where:

- the ζ parameter is solution of the following transcendental equation

$$\frac{\exp(-\zeta^2)}{\text{erf}(\zeta)} - \frac{k_s}{k_l}\sqrt{\frac{\alpha_l}{\alpha_s}} \frac{(T_{\text{melt}} - T_c)\exp\left(-\zeta^2 \alpha_l/\alpha_s\right)}{(T_h - T_{\text{melt}})\text{erfc}\left(\zeta\sqrt{\alpha_l/\alpha_s}\right)} = \frac{\zeta \Delta h_{\text{lat}} \sqrt{\pi}}{c_{p,l}(T_h - T_{\text{melt}})}\,. \tag{11.8}$$

- the position of the interface between the solid and the liquid as function of time t is given by

$$x^* = 2\zeta\sqrt{\alpha_l t}\,. \tag{11.9}$$

- T_{melt} is the melting temperature, T_c is the initial temperature of the solid, T_h is the temperature of the heated wall and $\alpha = \lambda/(\rho C_p)$ the thermal diffusivity.

Results Figure 11.40 shows the temperature diffusion into the domain and especially the diffuse jump in enthalpy between the liquid and solid phases, corresponding to the absorption of latent heat during the melting process. We can note in Figure 11.40a slight oscillations in the enthalpy field in the transition region. This illustrates the challenge of diffuse interface methods that require a sufficient resolution in the vicinity of the interface to be free of spurious oscillations. In this case, the jumps between the solid and liquid phases are moderate and this constraint is therefore not so restrictive. However, if we wanted to capture the gas-condensed phase interface with similar volume-averaged equations, the requirements in terms of local resolution around the interface would have been much more severe; hence the interest of relying on a sharp interface formulation here. The good agreement between the numerical and semi-analytical temperature profiles in Figure 11.41 certifies the correct implementation of the phase transformation model inside the condensed phase governing equations.

Solidification of an aluminum cylinder immersed in an air flow

Setup This last test case combines all the features of the code verified separately in the previous sections, except mass transfer effects which are not taken into account here. An aluminum cylinder with an initial radius of 0.0525m is placed in a channel at the inlet of which a uniform hot stream of air is injected.

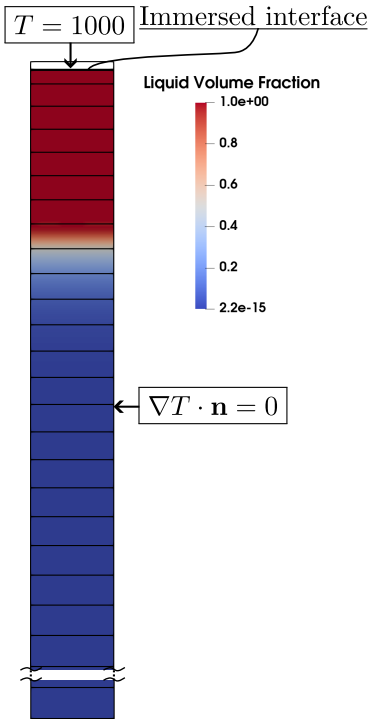
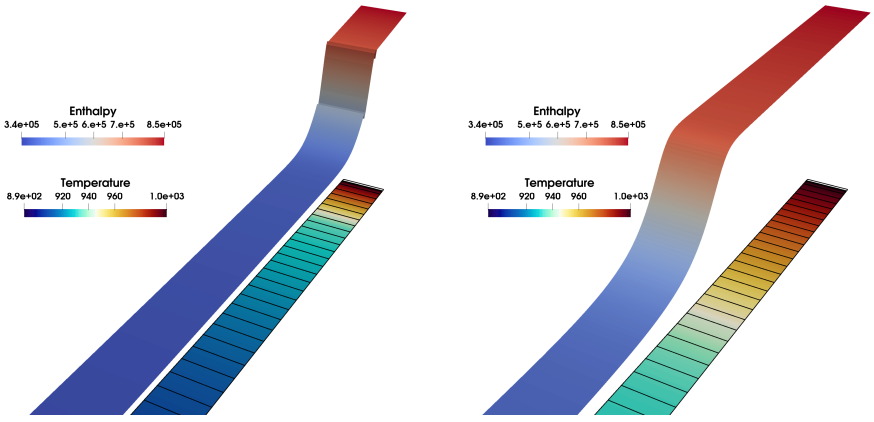


Figure 11.39 / *Stefan melting problem* – Initial configuration.



a / Initial fields, corresponding at $t = 50$s.

b / Fields at $t = 600$s.

Figure 11.40 / *Stefan melting problem* – Temperature and elevated enthalpy fields.

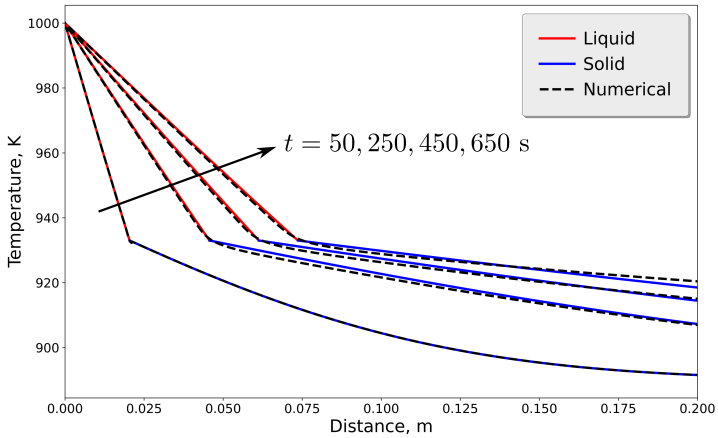
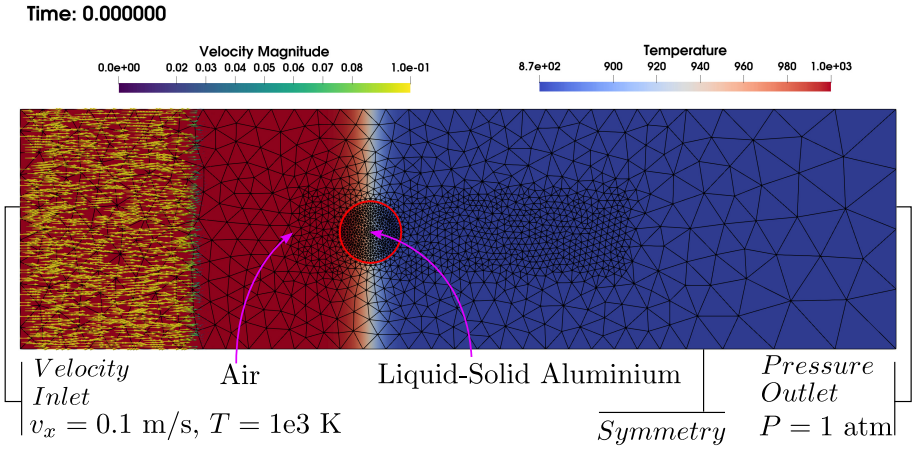


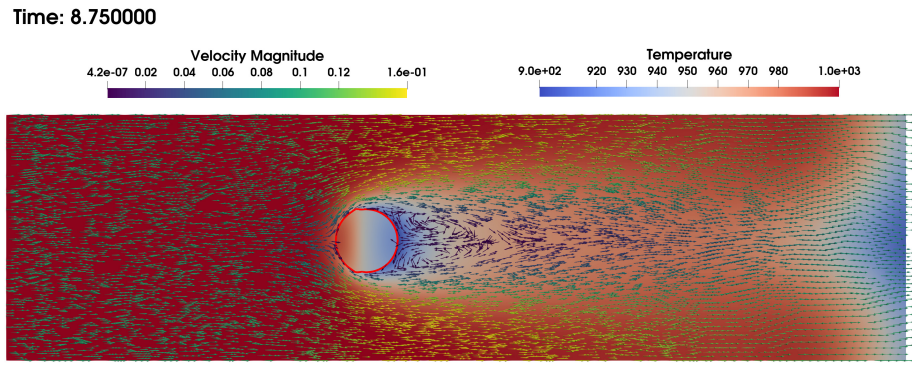
Figure 11.41 / *Stefan melting problem* – Comparison between numerical and semi-analytical temperature profiles at several instants.

The particularity of the setup lies in the fact that the initial temperature field is initialized such that the half of the cylinder facing the air flow is already in a liquid state while the other half is still solid. This way of initializing the temperature, as shown in Figure 11.42a, is obviously not physical but was necessary to obtain a configuration involving the interactions between the three phases in a tractable computational time.

Results Figure 11.42b displays the velocity field in the gas phase with the expected acceleration on top and bottom of the deformed cylinder and the recirculation region in the back. With a Reynolds number of Re = 35, that is the behavior we should expect to see. The purpose of Figure 11.43 is to demonstrate how sharply the density across the gas-condensed phase interface is captured, without the presence of any spurious oscillations despite jumps of the order of 10^4. This reveals the benefit of the sharp interface framework put in place in this work. We can also comment about the density variations inside the cylinder which is due to the distinct thermal dilatation properties between the liquid and the solid aluminum, taken into account by the Mie-Gruneisen EOS used here. Finally, Figure 11.44 shows the entrainment of the molten liquid layer by the incoming gas flow and the resulting interface deformation. At the same time, the liquid-solid region on the right is progressing downstream, indicating the solidification of the cylinder. The extent of the mushy region is increasing due to the diffusion of the temperature field. The development of a crest in the interfacial shape at the cylinder shoulder is attributed to the combined effects of the stressed exerted by the gas on the liquid region and the advancement of the solidification front in the opposite direction.

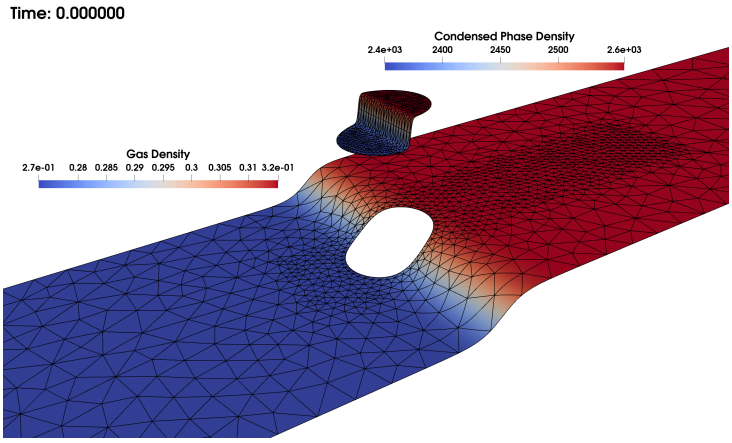


a / Initial configuration with boundary conditions.

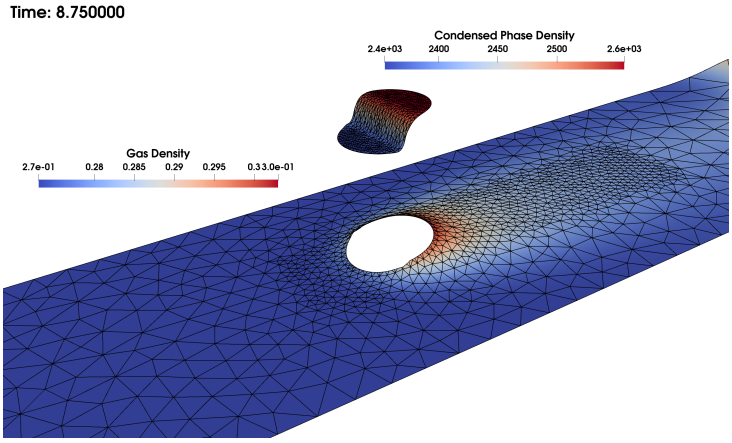


b / Velocity in the gas phase and temperature in both phases at time $t = 8.75$s.

Figure 11.42 / *Solidifying cylinder* – Velocity and temperature fields.

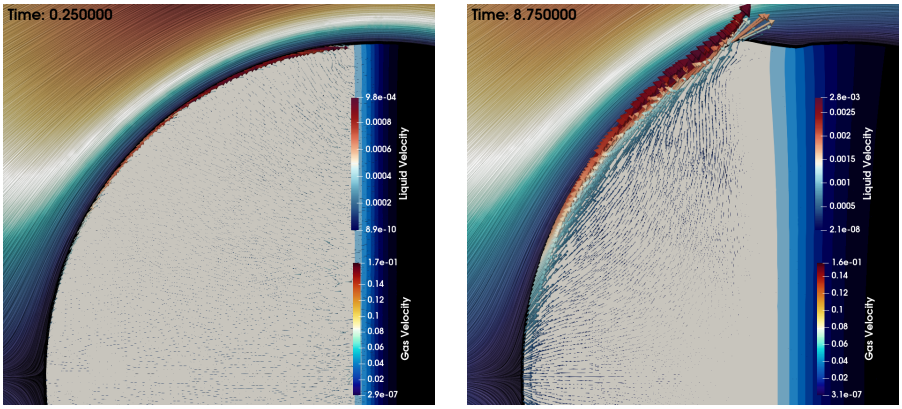


a / Initial densities fields.



b / Densities at $t = 8.75$s.

Figure 11.43 / *Solidifying cylinder* – Elevated density fields in the gas and condensed phases.



a / Velocities and condensed phase volume fraction at $t = 0.25$s.

b / Velocities and condensed phase volume fraction at $t = 8.75$s.

Figure 11.44 / *Solidifying cylinder* – close-up view on the gas-condensed phase interface with the velocity streamlines in the gas and the velocity arrows in the liquid phase. The volume fraction field identifies the purely solid region in black, the purely liquid one in light gray and the mushy region in between.

Rest of this page intentionally left blank.

12

Development of a cold melting ablation experimental setup

Most of the material of this chapter is the result of the master thesis of Andrea Perin (Perin (2023)) for which the present author was one of the advisors.

12.1 Similarity analysis for replication of reentry conditions 438
12.1.1 Similarity concepts for reentry flows 438
12.1.2 Specific dimensional analysis for melting ablation 440
12.2 Setup of a cold hypersonic experiment with melting 442
12.2.1 Our strategy 442
12.2.2 Cold hypersonic VKI H3 facility 443
12.2.3 Sample design and manufacturing 448
12.3 Experimental measurements 451

The first part of this manuscript was devoted to the derivation of the equations governing the physical problem of interest. In the second part, we presented a sophisticated way to extract information from those models by solving them in an approximate way by means of numerical methods. In this last section, we shall cover another complementary approach allowing to gain insight into some physical flow phenomena, namely the experimental approach. Some say that it is another way of "solving" the governing equations. We can also see it the other way around by recalling that the Navier-Stokes equations are nothing more than an approximated model of the true physics occurring during an experiment.

12.1 Similarity analysis for replication of reentry conditions

12.1.1 Similarity concepts for reentry flows

Introduction to dimensional analysis

The Navier-Stokes equations translate the effects of the competition between different effects on the flow field. Those effects have a mutual interaction between them. Hence, if we wanted to reproduce, let's say, the velocity of a real situation in a facility without paying attention to the other dimensional variables, this would most probably lead to a flow field that is not at all representative of the real one. The proper approach to adopt when designing an experiment is to reason in terms of dimensionless quantities and extract a set of independent dimensionless parameters describing completely the behavior of the physical system. This can be e.g. achieved by the well-known Vaschy-Buckingham π theorem which provides a systematic way to determine those independent dimensionless parameters in terms of the dimensional quantities. Because those dimensionless parameters are defined as the ratio between different phenomena (rather than considering them isolated), we can expect to achieve *similarity* between the experimental and real conditions if they are matched between the two situations. In other words, the flow fields will be identical even if the dimensional and the temporal evolution of the phenomena are on different scales. This description relates to the concept of dynamic similarity. For this to apply, it is also necessary that the "boundary conditions" between the two situations are similar. Thus, the scaled model in the facility has to be homothetic to the real one (which is called geometrical similarity) and for special applications, the incoming velocity profiles or turbulence levels have also to be the same (which is referred to as kinematic similarity). It is important to highlight at this point that in general, any experimental facility will not be able to fulfill all the similarity requirements at the same time. Most tests are designed for certain similitude conditions at the expense of other parameters (see Wolowicz et al. (1979)). It is the case for the study of reentry flows, as we will see below.

Common similitude approaches to study reentry flows with the corresponding facilities

A simplified explanation of the general operation of a wind tunnel could be summarized as follows. Whether in a closed-return or open-circuit configuration, flow conditions in the reservoir of the facility are first built up. Then, the flow is accelerated through a nozzle (converging-diverging one in general for high-speed tunnels) and ends up in the test section where the sample is mounted. To study reentry flows, it is simply impossible to simultaneously

achieve in the reservoir the extreme level of pressure and temperature that would be required to achieve the same level of total enthalpy as in flight over a representative period of time. That is the reason why there are different sorts of facilities, each meant to replicate some of the numerous similarity parameters that can be extracted from the variety of flow phenomena happening during a hypersonic reentry phase of an ablating object (see Section 1.2). The categorization of the existing facilities based on the phenomena they aim to reproduce is described below, based on Olivier Chazot (2020). The associated similarity parameters are also provided and commented, assuming that the test gas is the same as in the real flight, i.e. air:

- *Cold inviscid hypersonic flows*: if we are interested to study the pressure distribution over a slender body at small angle of attack without paying attention to viscous effects (neither to any high-temperature effects), the similarity parameter to consider is the product between the freestream Mach number and the slenderness ratio of the body, τ, i.e. $\mathrm{Ma}_\infty\ \tau$.
- *Cold viscous hypersonic flows*: because of the thick boundary layers present in hypersonic flows, taking into account their influence is in general important. The viscous hypersonic similarity requires reproducing the free-stream Reynolds, Re_∞, and Mach number, Ma_∞, as well as the temperature ratio T_w/T_∞, where the subscript w indicates the wall temperature and the ∞ symbol refers to the free-stream. To achieve the required very high Mach number, the facilities belong either to the intermittent or impulse family, characterized by very short test times. Those facilities are used to study the aerodynamics of reentry objects, e.g. the stability of capsules.
- *High-speed high-enthalpy chemically reactive flows*: the main similarity parameter in this case is the product between the freestream density and the characteristic sample dimension, $\rho_\infty\ D_{\mathrm{ref}}$, which comes from the assumption of a frozen flow in reentry conditions leading to a Damköhler number of unity. Some tunnels in this category are able to reach free-stream enthalpies characteristic of high-speed re-entry, which allows to correctly replicate the flow just after the shock. But this is only possible during tests that last fraction of a second. It is usually assumed long enough to let the steady flow establish itself such that aerothermodynamic effects, gas kinetic and radiation can be studied. However, it is way too short compared to the thermal inertia of the material surface.
- *Low-speed high-enthalpy chemically reactive boundary layer flows with gas-surface interactions*: contrary to the previous category, plasma wind tunnels, such as inductively coupled plasma facilities or arc-jets, are not able to reproduce the freestream enthalpy. They focus instead on the boundary layer region and relies on the so-called Local Heat Transfer Simulation (LHTS) methodology to duplicate the heat flux and the chemistry taking place between the gas and the sample surface. According to LHTS, if the total enthalpy (more or less constant along the stagnation line), the stagnation pressure and the velocity gradient at the stagnation point are

matched with the flight, the stagnation point region in the facility will correctly reproduce the heat flux and the gas-surface interactions. This approach is the only viable one to study in details ablation processes.

12.1.2 Specific dimensional analysis for melting ablation

Simplified methodology

From the overview of similitude approaches above, it seems that the LHTS methodology and the use of Plasma facilities would be the best choice to thoroughly analyze the ablation by melting process. This was indeed the approach followed by Andrea Fagnani at VKI who just completed his PhD thesis in the same research context as the present work (Fagnani (2023)). That is however not the approach investigated in this work for the following three reasons:

- *Simplified physics to ease code validation*: similarly to the verification process, the validation of a numerical tool is made easier and more efficient by considering an incremental process. In this case, this translates into the design of an experiment addressing some specific ablation phenomena, in accordance with the models already implemented in our numerical tool.
- *Isolation of phenomena to improve their comprehension*: the study of melting ablation is particularly difficult because of the interplay between several phenomena which, taken separately, are already very complex. An experiment that would allow isolating certain phenomena could make it easier to identify the dominant parameters influencing the ablation. This could in turn provide some leads to improve the design for demise of some spacecraft components.
- *Easier, faster and cheaper experimental setup*: the last motivation is to seek for an experimental setup which would not be as complicated as the one required in a plasma facility.

Following the reasoning above, our focus lies in the replication of the melting and solidification process accompanied by the molten layer dynamics and shear removal, isolated from the other phenomena identified before as playing a role in the global melting ablation process. This simplified methodology is inspired by several works from the sixties and seventies such as C.-J. Chen (1967), Edling and Ostrach (1966), Cochran and Ostrach (1968) and Kuzyk and C. J. Chen (1972). The objective pursued by those research is to reproduce the wave patterns observed on recovered meteorite samples which undergo a very similar melting ablation process as space debris. Crunching the variety of involved phenomena into just a few ones might lead to potential deficiencies of the proposed strategy, which can be summarized in two main points:

- *Proper duplication of targeted phenomena not possible*: even though previous studies seem to indicate that the similarity between reentry and on-ground

flow features is possible under certain assumptions, this is very much dependent on the operating conditions of the available wind tunnels, as well as the selected reentry conditions.

- *Decoupling of ablation phenomena not relevant*: the main concern is about the validity of restricting the experiment to some phenomena and extracting meaningful measurements. For instance, it is expected that the evaporation of the liquid layer or the formation of an oxide layer on its surface could greatly impact its evolution.

The extent to which these limitations can manifest themselves has not been analyzed as part of this thesis. This would require establishing matching criteria between plasma and cold hypersonic conditions and to identify and measure relevant quantities in both cases.

Determination of appropriate similarity parameters

The LHTS methodology, although perfectly suited to replicate gas-surface interactions between a reactive gas and an ablating solid surface, does not consider the apparition of a liquid molten layer in-between the two. One of the main objectives of this thesis is precisely to study the influence of this liquid layer on the global ablation process. It appears therefore that a new specific dimensional analysis, neglecting all the complex high-temperature effects included in LHTS but considering the presence of the liquid layer, needs to be developed. This analysis shall provide a set of dimensionless parameters representing the interactions between the three phases, i.e. gas, liquid and solid, such that a similarity in the molten layer dynamics and melting is achievable between the reentry and the experimental conditions. The procedure developed in this work is once again inspired by papers about meteorites ablation, in particular Edling and Ostrach (1966, App.), Cochran and Ostrach (1968, App. B) and Ching-Jen Chen and Ostrach (1971). Other related theoretical studies include Feldman (1959), Lees (1959) and Ostrach et al. (1962). The whole procedure is explained in Appendix E and we only give here values of some characteristic numbers estimated from the expressions of the reference quantities in Equation E.19:

$$\begin{aligned} \mathrm{Re}_g &= 1.9\mathrm{e}5\,, \\ \mathrm{Pr}_g &= 6.9\mathrm{e}\text{-}1\,, \\ \mathrm{Ec}_g &= 1.4\mathrm{e}\text{-}1\,, \\ \mathrm{Ma}_g &= 4.1\mathrm{e}\text{-}1\,, \\ \mathrm{Re}_l &= 4.8\mathrm{e}\text{-}1\,, \\ \mathrm{Pr}_l &= 1.1\mathrm{e}\text{-}2\,, \\ \mathrm{Ec}_l &= 5.7\mathrm{e}\text{-}4\,, \\ \mathrm{Ma}_l &= 5.2\mathrm{e}\text{-}4\,, \end{aligned}$$

where Re is the Reynolds numbers, Pr the Prandtl numbers, Ec the Eckert numbers and Ma the Mach numbers, of the gas, $\bullet_g$, and liquid, $\bullet_l$, phases, respectively. We should point out that the liquid Reynolds number includes in its definition the characteristic depth of the liquid boundary layer. Those values correspond to the H3 operational conditions and the size of the tested sample, discussed below. As explained in Appendix E, in order to obtain them, an estimation of the gas edge boundary layer conditions is required. To this end, we relied on the `Cabaret` code developed by A. I. d. Val et al. (2017) to first compute the flow expansion through the nozzle from the given reservoir conditions. The, from the obtained freestream conditions in the test section, the code is used to solve the normal shock relations and provide the post-shock as well as stagnation conditions. We picked the value of the post-shock velocity as an estimation of the gas edge boundary layer velocity and the pressure and temperature at the stagnation point for the gas edge boundary layer pressure and temperature. An alternative and more accurate option would have been for instance to use the Stagnation Line code developed by Munafò (2014).
Finally, let us remark that the exact same procedure should also be applied to the flight conditions in order to obtain the corresponding characteristic numbers and compare them to the experimental ones. We did not push the investigation to that point, for the reasons elucidated in Section 12.2.1.

12.2 Setup of a cold hypersonic experiment with melting

12.2.1 Our strategy

The dimensional analysis presented above should be performed both for the reentry and the experimental conditions in order to extract the dimensionless numbers for each situation and evaluate their matching. At this stage, we can remark that a lot of degrees freedom remain given the wide range of conditions that can be found along a typical reentry trajectory on one side and on the other side the range of operating conditions that can be achieved in a given facility. In addition to this, the choice of material, geometry and size of the object also comes into play. Usually in practice, a single point along the trajectory, corresponding for instance to the peak heating, is selected. An alternative (more relevant) approach in the LHTS framework is to consider an integrated heat flux over a portion of the trajectory and fix the testing time to reach that value based on the achievable heat flux in the facility. In our specific case, it is difficult to identify such integrated quantity and the choice of a specific reentry point seems the most straightforward. The facility operating conditions and the sample size should then be adapted accordingly to meet the required similarity requirements. This standard approach is however not the one adopted here. This has to do with the particularly important number of dimensionless

parameters to be matched (more than with the LHTS methodology). Picking a reentry point and a given object size prior to the definition of the facility operating conditions is very likely to lead to constraints that cannot be met on the experimental side. Because the space of free parameters to be adjusted is much more constrained on the experimental side, we decided to follow a reversed approach by (i) first fixing all the experimental parameters, (ii) and in a second step trying to determine consistent corresponding reentry parameters such that the similarity is achieved. Thus, the choices of the sample design and operating conditions described below are guided by the phenomena we want to reproduce and the limits of the facility rather than the real reentry conditions.

12.2.2 Cold hypersonic VKI H3 facility

Choice of the facility

For this experimental campaign, we selected the blowdown hypersonic wind tunnel H3 installed at the von Karman Institute (Simeonides (1990, Sec. 2)). The reasons for this choice, in addition of the question of availability of facilities, are threefold:

- *Cold environment*: as already explained before, our goal is to isolate some melting ablation phenomena from the others. In particular, we want to avoid high-temperature effects involving complex chemical reactions and radiation, as we can find in plasma facilities. This facility which provides a free-stream static temperature of about 67K with a Mach number of 5 indeed leads to a temperature of the jet of air impacting the sample way below the onset of high-temperature effects.
- *Sufficient test time*: contrary to another hypersonic facility present at VKI, namely the Longshot, the H3 wind tunnel is quite straightforward to operate and allows test durations of the order of 20 to 30 seconds. This was an obvious requirement to be able to overcome the thermal inertia of the sample and observe its melting.
- *Hypersonic conditions*: it must be pointed out here that supersonic and *a fortiori* hypersonic conditions are not strictly needed to achieve the similarity we presented before. As a matter of fact, all the work cited before dealing with the experimental replication of hypersonic melting ablation consider subsonic facilities. We chose this hypersonic facility because (i) we had the chance to have it at our disposal and (ii) we considered the future validation of the numerical tool which should be extended to capture strong shock and thought that recording the evolution of the shock shape as the sample surface recedes could constitute an interesting validation basis.

We used the newly installed contoured axisymmetric Mach 5 nozzle designed by Cakir et al. (2022) utilizing the method of characteristics based on the in-house code HYPNOZE developed by Grossir and Chazot (2022). Cakir

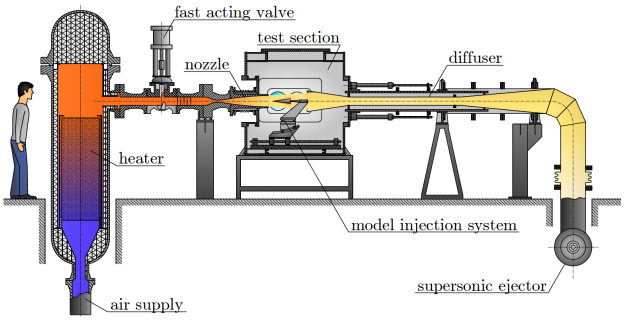


Figure 12.1 / Schematic representation of H3 hypersonic wind tunnel facility, taken from Cakir et al. (2022).

et al. (2022) describes that dried air is supplied from a pebble-bed heater at stagnation pressures from 7 to 35bar at a maximum stagnation temperature of 550K. Once the desired stagnation pressure and temperature conditions are established, a fast activating valve is opened and the air is expanded through the contoured nozzle into a vacuumed test chamber. A supersonic diffuser (located immediately downstream of the test chamber) is used to decelerate the flow and recover the kinetic energy in the form of pressure. A supersonic ejector is then used to discharge the nozzle mass flow rate into the atmosphere. A schematic of the wind tunnel is shown in Figure 12.1. Agostinelli et al. (2020) further explains that the test section of the H3 facility is equipped with a three-degree-of-freedom traversing mechanism allowing axial displacement of models and the variation of incidence and yaw angles. This mechanism is combined with a pneumatic system for the rapid injection of models into and out of the flow field: the usual procedure is to inject the model in the test chamber once the hypersonic flow has been established.

Operating conditions and test matrix

Preliminary tests As it is explained below, one of the material selected for the samples is the paraffin wax and the molds used to build those samples are 3D printed in polylactic acid (PLA) plastic. We started our experimental campaign by a series of three preliminary runs, with the corresponding operational conditions listed in Table 12.1. The first one consisted in blowing on a sample made of PLA with a hemispherical nose (see Section 12.2.3 for more details about the samples geometry). The goal was to verify that the geometry and the dimensions of that sample, which are the same as the ones used for the other samples tested afterward, would not lead to blockage effects preventing the flow establishment into the test section and compromising the wind tunnel startup. This was indeed not the case. With this material, we could not observe any melting taking place, which is not surprising given the relatively high melting point of PLA generally situated between 430 and 470 K. We then conducted two runs on wax samples, with a hemispherical and a rounded conical nose,

Table 12.1 / Test matrix for the preliminary melting ablation experiments.

Test #	Material	Config.	Run time (s)	p_0 (bar)	T_0 (K)
PT1	PLA	Hemis. nose	~14.0	10	300
PT2	Wax	Hemis. nose	~14.0	10	300
PT3	Wax	Conical nose	~14.0	10	300

respectively. The conclusion from those two tests was that the testing time or the reservoir temperature were not high enough to observe significant melting.

Final test matrix Based on the preliminary tests, we decided for the final runs to increase the reservoir temperature rather than making the runs longer. The reason is that we had a limited amount of pressurized air at our disposal, which prevented us from running successively longer runs. This is also the reason why we did not consider higher reservoir pressures which would have increased the mass flux. In addition to wax, we also tested samples made of gallium (Ga). Each material was tested in two configurations, with a hemispherical and a rounded conical nose. This is discussed in more details in Section 12.2.3. A last run with a gallium sample featuring a rounded conical nose and slightly inclined edges in the rear part was also considered, for a total of five runs. To have an idea about the required reservoir temperature, we performed a rough simulation with the `Argo` code to estimate the flow temperature adjacent to the sample. Results of this steady inviscid two-dimensional (not axisymmetric) simulation on the last sample configuration are shown in Figure 12.3, with the corresponding unstructured mesh in Figure 12.2. They were obtained with an interpolation order of $p = 0$ based on the reservoir conditions corresponding to the final test matrix presented in Table 12.2. More precisely, the computation of the freestream quantities at the exit of the nozzle to be provided as the inlet conditions to the CFD computations were obtained by using the `Cabaret` code. This code can solve iteratively the conservation of entropy and total enthalpy along the stagnation line during the flow expansion through the nozzle.

Visualization techniques

As the nature of this experimental campaign is exploratory, we have not dedicated time to the instrumentation of the samples (e.g. with internal thermocouples to measure the temperature) nor to the setup of advanced measurements techniques. We simply relied on the already calibrated shadowgraph system installed in the facility. Shadowgraph is an optical method that reveals non-uniformities in transparent media like air based on the refraction of uniform light beams (in our case due mainly to density variations across the shock wave) sent across the test section through the windows of the wind tunnel. Those rays are collected on a screen and the interferences between them result in darker

Table 12.2 / Test matrix for the final melting ablation experiments.

Test #	Material	Config.	Run time (s)	p_0 (bar)	T_0 (K)
WC1	Wax	Conical nose	~20.0	9	480
GC2	gallium	Conical nose	~20.0	9	480
GH3	gallium	Hemis. nose	~20.0	9	480
WH4	Wax	Hemis. nose	~20.0	9	480
GCA5	gallium	Conical nose ∠	~20.0	9	480

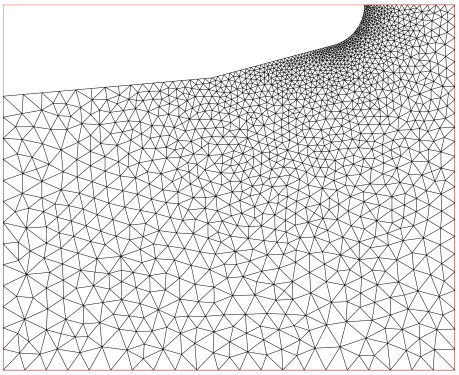


Figure 12.2 / Unstructured mesh of the GCA5 sample geometry with symmetry boundary conditions on the top.

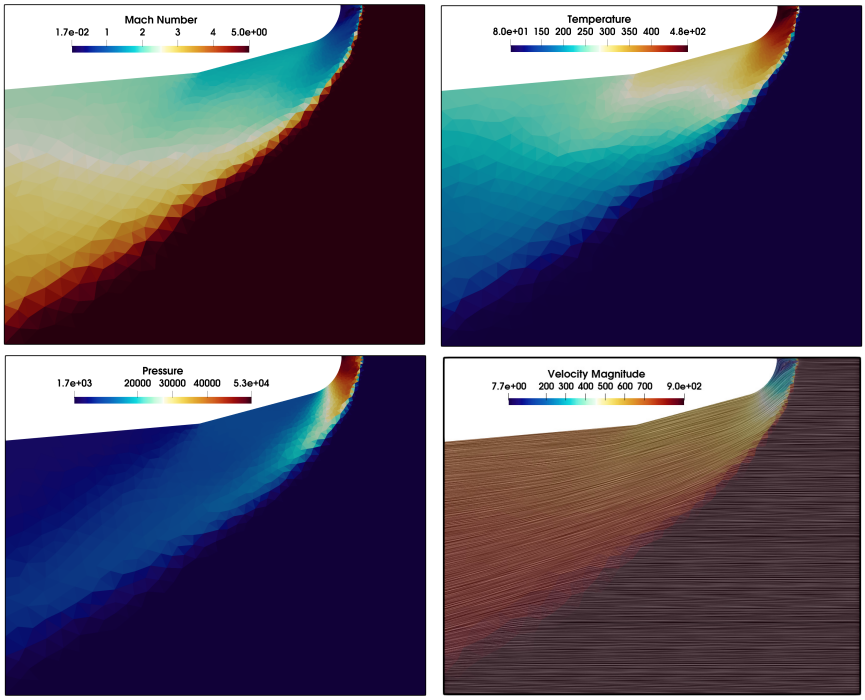


Figure 12.3 / Two-dimensional steady state results of the flow around the GCA5 sample geometry with inlet conditions corresponding to the reservoir conditions of the H3 facility listed in Table 12.2. Results obtained by a $p = 0$ DG discretization of the Euler equations with constant properties.

and brighter regions on the screen which are recorded by a high-speed camera. In this work, we used an acquisition frequency of 100 Hz. It can be remarked that shadowgraph is not the most accurate technique to extract surface recession profiles of an ablating surface as the contrast between the different regions is weak. Direct side-on measurements using a high-speed camera could facilitate the extraction of the time evolving surface profile. Other more advanced techniques such as photogrammetry used by Turchi et al. (2019) could also be considered.

12.2.3 Sample design and manufacturing

Materials choice

The choice of the materials to be tested has been driven by three main criteria:

- *Low melting temperature*: with the temperature expected in the vicinity of the sample, we could estimate the upper limit on the melting temperature that should not be exceeded by the material.
- *Negligible evaporation*: in the conditions of pressure and temperature predicted after the shock, we wanted a material for which evaporation of the liquid layer and sublimation of the solid is limited. Selecting camphor or napthalene as it has already done before in H3 by Turchi et al. (2019) would not have been a good option because those materials are expected to undergo non-negligible sublimation.
- *Negligible reactions with air*: finally, a material not subject to reactions with ambient air was desirable to avoid the formation of any oxide layer during the manufacturing process.

Based on those criteria, our choice fell on pure gallium and paraffin wax (the same type as the one used for candles). The choice of gallium was inspired by the work of Denis et al. (2022) who also studied thermal ablation by melting in a cold hypersonic environment. The advantage of the H3 facility compared to the shock tunnel considered in that work is that the testing time is sufficient to observe a complete deformation of the sample undergoing melting, as shown in Section 12.3. We also considered water following the work of Kubota (1960) who studied ablation at Ma = 5.8 on a water sample. We did not retain water at the end because of the issues we faced during the manufacturing of the samples. Some important properties of gallium and paraffin wax are given in Table 12.3. We can remark that despite its very low melting point, gallium presents properties (density, thermal conductivity, liquid viscosity, etc.) which are very similar to other high-melting point metals such as titanium that are critical in some spacecraft components. Its usage in the present context is thus particularly well suited.

Table 12.3 / Some properties of the two materials used for the melting ablation experiments.

Material	ρ (kg/m^3)	C_p (J/(kg K))	λ (W/(m K))	μ (kg/(m s))	T_{melt} (K)	Δh_{melt} (J/kg)
Solid gallium[a]	5.9e3	3.7e2	3.5e1	-	3.03e2	8.0e4
Liquid gallium[b]	6.1e3	3.6e2	2.8e1	2.0e-3	-	-
Solid wax[c]	9.1e2	2.0e3	2.4e-1	-	3.33e2	1.9e5
Liquid wax[d]	7.9e2	2.2e3	2.2e-1	5.5e-3	-	-

[a]Properties from Ayrinhac (2000), at $T = 293.15$K
[b]Properties from Ayrinhac (2000), at $T = 303.15$K
[c]Properties from Khot et al. (2011), at $T = 288.15$K
[d]Properties from Khot et al. (2011), at $T = 343.15$K

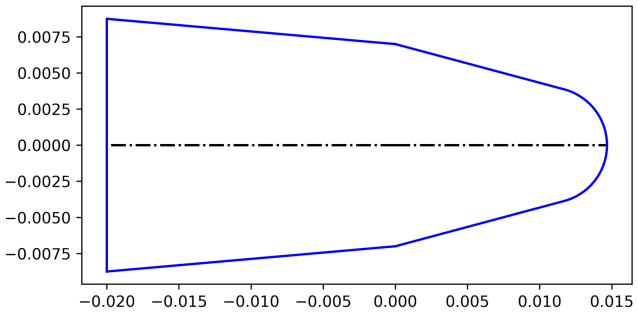
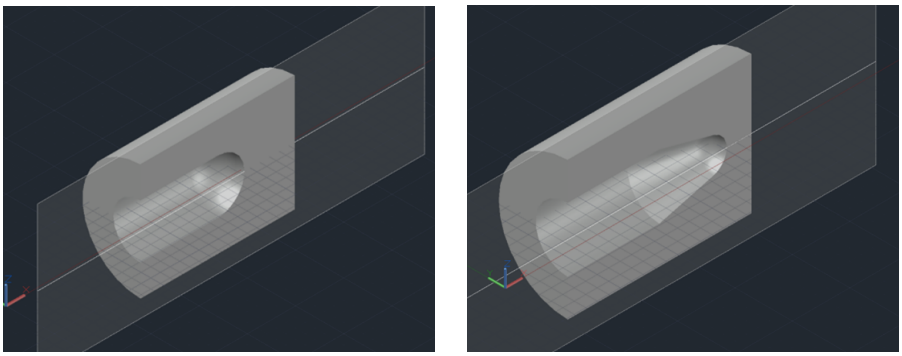


Figure 12.4 / Two-dimensional cut along the axial direction of the rounded cone-shaped sample geometry .

Geometry definition

Taking inspiration from Denis (2022, Sec. 4.1.3), we considered two main geometries. The first one is a hemisphere extended by a cylinder while the second one consists in a cone-shaped geometry with a rounded nose, also with a cylindrical rear part. Their dimensions are shown in Figure 12.7 (which in fact presents the dimensions of the molds from which the samples are built). For the last test, we considered a slightly different version of the cone-shaped geometry where the cylindrical rear part has inclined boundaries, as sketched in Figure 12.4. The rounded front part of the geometry was adopted for both configurations to have a detached shock, which is the most likely situation for a re-entering space debris. The extension with a cylindrical rear part is there with the objective of favoring the flowing of a liquid molten layer over the object and its visualization. As far as the dimensions of the samples are concerned, those were dictated by the diameter over which the jet exiting the H3 nozzle is expected to be uniform, as well as blocking effects.



a / Hemispherical nose.

b / Rounded conical nose.

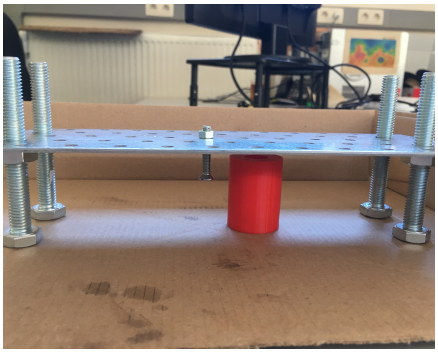
Figure 12.5 / Three-dimensional cuts along the axial direction on the two main geometries of molds used to manufacture the samples tested in the H3 facility.

Sample casting and holding mechanisms

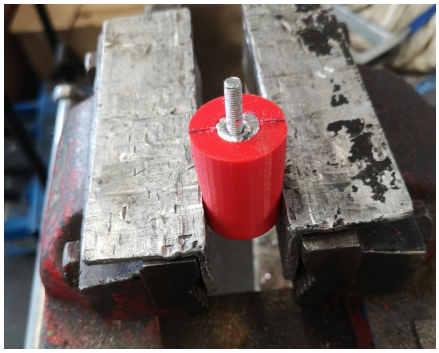
In order to manufacture the samples with the desired geometries, we prioritized the ease and speed of prototyping over the surface quality and respect of tolerances. Given the low melting points of paraffin wax and gallium, we opted for a casting process.

Molds manufacturing A 3D printer was used to make the molds in a very efficient way from CAD models that were created beforehand and shown in Figure 12.5. The molds were printed in polylactic acid (PLA) plastic, with a 40% filling density.

Casting process Many trials have been conducted to determine the best approach to cast the samples with different materials. A total of 17 trials were made, starting with wax and water ice. Gallium was only considered later on because of its limited quantity and its price which made its wasting highly undesirable. As shown in Figure 12.6a, a screw is maintained vertically on a leveled plate with a bolt allowing to adjust the depth with which it is dipped into the mold such that it corresponds to the sketch in Figure 12.7. As can be seen on those pictures, the screw is immersed into the mold with its head facing downwards to improve the coupling strength once the molten material solidifies around the screw. Note that the screw thread also plays this important role. We found out that in the case of paraffin wax, applying beforehand a thin layer of oil inside the mold improved its extraction after solidification. For water ice, this was not enough and all samples broke apart during extraction. Finally, for the gallium samples, each time we had to sacrifice the mold by cutting it in two to be able to extract the samples, as illustrated in Figure 12.6b. Possibilities to remediate those issues could be to polish the internal surface of the molds to remove all the asperities or to design the molds such that they can easily be



a / Screw and mold filled with liquid material just before dipping the screw into the mold and putting everything in the fridge for the solidification.



b / Extraction of the solidified gallium sample from the mold.

Figure 12.6 / Pre- and post-manipulations in the casting process of the samples to be tested in the H3 facility.

opened along the axial direction via a clamping mechanism similar to the one used in Denis et al. (2022).

Holding mechanism Concerning the mounting of the samples inside the H3 test section, we had first to drill a new thread in the existing bar making the connection between the fixed mechanism installed in H3 and the samples. Then, we could simply use the apparent part of the screw whose other part is merged with the sample to operate the connection between the bar and the samples, with an intermediate bolt to adjust the tightening. This can be seen in Figure 12.8, showing the mounting of two samples before and after the run. On those pictures, the pneumatic injection mechanism and the exit of the nozzle on the right can be identified. On the left, we can also see the inlet of the collecting diffuser which ensures that the hypersonic jet as well as the solidified droplets sheared away from the liquid molten layer do not impact the facility walls and windows.

12.3 Experimental measurements

On Figure 12.8b and Figure 12.8d, we can already see the state of a wax and a gallium sample at the end of the test and how they have been ablated. Particularly worth mentioning on those pictures is the solidified ablated material on the holding mechanism that has been deposited there during the run under the influence of the shear forces. For wax, it comes in the form of a film surrounding the connecting rod. For gallium, we can rather observe the signature of the splashing of some droplets on the holding mechanism. A closer

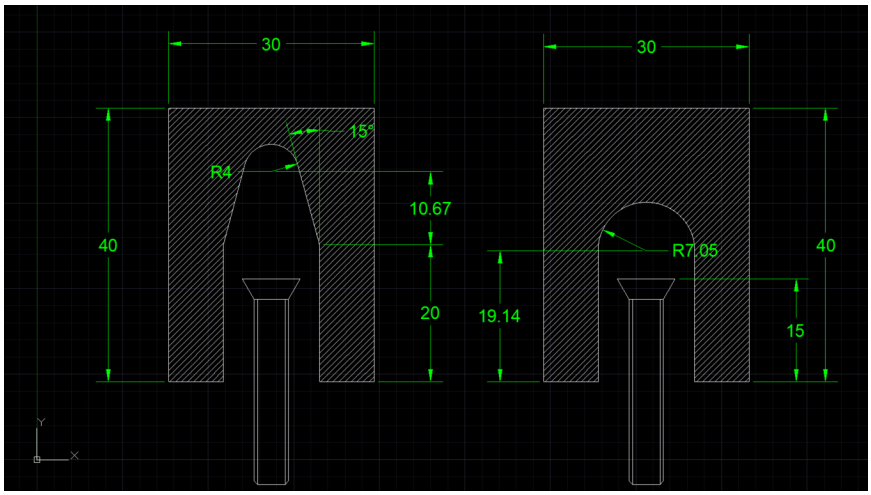


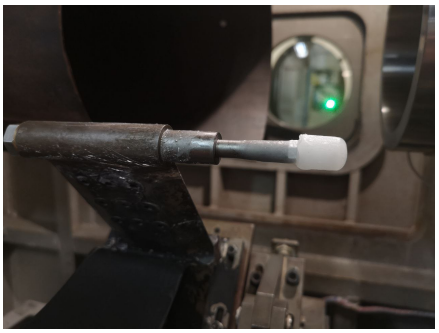
Figure 12.7 / Two-dimensional cuts with dimensions along the axial direction on the two main geometries of molds used to manufacture the samples tested in the H3 facility.

view and comparison between all the samples, before and after the tests, is presented in Figure 12.9. It is apparent that the ablated surface for the gallium samples presents much more irregularities than the smooth one for the two wax samples. This can be explained first by the fact that gallium samples undergo much more melting than wax samples over a comparable testing period because of the differences in melting temperatures and latent heat of fusion. We can in fact visualize the screw head that becomes apparent for the two first gallium samples. Another possible cause is that the quite "amateur" casting process we considered here lead to a very non-uniform solidification of the gallium samples, with the possible presence of trapped air bubbles. This non-uniformly is already visible on the surface of the samples, although they have been previously polished. Figure 12.10, Figure 12.11 and Figure 12.12 show some snapshots of the shadowgraph images we recorded during the tests with the high-speed camera. It is instructive to see that both for wax and gallium, with the different geometries, the liquid layer is clearly sheared away toward the back of the sample under the influence of the strong shear. The multiphase interface morphology transitions from a separated configuration in the front part to a mixed configuration with the formation of elongated liquid filaments and the release of droplets in the rear part. However, we do not observe in those conditions the formation of a highly dispersed droplet clouds, which would make our modelling approach not applicable. On the contrary, on the basis of those results, we can conclude that the local instant sharp interface treatment of the gas-condensed phase interface seems appropriate, provided that mesh adaptation capabilities to resolve the filaments and droplets are combined. In Figure 12.10 and Figure 12.12, the contrast has been adjusted compared to Figure 12.11 such that the shock is more apparent. It is interesting to observe, as

expected, the evolution of the shock profile as the sample is ablated. We can also point out the limitations of this visualization technique to distinguish the sample material from the holding mechanism. This brings some issues if we wanted to extract an accurate ablated profile from the recorded images. Finally, let us remark that gravity forces noticeably influence the development of the molten layer over the sample, as the pronounced non-symmetrical behavior observed in some figures attest.



a / Hemispherical wax sample (WH4) before the run.



b / Hemispherical wax sample (WH4) after the run.



c / Rounded conical cone gallium sample with inclined boundaries (GCA5) before the run.



d / Rounded conical cone gallium sample with inclined boundaries (GCA5) after the run.

Figure 12.8 / Pictures of two samples mounted in the H3 wind tunnel, before and after the test. The injection mechanism to which the samples are fixed, the start of the collecting diffuser on the top left and the end of the H3 nozzle on the top right are visible.

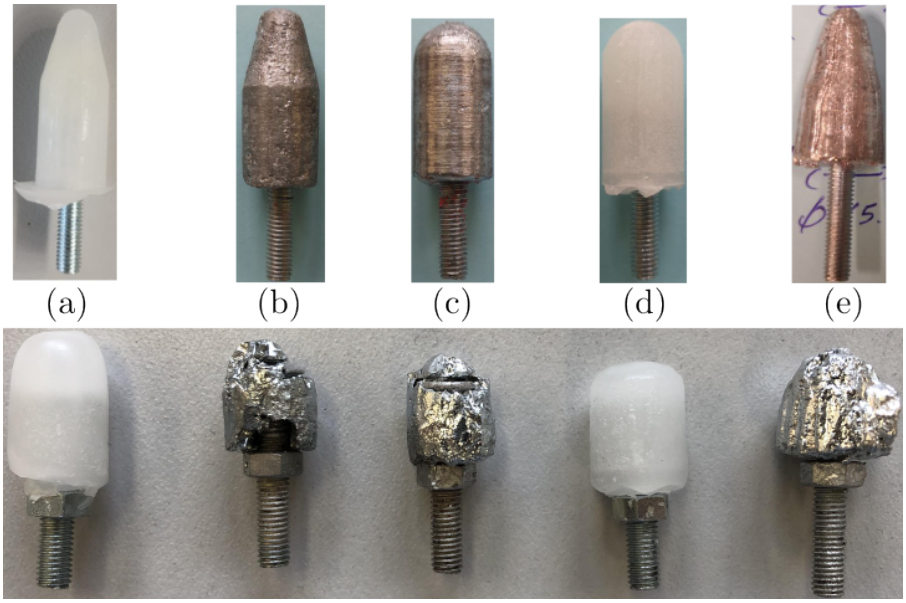


Figure 12.9 / Pictures of all the samples just after the casting process (top row) and after having been tested in H3 (bottom row). Following naming of Table 12.2: (a) WC1, (b) GC2, (c) GH3, (d) WH4, (e) GCA5.

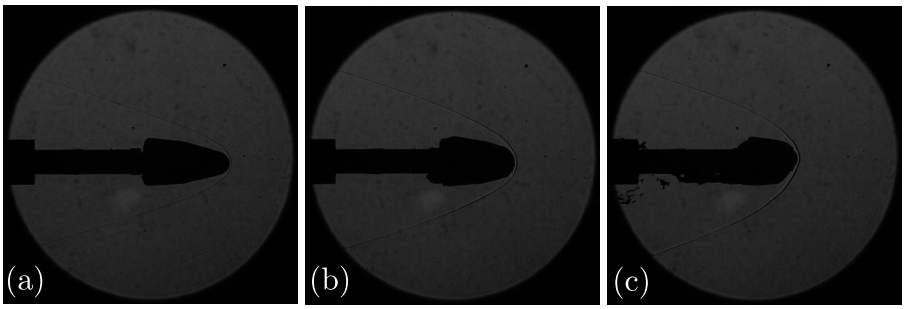


Figure 12.10 / Shadowgraph images of the GCA5 sample at the start (a), middle (b) and end (c) of the run.

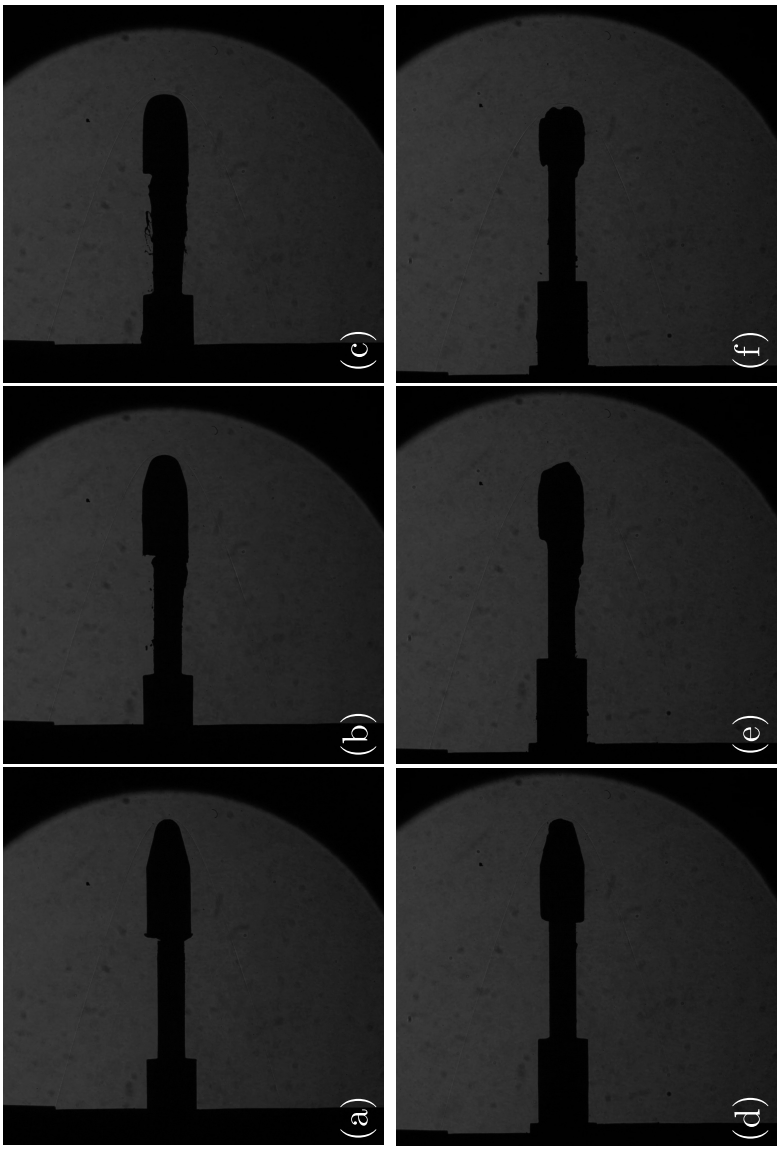


Figure 12.11 / Shadowgraph images of the WC1 (top row) and GC2 (bottom row) samples at the start (a, d), middle (b, e) and end (c, f) of the run.

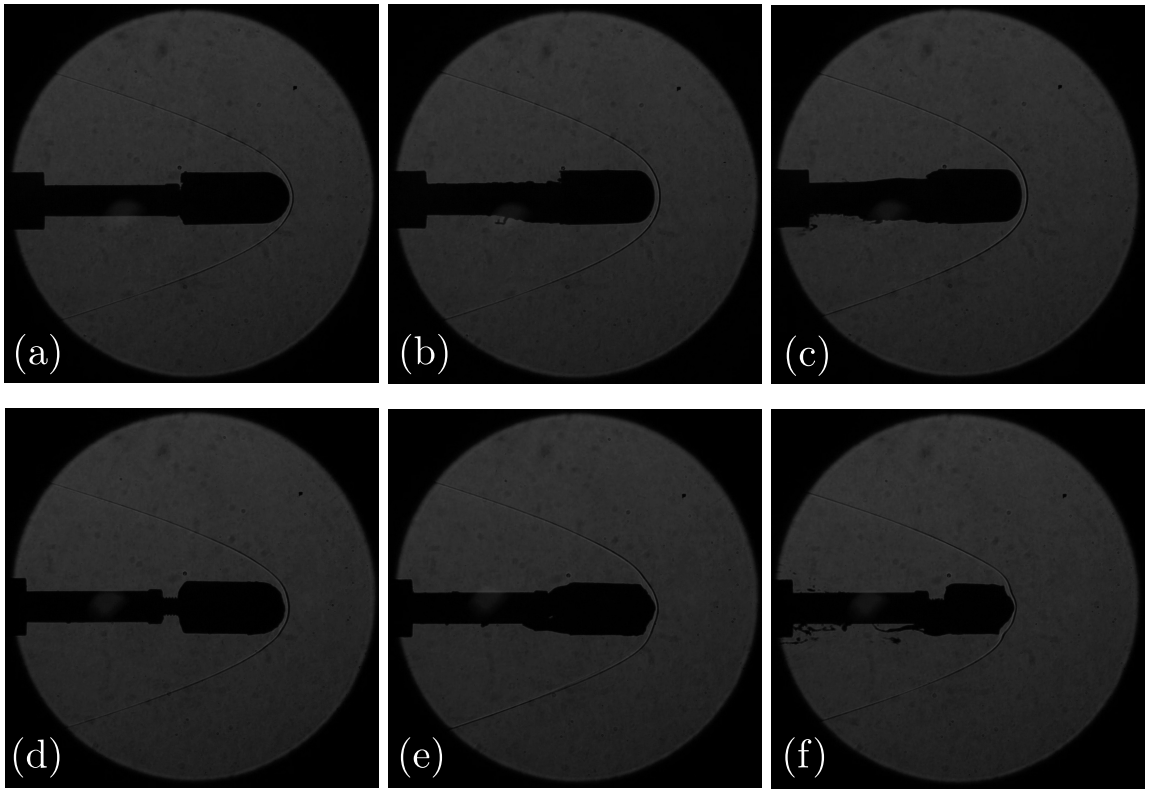


Figure 12.12 / Shadowgraph images of the WH_4 (top row) and GH_3 (bottom row) samples at the start (a, d), middle (b, e) and end (c, f) of the run.

Rest of this page intentionally left blank.

Part V

Conclusions

13

Conclusions and outlook

Interest in the space debris proliferation has been growing substantially over the last decade because of the ever-increasing threat posed by those derelict spacecrafts on the sustainability of future space activities and the risks for on-ground casualties and the environment. A promising technology under development to mitigate this problem consists in designing the spacecrafts in such a way that they can safely re-enter the Earth's atmosphere at the end of their mission and be completely destroyed before reaching the ground. Setting up this new design paradigm necessitates advanced prediction capabilities about the degradation process of re-entering debris. However, the simplified models currently in use in engineering tools lump most of the physics, disabling the detailed comprehension of the involved phenomena, yet essential for the design for demise principle. The work carried out in this thesis aims to remediate this lack of knowledge by paving the way to a numerical framework able to produce simulations with an unprecedented level of fidelity. In Section 1.4, we have identified three main objectives toward that overarching goal. In the next section, we summarize the outcomes and contributions to each of them. Afterward, based on the identified limitations of the present framework, we recommend some possible improvements as well as suggestions for future studies.

13.1 Summary and main achievements

The logic followed during this work and the writing of this thesis has been to (i) identify, (ii) to build and (iii) to assemble, one by one, the different constituent blocks of the proposed numerical tool. This logic is reflected in the structure of this manuscript which has been split into three main parts corresponding to the pillars on which this work is based, namely modeling, numerical methods, and verification and validation. Each one of them is subdivided into chapters

addressing specific aspects. The first identification phase involved a thorough screening of the literature data and is translated in this text at the beginning of each chapter (or sometimes even section) by a comprehensive state-of-the-art overview. The second building and implementation step usually necessitated extensive theoretical development that were afterward carefully coded into the main solver or a secondary script. Some of those developments were explained by means of algorithms and illustrative examples in the text. Finally, the assembly stage consisted in putting progressively together the various blocks and verifying the proper behavior of the solver based on tailored test cases. The bridges established between the blocks of our solver are apparent by the numerous connections made between the sections of the manuscript. Sticking to this philosophy, we formulated in Section 1.4 overarching objectives framing this project. We are now able to comment on them and analyze them in light of the work accomplished. We can already note that the splitting of the global objective into several ones indicates the variety of the fields that are brought together in this thesis.

Obj.1 – Modeling Derive compatible models to describe the gas, liquid and solid phases, as well as their coupling

Lay down general jump conditions at the gas–liquid interface taking into account all the considered phenomena via suitable constitutive laws We identified the sharp interface local instant formulation as being the best approach in the context of this work. Indeed, in the flow regions of interest, the interface topology between the gas and the condensed phase is expected to be such that a separated or mixed multiphase flow configuration will be present. A local instant formulation to study such flows is therefore adequate. It turns out that the rigorous and exact imposition of the jump conditions offered by this formulation is desirable, or even essential, to accurately simulate interfacial instabilities and breakup in a shear-dominated environment.
Starting from the bulk models presented in Section 2.2, we derived the most general form of the jump conditions in Section 3.2. We then introduced consistent assumptions related to the targeted physics and specified the jumps to the complete compressible Navier-Stokes system. Based on the entropy jump, we finally closed the jumps by thoroughly commented constitutive laws.
This development led to the formulation of generic jump conditions, including all the phenomena targeted in this work. Thanks to their generality, more complex models could easily be plugged into the jumps. For instance, the jumps including several species and heterogeneous reactions have already been identified and could be added to the numerical framework in a future step. Actually, the solver is ready to accommodate more complex models because the generic jumps have been implemented as they are presented in this chapter.

Obtain compressible volume-averaged equations with appropriate closure to model the liquid–solid phases including phase transformation To

describe the condensed phase, we opted for a single set of equations, following a macroscale diffuse interface approach. This choice was made because of the physical diffuse nature of the liquid-solid interface in general and to obtain equations which are amenable for computation by modeling all the intricate features of the interfacial region rather than explicitly representing them. More precisely, we selected an equilibrium homogeneous mixture formulation of the problem rather than a more complicated two-fluid model for facility reasons and because we believe that the level of modeling fidelity brought by the mixture model is sufficient for the targeted application.

With the final desired model in mind, we started our derivation by presenting the main steps in the volume averaging of the fluid equations in Section 4.2. This allowed us to develop the most complete form of the volume-averaged Navier-Stokes equations, to relate the appearing interfacial exchange terms with the jump conditions treated previously, and to discuss the closure of some of the terms. We introduced in a step-by-step manner through Section 4.3 duly justified hypotheses to simplify the general two-fluid model toward our final mixture model. By doing so, we included effective properties and source terms to model not only the liquid-solid diffuse interface region but also the phase transformation between the two phases.

The original aspects of the proposed mixture model are its modularity and the combination of several approaches gathered from the literature, including phase transformation models. Although simplifying assumptions were invoked at this stage, those were clearly identified and relaxing some of them should not be more complicated than adding in the solver some of the terms we have neglected so far. Furthermore, the mixture governing laws have been implemented as a derived class of the one featuring the Navier-Stokes equations with general equations of state (EOS). This means that in practice, any improvements to the mixture model will only impact the few overloaded routines in the derived class. This object-oriented design allowed a relatively rapid implementation of the mixture model into the solver.

Another important point to mention is the compressible nature of our mixture model which asked for closure by appropriate constitute laws. The purpose of Section 2.3.3 was to derive such closure models by enforcing thermodynamic constraints to a general form of an analytical EOS. The proposed methodology to develop thermodynamically consistent EOS stands out from what can be found in the literature because it provides a general way to derive several families of EOS, based on only a few parameters. Although the mixture model combined with the new Mie-Gruneisen EOS was only applied to the test cases in Section 11.7.2, it has proven its ability to take into account liquid-solid phase change for real-case properties and to be coupled with the compressible gas equations via the local instant formulation covered before.

Obj.2 – Numerical methods Extend a discontinuous Galerkin framework to a compressible two-phase sharp interface solver

Improve the accuracy of the interface-capturing method and couple it with the fluid solver To track the position of the moving sharp interface between the gas and the condensed phase, our choice fell on the level-set method. This method, which implicitly embeds the interface as an iso-contour of a scalar field defined on a fixed mesh, has several advantages. In addition of its inherent ability to handle complex interface deformations and topological changes, it also perfectly fits into our high-order framework as the interface and its geometrical properties are represented, like the flow variables, with a high-order polynomial function. There are however two main challenges associated to the level-set method: its reinitialization and velocity extension, and the computation of secondary quantities, especially the interface curvature.

For the reinitialization, we have extended the original implementation present in `Argo` of a geometrical closest point approach. We indeed noticed that it was prone to important interface deformations. That is why we upgraded the original algorithm by including a high-order computation of closest points based on the solution of a constrained optimization problem, as described in Section 6.3.2. We have invested quite a bit of time in improving the robustness of the global reinitialization procedure presented in Section 6.3.3 by implementing additional techniques which could not be found in the literature. However, it seems that under-resolved regions and the presence of kinks in the level-set function compromise this robustness and would require further improvements of the currently implemented approach.

Concerning the velocity extension, we could exploit the very interesting feature of the geometrical approach according to which the extended velocity field can be directly determined based on the closest points computations, hence directly fitting into the same algorithm as the reinitialization. Particular attention was paid to the transfer of the fluid solutions from the XDG to the level-set solver as those are needed to compute the interface velocity, without or with mass transfer effects. This was covered in Section 6.4.2.

A narrow band strategy, detailed in Section 6.5.2, has also been added tp the level-set solver in order to significantly speed up the advection, reinitialization and velocity extension procedures which can now be restricted to a band of elements around the iso-0 contour. To this end, dynamic data structures and access to the flagged entities belonging to the narrow band anywhere in the code were put in place.

We also added the possibility of creating a second level-set field with continuity enforcement across the element faces to be passed to the XDG solver. The four continuity enforcement methods were compared in Section 6.6.2.

Finally, a substantial amount of time was devoted to the implementation of state-of-the-art curvature filtering techniques, with the aim of simulating surface-tension driven flows. The novelty brought by this part of our work lies in the fact that those approaches have been included inside the same high-order computational framework, which allowed us to scrutinize them on the basis

of several common criteria in Section 7.3. From this study, we could extract some general guidelines about the estimation of interface curvature for surface tension computation based on sharp interface FE type methods. To this end, a general flow chart composed of decisive steps in the curvature estimation process was established in Section 7.2.1, in addition to the choice of the filtering technique.

Adapt the spatial and temporal schemes to the unfitted discretization of moving interface problems The centerpiece of this work has been the extension of an extended discontinuous Galerkin framework to the treatment of compressible two-phase flows. We should recall here that the XDG variational formulation had already been implemented in `Argo` by Pierre Schrooyen and coworkers at Cenaero when this thesis began, and presented in P. Schrooyen et al. (2017) and P. Schrooyen et al. (2018). However, the framework was not quite ready yet to handle two-phase flows situations. That was the starting point of this project.

Let's start by recalling the reasons which led us to select a high-order geometrically unfitted method such as XDG to handle the sharp gas-condensed phase interface. The first reason is that the well-known advantages of high-order methods to discretize the equations in the bulk of a phase are at least as valuable when it comes to discretize flows with sharp interfaces, in particular involving compressible effects. The second asset of the XDG method is that it relieves the coupling between the meshing process and the discretization of the equations, which is particularly appealing for large interface deformations.

To take advantage of the benefits offered by the XDG method, four computational challenges in comparison to standard mesh-fitting FEM had to be overcome. The first one concerned the computation of integrals on arbitrary shaped cut elements defined by surfaces that do not have an explicit parametrization. Although the hierarchical moment method was implemented by Pierre Schrooyen et. al (P. Schrooyen et al. (2017)), we opted for the coupling of the solver with two external libraries providing high-order accurate quadrature rules on implicitly-defined geometries, as explained in Section 8.2.4. Those turned out indeed to be more efficient and appropriate for our applications as they define quadrature points located strictly inside their respective phase. Apart from the interfacing, we did not invest more efforts on that aspect in this project.

The second crucial element to consider was the cell agglomeration procedure to prevent conditioning issues, intersection-dependent instabilities and suboptimal error estimates due to the presence of small cut cells. This technique was already available at the beginning of this thesis. In addition of adapting some routines to two-phase flows configurations, we also had to operate a profound re-design of the implemented algorithm such that agglomerated topologies over several time levels could be formed. Detailed explanations were provided in Section 8.3.1.

Another fundamental aspect that had to be investigated as part of this work is

the time integration over time-varying domains, which is associated to the XDG spatial discretization. This poses two main challenges: (i) the time-dependence of the domains of integration in the weak form, (ii) the evolution of the cut topology and the time-dependence of the functional space. The first difficulty was addressed by first correcting the already implemented splitting approach such that it worked for two-phase flows. We then implemented a simplified space-time domain approach, called the moving interface approach, as described in Section 10.2.2. To ensure a constant cut topology over time levels and thus overcome the second challenge, the use of the agglomeration technique was extended to accommodate for both deactivating and activating degrees of freedom in a moving interface context. The assembly of the temporal residual based on this approach is presented in Section 10.2.4 on a general system of equations solved in terms of non-conservative variables, for the first time in the literature.
The last missing piece concerns the enforcement of the jump conditions across the interface and was the subject of the next dedicated sub-objective.
We can emphasize at this point that although the XDG method and its associated techniques were not invented in the present work, the developed framework remains original by the fact that the techniques borrowed from the literature had to be adapted to the specificities of the `Argo` platform. In particular, the assembly of the variational formulation presented in Section 8.2.2 is novel by its generality. We should also recall that in addition to the residuals assembly, the assembly of the Jacobians had also to be adapted.

Devise a coupling strategy enforcing jump conditions between viscous compressible phases and preserving the high-order features of the method One of the most important challenges of this thesis was to come up with a strategy to couple, via a monolithic approach, two sets of compressible Navier-Stokes equations across an interface. Such strongly coupled approach was picked out for its inherent robustness to cope with the very important stiffness and the non-linear nature of the coupled problem investigated; two aspects that are magnified by the dissimilar nature between the gas and the condensed phases involved in some simulations.
To achieve that goal, two main multiphase Riemann solvers were included into the solver to deal with potential interface instabilities emanating from the non-linear convective terms. Those can incorporate mass transfer effects by solving an internal non-linear relation. They were detailed in Section 9.2.2.
For the viscous contributions to the jump conditions, when considering the assembling of the interface term for a given element, the solution, resp. the flux, on the other side of the interface is compensated by the jump condition to correctly account for the intended discontinuity in the solution, resp. the flux. This idea, which was elaborated in Section 9.2.3, represents an extension of the original interior penalty method to the case of an interface separating two phases.
To ease the implementation of new jump conditions, two dedicated classes

have been created, one for the multiphase Riemann solver, the other one for the viscous contributions, such that they provide, independently of the discretization routines, the required fluxes and Jacobians. Concerning the new variational formulation for the interface terms developed in Section 9.3.1 and the modifications in the associated Jacobian matrix mentioned in Section 9.3.2, this required to deeply amend the assembly of the iso-0 residual in the code.
At the end, the strategy established in this part of the thesis provides a systematic approach to enforce general non-linear jump conditions between non-trivial advection-diffusion systems of equations. As such, it could potentially be applied to other discretization techniques relying on sharp interface weak formulation of interface problems. To the author's best knowledge, no previous work has applied like done here the concept of geometrically unfitted FE discretization to the sharp interface treatment of viscous compressible two-phase flows. The accomplishment of this sub-objective and the previous one clearly represent the two cornerstones of the present work.
Finally, in addition to the interface treatment, it is important to stress the proposed unifying framework for flux preconditioning based on AUSM-type schemes of Section 5.3.4 that has been implemented to deal with the very low Mach number regimes implied by the models and/or conditions considered as part of this work.

Obj.3 – Verification and experimental testing Design test cases to verify the implementation and an experimental setup to reproduce specific ablation phenomena

Build up a manufactured solution framework generating reference data satisfying prescribed jump conditions During the implementation process of the various methods covered above, a serious need to be able to verify the correctness of the models and algorithms arose. More specifically, we needed a way to build reference solutions for compressible two-phase flow problems, which appeared to be a delicate task. That is why we decided to undertake the development of a secondary numerical framework based on the Method of Manufactured Solutions (MMS), which would provide us the required reference data. The idea we followed was to define the manufactured solutions (MS) with some unknown free parameters that are determined in a second step based on a set of user-defined conditions. In our case, those conditions were mostly the interface jumps relations. As described in Appendix C, we established a versatile tool that can accommodate any new governing equations and conditions. The free parameters of the user-defined MS are computed with the available `Python--Sympy` system of equations solver while the MMS source terms are automatically generated using symbolic differentiation. The methodology is applicable not only to static but also moving interface problems. The library, named `pyManufSol`, is proposed to the community as an open-source software and can be considered as an easy-to-use tool that complements the already available ones.

The reference solutions thus generated were used to conduct formal order of convergence verification studies, like the one presented in Section 11.3 and Section 11.7.1. It did not only allow to detect coding mistakes but also glitches in the implemented formulations and algorithms.

Conceive an experimental setup isolating some of the phenomena of interest in this work At the beginning of this thesis, we were planning to conduct experiments in the high-enthalpy Plasmatron wind tunnel installed at the von Karman Institute (VKI). The high-temperature environment achievable in this facility could indeed replicate most of the extreme phenomena taking place near the surface of a re-entering object and directly driving its ablation. This initial target ended up being revised in the course of the project, to be aligned with the state of development of the numerical tool. Consequently, we carried out the design of a simplified experimental setup in a cold environment in order to isolate the ablation phenomena that can be addressed by the numerical tool in its current state, namely the melting and solidification process accompanied by the molten layer dynamics and shear removal. This design consisted in preliminary numerical simulations, the definition of the samples geometry, the elaboration of the mounting mechanism, the choice of the materials and the casting process. We finally got the chance to mount those samples in the H3 hypersonic Mach 5 facility at VKI and to perform a series of runs. For both the wax and gallium samples tested, we could distinctly observe the melting and the subsequent shearing of the liquid layer, with the formation of elongated liquid filaments and the release of droplets in the back. At the end, we did not undertake the replication of the experimental tests with our numerical solver, for its potential validation.

Following up on the description of the last objective, we should concede that the initial plan of this thesis envisaged the coverage of most of the phenomena involved in the real-case scenario of an ablating re-entering object. Manifestly, we had to reduce our ambitions on the modeling side. The main reason has to do with the amount of time invested on the numerical side, i.e. in the global second objective of this work, that we did not foresee at the beginning. It turned out in particular that the development of a high-order sharp interface two-phase flows solver is a tortuous path disseminated with pitfalls. We could then argue that the initial scope was not narrowed but simply shifted according to the obstacles encountered along our research. We believe that the numerous numerical challenges we overcame during the project were priorities in the development of the complete numerical tool for which this work lays the foundation. And because all our developments were done by preserving as much as possible the flexibility of the solver, this work opens avenues for extensions to more complex coupled multiphysics problems, including but not limited to the melting ablation in reentry conditions. Indeed, in certain respects the actual physics plays only a subordinate role in most of the numerical developments done in this work.

Coming back to the overall objective of this work stated at the beginning, and based on the assessment of the sub-objectives given above, we can assert that the present thesis indeed marks a decisive step in the development of "**a unified high-order discontinuous Galerkin interface-resolved solver to simulate ablation by melting**". Besides the theoretical developments, the implementation of all the computational methods and algorithms proposed throughout this work inside the Cenaero's in-house `Argo` software environment is a further major contribution of this thesis. Although some important developments are still required (as discussed just after), the features added to the code open up potential industrialization perspectives in the relatively near future.
We would like to recall that the approach we stuck to all along the project is exploratory and is meant to build the basis for developing new fundamental concepts and computational approaches in this research area. As it is the case for the modeling side discussed just above, efforts are still required on the numerical side to improve the robustness, efficiency and range of applications of the solver. The verification test cases treated in this work demonstrated the potential of the solver to accurately simulate viscous compressible two-phase flows in a wide range of flows regimes. This contributes to give, we believe, a favorable outcome to the transversal objective that was to "**assess the capabilities of interface-resolved high-order methods for the simulation of multiphase flows in a regime spanning from almost incompressible to compressible**". Nonetheless, we should point out that many, if not most, of the two-dimensional unsteady simulations performed in this work ended up crashing in an abrupt and unexplained fashion after some time. Elucidating and solving those robustness issues represent undeniably the priority from a numerical point of view. Finally, to shed the light on those still unexplained issues as well as to assess the fidelity of the models put in place, the linear "top-bottom" approach of the objectives addressed one after the other should be reversed such that validation and more advanced verification procedures can contribute to improve the current state of the solver.

13.2 Directions for future work and perspectives

As summarized before, substantial progress toward high-order unified simulations of melting ablation phenomena could be achieved in the present work. Nevertheless, to allow for further extensions to more complex practical scenarios, the following aspects should be considered in the future.

Improvements in the modeling for space debris ablation application

Thermochemical non-equilibrium multispecies model The first priority step in the extension of the modeling capabilities of the solver toward reentry conditions would be to couple it with the available `DGAblation` module of the `Argo` platform. This module was developed in Pierre Schrooyen (2015) and implements the compressible multispecies Navier-Stokes equations with chemical reactions source terms and thermo-transport properties depending on the flow composition and variables. Along with that coupling, the jump conditions currently implemented would need to be extended to account for the balance of species across the interface.

Look-up tables approach for more accurate EOS If we want to tackle more challenging flow conditions and take into account, for instance, phase change inside the bulk of a given phase (e.g. cavitation or boiling), dedicated external thermodynamic libraries providing highly accurate equations of state would be desirable. Look-up tabulation methods to compute and store the EOS prior to the simulation would need to be implemented and coupled to the solver.

Oxidation modeling From recent experimental results obtained by Fagnani (2023), it is clear that oxidation plays a determinant role in the ablation by melting of some materials, specifically some metals like titanium. The oxide layer not only alters significantly the emissivity of the surface material, it can also be peeled off, hence being directly involved in the removal of some mass of the object. The addition of surface heterogeneous reactions terms (such as in Helber et al. (2019)) and the change of the emissivity to account for the effect of oxidation at the object surface seems necessary. However, this is not sufficient if we want to predict the evolution of the oxide layer which would require a volumetric approach such as the one investigated in Jorge (2019).

Incoming and outgoing radiation Taking into account re-radiation from the hot surface is very important because it can greatly influence the temperature evolution of the body, hence also its ablation. Re-radiation could be modeled, as commonly done, by the addition of a radiation sink term at the gas-condensed phase interface based on the Stefan-Boltzmann law. We would need to be particularly careful to the choice of the emissivity coefficient and its dependence on the temperature and surface properties. For certain reentry conditions, especially at high velocity and low densities, radiative heat transfer from the gas in the shock layer can substantially alter the aerothermodynamic environment. The numerical simulation of radiative transfer is a challenging problem because of the spatial, angular, and spectral dependence of the radiation field. The coupling of the flow solver with a method such as the Hybrid Statistical Narrow Band model proposed in Soucasse et al. (2016) could be an option to explore.

Break-up modeling The combination of aerodynamic forces, body forces and thermal loads may lead to the fracture of structural joints and the break-up of the debris in several pieces. This will obviously influence the survivability and point of impacts of the re-entering fragments and this is therefore logical that all the engineering tools to predict debris reentry include a fragmentation model. The accurate computation of fragmentation in a CFD context would require to couple the solver with a structural one, leading to a so-called Fluid-Structure Interaction (FSI) framework. This has already been investigated as part of the ESA's funded *DSCMFED* project where a Direct Simulation Monte Carlo solver was coupled with the Cenaero's in-house Finite Element solver `Morfeao` for thermo-mechanical analysis.

Effect of deceleration during re-entry The strong deceleration forces that a re-entering body could experience is expected to play a non-negligible role in the way the liquid molten layer will be removed from the surface. Modeling this effect by an appropriate body force source term would be an interesting additional feature to the solver.

Improvements of space debris reentry prediction It must be recalled that one of the main purposes of developing high-fidelity numerical tool like the one in this thesis is to contribute to the improvements of the over-simplified correlations currently present in most of the engineering tools in use. To this end, a convenient way to exploit the richness contained in data produced by high-fidelity simulations would be to build efficient surrogate-based which encompass complex physics while being suited to be incorporated in a low-fidelity tool. This approach was followed in the ESA's funded project *DEMISE* led by VKI and Cenaero.

Additional features for the bulk fluid solver

Three-dimensional equations The solver has only been verified on two-dimensional configurations up to now. Thanks to the structure of `Argo`, the extension to three dimensions of the conservation laws and the assembly of the variational formulation should not pose any specific difficulties. We might however face some peculiar issues in the operations linked to the XDG method such as cell-agglomeration, quadrature generation or flagging of cut entities.

Improved shock capturing capabilities Stability of high-order methods can be severely compromised by the Gibbs phenomena around discontinuous solutions. This is all the more true for strong shock waves that appear in hypersonic reentry conditions. The design of techniques ensuring stability of the computation while limiting as much as possible the order of accuracy degradation of the original scheme is an ongoing very active research area in the high-order community. The combination of localized artificial viscosity with entropy preserving schemes is the topic of the ongoing PhD thesis of Amaury Bilocq

entitled "Minimizing the impact of shock capturing method for scale resolved simulation of transonic turbulent flows" at ULiège under the supervision of Koen Hillewaert. Another thesis, realized by Maxime Borbouse at ULiège in the framework of the *Space4Relaunch* project funded by the Walloon region, just started on the same topic.

Time step adaptation for implicit scheme The local discretization errors introduced during the advancement of the solution from one time step to the next one consist of spatial, temporal and iterative errors. Spatial errors can be estimated and reduced through mesh- or order-adaptive techniques. Unstructured mesh adaptation based on the work of Compère (2009) and Coulaud and Loseille (2016) has already been applied in `Argo`. On the other hand, an adaptive time-stepping strategy combined with an adaptive Newton tolerance are not available and would also be desirable to automatically set a time step and a nonlinear tolerance such that the temporal and iterative errors are smaller than the spatial error. Otherwise, the benefits of high-order spatial accuracy can be drowned out. Such balanced time step adaptation strategies taking into account spatial discretization errors that could be explored have been recently proposed by Pan et al. (2022) and Fidkowski (2023).

Robust non-linear and linear iterative solvers for stiff problems Although the implicit schemes exploited in this thesis present the great advantage of being unconditionally stable, this does not mean that the solution will not diverge. One of the reasons why this may happen is that the local Newton's methods used to solve the non-linear system does not guarantee the convergence of the non-linear iterations. To overcome these robustness issues, globalization of standard local Newton's solvers, through backtracking or trust-region methods, are possible. In the inner iterations of the Newton's method, the use of an iterative linear solver can also be responsible for the divergence of the solutions. As a matter of fact, that is the reason why the majority of the simulations have been performed with an exact direct linear solver. To improve the reliability of the GMRES solver available in `Argo` applied to very stiff problems such as the ones considered in this work, the design of a proper preconditioning seems necessary. Inspiration could be taken e.g. from the work of Weston et al. (2017).

Improved liquid-solid phase change modeling It is clear from the derivation conducted in Section 4.3 that the volume-averaged mixture model to represent the liquid-solid region relies on a quite important number of simplifying assumptions. It would be interesting to relax some of them and evaluate their importance in the modeling fidelity we want to achieve for the targeted applications. Furthermore, we should mention that the use of the Mie-Gruneisen EOS presented in Section 2.3.3 leads for the moment to an important computational overhead because of the complexity of the expressions and the non-linear relations involved in some of them. A strategy to alleviate this slow-down should be found.

Additional features for the level-set method

More accurate interface transport It is well known that level-set methods are plagued with mass conservation issues. Marchandise et al. (2006, Sec. 1.3) pointed out that the severity of this issue is mainly dependent on the method used to discretize the level-set. The first step to be done would therefore be to quantify the conservation errors as a function of the spatial and temporal order of accuracy of the schemes used in this work. Based on the outcomes of this analysis, it may be necessary to put in place techniques that reduce the conservation issues, such as the coupled level-set and volume-of-fluid method (Sussman and Puckett (2000)) or the reference map method (Bellotti and Theillard (2019)). Moreover, to avoid under-resolved areas in regions of high interface deformations, mesh adaptation including a metric based on the local curvature of the level-set iso-0 should be applied. Another option to improve the accuracy of the interface transport, and which would require more efforts, would be to combine the level-set with Lagrangian markers sampling the interface. This was successfully applied in Robert Nourgaliev et al. (2010).

Improved reinitialization and extension procedures Concerning the reinitialization and extension procedures, the implementation of the two following elements would be highly profitable. The first one concerns the detection and treatment of ill-conditioned points during the closest point computation. The presence of those points can mislead the gradient descent and more widely impact the accuracy of level set methods. The algorithm developed in Henri et al. (2022b) should be considered. The second one is about the implementation of PDE-based approaches for both the reinitialization and velocity extension, with the aim of comparing them with the current closest-point geometrical approach. In particular, the elliptic extension procedure proposed in Utz and Kummer (2018) seems promising.

Treatment of more than one interface All the developments done in this work assumed the presence of a single interface (possibly broken apart) between two phases. There are however a host of physical problems that involve more than one interconnected moving interfaces. If we want to tackle this kind of problems in the future, the standard level-set method considered in this thesis would need to be upgraded. The first possibility would be to introduce multiple level-set functions. This is in theory already possible to do so in `Argo`. But this would necessitate a profound redesign of the XDG weak form assembly which has been implemented with a single interface in mind. Another promising technique is the Voronoi Implicit Interface Method presented by Robert I Saye and James A Sethian (2011).

Additional features for the XDG method

Treatment of agglomeration chains It was mentioned in Section 8.2.5 that the treatment of agglomeration chains in the cell-agglomeration procedure

has been avoided in this work. Thus, extension of target cells basis functions into source cells which do not share any common face are allowed. It seems imperative to extend the implemented algorithm such that those chains can be taken into account if needed. Indeed, we do not know at this stage whether the simplification we made in this thesis could introduce some inaccuracies perturbing the stability of the whole method.

Time integration Related to this same concern about stability of the computations, which is clearly one of the bottlenecks of our solver, investigations must be carried on about the temporal integration of the equations over time-varying domains. We are indeed convinced that, according to the few results we obtained with the two time integration techniques implemented, the splitting of the level-set from the XDG fluid equations may be responsible for the onset of instabilities during the temporal integration of the fluid equations. Therefore, a thorough analysis of the influence of the parameters involved in the process, such as the time step, the time-stepping scheme, the temporal integration strategy (splitting VS moving interface) or the spatial discretization, reveals necessary. A possible outcome of this analysis could be that to guarantee stability, a full space-time domain approach would be needed.

Mathematical analysis of the formulation Although the weak consistency of the novel XDG variational formulation was analyzed in Section 9.3.3, the approach presented in this work would benefit from being consolidated by mathematical analysis of inf-sup stability and coercivity as well as a priori error estimates.

Additional features for the XDG jumps enforcement strategy

Splitting strategies assessment for mass transfer We presented in Table 9.1 the two possible splittings between the convective and diffusive fluxes inside the jump conditions. Without mass transfer, they lead to identical formulation. However, it is not the case when mass transfer effects are present and only the first splitting has been applied in this work in such scenario. A comparison of those two proposed approaches, which each relies on a specific multiphase Riemann solver, would be expedient.

Additional features for the XDG–level-set coupling

More accurate coupling strategy We did not spend time implementing more advanced coupling algorithms than the simplest first-order Lie coupling. A natural perspective would be to start by implementing a second-order Strang coupling between the two solvers and compare the performances of the two approaches. We particularly think that an important point that remains to be elucidated is how the accuracy of the coupling between the level-set and XDG solvers can impact the global accuracy and stability of the whole solver. This

would be inevitably correlated to the study we proposed to perform above about the time integration of the fluid equations on moving domains. To go one step further, it would also be worth trying the operator splitting strategy with implicit coupling described in Smuda (2021, Sec. 3.9) where the level-set position is updated inside the fluid iterations.

Investigation of conservation properties Related to the previous point and the time integration of the XDG equations, it would be instructive to assess the global conservation properties of the solver by simultaneously bringing into play the two different integration approaches for moving domains and the operator splitting strategies with an explicit and implicit coupling.

Use of different interpolation orders The structure of `Argo` does not allow for the moment to select distinct interpolation orders between the level-set and XDG solvers. The modifications of the code to overcome this limitation were undertaken during this thesis but not applied in practice. The continuation of this effort is foreseen.

Improvements in code performance

Parallel implementation for XDG The `Argo` platform is designed to perform massively parallel simulations by relying on a hybrid parallelism between message passing interface (MPI) and open multiprocessing (OpenMP). However, the new module implementing the XDG method can only be run with OpenMP up to now. The main reason has to do with the important complications brought by the cell-agglomeration procedure in the domain partitioning required for MPI. Indeed, we would need to ensure that associated target and source elements remain on the same partition or if not, come up with a strategy to adapt the ghost values at the faces between source and target elements. This would necessarily involve on-the-fly rearrangement of the MPI data and therefore important implementation efforts. But those efforts are indispensable to be able to hope to apply the numerical tool to large-scale three-dimensional configurations representative of industrial situations.

Combination of h and p adaptation Another aspect on which a non-negligible amount of time has been devoted during this project concerned the extension of the available mesh adaptation framework in `Argo` to the case of two-phase flows based on a XDG discretization. The whole procedure of mesh-to-mesh interpolation has been updated to accommodate the presence of an interface separating the mesh into distinct computational domains (e.g. the solution in phase 1 on the old mesh cannot be used during the interpolation of the solution in phase 2 on the new mesh). This was not exploited as part of this work because the last step about the definition of appropriate metrics to drive the adaptation has still to be done. About p-adaptation, `Argo` is able to deal with different interpolation orders between elements but an automatic strategy

based on error estimations similar to the one for mesh adaptation has still to be put in place.

Extended verification

From a general perspective, the solver would clearly benefit from being further verified on a wider range of test cases, with the goal of exercising each one of its building blocks.

Truly high-order simulations An important aspect to address is the ability of the solver to truly exploit the advantages of high-order discretizations. Indeed, the test cases presented in this work were most of the time obtained with a third order accurate discretization ($p = 2$); some of them with $p = 3$. With all the efforts that have gone into setting up this high-order solver, it is legitimate to expect that its potential will be better exploited.

Sharp VS diffusive interface method The merits of the interface-resolved approach followed in this work concerning its ability to accurately capture arbitrarily important jumps at the interface should be put into perspectives with the capabilities of diffuse interface methods, which are usually the methods of choice in the context of compressible two-phase flows. Several benchmark test cases representative of a wide range of flow conditions should be selected and simulated with the two approaches. The conclusions that could be drawn from this study could certainly be the subject of a publication, given the scarcity of analyses of this kind in the literature.

Unified VS segregated approach Another highly valuable avenue of research is about the comparison between the current unified solver and the other segregated paradigm which consists in using separated solvers for each domain and coupling them. It is known that monolithic approaches tend to be more stable, but at the expenses of less flexibility and increased computational cost compared to their segregated counterparts. To be able to deduce more precise and meaningful conclusions, those two approaches should be applied on melting ablation situations, provided that we are able to build a segregated framework sharing the same discretization schemes as the unified one. To help in that direction, we could potentially rely on the segregated approach developed in the framework *DEMISE* ESA's project.

Experimental setup improvement and CFD replication

Finally, we should build upon the preliminary experimental tests presented in this work to arrive at an improved setup providing exploitable validation data for our CFD solver. We should in particular focus our attention on improving

the manufacturing quality and precision of the samples and on developing visualization techniques that would allow to accurately extract the time evolution of the gas-liquid interface.

Rest of this page intentionally left blank.

Appendices

A

Expression of the fluxes Jacobians

This appendix presents the analytical expression of the Jacobian matrices of some fluxes appearing in the DG weak formulation.

A.1 Expression of the homogeneity tensor for compressible equations closed by general EOS

The global homogeneity tensor $\underline{\boldsymbol{G}}(\mathbf{u})$ introduced in Equation 5.13 is composed, in two dimensions, by four homogeneity tensors $\underline{\boldsymbol{G}}_{kl}(\mathbf{u})$

$$\underline{\boldsymbol{G}} = \begin{bmatrix} \underline{\boldsymbol{G}}_{xx} & \underline{\boldsymbol{G}}_{xy} \\ \underline{\boldsymbol{G}}_{yx} & \underline{\boldsymbol{G}}_{yy} \end{bmatrix} \tag{A.1}$$

where the tensors $\underline{\boldsymbol{G}}_{kl}(\mathbf{u})$ are defined as the Jacobians of the diffusive fluxes $\underline{\boldsymbol{F}}^d(\mathbf{u}, \nabla\mathbf{u})$ with respect to the solution gradients:

$$\begin{aligned} \underline{\boldsymbol{G}}_{kl} &= \frac{\partial \mathbf{F}_k^d}{\partial \nabla_l \mathbf{u}} \,, \\ \iff \mathbf{F}_k^d &= \underline{\boldsymbol{G}}_{kl} \, \partial \nabla_l \mathbf{u} \,, \end{aligned} \tag{A.2}$$

with $k = \{x, y\}$ and $l = \{x, y\}$ in two dimensions. In order to explain how to derive analytical expressions for the different $\underline{\boldsymbol{G}}_{kl}$ when the conservative

variables set $\mathbf{u}$ is employed, the example of the tensor $\underline{\boldsymbol{G}}_{xx}$ is considered here:

$$\underline{\boldsymbol{G}}_{xx} = \frac{\partial \mathbf{F}_x^d}{\partial \nabla_x \mathbf{u}} = \begin{bmatrix} \frac{\partial F_{x,1}^d}{\partial \nabla_x u_1} & \frac{\partial F_{x,1}^d}{\partial \nabla_x u_2} & \frac{\partial F_{x,1}^d}{\partial \nabla_x u_3} & \frac{\partial F_{x,1}^d}{\partial \nabla_x u_4} \\ \frac{\partial F_{x,2}^d}{\partial \nabla_x u_1} & \frac{\partial F_{x,2}^d}{\partial \nabla_x u_2} & \frac{\partial F_{x,2}^d}{\partial \nabla_x u_3} & \frac{\partial F_{x,2}^d}{\partial \nabla_x u_4} \\ \frac{\partial F_{x,3}^d}{\partial \nabla_x u_1} & \frac{\partial F_{x,3}^d}{\partial \nabla_x u_2} & \frac{\partial F_{x,3}^d}{\partial \nabla_x u_3} & \frac{\partial F_{x,3}^d}{\partial \nabla_x u_4} \\ \frac{\partial F_{x,4}^d}{\partial \nabla_x u_1} & \frac{\partial F_{x,4}^d}{\partial \nabla_x u_2} & \frac{\partial F_{x,4}^d}{\partial \nabla_x u_3} & \frac{\partial F_{x,4}^d}{\partial \nabla_x u_4} \end{bmatrix} , \tag{A.3}$$

with the four components of the diffusive fluxes along the x-direction given by

$$\begin{aligned} F_{x,1}^d &= 0 , \\ F_{x,2}^d &= \tau_{xx} = \mu \left(\frac{4}{3} \frac{\partial v_x}{\partial x} - \frac{2}{3} \frac{\partial v_y}{\partial y} \right) , \\ F_{x,3}^d &= \tau_{xy} = \mu \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right) , \\ F_{x,4}^d &= v_x \cdot \tau_{xx} + v_y \cdot \tau_{xy} - q_x = v_x \cdot \tau_{xx} + v_y \cdot \tau_{xy} + \lambda \frac{\partial T}{\partial x} . \end{aligned} \tag{A.4}$$

The vector $\mathbf{F}_x^d$ thus involves five different derivatives:

$$\mathbf{F}_x^d \rightarrow \frac{\partial v_x}{\partial x} , \frac{\partial v_x}{\partial y} , \frac{\partial v_y}{\partial x} , \frac{\partial v_y}{\partial y} , \frac{\partial T}{\partial x} . \tag{A.5}$$

The first thing to do is to express v_x, v_y and T in terms of the conservative variables components, i.e. $\mathbf{u} = (\rho , \rho v_x , \rho v_y , \rho E) = (u_1 , u_2 , u_3 , u_4)$:

$$\begin{aligned} v_x &= \frac{\rho v_x}{\rho} = \frac{u_2}{u_1} , \\ v_y &= \frac{\rho v_y}{\rho} = \frac{u_3}{u_1} , \\ T(\rho, e) &= T\left(\rho, \frac{\rho E}{\rho} - \frac{1}{2} \frac{(\rho v_x)^2 + (\rho v_y)^2}{\rho^2} \right) = T\left(u_1, \frac{u_4}{u_1} - \frac{1}{2} \frac{u_2^2 + u_3^2}{u_1^2} \right) \\ &\triangleq T^*(\mathbf{u}) , \end{aligned} \tag{A.6}$$

where the temperature is expressed from the conservative variable through a relation T^* making use of the equation of state (EOS) employed. Then, the five

derivatives in Equation A.5 are expressed as a function of $\mathbf{u}$:

$$\begin{aligned}
\frac{\partial v_x}{\partial x} &= \frac{\partial}{\partial x}\left(\frac{u_2}{u_1}\right) = \frac{1}{u_1}\frac{\partial u_2}{\partial x} - \frac{u_2}{u_1^2}\frac{\partial u_1}{\partial x}\,, \\
\frac{\partial v_x}{\partial y} &= \frac{\partial}{\partial y}\left(\frac{u_2}{u_1}\right) = \frac{1}{u_1}\frac{\partial u_2}{\partial y} - \frac{u_2}{u_1^2}\frac{\partial u_1}{\partial y}\,, \\
\frac{\partial v_y}{\partial x} &= \frac{\partial}{\partial x}\left(\frac{u_3}{u_1}\right) = \frac{1}{u_1}\frac{\partial u_3}{\partial x} - \frac{u_3}{u_1^2}\frac{\partial u_1}{\partial x}\,, \\
\frac{\partial v_y}{\partial y} &= \frac{\partial}{\partial y}\left(\frac{u_3}{u_1}\right) = \frac{1}{u_1}\frac{\partial u_3}{\partial y} - \frac{u_3}{u_1^2}\frac{\partial u_1}{\partial y}\,, \\
\frac{\partial T^*}{\partial x} &= \frac{\partial T}{\partial \rho}\frac{\partial \rho}{\partial x} + \frac{\partial T}{\partial e}\frac{\partial e}{\partial x}\,,
\end{aligned} \tag{A.7}$$

where:

$$\begin{aligned}
\frac{\partial \rho}{\partial x} &= \frac{\partial u_1}{\partial x}\,, \\
\frac{\partial e}{\partial x} &= \frac{\partial}{\partial x}\left(\frac{u_4}{u_1} - \frac{1}{2}\frac{u_2^2+u_3^2}{u_1^2}\right) \\
&= \left[\frac{u_2^2+u_3^2}{u_1^3} - \frac{u_4}{u_1^2}\right]\frac{\partial u_1}{\partial x} - \frac{u_2}{u_1^2}\frac{\partial u_2}{\partial x} - \frac{u_3}{u_1^2}\frac{\partial u_3}{\partial x} + \frac{1}{u_1}\frac{\partial u_4}{\partial x}
\end{aligned} \tag{A.8}$$

To be able to assemble the Jacobian with respect to the conservative solution gradient along the x-direction, the last step is to insert Equation A.7 and Equation A.8 into Equation A.4 and compute the derivatives of $\mathbf{F}_x^d$ with respect to $\nabla_x \mathbf{u}$. This yields:

$$\underline{\boldsymbol{G}}_{xx} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
-\frac{4}{3}\frac{\mu}{\rho}v_x & \frac{4}{3}\frac{\mu}{\rho} & 0 & 0 \\
-\frac{\mu}{\rho}v_y & 0 & \frac{\mu}{\rho} & 0 \\
\lambda\left(\frac{\partial T}{\partial \rho} - \frac{1}{\rho}e_m\frac{\partial T}{\partial e}\right) + \phi_1^{xx} & -\lambda\frac{v_x}{\rho}\frac{\partial T}{\partial e} + \phi_2^{xx} & -\lambda\frac{v_y}{\rho}\frac{\partial T}{\partial e} + \phi_3^{xx} & \lambda\frac{1}{\rho}\frac{\partial T}{\partial e} + \phi_4^{xx}
\end{bmatrix}, \tag{A.9}$$

with:

$$\begin{aligned}
e_m &= E - v_x^2 - v_y^2\,, \\
\phi_i^{xx} &= v_x\frac{\partial F_{x,2}^d}{\partial(\nabla_x u_i)} + v_y\frac{\partial F_{x,3}^d}{\partial(\nabla_x u_i)}\,.
\end{aligned} \tag{A.10}$$

The three other tensors can be computed similarly to obtain:

$$\underline{\boldsymbol{G}}_{xy} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{2}{3}\frac{\mu}{\rho}v_y & 0 & -\frac{2}{3}\frac{\mu}{\rho} & 0 \\ -\frac{\mu}{\rho}v_x & \frac{\mu}{\rho} & 0 & 0 \\ \phi_1^{xy} & \phi_2^{xy} & \phi_3^{xy} & \phi_4^{xy} \end{bmatrix}, \tag{A.11}$$

with:

$$\phi_i^{xy} = v_x \frac{\partial F_{x,2}^d}{\partial(\nabla_y u_i)} + v_y \frac{\partial F_{x,3}^d}{\partial(\nabla_y u_i)}, \tag{A.12}$$

and

$$\underline{\boldsymbol{G}}_{yx} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{\mu}{\rho}v_y & 0 & \frac{\mu}{\rho} & 0 \\ \frac{2}{3}\frac{\mu}{\rho}v_x & -\frac{2}{3}\frac{\mu}{\rho} & 0 & 0 \\ \phi_1^{yx} & \phi_2^{yx} & \phi_3^{yx} & \phi_4^{yx} \end{bmatrix}, \tag{A.13}$$

with:

$$\phi_i^{yx} = v_x \frac{\partial F_{y,2}^d}{\partial(\nabla_x u_i)} + v_y \frac{\partial F_{y,3}^d}{\partial(\nabla_x u_i)}, \tag{A.14}$$

and

$$\underline{\boldsymbol{G}}_{yy} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{\mu}{\rho}v_x & \frac{\mu}{\rho} & 0 & 0 \\ -\frac{4}{3}\frac{\mu}{\rho}v_y & 0 & \frac{4}{3}\frac{\mu}{\rho} & 0 \\ \lambda\left(\frac{\partial T}{\partial \rho} - \frac{1}{\rho}e_m\frac{\partial T}{\partial e}\right) + & -\lambda\frac{v_x}{\rho}\frac{\partial T}{\partial e} + & -\lambda\frac{v_y}{\rho}\frac{\partial T}{\partial e} + & \lambda\frac{1}{\rho}\frac{\partial T}{\partial e} + \phi_4^{yy} \\ \phi_1^{yy} & \phi_2^{yy} & \phi_3^{yy} & \end{bmatrix}, \tag{A.15}$$

with:

$$\phi_i^{yy} = v_x \frac{\partial F_{y,2}^d}{\partial(\nabla_y u_i)} + v_y \frac{\partial F_{y,3}^d}{\partial(\nabla_y u_i)}. \tag{A.16}$$

It is worth noting that the choice of the equation of state will intervene in the homogeneity tensor through the two derivatives $\frac{\partial T}{\partial \rho}$ and $\frac{\partial T}{\partial e}$.

A.2 Construction of the Jacobian of the convective flux with respect to conservative variables

The vector of conservative variables in two dimensions is rewritten here for convenience:

$$\mathbf{u} = \left(\rho,\, \rho v_x,\, \rho v_y,\, \rho E\right)^T = \left(u_1,\, u_2,\, u_3,\, u_4\right)^T \,. \tag{A.17}$$

In the following, the steps to obtain the expression of the Jacobian of the convective flux in the x-direction are explained, based on Colonna and S. Rebay (2004). The determination of the Jacobian for the y-direction follows exactly the same procedure. The x-component of the convective flux and its associated Jacobian matrix are written respectively as:

$$\begin{aligned} \mathbf{F}_x^c(\mathbf{u}) &= \left(u_2,\, \frac{u_2^2}{u_1} + p,\, \frac{u_2 u_3}{u_1},\, \frac{u_2 u_4}{u_1} + \frac{u_2}{u_1} p\right) \,, \\ \mathbf{J}_x^c &= \frac{\partial \mathbf{F}_x^c(\mathbf{u})}{\partial \mathbf{u}} \,. \end{aligned} \tag{A.18}$$

Since the convective flux F^c depends on the pressure p and the Jacobian matrix is built from the derivatives of F^c with respect to the conservative variables $\mathbf{u}$, it is convenient to introduce a relation which expresses the pressure as a function of the conservative variable vector, p^*, as:

$$\begin{aligned} p(\rho, e) &= p\left(\rho,\, \frac{\rho E}{\rho} - \frac{1}{2}\frac{(\rho v_x + \rho v_y)^2}{\rho^2}\right) = p\left(u_1,\, \frac{u_4}{u_1} - \frac{1}{2}\frac{(u_2 + u_3)^2}{u_1^2}\right) \,, \\ &\triangleq p^*(\mathbf{u}) \,. \end{aligned} \tag{A.19}$$

The different terms of the matrix $\mathbf{J}_x^c$ can then be computed as follows:

$$\begin{aligned} J_x^c(1,1) &= \frac{\partial}{\partial u_1}(u_2) = 0 & J_x^c(2,1) &= \frac{\partial}{\partial u_1}\left(\frac{u_2^2}{u_1} + p\right) = -v_x^2 + \frac{\partial p^*}{\partial \rho} \\ J_x^c(1,2) &= \frac{\partial}{\partial u_2}(u_2) = 1 & J_x^c(2,2) &= \frac{\partial}{\partial u_2}\left(\frac{u_2^2}{u_1} + p\right) = 2v_x + \frac{\partial p^*}{\partial (\rho v_x)} \\ J_x^c(1,3) &= \frac{\partial}{\partial u_3}(u_2) = 0 & J_x^c(2,3) &= \frac{\partial}{\partial u_3}\left(\frac{u_2^2}{u_1} + p\right) = \frac{\partial p^*}{\partial (\rho v_y)} \\ J_x^c(1,4) &= \frac{\partial}{\partial u_4}(u_2) = 0 & J_x^c(2,4) &= \frac{\partial}{\partial u_4}\left(\frac{u_2^2}{u_1} + p\right) = \frac{\partial p^*}{\partial (\rho E)} \end{aligned}$$

$$
\begin{aligned}
J_x^c(3,1) &= \frac{\partial}{\partial u_1}\left(\frac{u_2 u_3}{u_1}\right) = -v_x v_y & J_x^c(4,1) &= \frac{\partial}{\partial u_1}\left(\frac{u_2 u_4}{u_1} + \frac{u_2}{u_1}p\right) = -v_x H + v_x \frac{\partial p^*}{\partial \rho} \\
J_x^c(3,2) &= \frac{\partial}{\partial u_2}\left(\frac{u_2 u_3}{u_1}\right) = v_y & J_x^c(4,2) &= \frac{\partial}{\partial u_2}\left(\frac{u_2 u_4}{u_1} + \frac{u_2}{u_1}p\right) = H + v_x \frac{\partial p^*}{\partial (\rho v_x)} \\
J_x^c(3,3) &= \frac{\partial}{\partial u_3}\left(\frac{u_2 u_3}{u_1}\right) = v_x & J_x^c(4,3) &= \frac{\partial}{\partial u_3}\left(\frac{u_2 u_4}{u_1} + \frac{u_2}{u_1}p\right) = v_x \frac{\partial p^*}{\partial (\rho v_y)} \\
J_x^c(3,4) &= \frac{\partial}{\partial u_4}\left(\frac{u_2 u_3}{u_1}\right) = 0 & J_x^c(4,4) &= \frac{\partial}{\partial u_4}\left(\frac{u_2 u_4}{u_1} + \frac{u_2}{u_1}p\right) = v_x + v_x \frac{\partial p^*}{\partial (\rho E)}
\end{aligned}
$$

It can be noticed that all the terms of the convective Jacobian can be obtained with the set of conservative variables and the derivatives of the function $p^*(\mathbf{u})$ with respect to the conservative variables:

$$
\begin{aligned}
\frac{\partial p^*}{\partial \rho} &= \left(\frac{\partial p}{\partial \rho}\right)_e - \left(\frac{\partial p}{\partial e}\right)_\rho \frac{1}{\rho}\left(e - \frac{1}{2}\mathbf{v}\cdot\mathbf{v}\right), \\
\frac{\partial p^*}{\partial (\rho v_x)} &= -\frac{v_x}{\rho}\left(\frac{\partial p}{\partial e}\right)_\rho, \\
\frac{\partial p^*}{\partial (\rho v_y)} &= -\frac{v_y}{\rho}\left(\frac{\partial p}{\partial e}\right)_\rho, \\
\frac{\partial p^*}{\partial (\rho E)} &= \frac{1}{\rho}\left(\frac{\partial p}{\partial e}\right)_\rho.
\end{aligned}
\qquad [A.20]
$$

The derivatives $\left(\frac{\partial p}{\partial \rho}\right)_e$ and $\left(\frac{\partial p}{\partial e}\right)_\rho$, which depend on the EOS, are needed to build the Jacobian of the convective flux for general EOS.

B

Molecular Dynamics Simulation to estimate thermophysical properties of liquid metals

This appendix is a shorter version of the technical report written by the author following his nine weeks scientific research stay at NASA Ames during the summer 2019.

B.1 Theory of Molecular Dynamics

B.1.1 Molecular Modeling Approaches

Cuendet (2008) describes Molecular Modeling as being "concerned with the description of the atomic and molecular interactions that govern microscopic and macroscopic behaviors of physical systems. The essence of molecular modeling resides in the connection between the macroscopic world and the microscopic world provided by the theory of statistical mechanics". Two main approaches can be identified to model and simulate a molecular system, as explained below.

Molecular Dynamics (MD) simulations

Molecular dynamics simulation consists of the numerical solution of the classical equations of motion (Newton's second law) applied to an atomic system:

$$m_i \frac{\mathrm{d}^2}{\mathrm{dt}^2}\mathbf{r}_i = \mathbf{F}_i(\mathbf{r}^N) \,, \quad \mathbf{F}_i = -\frac{\partial}{\partial \mathbf{r}_i} \mathcal{U}(\mathbf{r}^N) \,, \qquad [B.1]$$

where m_i is the mass of particle i, t is the time variable, $\mathbf{r}^N = (\mathbf{r}_1, \mathbf{r}_2, ...\mathbf{r}_N)$ represents the complete set of $3N$ atomic coordinates, F_i is the force that exerts on particle i and $\mathcal{U}$ is the potential energy field from which the forces derive. The use of MD simulation as a technique for computing equilibrium and transport properties of a many-body atomic system is justified by the fact that nuclear motion of the constituent particles can be assumed to obey the laws of classical mechanics. Indeed, in the vast majority of the cases, quantum effects can be neglected as the inter-particle distance can be shown to be much larger than the de-Broglie wavelength, as pointed out by Vojta (n.d.).
The most important point of MD is therefore to be able to define a potential energy function $\mathcal{U}(\mathbf{r}^N)$ that will represent accurately the interatomic interactions. Kühne (2014) explains that the better $\mathcal{U}(\mathbf{r}^N)$ reproduces the actual reality, the more valuable and predictive the eventual simulation will be. There are two different ways to proceed.

Classical MD The traditional route followed in molecular dynamics is to use predefined potentials that are determined in advance, before the simulation. These are based on models constructed by comparison with experiments (empirical potentials). The main advantage of this approach is the low computational cost that permits to consider large particle numbers. It works well for simple atomic systems but can lead to poor results when many atom or molecule types are involved, as mentioned by Marx and Hutter (2000).

Ab initio MD (AIMD) In reality, the interactions between atoms and molecules result from electronic structure involving quantum mechanical effects. Marx and Hutter (2000) explains that "the basic idea underlying every ab initio molecular dynamics method is to compute the forces acting on the nuclei from electronic structure calculations that are performed "on–the–fly" as the molecular dynamics trajectory is generated. In this way, the electronic variables are not integrated out beforehand, but are considered as active degrees of freedom". Density Functional Theory (DFT) is often used to approximate the solution of the many-body Schrödinger equations that need to be solved at each time step.

In this work, classical MD is considered and we will show that accurate results are obtained for simple system (Appendix B.2.1). On the contrary, the accuracy of the results for more complex systems with different atom types is lower (Appendix B.2.2).

Monte Carlo (MC) Simulations

Contrary to classical MD which is fully deterministic, the Monte Carlo method is probabilistic. Instead of using Newtons's law, at each iteration, all particles are given new random positions from the positions they had in the previous configuration. To ensure that the simulation will sample the Boltzmann distribution, these random moves can be accepted or not according to the Metropolis criterion which is based on the potential energy difference of each particle in their new and old configurations (Dror (2017)). The properties are computed from average values that are approximated by generating numerous trial configurations. Doing so, MC simulations can only give static, thermodynamic equilibrium properties while with MD simulations, transport properties can also be computed.

B.1.2 Main elements of a MD simulation

To introduce the different elements constituting any MD simulation, we borrow the approach presented in Frenkel and Smit (2001) under the form of a simple MD program: The few generic routines in this algorithm will now be explained

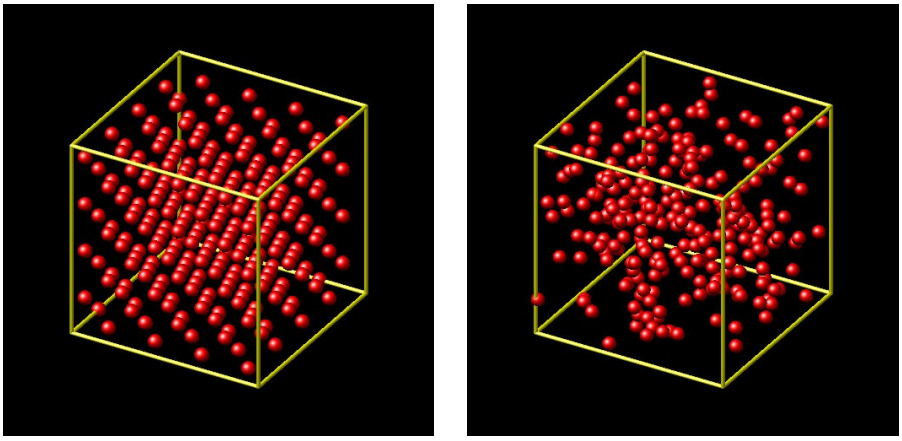
Algorithm 7 A simple molecular dynamics program

```
1: procedure GenericMDS
2:     r_0^N, ṙ_0^N ← init                                  ▷ Initialization
3:     while t < t_max do
4:         F ← force(U(r_{n-1}^N))                  ▷ Compute the forces on every particle
5:         r_n^N, ṙ_n^N ← integrate(F, r_{n-1}^N, ṙ_{n-1}^N)  ▷ Integrate equations of motion
6:         t ← t + Δt
7:         Properties ← sample(r_n^N, ṙ_n^N)                ▷ Sample averages
8:     end while
9: end procedure
```

in more details in the following.

Initialization

To start the simulation, we should assign initial positions and velocities to all particles in the system. The particle positions should be chosen compatible with the structure that we are aiming to simulate. If we want to simulate a solid system, the atoms should be arranged on a lattice structure whose topology corresponds to the one of the material of interest. On the other hand, if we want to start the simulation with a liquid state, the positions of the atoms can be set randomly, as it is represented in Figure B.1. Then we attribute to each velocity component of every particle a value that is drawn from a certain distribution. The magnitudes of these velocities reflect the desired total energy or temperature. This initial state is not the equilibrium state, it will take the



a / Solid – lattice configuration.

b / Liquid – random configuration.

Figure B.1 / Initial configuration of the particles in a MD simulation.

system some time to reach equilibrium from which averages can begin to be collected.

Geometry and boundary conditions

Periodic boundary conditions The number of particles that can be handled in a simulation is much smaller than the one in macroscopic systems (simply in one mole, there are $N_A = 6.022 \cdot 10^{23}$ particles). That is why Frenkel and Smit (2001) insists on the fact that "it is essential to choose boundary conditions that mimic the presence of an infinite bulk surrounding the N-particle model system". This is usually achieved by employing periodic boundary conditions. The volume containing the N particles is treated as the primitive cell of an infinite periodic lattice of identical cells (see Figure B.2). In theory, a given particle (say particle i) now interacts with all other particles in this infinite periodic system, that is, all other particles in the same periodic cell and all particles (including its own periodic image) in all other cells.

Truncation of interactions In practice however, for system with short-range interactions, a cutoff distance r_c is usually defined that determines, for each particle i, all the surrounding particles j that will be taken into account to compute the forces. In the case of r_c is smaller than half the diameter of the periodic box, we need to consider the interaction of a given particle i only with the nearest periodic image of any other particles j (Frenkel and Smit (2001)), represented by the dotted box in Figure B.2. In the particular case where long-range Coulomb or dipolar interactions are important, the interactions with all periodic images should be taken into account explicitly.

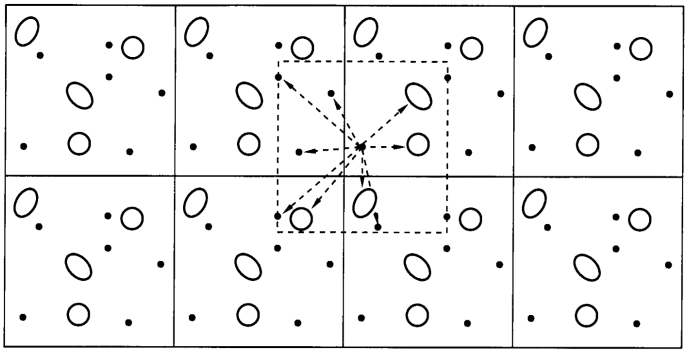


Figure B.2 / Schematic representation of periodic boundary conditions. The dotted box represents all the particles that are considered inside a certain cutoff distance to compute the forces on a given particle. Figure taken from Frenkel and Smit (2001).

Forces and potentials in classical MD

The calculation of the forces acting on every particle is the most time-consuming part of almost all MD simulations. These are directly related to the spatial gradient of the total potential energy function that is made of different terms in case of classical MD. Before describing these different contributions, it is probably important to recall some elementary notions of chemistry to make the explanations more clear in the following.

Atoms, molecules and lattices: Atoms are the most elementary particles composed of nucleus and electrons bound to the nucleus by electromagnetic forces and fixed within electronic orbitals, which are regions in which electrons have the highest probability of being found. Molecules are made up of more than one atom, of the same or different kind, bonded via *intra*-molecular forces. Many materials are composed by an ensemble of molecules which are hold together via *inter*-molecular interactions. It can also be a single giant molecule as in the case of perfect single crystals. There are however some exceptions. As we will see after, pure metals do not really form molecules. Instead, they can be seen as a mixture of atomic cores (positive nuclei + inner shell of electrons) in a sea of valence electrons (LibreTexts (n.d.[c])). A lattice, finally, is an ordered structure, usually of a crystalline solid, that is made up of units that might be molecules or atoms.

Intra-molecular forces: Chemical or intra-molecular bonds are typically classified into three main types (LibreTexts (n.d.[a])):

- Ionic bonds: they come from electrostatic forces that exist between ions of opposite charge. These bonds typically involve a metallic with a nonmetallic element;
- Covalent bonds: they result from the sharing of electrons between two atoms. The bonds typically involve one nonmetallic element with another;
- Metallic bonds: they occur among metal atoms. When they come together, the electrons in some atomic orbitals share space with the corresponding

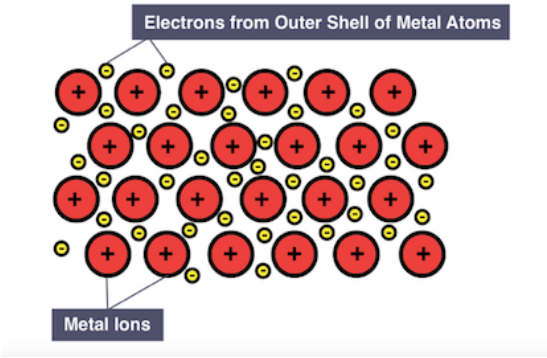


Figure B.3 / Schematic representation of the electron sea model explaining metallic bonding: positive atomic nuclei (red circles) surrounded by a sea of delocalized electrons (yellow circles). Figure taken from SaveMyExams (n.d.).

electrons on neighboring atoms to form molecular orbitals. Since metallic elements contain too few electrons or valence orbitals to form covalent bonds with each of their neighbors, the valence electrons can move freely within these molecular orbitals, and so each electron becomes detached from its parent atom. The electrons are said to be delocalized. The metal is held together by the strong forces of attraction between the positive nuclei and the delocalized electrons (LibreTexts (n.d.[c])). This is schematically represented in Figure B.3. This specificity is at the heart of the Embedded Atom Method (EAM) described after.

Inter-molecular forces: These forces are electrostatic in nature; that is, they arise from the interaction between positively and negatively charged species. Three types of inter-molecular forces can be identified, the two first ones being grouped under the common van der Waals forces denomination (LibreTexts (n.d.[b])):

- van der Waals forces:
  - Dipole-dipole interactions: they arise from the electrostatic interactions of the positive and negative ends of molecules with permanent dipole moments;
  - London dispersion forces: they are due to the formation of instantaneous dipole moments in polar or nonpolar molecules as a result of short-lived fluctuations of electron charge distribution, which in turn cause the temporary formation of an induced dipole in adjacent molecules;
- Hydrogen bonds: they are especially strong dipole–dipole interactions between molecules that have hydrogen bonded to a highly electronegative atom.

Inter-molecular forces are generally weaker than ionic or covalent bonds, so that changes between the different states of matter almost invariably occur without

breaking covalent or ionic bonds.
Gas, liquid and solid: In gas, the kinetic energy of the molecules is greater than any attractive forces between them. In liquids, the intermolecular attractive forces are strong enough to hold molecules close together but not strong enough to keep neighboring molecules in a fixed position. For solids, these same forces are able to keep the molecules locked in position. For the particular case of metals, even if metallic bonds were classified as intra-molecular interactions, their link with phase transition is closer to the one of inter-molecular forces in other substances. Indeed, on melting, metallic bonds are loosened, not broken. So, although the ordered solid structure has been broken down, metallic bonds are still present between the atoms inside a molten metal. Metallic bonds are not fully broken until the metal boils.

Non-bonding potentials: inter-molecular interactions The part of the potential energy $\mathcal{U}_{\text{non-bonded}}$ representing inter-molecular interactions between atoms is traditionally split into 1-body, 2-body, 3-body,... terms:

$$\mathcal{U}_{\text{non-bonded}}(\mathbf{r}^N) = \sum_i u(\mathbf{r}_i) + \sum_i \sum_{j>i} v(\mathbf{r}_i, \mathbf{r}_j) + \dots . \qquad [B.2]$$

The $u(\mathbf{r})$ term represents an externally applied potential field or the effects of the container walls; it is dropped for fully periodic simulations. Also, it is usual to concentrate on the pair potential $v(\mathbf{r}_i, \mathbf{r}_j) = v(r_{ij})$ and neglect three-body (and higher order) interactions. In some simulations of complex fluids, it is sufficient to use the simplest models that faithfully represent the essential physics. These pairwise potentials can be divided in two contributions according to M. P. Allen et al. (2004):

- van der Waals interactions: the most commonly used model is the Lennard-Jones (LJ) potential:

$$v^{LJ}(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] , \qquad [B.3]$$

  where ϵ is the depth of the potential well, σ is the distance at which the potential is zero and r is the distance between the particles. The first term accounts for Pauli repulsion at short-ranges due to overlapping electronic orbitals while the second term models the attraction at longer ranges due to van der Waals forces.
- Long-range electrostatic interactions: if long-range interactions for charged species (ions) are present, a Coulomb potential is usually used:

$$v^{\text{Coulomb}}(r) = \frac{Q_1 Q_2}{4\pi\epsilon_0 r} , \qquad [B.4]$$

  where Q_1, Q_2 are the charges and ϵ_0 the permittivity of free space.

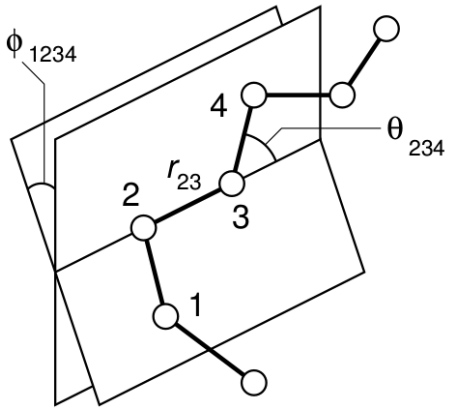


Figure B.4 / Geometry of a simple chain molecule, illustrating the definition of interatomic distance r_{23}, bend angle θ_{234}, and torsion angle (dihedral) ϕ_{1234}. Figure taken from M. P. Allen et al. (2004).

As discussed earlier, for each atom, only non-bonded interactions with neighboring atoms within a certain cutoff distance are calculated. For van der Waals interactions, this can be perfectly valid but for Coulombic ones, it becomes much less accurate.

Bonding potentials: covalent (and ionic) bonds Ideally, we would like to treat all aspects of chemical bonding, including the reactions which form and break bonds, in a proper quantum mechanical fashion. But since we are not considering AIMD in this report, we will only present mathematical molecular models, the most simple one including the following different terms:

$$\begin{aligned} U_{\text{bonded}} = & \frac{1}{2} \sum_{\text{bonds}} k_{ij}^{r} (r_{ij} - r_{eq})^2 \\ & + \frac{1}{2} \sum_{\text{bend angles}} k_{ijk}^{\theta} (\theta_{ijk} - \theta_{eq})^2 \\ & + \frac{1}{2} \sum_{\text{dihedrals}} \sum_{m} k_{ijkl}^{\phi,m} \left(1 + \cos(m\phi_{ijkl} - \gamma_m)\right) \\ & + \frac{1}{2} \sum_{\text{improper}} k^{\phi} \left(\psi - \psi_{eq}\right) \end{aligned} \qquad [B.5]$$

The different terms of this equation are illustrated by Figure B.4. A simulation package force-field will specify the precise form of Equation B.5, and the various strength parameters k and other constants therein. Actually, Equation B.5 is a considerable oversimplification. Molecular mechanics force-fields, aimed at accurately predicting structures and properties, will include many cross-terms (e.g. stretch-bend). Quantum mechanical calculations may give a guide to the

"best" molecular force-field. External libraries such as AMBER, CHARMM or OPLS can provide more accurate force-field.

Embedded Atom Method (EAM): metallic bonds As we have seen, metallic bonds are particular in the sense that they cannot really be classified as intra- or inter-molecular interactions. That is why a specific method was developed to treat them, called the Embedded Atom Method (EAM). A particularly appealing aspect of the EAM is its physical picture of metallic bonding: each atom is considered to be embedded in a host electron gas created by its neighboring atoms. Each atom will therefore contribute with its own embedding energy to the total cohesive energy of the metal. This embedding energy depends on the density of the background electron gas in which the atom is located, which is determined by evaluating at its nucleus the superposition of atomic-density tails from the other atoms. The total potential energy of a metallic system is composed of these embedding energies plus electrostatic interactions and can be expressed as:

$$U_{metal} = \sum_i G_i\left(\rho_{h,i}\right) + \frac{1}{2}\sum_{i,j\ (j\neq i)} v(r_{ij})\ , \qquad [B.6]$$

where:

- G_i is the embedding energy of atom i that is obtained from a common embedding energy function G;
- $\rho_{h,i} = \sum_{j\neq i} \rho_j^a(r_{ij})$ is the total electron density of the host at position r_i for atom i, ρ_j^a is the contribution to this density from the neighboring atom j and r_{ij} is the distance between atoms i and j;
- $v(r_{ij})$ is the short-range electrostatic pair potential.

The embedding function and pair potential can be obtained from first principles by considering the density-functional expression for the cohesive energy. However, Daw et al. (1993) explains that in practice, semi-empirical approaches are usually adopted where these functions are determined by fitting them to basic bulk properties such as lattice constant, cohesive energy, elastic constants, and vacancy formation energy.

Integrators and ensembles

A macroscopic state is totally described by a triplet of the following state variables: the number of particles N, the volume V, the energy E, the chemical potential μ, the pressure P, and the temperature T. A microscopic state of the system is given by a point $(\mathbf{r},\mathbf{p})$ of the phase space of the system, where $\mathbf{r}$ and $\mathbf{p}$ are the positions and the momenta, respectively. A thermodynamic ensemble is a collection of microscopic states that all realize an identical macroscopic state. The three most commonly used ensembles in MD simulations are detailed hereafter and different techniques must be considered for each of them to obtain a correct sampling.

Microcanonical: fixed (N, V, E) This is the ensemble a MD simulation samples naturally. By reformulating the Newton's equation under Hamiltonian's form, it can be shown that the Hamiltonian

$$\mathcal{H}(\mathbf{r}, \mathbf{p}) = \sum_i \frac{p_i^2}{2m_i} + \mathcal{U} \qquad [B.7]$$

is the sum of the kinetic and potential energy of a closed system, and that is constant over time (Maginn (1997)). So the total energy of the system is conserved. In this case, the most simple and efficient algorithm to integrate the equations of motion is the Velvet algorithm in which the basic equations for advancing the positions and compute the velocities (if needed) of a particle i are:

$$\mathbf{r}_i(t + \Delta t) = 2\mathbf{r}_i(t) - \mathbf{r}_i(t - \Delta t) + (\Delta t)^2 \frac{\mathbf{F}_i(t)}{m_i} ,$$
$$\mathbf{v}_i(t + \Delta t) = \frac{\mathbf{r}_i(t + \Delta t) - \mathbf{r}_i(t - \Delta t)}{2\Delta t} ,$$

in which the forces $\mathbf{F}_i(t)$ have been computed before based on the potentials described in the previous section.
It is however usually convenient to perform MD simulations in other ensembles. This requires more complex algorithms that change E and/or V to maintain other defined parameters constant, as it is very briefly discussed in the next two paragraphs. For more information on the topic, the reader is invited to consult Frenkel and Smit (2001).

Canonical: fixed (N, V, T) In order to maintain a constant temperature[1], thermostatting is applied to the system. There are two main algorithms for that:

- Berendsen: the velocities are scaled to obtain an exponential relaxation of the temperature to T;
- Nose-Hoover: an extra 'thermal reservoir' variable is introduced into the dynamical equations and acts as heat bath:

Constant P–T: fixed (N, P, T) In addition to the thermostat, the volume of the system is allowed to fluctuate, and is regulated by barostat algorithms such that the pressure is constant.

[1]When we say "constant", it does not mean that the temperature will be constant during the simulation. It will fluctuate but on average, its value should be very closed to the prescribed one.

B.1.3 Computation of thermophysical properties

Averaging

As we have seen in the previous section, a thermodynamic ensemble is represented by a collection of microscopic states determined by their positions $\mathbf{r}$ and momenta $\mathbf{p}$. These positions and momenta, noted $\mathbf{\Gamma} = (\mathbf{r}, \mathbf{p})$, can be thought of as coordinates in a multidimensional space, called the phase space, in which the thermodynamic ensemble appears to be a hypersurface. For a system of N atoms, this space has $6N$ dimensions. The points in the ensemble (i.e. over the hypersurface) are distributed according to a probability density $\rho(\Gamma, t)$, which depends on the fixed macroscopic parameters (NPT, NVT, etc.). Each point represents a typical microstate, or system, at any particular instant of time. Each system evolves in time, according to the usual mechanical equations of motion, quite independently of the other systems. Consequently, in general, the phase space density $\rho(\Gamma, t)$ will change with time. However, it can be shown by Liouville's theorem that if $\rho(\Gamma, t)$ represents an equilibrium ensemble, it is independent of time: $\rho(\Gamma, t) = \rho_{ens}(\Gamma)$. Then, the phase average, or ensemble average, of any macroscopic quantity Y, for an ensemble represented by a finite number of points, can be computed as

$$\begin{aligned} \langle Y \rangle_{\text{ens}} &= \sum_{\Gamma} Y(\Gamma) \rho_{ens}(\Gamma) \, , \\ &= \frac{\sum_{\Gamma} Y(\Gamma) W_{ens}(\Gamma)}{Q_{ens}} \, , \end{aligned}$$

where W_{ens}, called the weight function, is essentially a non-normalized form of ρ_{ens}, with the partition function Q_{ens} acting as the normalizing factor. The expression of this fundamental function for the three main ensembles is as follows:

- NVE: $Q_{NVE} = \sum_{\Gamma} \delta(\mathcal{H}(\Gamma) - E)$;
- NVT: $Q_{NVT} = \sum_{\Gamma} \exp(-\mathcal{H}(\Gamma)/k_B T)$;
- NPT: $Q_{NPT} = \sum_{\Gamma} \sum_{V} \exp(-(\mathcal{H}(\Gamma) + PV)/k_B T)$;

where k_B is the Boltzmann constant. One conceivable approach to the computation of averaged thermodynamic quantities, therefore, would be a direct evaluation of Q_{ens}, for a particular ensemble. This summation, over all possible microstates, is however not feasible for many-particle systems.
That is why the ergodic hypothesis is invoked to be able to replace the ensemble averages by time averages over the MD simulation:

$$\langle Y \rangle_{\text{ens}} \iff \overline{Y}_{time} = \frac{1}{T} \sum_{t=1}^{N_t} Y(\Gamma(t)) \, . \qquad [\text{B.8}]$$

In other words, it means that averaging over all state points on the hypersurface (Y_{ens}) is equivalent to taking averages over a series of 'snapshots' as the system

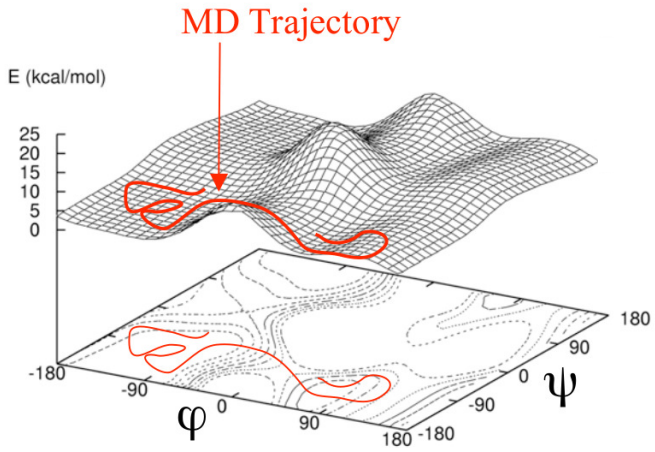


Figure B.5 / Constant energy hypersurface in phase space with a typical trajectory followed by a MD simulation. Figure taken from Cuendet (2008).

evolves with time at equilibrium (Y_{time}), i.e. riding a single trajectory on the hypersurface for sufficiently long, as it is depicted in Figure B.5. This hypothesis will be valid if two main conditions are met:

1. The potential energy should be accurate enough such that the trajectory followed by the MD simulation is closed enough to the "true" hypersurface.
2. The algorithm used to integrate the equations of motion should generate the right ensemble.

Thermodynamic properties

To take the average of an observable quantity in a MD simulation, we must first be able to express this observable as a function of the positions and momenta of the particles in the system (Keffer (2016)).

Number of particles N In the three ensembles described above, the number of particles is held constant and is always an input of the simulation.

Volume V In the NVE and NVT ensembles, the volume of the simulation box is fixed and does not fluctuate with time. In the NPT ensemble, it fluctuates in response to the barostat.

Density ρ The instantaneous density is directly related to the number of particles and the volume via

$$\rho = \frac{m}{V} = \frac{M \cdot N}{N_A \cdot V}, \qquad [B.9]$$

where M is the molar mass of the element considered and N_A the Avogadro number.
The averaged density is simply given by

$$\overline{\rho} = \frac{M \cdot N}{N_A \cdot \overline{V}} \,. \qquad [B.10]$$

Temperature T The temperature is related to the kinetic energy via the equipartition theorem. The kinetic energy is related to the velocities via classical mechanics:

$$\frac{3}{2} N k_B T = \frac{1}{2} \sum_{i=1}^{N} \sum_{\alpha=1}^{3} m_i v_{i,\alpha}^2 \,, \qquad [B.11]$$

where $v_{i,\alpha}$ is the αth-component of the velocity vector of particle i. Unlike the volume, the temperature fluctuates in all three ensembles. The NVT and NPT ensembles, in which the temperature is specified, are controlled by thermostats that do not keep the temperature exactly at the set point temperature at all times. Instead, a proper thermostat allows the temperature to fluctuate about the set point temperature in a manner that is still capable of rigorously generating trajectories that correspond to the ensemble.
The averaged temperature is obtained via

$$\overline{T} = \frac{1}{3 N k_B} \overline{\sum_{i=1}^{N} \sum_{\alpha=1}^{3} m_i v_{i,\alpha}^2} \,. \qquad [B.12]$$

Pressure p In three-dimensions, the pressure tensor is a symmetric 3x3 tensor, with each component defined by a kinetic p_{KE} and potential p_{PE} contribution:

$$p_{\alpha\beta} = p_{KE,\alpha\beta} + p_{PE,\alpha\beta} \,. \qquad [B.13]$$

The kinetic contribution to the $\alpha\beta$ element of the pressure tensor is given by

$$p_{KE,\alpha\beta} = \frac{1}{V} \sum_{i=1}^{N} m_i v_{i,\alpha} v_{i,\beta} \,, \qquad [B.14]$$

while the potential contribution is obtained from the virial equation

$$p_{PE,\alpha\beta} = \frac{1}{V} \sum_{i=1}^{N} r_{i,\alpha} F_{i,\alpha} \,, \qquad [B.15]$$

where $r_{i,\alpha}$ is the αth-coordinate of the position of particle i and $F_{i,\alpha}$ the force that exerts on this particle. The scalar thermodynamic pressure can be computed as

$$p = \frac{1}{3} \sum_{i=1}^{3} p_{\alpha\alpha} = \frac{1}{3V} \sum_{i=1}^{N} \sum_{\alpha=1}^{3} m_i v_{i,\alpha}^2 + \frac{1}{3V} \sum_{i=1}^{N} \sum_{\alpha=1}^{3} r_{i,\alpha} F_{i,\alpha} \,,$$

$$= \frac{Nk_BT}{V} + \frac{1}{3V}\sum_{i=1}^{N}\sum_{\alpha=1}^{3} r_{i,\alpha}F_{i,\alpha}\,,$$

where the kinetic contribution takes the form of the ideal gas law thanks to the equipartition theorem. The potential energy contribution (which is due to the presence of inter-molecular forces) can therefore be seen as a deviation from the ideal gas model (for which inter-molecular forces are absent). Like the temperature, the pressure fluctuates in all three ensembles. The NPT ensemble, in which the pressure is specified, is controlled by barostats that do not keep the pressure exactly at the set point pressure at all times.
The averaged pressure is given by

$$\overline{p} = \frac{Nk_B\overline{T}}{\overline{V}} + \overline{\frac{1}{3V}\sum_{i=1}^{N}\sum_{\alpha=1}^{3} r_{i,\alpha}F_{i,\alpha}}\,, \qquad [B.16]$$

Internal or total energy E The internal or total energy is the sum of the kinetic KE and potential PE energies:

$$E = KE + PE = \frac{1}{2}\sum_{i=1}^{N}\sum_{d=1}^{3} m_i v_{i,d}^2 + \mathcal{U}\,, \qquad [B.17]$$

where the potential energy results from the definition of the interatomic potential $\mathcal{U}$. In the NVE ensemble, the internal energy should be constant, as it is a measure of the conservation of energy of the classical equations of motion. In truth, it fluctuates based on the numerical approximations to the solution of these ordinary differential equations. In the other two ensembles, the internal energy fluctuates.

Fluctuations The computation of secondary thermodynamic quantities that are based on partial derivatives of primary ones are also relevant. The quantities we are interested in are the isobaric and isochoric specific heat capacities, c_p and c_v, the isobaric thermal expansion coefficient α_p and the isothermal compressibility coefficient β_T whose definitions are recalled here for convenience:

$$c_p = \left(\frac{\partial H}{\partial T}\right)_p\,, \quad c_v = \left(\frac{\partial E}{\partial T}\right)_V\,, \quad \alpha_p = -\frac{1}{\rho}\left(\frac{\partial \rho}{\partial T}\right)_p\,, \quad \beta_T = \frac{1}{\rho}\left(\frac{\partial \rho}{\partial p}\right)_T\,. \qquad [B.18]$$

The expressions given above are based on thermodynamic definitions. Statistical mechanics provides an elegant theoretical way to evaluate these properties based on the fluctuations of properties. These fluctuations are based on the variance or covariance of the instantaneous thermodynamic properties discussed in the previous section. For example, in the NVT ensemble, the formula

$$\overline{\delta E^2} = k_B\overline{T^2}\overline{c_v} \qquad [B.19]$$

can be used to compute the averaged isochoric heat capacity, while in the NPT ensemble, the formula

$$\overline{\delta V^2} = \overline{V} k_B \overline{T \beta_T} \qquad [B.20]$$

will be used to obtain the isothermal compressibility.
In practice however, a poor convergence of the quantities might be observed for some fluctuation formula. That is why in this work, we compute the secondary thermodynamic quantities in a post-processing step directly from their definitions, Equation B.18.

Transport properties

In the words of Maginn (1997), "transport coefficients describe the response ('flux') elicited in a system by the imposition of a perturbation ('driving force') that causes the system to depart from equilibrium. Autocorrelation functions describe the rate at which spontaneous fluctuations created within a system at equilibrium die out with time. Linear response theory describes the relationship between autocorrelation functions and transport coefficients in a system not too far from equilibrium" and the Green-Kubo formula establishes the exact mathematical relation between both:

$$\gamma = \int_0^\infty \langle \dot{Y}(t)\dot{Y}(0)\rangle \mathrm{d}t = \int_0^\infty C_{\dot{Y}\dot{Y}}(t)\mathrm{d}t \,, \qquad [B.21]$$

where γ is a transport coefficient and $C_{\dot{Y}\dot{Y}}$ is the autocorrelation function of the time derivative of quantity Y, with Y being the mechanical variable associated with the particular transport property under consideration:

$$C_{\dot{Y}\dot{Y}}(t) = \langle \dot{Y}(t)\dot{Y}(0)\rangle = \frac{1}{T}\sum_{\tau=1}^{N_t} \dot{Y}(\tau)\dot{Y}(\tau + t) \,. \qquad [B.22]$$

Another way to compute dynamic properties is by means of Non-Equilibrium MD (NEMD) simulations that require modifications of the underlying equations of motion and/or boundary conditions of the system. This will not be covered in this report.

Viscosity The Green-Kubo formula for the viscosity is

$$\eta_{\alpha\beta} = \frac{V}{k_B T}\int_0^\infty \langle p_{\alpha\beta}(t) p_{\alpha\beta}(0)\rangle \mathrm{d}t \,, \; \alpha \neq \beta \,, \qquad [B.23]$$

where $p_{\alpha\beta}$ is a non-diagonal component of the instantaneous pressure tensor. The viscosity of the system is the average of $\alpha\beta$-component values:

$$\eta = \frac{1}{3}\left(\eta_{xy} + \eta_{xz} + \eta_{yz}\right) \,. \qquad [B.24]$$

Thermal conductivity The thermal conductivity can be computed in an equilibrium MD simulation with

$$k_\alpha = \frac{V}{k_B T^2} \int_0^\infty \langle j_\alpha(t) j_\alpha(0) \rangle \mathrm{d}t \,, \qquad [B.25]$$

where the heat current along direction α is

$$j_\alpha = \frac{\mathrm{d}}{\mathrm{d}t} \left[\frac{1}{V} \sum_i r_{i,\alpha} (E_i - \overline{E}_i) \right] \,, \qquad [B.26]$$

with the energy per molecule E_i given as the sum of the kinetic and potential energies. The average conductivity for an isotropic system in three-dimensions is obtained by:

$$k = \frac{1}{3} \left(k_x + k_y + k_z \right) \,. \qquad [B.27]$$

Diffusivity The self-diffusion coefficient of a species S in direction α in a multicomponent system can be computed from the velocity autocorrelation:

$$D_{S,\alpha} = \int_0^\infty \langle v_{S,\alpha}(t) v_{S,\alpha}(0) \rangle \mathrm{d}t \,, \qquad [B.28]$$

where $v_{S,\alpha}$ is the instantaneous velocity in direction α averaged over all particles of type S. For an isotropic system, the global diffusion coefficient for species S in the mixture is

$$D_S = \frac{1}{3} \left(D_{S,x} + D_{S,y} + D_{S,z} \right) \,. \qquad [B.29]$$

The self-diffusion coefficient can also be equivalently computed from the mean-squared displacement.

Other properties

Surface tension One approach to estimate the surface tension is to construct a liquid-vapor surface in the simulation and directly calculate the surface tension from the mechanical expression. The surface tension can be expressed as the integrated imbalance of normal and tangential pressures near the interface:

$$\sigma = \int_{z_l}^{z_v} \left[p_N(z) - p_T(z) \right] \mathrm{d}z \,, \qquad [B.30]$$

where z_l and z_v are the arbitrary positions in the bulk liquid and vapor phases respectively. For a plane interface perpendicular to the z axis, tangential and normal pressure components are defined as

$$p_N(z) = p_{zz}(z) \,, \quad p_T(z) = \frac{1}{2} (p_{xx}(z) + p_{yy}(z)) \,. \qquad [B.31]$$

If the film is divided into N_s slabs with the length of L_z/N_s parallel to the $X-Y$ panel, the integral can be transformed to the sum in these slabs (Lv and M. Chen (2011)):

$$\sigma = \sum_{k=1}^{N_s} \sigma(k) \ . \qquad [B.32]$$

B.2 Results of Molecular Dynamics Simulations with LAMMPS

B.2.1 Thermophysical properties of pure copper

The first metallic component to be investigated is the pure copper, which is one of the metals often found in space debris. In the following, we present the results of the computations performed with the MD LAMMPS software and we compare them to some available experimental data.
From a general point of view, we can observe on the plots below that the agreement between the numerical results and the experimental ones is very good in general. The discrepancies that exist must be interpreted by recalling that experiments are difficult to conduct for obtaining some properties and the resulting data might be associated to large uncertainties, which are unfortunately not presented on the previous graphs because not available in the references that were consulted.
Some important points still need to be addressed to have a complete analysis:

- Thermal conductivity: this crucial property is missing for the moment. Different approaches to compute it via MD simulations were tested but none of them produced meaningful results. This is clearly the priority task to achieve.
- Equations of state: a way to find reliable expressions for the equation of state corresponding to our material is to build a map of the pressure and the internal energy as a function of the density and temperature before running the CFD simulations. This can be achieved by running NVT simulations over certain density and temperature ranges, average the pressure and internal energy and then build interpolation polynomials. Preliminary tests were made in this respect and the results, which cannot be considered reliable for the moment, are presented in Figure B.14.
- Uncertainties: the evaluation of the uncertainties for the transport properties, the bulk modulus and the surface tension have to be realized in order to reinforce the results.
- Melting and evaporation: in theory, it should be possible to perform simulations of phase changes like melting and evaporation. This could be very useful as the melting and the boiling temperatures as well as the vapor pressure could be obtained.

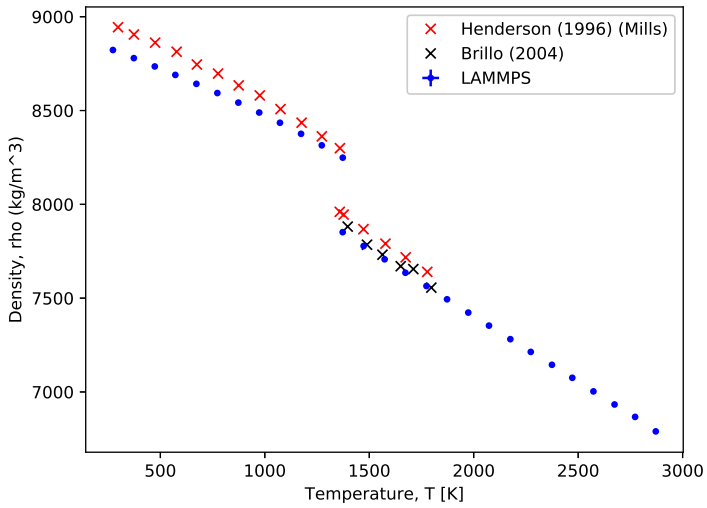


Figure B.6 / Density of copper as a function of temperature obtained with MDS and comparison with experimental results. Error bars for the MD results are drawn but are too small to be visible.

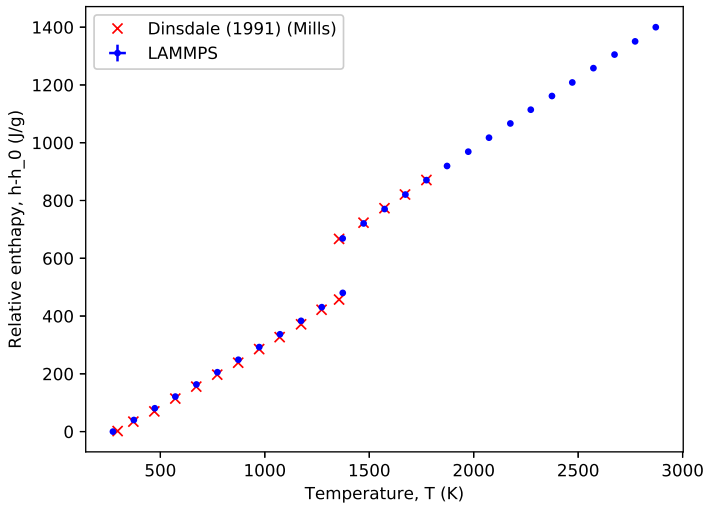


Figure B.7 / Enthalpy of copper as a function of temperature obtained with MDS and comparison with experimental results. Error bars for the MD results are drawn but are too small to be visible.

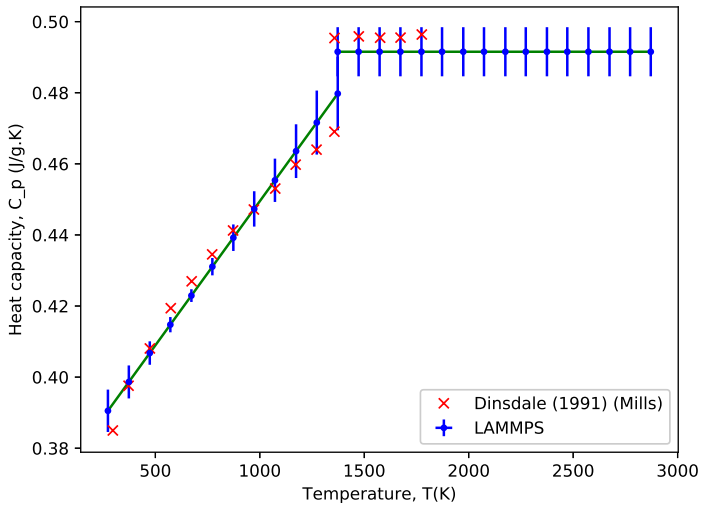


Figure B.8 / Isobaric specific heat of copper as a function of temperature obtained with MDS and comparison with experimental results. Error bars representing 95 % confidence intervals are drawn.

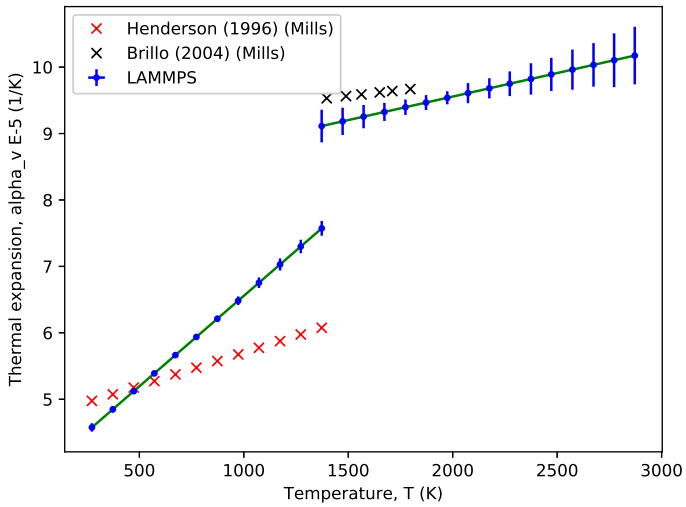


Figure B.9 / Volumetric thermal expansion coefficient of copper as a function of temperature obtained with MDS and comparison with experimental results. Error bars representing 95 % confidence intervals are drawn.

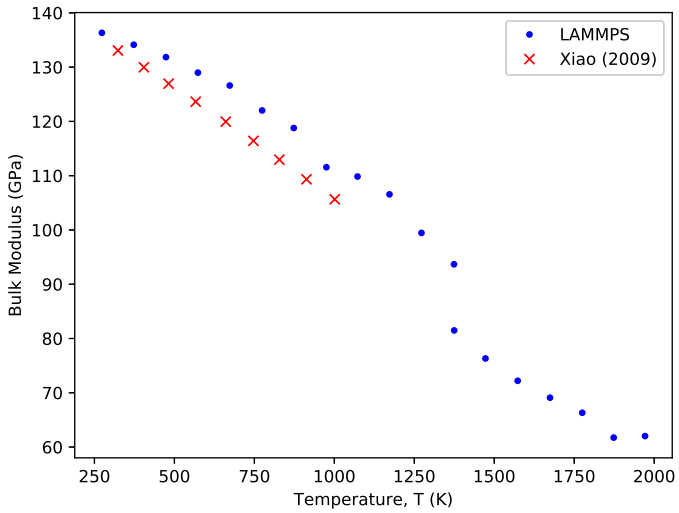


Figure B.10 / Bulk modulus of copper as a function of temperature obtained with MDS and comparison with experimental results.

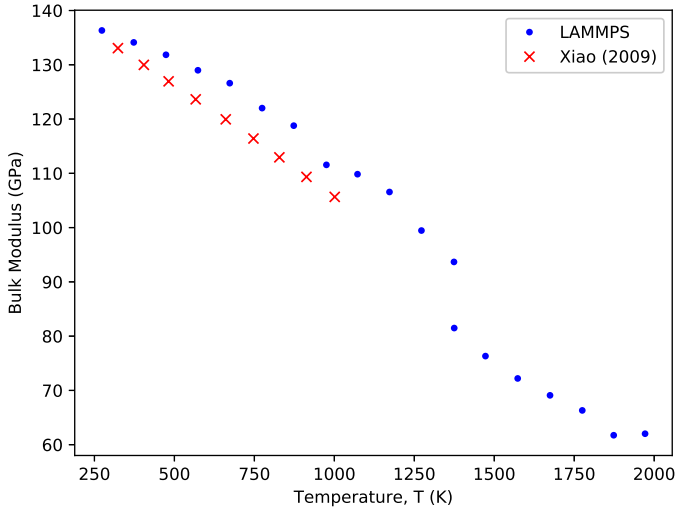


Figure B.11 / Dynamic viscosity of liquid copper as a function of temperature obtained with MDS and comparison with experimental results.

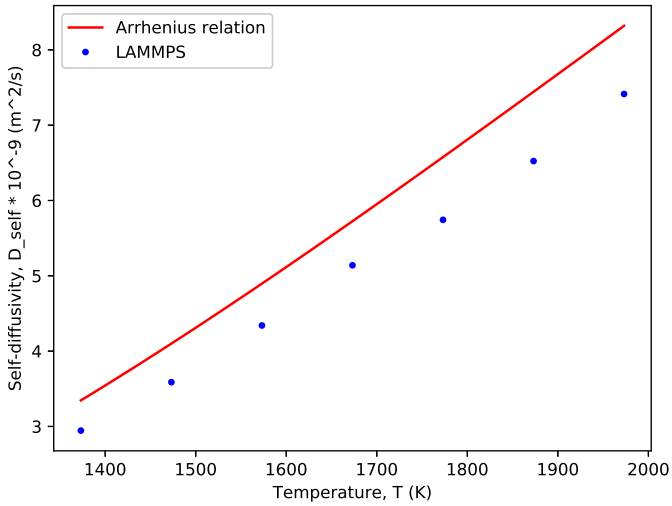


Figure B.12 / Self-diffusion coefficient of liquid copper as a function of temperature obtained with MDS and comparison with empirical relation.

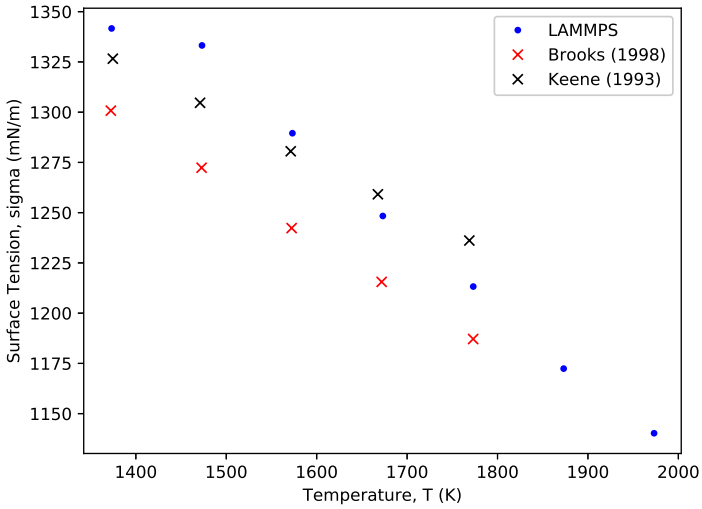


Figure B.13 / Surface tension of liquid copper as a function of temperature obtained with MDS and comparison with experimental results.

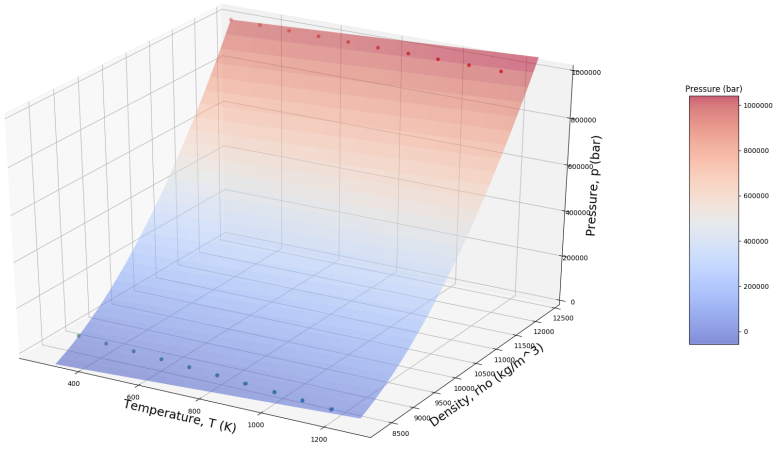


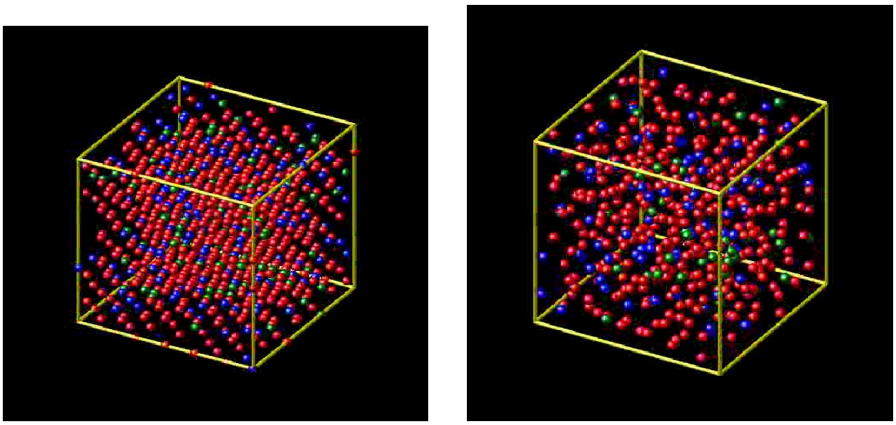
Figure B.14 / *Preliminary results – $p-v-T$* surface of pure copper built from a multivariate quadratic regression on the reference points computed via MDS (the dots).

B.2.2 Thermodynamic properties of an alloy: steel AISI 316L

The steel AISI 316L is one of the main alloys identified as critical for demisability. Its main constitutive elements are Iron Fe, Chromium Cr and Nickel Ni. It contains also some Carbon, Silicon, Manganese, Phosphorus, Sulfur, Molybdenum, Nitrogen and Cobalt. An interatomic potential for the Embedded Atom Method (EAM) taking into account all these constituents is not available. That is why three different potentials for a mixture of $Fe - Cr - Ni$ were taken from the database Standards and Technology (n.d.) to represent the interactions inside this alloy. After having tested all three, the one giving the most satisfactory results was selected.

To initialize the simulation, the concentrations of the three elements present in the mixture are specified and the attribution of the element type to each particle is done randomly. Figure B.15 illustrates this process. From the plots below, we can directly notice that the accuracy of the MD results for this multicomponent material is much lower than for the pure copper, especially for the secondary thermodynamic quantities. This observation is more than likely related to the inability of the simplified potential that was selected to model accurately the intermolecular interactions inside the real complex mixture. The limitations of classical MD simulations appear here as no more refined potential models were found for this alloy, the only remaining possibility being probably to go for Ab Initio MD, which is much more time-consuming and complex.

Concerning possible further investigations, the transport properties, including the diffusion coefficients of the different component, and the surface tension are the first results that will have to be obtained. Then, the same elements as



a / Solid – lattice configuration.

b / Liquid – random configuration.

Figure B.15 / Initial configuration of the particles for the steel 316 alloy. *Fe* atoms in red, *Cr* atoms in blue, and *Ni* atoms in green..

those mentioned for the pure copper will need be treated.

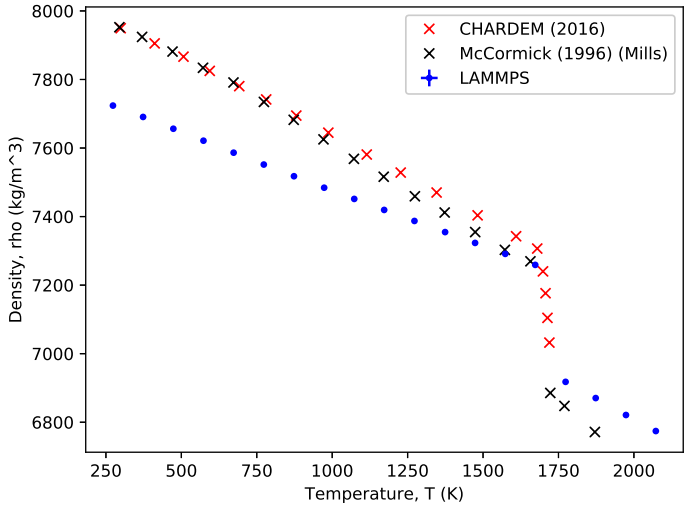


Figure B.16 / Density of steel 316 as a function of temperature obtained with MDS and comparison with experimental results. Error bars for the MD results are drawn but are too small to be visible.

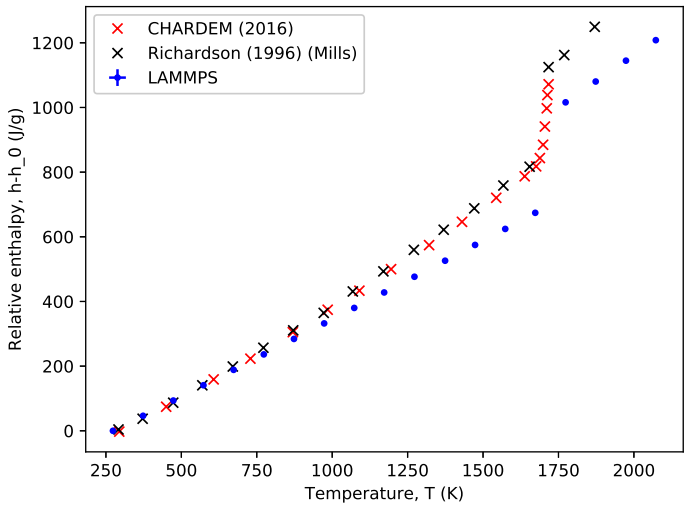


Figure B.17 / Enthalpy of steel 316 as a function of temperature obtained with MDS and comparison with experimental results. Error bars for the MD results are drawn but are too small to be visible.

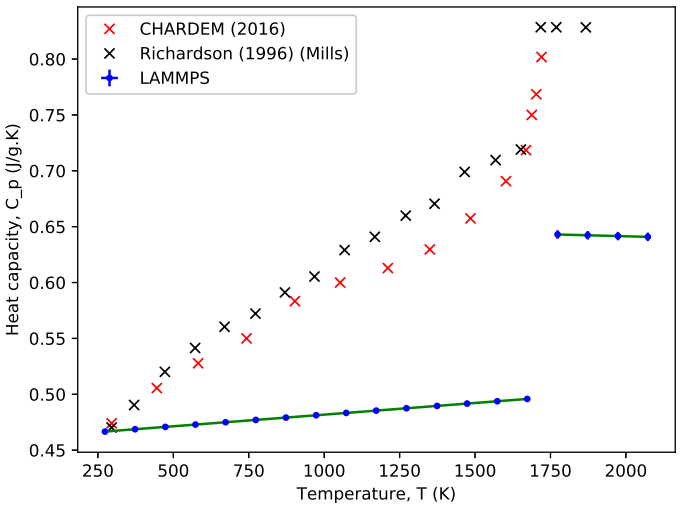


Figure B.18 / Isobaric specific heat of steel 316 as a function of temperature obtained with MDS and comparison with experimental results. Error bars representing 95 % confidence intervals are drawn.

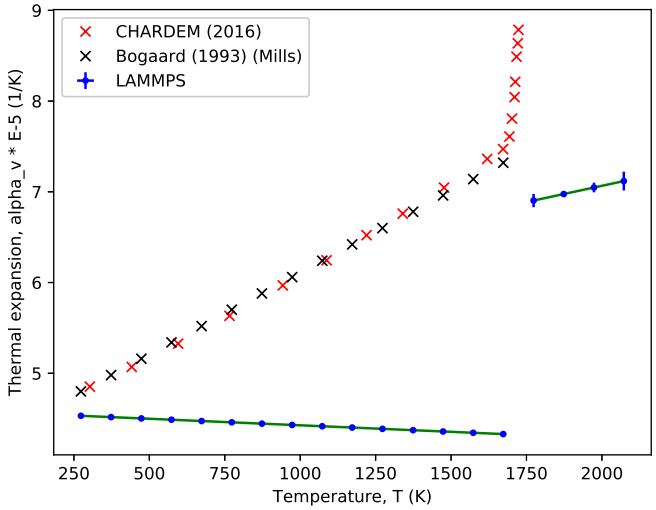


Figure B.19 / Volumetric thermal expansion coefficient of steel 316 as a function of temperature obtained with MDS and comparison with experimental results. Error bars representing 95 % confidence intervals are drawn.

Rest of this page intentionally left blank.

C

A general methodology for generating manufactured solutions satisfying prescribed conditions

The material of this appendix was submitted as it is in Henneaux et al. (2023b) and presented in Henneaux et al. (2023e). It presents the generation of reference solutions used in Chapter 11.

C.1 Introduction

Scientific computing plays an ever-growing role in the prediction of natural and engineering systems. In response to the demands of simulations encompassing increasing levels of fidelity, the complexity of the models, the numerical methods and the algorithms in computational tools has risen significantly. Along with this software sophistication, it is becoming more and more challenging, yet essential, to design strategies assessing the implementation and the suitability of the physical models in order to demonstrate the predictive capabilities of the numerical tool. These assessment strategies fall into two complementary categories: verification and validation. Validation evaluates the accuracy of the physical models implemented in the code for a given application. It is typically performed through comparison with experimental data and answers the

question "are we solving the right equations?". On the other hand, verification is a purely mathematical enterprise, which attempts to resolve the question "are we solving the equations right?". It is commonly further divided into code verification and solution verification. Solution verification estimates the numerical errors incurred by the discretization of the continuous model while code verification has for purpose to test that the numerical algorithms in the code are implemented correctly and that they are functioning as intended.

The present work focuses on code verification, which is the first mandatory action to be conducted in the verification and validation (V&V) process. The books of Patrick J Roache (1998), Knupp and Salari (2002) and Oberkampf and Roy (2010) provide a comprehensive discussion about V&V. We propose to extend the capabilities of the most widely used code verification approach, namely the Method of Manufactured Solutions (MMS). This powerful and elegant method was first proposed in Steinberg and Patrick J Roache (1985) and has since then been applied to many computational physics codes; a selection of them can be found in Patrick J. Roache (2019) and Freno et al. (2022). The MMS is a general procedure to construct arbitrarily complex exact analytical solutions to the governing equations. This reference solution can then be exploited to perform convergence and order of accuracy tests. Formal order verification not only allows to detect coding mistakes, it can also identify glitches in the implemented formulations and algorithms. Most of the time, the manufactured solution (MS) lacks physical realism and this is not a concern since verification is strictly a mathematical exercise. An artificial source term is added to the original interior equations such that the MS is the exact solution of the transformed system inside the domain. In Folkner et al. (2014), a boundary residual is considered in addition to the interior residual so that it is possible to add the MMS source term to both the interior and boundary equations. This approach necessitates a discretization method that employs boundary residual, which limits its generality. That is why in general, the auxiliary conditions like initial and boundary conditions (BC) are often considered *after* the solution to the interior equations has been manufactured. And it is precisely this flexibility that makes the MMS so attractive. However, in some situations, it might be required to incorporate *during* the solution design particular constraints dictated by the governing equations at hand. One important scenario has to do with the verification of the imposition of specialized BC in the software, as pointed out in R. B. Bond et al. (2007). In this case, it is convenient to build the solution such that it complies with the physical/mathematical conditions enforced on a portion of the domain boundaries. Several previous work have addressed this aspect. In R. B. Bond et al. (2007) and Veluri et al. (2012), the baseline MS is multiplied by a function involving the equation of the concerned boundary geometry so that the solution or its gradient evaluated on the boundary is equal to the value corresponding to the exact physical BC to be tested. The same approach has been applied and extended in Choudhary et al. (2016) to different BCs commonly used in compressible and incompressible CFD simulations on general curved boundaries, with a focus on identifying mathematically

well-posed boundary conditions formulations. Although flexible thanks to the possibility of considering complex boundary geometries, this strategy is mostly applicable to classical wall BC. When it comes to the prescription of conditions on boundaries with more complex models or across discontinuities like shocks or material interfaces, a more suitable methodology seems desirable. The idea is to define the MS with some unknown free parameters that are determined based on the set of conditions. Some examples are given here. In Eça et al. (2012), a MS consistent with the employed turbulence models, both in the interior of the domain and on the boundaries, is proposed. A physically-realistic MS is derived in Freno et al. (2021) and Freno et al. (2022) for moving walls with ablating boundary conditions. In Crockett et al. (2011) and Brady et al. (2012), the parameters of the MS are computed such as to respect the interface jump conditions associated to scalar two-phase equations. Finally, a least-squares regression to generate MS obeying the Rankine-Hugoniot jumps relations on curved shocks for the Euler equations is discussed in Grier et al. (2015). Those examples illustrate the case-by-case handling of MS assembly for complex models.

The aim of this work is to establish a generic numerical framework to build MS satisfying arbitrary complex conditions. It relies on the *Sympy Python* library for symbolic manipulations. It has been designed as a versatile tool that can accommodate any new governing equations and conditions. As such, it is proposed to the community as an open-source software and can be considered as an easy-to-use tool that complements the already available ones, like the *MASA* library of Malaya et al. (2013). The free parameters of the user-defined MS are computed with the available *Sympy* system of equations solver while the MMS source terms are automatically generated using symbolic differentiation. We name this new tool *PyManufSol* and it is available at `https://github.com/henneauxd/PyManufSol` (Henneaux (2023)).

This chapter is structured as follows. The theoretical foundations are first given in Appendix C.2, highlighting the various steps in the whole code verification process and discussing the limitations of the current proposed approach. In connection with this description, the structure and working principle of the numerical tool are detailed in Appendix C.3. Its capabilities are then illustrated in Appendix C.4 on two challenging test cases. Appendix C.5 concludes the work and provides some perspectives.

C.2 Building manufactured solutions satisfying prescribed conditions

C.2.1 General manufacturing and verification procedure

We start by presenting the whole manufacturing and verification procedure. Suppose the solver we want to verify handles an initial-boundary value model problem of the form: find $\mathbf{u}\colon \Omega \to \mathcal{U}$, where Ω is the physical domain expressed in terms of the spatial coordinates vector $\mathbf{x}$ and the scalar temporal coordinate t, such that

$$\begin{cases} \mathbf{F}(\mathbf{u}(\mathbf{x},t)) = 0\,, & \forall \mathbf{x} \in \Omega(\mathbf{x},t),\ \forall t > 0\,, \\ \mathbf{B}(\mathbf{u}(\mathbf{x},t)) = 0\,, & \forall \mathbf{x} \in \partial\Omega(\mathbf{x},t),\ \forall t > 0\,, \\ \mathbf{I}(\mathbf{u}(\mathbf{x},t)) = 0\,, & \forall \mathbf{x} \in \Omega(\mathbf{x},t),\ t = 0\,, \end{cases} \qquad [\text{C.1}]$$

where $\mathbf{F}\colon \mathcal{U} \to \mathcal{V}$ is a differential operator mapping elements of vector space $\mathcal{U}$ to the vector space $\mathcal{V}$ and representing the implemented equations in the interior domain Ω, $\mathbf{B}\colon \mathcal{U} \to \mathcal{W}$ is an operator gathering all the conditions on the domain boundaries $\partial\Omega$, and $\mathbf{I}\colon \mathcal{U} \to \mathcal{V}$ is the operator for the initial conditions. Let us remark that in the most general case, the domain can be time-dependent.

Manufacturing

The solution and source term manufacturing steps include:

1. the selection of an arbitrary solution expression, $\widetilde{\mathbf{u}}$, that may depend, in addition to the spatial and temporal coordinates, on a set of N_p free unknown parameters, noted $\boldsymbol{\lambda} = \{\lambda_1, \dots, \lambda_{N_p}\}$:

$$\widetilde{\mathbf{u}} = \widetilde{\mathbf{u}}(\mathbf{x}, t, \boldsymbol{\lambda})\,. \qquad [\text{C.2}]$$

   The choice of the expression does not need to bear any resemblance to the one of the exact physical solution $\mathbf{u}$ of the system Equation C.1 since this latter is most of the time unknown and impossible to derive analytically. Both the spatial and temporal coordinates, $\mathbf{x}$ and t, as well as the free parameters are considered as symbolic variables in the computational framework.
2. The definition of the interior governing equations implemented in the software to be verified, i.e. the operator $\mathbf{F}$ in Equation C.1. The derivatives that appears in the equations are also handled in a symbolic way.
3. The definition of a set of N_c constraints/conditions that the manufactured

solution $\widetilde{\mathbf{u}}$ will need to comply with and which is noted as

$$\mathbf{C}(\widetilde{\mathbf{u}}(\mathbf{x}, t, \boldsymbol{\lambda})) = \begin{Bmatrix} C_1(\widetilde{\mathbf{u}}(\mathbf{x}, t, \boldsymbol{\lambda})) = 0 \\ \vdots \\ C_{N_c}(\widetilde{\mathbf{u}}(\mathbf{x}, t, \boldsymbol{\lambda})) = 0 \end{Bmatrix} . \tag{C.3}$$

Those constraints can involve the manufactured solution itself and its derivatives and can be freely defined by the user. However, in most of the cases, the constraints will correspond to a subset or the entire set of the boundary conditions $\mathbf{B}$.

4. The computation of the unknown parameters $\boldsymbol{\lambda}$ from the set of constraints Equation C.3, i.e.

$$\text{find} \quad \boldsymbol{\lambda}^*(\mathbf{x}, t) \quad \text{s.t.} \quad \mathbf{C}(\widetilde{\mathbf{u}}(\mathbf{x}, t, \boldsymbol{\lambda}^*)) = \mathbf{0} . \tag{C.4}$$

If the constraints have been defined taking into account some of the original problem boundary conditions, this step will reconcile $\widetilde{\mathbf{u}}$ with some constraints imposed by the physical model. In the most general case, the system Equation C.3 is a non-linear system of equations that is solved symbolically with appropriate functions provided by the *Sympy* library. Some important remarks about the construction and the solution of this system are given in Appendix C.2.2.

5. The optional change of spatial coordinate system. To simplify the resolution of Equation C.3, it is sometimes advisable to define $\widetilde{\mathbf{u}}$ as being dependent of a single spatial coordinate while the constraints $\mathbf{C}$ should be defined on boundaries aligned with one of the coordinates (this is discussed in Appendix C.2.2). In order to make the resulting solution more complex such that it checks all the terms involved in $\mathbf{F}$, the original spatial coordinates $\mathbf{x}$ can be translated to a new reference system to become $\overline{\mathbf{x}}$:

$$\widetilde{\mathbf{u}}(\mathbf{x}, t, \boldsymbol{\lambda}^*) \Rightarrow \widetilde{\mathbf{u}}(\overline{\mathbf{x}}, t, \boldsymbol{\lambda}^*(\overline{\mathbf{x}}, t)) . \tag{C.5}$$

For the constraints to still hold after this transformation, the boundaries on which they have been defined should also be transformed in the new coordinate system for the verification step.

6. The assembly of the MMS source term. Because $\widetilde{\mathbf{u}}$ will in general not be the exact solution of the interior equations, inserting the generated manufactured solution into the previously defined operator $\mathbf{F}$ gives rise to an artificial residual:

$$\mathbf{F}(\widetilde{\mathbf{u}}(\overline{\mathbf{x}}, t, \boldsymbol{\lambda}^*)) = \mathbf{S}(\overline{\mathbf{x}}, t) , \tag{C.6}$$

where $\mathbf{S}$ is referred to as the MMS source term. It is automatically generated by passing the manufactured solution which is now fully defined (the $\boldsymbol{\lambda}^*$ parameters have been computed before) to the specified operator $\mathbf{F}$ thanks to symbolic manipulations.

Code verification

After having generated the MS and the MMS source term in the proposed symbolic software, the verification of the code can be undertaken via the following steps:

1. if not already available, the implementation into the code of a source term discretization. Indeed, the original problem we want to solve is modified according to Equation C.6. In other words, we seek the solution of the interior equations determined by a new operator $\tilde{\mathbf{F}}$:

$$\tilde{\mathbf{F}}(\widetilde{\mathbf{u}}) = \mathbf{F}(\widetilde{\mathbf{u}}) - \mathbf{S} = \mathbf{0}\,. \tag{C.7}$$

   It is worth mentioning that some alternative non-intrusive approaches that do not require the addition of a source term to the governing equations have been developed by Brglez (2014) and Freno et al. (2022). They require however the modification of the physical parameters involved in the equations.
2. The splitting of the boundary conditions between the ones to be verified and the "artificial" ones. It must be recalled that the main interest of the present methodology is to verify, in addition to the interior equations, the implementation of some specialized boundary conditions used in the code. It is therefore necessary to let those boundary conditions as such to be able to compute the errors made by the code during their imposition based on the exact reference value that is known if the boundary condition considered is part of the constraints, Equation C.3. On the other hand, the other remaining boundary conditions can simply be specified as Dirichlet conditions with the imposed value being equal to the value of the MS evaluated on the boundary. The modified boundary operator can then be written as

$$\begin{aligned} \widetilde{B}_i(\widetilde{\mathbf{u}}) &= B_i(\widetilde{\mathbf{u}}) \qquad \text{if} \quad B_i \in \mathbf{C}\,, \\ \widetilde{B}_i(\widetilde{\mathbf{u}}) &= \widetilde{\mathbf{u}} - \widetilde{\mathbf{u}}(\mathbf{x}_{\mathrm{BC}}, t) \quad \text{if} \quad B_i \notin \mathbf{C}\,, \end{aligned} \tag{C.8}$$

   where $\mathbf{x}_{\mathrm{BC}}$ identifies the evaluation of the MS at the corresponding boundary location.
3. The specification of the initial conditions depending on the problem type. In general, the initial solution will simply be set as the MS evaluated at $t = 0$,

$$\tilde{\mathbf{I}}(\widetilde{\mathbf{u}}) = \widetilde{\mathbf{u}} - \widetilde{\mathbf{u}}(\mathbf{x}, t = 0)\,. \tag{C.9}$$

   For time-independent problem solved by an iterative steady-state procedure, it may be useful to slightly perturb the initial condition given by Equation C.9 in order to inspect the convergence behavior of the solver.
4. The discretization of the continuous problem. The numerical discretization of the modified continuous problem composed by the modified

operators $\tilde{\mathbf{F}}$, $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{I}}$ described above writes: find $\widetilde{\mathbf{u}}^h : \Omega^h \to \mathcal{U}^h$ such that

$$\begin{cases} \mathbf{F}(\widetilde{\mathbf{u}}^h(\mathbf{x},t)) = \mathbf{S}(\mathbf{x},t)\,, & \forall \mathbf{x} \in \Omega^h(\mathbf{x},t),\ \forall t > 0\,, \\ \widetilde{\mathbf{B}}(\widetilde{\mathbf{u}}^h(\mathbf{x},t)) = 0\,, & \forall \mathbf{x} \in \partial\Omega^h(\mathbf{x},t),\ \forall t > 0\,, \\ \tilde{\mathbf{I}}(\widetilde{\mathbf{u}}^h(\mathbf{x},t)) = 0\,, & \forall \mathbf{x} \in \Omega^h(\mathbf{x},t),\ t = 0\,, \end{cases} \qquad [C.10]$$

where Ω^h represents the geometrical discretization and $\mathcal{U}^h$ is the discrete space in which the approximate solution $\widetilde{\mathbf{u}}^h$ is searched.

5. The order of accuracy test. Salari and Knupp (2000) points out that among the different criteria that can be used to perform code verification (code-to-code comparisons, discretization error quantification, convergence/consistency tests), the order of accuracy test is the most rigorous one. It evaluates if the discretization errors are reduced at the expected theoretical rate when refining the geometry discretization and/or the time step size. Any implementation or conceptual mistakes should prevent that rate to be achieved and indicate the presence of an error. If *a priori* bounds of the form

$$\|\widetilde{\mathbf{u}} - \widetilde{\mathbf{u}}^h\| \leq C\, h^p \qquad [C.11]$$

are available, for a constant C, the theoretical order of convergence p and the mesh spacing parameter or time step h, a systematic convergence study can be conducted for various levels of uniform h-refinement by evaluating the error in the appropriate norm between the exact and approximate MS.

C.2.2 Limitations concerning the constraints definition and the MS free parameters computation

This subsection provides more details about the steps 3 and 4 of the manufacturing process described in Appendix C.2.1 that are specific to the current approach compared to the usual MMS strategy. Although theoretically applicable to a wide range of constraints and solutions, the approach currently implemented in our software has some limitations regarding the steps 3 and 4:

- to be able to solve exactly the system Equation C.3, the number of free parameters introduced in the MS must be equal to the number of constraints specified on the MS,

$$N_p = N_c\,. \qquad [C.12]$$

- If Neumann or Robin conditions are included in the set of constraints, we need to ensure that the free parameters appearing in those constraints will not depend on the spatial coordinates $\mathbf{x}$. Indeed, when solving Equation C.3 at step 4, we implicitly assume that all spatial derivatives of the

free parameters are zero, i.e.

$$\nabla_{\boldsymbol{x}} \boldsymbol{\lambda}^* = \mathbf{0} \,. \tag{C.13}$$

Without that assumption, the non-linear algebraic system would be transformed to a non-linear differential algebraic system of equations (DAE), which appears to be very complicated to solve with complex MS expressions. A good practice to ensure that condition is to (i) choose the expression of the MS to be dependent on a single spatial coordinate, (ii) impose the constraints on boundaries which are oriented parallel to the other spatial coordinate direction. Doing so, it will be possible to substitute the constant numerical value defining the boundary equation into the MS gradient expression before solving the system, hence removing all spatial coordinates dependencies.

- Related to the previous remark, it is at the moment not possible to consider arbitrary boundary geometries. However, as explained in the manufacturing step 5, there is the possibility to operate a change of coordinates after having determined the MS, which allows rotating and translate a boundary, or even to transform it to a curve by going e.g. from a polar to Cartesian reference frame.

C.3 Description of the manufacturing solution software

An overview of the structure of the *PyManufSol* tool is given in Figure C.1 and the links between its constituent modules and the manufacturing steps described in Appendix C.2.1 are established below:

- *ManufSolution*: the user starts by defining the spatial coordinates $\mathbf{x}$, the temporal coordinate t and the set of free parameters $\boldsymbol{\lambda}$ as symbolic variables (using the `symbols` function of *Sympy*). The user then chooses expressions depending on those symbolic variables for each of the field f of the solution vector $\mathbf{u}$. The tool has been designed with the will to treat not only single phase situations but also two-phase configurations. That is why it is possible to assemble a solution vector for each phase k considered in the system.
- *PhysicalModels*: the MS is then fed to the second module which handles the physical models, i.e. the interior governing equations represented by the operator $\mathbf{F}$. Each phase will have its own model. The type of reference coordinate system considered is also specified. It must be noted that this module can easily be extended with new models by overriding the functions defining the convective and diffusive fluxes.
- *BoundaryConditions*: the third module might not be necessary if the assembled MS does not contain any unknown parameters. In this case, the

fourth module can directly be used. If there are some free parameters to determine, a set of constraints/ boundary conditions need to be compiled by the user. First, the geometry of each boundary on which a constraint is enforced is stipulated. The user has then the choice between two main boundary conditions types, i.e. Dirichlet and Neumann. The first kind only involves the solution vector and other quantities that can be obtained from it (e.g. the density from the pressure and temperature). The second kind of BC requires quantities depending on the spatial gradients of the solution (e.g. the heat flux). The physical model **F** is used at this stage to express the quantities required in the constraint definition. Let us remark that a constraint can be a condition concerning the MS of a specific phase, $\widetilde{\mathbf{u}}_k$, or a condition connecting the MS in both phases, which is useful for interface jump conditions (as it will be illustrated in Appendix C.4) but which could also be applied to the case of shock discontinuities for example.

- *ProblemSolving*: the fourth module has a central role as it takes as inputs the physical model and the set of constraints. From there, it can be used to straightforwardly compute the free parameters (by solving Equation C.3 calling the `solvers.solve` *Sympy* method) and possibly change the initial coordinate system. Functions to print in readable format files the final expressions of the MS and the source terms in all phases are also available. Those files can then be passed to the code to be verified which only needs to be able to extract the expressions contained in them and replace the (x, y, t) variables by the appropriate numerical values.
- *Outputs*: there is a last module dealing with the visual representation of the MS. One-dimensional plots along user-specified directions or two-dimensional plots over a predefined zone are possible. This can be useful to evaluate the suitability of the MS before using them in the verification process. Examples of typical outputs are provided in Appendix C.4.

C.4 Application examples

To illustrate the capabilities of the software, we consider here two test cases about the complex situation of a two-phase flow system governed by the compressible Navier-Stokes equations. We exploit the proposed procedure to build MS complying with the physical interface jump conditions, described in Section 3.4. Applying the same procedure on standard boundaries in a single-phase scenario or on other governing equations is also possible. For the interested reader, a series of example test cases featuring different situations is available online along with the source code of our software.
The Navier-Stokes model, Equation 2.51, as well as the equations of state, Equation 2.53, are implemented as a derived class in the *PhysicalModels* module of our software. In the two test cases that follow, we are only interested to verify

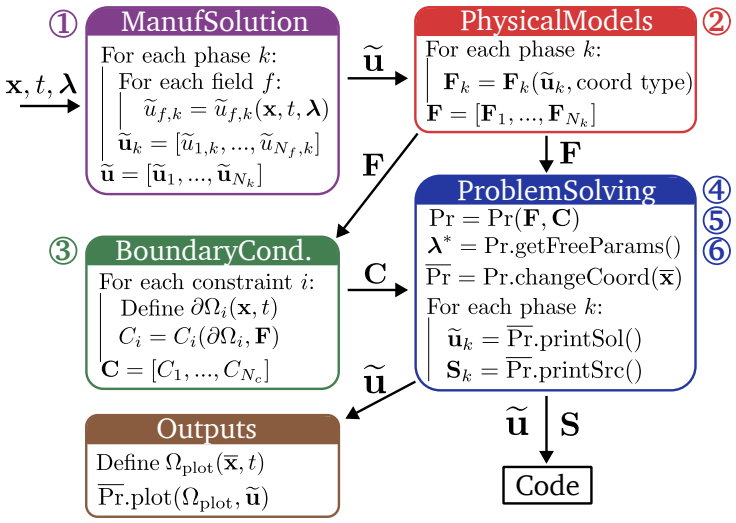


Figure C.1 / Modules constituting the *pyManufSol* numerical framework with their main functionalities, the way they are related and their link with the manufacturing steps described in Appendix C.2.1 (via the circled numbers).

the most challenging aspect our CFD solver has to deal with, namely the enforcement of the interface jump conditions (and not the classical wall boundary conditions). Those are specified in our manufacturing software via the routines implemented in the *BoundaryConditions* module. We recall from Table 9.1 that in practice, the normal momentum jump is split in two contributions such that the second equation in Equation 3.40 becomes:

$$\begin{aligned} [\![p]\!] &= \Delta_\Gamma p - \dot{m}_\Gamma \, [\![v_n]\!] \,, \\ [\![\tau_{nn}]\!] &= 0 \,. \end{aligned} \qquad [C.14]$$

This means that the two last equations in the system Equation 3.40 and the second equation in Equation C.14 are added as Neumann interface conditions using the normal and tangent vectors computed from the provided interface geometry. The jumps in Equation 3.32, Equation 3.33 and Equation 3.34 and the first equation in Equation C.14 are considered as Dirichlet interface conditions. The values chosen for the bulk properties and the jumps for both test cases are listed in Table C.1. Those two test cases are closely related to the inclined interface test case and the spinning bubble test case treated in Section 11.3, with the important difference that the inclined interface test case here is considered with a moving interface.

Table C.1 / Physical parameters selected for the test cases of Appendix C.4.1 and Appendix C.4.2..

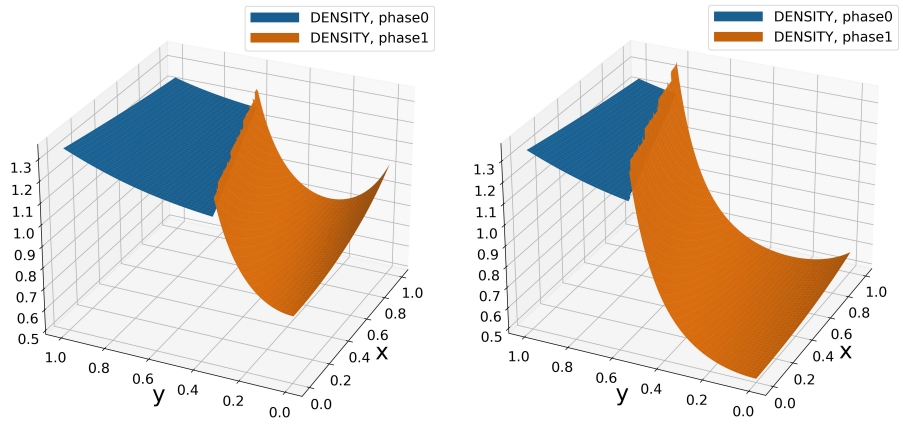
	Inclined interface	Circular interface

Bulk properties	γ_1 , γ_2	1.4 , 1.6	1.911 , 1.290
	$C_{v,1}$, $C_{v,2}$	2.5 , 2.0	977.704, 991.20
	$p_{\infty,1}$, $p_{\infty,2}$	0.0 , 0.0	7.16×10^9 , 0.0
	μ_1 , μ_2	1.0 , 0.25	4.45e-3 , 6.98e-5
	λ_1 , λ_2	1.0 , 0.25	35.24 , 0.127
Jumps values	$\dot{m}_\Gamma$	0.0	-0.001
	Δp	-0.2	200.0
	$\Delta\tau_{nt}$	0.0	0.2
	Δq	1.0	1000.0
	ΔT	0.5	100.0
	Δv_t	0.1	-50.0

C.4.1 Test case 1 – Unsteady inclined interface

The two-dimensional domain considered for this test case is $\Omega = [0,1] \times [0,1]$. The MS determination is realized on a preliminary configuration with a horizontal interface initially located at $y_\Gamma(t=0) = 1/3$ using (x,y) Cartesian coordinates. Phase $k=1$ is situated above the interface and phase $k=2$ below. We are seeking time-dependent manufactured solutions as the motion of the interface is prescribed by a vertical constant normal velocity of $1\,m/s$, noted $v_{\Gamma,n}$. Because this problem involves Neumann-type conditions, we choose the initial forms of the MS to be solely dependent on the y-coordinate following the remarks formulated in Appendix C.2.2. We manufacture the solutions for both phases in terms of the primitive "p-$\mathbf{v}$-T" variables set so that:

$$
\begin{aligned}
\tilde{p}_1(y) &= \cos(3\,\pi\,y/4) + 2.45\,, \\
\tilde{v}_{x,1}(y) &= 0.0\,, \\
\tilde{v}_{y,1}(y) &= v_{\Gamma,n}\,, \\
\tilde{T}_1(y) &= \cos(3\,\pi\,y/4) + 2.0\,, \\
\tilde{p}_2(y) &= \lambda_{p,1} + \cos(5\,\pi\,y/4)\,, \\
\tilde{v}_{x,2}(y) &= \lambda_{v_x,1}\cos(5\,\pi\,y/4) + \lambda_{v_x,2}\,, \\
\tilde{v}_{y,2}(y) &= \lambda_{v_y,1}\cos(5\,\pi\,y/4) + \lambda_{v_y,2}\,, \\
\tilde{T}_2(y) &= 0.1\,\lambda_{T,1}\cos(5\,\pi\,y/4) + \lambda_{T,2}\,,
\end{aligned}
\qquad [C.15]
$$



a / Time $t = 0.0$.

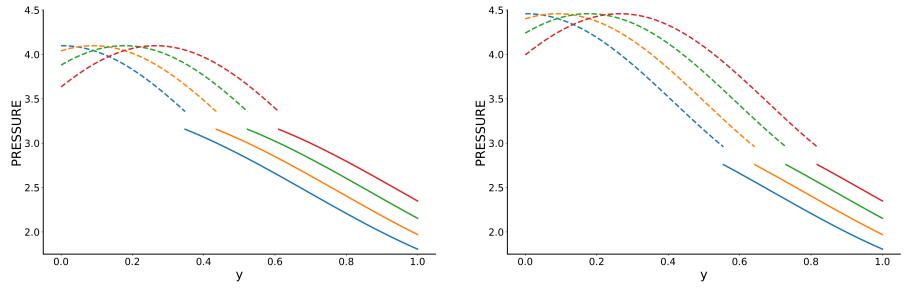
b / Time $t = 0.2$.

Figure C.2 / Surface plot of the density field over the two phases for the inclined interface test case.

with the set of unknown free parameters independent of the spatial coordinates but dependent on time t,

$$\boldsymbol{\lambda}(t) = \{\lambda_{p,1}, \lambda_{v_x,1}, \lambda_{v_x,2}, \lambda_{v_y,1}, \lambda_{v_y,2}, \lambda_{T,1}, \lambda_{T,2}\} \,,$$

and comprising seven components, which corresponds to the number of constraints to be satisfied. The jump conditions are here imposed on an interface whose position depends on the temporal symbolic variable t. This is possible to prescribe such motion via the module *BoundaryConditions*. The parameters are computed from the available routines in the *ProblemSolving* module after having assembled the constraints based on the jump conditions described earlier. The MS are then rotated by 15° in the anti-clockwise direction by means of a change of reference system, which is also provided by *ProblemSolving*. The MS is now dependent on both the x and y coordinates, in addition to t, and can exercise all the terms in the governing equations. In addition to the MS in both phases, the source terms in both phases are also computed and printed in a file to be read by the CFD solver. The surface plots of the density field are shown in Figure C.2 for two time instants. The one-dimensional profiles of the pressure field along the y-axis for several x coordinates values and at two time instants are plotted in Figure C.3. We can observe that the MS indeed satisfy the jumps we have prescribed all along the interface and for all times, and is therefore appropriate to be used as the reference solution for our order-of-accuracy study. This study helped us to detect flaws in our CFD solver both in the implementation of the models as well as the numerical formulation. After correcting them, we retrieved the theoretical order of convergence of the DG method, i.e. $p + 1$ with the polynomial order p, as shown in Section 11.3.



a / Time $t = 0.0$. **b /** Time $t = 0.2$.

Figure C.3 / One-dimensional profiles of the pressure field along the y-axis for four different x coordinates values for the inclined interface test case. Plain lines are for phase 1, dashed lines for phase 2.

C.4.2 Test case 2 – Steady circular interface

This second test case follows the same procedure as the first one, except that we determine the MS in polar coordinates (r, θ) before converting it to Cartesian coordinates. The interface is here fixed and we are therefore searching for a time-independent solution. The two-dimensional domain is (in Cartesian coordinates) $\Omega = [0.5, 1.5] \times [0.5, 1.5]$. We specify Dirichlet boundary conditions on velocities and temperature on the boundaries of the domain, in addition to the interface jump conditions. The interface is a circle given by the equation $\Gamma \equiv r = 0.25625$. Phase $k = 1$ is inside the circle, phase $k = 2$ is outside. We again choose the expressions of the MS to be uniquely dependent on a single coordinate, r in this case. We also manufacture the solutions in the primitive variables set such that:

$$\begin{aligned}
\tilde{p}_1(r) &= 3380.84473527662\, r^2 + 21978.0\,, \\
\tilde{v}_{r,1}(r) &= 59.4303622988639\, r^3\,, \\
\tilde{v}_{\theta,1}(r) &= 59.4303622988639\, r^3\,, \\
\widetilde{T}_1(r) &= 1942.94288270104 - 115.466222827242\, r^3\,, \\
\tilde{p}_2(r) &= \lambda_{p,1} - 16751.9333729923\, r^2 + 23300.0\,, \\
\tilde{v}_{r,2}(r) &= \lambda_{v_r,1} + \lambda_{v_r,2}\, r + \lambda_{v_r,3}\, r^2\,, \\
\tilde{v}_{\theta,2}(r) &= \lambda_{v_\theta,1} + \lambda_{v_\theta,2}\, r + \lambda_{v_\theta,3}\, r^2\,, \\
\widetilde{T}_2(r) &= \lambda_{T,1} + \lambda_{T,2}\, r + \lambda_{T,3}\, r^2\,,
\end{aligned} \qquad [C.16]$$

with the set of unknown free parameters

$$\boldsymbol{\lambda} = \{\lambda_{p,1}, \lambda_{v_r,1}, \lambda_{v_r,2}, \lambda_{v_r,3}, \lambda_{v_\theta,1}, \lambda_{v_\theta,2}, \lambda_{v_\theta,3}, \lambda_{T,1}, \lambda_{T,2}, \lambda_{T,3}\}$$

comprising ten components, which corresponds to the number of constraints to be satisfied (7 jumps + 3 wall BC). After having determined the parameters,

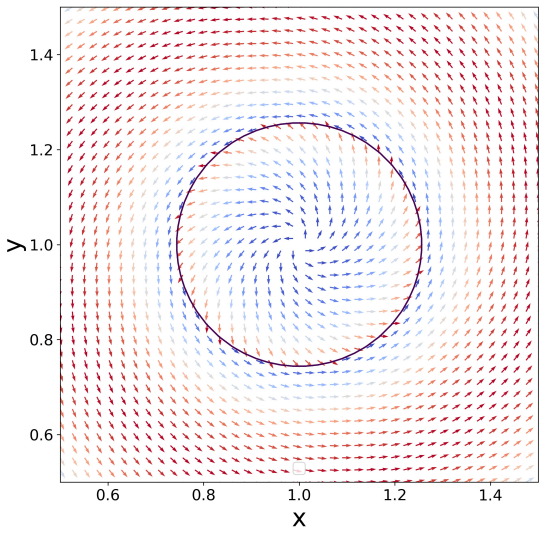


Figure C.4 / Velocity field over the two phases for the circular interface test case.

the problem is transferred to a Cartesian reference frame such that the MS gets significantly more complex to challenge the CFD solver (which solves the equations in Cartesian coordinates). The manufactured velocity field is represented in Figure C.4, the surface plot of the pressure in Figure C.5 and the radial profile of the tangential projection of the shear stress tensor with respect to the interface normal in Figure C.6. The figures have been generated by *PyManufSol* (from the *Outputs* module). In Figure C.6 in particular, we can notice that the intended jump on τ_{nt} is indeed respected, as it should by construction of the MS. This test case allowed us to confront our code with a non-trivial interface geometry while at the same time considering the whole set of non-linear Dirichlet and Neumann interface jump conditions.

C.5 Conclusions and perspectives

This chapter presents the first version of a flexible open-source numerical framework using symbolic manipulations to generate solutions in the context of the Method of Manufactured Solutions (MMS). Compared to previous work, the general approach on which it relies offers the possibility to design potentially any complicated solutions subjected to some user-defined conditions in a convenient and efficient way. This feature makes it an ideal tool to build reference solutions aiming at verifying the treatment of boundaries, interfaces or discontinuities inside a PDE numerical solver. This has been demonstrated on two challenging test cases about a two-phase compressible Navier-Stokes equations system, the first one with a moving straight inclined interface, the second one

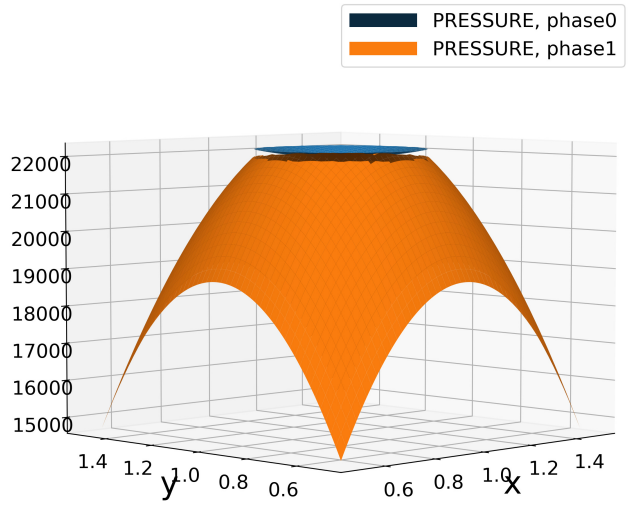


Figure C.5 / Surface plot of the pressure field over the two phases for the circular interface test case.

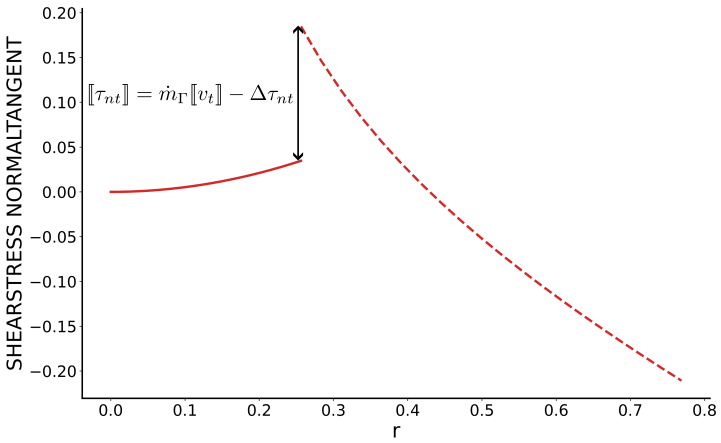


Figure C.6 / One-dimensional profile along the radial coordinate r of the projection of the shear stress tensor along the normal-tangent direction with respect to the interface. Plain line is for phase 1, dashed line for phase 2.

with a fixed circular interface.
As discussed in this chapter, the current approach has some limitations requiring further developments that should be inspired by previous work. In order to relax the constraint on the boundaries geometry, we could investigate the regression approach proposed in Grier et al. (2015) or the use of multiplicative functions encompassing the geometry as done in R. B. Bond et al. (2007). For what concerns the limitations about the use of Neumann-type conditions, the approaches in Brglez (2014) and Freno et al. (2022) that manufacture some parameters instead of the solution directly could be explored.

D

Some semi-analytical reference solutions

This appendix presents the derivation of some reference solutions used in Chapter 11.

D.1 Exact Riemann solver for general EOS

The exact Riemann solver for general EOS has been built based on Saurel et al. (1994) and Quartapelle et al. (2003). One considers the Euler equations governed by a general EOS under the form $p = p(\rho, T)$ with a piecewise constant initial condition at time $t = 0$ with a discontinuity located at $x = x_0$:

$$\mathbf{U} = \begin{cases} \mathbf{U}_L & \text{if} \quad x < x_0 \,, \\ \mathbf{U}_R & \text{if} \quad x > x_0 \,, \end{cases} \qquad [D.1]$$

where the set of unknowns is chosen as $\mathbf{U} = (\rho, u, p)$. This situation of hyperbolic systems of equations with piecewise constant initial condition is referred as a Riemann problem. The solution at $t > 0$ is made of piecewise constant states separated by curves in the plane (x, t), called the characteristics. Along these characteristics, certain quantities, called the invariants, are constant. Depending on the nature of the eigenvalues of the system, different flow features might appear: rarefaction waves, shocks or contact discontinuities. The first step is to determine the intermediate solutions in each region delimited by these special features and then to compute the solution in all the domain for a given time.

D.1.1 Determination of the intermediate states

The intermediate states one would like to determine at the left and at the right of the contact discontinuity are denoted by $\bullet_L^*$ and $\bullet_R^*$ respectively. The solver is based on the properties that the pressure and the velocity are continuous across a contact discontinuity. The density in the left and right intermediate states is then computed by solving the following non-linear system:

$$\begin{cases} p_L^*(\rho_L^*) - p_R^*(\rho_R^*) = 0 \,, \\ u_L^*(\rho_L^*) - u_R^*(\rho_R^*) = 0 \,, \end{cases} \tag{D.2}$$

with:

$$p_L^* = \begin{cases} p_{shock}(\mathbf{U}_L, \rho_L^*) & \text{if} \quad \rho_L^* > \rho_L \,, \\ p_{rar}(\mathbf{U}_L, \rho_L^*) & \text{if} \quad \rho_L^* < \rho_L \,, \end{cases} \quad p_R^* = \begin{cases} p_{shock}(\mathbf{U}_R, \rho_R^*) & \text{if} \quad \rho_R^* > \rho_R \,, \\ p_{rar}(\mathbf{U}_R, \rho_R^*) & \text{if} \quad \rho_R^* < \rho_R \,, \end{cases} \tag{D.3}$$

$$u_L^* = \begin{cases} u_{shock}(\mathbf{U}_L, \rho_L^*) & \text{if} \quad \rho_L^* > \rho_L \,, \\ u_{rar}(\mathbf{U}_L, \rho_L^*) & \text{if} \quad \rho_L^* < \rho_L \,, \end{cases} \quad u_R^* = \begin{cases} u_{shock}(\mathbf{U}_R, \rho_R^*) & \text{if} \quad \rho_R^* > \rho_R \,, \\ u_{rar}(\mathbf{U}_R, \rho_R^*) & \text{if} \quad \rho_R^* < \rho_R \,, \end{cases} \tag{D.4}$$

where the *rar* subscript indicates a solution computed through a rarefaction wave while *shock* is for a solution across a shock wave.

Rarefaction wave Starting from the expression of the Riemann invariant for a rarefaction wave, the velocity across the fan can be computed, from a known left or right state, as

$$u_{rar}(\rho) = \begin{cases} u_L - \int_{\rho_L}^{\rho} c(\rho')/\rho' \, \mathrm{d}\rho' \,, \\ u_R + \int_{\rho_R}^{\rho} c(\rho')/\rho' \, \mathrm{d}\rho' \,, \end{cases} \tag{D.5}$$

The challenge here stems from the evaluation of the Riemann invariant integral which has to be performed for constant entropy. To do so, the speed of sound c is expressed in the general form

$$c^2 = \frac{-C_p}{C_v} \left(\frac{\partial T}{\partial \rho} \right)_p \Big/ \left(\frac{\partial T}{\partial p} \right)_\rho \,, \quad \text{with} \quad C_p = C_v - \frac{T}{\rho^2} \left(\frac{\partial \rho}{\partial T} \right)_p \Big/ \left(\frac{\partial T_{is}}{\partial p} \right)_\rho \,, \tag{D.6}$$

in which the temperature T_{is} is computed along an isentrope from

$$\frac{C_v}{T_{is}} \mathrm{d}T = \frac{1}{\rho^2} \frac{\mathrm{d}\rho}{(\partial T / \partial p)_\rho} \,. \tag{D.7}$$

The pressure is expressed directly by the EOS with this isentropic temperature:

$$p_{rar}(\rho) = p(\rho, T_{is}) \,. \tag{D.8}$$

Shock The pressure and the velocity across a shock from a left or right known state are determined, as function of the density, by rearranging the Rankine-Hugoniot jump relations:

$$\frac{1}{2}\left(p_{shock}-p_{L,R}\right)\left(1/\rho_{L,R}-1/\rho\right)+e(p_{shock},\rho)-e(p_{L,R},\rho_{L,R})=0\,, \qquad [D.9]$$

where e is the internal energy and

$$u_{shock}(\rho)=\begin{cases} u_L-\sqrt{-(p_{shock}(\rho)-p_L)(1/\rho-1/\rho_L)}\,, \\ u_R+\sqrt{-(p_{shock}(\rho)-p_R)(1/\rho-1/\rho_R)}\,. \end{cases} \qquad [D.10]$$

D.1.2 Computation of the global solution

From the intermediate states, it is possible to represent the solution in all the domain for any time. To illustrate the procedure, the example of the shock-tube problem treated in this work is taken. In this situation, a rarefaction wave propagates to the left, a shock wave to the right and a contact discontinuity lies in between the two. The different regions are computed as follows:

- On the left of the rarefaction wave: $x-x_0<t\cdot V_{head}$, $V_{head}=u_L-c_L(\rho_L,p_L)$

$$\mathbf{U}=\mathbf{U}_L \qquad [D.11]$$

- Inside the rarefaction wave: $t\cdot V_{head}<x-x_0<t\cdot V_{tail}$, $V_{tail}=u_L^*-c_L^*(\rho_L^*,p_L^*)$

$$\rho\,:\,u_{rar}(\rho)-u(\rho)=0\,,\quad \text{with}\quad u(\rho)=\frac{x-x_0}{t}+c(\rho,T_{is}) \qquad [D.12]$$

$$T=T_{is}(\rho) \qquad [D.13]$$

$$p=p(\rho,T) \qquad [D.14]$$

- Between the rarefaction and the contact: $t\cdot V_{tail}<x-x_0<t\cdot V_{contact}$, $V_{contact}=u_L^*=u_R^*$

$$\mathbf{U}=\mathbf{U}_L^* \qquad [D.15]$$

- Between the contact and the shock: $t\cdot V_{contact}<x-x_0<t\cdot V_{shock}$, $V_{shock}=u_R-\frac{1}{\rho_R}\frac{u_R-u_R^*}{1/\rho_R-1/\rho_R^*}$

$$\mathbf{U}=\mathbf{U}_R^* \qquad [D.16]$$

- In the right of the shock: $x-x_0>t\cdot V_{shock}$

$$\mathbf{U}=\mathbf{U}_R \qquad [D.17]$$

D.2 Rayleigh-Plesset problem with mass transfer

We derive the Rayleigh-Plesset differential equation in two dimensions. We follow Brennen (2005, Ch. 4) and Jain et al. (2020) but extend their formulation by including here mass transfer effects.

D.2.1 Governing equations in the bulk

The Rayleigh-Plesset model relies on the assumption of an incompressible liquid phase surrounding a compressible circular uniform gas bubble.

Incompressible liquid phase

Inside the liquid, the incompressible equations in two-dimensional cylindrical coordinates write

$$\begin{aligned} \frac{1}{r}\frac{\partial}{\partial r}\left(v_l\, r\right) &= 0\,, \\ \rho_l\left(\frac{\partial v_l}{\partial t} + v_l\frac{\partial v_l}{\partial r}\right) &= -\frac{\partial p_l}{\partial r} + \mu_l\left[-\frac{v_l}{r^2} + \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v_l}{r}\right)\right], \\ \frac{\partial T_l}{\partial t} + v_l\frac{\partial T_l}{\partial r} &= \alpha_l\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T_l}{r}\right)\right], \end{aligned}$$

where r is the radial coordinate, v_l is the liquid radial velocity and the viscous dissipation has been neglected in the energy equation. We note that the liquid density ρ_l is a constant.

Uniform compressible gas phase

The assumption of uniform fields inside the gas bubble is required to be able to derive a semi-analytical solution of the problem. We need to define the way the density, temperature and pressure will be estimated inside the bubble.

Density The density inside the gas which depends on time (but not space) and noted $\rho_b(t)$ can be defined as the ratio between the mass contained inside the bubble and its area, i.e.

$$\rho_b(t) = \frac{m_b(t)}{\pi\, R_b^2(t)}\,,$$

where m_b is the mass of gas inside the bubble and R_b its radius, both depending on time. The idea we came up with is to include in the computation of the mass

the effect of mass transfer by solving the following ODE:

$$\frac{\mathrm{d}m_b(t)}{\mathrm{d}t} = \dot{m}_\Gamma(t)\, 2\pi R_b(t)\,,$$

where $\dot{m}_\Gamma$ is the mass transfer model specific as an input of the problem.

Temperature TO estimate the temperature inside the bubble, we assume an adiabatic compression/expansion process such that

$$\frac{T_b(t)}{T_{b,0}} = \left(\frac{V_{b,0}}{V_b(t)}\right)^{\gamma-1},$$

where $T_{b,0} = T_b(t=0)$ is the initial temperature, $V_b = \pi R_b^2(t)$ is the bubble area and $\gamma = 1.4$ is the specific heat ratio of the gas.

Pressure We could also consider an adiabatic process to estimate the pressure inside the bubble but in order to avoid over-constraining the problem, we rather compute the pressure from the perfect gas EOS, i.e.

$$p_b(t) = p_b^{\mathrm{EOS}}(\rho_b(t), T_b(t))\,.$$

D.2.2 Interface jump conditions

We also need to specify the jump conditions at the interface that will be used as boundary conditions when integrating the equations inside the liquid phase.

Mass jump

The mass balance at the interface can be expressed as

$$\dot{m}_\Gamma = \rho_l(v_l - v_\Gamma)\,,$$

where the interface velocity can be related to the time rate of change of the bubble radius, i.e.

$$v_\Gamma = \frac{\mathrm{d}R_b(t)}{\mathrm{d}t}\,.$$

This leads to a liquid velocity at the interface given by

$$v_l(r = R_b(t), t) = \frac{\mathrm{d}R_b(t)}{\mathrm{d}t} + \frac{\dot{m}_\Gamma(t)}{\rho_l}$$

Normal momentum jump

The momentum jump along the radial direction can be written as

$$[\![\tau_{rr}]\!] = [\![p]\!] + \dot{m}_\Gamma v_l - \sigma\,\kappa_\Gamma\,,$$

where:

- We have assumed that the velocity inside the gas bubble is zero, $v_b = 0$, which should be done to comply with the uniformity assumption.
- The shear stress tensor projected along the normal direction is

$$\tau_{rr} = 2\mu_l \frac{\partial v_l}{\partial r} .$$

- The pressure jump is

$$[[p]] = p_l(r = R_b, t) - p_b(t) .$$

- The bubble curvature is the inverse of its radius,

$$\kappa_\Gamma = \frac{1}{R_b(t)} .$$

D.2.3 ODE derivation for the bubble radius

The derivation of the Rayleigh-Plesset equations follow two main steps:

1. The incompressible mass liquid equation is integrated from the position of the interface toward a certain coordinate r and the mass jump condition is plugged into it to obtain the expression of the velocity inside the liquid:

$$v_l(r,t) = \frac{R_b(t)}{r}\left[\frac{\mathrm{d}R_b(t)}{\mathrm{d}t} + \frac{\dot{m}_\Gamma(t)}{\rho_l}\right] .$$

2. The momentum liquid equation is integrated from the interface position to a far field boundary situated in $r = R_\infty$ on which the pressure is imposed and which can varied in time, i.e.

$$p_l(r = R_\infty, t) = p_{\infty,l}(t) .$$

Rearranging the equation and inserting the expression for the liquid velocity derived above, we can obtain the final second-order ODE describing

the evolution of the bubble radius:

$$\begin{aligned}\ddot{R}_b R_b \log\left(\frac{R_\infty}{R_b}\right) = &\frac{p_b(t) - p_{\infty,l}(t)}{\rho_l} - \frac{\sigma}{\rho_l R_b} \\ &- 2\frac{\mu_l}{\rho_l R_b}\left[\dot{R}_b - \frac{\dot{m}_\Gamma(t)}{\rho_l}\right] + \frac{\dot{m}_\Gamma(t)}{\rho_l}\left[\dot{R}_b - \frac{\dot{m}_\Gamma(t)}{\rho_l}\right] \\ &- \left[\dot{R}_b^2 - \frac{1}{\rho_l}\left(\dot{R}_b \dot{m}_\Gamma(t) + R_b \frac{\mathrm{d}\dot{m}_\Gamma(t)}{\mathrm{d}t}\right)\right]\log\left(\frac{R_\infty}{R_b}\right) \\ &- \frac{1}{2}\left[\dot{R}_b^2 - \frac{\dot{m}_\Gamma(t)}{\rho_l}\left(2\dot{R}_b - \frac{\dot{m}_\Gamma(t)}{\rho_l}\right)\right]\left[\left(\frac{R_b}{R_\infty}\right)^2 - 1\right].\end{aligned} \quad [\text{D.18}]$$

This equation is solved with a `Python` script.

Rest of this page intentionally left blank.

E

Dimensional analysis for the melting ablation problem

This appendix presents the derivation of the similarity analysis used in Chapter 12.

Definition of the coordinate system and the boundary layer thicknesses

Coordinate system Following S.-Y. Chen and S. J. Allen (1962), we define the coordinate system for a body of revolution. The radial direction y is measured from the gas-liquid interface, identified by the subscript $\bullet_i$. The parametric polar coordinate is noted x, on which depends the body radius $r(x)$. The radial velocity is written v and the polar one u. The quantities at the edge of the boundary layer are identified by the subscript $\bullet_e$ and the ones at the liquid-solid interface by $\bullet_j$.

Boundary layer thicknesses We also define four thickness: the gas momentum and thermal boundary layer thicknesses, δ_g and $\delta_{T,g}$, respectively, and the liquid momentum and thermal boundary layer thicknesses, δ_l and $\delta_{T,l}$. Those variables are unknown and will be part of the system to be solved later in this chapter.

Dimensional governing equations

Gas phase The main assumptions for the gas phase can be summarized as:

- The melt dynamics, the body shape change or the body tumbling do not influence significantly the gas flow, which leads to the use of steady equations.
- Ching-Jen Chen and Ostrach (1971) explain that as far as the heat flux and shear effects on the liquid are concerned, Fay and Riddel show that ionization and dissociation (in the gas phase) are secondary effects. This motivates the use of single species equations for the gas phase, combined with constant thermophysical properties.
- We also make the hypothesis of laminar flow.

Since we are only interested in replicating the boundary layer region phenomena, the axisymmetric steady laminar compressible boundary layer equations are written, following Barbante et al. (2002), as

$$\begin{aligned} \frac{\partial(\rho_g\, u_g\, r)}{\partial x} + \frac{\partial(\rho_g\, v_g\, r)}{\partial y} &= 0 \\ \rho_g \left(u_g \frac{\partial u_g}{\partial x} + v_g \frac{\partial u_g}{\partial y} \right) &= \frac{\mathrm{d}p_g}{\mathrm{d}x} + \frac{\partial}{\partial y}\left(\mu_g \frac{\partial u_g}{\partial y} \right) \\ \rho_g\, C_{p,g} \left(u_g \frac{\partial T_g}{\partial x} + v_g \frac{\partial T_g}{\partial y} \right) &= u_g \frac{\mathrm{d}p_g}{\mathrm{d}x} + \mu_g \left(\frac{\partial u_g}{\partial y} \right)^2 + \frac{\partial}{\partial y}\left(\lambda_g \frac{\partial T_g}{\partial y} \right). \end{aligned} \tag{E.1}$$

Liquid phase The simplifying assumptions for the liquid phase are:

- The heterogeneous chemical reactions, such as oxidation, are not considered to influence the melt dynamics.
- Rotation and deceleration body forces are neglected, even though they can be taken into account, as done e.g. in Cochran and Ostrach (1968).
- The properties are assumed constant, even the viscosity which is usually expressed as a function of the temperature to describe glassy materials in such kind of analysis, as e.g. in Edling and Ostrach (1966, App.).
- The flow in the liquid layer is assumed laminar and incompressible, without thermal dilatation effects. However, the unsteady effects must be considered.
- The liquid layer is assumed thin compared to the body dimensions, which leads us to consider the boundary layer equations.

Following S.-Y. Chen and S. J. Allen (1962), the unsteady laminar incompressible boundary layer equations in the liquid phase write

$$\begin{aligned} \frac{\partial(u_l\, r)}{\partial x} + \frac{\partial(v_l\, r)}{\partial y} &= 0 \\ \rho_l \left(\frac{\partial u_l}{\partial t} + u_l \frac{\partial u_l}{\partial x} + v_l \frac{\partial u_l}{\partial y} \right) &= \frac{\mathrm{d}p_l}{\mathrm{d}x} + \frac{\partial}{\partial y}\left(\mu_l \frac{\partial u_l}{\partial y} \right) \\ \rho\, C_{p,l} \left(\frac{\partial T_l}{\partial t} + u_l \frac{\partial T_l}{\partial x} + v_l \frac{\partial T_l}{\partial y} \right) &= \mu_l \left(\frac{\partial u_l}{\partial y} \right)^2 + \frac{\partial}{\partial y}\left(\lambda_l \frac{\partial T_l}{\partial y} \right). \end{aligned} \tag{E.2}$$

Solid phase Inside the solid, the standard heat conduction equation is considered,

$$\rho_s C_{p,s} \frac{\partial T_s}{\partial t} = \frac{\partial}{\partial y}\left(\lambda_s \frac{\partial T_s}{\partial y}\right). \tag{E.3}$$

Gas–liquid wave motion relation Inspired by Cochran and Ostrach (1968, App.B) and Edling and Ostrach (1966, App.), we also add an equation governing the motion of the gas-liquid interface. Since the y coordinate is attached to the moving interface, we have that

$$y \equiv \Gamma(x,t)\,,$$

where $\Gamma(x,t)$ is the time-dependent equation of the interface depending on the coordinate x, similarly to the radius of the body $r(x)$, except that this latter is time independent. If we take the time derivative of this definition, we can write that

$$\frac{\partial y}{\partial t} = \frac{\partial \Gamma}{\partial t} + \frac{\partial \Gamma}{\partial x}\frac{\partial x}{\partial t}\,,$$

where, by construction,

$$\frac{\partial x}{\partial t} = u_i\,, \quad \frac{\partial y}{\partial t} = v_i\,,$$

the two interface velocity components appear. At the end, we get

$$\frac{\partial \Gamma}{\partial t} + u_i \frac{\partial \Gamma}{\partial x} - v_i = 0\,. \tag{E.4}$$

Dimensional interface and boundary conditions

Gas edge boundary layer At the gas boundary layer edge, with position $y = \delta_g(x)$, we need to prescribe the gas quantities of the boundary layer equations as being equal to the ones coming from the freestream evaluated at the boundary layer edge location, i.e.

$$\begin{aligned} u_g &= u_e(x)\,,\\ v_g &= v_e(x)\,,\\ p_g &= p_e(x)\,,\\ T_g &= T_e(x)\,. \end{aligned} \tag{E.5}$$

Gas–liquid interface At the gas-liquid interface with position $y = 0$, we apply the jump conditions of Equation 3.40 projected along the normal and tangential directions, i.e. along the y and x directions here, and simplify them with the following assumptions:

- Mass transfer effects are neglected, as already mentioned above.
- Surface tension effects are neglected in the normal momentum jump.
- We don't consider any jumps on tne temperature and tangential velocities.

- We neglect radiation, viscous dissipation, pressure work and surface tension work in the energy jump.

Based on those assumptions and the fact that the boundary layer equations are considered for both phases, we obtain

$$\begin{aligned} u_g &= u_l = u_i\,, \\ v_g &= v_l = v_i\,, \\ p_g &= p_l\,\kappa\,, \\ T_g &= T_l = T_i\,, \\ \mu_g \frac{\partial u_g}{\partial y} &= \mu_l \frac{\partial u_l}{\partial y}\,, \\ \lambda_g \frac{\partial T_g}{\partial y} &= \lambda_l \frac{\partial T_l}{\partial y}\,. \end{aligned} \tag{E.6}$$

Liquid–solid interface At the liquid-solid interface, considering it as sharp interface like the gas-liquid one, we have the following conditions at $y = -\delta_l(x)$:

$$\begin{aligned} u_l &= u_s = 0\,, \\ v_l &= v_s = 0\,, \\ T_l &= T_s = T_{\text{melt}}\,, \\ \lambda_l \frac{\partial T_l}{\partial y} &= \lambda_s \frac{\partial T_s}{\partial y} - \rho_s\,\Delta h_{\text{melt}}\,v_s\,. \end{aligned} \tag{E.7}$$

Unaffected solid core Finally, we should impose some conditions to fix the state of the solid body in its core, at "infinity" such that $y \to -\infty$:

$$\begin{aligned} u_s &= 0\,, \\ v_s &= 0\,, \\ T_s &= T_{\infty,s}\,, \\ \frac{\partial u_s}{\partial y} &= 0\,, \\ \frac{\partial T_s}{\partial y} &= 0\,. \end{aligned} \tag{E.8}$$

Nondimensionalization

The next step is to introduce common reference quantities to make all the quantities used above nondimensional.

Common non–dimensional variables We introduce two common reference quantities, namely a reference length D^*, which will typically be the body

dimension, and a reference timescale τ^*, which is *a priori* unknown. This allows us to define the following common non-dimensional quantities

$$\overline{x} = \frac{x}{D^*}, \quad \overline{r} = \frac{r}{D^*}, \quad \overline{t} = \frac{t}{\tau^*}, \quad \overline{\Gamma} = \frac{\Gamma}{D^* \delta_l}, \tag{E.9}$$

where the gas-liquid interface position, Γ, is nondimensionalized according to Edling and Ostrach (1966, App.). We can note that while the Nondimensionalization of the x dimension is common for all the regions of the flow, it is not the case for the y dimension which necessitates a specific scaling depending on the region.

Non-dimensional gas variables The non-dimensional variables associated to the gas phase are

$$\begin{aligned} \overline{y}_g &= \frac{y}{\delta_g D^*}, \quad \overline{y}_{T,g} = \frac{y}{\delta_{T,g} D^*}, \\ \overline{u}_g &= \frac{u_g}{u_e}, \quad \overline{v}_g = \frac{v_g}{\alpha_g u_e}, \\ \overline{p}_g &= \frac{p_g}{\rho_e u_e^2}, \quad \overline{T}_g = \frac{T_g - T_i}{T_e - T_i}, \end{aligned} \tag{E.10}$$

where the radial direction is made dimensionless based on the momentum or thermal boundary layer thickness depending on whether it appears in the mass/momentum or energy equation. In addition to δ_g and $\delta_{T,g}$, two more unknowns are added: the interface temperature T_i and the scaling factor for the radial velocity α_g.

Non-dimensional liquid variables Similarly to the gas phase, we perform the following nondimensionalization for the liquid:

$$\begin{aligned} \overline{y}_l &= \frac{y}{\delta_l D^*}, \quad \overline{y}_{T,l} = \frac{y}{\delta_{T,l} D^*}, \\ \overline{u}_l &= \frac{u_l}{u_l^*}, \quad \overline{v}_l = \frac{v_l}{\alpha_l u_l^*}, \\ \overline{p}_l &= \frac{p_l}{p_l^*}, \quad \overline{T}_l = \frac{T_l - T_{\infty,s}}{T_i - T_{\infty,s}}, \end{aligned} \tag{E.11}$$

where, contrary to the gas phase, the reference velocity and pressure introduced in the nondimensionalization, u_l^* and p_l^*, represent additional unknowns.

Non-dimensional solid variables Finally, for the solid phase, we propose:

$$\begin{aligned} \bar{t} &= \frac{t}{\tau_s^*}\,, \quad \bar{y}_{T,s} = \frac{y}{\delta_{T,s}\, D^*}\,, \\ \overline{T}_s &= \frac{T_s - T_{\infty,s}}{T_j - T_{\infty,s}}\,, \quad \bar{v}_s = \frac{v_s\, \tau_s^*}{\delta_{T,s}\, D^*}\,, \end{aligned} \qquad [\text{E.12}]$$

where a new reference timescale specific to the solid, τ_s^*, is introduced, as well as a characteristic dimension for the solid heat conduction, $\delta_{T,s}$. Those are two additional unknowns of the problem. The liquid-solid interface temperature is assumed to be known and equal to the melting temperature,

$$T_j = T_{\text{melt}}\,.$$

Reference dimensionless numbers

We anticipate the derivation of the dimensionless equations by presenting the reference non-dimensional parameters that can be formed based on the reference quantities reviewed above. Those characteristic non-dimensional parameters will appear in the dimensionless governing equations and each describe as ratios the relative magnitude of specific fluid and physical system characteristics.

Gas characteristic numbers The reference dimensionless parameters for the gas phase are defined as

$$\begin{aligned} \text{Re}_g &= \frac{\rho_e\, D^*\, u_e}{\mu_g} && \text{Reynolds}\,, \\ \text{Pr}_g &= \frac{C_{p,g}\, \mu_g}{\lambda_g} && \text{Prandtl}\,, \\ \text{Ec}_g &= \frac{u_e^2}{C_{p,g}\, (T_e - T_i)} && \text{Eckert}\,. \end{aligned}$$

Liquid characteristic numbers The reference dimensionless parameters for the liquid phase are defined as

$$\begin{aligned} \text{Re}_l &= \frac{\rho_l\, D^*\, \delta_l^2\, u_l^*}{\mu_l} && \text{Reynolds}\,, \\ \text{Pr}_l &= \frac{C_{p,l}\, \mu_l}{\lambda_l} && \text{Prandtl}\,, \\ \text{Ec}_l &= \frac{(u_l^*)^2}{C_{p,l}\, (T_i - T_{\infty,s})} && \text{Eckert}\,, \\ \text{N}_{p,l} &= \frac{p_l^*\, D^*\, \delta_l^2}{\mu_l\, u_l^*} && \text{Pressure parameter}\,, \end{aligned}$$

where the last pressure parameter is introduced based on Cochran and Ostrach (1968, App. A).

Solid characteristic numbers The two reference dimensionless parameters for the solid phase are defined as

$$\mathrm{Fo}_s = \frac{\alpha_s\ \tau_s^*}{(D^*)^2} \qquad \text{Fourier}\,,$$
$$\mathrm{Ste}_s = \frac{C_{p,s}(T_j - T_{\infty,s})}{\Delta h_{\mathrm{melt}}} \qquad \text{Stefan}\,.$$

Dimensionless equations

Now that we have introduced the reference quantities and the reference non-dimensional parameters that can be expressed based on them, we use those in order to transform the dimensional equations presented above into dimensionless equations, not only inside each phase but also for the jump conditions.

Gas–liquid wave motion relation The non-dimensionalization of the gas-liquid wave relation yields

$$\frac{D^*}{u_l^*\ \tau^*}\frac{\partial \overline{\Gamma}}{\partial \overline{t}} + \overline{u}_i \frac{\partial \overline{\Gamma}}{\partial \overline{x}} - \overline{v}_i = 0\,. \qquad \text{[E.13]}$$

Gas The non-dimensional gas equations are

$$\rho_e\ u_e \left(\frac{\partial(\overline{\rho}_g\ \overline{u}_g\ \overline{r})}{\partial \overline{x}} + \frac{\partial(\overline{\rho}_g\ \overline{v}_g\ \overline{r})}{\partial \overline{y}_g} \frac{\alpha_g}{\delta_g} \right) = 0 \qquad \text{[E.14a]}$$

$$\overline{\rho}_g \left(\overline{u}_g \frac{\partial \overline{u}_g}{\partial \overline{x}} + \overline{v}_g \frac{\partial \overline{u}_g}{\partial \overline{y}_g} \right) = -\frac{\mathrm{d}\overline{p}_g}{\mathrm{d}\overline{x}} + \frac{1}{\mathrm{Re}_g\ \delta_g^2} \frac{\partial^2 \overline{u}_g}{\partial \overline{y}_g^2} \qquad \text{[E.14b]}$$

$$\overline{\rho}_g \left(\overline{u}_g \frac{\partial \overline{T}_g}{\partial \overline{x}} + \frac{\delta_g}{\delta_{T,g}} \overline{v}_g \frac{\partial \overline{T}_g}{\partial \overline{y}_{T,g}} \right) = \mathrm{Ec}_g\ \overline{u}_g \frac{\mathrm{d}\overline{p}_g}{\mathrm{d}\overline{x}} + \frac{\mathrm{Ec}_g}{\mathrm{Re}_g} \frac{1}{\delta_{T,g}^2} \left(\frac{\partial \overline{u}_g}{\partial \overline{y}_{T,g}} \right)^2 + \frac{1}{\mathrm{Re}_g\ \mathrm{Pr}_g} \frac{1}{\delta_{T,g}^2} \frac{\partial^2 \overline{T}_g}{\partial \overline{y}_{T,g}^2}\,. \qquad \text{[E.14c]}$$

Liquid The non-dimensional liquid equations are

$$u_l^* \left(\frac{\partial(\overline{u}_l\ \overline{r})}{\partial \overline{x}} + \frac{\partial(\overline{v}_l\ \overline{r})}{\partial \overline{y}_l} \frac{\alpha_l}{\delta_l} \right) = 0 \qquad \text{[E.15a]}$$

$$\mathrm{Re}_l \left(\frac{D^*}{u_l^*\ \tau^*} \frac{\partial \overline{u}_l}{\partial \overline{t}} + \overline{u}_l \frac{\partial \overline{u}_l}{\partial \overline{x}} + \overline{v}_l \frac{\partial \overline{u}_l}{\partial \overline{y}_l} \right) = -\mathrm{N}_{p,l} \frac{\mathrm{d}\overline{p}_l}{\mathrm{d}\overline{x}} + \frac{\partial^2 \overline{u}_l}{\partial \overline{y}_l^2} \qquad \text{[E.15b]}$$

$$\frac{D^*}{u_l^*\ \tau^*} \frac{\overline{T}_l}{\partial \overline{t}} + \overline{u}_l \frac{\partial \overline{T}_l}{\partial \overline{x}} + \frac{\delta_l}{\delta_{T,l}} \overline{v}_l \frac{\partial \overline{T}_l}{\partial \overline{y}_{T,l}} = \frac{\mathrm{Ec}_l}{\mathrm{Re}_l} \frac{\delta_l^2}{\delta_{T,l}^2} \left(\frac{\partial \overline{u}_l}{\partial \overline{y}_{T,l}} \right)^2 + \frac{1}{\mathrm{Re}_l\ \mathrm{Pr}_l} \frac{\delta_l^2}{\delta_{T,l}^2} \frac{\partial^2 \overline{T}_l}{\partial \overline{y}_{T,l}^2}\,. \qquad \text{[E.15c]}$$

Solid The non-dimensional temperature equation in the solid phase is

$$\overline{\rho}_s \frac{\partial \overline{T}_s}{\partial \overline{t}_s} = \frac{\mathrm{Fo}_s}{\delta_{T,s}^2} \frac{\partial^2 \overline{T}_s}{\partial \overline{y}_{T,s}^2} \,. \qquad \text{[E.16]}$$

Gas–liquid jump conditions If we focus on the normal and tangential momentum jumps and the energy jump, the non-dimensional jump conditions at the gas-liquid interface can be written as

$$\rho_e \, u_e^2 \, \overline{p}_g = p_l^* \, \overline{p}_l \qquad \text{[E.17a]}$$

$$\frac{\mu_e \, u_e}{D^* \, \delta_g} \frac{\partial \overline{u}_g}{\partial \overline{y}_g} = \frac{\mu_l \, u_l^*}{D^* \, \delta_l} \frac{\partial \overline{u}_l}{\partial \overline{y}_l} \qquad \text{[E.17b]}$$

$$\lambda_e \frac{T_e - T_i}{D^* \, \delta_{T,g}} \frac{\partial \overline{T}_g}{\partial \overline{y}_{T,g}} = \lambda_l \frac{T_i - T_{\infty,s}}{D^* \, \delta_{T,l}} \frac{\partial \overline{T}_l}{\partial \overline{y}_{T,l}} \,. \qquad \text{[E.17c]}$$

Liquid–solid jump conditions

$$\lambda_l \frac{T_i - T_{\infty,s}}{D^* \, \delta_{T,l}} \frac{\partial \overline{T}_l}{\partial \overline{y}_{T,l}} = \lambda_s \frac{T_j - T_{\infty,s}}{D^* \, \delta_{T,s}} \frac{\partial \overline{T}_s}{\partial \overline{y}_{T,s}} - \frac{1}{\mathrm{Ste}_s} \rho_s \, C_{p,s} (T_j - T_{\infty,s}) \frac{D^* \, \delta_{T,s}}{\tau_s^*} \overline{v}_s \,. \qquad \text{[E.18]}$$

Determination of reference quantities based on order-of-magnitude analysis

We have introduced 12 unknown reference quantities in the non-dimensionalization process:

$$\tau^*, \, \delta_g, \, \delta_{T,g}, \, \alpha_g, \, \delta_l, \, \delta_{T,l}, \, \alpha_l, \, u_l^*, \, p_l^*, \, T_i, \, \tau_s^*, \, \delta_{T,s} \,.$$

We are now in a position to express those quantities based on known ones such that they are consistent with the non-dimensional relations derived above, hence complying with the physical laws governing the problem considered. Since they are 12 equations (3 for the gas, 3 for the liquid, 1 for the solid, 1 for the gas-liquid wave motion, 3 for the gas-liquid jump conditions, 1 for the liquid-solid jumps) for 12 unknowns, our approach leads to a well-posed problem. More precisely, all the terms involving non-dimensional quantities, $\overline{\bullet}$, should be of the order of unity if our scaling is properly chosen. From this observation, we can isolate in each equation the dimensional parameters and express them based on other ones or characteristic non-dimensional numbers. We give here

directly the final expressions resulting from some algebraic manipulations:

$$
\begin{aligned}
\delta_g &= \frac{1}{\mathrm{Re}_g^{0.5}} && \text{from Equation E.14b,}\\
\delta_{g,T} &= \frac{1}{\mathrm{Re}_g^{0.5}\,\mathrm{Pr}_g} && \text{from Equation E.14c, assuming } \mathrm{Ec}_g \ll 1,\\
p_L &= \rho_e\, u_e^2 && \text{from Equation E.17a,}\\
u_l^* &= \sqrt{\frac{\rho_e}{\rho_l}}\, u_e && \text{from Equation E.15b,}\\
\delta_l &= \frac{\mu_l\, u_l^*\, \delta_g}{\mu_e\, u_e} && \text{from Equation E.17b,}\\
\delta_{T,l} &= \frac{\lambda_l}{\delta_l \rho_l D^* C_{p,l} u_l^*} && \text{from Equation E.15c, assuming } \mathrm{Ec}_l \ll 1,\\
T_i &= \frac{\lambda_l T_{\infty,s} + \delta_{T,l}/\delta_{T,g}\lambda_g T_e}{\lambda_l + \delta_{T,l}/\delta_{T,g}\lambda_g} && \text{from Equation E.17c,}\\
\tau &= \frac{D^*}{u_l^*} && \text{from Equation E.13,}\\
\delta_{T,s} &= \frac{1}{\mathrm{Ste}_s}\delta_{T,l}\frac{\lambda_s}{\lambda_l}\frac{(T-j-T_{\infty,s})}{(T-i-T_{\infty,s})} && \text{from Equation E.18,}\\
\tau_s^* &= \frac{(\delta_{T,s} D^*)^2}{\alpha_s} && \text{from Equation E.16,}\\
\alpha_g &= \delta_g && \text{from Equation E.14a,}\\
\alpha_l &= \delta_l && \text{from Equation E.15a.}
\end{aligned}
\qquad [E.19]
$$

Required input parameters

In order to be able to compute numerical values for the reference quantities above, and so also for the associated non-dimensional characteristic numbers, we need as inputs of our procedure the following quantities:

- *Gas edge boundary layer conditions*: the flow conditions at the edge of the gas boundary layer in Equation E.5 are required. Those can be estimated by simplified correlations or extracted directly from CFD simulations.
- *Unaffected solid core conditions*: we also need the characteristic dimension of the body under consideration (e.g. its radius) and its initial temperature, $T_{\infty,s}$.
- *Reference material properties*: finally, we need to collect representative values (assumed constant over all the domain!) for the material properties, namely the density for the liquid and solid, ρ, the viscosity for the gas and liquid, μ, the melting temperature, T_{melt} for the condensed phase, the thermal conductivity, λ and the specific heat capacity, C_p, for the three phases.

Rest of this page intentionally left blank.

Bibliography

Aalilija, A, Ch-A Gandin, and Elie Hachem (2020). "On the analytical and numerical simulation of an oscillating drop in zero-gravity." In: *Computers & Fluids* 197, p. 104362 (cit. on pp. 414, 415).

Abadie, Thomas, Joelle Aubin, and Dominique Legendre (2015). "On the combined effects of surface tension force calculation and interface advection on spurious currents within volume of fluid and level set frameworks." In: *Journal of Computational Physics* 297, pp. 611–636 (cit. on p. 246).

Abdi, Daniel S et al. (2019). "A GPU-accelerated continuous and discontinuous Galerkin non-hydrostatic atmospheric model." In: *The International Journal of High Performance Computing Applications* 33.1, pp. 81–109 (cit. on p. 158).

Abeyaratne, Rohan (2006). *Lecture Notes on The Mechanics of Elastic Solids - Volume I: A Brief Review of Some Mathematical Preliminaries* (cit. on p. 56).

– (2012). *Lecture Notes on The Mechanics of Elastic Solids - Volume II: Continuum Mechanics* (cit. on pp. 48, 51, 58, 59, 90).

Abgrall, Rémi (Apr. 1996). "How to Prevent Pressure Oscillations in Multicomponent Flow Calculations: A Quasi Conservative Approach." In: *Journal of Computational Physics* 125.1, pp. 150–160 (cit. on p. 87).

Adalsteinsson, D and J.A Sethian (Jan. 1999). "The Fast Construction of Extension Velocities in Level Set Methods." In: *Journal of Computational Physics* 148.1, pp. 2–22 (cit. on pp. 220, 221).

Adalsteinsson, David and James A Sethian (1995). "A fast level set method for propagating interfaces." In: *Journal of computational physics* 118.2, pp. 269–277 (cit. on p. 229).

Adams, Thomas (2020). "Discontinuous Galerkin Discretised Level Set Methods with Applications to Topology Optimisation." PhD thesis. Durham University (cit. on p. 229).

Adams, Thomas et al. (Nov. 2019). "A Parabolic Level Set Reinitialisation Method Using a Discontinuous Galerkin Discretisation." In: *Computers & Mathematics with Applications* 78.9, pp. 2944–2960 (cit. on p. 212).

Agarwal, Ramesh K., Keon-Young Yun, and Ramesh Balakrishnan (Oct. 2001). "Beyond Navier–Stokes: Burnett Equations for Flows in the Continuum–Transition Regime." In: *Physics of Fluids* 13.10, pp. 3061–3085 (cit. on p. 103).

Agostinelli, Pasquale Walter et al. (Mar. 10, 2020). "Investigation of Hypersonic Flow in the VKI H3 Wind Tunnel: From Facility Characterization to Boundary-Layer Interaction over Low-Temperature Ablators." In: *23rd AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. 23rd AIAA International Space Planes and Hypersonic Systems and Technologies Con-

ference. Montreal, Quebec, Canada: American Institute of Aeronautics and Astronautics (cit. on p. 444).

Ahrens, James et al. (2005). "Paraview: An end-user tool for large-data visualization." In: *The visualization handbook* 717, pp. 50038–1 (cit. on p. 378).

Alexiades, Vasilios and Alan D. Solomon (1993). *Mathematical Modeling of Melting and Freezing Processes*. Washington: Hemisphere Pub. Corp. 323 pp. (cit. on p. 111).

Allen, Michael P et al. (2004). "Introduction to molecular dynamics simulation." In: *Computational soft matter: from synthetic polymers to proteins* 23, pp. 1–28 (cit. on pp. 493, 494).

Anderson, John David (1990). *Modern compressible flow: with historical perspective*. Vol. 12. McGraw-Hill New York (cit. on p. 18).

Anderson Jr., John D. (Oct. 31, 2006). *Hypersonic and High-Temperature Gas Dynamics, Second Edition*. Reston ,VA: American Institute of Aeronautics and Astronautics (cit. on pp. 18, 20, 25, 27, 28).

Annaloro, Julien et al. (2014). "Elaboration of a new spacecraft-oriented tool: PAMPERO." In: *Proceedings of the 8th European Symposium on Aerothermodynamics for Space Vehicles* (cit. on p. 14).

Annaloro, Julien et al. (June 2020). "Aerothermodynamics Modelling of Complex Shapes in the DEBRISK Atmospheric Reentry Tool: Methodology and Validation." In: *Acta Astronautica* 171, pp. 388–402 (cit. on p. 19).

Anumolu, Lakshman and Mario F. Trujillo (Aug. 2013). "Gradient Augmented Reinitialization Scheme for the Level Set Method." In: *International Journal for Numerical Methods in Fluids*, n/a–n/a (cit. on pp. 212, 217).

Appel, Daniel et al. (2021). "A narrow band-based dynamic load balancing scheme for the level-set ghost-fluid method." In: *International Conference on High Performance Computing in Science and Engineering*. Springer, pp. 305–320 (cit. on p. 229).

Aris, Rutherford (1962). *Vectors, tensors and the basic equations of fluid mechanics*. Dover Publication (cit. on p. 93).

Arnold, Douglas N et al. (2002). "Unified analysis of discontinuous Galerkin methods for elliptic problems." In: *SIAM journal on numerical analysis* 39.5, pp. 1749–1779 (cit. on pp. 170, 172).

Arya, Sunil et al. (1998). "An optimal algorithm for approximate nearest neighbor searching fixed dimensions." In: *Journal of the ACM (JACM)* 45.6, pp. 891–923 (cit. on p. 213).

Askes, Harm, Ellen Kuhl, and Paul Steinmann (2004). "An ALE formulation based on spatial and material settings of continuum mechanics. Part 2: Classification and applications." In: *Computer methods in applied mechanics and engineering* 193.39-41, pp. 4223–4245 (cit. on pp. 48, 54).

Atallah, Nabil M., Claudio Canuto, and Guglielmo Scovazzi (May 2022). "The High-Order Shifted Boundary Method and Its Analysis." In: *Computer Methods in Applied Mechanics and Engineering* 394, p. 114885 (cit. on p. 270).

Ayrinhac, S (2000). *Compilation of thermodynamic and elastic data of liquid Ga*. Tech. rep. Université Pierre et Marie Curie (UPMC), Sorbonne Univ (cit. on p. 449).

Badia, Santiago, Francesc Verdugo, and Alberto F. Martín (July 2018). "The Aggregated Unfitted Finite Element Method for Elliptic Problems." In: *Computer Methods in Applied Mechanics and Engineering* 336, pp. 533–553 (cit. on p. 272).

Baer, Melvin R and Jace W Nunziato (1986). "A two-phase mixture theory for the deflagration-to-detonation transition (DDT) in reactive granular materials." In: *International journal of multiphase flow* 12.6, pp. 861–889 (cit. on p. 116).

Bai, Xiao and Xiaolong Deng (Oct. 2017). "A Sharp Interface Method for Compressible Multi-Phase Flows Based on the Cut Cell and Ghost Fluid Methods." In: *Advances in Applied Mathematics and Mechanics* 9.5, pp. 1052–1075 (cit. on p. 317).

Bai, Xiao and Maojun Li (June 2022). "Simulating Compressible Two-Phase Flows with Sharp-Interface Discontinuous Galerkin Methods Based on Ghost Fluid Method and Cut Cell Scheme." In: *Journal of Computational Physics* 459, p. 111107 (cit. on p. 317).

Barbante, P. F., G. Degrez, and G. S. R. Sarma (Oct. 2002). "Computation of Nonequilibrium High-Temperature Axisymmetric Boundary-Layer Flows." In: *Journal of Thermophysics and Heat Transfer* 16.4, pp. 490–497 (cit. on p. 538).

Bariselli, Federico (2021). "Development of an ablation model for rarefied flows with application to meteoroid entry." PhD thesis. Politecnico di Milano (cit. on p. 46).

Barka, L. et al. (Jan. 2019). "Oxidation and Emissivity of Invar 36 Alloy in Air Plasma at High Temperatures." In: *Journal of Alloys and Compounds* 772, pp. 1003–1016 (cit. on p. 27).

Barlow, Andrew J. et al. (Oct. 2016). "Arbitrary Lagrangian–Eulerian Methods for Modeling High-Speed Compressible Multimaterial Flows." In: *Journal of Computational Physics* 322, pp. 603–665 (cit. on pp. 48, 199).

Barter, Garrett E and David L Darmofal (2010). "Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation." In: *Journal of Computational Physics* 229.5, pp. 1810–1827 (cit. on pp. 303, 305).

Barton, Iain Edward (1998). "Comparison of SIMPLE-and PISO-type algorithms for transient flows." In: *International Journal for numerical methods in fluids* 26.4, pp. 459–483 (cit. on p. 180).

Bassi, F and S Rebay (2000). "GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations." In: *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Springer, pp. 197–208 (cit. on p. 157).

– (Mar. 1997). "A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier–Stokes Equations." In: *Journal of Computational Physics* 131.2, pp. 267–279 (cit. on p. 170).

Bassi, F. et al. (2015). "Time Integration in the Discontinuous Galerkin Code MIGALE - Unsteady Problems." In: *IDIHOM: Industrialization of High-Order Methods - A Top-Down Approach*. Ed. by Norbert Kroll et al. Vol. 128. Cham: Springer International Publishing, pp. 205–230 (cit. on pp. 185, 351).

Bassi, Francesco et al. (1997). "A high-order accurate discontinuous finite element method for inviscid and viscous turbomachinery flows." In: *Proceedings of the 2nd European Conference on Turbomachinery Fluid Dynamics and Thermodynamics*. Antwerpen, Belgium, pp. 99–109 (cit. on p. 170).

Bassi, Francesco et al. (2009). "A discontinuous Galerkin method for inviscid low Mach number flows." In: *Journal of Computational Physics* 228.11, pp. 3996–4011 (cit. on p. 189).

Bassi, Francesco et al. (2014). "Investigation of high-order temporal schemes for the discontinuous Galerkin solution of the Navier–Stokes equations." In: *11th World Congress on Computational Mechanics, WCCM*, pp. 5651–5662 (cit. on p. 343).

Basting, Christopher and Dmitri Kuzmin (2013). "A minimization-based finite element formulation for interface-preserving level set reinitialization." In: *Computing* 95.Suppl 1, pp. 13–25 (cit. on p. 212).

Basu, Biswajit and A W Date (Nov. 1988). "Numerical Modelling of Melting and Solidification Problems—A Review." In: *Sadhana* 13.3, pp. 169–213 (cit. on p. 132).

Batchelor, George Keith (1967). *An introduction to fluid dynamics*. Cambridge university press (cit. on p. 66).

Battista, Ruben Di (2021). "Towards a unified eulerian modeling framework for two-phase flows : geometrical small scale phenomena and associated flexible computing strategies." PhD thesis. Ecole polytechnique, Paris (cit. on p. 116).

Beck, Andrea (2015). "High order discontinuous Galerkin methods for the simulation of multiscale problems." PhD thesis. University of Stuttgart (cit. on pp. 154, 157, 159).

Beck, J. et al. (2017). "Progress in Hybrid Spacecraft/Object Oriented Destructive Re-entry Modelling Using the SAM Code." In: *7th European Conference on Space Debris*. ESA Space Debris Office (cit. on p. 15).

Beck, Lauritz and Florian Kummer (2023). *High-Order Numerical Integration on Domains Bounded by Intersecting Level Sets*. arXiv: 2308.10698 [cs]. URL: https://arxiv.org/abs/2308.10698 (cit. on p. 286).

Beckermann, C. and R. Viskanta (1993). "Mathematical Modeling of Transport Phenomena During Alloy Solidification." In: *Applied Mechanics Reviews* 46.1, pp. 1–27 (cit. on pp. 114, 126, 127).

Bell, Ian H et al. (2014). "Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library CoolProp." In: *Industrial & engineering chemistry research* 53.6, pp. 2498–2508 (cit. on p. 72).

Bellotti, Thomas and Maxime Theillard (Sept. 2019). "A Coupled Level-Set and Reference Map Method for Interface Representation with Applications to Two-Phase Flows Simulation." In: *Journal of Computational Physics* 392, pp. 266–290 (cit. on pp. 203, 473).

Benedetti, Gianluca et al. (2018). "Low-Fidelity Modelling for Aerodynamic Characteristics of Re-Entry Objects." In: *Stardust Final Conference*. Springer, pp. 247–264 (cit. on p. 11).

Bennon, W.D. and F.P. Incropera (Oct. 1987). "A Continuum Model for Momentum, Heat and Species Transport in Binary Solid-Liquid Phase Change Systems—II. Application to Solidification in a Rectangular Cavity." In: *International Journal of Heat and Mass Transfer* 30.10, pp. 2171–2187 (cit. on pp. 111, 135).

Bergmann, Michael et al. (2016). "A Comparison of various Nodal Discontinuous Galerkin Methods for the 3D Euler Equations." In: *Proceedings of the VII European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2016)*. VII European Congress on Computational Methods in Applied Sciences and Engineering. Crete Island, Greece: Institute of Structural Analysis and Antiseismic Research School of Civil Engineering National Technical University of Athens (NTUA) Greece, pp. 7956–7966 (cit. on p. 160).

Bethe, H.A. and M.C. Adams (June 1959). "A Theory for the Ablation of Glassy Materials." In: *Journal of the Aerospace Sciences* 26.6, pp. 321–328 (cit. on p. 24).

Bijl, Hester et al. (2002). "Implicit time integration schemes for the unsteady compressible Navier–Stokes equations: laminar flow." In: *Journal of Computational Physics* 179.1, pp. 313–329 (cit. on pp. 341, 342, 344).

Billaud, Marie, Gérard Gallice, and Boniface Nkonga (Feb. 2011). "A Simple Stabilized Finite Element Method for Solving Two Phase Compressible–Incompressible Interface Flows." In: *Computer Methods in Applied Mechanics and Engineering* 200.9-12, pp. 1272–1290 (cit. on p. 70).

Blom, David S et al. (2016). "A comparison of Rosenbrock and ESDIRK methods combined with iterative solvers for unsteady compressible flows." In: *Advances in Computational Mathematics* 42, pp. 1401–1426 (cit. on p. 343).

Boettinger, W. J. et al. (Aug. 2002). "Phase-Field Simulation of Solidification." In: *Annual Review of Materials Research* 32.1, pp. 163–194 (cit. on p. 114).

Boger, Markus et al. (Feb. 2015). "Coupling of Compressible and Incompressible Flow Regions Using the Multiple Pressure Variables Approach." In: *Mathematical Methods in the Applied Sciences* 38.3, pp. 458–477 (cit. on p. 70).

Bond, Maurice and Henning Struchtrup (Dec. 22, 2004). "Mean Evaporation and Condensation Coefficients Based on Energy Dependent Condensation Probability." In: *Physical Review E* 70.6, p. 061605 (cit. on p. 102).

Bond, Ryan B et al. (2007). "Manufactured solution for computational fluid dynamics boundary condition verification." In: *AIAA journal* 45.9, pp. 2224–2236 (cit. on pp. 514, 528).

Boniou, Victor, Thomas Schmitt, and Aymeric Vié (2022a). "Comparison of interface capturing methods for the simulation of two-phase flow in a unified low-Mach framework." In: *International Journal of Multiphase Flow* 149, p. 103957 (cit. on p. 200).

– (Dec. 2022b). "Consistency and Accuracy in the Simulation of Two-Phase Flows with Phase Change Using Sharp Interface Capturing Methods." In: *Journal of Computational Physics* 470, p. 111604 (cit. on p. 101).

Bordes, M Cruellas (2019). *A Study of an Artificial Viscosity Technique for High-Order Discontinuous Galerkin Methods* (cit. on p. 303).

Brackbill, Jeremiah U, Douglas B Kothe, and Charles Zemach (1992). "A continuum method for modeling surface tension." In: *Journal of computational physics* 100.2, pp. 335–354 (cit. on p. 242).

Brady, PT, Marcus Herrmann, and JM Lopez (2012). "Code verification for finite volume multiphase scalar equations using the method of manufactured solutions." In: *Journal of Computational Physics* 231.7, pp. 2924–2944 (cit. on p. 515).

braintruffle (2022). *braintruffle*. Youtube. URL: https://www.youtube.com/@braintruffle (cit. on p. 46).

Brennen, Christopher E. (2005). *Fundamentals of Multiphase Flow*. Cambridge University Press (cit. on p. 532).

Brglez, Špela (2014). "Code verification for governing equations with arbitrary functions using adjusted method of manufactured solutions." In: *Engineering with Computers* 30.4, pp. 669–678 (cit. on pp. 518, 528).

Bronshten, V. A. (1983). *Physics of Meteoric Phenomena*. Dordrecht: Springer Netherlands (cit. on pp. 26, 27).

Buelow, P.E.O., S. Venkateswaran, and C.L. Merkle (1994). "Effect of grid aspect ratio on convergence." In: *AIAA journal* 32.12, pp. 2401–2408 (cit. on p. 191).

Buresti, Guido (Oct. 2015). "A Note on Stokes' Hypothesis." In: *Acta Mechanica* 226.10, pp. 3555–3559 (cit. on p. 63).

Burman, Erik (2010). "Ghost penalty." In: *Comptes Rendus Mathematique* 348.21-22, pp. 1217–1220 (cit. on p. 272).

Burman, Erik, Stefan Frei, and Andre Massing (Feb. 2022). "Eulerian Time-Stepping Schemes for the Non-Stationary Stokes Equations on Time-Dependent Domains." In: *Numerische Mathematik* 150.2, pp. 423–478 (cit. on p. 354).

Burman, Erik and Paolo Zunino (2011). "Numerical Approximation of Large Contrast Problems with the Unfitted Nitsche Method." In: *Frontiers in Numerical Analysis - Durham 2010*. Ed. by James Blowey and Max Jensen. Vol. 85. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 227–282 (cit. on pp. 273, 326).

Burman, Erik et al. (Nov. 16, 2015). "CutFEM: Discretizing Geometry and Partial Differential Equations." In: *International Journal for Numerical Methods in Engineering* 104.7, pp. 472–501 (cit. on pp. 270, 272).

Burman, Erik et al. (Jan. 2021). "An Unfitted Hybrid High-Order Method with Cell Agglomeration for Elliptic Interface Problems." In: *SIAM Journal on Scientific Computing* 43.2, A859–A882 (cit. on p. 275).

Cagnone, Jean Sébastien, Koen Hillewaert, and Nicolas Poletz (May 2014). "A Discontinuous Galerkin Method for Multiphysics Welding Simulations." In: *Key Engineering Materials*, pp. 1319–1326 (cit. on p. 132).

Caiden, Rachel, Ronald P. Fedkiw, and Chris Anderson (Jan. 2001). "A Numerical Method for Two-Phase Flow Consisting of Separate Compressible and Incompressible Regions." In: *Journal of Computational Physics* 166.1, pp. 1–27 (cit. on p. 70).

Cakir, Bora O. et al. (2022). "On the design of a Mach 5 axisymmetric contoured nozzle for the VKI H3 hypersonic tunnel." In: *2nd International Conference on High-Speed Vehicle Science & Technology, Bruges, Belgium* (cit. on pp. 443, 444).

Carnevali, Emanuele et al. (2023). "On the performance of different sets of variables for the discontinuous Galerkin discretization of the Euler equations." 22nd IACM Computational Fluids Conference (cit. on p. 183).

Carslaw, H.S. and J.C. Jaeger (1959). *Conduction of Heat in Solids*. Oxford at the Clarednon Press (cit. on p. 430).

Carton de Wiart, Corentin (2014). "Towards a discontinuous Galerkin solver for scale-resolving simulations of moderate Reynolds number flows, and application to industrial cases." PhD thesis. Université Catholique de Louvain (cit. on p. 149).

Casey, James (2011). "On the derivation of jump conditions in continuum mechanics." In: *The International Journal of Structural Changes in Solids* 3.2, pp. 61–84 (cit. on pp. 90, 94).

Cash, Jeff R (1983). "The integration of stiff initial value problems in ODEs using modified extended backward differentiation formulae." In: *Computers & mathematics with applications* 9.5, pp. 645–657 (cit. on p. 343).

Cattani, BM et al. (2021). "An Overview of Design for Demise Technologies." In: *Proceedings of the 8th European Conference on Space Debris, Darmstadt, Germany*, pp. 20–23 (cit. on p. 10).

Cermelli, Paolo, Eliot Fried, and Morton E Gurtin (2005). "Transport relations for surface integrals arising in the formulation of balance laws for evolving fluid interfaces." In: *Journal of Fluid Mechanics* 544, pp. 339–351 (cit. on p. 94).

Chalons, Christophe, Christian Rohde, and Maria Wiebe (Sept. 2017). "A Finite Volume Method for Undercompressive Shock Waves in Two Space Dimensions." In: *ESAIM: Mathematical Modelling and Numerical Analysis* 51.5, pp. 1987–2015 (cit. on p. 267).

Chan, Jesse, David C. Del Rey Fernandez, and Mark H. Carpenter (Aug. 3, 2019). *Efficient Entropy Stable Gauss Collocation Methods*. arXiv: 1809.01178 [cs, math]. URL: http://arxiv.org/abs/1809.01178 (visited on 06/11/2023). preprint (cit. on p. 159).

Chang, Chih-Hao, Xiaolong Deng, and Theo G. Theofanous (June 2013). "Direct Numerical Simulation of Interfacial Instabilities: A Consistent, Conservative, All-Speed, Sharp-Interface Method." In: *Journal of Computational Physics* 242, pp. 946–990 (cit. on pp. 86, 104, 243, 268, 317).

Chauchat, Nicolas, Roland Becker, and Eric Schall (Apr. 2021). "Performance of DG Methods Based on Different Variables for Low Mach Number Flows." In: *Communications in Nonlinear Science and Numerical Simulation* 95, p. 105580 (cit. on pp. 182, 183, 185).

Chazot, Olivier (2020). "Aerospace Flight Modeling and Experimental Testing." In: *Uncertainty in Engineering: Introduction to Methods and Applications*. Ed. by L.J.M. Aslett, F.P.A. Coolen, and J. De Bock. Springer, pp. 131–147 (cit. on p. 439).

Chen, C.-J. (1967). "Melting ablation for two-dimensional and axisymmetric blunt bodies with a body force." PhD thesis. Case Institute of Technology (cit. on p. 440).
Chen, Ching-Jen and Simon Ostrach (1971). "Low-Temperature Simulation of Hypersonic Melting Ablation and the Observed Wave Patterns." In: *AIAA Journal* 9.6, pp. 1120–1125 (cit. on pp. 441, 538).
Chen, Gang (2022). "On the molecular picture and interfacial temperature discontinuity during evaporation and condensation." In: *International Journal of Heat and Mass Transfer* 191, p. 122845 (cit. on p. 104).
Chen, Shih-Yuan and Stuart J. Allen (Oct. 1962). "Similarity Analysis for Transient Melting and Vaporizing Ablation." In: *ARS Journal* 32.10, pp. 1536–1543 (cit. on pp. 537, 538).
Chen, Shu-sheng et al. (2020). "An improved AUSM-family scheme with robustness and accuracy for all Mach number flows." In: *Applied Mathematical Modelling* 77, pp. 1065–1081 (cit. on p. 196).
Chen, Shusheng et al. (2020). "HLLC+: low-Mach shock-stable HLLC-type Riemann solver for all-speed flows." In: *SIAM Journal on Scientific Computing* 42.4, B921–B950 (cit. on p. 196).
Chen, T., P. D. Minev, and K. Nandakumar (May 10, 2004). "A Projection Scheme for Incompressible Multiphase Flow Using Adaptive Eulerian Grid." In: *International Journal for Numerical Methods in Fluids* 45.1, pp. 1–19 (cit. on p. 254).
Cheng, K-W and T-P Fries (2012). "XFEM with hanging nodes for two-phase incompressible flow." In: *Computer Methods in Applied Mechanics and Engineering* 245, pp. 290–312 (cit. on p. 244).
Chevaugeon, Nicolas et al. (2007). "Optimal numerical parameterization of discontinuous Galerkin method applied to wave propagation problems." In: *Journal of Computational Physics* 223.1, pp. 188–207 (cit. on p. 149).
Chopp, David L. (Jan. 2001). "Some Improvements of the Fast Marching Method." In: *SIAM Journal on Scientific Computing* 23.1, pp. 230–244 (cit. on p. 212).
– (Jan. 2009). "Another Look at Velocity Extensions in the Level Set Method." In: *SIAM Journal on Scientific Computing* 31.5, pp. 3255–3273 (cit. on p. 221).
Chopp, David Layne (1991). "Computing minimal surfaces via level set curvature flow." PhD thesis. University of California, Berkeley (cit. on p. 229).
Choudhary, Aniruddha et al. (2016). "Code verification of boundary conditions for compressible and incompressible computational fluid dynamics codes." In: *Computers & Fluids* 126, pp. 153–169 (cit. on p. 514).
Chouly, Franz (2022). "A review on some discrete variational techniques for the approximation of essential boundary conditions." In: (cit. on p. 273).
Cochran, Thomas H and Simon Ostrach (1968). *Body rotation effects on melting ablation*. Tech. rep. NASA (cit. on pp. 19, 440, 441, 538, 539, 542).
Cockburn, Bernardo and Chi-Wang Shu (1998). "The local discontinuous Galerkin method for time-dependent convection-diffusion systems." In: *SIAM journal on numerical analysis* 35.6, pp. 2440–2463 (cit. on p. 170).

Colomés, Oriol et al. (Jan. 2021). "A Weighted Shifted Boundary Method for Free Surface Flow Problems." In: *Journal of Computational Physics* 424, p. 109837 (cit. on p. 415).

Colonna, P. and S. Rebay (Nov. 10, 2004). "Numerical Simulation of Dense Gas Flows on Unstructured Grids with an Implicit High Resolution Upwind Euler Solver." In: *International Journal for Numerical Methods in Fluids* 46.7, pp. 735–765 (cit. on p. 485).

Compère, Gaëtan (2009). "Mesh adaptation with large deformations: theory, algorithms and implementation." PhD thesis. UCL-Université Catholique de Louvain (cit. on p. 472).

Coquerelle, Mathieu and Stéphane Glockner (Jan. 2016). "A Fourth-Order Accurate Curvature Computation in a Level Set Framework for Two-Phase Flows Subjected to Surface Tension Forces." In: *Journal of Computational Physics* 305, pp. 838–876 (cit. on p. 253).

Cordesse, Pierre (2020). "Contribution to the study of combustion instabilities in cryotechnic rocket engines: coupling diffuse interface models with kinetic-based moment methods for primary atomization simulations." PhD thesis. Université Paris-Saclay (cit. on pp. 115, 117).

Cordesse, Pierre et al. (2019). *Derivation of a two-phase flow model with two-scale kinematics, geometric variables and surface tension using variational calculus*. Preprint at https://arxiv.org/abs/1910.14557 (cit. on p. 243).

Corrigan, Andrew, Andrew D. Kercher, and David A. Kessler (Mar. 30, 2019). "A Moving Discontinuous Galerkin Finite Element Method for Flows with Interfaces." In: *International Journal for Numerical Methods in Fluids* 89.9, pp. 362–406 (cit. on p. 199).

Coulaud, Olivier and Adrien Loseille (2016). "Very high order anisotropic metric-based mesh adaptation in 3D." In: *Procedia engineering* 163, pp. 353–365 (cit. on p. 472).

Courant, Richard, Eugene Isaacson, and Mina Rees (1952). "On the solution of nonlinear hyperbolic differential equations by finite differences." In: *Communications on pure and applied mathematics* 5.3, pp. 243–255 (cit. on p. 167).

Cowperthwaite, M (1969). "Relationships between incomplete equations of state." In: *Journal of the Franklin Institute* 287.5, pp. 379–387 (cit. on p. 65).

Crockett, RK, Phillip Colella, and Daniel T Graves (2011). "A Cartesian grid embedded boundary method for solving the Poisson and heat equations with discontinuous coefficients in three dimensions." In: *Journal of Computational Physics* 230.7, pp. 2451–2469 (cit. on p. 515).

Cuendet, Michel (2008). *Molecular Dynamics Simulation - A short introduction* (cit. on pp. 487, 498).

Cui, T. et al. (n.d.). *PHG (0.9.7)*, http://lsec.cc.ac.cn/phg/download/ (cit. on p. 286).

Cui, Tao et al. (Mar. 31, 2020). "High-Order Numerical Quadratures in a Tetrahedron with an Implicitly Defined Curved Interface." In: *ACM Transactions on Mathematical Software* 46.1, pp. 1–18 (cit. on p. 286).

Daru, V. et al. (Nov. 2010). "A Numerical Method for the Simulation of Low Mach Number Liquid–Gas Flows." In: *Journal of Computational Physics* 229.23, pp. 8844–8867 (cit. on p. 70).

Darwish, Marwan, Ihab Sraj, and Fadl Moukalled (2009). "A coupled finite volume solver for the solution of incompressible flows on unstructured grids." In: *Journal of Computational Physics* 228.1, pp. 180–201 (cit. on p. 180).

Das, Pratik and H.S. Udaykumar (Mar. 2020). "A Sharp-Interface Method for the Simulation of Shock-Induced Vaporization of Droplets." In: *Journal of Computational Physics* 405, p. 109005 (cit. on pp. 316, 320).

Daw, Murray S, Stephen M Foiles, and Michael I Baskes (1993). "The embedded-atom method: a review of theory and applications." In: *Materials Science Reports* 9.7-8, pp. 251–310 (cit. on p. 495).

De Boer, P.C.T. (Dec. 1984). "Thermally Driven Motion of Strongly Heated Fluids." In: *International Journal of Heat and Mass Transfer* 27.12, pp. 2239–2251 (cit. on p. 68).

De Lorenzo, Marco et al. (2017). "Homogeneous two-phase flow models and accurate steam-water table look-up method for fast transient simulations." In: *International journal of multiphase flow* 95, pp. 199–219 (cit. on p. 72).

Delaissé, Nicolas et al. (2023). "Quasi-Newton Methods for Partitioned Simulation of Fluid–Structure Interaction Reviewed in the Generalized Broyden Framework." In: *Archives of Computational Methods in Engineering*, pp. 1–30 (cit. on p. 315).

Delhaye, Jean-Marc (1974). "Jump conditions and entropy sources in two-phase systems. Local instant formulation." In: *International Journal of Multiphase Flow* 1.3, pp. 395–409 (cit. on pp. 90, 99).

Deng, Xiao-Long and Maojun Li (Mar. 2018). "Simulating Compressible Two-Medium Flows with Sharp-Interface Adaptive Runge–Kutta Discontinuous Galerkin Methods." In: *Journal of Scientific Computing* 74.3, pp. 1347–1368 (cit. on pp. 268, 317).

Denis, Flavien (2022). "Study of the ablation by melting in hypersonic flow." PhD thesis. Université de Strasbourg (cit. on p. 449).

Denis, Flavien et al. (2022). "Experimental Investigation of Thermal Ablation by Melting in a Hypersonic Shock Tunnel." In: *Journal of Spacecraft and Rockets* 59.4, pp. 1406–1411 (cit. on pp. 448, 451).

Denner, Fabian (2018). "Fully-coupled pressure-based algorithm for compressible flows: Linearisation and iterative solution strategies." In: *Computers & Fluids* 175, pp. 53–65 (cit. on p. 183).

Denner, Fabian, Fabien Evrard, and Berend GM van Wachem (2020). "Conservative finite-volume framework and pressure-based algorithm for flows of incompressible, ideal-gas and real-gas fluids at all speeds." In: *Journal of Computational Physics* 409, p. 109348 (cit. on p. 179).

De Prenter, Frits et al. (Aug. 17, 2022). *Stability and Conditioning of Immersed Finite Element Methods: Analysis and Remedies*. arXiv: 2208.08538 [cs, math]. URL: http://arxiv.org/abs/2208.08538 (visited on 05/13/2023). preprint (cit. on p. 271).

Di Pietro, Daniele Antonio and Alexandre Ern (2012). *Mathematical Aspects of Discontinuous Galerkin Methods*. Vol. 69. Mathématiques et Applications. Berlin, Heidelberg: Springer Berlin Heidelberg (cit. on p. 164).

Dias, Bruno (2020). "Thermal ablation and radiation modeling of meteor phenomena." PhD thesis. UCL-Université Catholique de Louvain (cit. on pp. 102, 103).

Dias, Bruno et al. (2016). "Development of a Melting Model for Meteors." In: 30TH INTERNATIONAL SYMPOSIUM ON RAREFIED GAS DYNAMICS: RGD 30. Victoria, BC, Canada, p. 160004 (cit. on p. 26).

Donea, Jean et al. (2004). "Arbitrary Lagrangian–Eulerian Methods." In: *Encyclopedia of Computational Mechanics*. John Wiley & Sons, Ltd. Chap. 14. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470091355.ecm009 (cit. on p. 48).

Dorrance, William H. (2017). *Viscous Hypersonic Flow: Theory of Reacting and Hypersonic Boundary Layers*. Dover edition. Mineola, New York: Dover Publications, Inc. 334 pp. (cit. on p. 30).

Drew, Donald A (1983). "Mathematical modeling of two-phase flow." In: *Annual review of fluid mechanics* 15.1, pp. 261–291 (cit. on p. 116).

Dreyer, Wolfgang (2005). "On jump conditions at phase boundaries for ordered and disordered phases." In: *Weierstrass -Institut für Angewandte Analysis und Stochastik* (cit. on pp. 90, 91, 98, 100).

Dror, Ron (2017). *Molecular Dynamics Simulation* (cit. on p. 489).

Duffa, Georges (2013). *Ablative Thermal Protection Systems Modeling*. In collab. with American Institute of Aeronautics and Astronautics. AIAA Education Series. Reston, Virginia: American Institute of Aeronautics and Astronautics, Inc. 431 pp. (cit. on p. 29).

Dumbser, Michael (Jan. 2010). "Arbitrary High Order PNPM Schemes on Unstructured Meshes for the Compressible Navier–Stokes Equations." In: *Computers & Fluids* 39.1, pp. 60–76 (cit. on p. 245).

Dumbser, Michael et al. (2014). "A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws." In: *Journal of Computational Physics* 278, pp. 47–75 (cit. on p. 169).

Dziuk, Gerhard and Charles M Elliott (2007). "Finite elements on evolving surfaces." In: *IMA journal of numerical analysis* 27.2, pp. 262–292 (cit. on pp. 94, 95).

Eça, L, M Hoekstra, and G Vaz (2012). "Manufactured solutions for steady-flow Reynolds-averaged Navier–Stokes solvers." In: *International Journal of Computational Fluid Dynamics* 26.5, pp. 313–332 (cit. on p. 515).

Edling, Walter H and Simon Ostrach (1966). *An Experimental Study of Melting Wave Behavior*. Tech. rep. NASA (cit. on pp. 440, 441, 538, 539, 541).

Einfeldt, Bernd (1988). "On Godunov-type methods for gas dynamics." In: *SIAM Journal on numerical analysis* 25.2, pp. 294–318 (cit. on p. 168).

Engwer, Christian (2009). "An Unfitted Discontinuous Galerkin Scheme for Micro-scale Simulations and Numerical Upscaling." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 269, 272).

Engwer, Christian and Felix Heimann (2012). "Dune-udg: A cut-cell framework for unfitted discontinuous galerkin methods." In: *Advances in DUNE*. Springer, pp. 89–100 (cit. on p. 276).

Engwer, Christian et al. (Jan. 2020). "A Stabilized DG Cut Cell Method for Discretizing the Linear Transport Equation." In: *SIAM Journal on Scientific Computing* 42.6, A3677–A3703 (cit. on p. 273).

Enright, Douglas et al. (2002). "A hybrid particle level set method for improved interface capturing." In: *Journal of Computational physics* 183.1, pp. 83–116 (cit. on p. 203).

Ervik, Åsmund, Karl Yngve Lervåg, and Svend Tollak Munkejord (2014). "A robust method for calculating interface curvature and normal vectors using an extracted local level set." In: *Journal of Computational Physics* 257, pp. 259–277 (cit. on pp. 204, 228).

ESA, ESA Re-entry Safety working group (2017). *ESA Re-entry Safety Requirements*. 1. Version Revision 0 (cit. on p. 6).

ESA, European Space Agency (2020). *DIVE - Guidelines for Analysing and Testing the Demise of Man Made Space Objects During Re-entry*. Tech. rep. ESA - ESTEC (cit. on pp. 22, 24).

ESA, The ESA Space Debris Office (2023). *ESA'S Annual Space Environment Report*. GEN-DB-LOG-00288-OPS-SD. Version Revision 7.1 (cit. on pp. 4, 5).

Faghri, Amir and Yuwen Zhang (2006). *Transport Phenomena in Multiphase Systems*. Burlington, Mass: Elsevier Academic Press. 1030 pp. (cit. on pp. 26, 90, 99).

Fagnani, Andrea (2023). "Development of Measurement Techniques and Study of the Aerothermal Response of Space Debris Materials to Atmospheric Entry Plasmas." PhD thesis. Vrije Universiteit Brussel (cit. on pp. 440, 470).

Farhat, Charbel, Jean-Frédéric Gerbeau, and Arthur Rallu (Aug. 2012). "FIVER: A Finite Volume Method Based on Exact Two-Phase Riemann Problems and Sparse Grids for Multi-Material Flows with Large Density Jumps." In: *Journal of Computational Physics* 231.19, pp. 6360–6379 (cit. on p. 268).

Farrell, P.E. and J.R. Maddison (Jan. 2011). "Conservative Interpolation between Volume Meshes by Local Galerkin Projection." In: *Computer Methods in Applied Mechanics and Engineering* 200.1-4, pp. 89–100 (cit. on p. 310).

Fechter, S., F. Jaegle, and V. Schleper (Apr. 2013). "Exact and Approximate Riemann Solvers at Phase Boundaries." In: *Computers & Fluids* 75, pp. 112–126 (cit. on pp. 320, 322, 407).

Fechter, S. and C.-D. Munz (July 7, 2015). "A Discontinuous Galerkin-based Sharp-Interface Method to Simulate Three-Dimensional Compressible Two-Phase Flow." In: *International Journal for Numerical Methods in Fluids* 78.7, pp. 413–435 (cit. on p. 245).

Fechter, Stefan (2015). "Compressible multi-phase simulation at extreme conditions using a discontinuous Galerkin scheme." PhD thesis. University of Stuttgart (cit. on pp. 86, 323, 324).

Fechter, Stefan et al. (Oct. 20, 2015). *A Sharp Interface Method for Compressible Liquid-Vapor Flow with Phase Transition and Surface Tension*. arXiv: 1511.

03612 [physics]. URL: http://arxiv.org/abs/1511.03612 (visited on 05/13/2023). preprint (cit. on pp. 88, 102).
– (June 2018). "Approximate Riemann Solver for Compressible Liquid Vapor Flow with Phase Transition and Surface Tension." In: *Computers & Fluids* 169, pp. 169–185 (cit. on pp. 103, 104, 243, 321).
Fedkiw, Ronald P et al. (July 1999). "A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method)." In: *Journal of Computational Physics* 152.2, pp. 457–492 (cit. on pp. 243, 267, 316).
Feldman, Saul (1959). "On the instability theory of the melted surface of an ablating body when entering the atmosphere." In: *Journal of Fluid Mechanics* 6.1, pp. 131–155 (cit. on p. 441).
Fernández, Pablo (2019). "Entropy-stable hybridized discontinuous Galerkin methods for large-eddy simulation of transitional and turbulent flows." PhD thesis. Massachusetts Institute of Technology (cit. on p. 158).
Fidkowski, Krzysztof (2023). "Residual-Based Time-Step Control for High-Order Discretizations." In: *AIAA SCITECH 2023 Forum*, p. 2294 (cit. on pp. 341, 344, 472).
Folkner, David, Aaron Katz, and Venke Sankaran (2014). "Design and verification methodology of boundary conditions for finite volume schemes." In: *Computers & Fluids* 96, pp. 264–275 (cit. on p. 514).
Föll, F. et al. (July 2019). "On the Use of Tabulated Equations of State for Multi-Phase Simulations in the Homogeneous Equilibrium Limit." In: *Shock Waves* 29.5, pp. 769–793 (cit. on pp. 72, 88).
Föll, Fabian et al. (2020). "A Novel Regularization Strategy for the Local Discontinuous Galerkin Method for Level-Set Reinitialization." arXiv: 2007.06883 [cs, math] (cit. on p. 204).
Francois, Marianne M. et al. (Mar. 2006). "A Balanced-Force Algorithm for Continuous and Sharp Interfacial Surface Tension Models within a Volume Tracking Framework." In: *Journal of Computational Physics* 213.1, pp. 141–173 (cit. on p. 243).
Franquet, Erwin and Vincent Perrier (June 2012). "Runge–Kutta Discontinuous Galerkin Method for the Approximation of Baer and Nunziato Type Multiphase Models." In: *Journal of Computational Physics* 231.11, pp. 4096–4141 (cit. on p. 269).
Fredriksson, Hasse and Ulla Åkerlind (Aug. 3, 2012). *Solidification and Crystallization Processing in Metals and Alloys*. 1st ed. Wiley (cit. on p. 110).
Frenkel, Daan and Berend Smit (2001). *Understanding molecular simulation: from algorithms to applications*. Vol. 1. Elsevier (cit. on pp. 489–491, 496).
Freno, Brian A, Brian R Carnes, and Neil R Matula (2021). "Nonintrusive manufactured solutions for ablation." In: *Physics of Fluids* 33.1, p. 017104 (cit. on p. 515).
Freno, Brian A et al. (2022). "Nonintrusive manufactured solutions for non-decomposing ablation in two dimensions." In: *Journal of Computational Physics* 463, p. 111237 (cit. on pp. 514, 515, 518, 528).

Fries, Thomas-Peter and Ted Belytschko (Oct. 15, 2010). "The Extended/Generalized Finite Element Method: An Overview of the Method and Its Applications." In: *International Journal for Numerical Methods in Engineering* 84.3, pp. 253–304 (cit. on p. 270).

Fries, Thomas-Peter and Andreas Zilian (July 2, 2009). "On Time Integration in the XFEM." In: *International Journal for Numerical Methods in Engineering* 79.1, pp. 69–93 (cit. on pp. 354, 355).

Fuentes, Irene Pontijas et al. (2017). "Upgrade of ESA's Debris Risk Assessment and Mitigation Analysis (DRAMA) tool: Spacecraft Entry Survival Analysis Module." In: *Acta Astronautica* (cit. on pp. 13, 14).

Fujimoto, Keiichiro et al. (2017). "High-Fidelity Numerical Simulations for Destructive Re-entry of Upper Stages." In: *7th European Conference on Space Debris* (cit. on p. 16).

Furfaro, Damien and Richard Saurel (2015). "A simple HLLC-type Riemann solver for compressible non-equilibrium two-phase flows." In: *Computers & Fluids* 111, pp. 159–178 (cit. on p. 269).

Gaburro, Elena et al. (Apr. 2020). "High Order Direct Arbitrary-Lagrangian-Eulerian Schemes on Moving Voronoi Meshes with Topology Changes." In: *Journal of Computational Physics* 407, p. 109167 (cit. on p. 199).

Gamnitzer, Peter (2010). "Residual-based variational multiscale methods for turbulent flows and fluid-structure interaction." PhD thesis. Technische Universität München (cit. on pp. 48, 55).

Garbacz, C., F. Morgado, and M. Fossati (2022). "Effect of thermal nonequilibrium on the shock interaction mechanism for carbon dioxide mixtures on double-wedge geometries." In: *Physics of Fluids* 34.2 (cit. on p. 20).

Gassner, Gregor, Frieder Lörcher, and Claus-Dieter Munz (June 2007). "A Contribution to the Construction of Diffusion Fluxes for Finite Volume and Discontinuous Galerkin Schemes." In: *Journal of Computational Physics* 224.2, pp. 1049–1063 (cit. on p. 320).

Gassner, Gregor J and Andrew R Winters (2021). "A novel robust strategy for discontinuous Galerkin methods in computational fluid mechanics: Why? When? What? Where?" In: *Frontiers in Physics* 8, p. 500690 (cit. on p. 159).

Ge, Zhouyang et al. (Jan. 2018). "An Efficient Mass-Preserving Interface-Correction Level Set/Ghost Fluid Method for Droplet Suspensions under Depletion Forces." In: *Journal of Computational Physics* 353, pp. 435–459 (cit. on p. 202).

Geisenhofer, Markus, Florian Kummer, and Björn Müller (Nov. 30, 2019). "A Discontinuous Galerkin Immersed Boundary Solver for Compressible Flows: Adaptive Local Time Stepping for Artificial Viscosity–Based Shock-capturing on Cut Cells." In: *International Journal for Numerical Methods in Fluids* 91.9, pp. 448–472 (cit. on p. 304).

Geisenhofer, Markus, Florian Kummer, and Martin Oberlack (Dec. 17, 2020). *An Extended Discontinuous Galerkin Method for High-Order Shock-Fitting*. arXiv: 2012.08860 [physics]. URL: http://arxiv.org/abs/2012.08860 (visited on 05/13/2023). preprint (cit. on p. 304).

Gelhard, Tobias et al. (2005). "Stabilized finite element schemes with LBB-stable elements for incompressible flows." In: *Journal of computational and applied mathematics* 177.2, pp. 243–267 (cit. on p. 180).

Georges, Laurent, Grégoire Winckelmans, and Philippe Geuzaine (2008). "Improving shock-free compressible RANS solvers for LES on unstructured meshes." In: *Journal of Computational and Applied Mathematics* 215.2, pp. 419–428 (cit. on p. 149).

Gerasimov, Denis N. and Eugeny I. Yurin (2018). *Kinetics of Evaporation*. Vol. 68. Springer Series in Surface Sciences. Cham: Springer International Publishing (cit. on p. 102).

Geuzaine, Christophe and Jean-François Remacle (2009). "Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities." In: *International journal for numerical methods in engineering* 79.11, pp. 1309–1331 (cit. on p. 378).

Ghasemi, Fatemeh and Jan Nordström (Nov. 2019). "An Energy Stable Coupling Procedure for the Compressible and Incompressible Navier-Stokes Equations." In: *Journal of Computational Physics* 396, pp. 280–302 (cit. on p. 70).

Giacomini, Matteo, Ruben Sevilla, and Antonio Huerta (May 2021). "HDGlab: An Open-Source Implementation of the Hybridisable Discontinuous Galerkin Method in MATLAB." In: *Archives of Computational Methods in Engineering* 28.3, pp. 1941–1986 (cit. on p. 159).

Giovangigli, Vincent (July 6, 2020). "Kinetic Derivation of Diffuse-Interface Fluid Models." In: *Physical Review E* 102.1, p. 012110 (cit. on p. 83).

Girin, A.G. and N.Kh. Kopyt (1994). "Generation of fine aerosol as a dispersion of unstable fusion film of ablating meteoroid." In: *Journal of aerosol science* 25.7, pp. 1353–1357 (cit. on p. 28).

Girin, O.G. (Oct. 2017). "A Hydrodynamic Mechanism of Meteor Ablation: The Melt-Spraying Model⋆." In: *Astronomy & Astrophysics* 606, A63 (cit. on pp. 25, 26).

Glimm, James et al. (Dec. 4, 2001). "Conservative Front Tracking and Level Set Algorithms." In: *Proceedings of the National Academy of Sciences* 98.25, pp. 14198–14201 (cit. on p. 274).

Godounov, SK (1959). "A difference method for numerical calculation of discontinuous solutions of the equation of hydrodynamics." In: *Matematicheskii Sbornik* 47.89-3, pp. 271–306 (cit. on p. 167).

Gogoi, A and Jadav Chandra Mandal (2022). "A simple HLLE-type scheme for all Mach number flows." In: *arXiv preprint arXiv:2207.06830* (cit. on p. 196).

Goldman, Ron (Oct. 2005). "Curvature Formulas for Implicit Curves and Surfaces." In: *Computer Aided Geometric Design* 22.7, pp. 632–658 (cit. on p. 206).

Gomes, José and Olivier Faugeras (2000). "Reconciling distance functions and level sets." In: *Journal of Visual Communication and Image Representation* 11.2, pp. 209–223 (cit. on p. 206).

Grassi, Lilith et al. (2017). "Design for Demise techniques for medium/large LEO satellites reentry." In: *7th European Conference on Space Debris*. Vol. 7. European Space Agency (cit. on p. 10).

Grier, Benjamin et al. (2015). "Discontinuous solutions using the method of manufactured solutions on finite volume solvers." In: *AIAA Journal* 53.8, pp. 2369–2378 (cit. on pp. 515, 528).

Gross, Sven, Maxim A Olshanskii, and Arnold Reusken (2015). "A trace finite element method for a class of coupled bulk-interface transport problems." In: *ESAIM: Mathematical modelling and numerical analysis* 49.5, pp. 1303–1330 (cit. on p. 97).

Gross, Sven and Arnold Reusken (Jan. 2007). "Finite Element Discretization Error Analysis of a Surface Tension Force in Two-Phase Incompressible Flows." In: *SIAM Journal on Numerical Analysis* 45.4, pp. 1679–1700 (cit. on p. 244).

Grossir, G and O Chazot (2022). "Design of axisymmetric contoured nozzles for calorically and thermallyimperfect gases using the HYPNOZE code." In: *2nd International Conference on High-SpeedVehicle Science and Technology, Bruges, Belgium*, pp. 1–16 (cit. on p. 443).

Grossir, Guillaume et al. (2020). "Aerodynamic characterization of space debris in the VKI Longshot hypersonic tunnel using a free-flight measurement technique." In: *Experiments in Fluids* 61, pp. 1–13 (cit. on pp. 18, 19).

Grüneisen, Eduard (1912). "Theorie des festen Zustandes einatomiger Elemente." In: *Annalen der Physik* 344.12, pp. 257–306 (cit. on p. 74).

Guédon, Gaël (2013). "Two-Phase Heat and Mass Transfer Modeling: Flexible Numerical Methods for Energy Engineering Analyses." PhD thesis. Politecnico di Milano (cit. on p. 90).

Guillard, Hervé and Angelo Murrone (2004). "On the behavior of upwind schemes in the low Mach number limit: II. Godunov type schemes." In: *Computers & fluids* 33.4, pp. 655–675 (cit. on pp. 189, 193).

Guillard, Hervé and Cécile Viozat (1999). "On the behaviour of upwind schemes in the low Mach number limit." In: *Computers & Fluids* 28.1, pp. 63–86 (cit. on p. 189).

Gürkan, Ceren, Martin Kronbichler, and Sonia Fernández-Méndez (Feb. 17, 2019). "eXtended Hybridizable Discontinuous Galerkin for Incompressible Flow Problems with Unfitted Meshes and Interfaces." In: *International Journal for Numerical Methods in Engineering* 117.7, pp. 756–777 (cit. on p. 275).

Gürkan, Ceren and André Massing (May 2019). "A Stabilized Cut Discontinuous Galerkin Framework for Elliptic Boundary Value and Interface Problems." In: *Computer Methods in Applied Mechanics and Engineering* 348, pp. 466–499 (cit. on p. 272).

Gürkan, Ceren, Simon Sticko, and André Massing (Jan. 2020). "Stabilized Cut Discontinuous Galerkin Methods for Advection-Reaction Problems." In: *SIAM Journal on Scientific Computing* 42.5, A2620–A2654 (cit. on p. 273).

Gutiérrez-Jorquera, Juan and Florian Kummer (Apr. 2022). "A Fully Coupled High-order Discontinuous Galerkin Method for Diffusion Flames in a

low-Mach Number Framework." In: *International Journal for Numerical Methods in Fluids* 94.4, pp. 316–345 (cit. on p. 344).

Haas, J.-F. and B. Sturtevant (Sept. 1987). "Interaction of Weak Shock Waves with Cylindrical and Spherical Gas Inhomogeneities." In: *Journal of Fluid Mechanics* 181 (-1), p. 41 (cit. on p. 411).

Haegeman, Ward (2021). *Study of the impact of a gas jet on a free water surface* (cit. on p. 119).

– (2022). *Implementation of a curvature reconstruction algorithm for a high-order and sharp interface multiphase flow solver* (cit. on pp. xix, 241).

Hakimi, Nouredine (1997). "Preconditioning methods for time dependent Navier-Stokes equations." PhD thesis. Vrije Universiteit Brussels (cit. on p. 192).

Hansbo, Anita and Peter Hansbo (2002). "An unfitted finite element method, based on Nitsche's method, for elliptic interface problems." In: *Computer methods in applied mechanics and engineering* 191.47-48, pp. 5537–5552 (cit. on p. 270).

Hardy, Baptiste et al. (2022). "Simulation of the flow past random arrays of spherical particles: Microstructure-based tensor quantities as a tool to predict fluid–particle forces." In: *International Journal of Multiphase Flow* 149, p. 103970 (cit. on p. 86).

Hartmann, Ralf and Paul Houston (Nov. 2008). "An Optimal Order Interior Penalty Discontinuous Galerkin Discretization of the Compressible Navier–Stokes Equations." In: *Journal of Computational Physics* 227.22, pp. 9670–9685 (cit. on pp. 157, 171, 328, 330, 392).

Hartmann, Ralph (2008). "Numerical Analysis of Higher Order Discontinuous Galerkin Finite Element Methods." In: *VKI LS 2008-08: CFD - ADIGMA course on very high order discretization methods*. Rhode Saint Genese, Belgium: von Karman Institute for Fluid Dynamics (cit. on pp. 165, 196, 334).

Hashemi, Mohammad R., Pavel B. Ryzhakov, and Riccardo Rossi (Oct. 2020). "An Enriched Finite Element/Level-Set Method for Simulating Two-Phase Incompressible Fluid Flows with Surface Tension." In: *Computer Methods in Applied Mechanics and Engineering* 370, p. 113277 (cit. on pp. 243, 250).

Hauke, Guillermo and Thomas J.R. Hughes (Jan. 1998). "A Comparative Study of Different Sets of Variables for Solving Compressible and Incompressible Flows." In: *Computer Methods in Applied Mechanics and Engineering* 153.1-2, pp. 1–44 (cit. on p. 182).

Heimann, F. et al. (Jan. 30, 2013). "An Unfitted Interior Penalty Discontinuous Galerkin Method for Incompressible Navier–Stokes Two-Phase Flow." In: *International Journal for Numerical Methods in Fluids* 71.3, pp. 269–293 (cit. on pp. 272, 276, 370).

Heimann, Felix (2013). "An Unfitted Higher-Order Discontinuous Galerkin Method for Incompressible Two-Phase Flow with Moving Contact Lines." PhD thesis. Heidelberg University (cit. on pp. 245, 250).

Helber, Bernd et al. (2014). "Material response characterization of a low-density carbon composite ablator in high-enthalpy plasma flows." In: *Journal of materials science* 49, pp. 4530–4543 (cit. on p. 27).

Helber, Bernd et al. (May 10, 2019). "Analysis of Meteoroid Ablation Based on Plasma Wind-tunnel Experiments, Surface Characterization, and Numerical Simulations." In: *The Astrophysical Journal* 876.2, p. 120 (cit. on pp. 27, 470).

Henneaux, David (2023). *PyManufSol*. https://github.com/henneauxd/PyManufSol. v1.0.0 (cit. on p. 515).

Henneaux, David, Philippe Chatelain, and Thierry E Magin (2020a). "Towards a High-Fidelity Multiphase Solver with Application to Space Debris Aerothermal Ablation Modeling." In: *11th Symposium of VKI PhD Research* (cit. on p. xviii).

– (2021a). "A high-order level-set method to follow the gas-liquid interface of melting materials." In: *12th Symposium of VKI PhD Research* (cit. on p. xviii).

– (2022a). "Enforcing Interface Coupling Conditions within a High-Order Sharp Interface Method: Application to Viscous Compressible Gas-Liquid Flows." In: *13th Symposium of VKI PhD Research* (cit. on p. xviii).

– (2023a). "Assessment of curvature filtering strategies for high-order sharp interface simulation of surface tension driven flows." In: *14th Symposium of VKI PhD Research* (cit. on p. xviii).

Henneaux, David et al. (2019a). "Towards a high-fidelity multiphase solver with application to space debris aerothermal ablation modeling." In: *AIAA Aviation 2019 Forum*, p. 2876 (cit. on pp. xvii, 151).

– (2019b). "Towards a high-fidelity multiphase solver with application to space debris aerothermal ablation modeling." Oral presentation at the AIAA Aviation 2019 Forum (Dallas) (cit. on pp. xvii, 151).

Henneaux, David et al. (2020b). "Extended discontinuous Galerkin method for solving gas-liquid compressible flows with phase transition." In: *AIAA AVIATION 2020 FORUM*, p. 2971 (cit. on pp. xvii, 265).

– (2020c). "Extended discontinuous Galerkin method for solving gas-liquid compressible flows with phase transition." Oral presentation at the AIAA Aviation 2020 Forum (virtual) (cit. on pp. xviii, 265).

Henneaux, David et al. (2021b). "A High-Order Level-Set Method Coupled with an Extended Discontinuous Galerkin Method for Simulating Moving Interface Problems." In: *AIAA AVIATION 2021 FORUM*, p. 2742 (cit. on pp. xvii, 197, 339).

– (2021c). "A High-Order Level-Set Method Coupled with an Extended Discontinuous Galerkin Method for Simulating Moving Interface Problems." Oral presentation at the AIAA AVIATION 2021 FORUM (virtual) (cit. on pp. xviii, 197, 339).

Henneaux, David et al. (2021d). "Sharp Interface Compressible Two-Phase Flow Solver Based On An Extended Discontinuous Galerkin Method." Oral presentation at the 14th World Congress on Computational Mechanics & ECCOMAS Congress (virtual) (cit. on pp. xviii, 265).

Henneaux, David et al. (2022b). "A High-Order Extended Discontinuous Galerkin Method for Coupled Multi-Material Sharp Interface Problems." Oral presentation at the 15th World Congress on Computational Mechanics & 8th Asian Pacific Congress on Computational Mechanics (virtual) (cit. on pp. xviii, 265, 313).
– (2022c). "Enforcing Interface Coupling Conditions within a Discontinuous Galerkin Sharp Interface Method: Application to Viscous Compressible Gas-Liquid Flows." Oral presentation at the 10th International Conference on Numerical Methods for Multi-Material Fluid Flow (Zurich) (cit. on pp. xviii, 265, 313).
– (2023b). "A General Methodology for Symbolically Generating Manufactured Solutions Satisfying Prescribed Conditions - Application to Two-phase Flows Equations." Submitted to Mathematics in Computer Science (cit. on pp. xvii, 377, 513).
– (2023c). "An All-Speed Extended Discontinuous Galerkin Sharp Interface Solver for Simulating Hydrodynamics of Melting Materials." Oral presentation at the 22nd IACM Computational Fluids Conference (Cannes) (cit. on pp. xviii, 45, 109).
Henneaux, David et al. (2023d). "Assessment of curvature filtering strategies for high-order sharp interface simulation of surface tension driven flows." Submitted to Flow, Turbulence and Combustion (cit. on pp. xvii, 241).
Henneaux, David et al. (2023e). "General Methodology for Symbolically Generating Manufactured Solutions Satisfying Prescribed Conditions Application to Two Phase Flows." Oral presentation at the 6th International Conference on Numerical and Symbolic Computation (virtual) (cit. on pp. xviii, 377, 513).
– (2023f). "High-order enforcement of jumps conditions between compressible viscous phases: An extended interior penalty discontinuous Galerkin method for sharp interface simulation." In: *Computer Methods in Applied Mechanics and Engineering* 415, p. 116215 (cit. on pp. xvii, 81, 265, 313).
Hennink, Aldo, Marco Tiberga, and Danny Lathouwers (2021). "A pressure-based solver for low-Mach number flow using a discontinuous Galerkin method." In: *Journal of Computational Physics* 425, p. 109877 (cit. on p. 180).
Henri, Félix, Mathieu Coquerelle, and Pierre Lubin (Jan. 2022a). "Geometrical Level Set Reinitialization Using Closest Point Method and Kink Detection for Thin Filaments, Topology Changes and Two-Phase Flows." In: *Journal of Computational Physics* 448, p. 110704 (cit. on pp. 212, 228).
– (2022b). "Geometrical level set reinitialization using closest point method and kink detection for thin filaments, topology changes and two-phase flows." In: *Journal of Computational Physics* 448, p. 110704 (cit. on p. 473).
Herman, Rudolf and WL Melnik (1962). "Aerodynamic and Heat Transfer Studies with Evaporative Film Cooling at Hypersonic Mach Numbers." In: *Rosemount Aeronautical Laboratories Research Report* 189 (cit. on p. 26).
Heuzé, Olivier (Oct. 2012). "General Form of the Mie–Grüneisen Equation of State." In: *Comptes Rendus Mécanique* 340.10, pp. 679–687 (cit. on pp. 74, 75).

Hieber, Simone E and Petros Koumoutsakos (2005). "A Lagrangian particle level set method." In: *Journal of Computational Physics* 210.1, pp. 342–367 (cit. on p. 203).

Hillewaert, Koen (2013). "Development of the discontinuous Galerkin method for high-resolution, large scale CFD and acoustics in industrial geometries." PhD thesis. Université catholique de Louvain (cit. on pp. 149, 171, 174, 180, 349, 350).

– (2023). *The discontinuous Galerkin method - A short intro to theory and practical implementation* (cit. on pp. 160, 170, 173).

Hoff, David (1998). "The zero-Mach limit of compressible flows." In: *Communications in mathematical physics* 192, pp. 543–554 (cit. on p. 182).

Holst, Kevin R, Ryan S Glasby, and Ryan B Bond (2020). "On the effect of temporal error in high-order simulations of unsteady flows." In: *Journal of Computational Physics* 402, p. 108989 (cit. on pp. 340–342).

Hoppe, Nils et al. (Mar. 2022). "ALPACA - a Level-Set Based Sharp-Interface Multiresolution Solver for Conservation Laws." In: *Computer Physics Communications* 272, p. 108246 (cit. on p. 317).

Hou, Songming and Xu-Dong Liu (2007). "Solutions of multi-dimensional hyperbolic systems of conservation laws by square entropy condition satisfying discontinuous Galerkin method." In: *Journal of Scientific Computing* 31, pp. 127–151 (cit. on p. 169).

Houim, Ryan W. and Kenneth K. Kuo (Feb. 2013). "A Ghost Fluid Method for Compressible Reacting Flows with Phase Change." In: *Journal of Computational Physics* 235, pp. 865–900 (cit. on pp. 87, 89, 103, 104, 316, 320).

Hu, X.Y. et al. (Dec. 2006). "A Conservative Interface Method for Compressible Flows." In: *Journal of Computational Physics* 219.2, pp. 553–578 (cit. on p. 267).

Huang, Tianci and Matthew J Zahr (2022). "A robust, high-order implicit shock tracking method for simulation of complex, high-speed flows." In: *Journal of Computational Physics* 454, p. 110981 (cit. on p. 201).

Huber, Marcia L et al. (2022). "The NIST REFPROP database for highly accurate properties of industrially important fluids." In: *Industrial & Engineering Chemistry Research* 61.42, pp. 15449–15472 (cit. on p. 72).

Hysing, S et al. (2009). "Quantitative benchmark computations of two-dimensional bubble dynamics." In: *International Journal for Numerical Methods in Fluids* 60.11, pp. 1259–1288 (cit. on pp. 416, 418, 419).

Hysing, S. (June 30, 2006). "A New Implicit Surface Tension Implementation for Interfacial Flows." In: *International Journal for Numerical Methods in Fluids* 51.6, pp. 659–672 (cit. on p. 244).

IADC, Inter-Agency Space Debris Coordination Committee steering group (2020). *IADC Space Debris Mitigation Guidelines*. IADC-02-01. Version Revision 2 (cit. on p. 6).

Irfan, Muhammad and Metin Muradoglu (May 2017). "A Front Tracking Method for Direct Numerical Simulation of Evaporation Process in a Multiphase System." In: *Journal of Computational Physics* 337, pp. 132–153 (cit. on p. 101).

Ishii, Mamoru and Takashi Hibiki (2011). *Thermo-Fluid Dynamics of Two-Phase Flow*. New York, NY: Springer New York (cit. on pp. 82, 90, 91, 97, 99–101, 116, 125).

Jaafar, Mohamad Ali et al. (2017). "A review of dendritic growth during solidification: Mathematical modeling and numerical simulations." In: *Renewable and Sustainable Energy Reviews* 74, pp. 1064–1079 (cit. on p. 114).

Jain, Suhas S., Ali Mani, and Parviz Moin (Oct. 2020). "A Conservative Diffuse-Interface Method for Compressible Two-Phase Flows." In: *Journal of Computational Physics* 418, p. 109606. arXiv: 1911.03619 [physics] (cit. on pp. 269, 404, 532).

Jiang, Guang Shan and Chi-Wang Shu (1994). "On a cell entropy inequality for discontinuous Galerkin methods." In: *Mathematics of Computation* 62.206, pp. 531–538 (cit. on p. 169).

Jibben, Z. and M. Herrmann (Nov. 2017). "An Arbitrary-Order Runge–Kutta Discontinuous Galerkin Approach to Reinitialization for Banded Conservative Level Sets." In: *Journal of Computational Physics* 349, pp. 453–473 (cit. on p. 212).

Johansson, August and Mats G. Larson (Apr. 2013). "A High Order Discontinuous Galerkin Nitsche Method for Elliptic Problems with Fictitious Boundary." In: *Numerische Mathematik* 123.4, pp. 607–628 (cit. on pp. 268, 270).

Jöns, Steven and Claus-Dieter Munz (Sept. 2020). "An Approximate Riemann Solver for Advection–Diffusion Based on the Generalized Riemann Problem." In: *Communications on Applied Mathematics and Computation* 2.3, pp. 515–539 (cit. on p. 320).

– (July 5, 2022). *Riemann Solvers for Phase Transition in a Compressible Sharp-Interface Method*. arXiv: 2207.02067 [physics]. URL: http://arxiv.org/abs/2207.02067 (visited on 05/13/2023). preprint (cit. on pp. 102, 323).

Jöns, Steven et al. (May 17, 2023). *A Viscous and Heat Conducting Ghost Fluid Method for Multi-Fluid Simulations*. arXiv: 2305.10278 [physics]. URL: http://arxiv.org/abs/2305.10278 (visited on 06/11/2023). preprint (cit. on pp. 87, 275, 320).

Jorge, Pedro (2019). *Development of oxidation models for space debris demise* (cit. on pp. xix, 27, 470).

Juric, Damir and Grétar Tryggvason (1998). "Computations of boiling flows." In: *International journal of multiphase flow* 24.3, pp. 387–410 (cit. on pp. 102, 104).

Kaiser, Klaus and Jochen Schütz (2017). "A high-order method for weakly compressible flows." In: *Communications in Computational Physics* 22.4, pp. 1150–1174 (cit. on p. 180).

Karni, Smadar (Sept. 1996). "Hybrid Multifluid Algorithms." In: *SIAM Journal on Scientific Computing* 17.5, pp. 1019–1039 (cit. on p. 274).

Kataoka, I. (Sept. 1986). "Local Instant Formulation of Two-Phase Flow." In: *International Journal of Multiphase Flow* 12.5, pp. 745–758 (cit. on p. 90).

Keffer, David (2016). *Evaluating Thermodynamic Properties in LAMMPS* (cit. on p. 498).

Kennedy, Christopher A and Mark H Carpenter (2016). *Diagonally implicit Runge-Kutta methods for ordinary differential equations. A review*. Tech. rep. NASA (cit. on pp. 341, 342).

Kercher, Andrew D, Andrew Corrigan, and David A Kessler (2021). "The moving discontinuous Galerkin finite element method with interface condition enforcement for compressible viscous flows." In: *International Journal for Numerical Methods in Fluids* 93.5, pp. 1490–1519 (cit. on p. 201).

Keshtiban, IJ, F Belblidia, and MF Webster (2004). "Compressible flow solvers for low Mach number flows-a review." In: *Int. J. Numer. Methods Fluids* 23, pp. 77–103 (cit. on p. 180).

Khot, SA, NK Sane, and BS Gawali (2011). "Experimental investigation of phase change phenomena of paraffin wax inside a capsule." In: *International Journal of Engineering Trends and Technology* 2.2, pp. 67–71 (cit. on p. 449).

Kim, H. and M.-S. Liou (Nov. 2019). "Adaptive Cartesian Cut-Cell Sharp Interface Method (aC3SIM) for Three-Dimensional Multi-Phase Flows." In: *Shock Waves* 29.8, pp. 1023–1041 (cit. on p. 317).

Kim, Hyeongjun, Hyunji Kim, and Chongam Kim (2018). "Computations of homogeneous multiphase real fluid flows at all speeds." In: *AIAA journal* 56.7, pp. 2623–2634 (cit. on p. 196).

Kitamura, Keiichi and Atsushi Hashimoto (Mar. 2016). "Reduced Dissipation AUSM-family Fluxes: HR-SLAU2 and HR-AUSM+-up for High Resolution Unsteady Flow Simulations." In: *Computers & Fluids* 126, pp. 41–57 (cit. on p. 195).

Kitamura, Keiichi and Eiji Shima (July 2013). "Towards Shock-Stable and Accurate Hypersonic Heating Computations: A New Pressure Flux for AUSM-family Schemes." In: *Journal of Computational Physics* 245, pp. 62–83 (cit. on p. 195).

Klein, B. et al. (July 20, 2016). "A High-Order Discontinuous Galerkin Solver for Low Mach Number Flows." In: *International Journal for Numerical Methods in Fluids* 81.8, pp. 489–520 (cit. on p. 180).

Knobbe, Edwin Michel (2010). "On the integration of the arbitrary Lagrangian-Eulerian concept and non-equilibrium thermodynamics." PhD thesis. TU Delft (cit. on pp. 48–55).

Knupp, Patrick and Kambiz Salari (2002). *Verification of computer codes in computational science and engineering*. Boca raton, Florida: CRC Press (cit. on p. 514).

Kollermeier, Julian (2012). *Comparison of Time Stepping Techniques for Compressible Gas Dynamics* (cit. on p. 347).

Koppenwallner, Georg et al. (2005). "SCARAB-a multi-disciplinary code for destruction analysis of space-craft during re-entry." In: *Fifth European Symposium on Aerothermodynamics for Space Vehicles*. Vol. 563, p. 281 (cit. on p. 14).

Kovács, Dániel G et al. (2023). "Space debris interaction across a two-dimensional oblique shock wave." In: *Experiments in Fluids* 64.8, p. 146 (cit. on p. 19).

Kronbichler, Martin and Per-Olof Persson, eds. (2021). *Efficient High-Order Discretizations for Computational Fluid Dynamics*. Vol. 602. CISM International

Centre for Mechanical Sciences. Cham: Springer International Publishing (cit. on p. 157).
Kubota, Toshi (Dec. 1960). "Ablation With Ice Model at M = 5.8." In: *ARS Journal* 30.12, pp. 1164–1169 (cit. on p. 448).
Kučera, Václav (2016). "On necessary and sufficient conditions for finite element convergence." In: *arXiv preprint arXiv:1601.02942* (cit. on p. 199).
Kühne, Thomas D (2014). "Second generation Car–Parrinello molecular dynamics." In: *Wiley Interdisciplinary Reviews: Computational Molecular Science* 4.4, pp. 391–406 (cit. on p. 488).
Kummer, Florian (2012). "." In: (cit. on p. 272).
– (Jan. 13, 2017). "Extended Discontinuous Galerkin Methods for Multiphase Flows: The Spatial Discretization." In: *International Journal for Numerical Methods in Engineering* 109.2, pp. 259–289 (cit. on pp. 87, 272, 273, 276, 288, 291, 329).
Kummer, Florian, Björn Müller, and Thomas Utz (Feb. 3, 2018). "Time Integration for Extended Discontinuous Galerkin Methods with Moving Domains." In: *International Journal for Numerical Methods in Engineering* 113.5, pp. 767–788 (cit. on pp. 354, 356, 357, 359, 424).
Kummer, Florian and Tim Warburton (Feb. 2016a). "Patch-Recovery Filters for Curvature in Discontinuous Galerkin-Based Level-Set Methods." In: *Communications in Computational Physics* 19.2, pp. 329–353 (cit. on pp. 251, 254, 256).
– (Feb. 2016b). "Patch-Recovery Filters for Curvature in Discontinuous Galerkin-based Level-Set Methods." In: *Communications in Computational Physics* 19.2, pp. 329–353. arXiv: 1501.02900 [physics] (cit. on pp. 243–245).
Kummer, Florian, Jens Weber, and Martin Smuda (2021). "BoSSS: A package for multigrid extended discontinuous Galerkin methods." In: *Computers & Mathematics with Applications* 81, pp. 237–257 (cit. on pp. 276, 277).
Kundu, A (2021). "Numerical simulation of a shock–helium bubble interaction." In: *Shock Waves* 31.1, pp. 19–30 (cit. on p. 409).
Kuzmin, Dmitri (2021). "Entropy stabilization and property-preserving limiters for □1 discontinuous Galerkin discretizations of scalar hyperbolic problems." In: *Journal of Numerical Mathematics* 29.4, pp. 307–322 (cit. on p. 159).
Kuzyk, William and Ching Jen Chen (1972). "Investigation of Simulated Melting Instability Waves Near Stagnation Region in Hypersonic Flow." In: *AIAA Journal* 10.11, pp. 1505–1508 (cit. on p. 440).
Labois, Mathieu (2008). "Modélisation des déséquilibres mécaniques pour les écoulements diphasiques: approches par relaxation et par modèle réduit." PhD thesis. Université de Provence-Aix-Marseille I (cit. on p. 122).
Lamb, Horace (1932). *Hydrodynamics*. Cambridge University Press (cit. on p. 414).
Lamonte, Aaricia (2020). *Evaporation modeling based on thermogravimetric analysis* (cit. on p. xix).
Landau, Lev Davidovich and Evgenii Mikhailovich Lifshitz (1987). *Fluid Mechanics (Second Edition)*. Vol. 2. Pergamon Press (cit. on p. 116).

Latige, Manuel (2013). "Simulation numérique de l'ablation liquide." PhD thesis. Université Sciences et Technologies-Bordeaux I (cit. on pp. 75, 77).

Le Métayer, O, J Massoni, and R Saurel (Mar. 2004). "Élaboration des lois d'état d'un liquide et de sa vapeur pour les modèles d'écoulements diphasiques." In: *International Journal of Thermal Sciences* 43.3, pp. 265–276 (cit. on pp. 75, 76).

Le Métayer, Olivier and Richard Saurel (Apr. 2016). "The Noble-Abel Stiffened-Gas Equation of State." In: *Physics of Fluids* 28.4, p. 046102 (cit. on pp. 71, 74).

Le Touze, Clement (2016). "Couplage entre modèles diphasiques à " phases séparées " et à " phase dispersée " pour la simulation de l'atomisation primaire en combustion cryotechnique." PhD thesis. Université Nice Sofia Antipolis (cit. on pp. 117, 120).

Lebon, G., D. Jou, and J. Casas-Vázquez (2008). *Understanding Non-Equilibrium Thermodynamics: Foundations, Applications, Frontiers*. Berlin: Springer. 325 pp. (cit. on p. 64).

Lee, Moon Soo, Amir Riaz, and Vikrant Aute (Sept. 2017). "Direct Numerical Simulation of Incompressible Multiphase Flow with Phase Change." In: *Journal of Computational Physics* 344, pp. 381–418 (cit. on p. 87).

Lees, Lester (1959). "Similarity parameters for surface melting of a blunt nosed body in a high velocity gas stream." In: *ARS Journal* 29.5, pp. 345–354 (cit. on p. 441).

Lehrenfeld, Christoph and Maxim Olshanskii (2019). "An Eulerian finite element method for PDEs in time-dependent domains." In: *ESAIM: Mathematical Modelling and Numerical Analysis* 53.2, pp. 585–614 (cit. on p. 354).

Levaux, N. et al. (2023). "Interior Penalty Method for a Cartesian Discontinuous Galerkin Solver with Immersed Boundaries." IACM Computational Fluids Conference (cit. on p. 329).

Levý, Tomáš and Georg May (Jan. 23, 2023). "Conservative Solution Transfer Between Anisotropic Meshes for Adaptive Time-Accurate Hybridized Discontinuous Galerkin Methods." In: *AIAA SCITECH 2023 Forum*. AIAA SCITECH 2023 Forum. National Harbor, MD & Online: American Institute of Aeronautics and Astronautics (cit. on p. 310).

Li, Chunming et al. (2010). "Distance regularized level set evolution and its application to image segmentation." In: *IEEE transactions on image processing* 19.12, pp. 3243–3254 (cit. on p. 212).

LibreTexts, Chemistry (n.d.[a]). *Chemical Bonds, Lewis Symbols, and the Octet Rule*. URL: https://chem.libretexts.org/Bookshelves/General_Chemistry/Map%5C%3A_Chemistry_-_The_Central_Science_(Brown_et_al.)/08._Basic_Concepts_of_Chemical_Bonding/8.1%5C%3A_Chemical_Bonds%5C%2C_Lewis_Symbols%5C%2C_and_the_Octet_Rule (cit. on p. 491).

– (n.d.[b]). *Intermolecular Forces: The Forces that Hold Condensed Phases Together*. URL: https://chem.libretexts.org/Bookshelves/General_Chemistry/Map%5C%3A_A_Molecular_Approach_(Tro)/11%5C%3A_Liquids%5C%2C_Solids%5C%2C_and_Intermolecular_Forces/11.03%5C%3A_Intermolecular_

Forces%5C%3A_The_Forces_that_Hold_Condensed_Phases_Together (cit. on p. 492).
– (n.d.[c]). *Metallic Bonding*. URL: https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Chemical_Bonding/Fundamentals_of_Chemical_Bonding/Metallic_Bonding (cit. on pp. 491, 492).
Lin, Jian-Yu et al. (Jan. 2017). "Simulation of Compressible Two-Phase Flows with Topology Change of Fluid–Fluid Interface by a Robust Cut-Cell Method." In: *Journal of Computational Physics* 328, pp. 140–159 (cit. on pp. 268, 317).
Liou, Meng-Sing (Dec. 1996). "A Sequel to AUSM: AUSM+." In: *Journal of Computational Physics* 129.2, pp. 364–382 (cit. on p. 194).
– (May 2006). "A Sequel to AUSM, Part II: AUSM+-up for All Speeds." In: *Journal of Computational Physics* 214.1, pp. 137–170 (cit. on p. 194).
Liou, Meng-Sing and Christopher J Steffen Jr (1993). "A new flux splitting scheme." In: *Journal of Computational physics* 107.1, pp. 23–39 (cit. on pp. 168, 194).
Lips, T et al. (2004). "Spacecraft destruction during re-entry–latest results and development of the SCARAB software system." In: *Advances in Space Research* 34.5, pp. 1055–1060 (cit. on p. 13).
Lips, Tobias and Bent Fritsche (July 2005). "A Comparison of Commonly Used Re-Entry Analysis Tools." In: *Acta Astronautica* 57.2-8, pp. 312–323 (cit. on p. 11).
Liu, I-Shih et al. (2002). *Continuum mechanics*. Vol. 5. Springer (cit. on p. 60).
Liu, I-Shih (2019). *Introduction to continuum mechanics*. Instituto de matematica, Universidade federal de Rio de Janeiro (cit. on pp. 61, 62).
Liu, T.G., B.C. Khoo, and K.S. Yeo (Sept. 2003). "Ghost Fluid Method for Strong Shock Impacting on Material Interface." In: *Journal of Computational Physics* 190.2, pp. 651–681 (cit. on p. 316).
Liu, Tiegang, Chengliang Feng, and Liang Xu (Dec. 2019). "Modified Ghost Fluid Method with Acceleration Correction (MGFM/AC)." In: *Journal of Scientific Computing* 81.3, pp. 1906–1944 (cit. on p. 316).
Liu, Yun-Long, Chi-Wang Shu, and A-Man Zhang (Feb. 2021). "Weighted Ghost Fluid Discontinuous Galerkin Method for Two-Medium Problems." In: *Journal of Computational Physics* 426, p. 109956 (cit. on p. 316).
Long, Tian, Jinsheng Cai, and Shucheng Pan (Oct. 15, 2021a). *A Fully Conservative Sharp-Interface Method for Compressible Mulitphase Flows with Phase Change*. arXiv: 2110.07995 [physics]. URL: http://arxiv.org/abs/2110.07995 (visited on 05/13/2023). preprint (cit. on pp. 324, 325).
– (Mar. 2021b). "An Accelerated Conservative Sharp-Interface Method for Multiphase Flows Simulations." In: *Journal of Computational Physics* 429, p. 110021 (cit. on p. 317).
Lu, Haitian et al. (2016). "Runge–Kutta discontinuous Galerkin method with front tracking method for solving the compressible two-medium flow." In: *Computers & Fluids* 126, pp. 1–11 (cit. on p. 200).

Luo, J., X.Y. Hu, and N.A. Adams (Mar. 2015). "A Conservative Sharp Interface Method for Incompressible Multiphase Flows." In: *Journal of Computational Physics* 284, pp. 547–565 (cit. on p. 317).

Lv, Yong J and Min Chen (2011). "Thermophysical properties of undercooled alloys: an overview of the molecular simulation approaches." In: *International journal of molecular sciences* 12.1, pp. 278–316 (cit. on p. 503).

Magiera, Jim and Christian Rohde (Nov. 2022). "A Molecular-Continuum Multiscale Model for Inviscid Liquid-Vapor Flow with Sharp Interfaces." In: *Journal of Computational Physics* 469, p. 111551. arXiv: 2204.02233 [physics] (cit. on p. 100).

Magin, Thierry (2004). "A model for inductive plasma wind tunnels." PhD thesis. Université libre de Bruxelles (cit. on p. 73).

Maginn, Edward J. (1997). *Chemical Engineering: Molecular Theory and Modeling* (cit. on pp. 496, 501).

Main, Alex and Guglielmo Scovazzi (2018). "The shifted boundary method for embedded domain computations. Part II: Linear advection–diffusion and incompressible Navier–Stokes equations." In: *Journal of Computational Physics* 372, pp. 996–1026 (cit. on pp. 268, 270).

Majidi, Sahand and Asghar Afshari (2016). "A ghost fluid method for sharp interface simulations of compressible multiphase flows." In: *Journal of Mechanical Science and Technology* 30.4, pp. 1581–1593 (cit. on pp. 316, 320).

Malaya, Nicholas et al. (2013). "MASA: a library for verification using manufactured and analytical solutions." In: *Engineering with Computers* 29, pp. 487–496 (cit. on p. 515).

Malladi, R., J.A. Sethian, and B.C. Vemuri (1995). "Shape Modeling with Front Propagation: A Level Set Approach." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.2, pp. 158–175 (cit. on p. 220).

Maltsev, V., M. Skote, and P. Tsoutsanis (Feb. 2022). "High-Order Methods for Diffuse-Interface Models in Compressible Multi-Medium Flows: A Review." In: *Physics of Fluids* 34.2, p. 021301 (cit. on pp. 122, 269).

Manikantan, Harishankar and Todd M Squires (2020). "Surfactant dynamics: hidden variables controlling fluid flows." In: *Journal of fluid mechanics* 892, P1 (cit. on p. 97).

Marchandise, Emilie et al. (2006). "Simulation of three-dimensional two-phase flows: coupling of a stabilized finite element method with a discontinuous level set approach." PhD thesis. PhD thesis, Université catholique de Louvain (cit. on pp. 202, 473).

Marchandise, Emilie and Jean-François Remacle (2006). "A stabilized finite element method using a discontinuous level set approach for solving two phase incompressible flows." In: *Journal of computational physics* 219.2, pp. 780–800 (cit. on p. 207).

Marchandise, Emilie et al. (July 2007). "A Stabilized Finite Element Method Using a Discontinuous Level Set Approach for the Computation of Bubble Dynamics." In: *Journal of Computational Physics* 225.1, pp. 949–974 (cit. on pp. 212, 244).

Marmottant, Philippe and Emmanuel Villermaux (2004). "On spray formation." In: *Journal of fluid mechanics* 498, pp. 73–111 (cit. on pp. 26, 28).

Marschall, Holger et al. (Oct. 2014). "Validation of Interface Capturing and Tracking Techniques with Different Surface Tension Treatments against a Taylor Bubble Benchmark Problem." In: *Computers & Fluids* 102, pp. 336–352 (cit. on p. 246).

Martinez, Mario J. and David K. Gartling (May 2004). "A Finite Element Method for Low-Speed Compressible Flows." In: *Computer Methods in Applied Mechanics and Engineering* 193.21-22, pp. 1959–1979 (cit. on pp. 68, 180).

Marx, Dominik and Jurg Hutter (2000). "Ab initio molecular dynamics: Theory and implementation." In: *Modern methods and algorithms of quantum chemistry* 1.301-449, p. 141 (cit. on p. 488).

May, S. and F. Thein (Jan. 2023). "Explicit Implicit Domain Splitting for Two Phase Flows with Phase Transition." In: *Physics of Fluids* 35.1, p. 016108 (cit. on p. 88).

Mehta, RC (2017). "Effect of geometrical parameters of reentry capsule over flowfield at high speed flow." In: *Advances in aircraft and spacecraft science* 4.4, p. 487 (cit. on p. 19).

Menikoff, Ralph and Bradley J. Plohr (Jan. 1, 1989). "The Riemann Problem for Fluid Flow of Real Materials." In: *Reviews of Modern Physics* 61.1, pp. 75–130 (cit. on pp. 65, 66, 186, 191).

Merkle, Charles, Sankaran Venkateswaran, and Philip Buelow (1992). "The relationship between pressure-based and density-based algorithms." In: *30th Aerospace Sciences Meeting and Exhibit*, p. 425 (cit. on p. 180).

Merrifield, J. et al. (2015). "Aerothermal Heating Methodology in the Spacecraft Aerothermal Model (SAM)." In: *Space Safety Is No Accident*. Ed. by Tommaso Sgobba and Isabelle Rongier. Cham: Springer International Publishing, pp. 463–468 (cit. on p. 15).

Meyer, Lisa (2022). "Development of a Preconditioning Scheme for Real Gases using Asymptotic Expansions." PhD thesis. Universität Kassel (cit. on p. 190).

Miccoli, Claudio (2021). *Development of a low-Mach number compressible solver to simulate liquid metal flows for space debris ablation* (cit. on pp. xix, 151).

Mie, Gustav (1903). "Zur kinetischen Theorie der einatomigen Körper." In: *Annalen der Physik* 316.8, pp. 657–697 (cit. on p. 74).

Miettinen, Ari and Timo Siikonen (2015). "Application of pressure-and density-based methods for different flow speeds." In: *International Journal for Numerical Methods in Fluids* 79.5, pp. 243–267 (cit. on p. 180).

Mihalef, Viorel, Dimitris Metaxas, and Mark Sussman (2007). "Textured liquids based on the marker level set." In: *Computer Graphics Forum*. Vol. 26. 3. Wiley Online Library, pp. 457–466 (cit. on p. 203).

Mirjalili, Shahab, Suhas S Jain, and Micheal Dodd (2017). "Interface-capturing methods for two-phase flows: An overview and recent developments." In: *Center for Turbulence Research Annual Research Briefs* 2017.117-135, p. 13 (cit. on p. 200).

Miró Miró, Fernando (2020). "Numerical investigation of hypersonic boundary-layer stability and transition in the presence of ablation phenomena." PhD thesis. Université libre de Bruxelles (cit. on p. 25).

Moes, Nicolas et al. (Mar. 2023). "The eXtreme Mesh Deformation Approach (X-MESH) for the Stefan Phase-Change Model." In: *Journal of Computational Physics* 477, p. 111878. arXiv: 2111.04179 [cs] (cit. on p. 199).

Moran, Michael J et al. (2010). *Fundamentals of engineering thermodynamics*. John Wiley & Sons (cit. on pp. 59, 75).

Morel, Christophe (2015). *Mathematical Modeling of Disperse Two-Phase Flows*. Vol. 114. Fluid Mechanics and Its Applications. Cham: Springer International Publishing (cit. on pp. 116, 117, 120).

Morozova, Viktoriya, James G Coder, and Kevin Holst (2023). "High-Order, Implicit Time Integration of Discrete, Chaotic Dynamical Systems." In: *arXiv preprint arXiv:2401.00576* (cit. on p. 343).

Mossier, Pascal, Daniel Appel, and Claus-Dieter Munz (n.d.). "An Efficient Hp-Adaptive Strategy for a Level-Set Ghost Fluid Method." In: () (cit. on p. 204).

Mousavi, Roozbeh (2014). "Level set method for simulating the dynamics of the fluid-fluid interfaces: Application of a Discontinuous Galerkin Method." PhD thesis. Technische Universität Darmstadt (cit. on pp. 225, 226).

Muller, B et al. (2016). "A High-Order Discontinuous Galerkin Method for Compressible Flows with Immersed Boundaries." In: (cit. on pp. 251, 272, 277, 288, 296, 298, 300).

Müller, B., F. Kummer, and M. Oberlack (Nov. 23, 2013). "Highly Accurate Surface and Volume Integration on Implicit Domains by Means of Moment-Fitting." In: *International Journal for Numerical Methods in Engineering* 96.8, pp. 512–528 (cit. on pp. 277, 286, 288).

Müller, Ingo (1985). *Thermodynamics*. London: Pitman (cit. on pp. 90, 98).

Munafò, Alessandro (2014). "Multi-Scale Models and Computational Methods for Aerothermodynamics." PhD thesis. Ecole Central Paris (cit. on p. 442).

Munz, C-D (2003). "The extension of incompressible flow solvers to the weakly compressible regime." In: *Computers & Fluids* 32.2, pp. 173–196 (cit. on p. 180).

Nayigizente, Davy (2021). "Unsteady simulations of liquid/gas interfaces in real gas flows using the Second Gradient theory." PhD thesis. Université Paris-Saclay (cit. on pp. 82, 83).

Ngo, Long Cu, Hyoung Gwon Choi, and Muhammad Sufyan (2019). "Treatments of the singularity in the level set method for unstructured meshes." In: *Finite Elements in Analysis and Design* 162, pp. 34–50 (cit. on p. 204).

Ngo, Long Cu, Quang-Ngoc Dinh, and Hyoung Gwon Choi (2022). "High-order level set reinitialization for multiphase flow simulations based on unstructured grids." In: *Computers & Mathematics with Applications* 120, pp. 60–77 (cit. on p. 212).

Nguyen, Vinh-Tan et al. (Jan. 2010). "A Discontinuous Galerkin Front Tracking Method for Two-Phase Flows with Surface Tension." In: *Computers & Fluids* 39.1, pp. 1–14 (cit. on pp. 267, 317).

Ni, J. and C. Beckermann (June 1991). "A Volume-Averaged Two-Phase Model for Transport Phenomena during Solidification." In: *Metallurgical Transactions B* 22.3, pp. 349–361 (cit. on pp. 124, 125).

Ni, Jun and Frank P Incropera (1995a). "Extension of the continuum model for transport phenomena occurring during metal alloy solidification-II. Microscopic considerations." In: *International Journal of Heat and Mass Transfer* 38.7, pp. 1285–1296 (cit. on p. 135).

– (1995b). "Extension of the continuum model for transport phenomena occurring during metal alloy solidification—I. The conservation equations." In: *International Journal of Heat and Mass Transfer* 38.7, pp. 1271–1284 (cit. on pp. 114, 127, 134–137).

Nigro, A. et al. (May 10, 2013). "A High-Order Accurate Discontinuous Galerkin Finite Element Method for Laminar Low Mach Number Flows." In: *International Journal for Numerical Methods in Fluids* 72.1, pp. 43–68 (cit. on p. 182).

Nigro, Alessandra et al. (2014). "A comparison between high-order temporal integration methods applied to the Discontinuous Galerkin discretized Euler equations." In: *Energy Procedia* 45, pp. 518–527 (cit. on p. 343).

Nikrityuk, Petr A. (Aug. 24, 2011). *Computational Thermo-Fluid Dynamics: In Materials Science and Engineering*. 1st ed. Wiley (cit. on pp. 111, 132, 134).

Nishikawa, Hiroaki (2021). *On Adding Pseudo-Time Derivative* (cit. on p. 347).

Nitsche, Joachim (1971). "Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind." In: *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*. Vol. 36. 1. Springer, pp. 9–15 (cit. on p. 270).

Nourgaliev, R et al. (2016). "Fully-implicit orthogonal reconstructed Discontinuous Galerkin method for fluid dynamics with phase change." In: *Journal of Computational Physics* 305, pp. 964–996 (cit. on pp. 142, 350, 351).

Nourgaliev, R. et al. (Feb. 18, 2015). *Fully-Implicit Orthogonal Reconstructed Discontinuous Petrov-Galerkin Method for Multiphysics Problems*. LLNL-TR–664250, 1178386, LLNL-TR–664250, 1178386 (cit. on p. 182).

Nourgaliev, R. et al. (July 2019). "High-Order Fully Implicit Solver for All-Speed Fluid Dynamics: AUSM Ride from Nearly Incompressible Variable-Density Flows to Shock Dynamics." In: *Shock Waves* 29.5, pp. 651–689 (cit. on pp. 181, 182, 185).

Nourgaliev, R.R., M.-S. Liou, and T.G. Theofanous (Apr. 2008). "Numerical Prediction of Interfacial Instabilities: Sharp Interface Method (SIM)." In: *Journal of Computational Physics* 227.8, pp. 3940–3970 (cit. on pp. 86, 243, 317).

Nourgaliev, Robert, Nam Dinh, and Theo Theofanous (June 6, 2005). "Sharp Treatment of Surface Tension and Viscous Stresses in Multifluid Dynamics." In: *17th AIAA Computational Fluid Dynamics Conference*. 17th AIAA Computational Fluid Dynamics Conference. Toronto, Ontario, Canada: American Institute of Aeronautics and Astronautics (cit. on p. 86).

Nourgaliev, Robert, Patrick T. Greene, and Samuel P. Schofield (July 30, 2017). "Physics-Based Reconstruction of Interfacial Jump Conditions in All-Speed Multi-Fluid Dynamics." In: *ASME 2017 Fluids Engineering Division Summer*

Meeting. ASME 2017 Fluids Engineering Division Summer Meeting. Waikoloa, Hawaii, USA: American Society of Mechanical Engineers, V01BT11A017 (cit. on pp. 275, 317, 393).

Nourgaliev, Robert, Samet Kadioglu, and Vincent Mousseau (Jan. 2010). "Marker Redistancing/Level Set Method for High-Fidelity Implicit Interface Tracking." In: *SIAM Journal on Scientific Computing* 32.1, pp. 320–348 (cit. on pp. 203, 473).

Nourgaliev, Robert et al. (Jan. 7, 2008). "Direct Numerical Simulation of Interfacial Flows: Implicit Sharp-Interface Method (I-SIM)." In: *46th AIAA Aerospace Sciences Meeting and Exhibit*. 46th AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada: American Institute of Aeronautics and Astronautics (cit. on p. 371).

Oberkampf, William L and Christopher J Roy (2010). *Verification and validation in scientific computing*. Cambridge,UK: Cambridge University Press (cit. on pp. 16, 514).

OECD, Organization for Economic Cooperation and Development (Apr. 8, 2020). *Space Sustainability: The Economics of Space Debris in Perspective*. OECD Science, Technology and Industry Policy Papers 87 (cit. on p. 6).

Oldenburg, Curtis M. and Frank J. Spera (Mar. 1992). "Hybrid Model for Solidification and Convection." In: *Numerical Heat Transfer, Part B: Fundamentals* 21.2, pp. 217–229 (cit. on p. 138).

Olshanskii, Maxim A., Arnold Reusken, and Paul Schwering (Feb. 1, 2023). *An Eulerian Finite Element Method for Tangential Navier-Stokes Equations on Evolving Surfaces*. arXiv: 2302.00779 [math-ph]. URL: http://arxiv.org/abs/2302.00779 (visited on 06/11/2023). preprint (cit. on p. 354).

Olsson, Elin and Gunilla Kreiss (2005). "A conservative level set method for two phase flow." In: *Journal of computational physics* 210.1, pp. 225–246 (cit. on p. 202).

Osher, Stanley and James A Sethian (1988). "Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations." In: *Journal of computational physics* 79.1, pp. 12–49 (cit. on p. 204).

Osher, Stanley and Fred Solomon (1982). "Upwind difference schemes for hyperbolic systems of conservation laws." In: *Mathematics of computation* 38.158, pp. 339–374 (cit. on p. 168).

Ostrach, Simon, Arthur W Goldstein, and Jesse Hamman (1962). *Analysis of melting boundary layers on decelerating bodies*. Tech. rep. NASA (cit. on p. 441).

Ovsyannikov, A, V Sabel'nikov, and M Gorokhovski (2012). "A new level set equation and its numerical assessments." In: *Proceedings of the Summer Program*. Center for Turbulence Research, p. 315 (cit. on p. 221).

Owkes, Mark and Olivier Desjardins (Sept. 2013). "A Discontinuous Galerkin Conservative Level Set Scheme for Interface Capturing in Multiphase Flows." In: *Journal of Computational Physics* 249, pp. 275–302 (cit. on p. 202).

Pan, Yu (2022). "Implicit time integration for high-order compressible flow solvers." PhD thesis. Imperial College London (cit. on p. 344).

Pan, Yu et al. (June 2022). "Development of a Balanced Adaptive Time-Stepping Strategy Based on an Implicit JFNK-DG Compressible Flow Solver." In: *Communications on Applied Mathematics and Computation* 4.2, pp. 728–757 (cit. on pp. 344, 472).

Papanastasiou, Tasos, Georgios Georgiou, and Andreas N Alexandrou (2000). *Viscous fluid flow*. CRC press (cit. on pp. 51, 63).

Park, Jin Seok et al. (2014). "Comparative study of shock-capturing methods for high-order CPR: MLP and artificial viscosity." In: *Proceedings of the 8th International Conference on CFD, ICCFD8-2014-0067* (cit. on p. 305).

Paula, Thomas, Stefan Adami, and Nikolaus A. Adams (Apr. 15, 2019). "Analysis of the Early Stages of Liquid-Water-Drop Explosion by Numerical Simulation." In: *Physical Review Fluids* 4.4, p. 044003 (cit. on p. 103).

Pawlowski, Roger P. et al. (Jan. 2006). "Globalization Techniques for Newton–Krylov Methods and Applications to the Fully Coupled Solution of the Navier–Stokes Equations." In: *SIAM Review* 48.4, pp. 700–721 (cit. on p. 344).

Pazner, Will and Terry Haut (2021). "A short note on the accuracy of the discontinuous Galerkin method with reentrant faces." In: *Journal of Computational Physics* 443, p. 110448 (cit. on pp. 173, 359).

Peluchon, S., G. Gallice, and L. Mieussens (Jan. 2022). "Development of Numerical Methods to Simulate the Melting of a Thermal Protection System." In: *Journal of Computational Physics* 448, p. 110753 (cit. on pp. 33, 386).

Peluchon, Simon (2017). "Approximation numérique et modélisation de l'ablation liquide." PhD thesis. Université de Bordeaux (cit. on p. 77).

Peng, Danping et al. (Nov. 1999). "A PDE-Based Fast Local Level Set Method." In: *Journal of Computational Physics* 155.2, pp. 410–438 (cit. on pp. 219, 221).

Perin, Andrea (2023). *Wind Tunnel Testing of Space Debris Demise (operation and application)* (cit. on pp. xix, 437).

Persson, Per-Olof (2011). "High-order LES simulations using implicit-explicit Runge-Kutta schemes." In: *49th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, p. 684 (cit. on p. 342).

Persson, Per-Olof and Jaime Peraire (2006). "Sub-cell shock capturing for discontinuous Galerkin methods." In: *44th AIAA Aerospace Sciences Meeting and Exhibit*, p. 112 (cit. on pp. 303, 304).

Pochet, François et al. (Oct. 2013). "A 3D Strongly Coupled Implicit Discontinuous Galerkin Level Set-Based Method for Modeling Two-Phase Flows." In: *Computers & Fluids* 87, pp. 144–155 (cit. on pp. 201, 371).

Poling, Bruce E., J. M. Prausnitz, and John P. O'Connell (2001). *The Properties of Gases and Liquids*. 5th ed. New York: McGraw-Hill. 1 p. (cit. on pp. 72–74).

Popinet, Stéphane (Jan. 5, 2018). "Numerical Models of Surface Tension." In: *Annual Review of Fluid Mechanics* 50.1, pp. 49–75 (cit. on p. 242).

Potghan, Nilesh and Chandrashekhar S. Jog (Dec. 2022). "An ALE-based Finite Element Strategy for Modeling Compressible Two-phase Flows." In: *International Journal for Numerical Methods in Fluids* 94.12, pp. 2040–2086 (cit. on pp. 267, 317).

Potter, Daniel (2011). "Modelling of radiating shock layers for atmospheric entry at Earth and Mars." PhD thesis. The University of Queensland (cit. on pp. 20, 23).

Prenter, Frits de, Christoph Lehrenfeld, and André Massing (2018). "A note on the stability parameter in Nitsche's method for unfitted boundary value problems." In: *Computers & Mathematics with Applications* 75.12, pp. 4322–4336 (cit. on p. 272).

Prenter, Frits de et al. (2017). "Condition number analysis and preconditioning of the finite cell method." In: *Computer Methods in Applied Mechanics and Engineering* 316, pp. 297–327 (cit. on p. 271).

Prescott, P.J, F.P. Incropera, and W.D. Bennon (1991). "Modeling of dendritic solidification systems: reassessment of the continuum momentum equation." In: *International Journal of Heat and Mass Transfer* 34.9, pp. 2351–2359 (cit. on pp. 129, 135–137).

Prescott, Patrick J and Frank P Incropera (1996). "Convection heat and mass transfer in alloy solidification." In: *Advances in heat transfer* 28, pp. 231–338 (cit. on p. 114).

Prévereaud, Y. et al. (May 2016). "Numerical and Experimental Study of the Thermal Degradation Process during the Atmospheric Re-Entry of a TiAl 6 V 4 Tank." In: *Acta Astronautica* 122, pp. 258–286 (cit. on p. 27).

Prosperetti, A. (Mar. 29, 2007). "Averaged Equations for Multiphase Flow." In: *Computational Methods for Multiphase Flow*. Ed. by Andrea Prosperetti and Grétar Tryggvason. 1st ed. Cambridge University Press, pp. 237–281 (cit. on p. 116).

Prosperetti, Andrea (1981). "Motion of two superposed viscous fluids." In: *The Physics of Fluids* 24.7, pp. 1217–1223 (cit. on p. 420).

Psihoyios, G -Y and JR Cash (1998). "A stability result for general linear methods with characteristic function having real poles only." In: *BIT Numerical Mathematics* 38, pp. 612–617 (cit. on p. 343).

Qu, Feng et al. (2014). "A new flux splitting scheme for the Euler equations." In: *Computers & Fluids* 102, pp. 203–214 (cit. on p. 169).

Qu, Feng et al. (2015). "A new Roe-type scheme for all speeds." In: *Computers & Fluids* 121, pp. 11–25 (cit. on p. 196).

Qu, Feng et al. (2017). "An improvement on the AUSMPWM scheme for hypersonic heating predictions." In: *International Journal of Heat and Mass Transfer* 108, pp. 2492–2501 (cit. on p. 196).

Qu, Feng et al. (2021). "A review of riemann solvers for hypersonic flows." In: *Archives of Computational Methods in Engineering*, pp. 1–30 (cit. on p. 196).

Quartapelle, L. et al. (Sept. 2003). "Solution of the Riemann Problem of Classical Gasdynamics." In: *Journal of Computational Physics* 190.1, pp. 118–140 (cit. on p. 529).

Quiriny, Antoine et al. (Feb. 8, 2023). *X-Mesh: A New Approach for the Simulation of Two-Phase Flow with Sharp Interface*. arXiv: 2302.03983 [cs]. URL: http://arxiv.org/abs/2302.03983 (visited on 05/13/2023). preprint (cit. on p. 201).

Radulescu, Matei Ioan (Apr. 18, 2020). *Compressible Flow in a Noble-Abel Stiffened-Gas Fluid*. arXiv: 2004.08750 [physics]. URL: http://arxiv.org/abs/2004.08750 (visited on 05/13/2023). preprint (cit. on p. 75).

Rasquin, M, A Bauer, and Koen Hillewaert (2019). "Scientific post hoc and in situ visualisation of high-order polynomial solutions from massively parallel simulations." In: *International Journal of Computational Fluid Dynamics* (cit. on p. 378).

Rayleigh, Lord et al. (1879). "On the capillary phenomena of jets." In: *Proc. R. Soc. London* 29.196-199, pp. 71–97 (cit. on p. 414).

Register, PJ et al. (2020). "Interactions between asteroid fragments during atmospheric entry." In: *Icarus* 337, p. 113468 (cit. on p. 19).

Remacle, Jean-François et al. (2007). "Efficient visualization of high-order finite elements." In: *International Journal for Numerical Methods in Engineering* 69.4, pp. 750–771 (cit. on pp. 213, 378).

Renda, S. M. et al. (Feb. 10, 2015). "A High-Order Discontinuous Galerkin Method for All-Speed Flows." In: *International Journal for Numerical Methods in Fluids* 77.4, pp. 224–247 (cit. on p. 181).

Reusken, A., X. Xu, and L. Zhang (Oct. 12, 2015). *Finite Element Methods for a Class of Continuum Models for Immiscible Flows with Moving Contact Lines*. arXiv: 1510.03160 [physics]. URL: http://arxiv.org/abs/1510.03160 (visited on 05/13/2023). preprint (cit. on p. 104).

Rieckmann, Matthias et al. (2023). "The Extended Discontinuous Galerkin Method for Two-Phase Flows with Evaporation." In: *Journal of Computational Physics, In Press* (cit. on p. 372).

Rieper, Felix (June 2011). "A Low-Mach Number Fix for Roe's Approximate Riemann Solver." In: *Journal of Computational Physics* 230.13, pp. 5263–5287 (cit. on p. 189).

Rinaldi, Enrico, Rene Pecnik, and Piero Colonna (Aug. 2014). "Exact Jacobians for Implicit Navier–Stokes Simulations of Equilibrium Real Gas Flows." In: *Journal of Computational Physics* 270, pp. 459–477 (cit. on p. 380).

Rivière, Béatrice (Jan. 2008). *Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation*. Society for Industrial and Applied Mathematics (cit. on pp. 391, 396).

Roache, Patrick J (1998). *Verification and validation in computational science and engineering*. Vol. 895. Albuquerque, NM: Hermosa (cit. on p. 514).

– (2019). "The Method of Manufactured Solutions for Code Verification." In: *Computer Simulation Validation: Fundamental Concepts, Methodological Frameworks, and Philosophical Perspectives*. Ed. by Claus Beisbart and Nicole J. Saam. Cham: Springer International Publishing, pp. 295–318 (cit. on p. 514).

Roe, Philip L (1981). "Approximate Riemann solvers, parameter vectors, and difference schemes." In: *Journal of computational physics* 43.2, pp. 357–372 (cit. on pp. 168, 195).

Rohde, Christian (2018). "Fully resolved compressible two-phase flow: modelling, analytical and numerical issues." In: *New trends and results in mathematical description of fluid flows*, pp. 115–181 (cit. on p. 316).

Rohde, Christian and Christoph Zeiler (June 2018). "On Riemann Solvers and Kinetic Relations for Isothermal Two-Phase Flows with Surface Tension." In: *Zeitschrift für angewandte Mathematik und Physik* 69.3, p. 76 (cit. on p. 87).

Rosenholm, Jarl B. (Aug. 2021). "A Rational Approach to Basic Equilibrium Thermodynamics." In: *Advances in Colloid and Interface Science* 294, p. 102433 (cit. on p. 65).

Rouy, S, P Helluy, and S Kokh (2004). "Test-case number 13: Shock tubes(PA)." In: *Multiphase Science and Technology* 16.1-3 (cit. on p. 408).

Saad, Yousef (2003). *Iterative methods for sparse linear systems*. SIAM (cit. on pp. 343, 344, 349).

Sabelnikov, Vladimir, Andrey Yu Ovsyannikov, and Mikhael Gorokhovski (2014). "Modified level set equation and its numerical assessment." In: *Journal of computational physics* 278, pp. 1–30 (cit. on p. 221).

Sachdev, Jai, Ashvin Hosangadi, and V Sankaran (2012). "Improved flux formulations for unsteady low Mach number flows." In: *42nd AIAA Fluid Dynamics Conference and Exhibit*, p. 3067 (cit. on p. 193).

Salac, David and Wei Lu (2008). "A local semi-implicit level-set method for interface motion." In: *Journal of Scientific Computing* 35, pp. 330–349 (cit. on p. 204).

Salari, Kambiz and Patrick Knupp (2000). *Code verification by the method of manufactured solutions*. Tech. rep. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States) (cit. on p. 519).

Saurel, Richard and Rémi Abgrall (Apr. 1999). "A Multiphase Godunov Method for Compressible Multifluid and Multiphase Flows." In: *Journal of Computational Physics* 150.2, pp. 425–467 (cit. on p. 318).

Saurel, Richard, Michel Larini, and Jean Claude Loraud (1994). "Exact and approximate Riemann solvers for real gases." In: *Journal of Computational Physics* 112.1, pp. 126–137 (cit. on pp. 380, 529).

Saurel, Richard and Carlos Pantano (2018). "Diffuse-interface capturing methods for compressible two-phase flows." In: *Annual Review of Fluid Mechanics* 50, pp. 105–130 (cit. on pp. 87, 122, 269).

SaveMyExams (n.d.). *Metallic Bonding*. URL: https://www.savemyexams.co.uk/igcse-chemistry-edexcel-new/revision-notes/metallic-bonding/metallic-bonding/ (cit. on p. 492).

Saye, R. I. (Jan. 2015). "High-Order Quadrature Methods for Implicitly Defined Surfaces and Volumes in Hyperrectangles." In: *SIAM Journal on Scientific Computing* 37.2, A993–A1019 (cit. on pp. 276, 286).

Saye, Robert (May 17, 2014). "High-Order Methods for Computing Distances to Implicitly Defined Surfaces." In: *Communications in Applied Mathematics and Computational Science* 9.1, pp. 107–141 (cit. on pp. 212, 240).

– (Sept. 2017a). "Implicit Mesh Discontinuous Galerkin Methods and Interfacial Gauge Methods for High-Order Accurate Interface Dynamics, with Applications to Surface Tension Dynamics, Rigid Body Fluid–Structure Interaction, and Free Surface Flow: Part I." In: *Journal of Computational Physics* 344, pp. 647–682 (cit. on pp. 232, 233, 270, 276, 285, 372).

– (2017b). "Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part II." In: *Journal of Computational Physics* 344, pp. 683–723 (cit. on pp. 235, 236, 244, 245, 251, 252, 256).
– (Nov. 19, 2020). "Fast Multigrid Solution of High-Order Accurate Multi-Phase Stokes Problems." In: *Communications in Applied Mathematics and Computational Science* 15.2, pp. 147–196. arXiv: 1911.05888 [cs, math] (cit. on p. 326).
– (n.d.). *Algoim* (*v2*) (cit. on p. 286).
Saye, Robert I and James A Sethian (2011). "The Voronoi implicit interface method for computing multiphase physics." In: *Proceedings of the National Academy of Sciences* 108.49, pp. 19498–19503 (cit. on p. 473).
Saye, Robert I. (Jan. 2022). "High-Order Quadrature on Multi-Component Domains Implicitly Defined by Multivariate Polynomials." In: *Journal of Computational Physics* 448, p. 110720 (cit. on pp. 273, 286).
Saye, Robert I. and James A. Sethian (2020). "A Review of Level Set Methods to Model Interfaces Moving under Complex Physics: Recent Challenges and Advances." In: *Handbook of Numerical Analysis*. Vol. 21. Elsevier, pp. 509–554 (cit. on p. 243).
Schillinger, Dominik et al. (2013). "A Review of the Finite Cell Method for Nonlinear Structural Analysis of Complex CAD and Image-Based Geometric Models." In: vol. 93, pp. 1–23. arXiv: 1807.01285 [math] (cit. on p. 270).
Schmidmayer, Kevin et al. (Apr. 2017). "A Model and Numerical Method for Compressible Flows with Capillary Effects." In: *Journal of Computational Physics* 334, pp. 468–496 (cit. on p. 243).
Schoeder, Svenja et al. (2020). "High-order cut discontinuous Galerkin methods with local time stepping for acoustics." In: *International Journal for Numerical Methods in Engineering* 121.13, pp. 2979–3003 (cit. on p. 275).
Schott, B. et al. (Oct. 30, 2016). "A Stabilized Nitsche-type Extended Embedding Mesh Approach for 3D Low- and High-Reynolds-number Flows." In: *International Journal for Numerical Methods in Fluids* 82.6, pp. 289–315 (cit. on p. 273).
Schott, Benedikt (2017). "Stabilized cut finite element methods for complex interface coupled flow problems." PhD thesis. Technische Universität München (cit. on pp. 55, 158, 270–272, 288, 318, 326, 353, 354).
Schott, Benedikt, Christoph Ager, and Wolfgang A Wall (2019). "Monolithic cut finite element–based approaches for fluid-structure interaction." In: *International Journal for Numerical Methods in Engineering* 119.8, pp. 757–796 (cit. on p. 360).
Schrooyen, P. et al. (2017). "A High-order Extended Discontinuous Galerkin Method for Moving Immersed Interface Problem." XDMS Conference (cit. on pp. 149, 286, 288, 465).
Schrooyen, P. et al. (2018). "A High-order Extended Discontinuous Galerkin Method to Treat Hydrodynamics Problems." ECCM-ECFD Conference (cit. on pp. 149, 288, 465).

Schrooyen, Pierre (2015). "Numerical Simulation of Aerothermal Flows through Ablative Thermal Protection Systems." PhD thesis. Université catholique de Louvain (cit. on pp. 23, 32, 58, 182, 470).

Schrooyen, Pierre et al. (June 24, 2013). "Discontinuous Galerkin Discretization Coupled with Sharp Interface Method for Ablative Materials." In: *21st AIAA Computational Fluid Dynamics Conference*. 21st AIAA Computational Fluid Dynamics Conference. San Diego, CA: American Institute of Aeronautics and Astronautics (cit. on p. 353).

Schrooyen, Pierre et al. (Dec. 2018). "Two-Way Coupled Simulations of Stagnation-Point Ablation with Transient Material Response." In: *International Journal of Thermal Sciences* 134, pp. 639–652 (cit. on p. 314).

Schulze, Lennart, Sachin KT Veettill, and Ivo F Sbalzarini (2023). "A high-order fully Lagrangian particle level-set method for dynamic surfaces." In: *arXiv preprint arXiv:2306.07986* (cit. on pp. 203, 212).

Scoggins, James B. and Thierry E. Magin (June 16, 2014). "Development of Mutation++: Multicomponent Thermodynamic and Transport Properties for Ionized Plasmas Written in C++." In: *11th AIAA/ASME Joint Thermophysics and Heat Transfer Conference*. 11th AIAA/ASME Joint Thermophysics and Heat Transfer Conference. Atlanta, GA: American Institute of Aeronautics and Astronautics (cit. on p. 73).

Sethian, J A (Feb. 20, 1996). "A Fast Marching Level Set Method for Monotonically Advancing Fronts." In: *Proceedings of the National Academy of Sciences* 93.4, pp. 1591–1595 (cit. on p. 211).

Sethian, J. A. and Peter Smereka (Jan. 2003). "Level set methods for fluid interfaces." In: *Annual Review of Fluid Mechanics* 35.1, pp. 341–372 (cit. on p. 204).

Shen, Yi, Yi Ren, and Hang Ding (Feb. 2020). "A 3D Conservative Sharp Interface Method for Simulation of Compressible Two-Phase Flows." In: *Journal of Computational Physics* 403, p. 109107 (cit. on p. 317).

Shima, Eiji and Keiichi Kitamura (Jan. 5, 2009). "On New Simple Low-Dissipation Scheme of AUSM-Family for All Speeds." In: *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*. 47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition. Orlando, Florida: American Institute of Aeronautics and Astronautics (cit. on p. 195).

Shu, Chi-Wang (2009). "Discontinuous Galerkin methods: general approach and stability." In: *Numerical solutions of partial differential equations* 201 (cit. on p. 169).

Shyue, Keh-Ming (Aug. 2001). "A Fluid-Mixture Type Algorithm for Compressible Multicomponent Flow with Mie–Grüneisen Equation of State." In: *Journal of Computational Physics* 171.2, pp. 678–707 (cit. on p. 74).

Shyue, Keh-Ming and Feng Xiao (July 2014). "An Eulerian Interface Sharpening Algorithm for Compressible Two-Phase Flow: The Algebraic THINC Approach." In: *Journal of Computational Physics* 268, pp. 326–354 (cit. on p. 269).

Simeonides, G. (1990). *The VKI hypersonic wind tunnels and associated measurement techniques*. Tech. rep. von Karman Institute (cit. on p. 443).

Sinha, Krishnendu and Ayanangshu Dey (2010). "Simulation of flow separation and reattachment on a re-entry capsule afterbody frustum." In: *48th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, p. 1561 (cit. on p. 19).

Slattery, John C, Leonard Sagis, and Eun-Suok Oh (2007). *Interfacial transport phenomena*. Springer Science & Business Media (cit. on pp. 82, 104).

Smuda, Martin (2021). "Direct Numerical Simulation of Multi-Phase Flows using Extended Discontinuous Galerkin Methods." PhD thesis. Technische Universität Darmstadt (cit. on pp. 221, 222, 234, 370, 475).

Smuda, Martin and Florian Kummer (Sept. 2021). "The Extended Discontinuous Galerkin Method Adapted for Moving Contact Line Problems via the Generalized Navier Boundary Condition." In: *International Journal for Numerical Methods in Fluids* 93.9, pp. 2921–2945 (cit. on pp. 290, 292).

– (Jan. 15, 2022). "On a Marching Level-set Method for Extended Discontinuous Galerkin Methods for Incompressible Two-phase Flows: Application to Two-dimensional Settings." In: *International Journal for Numerical Methods in Engineering* 123.1, pp. 197–225 (cit. on pp. 232, 233, 244, 372).

Sollie, W.E.H., O. Bokhove, and J.J.W. Van Der Vegt (Feb. 2011). "Space–Time Discontinuous Galerkin Finite Element Method for Two-Fluid Flows." In: *Journal of Computational Physics* 230.3, pp. 789–817 (cit. on p. 275).

Sollie, WEH, Jacobus JW van der Vegt, and Onno Bokhove (2006). "A space-time discontinuous Galerkin finite element method for two-fluid problems." In: *Journal of Computational Physics* 141, pp. 46–77 (cit. on p. 275).

Sonin, Ain A (2001). "Fundamental Laws of Motion for Particles, Material Volumes, and Control Volumes." In: *Massachusetts Institute of Technology, Cambridge, MA* (cit. on p. 94).

Soucasse, Laurent et al. (2016). "Flow-radiation coupling for atmospheric entries using a Hybrid Statistical Narrow Band model." In: *Journal of Quantitative Spectroscopy and Radiative Transfer* 180, pp. 55–69 (cit. on p. 470).

Span, R et al. (2001). "Multiparameter equations of state—recent trends and future challenges." In: *Fluid Phase Equilibria* 183, pp. 1–20 (cit. on p. 71).

Standards, National Institute of and Technology (n.d.). *Interatomic Potentials Repository*. URL: https://www.ctcms.nist.gov/potentials/ (cit. on p. 508).

Standart, G (1968). "The second law of thermodynamics for heterogenous flow systems—III Effect of the conditions of mechanical equilibrium and electroneutrality on simultaneous heat and mass transfer and the Prigogine theorem." In: *Chemical Engineering Science* 23.3, pp. 279–285 (cit. on p. 100).

Steger, Joseph L and RF617098 Warming (1981). "Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods." In: *Journal of computational physics* 40.2, pp. 263–293 (cit. on p. 168).

Steinberg, Stanly and Patrick J Roache (1985). "Symbolic manipulation and computational fluid dynamics." In: *Journal of Computational Physics* 57.2, pp. 251–284 (cit. on p. 514).

Sun, Pengtao (Aug. 2019). "Fictitious Domain Finite Element Method for Stokes/Elliptic Interface Problems with Jump Coefficients." In: *Journal of Computational and Applied Mathematics* 356, pp. 81–97 (cit. on p. 270).

Sun, Y. and C. Beckermann (Nov. 2004). "Diffuse Interface Modeling of Two-Phase Flows Based on Averaging: Mass and Momentum Equations." In: *Physica D: Nonlinear Phenomena* 198.3-4, pp. 281–308 (cit. on pp. 122, 125).

Sussman, Mark and Emad Fatemi (1999). "An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow." In: *SIAM Journal on scientific computing* 20.4, pp. 1165–1191 (cit. on p. 202).

Sussman, Mark and Elbridge Gerry Puckett (2000). "A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows." In: *Journal of computational physics* 162.2, pp. 301–337 (cit. on pp. 202, 473).

Sussman, Mark, Peter Smereka, and Stanley Osher (1994). "A level set approach for computing solutions to incompressible two-phase flow." In: *Journal of Computational physics* 114.1, pp. 146–159 (cit. on pp. 212, 404, 405).

Tellep, DM (1959). "The effect of vehicle deceleration on a melting surface." In: *Journal of the Aerospace Sciences* 26.8, pp. 537–538 (cit. on p. 19).

Theillard, Maxime (Oct. 2021). "A Volume-Preserving Reference Map Method for the Level Set Representation." In: *Journal of Computational Physics* 442, p. 110478 (cit. on p. 203).

Theillard, Maxime, Frédéric Gibou, and Tresa Pollock (May 2015). "A Sharp Computational Method for the Simulation of the Solidification of Binary Alloys." In: *Journal of Scientific Computing* 63.2, pp. 330–354 (cit. on p. 114).

Thompson, Ross Anthony (2015). *Galerkin projections between finite element spaces* (cit. on p. 307).

Tornberg, Anna-Karin and Björn Engquist (May 2000). "A Finite Element Based Level-Set Method for Multiphase Flow Applications." In: *Computing and Visualization in Science* 3.1-2, pp. 93–101 (cit. on pp. 245, 250).

Toro, E. F. (2009). *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. 3rd ed. Dordrecht ; New York: Springer. 724 pp. (cit. on p. 167).

– (Nov. 2019). "The HLLC Riemann Solver." In: *Shock Waves* 29.8, pp. 1065–1082 (cit. on p. 396).

Trujillo, Mario F, Lakshman Anumolu, and Doug Ryddner (2017). "The distortion of the level set gradient under advection." In: *Journal of Computational Physics* 334, pp. 81–101 (cit. on pp. 219, 228).

Tryggvason, Grétar, Ruben Scardovelli, and Stéphane Zaleski (2011). *Direct numerical simulations of gas–liquid multiphase flows*. Cambridge university press (cit. on p. 83).

Tryggvason, Grétar et al. (2001). "A front-tracking method for the computations of multiphase flow." In: *Journal of computational physics* 169.2, pp. 708–759 (cit. on p. 199).

Tsai, Yen-Hsi Richard et al. (Jan. 2003). "Fast Sweeping Algorithms for a Class of Hamilton–Jacobi Equations." In: *SIAM Journal on Numerical Analysis* 41.2, pp. 673–694 (cit. on p. 212).

Turchi, Alessandro et al. (2019). "Assessment of the effect of heat-shield ablation on the aerodynamic performance of re-entry capsules in hypersonic flows." In: *8th European Conference for Aeronautics and Aerospace Sciences* (cit. on p. 448).

Turkel, E. (May 1993). "Review of Preconditioning Methods for Fluid Dynamics." In: *Applied Numerical Mathematics* 12.1-3, pp. 257–284 (cit. on p. 190).

Uekermann, Benjamin Walter (2016). "Partitioned fluid-structure interaction on massively parallel systems." PhD thesis. Technische Universität München (cit. on pp. 314, 315).

Utz, Thomas (2018). "Level set methods for high-order unfitted discontinuous Galerkin schemes." PhD thesis. TU Darmstadt (cit. on pp. 220, 224, 353, 356).

Utz, Thomas and Florian Kummer (Mar. 20, 2018). "A High-Order Discontinuous Galerkin Method for Extension Problems." In: *International Journal for Numerical Methods in Fluids* 86.8, pp. 509–518 (cit. on pp. 221, 473).

Utz, Thomas, Florian Kummer, and Martin Oberlack (June 10, 2017). "Interface-Preserving Level-Set Reinitialization for DG-FEM." In: *International Journal for Numerical Methods in Fluids* 84.4, pp. 183–198 (cit. on p. 212).

Val, Ana Isabel del, Thierry Magin, and Olivier Chazot (2017). "Uncertainty Assessment on the Characterization of Testing Conditions in Arc-jet Facilities." In: *33rd AIAA Aerodynamic Measurement Technology and Ground Testing Conference*, p. 4526 (cit. on p. 442).

Val, Anabel del (2021). "Bayesian calibration and assessment of gas-surface interaction models and experiments in atmospheric entry plasmas." PhD thesis. Institut polytechnique de Paris; Ecole Polytechnique (cit. on p. 16).

Valencia, Juan J and PN Quested (2001). "Thermophysical properties." In: *Modeling for casting and solidification processing* 189 (cit. on p. 73).

Van Leer, Bram (1982). "Flux-vector splitting for the Euler equations." In: *Eighth International Conference on Numerical Methods in Fluid Dynamics: Proceedings of the Conference, Rheinisch-Westfälische Technische Hochschule Aachen, Germany, June 28–July 2, 1982*. Springer, pp. 507–512 (cit. on pp. 168, 194).

Vandenhoeck, Ray and Andrea Lani (Sept. 2019). "Implicit High-Order Flux Reconstruction Solver for High-Speed Compressible Flows." In: *Computer Physics Communications* 242, pp. 1–24. arXiv: 1811.08666 [physics] (cit. on pp. 303–305).

Vandergrift, Jakob and Florian Kummer (2023). "An Extended Discontinuous Galerkin Method for High-Order Shock Treatment." In: *Available at SSRN 4455785* (cit. on p. 304).

Veluri, Subrahmanya P, Christopher J Roy, and Edward A Luke (2012). "Comprehensive code verification techniques for finite volume CFD codes." In: *Computers & fluids* 70, pp. 59–72 (cit. on p. 514).

Venkateswaran, S and L Merkle (1999). "Analysis of preconditioning methods for the Euler and Navier-Stokes equations." In: *Lecture series-van Kareman Institute for fluid dynamics* 3, B1–B155 (cit. on pp. 188, 189).

Vigneron, Didier, Geoffrey Deliége, and Jean-André Essers (2006). "Low Mach number local preconditioning for unsteady viscous finite volumes simulations on 3D unstructured grids." In: *European conference on computational fluid dynamics-ECCOMAS CFD 2006*. Delft University of Technology, Delft, Netherlands (cit. on p. 192).

Vittoz, L. et al. (Jan. 2019). "Comparisons of Weakly-Compressible and Truly Incompressible Approaches for Viscous Flow into a High-Order Cartesian-grid Finite Volume Framework." In: *Journal of Computational Physics: X* 1, p. 100015 (cit. on p. 69).

Vogl, Chris J. (Jan. 2016). "A Curvature-Augmented, REA Approach to the Level Set Method." In: *SIAM Journal on Scientific Computing* 38.2, A833–A855 (cit. on p. 244).

Vojta, Thomas (n.d.). *Physics 5403: Computational Physics - Chapter 6: Molecular Dynamics* (cit. on p. 488).

Voller, Vaughan R, AD Brent, and C Prakash (1989). "The modelling of heat, mass and solute transport in solidification systems." In: *International Journal of Heat and Mass Transfer* 32.9, pp. 1719–1731 (cit. on pp. 133, 139).

Wang, Yuntian et al. (June 10, 2021). "Hugoniot States and Mie–Grüneisen Equation of State of Iron Estimated Using Molecular Dynamics." In: *Crystals* 11.6, p. 664 (cit. on p. 74).

Wanner, Gerhard and Ernst Hairer (1996). *Solving ordinary differential equations II*. Vol. 375. Springer Berlin Heidelberg New York (cit. on p. 341).

Watkins, David S (2004). *Fundamentals of matrix computations*. John Wiley & Sons (cit. on p. 271).

Weiss, Jonathan M. and Wayne A. Smith (Nov. 1995). "Preconditioning Applied to Variable and Constant Density Flows." In: *AIAA Journal* 33.11, pp. 2050–2057 (cit. on pp. 190, 191).

Westerheide, Sebastian (2018). "Unfitted Discontinuous Galerkin Schemes for Applications with PDEs on Complex-Shaped Surfaces." In: (cit. on p. 269).

Weston, Brian, Jean-Pierre Delplanque, and Robert Nourgaliev (July 30, 2017). "Preconditioning for the All-Speed Compressible Navier-Stokes Equations With Laser-Induced Phase Change." In: *ASME 2017 Fluids Engineering Division Summer Meeting*. ASME 2017 Fluids Engineering Division Summer Meeting. Waikoloa, Hawaii, USA: American Society of Mechanical Engineers, V01BT11A005 (cit. on p. 472).

Weston, Brian et al. (Nov. 2019). "Preconditioning a Newton-Krylov Solver for All-Speed Melt Pool Flow Physics." In: *Journal of Computational Physics* 397, p. 108847 (cit. on pp. 181, 340, 350).

Wolowicz, Chester H, JS Brown Jr, and William P Gilbert (1979). *Similitude requirements and scaling relationships as applied to model testing*. Tech. rep. NASA (cit. on p. 438).

Wood, Albert Beaumont (1930). *A textbook of sound*. Vol. 1. G. Bell and Sons Ltd. (cit. on p. 141).

Woodruff, D. P. (July 1968). "The Morphology of the Solid/Liquid Interface during Melting." In: *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics* 18.151, pp. 123–127 (cit. on p. 111).

Xiao, Cheng-Nian, Fabian Denner, and Berend GM van Wachem (2017). "Fully-coupled pressure-based finite-volume framework for the simulation of fluid flows at all speeds in complex geometries." In: *Journal of Computational Physics* 346, pp. 91–130 (cit. on p. 180).

Xu, Liang and Tiegang Liu (June 2011). "Accuracies and Conservation Errors of Various Ghost Fluid Methods for Multi-Medium Riemann Problem." In: *Journal of Computational Physics* 230.12, pp. 4975–4990 (cit. on p. 317).

Xu, Liang, Wubing Yang, and Tiegang Liu (Jan. 2022). "An Interface Treatment for Two-Material Multi-Species Flows Involving Thermally Perfect Gases with Chemical Reactions." In: *Journal of Computational Physics* 448, p. 110707 (cit. on p. 316).

Yan, Zhenguo (2021). "Efficient implicit spectral/hp element DG techniques for compressible flows." PhD thesis. Imperial College London (cit. on pp. 343, 344).

Yeoh, Guan Heng and Jiyuan Tu (2010). *Computational Techniques for Multi-Phase Flows: Basics and Applications*. Oxford: Butterworth-Heinemann Elsevier (cit. on p. 118).

Yoo, Young-Lin and Hong-Gye Sung (Sept. 2021). "A Hybrid AUSM Scheme (HAUS) for Multi-Phase Flows with All Mach Numbers." In: *Computers & Fluids* 227, p. 105050 (cit. on pp. 195, 196).

Ytrehus, T and S Østmo (1996). "Kinetic theory approach to interphase processes." In: *International journal of multiphase flow* 22.1, pp. 133–155 (cit. on p. 103).

Zahr, Matthew J and P-O Persson (2018). "An optimization-based approach for high-order accurate discretization of conservation laws with discontinuous solutions." In: *Journal of Computational Physics* 365, pp. 105–134 (cit. on p. 199).

Zahr, Matthew J. (2021). *AME40541/60541: Finite Element Methods* (cit. on pp. 153, 172).

Zeifang, Jonas and Andrea Beck (June 2023). "A Low Mach Number IMEX Flux Splitting for the Level Set Ghost Fluid Method." In: *Communications on Applied Mathematics and Computation* 5.2, pp. 722–750 (cit. on p. 342).

Zha, G-C and E Bilgen (1993). "Numerical solutions of Euler equations by using a new flux vector splitting scheme." In: *International Journal for Numerical Methods in Fluids* 17.2, pp. 115–144 (cit. on p. 168).

Zhang, G.G.Z. and D. Zhou (2017). "Crystalline and Amorphous Solids." In: *Developing Solid Oral Dosage Forms*. Elsevier, pp. 23–57 (cit. on p. 113).

Zhang, Jiaqi and Pengtao Yue (Feb. 2019). "A High-Order and Interface-Preserving Discontinuous Galerkin Method for Level-Set Reinitialization." In: *Journal of Computational Physics* 378, pp. 634–664 (cit. on p. 212).

Zhang, Xiangxiong (2017). "On positivity-preserving high order discontinuous Galerkin schemes for compressible Navier–Stokes equations." In: *Journal of Computational Physics* 328, pp. 301–343 (cit. on pp. 159, 170).

Zhang, Xiangxiong and Chi-Wang Shu (2011). "Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: survey and new developments." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 467.2134, pp. 2752–2776 (cit. on p. 169).

Zhang, Xiaotao et al. (Oct. 2022). "A Second-order Modified Ghost Fluid Method (2nd-MGFM) with Discontinuous Galerkin Method for 1-D Compressible Multi-medium Problem with Cylindrical and Spherical Symmetry." In: *Journal of Scientific Computing* 93.1, p. 14 (cit. on p. 316).

Zhang, Yu et al. (Sept. 2019). "A Locally Discontinuous ALE Finite Element Formulation for Compressible Phase Change Problems." In: *Journal of Computational Physics* 393, pp. 438–464 (cit. on pp. 267, 317).

Zong, Nan and Vigor Yang (June 2007). "An Efficient Preconditioning Scheme for Real-Fluid Mixtures Using Primitive Pressure–Temperature Variables." In: *International Journal of Computational Fluid Dynamics* 21.5-6, pp. 217–230 (cit. on pp. 190, 191).

Zou, Ziqiang (2021). "A Sharp Interface Method for Low Mach Two-Phase Flows with Phase Change." PhD thesis. Université Paris-Saclay (cit. on p. 403).