

Université Paris VII - Denis Diderot



Université Pierre et Marie Curie

École Doctorale Paris Centre

Thèse de doctorat

Discipline : Mathématiques Appliquées

présentée par

Charles Dapogny

Shape optimization, level set methods on unstructured meshes and mesh evolution

dirigée par Grégoire Allaire & Pascal Frey

Rapporteurs :

Dorin Bucur	Université de Savoie
Antoine Henrot	École des Mines de Nancy
James Sethian	University of Berkeley, California

Soutenue le 4 Décembre 2013 devant le jury composé de :

Marc Albertelli	Technocentre Renault	Encadrant industriel
Grégoire Allaire	Centre de Mathématiques Appliquées (CMAP)	Directeur de thèse
Jérôme Fehrenbach	Institut de Mathématiques de Toulouse	Examinateur
Pascal Frey	Laboratoire Jacques-Louis Lions	Directeur de thèse
Frédéric Hecht	Laboratoire Jacques-Louis Lions	Examinateur
Antoine Henrot	École des Mines de Nancy	Rapporteur
Antony Patera	Massachussets Institute of Technology	Examinateur

Laboratoire Jacques-Louis Lions
4 place Jussieu
75 005 Paris

École doctorale Paris centre Case 188
4 place Jussieu
75 252 Paris cedex 05

Remerciements

Dur, dur, de faire original dans l'écriture des remerciements... A défaut, j'espère simplement avoir l'occasion de témoigner de mon affection sincère pour les gens que j'apprécie, et à qui je n'ai que trop rarement eu la présence d'esprit de dire explicitement combien ils ont compté, et continuent de compter pour moi.

Mes tous premiers remerciements s'adressent naturellement aux trois personnes qui ont dirigé et encadré cette thèse. Depuis le master, Grégoire m'a toujours témoigné beaucoup d'attention lorsqu'il s'agissait d'essayer de me communiquer une modeste partie de son savoir encyclopédique, de me corriger ou de me conseiller. Pascal n'a pas hésité non plus à partager (autour d'un, ou de nombreux cafés) beaucoup plus que ses grandes connaissances en maillage, ou ses idées d'application insolites du calcul scientifique. Enfin, cette alchimie n'aurait sans doute pas été possible sans Marc, dont la grande culture n'a d'égale que le dynamisme invariable. Je leur dois en grande partie le fait que ces années aient été si riches pour moi, et pas du seul point de vue mathématique !

Je suis très flatté que Dorin Bucur, Antoine Henrot et James Sethian, dont les livres et articles de référence ont alimenté mes lectures durant ces années, aient accepté de donner de leur rare temps libre pour rapporter cette thèse. Jérôme Fehrenbach, Frédéric Hecht et Antony Patera ont également accepté de prendre part au jury de soutenance, et j'en suis très honoré.

J'ai bien conscience d'avoir pu préparer cette thèse dans d'excellentes conditions et un environnement extraordinaire - en réalité des environnements extraordinaires.

Malgré une présence épisodique, j'ai passé d'excellents moments dans l'équipe optimisation de Renault. Les croissants du matin, discussions politiques et interminables séances de vannes (et de travail) partagées avec Marc, Fred, CriCri, Pascal, Laurent, Paul, Yves y sont évidemment pour beaucoup... Mention spéciale à Pierre, ses piques et ses blagues, composantes incontournables d'une virée californienne mémorable !

J'ai pu profiter au laboratoire Jacques-Louis Lions d'une atmosphère de travail sans pareille. Des discussions matinales autour d'un café avec Frédéric Hecht, Jean-Yves Chemin, (et bien sûr Pascal) aux cours du soir - ou de la nuit - de François Murat, j'ai eu la chance d'y rencontrer et d'y croiser régulièrement des gens d'une grande gentillesse, qui ont laissé leur porte ouverte pour accueillir mes questions mathématiques ou métaphysiques parfois très naïves : Albert Cohen, Yvon Maday, Edwige Godlewski, François Jouve, Frédéric Lagoutière, Cécile Dobrzynski, Frédéric Coquel, Laurent Boudin, Benjamin Stamm... Merci enfin à Nadine et Catherine - et plus anciennement Mme Boulic et Ruprecht - pour leur aide face aux remparts de l'administration, à Salima, Khash, Altaïr, qui non contents d'être indispensables au fonctionnement du laboratoire, rendent ce lieu si agréable, drôle et vivant.

Cette tranche de vie, comme disent les anciens, n'aurait pas pu bien se passer sans les super rencontres que j'y ai faites, et tout d'abord mes trois 'frère et soeurs' de thèse, LiMa, Nicole, Mamadou; merci à vous trois pour les cinés, les apéros, les grandes soirées de N'Importe Quoi, et juste... pour avoir été là ! Merci aussi à J.-P. (et tant pis si Jo s'est fait rosser), Casimir, Giacomo, Benjamin, Oana, Tommaso, Pierre L. (et son Hamster), Pierre J., Maxime, Nastasia, Malik, Thibault B., Thibault L., Eugénie, Vincent (ma nemesis au Mayflower), Dorian, Anne-Céline, Magali, Juliette, Yannick le taïwanais, Jérémy, Étienne, Mai, Nico C., Nico K. Extra muros, mille mercis à Georgios, pour sa générosité, les souvenirs inoubliables en Grèce, aux États-Unis, à Paris, et même au Danemark, à Harsha et Matteo, fraîchement partis, Jean-Léopold et Aymeric, fraîchement arrivés, et... très en retard, à Gabriel évidemment !

Je ne saurais terminer convenablement ces remerciements sans évoquer mes amis de plus longue date.

Un grand merci à Fneuh, Mekki, Pougnezu, pour toutes ces soirées de chombier (malheureusement plus rares ces derniers temps, en grande partie par ma faute), Nico, pour des vacances rafraîchissantes en Islande et en Norvège, Manu, idem, et pour les leçons (dans les deux sens du terme) de tennis, Peej, David, Jonathan, PC, La Coisse,... Merci aussi au petit groupe toulousaing, Lucas, Britney, Gwen,... pour les quelques week-end que j'ai adoré passer avec vous !

Bien sûr, pour finir, un énorme merci à la fine équipe rémoise, mes plus vieux amis entre tous : Alex, Mathieu/Patoche (plus jamais ça !), Nono, Maro (plus de tractions !), Clém, Stéphane (fous ta cagoule !), Joe, Bber, pour les nanars, les Fujin, les visites et tucages chez l'Oncle à Lamelle (RIP), et ces innombrables soirées complètement absurdes, depuis de si nombreuses années, que même des voisins en furie n'ont jamais réussi à arrêter.

Une dernière pensée pour les six autres membres de ma famille, à qui je dois bien plus que j'ai jamais osé leur montrer, pour des choses compliquées (une aventure népalaise), comme des choses plus simples (un repas, une bière, une discussion, une balade, ...) mais toujours plus essentielles, quand tout va bien, ou moins bien.

Résumé

Optimisation de formes, méthode des lignes de niveaux sur maillages non structurés et évolution de maillages

Résumé

L'objectif principal de cette thèse est de concevoir une méthode d'optimisation de structures qui jouit d'une description exacte (i.e. au moyen d'un maillage) de la forme à chaque itération du processus, tout en bénéficiant des avantages de la méthode des lignes de niveaux lorsqu'il s'agit de suivre leur évolution. Indépendamment, on étudie également deux problèmes de modélisation en optimisation structurale.

Dans une première partie bibliographique, on présente quelques notions classiques, ainsi qu'un état de l'art sommaire autour des trois thématiques principales de la thèse - méthode des lignes de niveaux (Chapitre 1), optimisation de formes (Chapitre 2) et maillage (Chapitre 3).

La seconde partie de ce manuscrit traite de deux questions en optimisation de formes, celle de la répartition optimale de plusieurs matériaux au sein d'une structure donnée (Chapitre 4), et celle de l'optimisation robuste de fonctions dépendant du domaine lorsque des perturbations s'exercent sur le modèle (Chapitre 5).

Dans une troisième partie, on étudie la conception de schémas numériques en lien avec la méthode des lignes de niveaux lorsque le maillage de calcul est simplicial (et potentiellement adapté). Le calcul de la distance signée à un domaine est étudié dans le chapitre 6, et la résolution de l'équation de transport d'une fonction 'level set' est détaillée dans le chapitre 7.

La quatrième partie (Chapitre 8) traite des aspects de la thèse liés à la modification locale de maillages surfaciques et volumiques.

Enfin, la dernière partie (Chapitre 9) détaille la stratégie conçue pour l'évolution de maillage en optimisation de formes, à partir des ingrédients des chapitres 6, 7 et 8.

Mots-clefs

Méthode des lignes de niveaux, optimisation de formes, maillage, fonction de distance signée, equation d'advection, simulation numérique $3d$.

Shape optimization, level set methods on unstructured meshes and mesh evolution

Abstract

The main purpose of this thesis is to propose a method for structural optimization which combines the accuracy of featuring an exact description of shapes (i.e. with a mesh) at each iteration of the process with the versatility of the level set method for tracking their evolution. Independently, we also study two problems related to modeling in structural optimization.

In the first, bibliographical part, we present several classical notions, together with some recent developments about the three main issues of this thesis - namely level set methods (Chapter 1), shape optimization (Chapter 2), and meshing (Chapter 3).

The second part of this manuscript deals with two issues in shape optimization, that of the optimal repartition of several materials within a fixed structure (Chapter 4), and that of the robust optimization of functions depending on the domain when perturbations are expected over the considered mechanical model.

In the third part, we study the design of numerical schemes for performing the level set method on simplicial (and possibly adapted) computational meshes. The computation of the signed distance function to a domain is investigated in Chapter 6, and the resolution of the level set advection equation is presented in Chapter 7.

The fourth part (Chapter 8) is devoted to the meshing techniques introduced in this thesis.

Eventually, the last part (Chapter 9) describes the proposed strategy for mesh evolution in the context of shape optimization, relying on the numerical ingredients introduced in Chapters $7, 8, 9$.

Keywords

Level set methods, shape optimization, meshing, signed distance function, advection equation, three-dimensional numerical simulation.

Contents

Introduction **11**

I Background and state of the art 25

1 The level set method **27**
1.1 Presentation of the level set method . . . 29
1.1.1 Implicitly-defined domains and geometry . . . 29
1.1.2 Main notations and first examples . . . 29
1.1.3 From an explicit to an implicit description of the evolution . . . 30
1.1.4 Domain evolution as a boundary value problem: Eikonal equations . . . 34
1.2 Numerical algorithms for the level set method . . . 36
1.2.1 Solving the Level Set Hamilton-Jacobi equation on Cartesian grids . . . 36
1.2.2 Solving the Level Set Hamilton-Jacobi equation on triangular meshes . . . 38
1.2.3 Semi-Lagrangian schemes . . . 39
1.3 Initializing level set functions . . . 40
1.3.1 The fast marching method . . . 41
1.3.2 The fast sweeping method . . . 43
1.3.3 Re-initializing level set functions . . . 45

2 Shape optimization **47**
2.1 A quick overview of shape optimization and applications . . . 49
2.1.1 An overview of the main methods . . . 49
2.1.2 Numerical difficulties in shape optimization: non existence of optimal shapes . . . 51
2.2 Shape sensitivity analysis using Hadamard's boundary variation method . . . 53
2.2.1 Hadamard's boundary variation method . . . 53
2.2.2 Shape differentiability and computation of shape derivatives in linear elasticity . . . 55
2.2.3 Shape optimization using Hadamard's method . . . 62
2.3 Shape optimization using the level set method . . . 64

3 Mesh generation, modification and evolution **67**
3.1 Generalities around meshes: definitions, notations, and useful concepts . . . 68
3.1.1 Definitions and notations . . . 68
3.1.2 Appraising the quality of a mesh . . . 70
3.1.3 The Riemannian paradigm for size and orientation specifications in meshing . . . 71
3.2 Mesh generation techniques . . . 73
3.2.1 Two and three-dimensional 'volume' mesh generation . . . 73
3.2.2 Surface mesh generation . . . 85
3.3 Local remeshing . . . 87

3.3.1 Volume remeshing . . . 87
3.3.2 Surface remeshing . . . 90
3.4 Mesh evolution . . . 92
3.4.1 Purely Lagrangian methods . . . 92
3.4.2 Hybrid methods . . . 96

II Two problems in shape optimization 99

4 Multi-phase optimization via a level set method 101
4.1 Introduction . . . 102
4.2 Around the shape differentiability of the signed distance function . . . 104
4.2.1 Some facts around the signed distance function . . . 104
4.2.2 Shape derivative of the signed distance function . . . 106
4.2.3 Another expression for these derivatives . . . 111
4.2.4 More differentiability results for the signed distance function . . . 113
4.2.5 Some words around the notion of minimum thickness of a domain . . . 120
4.3 Sharp-interface formulation in a fixed mesh framework . . . 124
4.3.1 Description of the problem . . . 124
4.3.2 Shape-sensitivity analysis of the sharp-interface problem . . . 125
4.4 Shape derivative in the smoothed-interface context . . . 131
4.4.1 Description of the problem . . . 131
4.4.2 Shape derivative of the compliance in the multi-materials setting . . . 131
4.4.3 Approximate formulas for the shape derivative . . . 133
4.4.4 Convergence of the smoothed-interface shape optimization problem to the sharp-interface problem . . . 134
4.5 Extension to more than 2 materials . . . 136
4.6 Discussion and comparison with previous formulae in the literature . . . 138
4.7 Numerical results . . . 140
4.7.1 Level-set representation . . . 140
4.7.2 Two materials in the sharp interface context . . . 141
4.7.3 Two materials in the smoothed-interface context . . . 142
4.7.4 Four phases in the smoothed interface context . . . 145
4.8 Appendix: convergence of the smoothed-interface shape optimization problem to its sharp-interface equivalent . . . 150
4.8.1 A model problem in the context of thermal conductivity . . . 150
4.8.2 Extension to the context of linearized elasticity . . . 159

5 A linearized approach to worst-case design in parametric and geometric shape optimization 163
5.1 Introduction . . . 164
5.2 General setting and main notations . . . 165
5.3 Worst-case design in parametric optimization . . . 167
5.3.1 Description of the model problem . . . 167
5.3.2 Worst-case design of an elastic plate under perturbations on the body forces . . . 168
5.3.3 Extension to a worst-case optimization problem, with respect to a perturbation on surface loads . . . 176
5.3.4 Parametric optimization of a worst-case scenario problem under geometric uncertainty 177
5.3.5 Worst-case design with uncertainties over the elastic material's properties . . . 181
5.4 Worst-case design in shape optimization . . . 183
5.4.1 Description of the model problem . . . 183

5.4.2 Worst-case design in shape optimization under uncertainties over the applied body forces 184
5.4.3 Worst-case design in shape optimization under uncertainties on the Lamé moduli of the material . . . 188
5.4.4 Worst-case design in shape optimization under geometric uncertainties . . . 190
5.5 Numerical results . . . 198
5.5.1 Worst-case optimization problems in parametric structural optimization . . . 198
5.5.2 Examples of shape optimization problems under uncertainties . . . 200
5.5.3 General tools . . . 218
5.5.4 Several Green's formulae . . . 219

III Level set methods on unstructured meshes; connections with mesh adaptation 223

6 Computation of the signed distance function to a discrete contour on adapted simplicial mesh. 225
6.1 Introduction . . . 226
6.2 Some preliminaries about the signed distance function . . . 226
6.3 A short study of some properties of the solution to the unsteady Eikonal equation . . . 228
6.4 A numerical scheme for the signed distance function approximation . . . 230
6.4.1 Extending the signed distance function from the boundary . . . 230
6.4.2 Initialization of the signed distance function near $\partial\Omega$ . . . 231
6.5 Mesh adaptation for a sharper approximation of the signed distance function . . . 233
6.5.1 Anisotropic mesh adaptation . . . 234
6.5.2 Computation of a metric tensor associated to the minimization of the $\mathbb{P}^1$ interpolation error . . . 235
6.5.3 Mesh adaptation for a geometric reconstruction of the 0 level set of a function . . . 236
6.6 A remark about level-set redistancing . . . 239
6.7 Extension to the computation of the signed distance function in a Riemannian space . . . 240
6.8 Numerical Examples . . . 240

7 An accurate anisotropic adaptation method for solving the level set advection equation 249
7.1 Introduction . . . 250
7.2 Some theoretical facts around the advection equation . . . 251
7.3 The proposed numerical method, and its error analysis . . . 252
7.3.1 A numerical method for the advection equation based on the method of characteristics 252
7.3.2 A priori error analysis of the proposed method . . . 253
7.3.3 A priori error estimate in terms of Hausdorff distance in the case of level-set functions 257
7.4 Mesh adaptation for the advection equation . . . 258
7.4.1 Metric-based mesh adaptation . . . 258
7.4.2 The proposed adaptation method . . . 259
7.5 Additional numerical features . . . 262
7.5.1 The need for mesh gradation control . . . 262
7.5.2 Importance of redistancing . . . 263
7.6 Numerical examples . . . 264
7.6.1 Rotation of Zalesak's slotted disk . . . 264
7.6.2 Time-reversed vortex flow . . . 265
7.6.3 Deformation test flow . . . 266
7.6.4 Rotation of Zalesak's sphere . . . 268
7.6.5 Three-dimensional deformation test case . . . 270
Appendix . . . 274

IV Three-dimensional surface and domain remeshing 275

8 Discrete three-dimensional surface and domain remeshing 277
8.1 Remeshing of surface triangulations . . . 280
8.1.1 Reconstruction of the geometry . . . 281
8.1.2 Local reconstruction of the ideal surface from the discrete geometry . . . 284
8.1.3 Description of the local remeshing operators . . . 286
8.1.4 Definition of a size map adapted to the geometric approximation of a surface . . . 290
8.1.5 The complete strategy . . . 296
8.1.6 Numerical examples . . . 297
8.2 Discrete surface remeshing in the anisotropic context . . . 303
8.2.1 A wee bit of Riemannian geometry . . . 303
8.2.2 Numerical approximation of parallel transport on a submanifold of $\mathbb{R}^d$ . . . 310
8.2.3 Definition of a suitable Riemannian structure for anisotropic surface remeshing . . . 315
8.2.4 Metric tensor fields on triangulated surfaces in numerical practice . . . 318
8.2.5 Geometric anisotropic surface remeshing of a discrete surface . . . 320
8.2.6 Numerical examples . . . 323
8.3 Discrete three-dimensional domain remeshing . . . 327
8.3.1 Description of the local remeshing operators . . . 328
8.3.2 Construction of a size map adapted to the geometric approximation of the ideal domain 331
8.3.3 The complete remeshing strategy . . . 332
8.3.4 Numerical examples . . . 334
8.4 Implicit domain meshing and applications . . . 339
8.4.1 Explicit discretization of the 0 level set of $\phi_{\mathcal{T}}$ into $\mathcal{T}$ . . . 340
8.4.2 From discrete domain remeshing to discrete domain and subdomains remeshing . . . 341
8.4.3 Numerical examples and applications . . . 343

V A level-set based method for mesh evolution for shape optimization 349

9 Shape optimization with a level set based mesh evolution method 351
9.1 Introduction . . . 352
9.2 A model problem in shape optimization of elastic structures . . . 354
9.3 Two complementary ways for representing shapes . . . 356
9.3.1 Generating the signed distance function to a discrete domain . . . 356
9.3.2 Meshing the negative subdomain of a scalar function . . . 357
9.4 Accounting for shape evolution . . . 360
9.4.1 A brief reminder of the level set method . . . 360
9.4.2 Resolution of the level set advection equation on an unstructured mesh . . . 361
9.4.3 Computation of a descent direction . . . 361
9.5 The global algorithm . . . 362
9.6 Numerical examples . . . 363
9.6.1 Minimization of the compliance . . . 363
9.6.2 Multi-loads compliance minimization . . . 371
9.6.3 Chaining topological and geometric optimization . . . 373
9.6.4 Multi-materials compliance minimization . . . 376
9.6.5 Minimization of least-square criteria . . . 379
9.6.6 Stress criterion minimization . . . 383
9.7 Conclusions and perspectives . . . 384

Bibliography 389

Introduction

This thesis is devoted to a large extent to the design of a mesh evolution strategy in the context of structural shape optimization; by extension, it also addresses the topics of level set methods on unstructured computational meshes, meshing techniques, and, on a rather independent basis, it analyzes two specific problems in structural shape optimization.

The manuscript is composed of nine chapters, grouped into five parts, which can be read independently from one another, insofar as possible (to the cost of some redundancies between them). Each chapter contains an introduction to the tackled topic and provides associated references. This introduction is however more general, and contains neither technical details, nor references.

In an attempt to enlighten the structure of this document, it may be worthwhile to give an idea of the prime motivations of the work at stake.

The main concern of this work is about shape optimization problems, which can, broadly speaking, be formulated as the minimization of an objective function $J(\Omega)$ of the domain variable Ω. In this way, much like in the case of more 'classical' optimization problems, the study of the derivative of J with respect to the domain makes it possible to compute a descent direction for J from a given shape Ω, as a vector field V_Ω - see the introductory material in Chapter 2 for a more technical explanation, and related bibliographical references. In other terms, trading Ω for $\Omega(t) := (I + tV_\Omega)(\Omega)$ for sufficiently small $t > 0$ allows for a decrease in the value of J.

At this point, difficulties arise, which are not specific features of shape optimization problems, but are on the contrary encountered in the study of most free or moving boundary problems:

- The practical computation of the descent direction V_Ω is not trivial; in the cases we shall be interested in, it requires the resolution of one, or several PDE systems posed on Ω (typically linearized elasticity systems). Numerical methods for solving such PDE systems are numerous (e.g. the finite element method), yet most of them rely on Ω being equipped with a computational mesh.
- Advecting Ω along the velocity field V_Ω (i.e. updating Ω to $\Omega(t)$) is fairly straightforward in the theoretical framework, and unfortunately much harder in numerical practice. In particular, it inherently depends on how Ω is parametrized. For instance, if Ω is described by a mesh, the naive and very tempting operation of just 'translating' the associated vertices in the direction of V_Ω is very likely to produce an ill-shaped (or even invalid) mesh for the new shape $\Omega(t)$ (see the example in Figure 1, where the orientations of some displaced triangles have been inverted). In general, mesh evolution is a difficult issue, especially in three space dimensions (see Chapter 3 for a discussion, and a presentation of several techniques).

So as to reconcile the antagonist requirements of the computation of a descent direction for J and of the description of the domain evolution, several authors proposed to combine the aforementioned techniques of shape sensitivity analysis with the *level set method* (presented in Chapter 1). For now, let us just mention its general idea, which consists in enclosing all the possible shapes in a fixed, large computational domain D (e.g. a box), equipped with a fixed mesh (e.g. a Cartesian grid) - say $\mathcal{T}$ - and to describe any shape $\Omega \subset D$ from an implicit point of view, via a scalar 'level set' function $\phi : D \to \mathbb{R}$ which fulfills the following

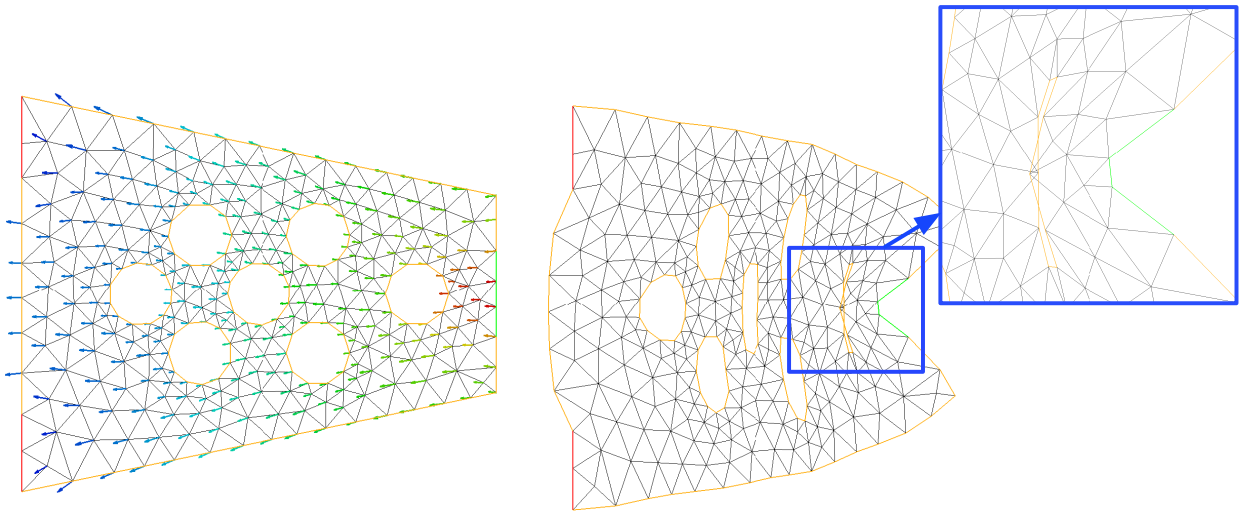


Figure 1: *(Left)* A velocity field V, defined at the vertices of a mesh; *(right)* deformed, invalid mesh obtained by translating its vertices along V.

properties (see Figure 2):

$$\forall x \in D, \begin{cases} \phi(x) < 0 & \text{if } x \in \Omega \\ \phi(x) = 0 & \text{if } x \in \partial\Omega \\ \phi(x) > 0 & \text{if } x \in {}^{c}\overline{\Omega} \end{cases}.$$

The evolution of $\Omega(t)$ along the velocity field V_Ω is reformulated in terms of an associated level set function $\phi(t,.)$ as the following *level set advection equation*:

$$\frac{\partial \phi}{\partial t} + V_\Omega \cdot \nabla \phi = 0,$$

which can be solved on D, e.g. using its mesh $\mathcal{T}$. This elegant change in perspectives allows to account for dramatic evolutions of shapes (including topological changes).

The computation of V_Ω is however not so easy in this context: we indeed evoked the fact that it requires solving PDE systems posed on Ω - a mesh of which is not available. These systems must then be approximated as PDE systems posed on the whole domain D (in the context of linearized elasticity, this is generally achieved by using the *Ersatz material approach*), and solved on the fixed mesh $\mathcal{T}$. This operation may turn out to be difficult in the study of mechanical models which require a high accuracy in the description of the boundaries of shapes (we shall see an illustration of this fact in Chapter 4).

The work of this thesis starts with the observation that a slight modification in this methodology allows us to retain its great versatility when it comes to tracking the evolution of shapes, while benefiting from an exact description of any considered shape $\Omega \subset D$.

Indeed, the use of a fixed mesh $\mathcal{T}$ of D in this procedure is essentially a commodity; each time a mesh of a shape Ω is needed, one could imagine to modify $\mathcal{T}$ in such a way that an explicit discretization of Ω appears in it (see Figure 2). Hence, the computation of the descent direction V_Ω from Ω would become straightforward, and would not involve any approximation of the considered mechanical problem. Carrying out this idea inherently requires to be able to perform local mesh operations on $\mathcal{T}$, hence to work with fully *unstructured* meshes; in our case, we shall use simplicial meshes, that is, meshes consisting of triangles in $2d$, or tetrahedra in $3d$. Moreover, it implies the following ingredients:

– A numerical method for generating a level set function for a shape $\Omega \subset D$ at the vertices of a *simplicial* mesh of D, from the datum of a mesh for Ω. This is the main goal of Chapter 6.

– A numerical method for solving the level set advection equation on a simplicial computational mesh of D. This is one of the purposes of the work in Chapter 7.
– A meshing technique for discretizing explicitly a shape Ω known via an associated level set function, on a mesh of D. This is the aim of Chapter 8.

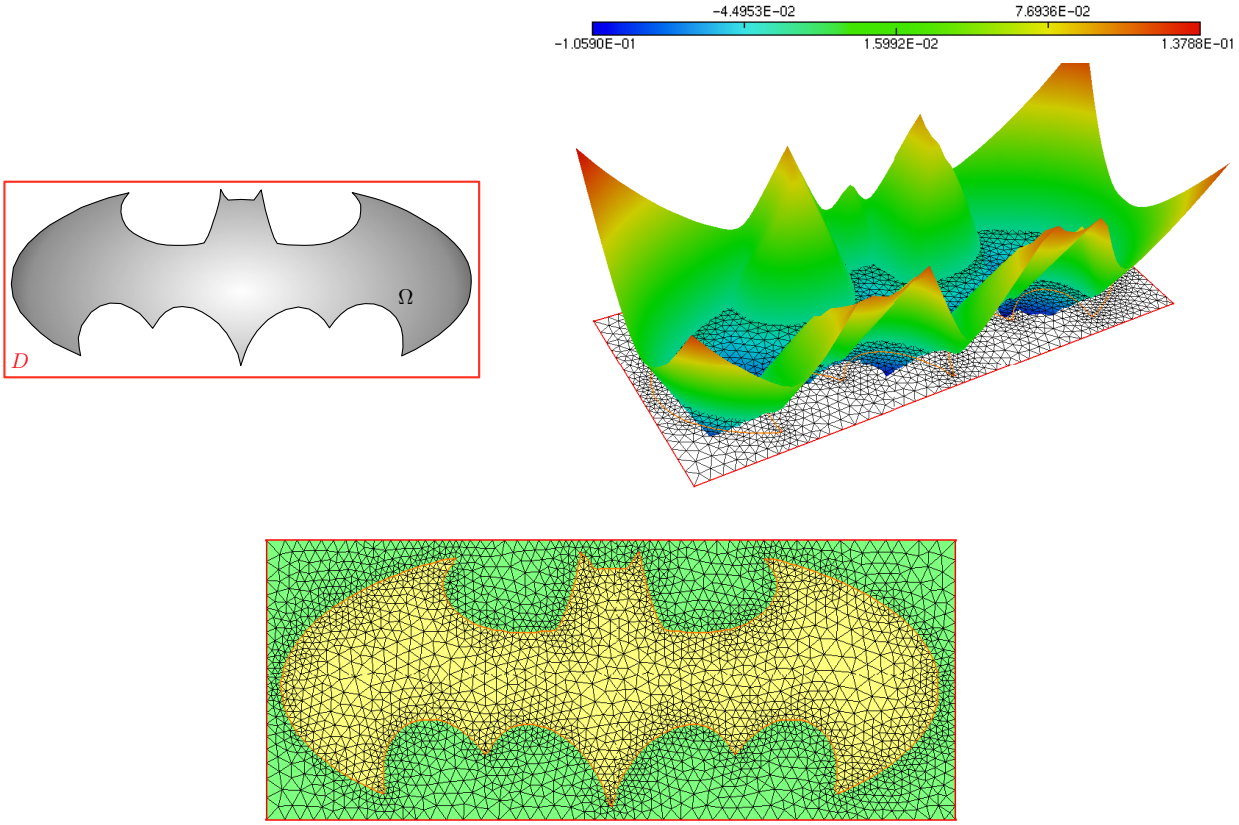


Figure 2: *(Top-left)* A domain $\Omega \subset D$, *(top-right)* graph of an associated level set function ϕ, and *(bottom)* triangular mesh of D enclosing a mesh of Ω (yellow elements).

Let us now turn to an informal description of the different parts of this work.

Part 1: Background and state of the art

The first part of this manuscript is purely bibliographical. The three main domains of the proposed work - namely level set methods, shape optimization, and meshing - are presented in three separated chapters.

Chapter 1: Level set methods

This first chapter opens with a general discussion around the notion of domain evolution. We introduce the famous *level set 'advection' equation*, which translates the motion of a domain $\Omega(t) \subset \mathbb{R}^d$ according to a velocity field $V(t,x)$ into a partial differential equation for an associated level set function $\phi(t,x)$:

$$\frac{\partial \phi}{\partial t} + V \cdot \nabla \phi = 0. \tag{1}$$

This equation rewrites as the following Hamilton-Jacobi equation, when $V = v\frac{\nabla\phi}{|\nabla\phi|}$ is oriented along the normal vector to $\Omega(t)$:

$$\frac{\partial\phi}{\partial t} + v|\nabla\phi| = 0. \tag{2}$$

We also evoke the (difficult) mathematical framework for the study of equations such as (1) or (2), trying to provide a physical intuition of the need for an adequate notion of solutions, appealing to the theory of *viscosity solutions*. These concerns lie far beyond the scope of our work, and we then turn to the numerical aspects of the level set method, two of which are discussed:

- First, we describe several numerical methods for solving equations (1)-(2). The techniques involved prove rather different depending on whether the computational support is a finite difference grid (one such numerical scheme will be used in Chapter 4), or a simplicial mesh (the work of chapter **??** is strongly influenced by the presented methods).
- The second operation of interest is the initialization (or reinitialization) of a level set function associated to a given domain Ω, which is usually achieved by computing the *signed distance function* d_Ω to Ω, defined by:

$$\forall x \in \mathbb{R}^d,\ d_\Omega(x) = \begin{cases} -d(x,\partial\Omega) & \text{if } x \in \Omega \\ 0 & \text{if } x \in \partial\Omega \\ d(x,\partial\Omega) & \text{if } x \in {}^c\overline{\Omega} \end{cases},$$

  where $d(.,\partial\Omega)$ stands for the usual Euclidean distance function to $\partial\Omega$. The most notorious methods - the *fast marching method*, the *fast sweeping method* to name a few - are presented. Although none of them shall be used in this thesis, they express deep features of the signed distance function and *Eikonal equations* which inspired to a large extent the device of the algorithm of Chapter 6.

Chapter 2: Shape optimization

After proposing a biased and non-exhaustive glimpse of the numerous applications of shape optimization techniques, we briefly describe the most commonly used methods for accounting for shapes (e.g. explicit representations, representations as density functions, etc...) and the sensitivity of functions with respect to shapes (e.g. the homogenization method, the SIMP method, Hadamard's method, etc...), emphasizing on their respective assets and drawbacks.

Pretty quickly, we focus on the framework of Hadamard's method, whereby variations of a given shape Ω of the form $(I+\theta)(\Omega)$ are considered, for 'small vector fields' θ. We recall various related notions of differentiation with respect to the shape, and notably introduce and illustrate the ideas of *shape derivative* of a scalar function $\Omega \mapsto J(\Omega) \in \mathbb{R}$, and of *material* and *Eulerian* derivatives of an application $\Omega \mapsto u_\Omega \in \mathcal{W}(\Omega)$ taking its values in a functional space $\mathcal{W}(\Omega)$ which itself depends on the shape.

Then, we narrow once again the scope of the presentation to the context of *linear elastic shapes* (which is a particular case of the general theory of distributed systems in optimal control), at stake in a great part of this manuscript. The shapes are now filled with a linear isotropic material, with Hooke's law A, and the considered objective functions $J(\Omega)$ depend on Ω via the displacement field $u_\Omega : \Omega \to \mathbb{R}^d$, solution to the linearized elasticity system:

$$\begin{cases} -\mathrm{div}(Ae(u)) = f & \text{on } \Omega \\ u = 0 & \text{on } \Gamma_D \\ Ae(u)n = g & \text{on } \Gamma_N \\ Ae(u)n = 0 & \text{on } \Gamma \end{cases}, \text{ where } \begin{cases} f \text{ are body forces applied on shapes} \\ g \text{ are surface loads applied on a subset } \Gamma_N \subset \partial\Omega \\ \Gamma_D \subset \partial\Omega \text{ is a clamping region for shapes} \\ \Gamma \subset \partial\Omega \text{ is traction-free} \end{cases}.$$

The systematic (and extremely useful in practice) *Céa's method* for differentiating such objective functions $J(\Omega)$ is introduced, which prepares the ground for Chapters 4 and 5.

Eventually, one particular numerical method for optimizing linear elastic shapes is described - namely the aforementioned *level set method*. We shall use this method as such for the numerical simulations of Chapters 4 and 5, and the mesh evolution method for shape optimization presented in Chapter 9 is heavily based on it.

Chapter 3: Mesh generation, modification and evolution

This last bibliographical chapter deals with meshing, and puts a particular emphasis on three-dimensional issues. Basic definitions and notations which we shall use throughout the subsequent chapters are recalled at first; moreover, the ubiquitous and application-dependent notion of *mesh quality* is discussed, as well as the idea of *metric-based mesh adaption* (on which we shall rely in Chapters 6 and 7). In the remainder of this chapter, three topics of utmost importance are discussed:

- The first one of them is *mesh generation*; most often, a mesh generation operation assumes the knowledge of a surface triangulation of the boundary $\partial\Omega$ of the domain Ω to be meshed. Some of the most popular methods working in this context are presented (Delaunay-based methods, advancing front methods,...). Properly speaking, we shall not use any of them in this manuscript, but we believe that an illustration of their difficulties should help in understanding why the mesh evolution method of Chapter 9 strives to avoid any mesh generation step.
  Closer to the work of this manuscript, we also present mesh generation techniques for implicitly-defined domains (e.g. the marching cubes method); this topic will find an echo in Chapters 8 and 9.
- We then discuss surface and volume *remeshing* techniques. Several methods and aspects are described in both cases; in particular, the local remeshing operators (*edge split, edge collapse, edge swap* and *vertex relocation*) are presented, as the common ingredients shared by all local remeshing strategies. We shall return to this description in Chapter 8, where they will be more extensively described, in the context of our particular application.
- Eventually, we look into the topic of *mesh deformation* (or *mesh evolution*) with respect to a user-defined displacement vector field, which is one of the main axis of this thesis; in this perspective, an overview of several existing methods is proposed, which highlights their respective assets and drawbacks.

Part 2: Two problems in shape optimization

This part is almost essentially concerned with the field of structural shape optimization, and its two chapters address altogether independent problems.

Chapter 4: Multi-phase optimization via a level set method

This chapter investigates the optimal repartition of several materials within a fixed mechanical domain. It is divided into two parts.

The first one is a long digression about the signed distance function d_Ω to a domain $\Omega \subset \mathbb{R}^d$, and its dependence on Ω. One of the main conclusions of this study concerns functionals of the domain of the form:

$$J(\Omega) = \int_D j(d_\Omega)\, dx,$$

where D is a fixed working domain, enclosing all the shapes of interest, and $j : \mathbb{R} \to \mathbb{R}$ is a smooth enough function. The shape derivative of such a function is proved to be given by the following convenient formula (see Chapter 4, Cor. 4.2 for a precise statement):

$$J'(\Omega)(\theta) = -\int_{\partial\Omega} j'(y) \left(\int_{p_{\partial\Omega}^{-1}(y)\cap D} \prod_{i=1}^{d-1} (1 + d_\Omega(s)\kappa_i(y)) ds \right) \theta(y).n(y) dy,$$

where the κ_i are the principal curvatures of $\partial\Omega$, and $p_{\partial\Omega} : \mathbb{R}^d \to \partial\Omega$ is the projection application.

The second part is the one which indeed studies the optimal repartition of two materials, with respective Hooke's law A_0, A_1, occupying respective subdomains Ω^0, Ω^1 of a fixed domain D.

The natural 'sharp-interface' model for this situation assumes a discontinuous Hooke's tensor $A_{\Omega^0} := A_0 + (1-\chi_0)(A_1 - A_0)$ over D, χ_i standing for the characteristic function of Ω^i. The displacement u_{Ω^0} of D is then solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(A_{\Omega^0} e(u)) & = & f & \text{on } D \\ u & = & 0 & \text{on } \Gamma_D \\ A_{\Omega^0} e(u) n & = & g & \text{on } \Gamma_N \end{array} \right. , \text{ where } \left\{ \begin{array}{l} f \text{ are body forces applied on shapes} \\ g \text{ are surface loads applied on a subset } \Gamma_N \subset \partial D \\ \Gamma_D \subset \partial D \text{ is a clamping region} \end{array} \right. .$$

The optimization of a functional $J(\Omega^0)$ (e.g. the compliance of the total structure D) is considered, and shape derivatives can be computed in this context. They involve in particular the jumps of the stress and strain tensors of u_Ω over the interface between Ω^0 and Ω^1, which are unfortunately inaccurately computed in a numerical context where all the computations are performed on a fixed mesh of D (i.e. in which Ω^0 is not explicitly discretized). Several possibilities are discussed to overcome this difficulty.

Next, we turn to a different modeling of the initial mechanical problem: the interface between Ω^0 and Ω^1 is 'smeared' into a thick band of uniform (small) thickness ε. The discontinuous Hooke's tensor A_{Ω^0} is then approximated by the continuous one $A_{\Omega^0,\varepsilon}$:

$$A_{\Omega^0,\varepsilon} = A_0 + h_\varepsilon(d_{\Omega^0})(A_1 - A_0),$$

where h_ε is a smooth approximation of the Heaviside function.

Using the study of the first part of this chapter allows to compute the shape derivative of the smeared approximation $J_\varepsilon(\Omega^0)$ of $J(\Omega^0)$, which lends itself to an easier numerical treatment in a fixed mesh setting (see the result of Figure 3 for a three-phase plus void test case).

Eventually, the 'smoothed-interface' problem is proved to converge to the 'sharp-interface problem' as the thickness ε of the transition zone between subdomains goes to 0, in the sense that the shape derivative $J'_\varepsilon(\Omega^0)$ converges to $J'(\Omega^0)$ for a fixed, arbitrary subdomain Ω^0.

Chapter 5: A linearized approach to worst-case design in parametric and geometric shape optimization

This chapter proposes a general framework for the optimization of linear elastic shapes in the worst-case scenario when 'small' perturbations are expected (e.g. on the loads, on the material's properties, etc...).

To set ideas, consider the following abstract situation: let $\mathcal{H}$ be a set of admissible designs characterized by $h \in \mathcal{H}$, and $(\mathcal{P}, ||.||_\mathcal{P})$ be a Banach space enclosing the 'small' potential perturbations $||\delta||_\mathcal{P} \leq m$. The *state* $u(h,\delta)$ of the shape is described by the following system:

$$\mathcal{A}(h)u(h,\delta) = b(\delta),$$

where $\mathcal{A}(h)$ is a (design-dependent) invertible operator; without loss of generality, perturbations δ only appear at the right-hand side of this system. The *cost* $\mathcal{C}(u(h,\delta))$ of the shape depends on its design h (and perturbations δ) via the state $u(h,\delta)$, and the *worst-case optimization problem* reads:

$$\min_{h\in\mathcal{H}} \mathcal{J}(h), \text{ where } \mathcal{J}(h) := \sup_{\substack{\delta\in\mathcal{P}, \\ ||\delta||_\mathcal{P}\leq m}} \mathcal{C}(u(h,\delta)).$$

As this problem is very difficult in general, we propose to take advantage of the smallness of the expected perturbations to linearize the cost function with respect to δ. This leads to the *approximated worst-case optimization problem*:

$$\min_{h\in\mathcal{H}} \widetilde{\mathcal{J}}(h), \text{ where } \widetilde{\mathcal{J}}(h) := \sup_{\substack{\delta\in\mathcal{P}, \\ ||\delta||_\mathcal{P}\leq m}} \left(\mathcal{C}(u(h,0)) + \frac{d\mathcal{C}}{du}(u(h,0))\frac{\partial u}{\partial \delta}(h,0)(\delta) \right).$$

Now, standard duality results in Banach space and techniques from optimal control theory allow to rewrite:

$$\widetilde{\mathcal{J}}(h) = \mathcal{C}(u(h,0)) + ||p(h)||_\mathcal{Q},$$

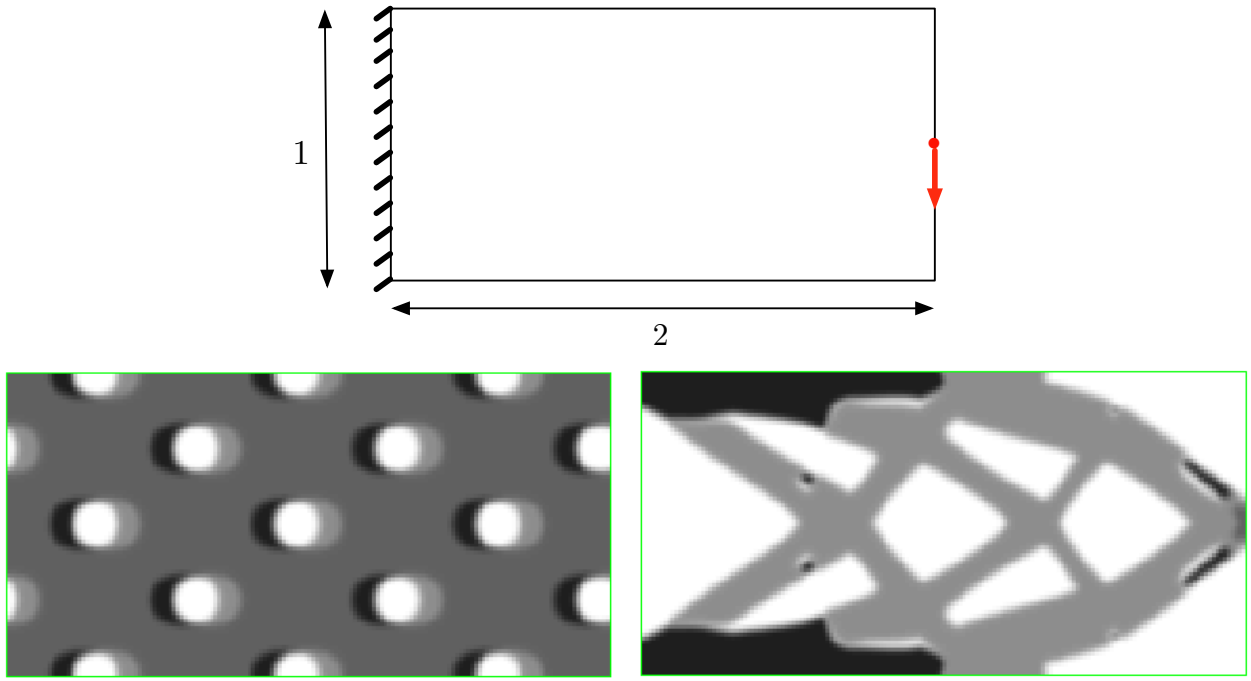


Figure 3: *(Top)* Boundary conditions of the Cantilever test-case, *(bottom-left)* initial distribution of materials within D; here the black material is 'strong', and has Young's modulus $E = 1$, the dark grey one has $E = 0.7$, the light grey one has $E = 0.5$, and the white material mimicks void: $E = 1.e^{-3}$, *(bottom-right)* optimal distribution of the three materials and void within D.

where $\mathcal{C}(u(h, 0))$ is the cost of the *unperturbed* design h, $(\mathcal{Q}, ||.||_{\mathcal{Q}})$ is the pre-dual Banach space of $\mathcal{P}$, and $p(h)$ is an adjoint state. Under this form, the minimization problem of $\widetilde{\mathcal{J}}$ can be tackled 'almost' like any standard shape optimization problem.

This methodology is applied to the theoretical and numerical studies of two usual settings in shape optimization - namely the parametric case (where $\mathcal{H}$ is typically a set of thickness functions of a plate with fixed cross-section), and the geometric shape optimization case (where $\mathcal{H}$ is a set of open and bounded domains in $\mathbb{R}^d$). In the latter case, three main sources of perturbations are considered, in the context of various cost functions, e.g. the compliance, least-square and stress-based criteria (see Figure 4):

- perturbations over the applied loads on the shapes
- perturbations over the properties of the elastic material filling the shapes
- perturbations over the geometry of the shape itself.

Part 3: Level set methods on unstructured meshes; connections with mesh adaptation

This part is almost solely concerned with level set methods; its two chapters present algorithms for initializing and advecting level set functions on a simplicial, potentially adapted computational mesh (which less usual a framework than that of finite difference grids).

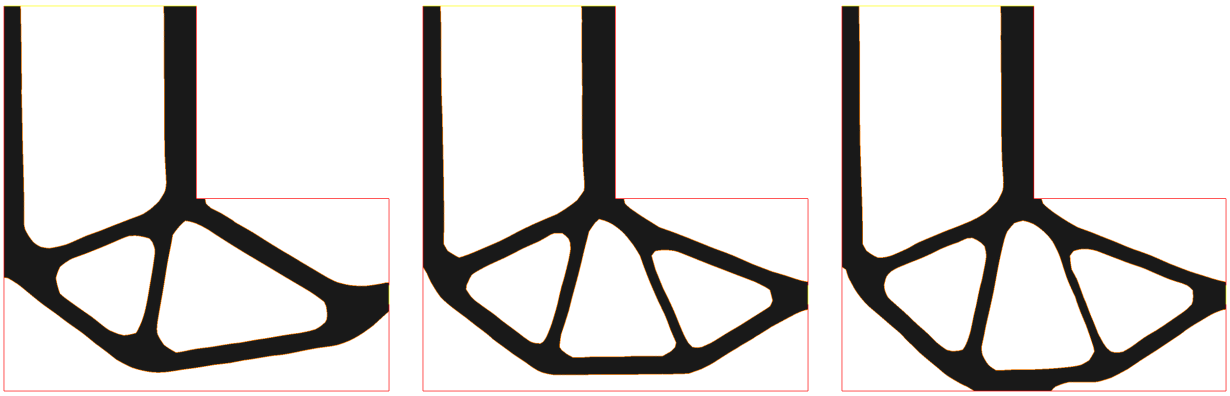


Figure 4: Optimization of an L-Beam, clamped on its upper side, subject to vertical surface loads at the middle of its right-hand side, with respect to the cost function $\mathcal{C}(\Omega) = \int_\Omega k(x)||\sigma(u_\Omega)||^5\, dx$, under perturbations over the geometry of the shape (see Chapter 5 for details). The same target volume is imposed on each shape. *(From left to right):* optimal shape for $m = 0, 0.01, 0.02$.

Chapter 6: Computation of the signed distance function to a discrete contour on adapted simplicial mesh

The purpose of this chapter is to devise and analyze a numerical method for generating the signed distance function d_Ω to a domain $\Omega \subset \mathbb{R}^d$ at the vertices of a *simplicial* mesh $\mathcal{T}$ of a computational domain D, in two and three space dimensions.

The proposed method starts with an easy step of generation of a 'very irregular' level set function ϕ_0 for Ω; ϕ_0 is then 'regularized' into d_Ω relying on the fact that d_Ω is the steady state of the *unsteady Eikonal equation*:

$$\begin{cases} \frac{\partial \phi}{\partial t} + \mathrm{sgn}(\phi_0)\,(|\nabla \phi| - 1) = 0 & \text{on } [0,\infty] \times \mathbb{R}^d \\ \phi(t=0,.) = \phi_0 & \text{on } \mathbb{R}^d \end{cases}.$$

More accurately, 'the' solution to this equation admits an explicit expression, which can be given an iterative form, an then be converted into a numerical scheme (see Figure 5 for an example).

In a second time, an adaptation process for the computational mesh $\mathcal{T}$ of D is formulated; it produces a new mesh $\widetilde{\mathcal{T}}$ of D which guarantees an enhanced approximation of the signed distance function in two ways:

- the computed approximation $d_{\widetilde{\mathcal{T}}}$ of d_Ω on $\widetilde{\mathcal{T}}$ is 'close' to d_Ω up to a user-defined tolerance,
- the piecewise affine reconstruction of Ω as the negative subdomain of $d_{\widetilde{\mathcal{T}}}$ is 'close' to Ω up to a user-defined tolerance.

Chapter 7: An accurate anisotropic adaptation method for solving the level set advection equation

In this chapter, we study the numerical resolution of the transport equation for a scalar quantity ϕ, according to a vector field $V : \mathbb{R}^d \to \mathbb{R}^d$, over a time period $[0, T]$:

$$\begin{cases} \frac{\partial \phi}{\partial t} + V \cdot \nabla \phi = 0 & \text{on } [0,T] \times \mathbb{R}^d \\ \phi(t=0,.) = \phi_0 \text{ given} & \text{on } \mathbb{R}^d \end{cases}.$$

A particular emphasis is put on the case where ϕ is a level set function for an evolving domain $\Omega(t)$ along V.

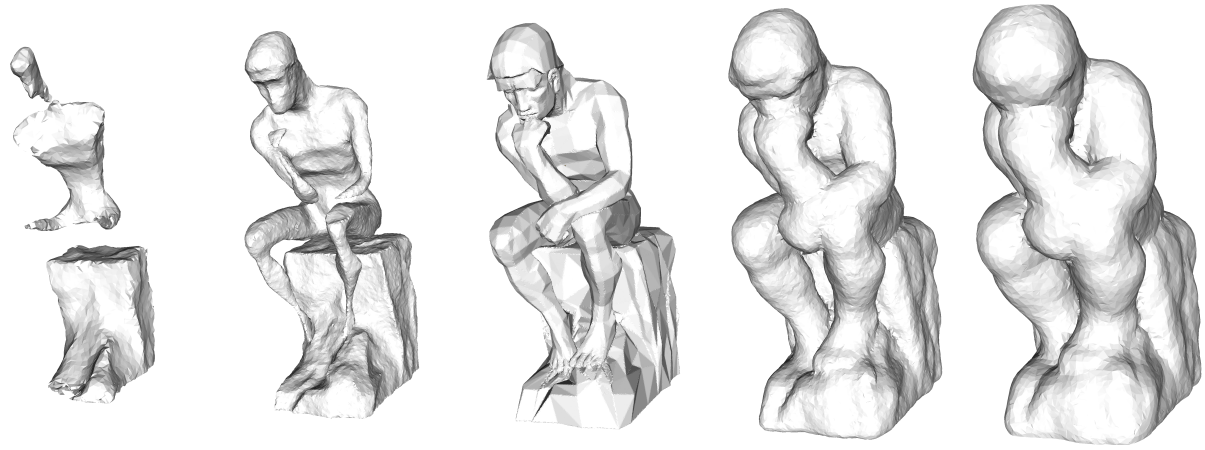


Figure 5: *(From left to right):* Isosurfaces $-0.05, -0.02, 0, 0.03, 0.05$ of the signed distance function to the (nondimensionalized) Rodin's 'Thinker' model.

We first present and analyze a numerical scheme based on the *method of characteristics*, and derive an associated a priori error estimate.

Based upon this estimate, we then devise a mesh adaptation procedure which focuses on the quality of the approximation of each intermediate domain $\Omega(t^n)$ arising in the course of the iterative process; more specifically, at each step t^n of the evolution, the computational mesh $\mathcal{T}^n$ is adapted in such a way that the piecewise affine reconstruction of $\Omega(t^n)$ as the negative subdomain of the computed approximation of the level set function $\phi(t^n, .)$ is no larger than a user-defined tolerance (see Figure 6 for an example).

Part 4 - Chapter 8: Three-dimensional surface and domain remeshing

This chapter covers all the meshing aspects of the thesis, and mainly deals with three-dimensional issues. Its contributions are threefold.

1. In a first part, the issue of (isotropic) local remeshing is addressed; the aim is to iteratively modify an initial surface triangulation $\mathcal{S}$, which may be ill-shaped, oversampled or undersampled, into a new, well-shaped and well-sampled triangulation $\widetilde{\mathcal{S}}$ which retains the geometrical features of $\mathcal{S}$. The proposed algorithm relies on four ingredients:
   - A continuous surface model Γ for $\mathcal{S}$ is created, as a set of rules for associating a local parameterization $\sigma : T \to \mathcal{S}$ of Γ to each triangle $T \in \mathcal{S}$. This model serves as a safeguard when it comes to evaluating whether a performed operation degrades the geometry expressed by $\mathcal{S}$.
   - The usual surface remeshing operators are described, with a special focus on the way they fit into our particular setting.
   - A *size map* $h : \mathcal{S} \to \mathbb{R}_+$ is defined on account of the geometrical features of $\mathcal{S}$ (notably of its curvature), and is combined with a user-defined size prescription, if any.
   - We eventually present a very heuristic strategy, yet essential in practice, to intertwine the three previous tools.
2. Still in the context of surface remeshing, we adapt the previous framework to deal with *anisotropic surface remeshing*: a size prescription is supplied by the user (or computed on account of the geometrical features of $\mathcal{S}$), and encoded as a Riemannian metric b over $\mathcal{S}$.

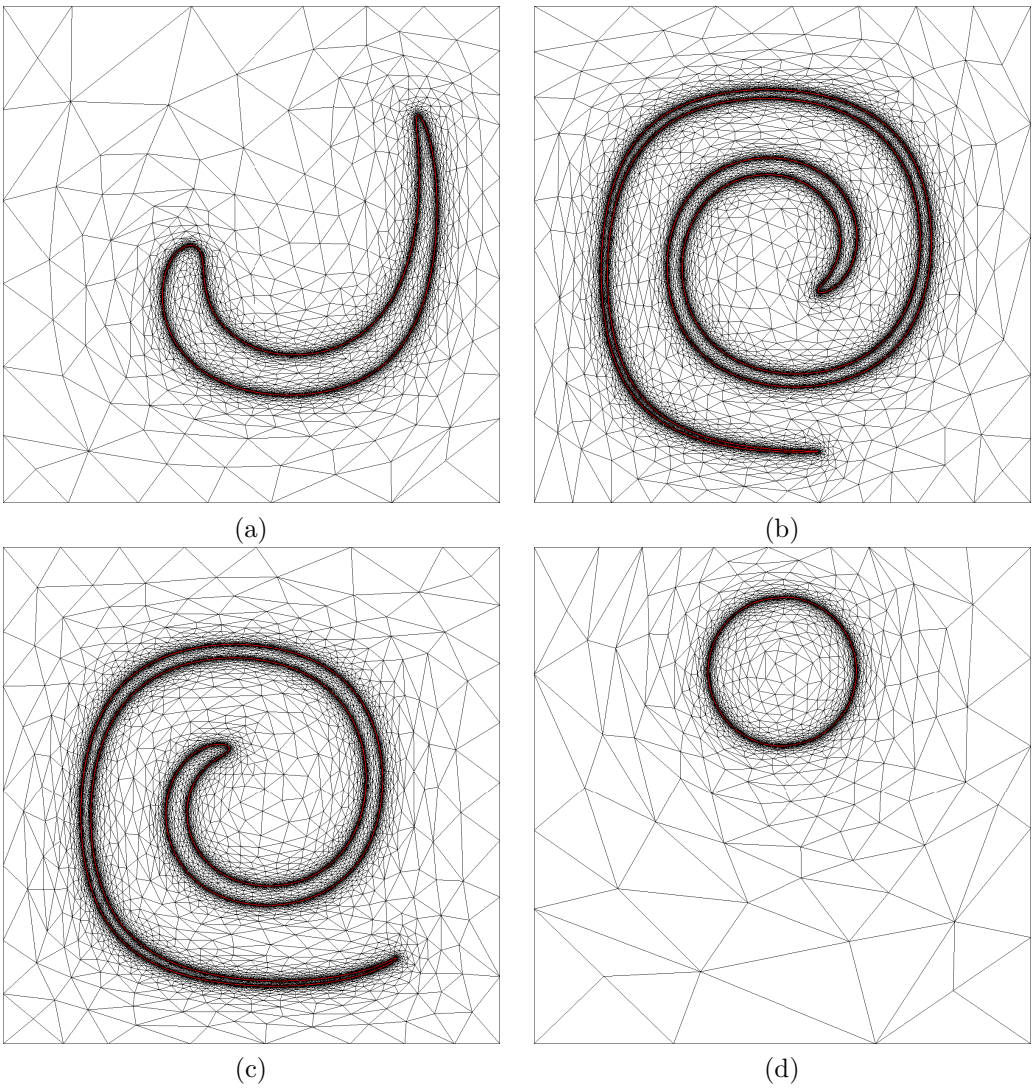


Figure 6: The *time-reversed vortex flow* example: a bubble is transported along a velocity field with high vorticity, cooked in such a way that the initial and final steps are theoretically identical (the numerical comparison between them allows to assess the accuracy of the method). Four steps of the evolution are represented: *(a)* $t = 0.8$, *(b)* $t = 4$, *(c)* $t = 5.6$ and *(d)* $t = 8$, with the corresponding 0 isolines (in red).

Making the connection between this new setting and that of the previous point mainly requires to generalize the easy handling of size maps (e.g. of the interpolation and transport operations) to the case of metric tensors. To this end, we propose to rely on the notion of *parallel transport* which can be conveniently approximated in numerical practice, owing to *Schild's ladder's algorithm*.

3. Last but not least, the issue of (isotropic) local *domain remeshing* is considered: a tetrahedral mesh $\mathcal{T}$, which may be ill-shaped, oversampled or undersampled is modified into a new, well-shaped and well-sampled mesh $\widetilde{\mathcal{T}}$, which is still a good representative of the geometry of $\mathcal{T}$, as far as their surface parts are concerned (see Figure 7 for an example). To achieve this, the very same strategy as in the first point is carried out, except that each remeshing operator now exists under two different forms, depending on whether it is applied on a surface configuration (in which case it is very similar to its

counterpart in the context of surface remeshing) or on an internal one.

Up to a slight increment, this algorithm can be converted into an algorithm for generating a computational mesh for an implicit geometry (which is the one we shall be using in Chapter 9). Indeed, the negative subdomain of a scalar function ϕ defined on a mesh of a computational domain D can be easily provided with an ill-shaped simplicial mesh $\mathcal{T}$, thanks to the use of a *marching cubes* or *marching tetrahedra* algorithm; $\mathcal{T}$ can then be modified into a well-shaped mesh $\widetilde{\mathcal{T}}$ using our algorithm.

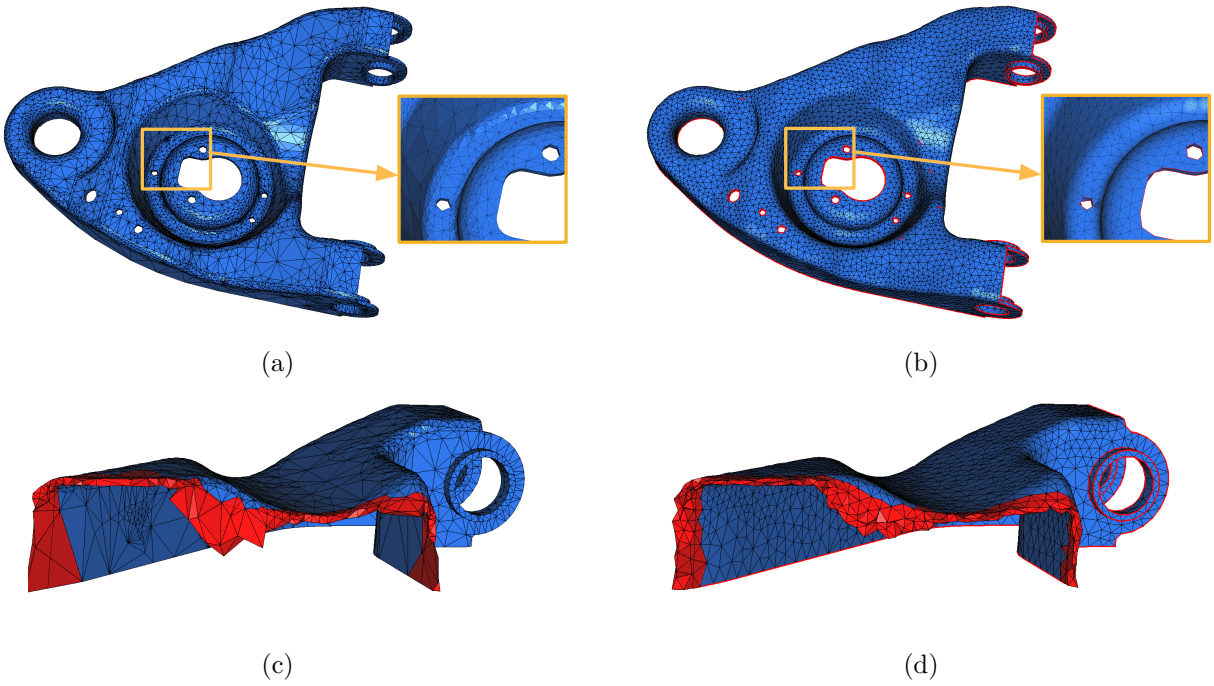


Figure 7: *(a)* Initial tetrahedral mesh of a domain, *(b)* well-shaped remeshed configuration; *(c)* a cut in the initial mesh pq; *(d)* a cut in the final mesh.

Part 5 - Chapter 9: A level-set based mesh evolution method for shape optimization

This last chapter is the one devoted to the main motivation of this thesis, which we already sketched in the preamble. So to speak, it does not introduce any additional material to that of the previous chapters, but only merges the concepts and numerical techniques of Chapters 2, 6, 7 and 8 into a general strategy for mesh evolution in the context of structural shape optimization.

This method relies on two alternative descriptions of a shape Ω: on the one hand, it is equipped with a computational mesh, a description which is very natural when mechanical analyses are considered; on the other hand, it is described via a level set function ϕ defined on (a mesh of) a larger computational domain D. As we have seen, this representation is very convenient when it comes to tracking the motion of Ω - an operation which can be carried out numerically thanks to the scheme of Chapter 7 for the advection equation.

The consistent switch between both descriptions is achieved by using the distancing algorithm of Chapter 6 for passing from a mesh description of Ω to a level set description, and the meshing algorithm of Chapter

8 for the converse operation.

Eventually, several models in shape optimization are addressed using this method, in two and three space dimensions (see Figure 8 for an illustration).

This method, together with the numerical ingredients it brings into play, have been developed in the context of the *RODIN* project (FUI AAP 13), as parts of the geometric shape optimization component of a general structural shape and topology optimization software platform.

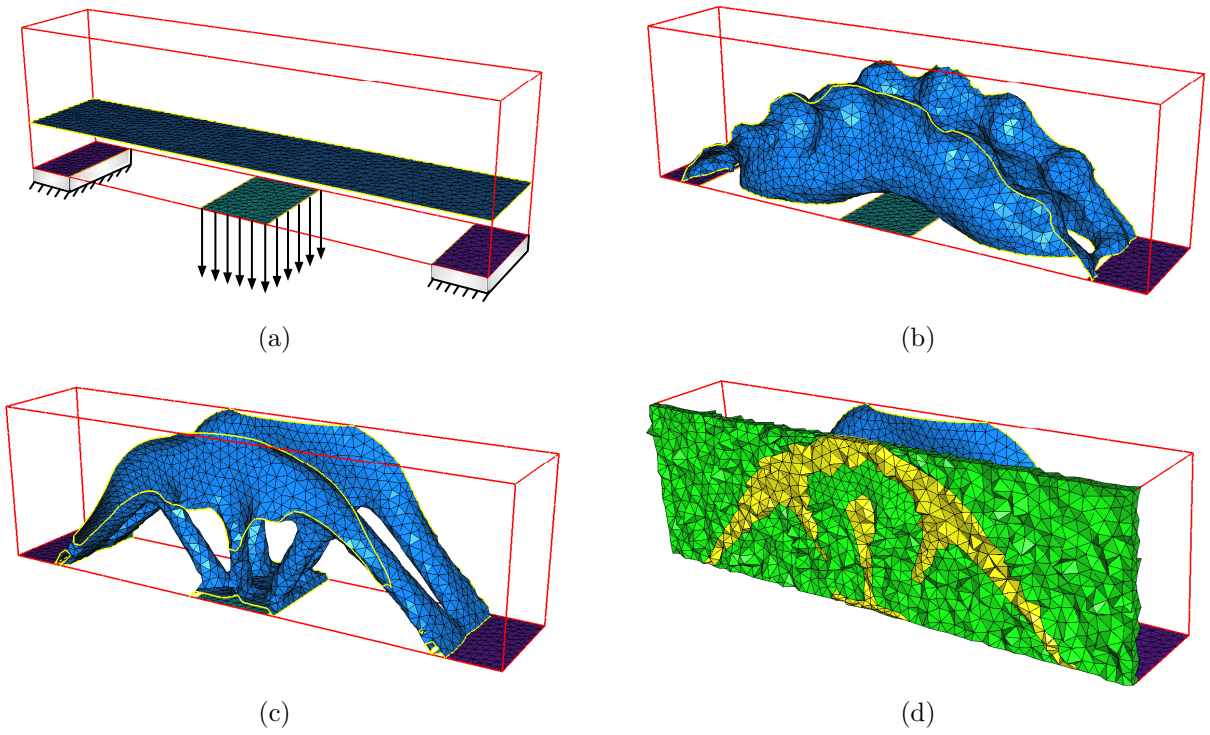


Figure 8: Shape optimization of a bridge; *(a)* initial (with boundary conditions) *(b)* 20^{th} and *(c)* final (70^{th}) steps of the algorithm; Only the implicit part of each boundary is represented. *(d)* A cut in the final mesh of the bounding box; the interior part of the shape is composed of the yellow elements.

The work of this thesis gave rise to four publications, which are listed below:

C. DAPOGNY AND P. FREY, *Computation of the signed distance function to a discrete contour on adapted triangulation*, Calcolo, Volume 49, Issue 3, pp. 193-219 (2012).

C. BUI, C. DAPOGNY AND P. FREY, *An accurate anisotropic adaptation method for solving the level set advection equation*, Int. J. Numer. Methods in Fluids, Volume 70, Issue 7, pp. 899–922 (2012).

G. ALLAIRE, C. DAPOGNY AND P. FREY, *Topology and Geometry Optimization of Elastic Structures by Exact Deformation of Simplicial Mesh*, C. R. Acad. Sci. Paris, Ser. I, vol. 349, no. 17, pp. 999-1003 (2011).

G. ALLAIRE, C. DAPOGNY AND P. FREY, *A mesh evolution algorithm based on the level set method for*

geometry and topology optimization, to appear in SMO (2013), DOI 10.1007/s00158-013-0929-2.

Two other articles have been submitted, whose titles follow:

C. DAPOGNY, C. DOBRZYNSKI AND P. FREY, *Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems*, submitted (2013).

G. ALLAIRE, C. DAPOGNY, G. DELGADO AND G. MICHAILIDIS, *Multi-phase optimization via a level set method*, submitted (2013).

Moreover, three preprints are in preparation, based upon the work in Chapter 5, 8 and 9:

G. ALLAIRE AND C. DAPOGNY, *A linearized approach to worst-case design in parametric and geometric shape optimization*, in preparation (2013).

C. DAPOGNY AND P. FREY, *A rigorous setting for anisotropic surface remeshing*, in preparation (2013).

G. ALLAIRE, C. DAPOGNY AND P. FREY, *A level-set based mesh evolution method for shape optimization*, in preparation (2013).

Eventually, the following two conference proceedings were issued from these works:

G. ALLAIRE, C. DAPOGNY AND P. FREY, *Shape optimization of elastic structures using a level-set based mesh evolution method*, Fifth International Conference on Advanced COmputational Methods in ENgineering (ACOMEN), Liège, Belgium, 2011,

G. ALLAIRE, C. DAPOGNY AND P. FREY, *A mesh evolution algorithm based on the level set method for geometry and topology optimization*, 10^{th} World Congress on Structural and Multidisciplinary Optimization (2013), Orlando, Florida, USA.

Part I

Background and state of the art

Chapter 1

The level set method

Contents

1.1 Presentation of the level set method 29
1.1.1 Implicitly-defined domains and geometry 29
1.1.2 Main notations and first examples 29
1.1.3 From an explicit to an implicit description of the evolution 30
1.1.4 Domain evolution as a boundary value problem: Eikonal equations 34
1.2 Numerical algorithms for the level set method 36
1.2.1 Solving the Level Set Hamilton-Jacobi equation on Cartesian grids 36
1.2.2 Solving the Level Set Hamilton-Jacobi equation on triangular meshes 38
1.2.3 Semi-Lagrangian schemes 39
1.3 Initializing level set functions 40
1.3.1 The fast marching method 41
1.3.1.1 The fast marching method on Cartesian grids 41
1.3.1.2 Extension of the Fast Marching Method to triangular meshes 42
1.3.2 The fast sweeping method 43
1.3.3 Re-initializing level set functions 45

Since the seminal work of Osher and Sethian [245], the level set method has been one method of choice for the description of the motion of a domain (or an interface between subdomains). The main idea is to trade the usual representation of a domain $\Omega \subset \mathbb{R}^d$ for an implicit representation, as the negative subdomain of an auxiliary scalar function ϕ defined on the whole space $\mathbb{R}^d$ (or a large computational domain in numerical practice). The function ϕ is sometimes referred to as a *level set function* for Ω. More precisely, Ω is known via a function $\phi : \mathbb{R}^d \to \mathbb{R}$ defined so that the following holds (see figure 1.1):

$$\begin{cases} \phi(x) < 0 & \text{if} \quad x \in \Omega \\ \phi(x) = 0 & \text{if} \quad x \in \partial\Omega \\ \phi(x) > 0 & \text{if} \quad x \in {}^c\overline{\Omega} \end{cases} . \tag{1.1}$$

Note that such a function always exists and can be constructed using techniques of partition of unity.

The main asset of this representation lies in that the motion of an evolving domain $\Omega(t)$ over a period of time $[0, T]$ can be translated in terms of a partial differential equation for an associated time-dependent level set function $\phi(t, .)$. This is a very convenient framework for conducting both theoretical and numerical studies.

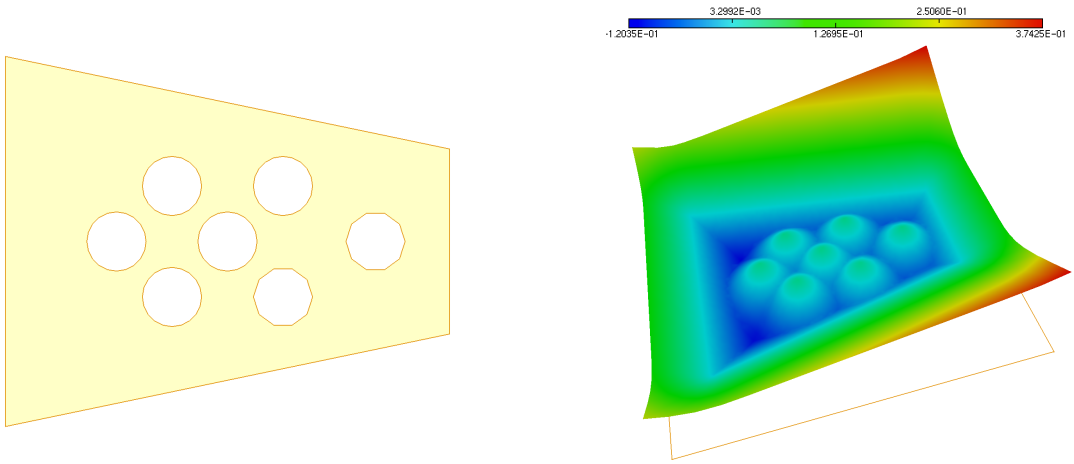


Figure 1.1: *(Left):* A subdomain Ω of $\mathbb{R}^2$; *(right):* graph of an associated level set function on a computational domain.

The level set method has given rise to particularly interesting developments in a wide variety of domains, a non exhaustive list of which follows (see the monographs [242, 274] for more examples):

- The level set method appeared in Computational Fluid Dynamics with the study of the motion of two compressible gases, separated by a sharp interface [233]. Soon after, it was used in [297] for describing the interface between two immiscible incompressible fluids, driven by the Navier-Stokes equations. Since these seminal works, it has become very popular for describing boundaries of domains filled with fluids or interfaces between them, and many improvements and extensions of the original techniques have come out (improvement of mass conservation in the incompressible case, management of more than two phases, etc...); see [277] for a more complete discussion.
- Numerous other issues from computational physics and mechanics were addressed using the level set method. For instance, a study of a solidification problem is proposed in [78], in which the interface between the solid and liquid phases in described and tracked using the level set method. The level set method was also a key ingredient in several studies around combustion [337], or geometrical optics [241]; last but not least, and closer to our concerns in this manuscript, since the seminal works [14, 278, 319], it gave rise to a framework of choice in structural optimization (see Chapter 2, §2.3 for a description of the level set method for shape optimization introduced in [14]).
- The level set method was also successfully applied to various issues in image processing. For instance, a variant of the active contour model using the level set method was introduced to tackle the problem of image segmentation in [213, 71]. In [270], the problem of image denoising using the Rudin-Osher-Fatemi model was dealt with using the level set framework. Let us eventually mention the work [132], which uses the level set method to tackle the *stereo problem*, that is the problem of reconstructing a three-dimensional scene, from the data of several two-dimensional views.
- The level set method finds very interesting applications in Computational Geometry, and Computer-Aided Design. In [193], the authors use related techniques to construct Voronoi diagrams on surfaces. In [274] (chap. 15, 19), the author discusses a grid generation technique, and a shape construction from simple primitives using the Level Set Method.

The outline of this chapter is as follows: in the first section, we discuss the derivation of the level set advection equation, which translates the motion of an evolving-in-time domain into a partial differential equation; some theoretical difficulties which naturally arise in this construction are briefly evoked. When using the level set method for the numerical description of the motion of a domain, two operations are of particular importance: the next section overviews several numerical methods for solving the partial differen-

tial equation over the level set function which accounts for the motion of a domain. One of them appears as a component of the level set method for shape optimization we shall rely on in Chapter 4. The other ones will not be used in this manuscript; however, we deem interesting to provide an overview of some of them, for they reflect many important properties of the level set evolution equations which underlie the study of chapter 7. Eventually, the last section is devoted to a question of major importance in practice, namely: how to generate a level set function associated to a given domain ? Several classical numerical methods to answer this question are presented. Here again, strictly speaking, we shall not use them, but they are very similar in spirit to the method proposed in Chapter 6.

1.1 Presentation of the level set method

1.1.1 Implicitly-defined domains and geometry

Although very different in appearance, the usual and implicit descriptions of a domain Ω are equivalent, and local geometric quantities of Ω can be expressed in terms of an associated level set function (see [329] for details).

Let $\Omega \subset \mathbb{R}^d$ a domain which is at least of class $\mathcal{C}^1$, and let $\phi : \mathbb{R}^d \to \mathbb{R}$ an associated level set function, in the sense that (1.1) holds. For any point $x \in \partial\Omega$ at which $\nabla\phi(x) \neq 0$, the unit normal vector $n(x)$ to $\partial\Omega$, pointing outward Ω, can be expressed as:

$$n(x) = \frac{\nabla\phi(x)}{|\nabla\phi(x)|}. \tag{1.2}$$

Furthermore, if Ω is of class $\mathcal{C}^2$, denote as II_x the second fundamental form (resp. $\kappa(x)$ the mean curvature) of $\partial\Omega$ at x, oriented in the sense it is positive definite (resp. positive) if $\partial\Omega$ is locally convex near x. One has:

$$II_x = \nabla\left(\frac{\nabla\phi(x)}{|\nabla\phi(x)|}\right), \text{whence } \kappa(x) = \operatorname{div}\left(\frac{\nabla\phi(x)}{|\nabla\phi(x)|}\right). \tag{1.3}$$

Other formulae exist in the same spirit for different geometric quantities (e.g. the Gaussian curvature of $\partial\Omega$, etc...), which we shall not require in the following.

1.1.2 Main notations and first examples

In this whole chapter, $\Omega(t) \subset \mathbb{R}^d$ stands for an evolving domain over a period of time $[0,T]$, and $\phi : [0,T] \times \mathbb{R}^d \ni (t,x) \mapsto \phi(t,x) \in \mathbb{R}$ is an associated level set function. The evolution of $\Omega(t)$ is assumed to be dictated by a velocity field $V : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$, which is best rewritten as:

$$\forall (t,x) \in [0,T] \times \mathbb{R}^d, \;\; V(t,x) = f(t,x,\Omega(t))$$

for a given function f, which quantifies the possible influence of the domain itself on the velocity field.

As far as f is concerned, several different behaviors may be of interest:

- f may be completely independent on the shape of the domain $\Omega(t)$. In this case, we will see that ϕ is passively transported along the velocity field V.
- f may involve local features of $\Omega(t)$, that is, for all $x \in \partial\Omega(t)$, $V(t,x)$ depends on t,x, and on local quantities of $\Omega(t)$ at x, such as the outer normal vector $n(t,x)$, to $\Omega(t)$ at x, the mean curvature $\kappa(t,x)$ of $\partial\Omega(t)$ at x, its Gaussian curvature, etc... A very important case is that of a vector field whose direction is always normal to the moving boundary, that is:

  $$V(t,x) = v(t,x)\, n(t,x), \tag{1.4}$$

  for some scalar function $v(t,x)$. In what follows, we will rely on two illustrative examples as regards such a form for V:

– The *flame propagation model*:
$$V(t,x) = c\, n(t,x), \tag{1.5}$$
where c is a constant. This models the behavior of a flame front, progressing at constant speed, along the normal direction.
– The *mean curvature flow*:
$$V(t,x) = -\kappa(t,x)\, n(t,x), \tag{1.6}$$
according to which the area of the boundary of the initial domain $\Omega(0)$ is extremalized.
– f may bring into play global features of $\Omega(t)$. For instance, if $\Omega(t)$ is to represent a domain filled with an incompressible fluid, $f(t,x,\Omega(t))$ is the velocity of the considered fluid at (t,x), solution to the Navier-Stokes equations posed on $\Omega(t)$.

The first two kinds of velocity fields may seem very restrictive in comparison with the last one. Actually, the last case is generally far too complex to study (as well in the theoretical field as in the numerical one), and approximations have to be made to bring it back to the framework of the first two. To achieve this, the most common approach consists in splitting the time interval $[0,T]$ into several (small) subintervals of the form $[t^n, t^{n+1}]$. On each subinterval (t^n, t^{n+1}), $f(t,x,\Omega(t))$ is frozen, i.e. is approximated by:

$$\forall t \in (t^n, t^{n+1}),\ \ f(t,x,\Omega(t)) \approx f(t^n, x, \Omega(t^n)).$$

In the particular case when f is directed along the normal vector to $\Omega(t)$, that is, when there exists a scalar function $g(t,x,\Omega(t))$ such that $f(t,x,\Omega(t)) = g(t,x,\Omega(t))\, n(t,x)$, f can also be approximated as:

$$\forall t \in (t^n, t^{n+1}),\ \ f(t,x,\Omega(t)) \approx g(t^n, x, \Omega(t^n))\, n(t,x).$$

The forthcoming discussions will thus rest on the first two kinds of velocity fields.

1.1.3 From an explicit to an implicit description of the evolution

Let us now focus on the understanding of domain evolution problems in the level set framework. Actually, we are about to see that the intuitive notion of an evolving domain is rather hazy in most cases. In this respect, note that we dutifully avoided any formal definition of this notion, and neglected regularity assumptions in the previous discussions.

Let us first examine a case when everything unfolds according to intuition: let $\mathcal{O} \subset [0,T] \times \mathbb{R}^d$ an open region containing $\partial\Omega(t)$ for small t, where V is well-defined and smooth, and where ϕ is smooth enough. Saying that $\Omega(t)$ smoothly evolves according to V in $\mathcal{O}$ should mean that, for all $(t_0,x_0) \in \mathcal{O}$ such that $x_0 \in \Gamma(t_0)$, there exists a curve $x(t)$, defined on some interval $(t_0-\varepsilon, t_0+\varepsilon)$, passing in x_0 at $t=t_0$, such that $(t,x(t)) \in \mathcal{O}$, and for all t, $x(t) \in \partial\Omega(t)$, with the speed vector of the curve being: $x'(t) = V(t,x(t))$ (see figure 1.2).

Since $x(t) \in \partial\Omega(t)$, one has:
$$\forall t \in (t_0-\varepsilon, t_0+\varepsilon),\ \ \phi(t,x(t)) = 0.$$

As this is true for any $(t_0,x_0) \in \mathcal{O}$, a simple use of the chain-rule yields the so-called *level set advection equation*:
$$\forall (t,x) \in \mathcal{O},\ \ \frac{\partial\phi}{\partial t}(t,x) + V(t,x).\nabla\phi(t,x) = 0. \tag{1.7}$$

As evoked in the previous section, the velocity field V often happens to be directed along the normal direction to the interface (or, more accurately to the level sets of ϕ), that is $V(t,x) = v(t,x)\frac{\nabla\phi(t,x)}{|\nabla\phi(t,x)|}$, for a certain scalar field $v(t,x)$. Equation (1.7) then rewrites as a *Hamilton-Jacobi equation*:
$$\forall (t,x) \in \mathcal{O},\ \ \frac{\partial\phi}{\partial t}(t,x) + v(t,x)|\nabla\phi(t,x)| = 0. \tag{1.8}$$

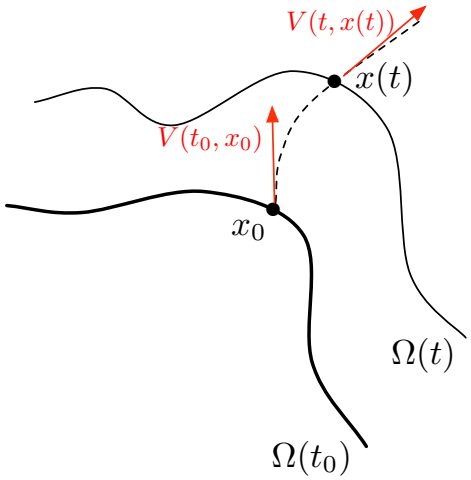


Figure 1.2: A domain $\Omega(t)$, evolving according to a velocity field $V(t, x)$.

This analysis is rather straightforward. Unfortunately, it cannot be deemed to be representative of the general case. Indeed, it has been shown that even domains evolving according to very simple vector fields V may develop singularities in finite time. In other terms, even if $\Omega(0)$ is very smooth, and so is $V(t, .)$ (or $v(t, .)$), $\Omega(t)$ is bound not to stay smooth at all times. In terms of an associated level function, this means that even if $\phi(0, .)$ is very smooth, and so is V (or v), there is no guarantee that $\phi(t, .)$ will stay smooth enough so that (1.8) makes sense everywhere. This feature is particularly expressive in the case of the two models mentioned in the previous section:

- As far as the flame propagation model is considered, in [274] (sec. 2.3) the author provides an example of a bounded domain $\Omega(0) \subset \mathbb{R}^2$ of class $\mathcal{C}^\infty$, which is evolved in the normal direction with constant unit speed (that is, V is of the form (1.5) with $c = 1$), and develops a singularity at a finite time $t = t_c > 0$. Suppose that $\partial\Omega(0)$ is locally described by the curve γ, defined as:

$$\forall s \in [0, 1], \ \gamma(s) = \left(1 - s, \frac{1 + \cos(2\pi s)}{2}\right).$$

  A simple computation gives an explicit formula for a parametrization of the corresponding boundary curves on $\partial\Omega(t)$, as long as $\Omega(t)$ stays smooth. Several of these curve are drawn on figure 1.3, and one can observe the development of a singularity in finite time. Actually, with the material of chapter 6, it will be fairly easy to see that the conclusion would have been similar, should have we considered any smooth non convex initial domain instead of this particular one.
- In the case of the mean curvature flow (1.6), suppose the evolution starts from the 'dumbbell'-like domain $\Omega(0)$, depicted on figure 1.4, left (see [90]). One can show that the domain evolves by shrinking, until its two ends join, producing a singular domain.

What happens once singularities have appeared ? Obviously, the previous way to understand domain evolution no longer holds, and several very different behaviors might be reckoned as admissible, depending on the context. For instance, as pointed out by Sethian (see section 2.3 in [274]), in the case of the flame propagation model, once the first singularity has appeared, the normal vector $n(t, x)$ to $\partial\Omega(t)$ is no longer everywhere defined. Then, (at least) both situations depicted on figure 1.5 could be considered as a potential further evolution of the considered domain.

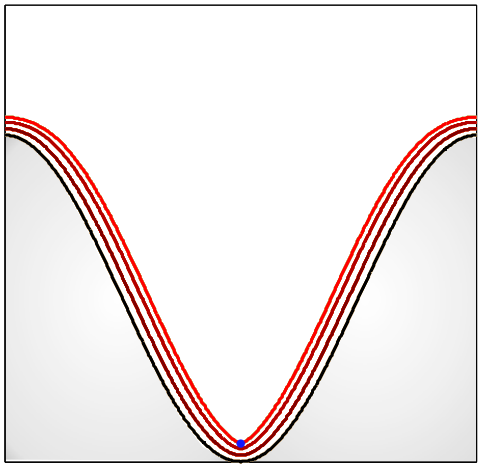


Figure 1.3: Several positions $\Omega(t)$, for $t = 0, 0.02, 0.04$, and $t = 0.055$ (from bottom to top), in the flame propagation model. $\Omega(0)$ (in grey) is of class $\mathcal{C}^\infty$; however, a singularity (blue dot) develops at a finite time, at approximately $t = t_c = 0.055$.

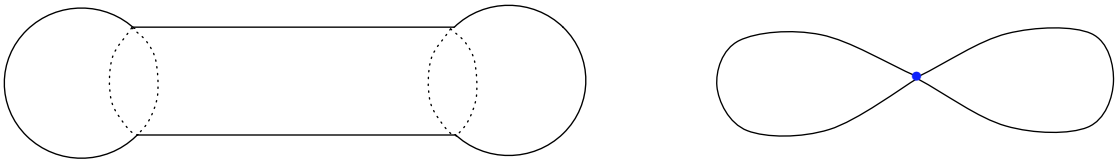


Figure 1.4: Evolution of a three-dimensional dumbbell under the mean curvature flow. The central part of the bar ends up pinching.

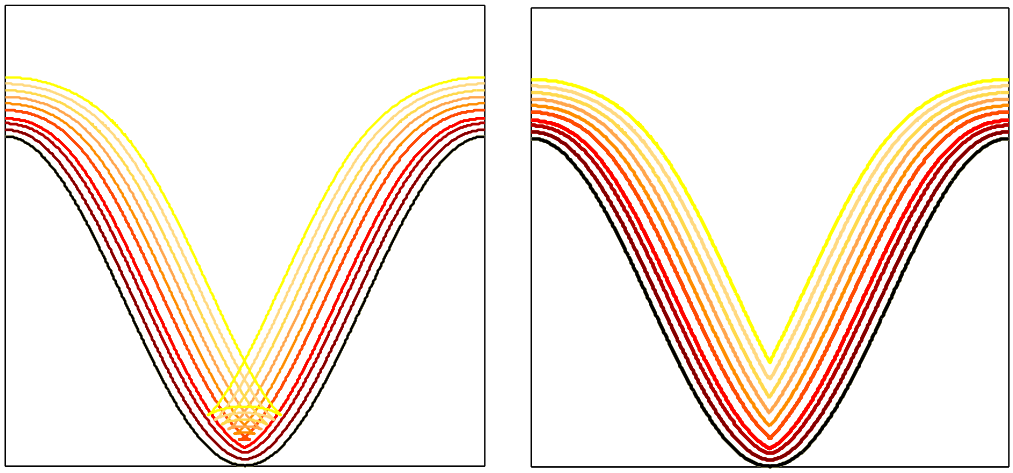


Figure 1.5: Two potential ways of pursuing evolution after the first singularity has appeared, in the example of figure 1.3: *(left)* the domains $\Omega(t)$, for $t > t_c$, obtained by pushing all the points of $\partial\Omega(t_c)$ in which a normal vector is well-defined along this normal show a 'swallowtail' pattern; *(right)* the obtained evolution by imposing monotonicity on the evolution: $\Omega(t_1) \subset \Omega(t_2)$ if $t_1 \leq t_2$.

In terms of the associated equation (1.8) over $\phi(t,x)$, this corresponds to the well-known fact that defining 'generalized solutions' of (1.7) (or (1.8)) by the fact that equality holds wherever it makes sense is not satisfactory, for it leads to too many solutions.

Actually, the way to account for such singularities is non trivial and case-dependent. Most of the time, equations (1.7,1.8) have to be understood in a weaker sense, which involves additional information about the physics of the evolution process. The physics at play is generally incorporated by means of a process to select 'good solutions' of such equations. It is then expected that under general enough assumptions, these solutions exist and are unique. This is (one of) the great achievement of the theory of *viscosity solutions* to equations of the form (1.7,1.8), initiated by P.-L. Lions and M.G. Crandall, whose definition is recalled below:

Definition 1.1. *Let $U \subset \mathbb{R}^d$ an open set and $H : \mathbb{R}^d_x \times \mathbb{R}_u \times \mathbb{R}^d_p \times \mathcal{S}(\mathbb{R}^d)$ a continuous function. Consider the following general second-order Hamilton-Jacobi equation posed on $(0,T) \times U$:*

$$\frac{\partial u}{\partial t}(t,x) + H(x,u,\nabla u, \mathcal{H}u)(t,x) = 0. \tag{1.9}$$

- *A function u is a viscosity subsolution of equation (1.9) if it is upper semicontinuous on U, and, for any function ϕ of class $\mathcal{C}^2$ on U such that $u - \phi$ reaches a local maximum at x,*

$$\frac{\partial u}{\partial t}(t,x) + H(x,u(x),\nabla\phi(x), \mathcal{H}\phi(x)) \leq 0.$$

- *A function u is a viscosity supersolution of (1.9) if it is lower semicontinuous on U, and, for any function ϕ of class $\mathcal{C}^2$ on U such that $u - \phi$ reaches a local minimum at x,*

$$\frac{\partial u}{\partial t}(t,x) + H(x,u(x),\nabla\phi(x), \mathcal{H}\phi(x)) \geq 0.$$

- *u is a viscosity solution of (1.9) if it is both a viscosity subsolution and a viscosity supersolution.*

The 'physical meaning' of such generalized solutions to Hamilton-Jacobi equations comes from that, in some cases (see [100]), they can be seen as the limit of the 'associated viscous equation' to (1.9) (i.e. the resulting equation when an artificial viscous term $-\varepsilon\Delta u$ is added), when the viscosity term vanishes. This standpoint was the original framework for showing existence of viscosity solutions to some Hamilton-Jacobi equations, and the paradigm remained attached to the theory, whereas other techniques are now involved to achieve such existence results.

According to [157], the motion of a domain according to a velocity field V is then *defined* as the negative subdomain of the (hopefully unique) viscosity solution ϕ to the associated level set advection equation:

$$\left\{ \begin{array}{ll} \dfrac{\partial \phi}{\partial t}(t,x) + V(t,x).\nabla\phi(t,x) = 0 & \text{for } (t,x) \in (0,T)\times\mathbb{R}^d \\ \phi(0,x) = \phi_0(x) & \text{for } x \in \mathbb{R}^d \end{array} \right. , \tag{1.10}$$

where ϕ_0 is a level set function associated to the initial domain.

For this approach indeed to make sense, we ought to mention the following theorem, which ensures among other things that $\Omega(t)$ is actually only dependent on $\Omega(0)$ (and not on the choice of a particular associated level set function $\phi(0,.)$):

Theorem 1.1. *Assume that either V is independent of Ω, and $V \in BUC\left([0,T]\times\mathbb{R}^d\right)^d$, or is of the form (1.4), with $v \in BUC\left([0,T]\times\mathbb{R}^d\right)$. Let $\phi_0 \in BUC\left(\mathbb{R}^d\right)$. Then, equation (1.10) admits a unique viscosity solution in $BUC\left([0,T]\times\mathbb{R}^d\right)$.*

Furthermore, let $\phi, \psi : [0,T] \times \mathbb{R}^d$ two viscosity solutions of (1.10), which are bounded and uniformly continuous over $[0,T] \times \mathbb{R}^d$, and whose associated initial negative subdomain match, that is:

$$\left\{ \begin{array}{rcl} \left\{x \in \mathbb{R}^d,\ \phi(0,x) < 0\right\} & = & \left\{x \in \mathbb{R}^d,\ \psi(0,x) < 0\right\} \\ \left\{x \in \mathbb{R}^d,\ \phi(0,x) = 0\right\} & = & \left\{x \in \mathbb{R}^d,\ \psi(0,x) = 0\right\} \\ \left\{x \in \mathbb{R}^d,\ \phi(0,x) > 0\right\} & = & \left\{x \in \mathbb{R}^d,\ \psi(0,x) > 0\right\} \end{array} \right. .$$

Assume moreover that:

$$\lim_{|x| \to \infty} |\phi(0,x)| > 0\,, \ \lim_{|x| \to \infty} |\psi(0,x)| > 0.$$

Then the associated negative subdomains of ϕ and ψ match at each time, i.e:

$$\forall t \in [0,T) \left\{ \begin{array}{rcl} \left\{x \in \mathbb{R}^d,\ \phi(t,x) < 0\right\} & = & \left\{x \in \mathbb{R}^d,\ \psi(t,x) < 0\right\} \\ \left\{x \in \mathbb{R}^d,\ \phi(t,x) = 0\right\} & = & \left\{x \in \mathbb{R}^d,\ \psi(t,x) = 0\right\} \\ \left\{x \in \mathbb{R}^d,\ \phi(t,x) > 0\right\} & = & \left\{x \in \mathbb{R}^d,\ \psi(t,x) > 0\right\} \end{array} \right. .$$

The proof of this theorem can be found in [34, 36]. Note that the exact statement goes far beyond the sole case presented here. It holds in the general context of Hamilton-Jacobi equations such as (1.9), provided the Hamiltonian function H satisfies some technical assumptions to guarantee existence and uniqueness of bounded and uniformly continuous viscosity solutions, as well as a *geometric assumption*, which implies that, roughly speaking, the level sets of ϕ evolve independently from one another.

As a conclusion to this section, let us try out the 'physical behavior' of viscosity solutions to the level set advection equation on our two favorite examples:

- In the case of the flame propagation model (1.5), studied theoretically in [34], it is shown that if $\Omega(0) \subset \mathbb{R}^d$ is an initial 'burnt' domain, and ϕ_0 is an associated continuous level set function, then there exists a unique viscosity solution ϕ to the system:

$$\left\{ \begin{array}{cc} \dfrac{\partial \phi}{\partial t} + c\,|\nabla \phi| = 0 & \text{on } [0,\infty) \times \mathbb{R}^d \\ \phi(0,.) = \phi_0 & \text{on } \mathbb{R}^d \end{array} \right. .$$

  The associated evolving domain $\Omega(t) := \left\{x \in \mathbb{R}^d,\ \phi(t,x) < 0\right\}$ happens to fulfill a so-called entropy (or monotonicity) criterion, meaning that 'a burnt point at some time stays burnt forever' (i.e. for any $t, s \geq 0$, $\Omega(t) \subset \Omega(t+s)$). The evolution of $\Omega(t)$ looks like that depicted on figure 1.5, right.
- As for the Mean Curvature Flow equation, a specific notion of viscosity solutions has to be introduced (for the Hamiltonian is not even defined at the critical points of ϕ). This work is carried out in [128], and the authors show existence and uniqueness of a solution ϕ, starting from any continuous level set function ϕ_0 as initial data. Furthermore, the associated evolving domain $\Omega(t)$ happens to be well-defined, in the sense that the conclusion of theorem 1.1 holds in that case too. This paper also proposes several simple examples - e.g. when $\Omega(0)$ is a sphere - which testify of the nice behavior of this notion of evolution in cases where intuition can be brought into play.

1.1.4 Domain evolution as a boundary value problem: Eikonal equations

An interesting particular case of the previous considerations arises when $\Omega(t)$ is assumed to *expand* according to a normal velocity function, i.e. $V(t,x)$ is of the form:

$$V(t,x) = c(x) n(t,x),$$

with $c(x) > 0$. The problem is then equivalently described by a stationary function, the *time function* $T : \mathbb{R}^d \setminus \Omega(0) \to \mathbb{R}$, defined as:

$$T(x) = \inf \left\{t \geq 0,\ x \in \Omega(t)\right\},$$

that is, for any $x \in \mathbb{R}^d \setminus \Omega(0)$, $T(x)$ is the smallest time t at which $\Omega(t)$ reaches x. The derivation of a boundary value problem for T follows the same trail as that of the level set advection equation: in a first step, it is rigorously established in regions of the space where all the quantities involved are smooth enough. Next, T is understood as the solution to this partial differential equation in an adequately generalized sense to impose a physical behavior on T at those regions where it is not smooth enough.

Let $x_0 \in \mathbb{R}^d$, and assume that, in a vicinity U of x_0 in $\mathbb{R}^d$, all the data at hand (T, etc...) are smooth. Using again the intuitive notion of an evolving domain, let $x(t), t \in (t_0 - \varepsilon, t_0 + \varepsilon)$ a curve defined such that $x(t_0) = x_0$, and, at any time, $x(t) \in \partial\Omega(t)$. By definition, one has:

$$\forall t \in (t_0 - \varepsilon, t_0 + \varepsilon), \;\; x'(t) = c(x(t))n(t, x(t)).$$

On the other hand, it follows from the very definition of T that a level set function associated to $\Omega(t)$ is

$$\forall (t, x) \in [0, T] \times \mathbb{R}^d, \;\; \phi(t, x) := T(x) - t,$$

whence $n(t, x) = \frac{\nabla T(x)}{|\nabla T(x)|}$. Differentiating with respect to t in the relation $T(x(t)) = t$ and incorporating the Dirichlet boundary condition $T(x) = 0$, for all $x \in \partial\Omega(0)$, we eventually end up with the *Eikonal equation*:

$$\left\{ \begin{array}{ll} c|\nabla T| = 1 & \text{in } \mathbb{R}^d \setminus \overline{\Omega(0)} \\ T = 0 & \text{on } \partial\Omega(0) \end{array} \right. . \tag{1.11}$$

Actually, in the sequel, we will also get interested in the very similar case when $\Omega(t)$ *shrinks* according to a normal velocity c (or $\mathbb{R}^d \setminus \overline{\Omega(t)}$ expands with velocity c):

$$V(t, x) = -c(x)\, n(t, x),$$

with $c(x) > 0$. A similar argument shows that the associated time function $T : \Omega(0) \to \mathbb{R}$ is solution to the Eikonal equation:

$$\left\{ \begin{array}{ll} c|\nabla T| = 1 & \text{in } \Omega(0) \\ T = 0 & \text{on } \partial\Omega(0) \end{array} \right. . \tag{1.12}$$

So as to select unambiguously a 'physical' behavior for T, solutions to (1.11) or (1.12) have to be taken into account in a generalized setting, which once again happens to be that of viscosity solutions. The result of interest is now the following (see [31]).

Theorem 1.2. *Assume $\Omega \subset \mathbb{R}^d$ is a bounded domain, and the normal velocity function c is positive and uniformly continuous on Ω. Then, there exists a unique viscosity solution to the Dirichlet problem (1.12).*

At this point, it is worth mentioning the very interesting discussion in [261], chap. 2, about the steady Eikonal equation, and the meaning of the 'good' viscosity solution.

Example 1.1. Let us briefly look into the interesting particular case in equation (1.12) when a unit normal velocity field, $c \equiv 1$ is considered. It then turns out that the unique viscosity solution to (1.12) is the *Euclidean distance function* $d(., \partial\Omega)$ to $\partial\Omega$, defined as:

$$\forall x \in \Omega, \;\; d(x, \partial\Omega) = \inf_{y \in \partial\Omega} |x - y|.$$

This fact translates the regular spacing out of the level sets of distance functions, as can be seen on figure 1.6. We will discuss this very important property once again in section 1.3.

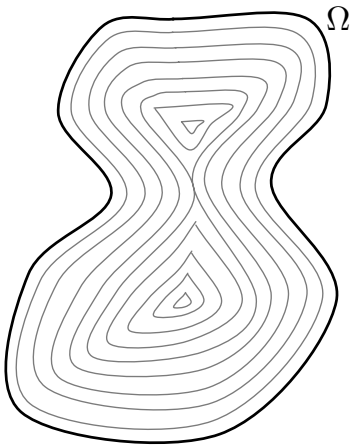


Figure 1.6: A domain Ω, together with several isolines of the distance function to $\partial\Omega$.

1.2 Numerical algorithms for the level set method

In this section, we discuss the numerical discretization of the level set advection equation, especially in the case it can be put under Hamilton-Jacobi form (which will be the case at stake in applications in shape optimization), namely:

$$\begin{cases} \dfrac{\partial \phi}{\partial t} + v|\nabla \phi| = 0 & \text{on } (0,T)\times\mathbb{R}^d \\ \phi(0,.) = \phi_0 & \text{on } \mathbb{R}^d \end{cases} \tag{1.13}$$

for given normal velocity field v, and initial function ϕ_0 which are both assumed to be bounded and uniformly continuous. Actually, the theory of numerical schemes for (1.13) is part of a more general theory associated to the numerical schemes for first order Hamilton-Jacobi equations of the form:

$$\begin{cases} \dfrac{\partial \phi}{\partial t} + H(x,\nabla \phi) = 0 & \text{on } (0,T)\times\mathbb{R}^d \\ \phi(0,.) = \phi_0 & \text{on } \mathbb{R}^d \end{cases} \tag{1.14}$$

and arises as the particular case when $H(x,p) = v(x)|p|$.

For the sake of clarity, the whole forthcoming discussion holds in the two-dimensional case ($d = 2$), without loss of generality.

1.2.1 Solving the Level Set Hamilton-Jacobi equation on Cartesian grids

Notations: Let $N \in \mathbb{N}$, and $\Delta t = \frac{T}{N}$ a time step for the discretization of the time interval $(0,T)$, into subintervals (t^n, t^{n+1}), $n = 0, ..., N-1$, with $t^n = n\Delta t$. The plane $\mathbb{R}^2$ is endowed with a Cartesian grid, with step Δx in the x-direction, and Δy in the y-direction. For a numerical quantity ϕ defined at the vertices of the grid, and for any $i, j \in \mathbb{Z}$, denote as ϕ_{ij} the value assigned to the node $x_{ij} := (i\Delta x, j\Delta y)$.

As we shall see, similarly to what happens in the case of systems of conservation laws, upwind quantities play a key role in the device of numerical schemes for Hamilton-Jacobi equations. Denote the upwind difference quantities:

$$D_{ij}^{+x}\phi = \frac{\phi_{i+1j} - \phi_{ij}}{\Delta x} \quad ; \quad D_{ij}^{-x}\phi = \frac{\phi_{ij} - \phi_{i-1j}}{\Delta x},$$

and likewise for $D_{ij}^{+y}\phi$ and $D_{ij}^{-y}\phi$.

Our aim is to compute an approximation to the viscosity solution ϕ of (1.13) as a sequence $\phi^n = \left(\phi^n_{ij}\right)_{i,j\in\mathbb{Z}}$, with the meaning that ϕ^n_{ij} is an approximation of $\phi(t^n, (i\Delta x, j\Delta y))$. A general explicit, first-order numerical scheme which fulfills this role can be written under the form:

$$\left\{ \begin{array}{ll} \forall i,j \in \mathbb{Z}, & \phi^0_{ij} = \phi_0(i\Delta x, j\Delta y) \\ \forall n \in \mathbb{N}, i,j \in \mathbb{Z}, & \phi^{n+1}_{ij} = \phi^n_{ij} - \Delta t\, \mathcal{H}\left(x_{ij}, D^{-x}_{ij}\phi^n, D^{+x}_{ij}\phi^n, D^{-y}_{ij}\phi^n, D^{+y}_{ij}\phi^n\right) \end{array} \right. , \tag{1.15}$$

where the *numerical Hamiltonian*

$$\mathcal{H}\left(x_{ij}, D^{-x}_{ij}\phi^n, D^{+x}_{ij}\phi^n, D^{-y}_{ij}\phi^n, D^{+y}_{ij}\phi^n\right)$$

is intended as an approximation of $H(x_{ij}, \nabla\phi(x_{ij}))$.

It turns out that, in the design of 'nice' numerical schemes for (1.14), two properties are desirable as far as $\mathcal{H}$ is concerned:

Definition 1.2. *An explicit, first-order scheme for (1.14) of the form (1.15) is said:*
- *consistent if, for any $x \in \mathbb{R}^2$, $p \in \mathbb{R}^2$, $\mathcal{H}(x, p_x, p_x, p_y, p_y) = H(x, p)$. In other words, the difference terms stand for the corresponding first-order derivatives where they should.*
- *monotone if, for any $x \in \mathbb{R}^2$, the function $\mathcal{H}(x, ., ., ., .)$ is increasing in the first and third arguments (i.e. those involving upwind finite differences), and decreasing in the second and fourth arguments (i.e. those involving downwind finite differences).*

It can indeed be proved that, under 'reasonable' assumptions over the theoretical Hamiltonian H and the initial data ϕ_0, consistent and monotone first order schemes are convergent to the viscosity solution of (1.14), see e.g. [98, 291].

As regards the particular Cauchy problem (1.13), the most simple approximation consists in the following explicit first-order finite difference scheme:

$$\left\{ \begin{array}{ll} \forall n \in \mathbb{N}, i,j \in \mathbb{Z}, & \phi^{n+1}_{ij} = \phi^n_{ij} - \Delta t \left(\max(v_{ij}, 0)\nabla^+_{ij}\phi^n + \min(v_{ij}, 0)\nabla^-_{ij}\phi^n\right) \\ \forall i,j \in \mathbb{Z}, & \phi^0_{ij} = \phi_0(i\Delta x, j\Delta y) \end{array} \right. , \tag{1.16}$$

where the respectively upwind and downwind discretizations $\nabla^+_{ij}\phi$ and $\nabla^-_{ij}\phi$ of $|\nabla\phi|$ are defined as:

$$\nabla^+_{ij}\phi = \left(\begin{array}{c} \max(\max(D^{-x}_{ij}\phi, 0), -\min(D^{+x}_{ij}\phi, 0))^2 \\ +\max(\max(D^{-y}_{ij}\phi, 0), -\min(D^{+y}_{ij}\phi, 0))^2 \end{array} \right)^{\frac{1}{2}} \tag{1.17}$$

and

$$\nabla^-_{ij}\phi = \left(\begin{array}{c} \max(\max(D^{+x}_{ij}\phi, 0), -\min(D^{-x}_{ij}\phi, 0))^2 \\ +\max(\max(D^{+y}_{ij}\phi, 0), -\min(D^{-y}_{ij}\phi, 0))^2 \end{array} \right)^{\frac{1}{2}} \tag{1.18}$$

The discretization of the (exact) Hamiltonian $H(x, p) = v(x)||p||$ by means of the numerical Hamiltonian

$$\forall n \in \mathbb{N}, i,j \in \mathbb{Z},\ H(x_{ij}, \nabla\phi(x_{ij})) \approx \mathcal{H}_{ij}(\{\phi^n_{kl}\}_{k,l\in\mathbb{Z}}) := \max(v_{ij}, 0)\nabla^+_{ij}\phi^n + \min(v_{ij}, 0)\nabla^-_{ij}\phi^n$$

can be deemed *upwind* in the sense that, for given i, j, n, the update $\phi^n_{ij} \to \phi^{n+1}_{ij}$ is only carried out using information
- coming from smaller values than ϕ^n_{ij} if v_{ij} is positive,
- coming from larger values than ϕ^n_{ij} if it is negative.

Once again, this is actually the key feature in the device of convergent schemes for Hamilton-Jacobi equations, which generally require consistency of the numerical approximation of the Hamiltonian with the continuous

one (to be understood in some particular sense), as well as a monotonicity assumption, which is here conveyed by the use of upwind finite differences.

To amend this, there is actually a CFL-like relation between Δt and $\Delta x, \Delta y$, which must be satisfied for this monotonicity assumption to hold. It reads:

$$\left(\sup_{i,j} v_{ij}\right) \frac{\Delta t}{\min(\Delta x, \Delta y)} \leq 1. \tag{1.19}$$

Grossly speaking, this means that the information should not travel farther than one space step within one time step. Under this CFL condition, this scheme turns out to be convergent. Moreover, an explicit error estimate with respect to the exact viscosity solution can be derived, with a residual in $\mathcal{O}\left(\sqrt{\max(\Delta x, \Delta y)}\right)$. See [98, 291] for further details regarding this very technical matter.

Unfortunately, this scheme turns out to be very diffusive, hence the need for higher order schemes. Mimicking conservation laws, high order, adaptive-stencil schemes, known as *(weighted) essentially non oscillatory* schemes (abridged as (w)ENO schemes) can be devised for solving (1.13,1.14). See [170, 246] for further details.

1.2.2 Solving the Level Set Hamilton-Jacobi equation on triangular meshes

The theory we just skimmed through in the previous subsection, about the device of convergent numerical schemes to the viscosity solutions of Hamilton-Jacobi equations of the form (1.14) can be extended to the context of triangular meshes. In [1], a convenient framework is established in this context, which notably relies on an adequate generalization of the notions of *consistency* and *monotonicity* (see definition 1.2). This allows to build convergent schemes on triangular meshes, a glimpse of which is now provided.

Notations: The plane $\mathbb{R}^2$ is endowed with a conforming triangular mesh $\mathcal{T}$. The vertices of $\mathcal{T}$ are denoted as $\{p_i\}_{i\in I}$. For any piecewise affine function ϕ on $\mathcal{T}$, denote as ϕ_i the value assigned to the node p_i.

The time interval $(0, T)$ is still split into subintervals (t^n, t^{n+1}), $n = 0, ..., N-1$, $t^n = n\Delta t$. At each time t^n, an approximation of the viscosity solution $\phi(t^n, .)$ to (1.14) is sought as a piecewise affine function ϕ^n on mesh $\mathcal{T}$. A general explicit, first-order numerical scheme for (1.14) on mesh $\mathcal{T}$ can be written as:

$$\forall i \in I, \;\; \phi_i^{n+1} = \phi_i^n - \Delta t \mathcal{H}\left(p_i, \phi^n\right), \tag{1.20}$$

where $\mathcal{H}(p_i, \phi)$ is the *numerical Hamiltonian*. This notation may seem a bit odd at first glance, since the theoretical Hamiltonian H only depends on ϕ through its gradient, and one could expect the same behavior from $\mathcal{H}\left(p_i, \phi\right)$. Actually, $\mathcal{H}\left(p_i, \phi\right)$ will depend only on $\nabla\phi$, but is better expressed in terms of ϕ.

As for the construction of a numerical Hamiltonian $\mathcal{H}(p_i, \phi)$, one may use the following procedure, which is very reminiscent of Lax-Friedrichs numerical schemes: if $h > 0$ is smaller than the smallest edge of $\mathcal{T}$, and $\varepsilon = \frac{C(H)}{2\pi}$, where $C(H)$ is one Lipschitz constant for H,

$$\forall i \in I, \;\; \mathcal{H}(p_i, \phi^n) = H(p_i, \overline{\Phi_i}) - \frac{\varepsilon}{h} \int_{\mathcal{C}_h(p_i)} \left(\phi(x) - \phi(p_i)\right) \, d\ell,$$

where $\mathcal{C}_h(p_i)$ is the circle of center p_i and radius h, and $\overline{\Phi_i}$ is the mean value of $\nabla\phi$ over the disk $\mathcal{D}_h(p_i)$ of center p_i and radius h (see figure 1.7):

$$\overline{\Phi_i} = \frac{1}{\pi h^2} \int_{\mathcal{D}_h(p_i)} \nabla\phi \, dx.$$

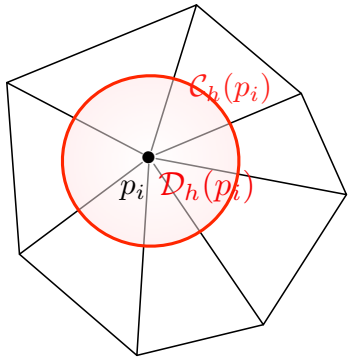


Figure 1.7: The numerical scheme of Abgrall: setting and notations.

One can show that the sequence $\{\phi^n\}_{n=0,\dots,N}$ obtained using (1.20) converges (in an appropriate sense) to the viscosity solution of (1.14), provided the following CFL condition holds:

$$\Delta t \leq \frac{h}{2\pi\varepsilon}. \tag{1.21}$$

1.2.3 Semi-Lagrangian schemes

Contrary to the aforementioned approaches, *semi-Lagrangian* techniques attempt to solve (1.13) or (1.14) by incorporating the explicit knowledge of the flow of information expressed by the equation into the device of a numerical scheme: subdividing the time interval $(0,T)$ into subintervals of the form (t^n, t^{n+1}), the value $\phi(t^{n+1},x)$ of the solution ϕ at time t^{n+1} and at a (grid) point $x \in \mathbb{R}^d$ is computed by tracking the point $y \in \mathbb{R}^d$ such that $\phi(t^{n+1},x)$ 'comes from' $\phi(t^n,y)$.

Let us illustrate this idea with a short and formal description of the semi-Lagrangian scheme of Strain [292] devoted to the level set Hamilton-Jacobi equation (1.13). This presentation is independent of the particular choice of a computational support (Cartesian grid, simplicial mesh, etc...), as are generally semi-Lagrangian methods in their broad lines.

Recall that equation (1.13) stems (at least in a formal way) from the level set advection equation:

$$\left\{ \begin{array}{ll} \frac{\partial \phi}{\partial t} + V(t,x).\nabla\phi = 0 & \text{on } (0,T)\times\mathbb{R}^d \\ \phi(0,.) = \phi_0 & \text{on } \mathbb{R}^d \end{array} \right. , \tag{1.22}$$

where the velocity field $V : (t,x) \mapsto V(t,x) \in \mathbb{R}^d$ depends itself on ϕ as:

$$V(t,x) = v(t,x)\,\frac{\nabla\phi(t,x)}{|\nabla\phi(t,x)|}.$$

The nonlinear equation (1.22) is approximated on each subinterval (t^n, t^{n+1}) by the linear advection equation obtained by freezing the value of $V(t,.)$ over (t^n, t^{n+1}), i.e. by setting:

$$\forall t \in (t^n, t^{n+1}), \forall x \in \mathbb{R}^d,\ \ V(t,x) = V(t^n,x) =: V^n(x).$$

Now assuming that an approximation ϕ^n of $\phi(t^n,.)$ has already been computed, $\phi(t,.)$ is approximated on (t^n, t^{n+1}) by the solution $\psi : (t^n, t^{n+1}) \times \mathbb{R}^d \to \mathbb{R}$ to:

$$\left\{ \begin{array}{ll} \frac{\partial \psi}{\partial t} + V^n(x).\nabla\psi = 0 & \text{on } (0,T)\times\mathbb{R}^d \\ \psi(t^n,.) = \phi^n & \text{on } \mathbb{R}^d \end{array} \right. ,$$

which is well-posed provided V^n is Lipschitz, and whose exact solution can be computed owing to the *method of characteristics*:

$$\forall x \in \mathbb{R}^d,\ \ \psi(t^{n+1}, x) = \psi(t^n, X(t^n, t^{n+1}, x)) = \phi^n(X(t^n, t^{n+1}, x)), \tag{1.23}$$

where $(t^n, t^{n+1}) \ni t \mapsto X(t, t^{n+1}, x)$ is the *characteristic curve* of V^n, reaching x at time t^{n+1}, defined as:

$$\begin{cases} X'(t, t^{n+1}, x) = V^n(X(t, t^{n+1}, x)) & \text{for } t \in (t^n, t^{n+1}) \\ X(t^{n+1}, t^{n+1}, x) = x \end{cases} \tag{1.24}$$

With this guidance in hand, the following procedure for approximating the solution ϕ to (1.13) is derived:

- **Initialization:**
  Start with an approximation ϕ^0 of $\phi(0, .)$ on the considered computational support (e.g. at the nodes of a Cartesian grid, or as a piecewise linear function on a simplicial mesh).
- **Loop (for $n = 1, ..., N-1$):**
  Loop (for each degree of freedom x of the computational support):
  1. Solve the ODE (1.24) and search for the 'foot' $y := X(t^n, t^{n+1}, x)$ of the characteristic curve reaching x at time t^{n+1} (for instance, using a Runge-Kutta 4 approximation).
  2. Mimicking formula (1.23), the value of ϕ^n at y is computed (using interpolation on the computational support) to produce $\phi^{n+1}(x)$.

The benefits of this approach are numerous: among others, it is easily parallelized since the nodes of the mesh are consistently processed independently from one another. What's more, the stability of the method does not depend on a CFL condition over the time step Δt such as (1.19,1.21). Of course, this technique also retains some drawbacks, such as the lack of accuracy in regions where the exact solution ϕ is not smooth.

This particular example brings into play the three typical stages of any general semi-Lagrangian method. An approximation $\phi^{n+1}(x)$ of $\phi(t^{n+1}, x)$ is computed using:

1. A *space-time integration step*, which amounts to searching for the point $y \in \mathbb{R}^d$ such that '$\phi(t^{n+1}, x)$ can be computed from $\phi(t^n, y)$' (here y is the foot $X(t^n, t^{n+1}, x)$ of the characteristic curve (1.24)).
2. A *spatial interpolation step*, during which $\phi^n(y)$ is interpolated from the values of ϕ^n at the grid nodes.
3. An *update step*, during which $\phi^{n+1}(x)$ is computed from $\phi^n(y)$ (in the presented case, this step is trivial, thanks to formula (1.23)).

This idea can be extended to more general Hamilton-Jacobi equations of the form (1.14). To this end, one needs a means to complete the first and third stages of the previous program. This is generally achieved thanks to representation formulae for the exact solutions to (1.14), which generalize the method of characteristics in the case of more general Hamiltonian functions H (e.g. the *Hopf-Lax formula* when H is convex and does not depend on the x variable). See [130] for more details.

1.3 Initializing level set functions

Plenty many level set functions can be associated to a domain $\Omega \subset \mathbb{R}^d$, and the theoretical framework developed hitherto is independent of which particular function ϕ is chosen. Unfortunately, things happen to be very different in numerical practice. Since the early hours of the Level Set Method [89], it has been acknowledged that too steep or too loose variations of ϕ near $\partial\Omega$ may cause instabilities in locating accurately $\partial\Omega$, or difficulties in the computation of the normal vector or curvatures of $\partial\Omega$ by means of formulae such as (1.2,1.3). This advocates for initializing a level set function for Ω as the distance function - and more precisely (for sign purposes) as the *signed distance function* d_Ω to Ω, which is defined as:

$$\forall x \in \mathbb{R}^d,\ \ d_\Omega(x) = \begin{cases} -d(x, \partial\Omega) & \text{if } x \in \Omega \\ 0 & \text{if } x \in \partial\Omega \\ d(x, \partial\Omega) & \text{if } x \in {}^c\overline{\Omega} \end{cases}$$

Anticipating a bit on the material of chapter 6, this signed distance function has furthermore the good taste of being smooth near $\partial\Omega$, provided $\partial\Omega$ is a smooth boundary.

In this section, we describe several 'classical' algorithms for generating - or restoring - the (signed) distance function to a domain Ω on a computational support. A more complete overview of the different possible methods - with a particular emphasis on the case of simplicial computational meshes - can be found in chapter 6.

1.3.1 The fast marching method

Most numerical methods for generating the signed distance function to a domain Ω allow more generally to solve Eikonal equations of the form (1.11) or (1.12). The fast marching method, originally introduced in [275], is no exception in this regard.

1.3.1.1 The fast marching method on Cartesian grids

Let us briefly present the fast marching method for generating the distance function to a bounded domain $\Omega \subset \mathbb{R}^2$, at those nodes of a Cartesian grid of the plane lying outside Ω (i.e. we solve (1.11). Extensions of this discussion to the nodes lying inside Ω and to the three-dimensional setting are rather straightforward.

Reusing the notations of section 1.2.1, the fast marching method is an iterative process which produces at each step $n \in \mathbb{N}$ a scalar numerical quantity $T := (T^n_{ij})_{i,j\in\mathbb{Z}}$ intended as an increasingly accurate approximation of $d(.,\partial\Omega)$.

It relies first on an update strategy, according to which the values of T are computed in a one-by-one, upwind fashion, mimicking the propagation of a front starting from $\partial\Omega$. More accurately, the nodes x_{ij} of the Cartesian grid are parted into three categories:

- The *accepted* nodes: those are the nodes x_{ij} 'where the front has already passed', i.e. at which the current value T_{ij} is assumed converged. Once a value has become accepted, it is no longer updated.
- The *active* nodes: those are the nodes x_{ij} 'on the front'. One of their four neighbors $x_{i-1j}, x_{i+1j}, x_{ij-1}$ or x_{ij+1} is an accepted node, and a first approximation (*trial value*) T_{ij} to the desired solution has been computed, but may still be subject to updates.
- The *far* nodes: those are the nodes 'still far from the front', whose values have not even been approximated (and are set to ∞).

At each iteration, the algorithms *accepts* one node, to be selected among the set of active nodes. The set of active nodes is then redefined, and their values are updated according to this new information about the front.

The update procedure of the value T^n_{ij} of T^n at an active node x_{ij} is the second key feature of the algorithm. A trial value $\widetilde{T^n_{ij}}$ is computed as the solution to the following equation, which relies on the upwind discretization of the Eikonal equation proposed in [269]:

$$\sqrt{\begin{array}{l}\max\left(\max\left(\frac{\widetilde{T^n_{ij}}-T^n_{i-1j}}{\Delta x},0\right),-\min\left(\frac{T^n_{i+1j}-\widetilde{T^n_{ij}}}{\Delta x},0\right)\right)^2\\ +\max\left(\max\left(\frac{\widetilde{T^n_{ij}}-T^n_{ij-1}}{\Delta y},0\right),-\min\left(\frac{T^n_{ij+1}-\widetilde{T^n_{ij}}}{\Delta y},0\right)\right)^2\end{array}} = \frac{1}{c_{ij}}. \tag{1.25}$$

Note that this rule is intrinsically *upwind*, since the derived value $\widetilde{T^n_{ij}}$ is only influenced by the values of T^n at the four neighbors of x_{ij} that are smaller than $\widetilde{T^n_{ij}}$: this is a means to impose a *causality principle* which is inherent to Hamilton-Jacobi equations. Only the *accepted* values among the set $\left\{T^n_{i-1j}, T^n_{i+1j}, T^n_{ij-1}, T^n_{ij+1}\right\}$ are used in the second order polynomial equation (1.25), and it must be checked that the obtained solution

$\widetilde{T_{ij}^n}$ is larger than those values. In the end, T_{ij}^{n+1} is obtained as:

$$T_{ij}^{n+1} = \min\left(\widetilde{T_{ij}^n}, T_{ij}^n\right).$$

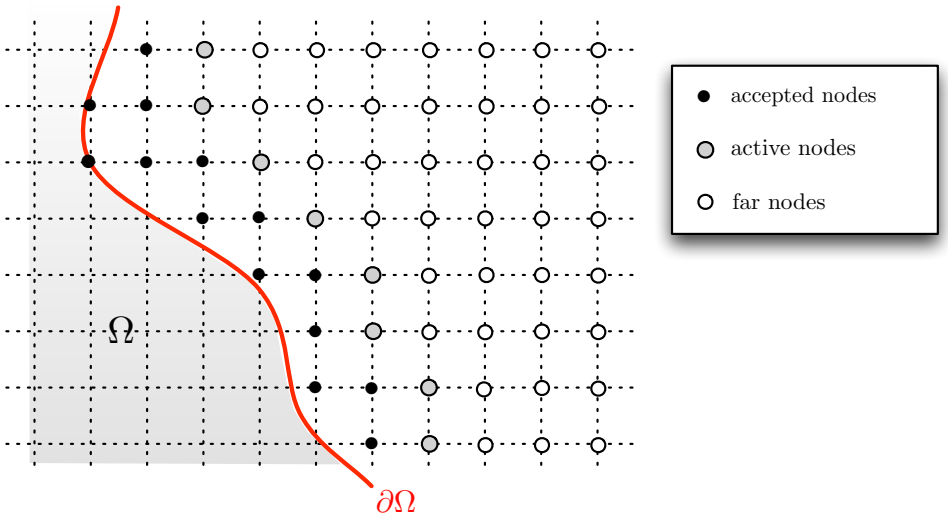


Figure 1.8: The Fast Marching Method

To sum up, the algorithm proceeds along the lines of the following sketch:

– **Initialization:**

1. Compute the exact distance function at the nodes of the cells which intersect $\partial\Omega$, and classify them as *accepted*.
2. Use the local update procedure (1.25) to compute a trial value at the neighbor points to the accepted points which have not been yet accepted, and classify them as *active*.
3. Classify all the remaining nodes as *far*, and assign them value ∞.

– **Loop (while the set of active nodes is non empty):**

1. Travel the set of active nodes, and identify the one with minimum trial value. This node becomes accepted.
2. Identify the new set of active nodes, and compute a new trial value for each one of them, using the local update solver (1.25) for the Eikonal equation.

This algorithm produces a sequence (T_{ij}^n) which converges towards the unique viscosity solution to (1.11); see [101] for a precise statement of this fact and a proof.

Furthermore, it can easily be seen that, if in practice, the computation is restrained to a large bounding domain (e.g. a box), equipped with a Cartesian mesh consisting of N vertices, the Fast Marching procedure converges within $\mathcal{O}(N\log(N))$ operations.

1.3.1.2 Extension of the Fast Marching Method to triangular meshes

Let us now give a hint of how the Fast Marching Method can be adapted to the context of a triangular mesh of $\mathbb{R}^2$, using the notations introduced in section 1.2.2, and following the work [194].

The general outline of the previous algorithm is unchanged: the vertices of the mesh are still tagged as either *accepted*, *active*, or *far*. At each iteration, the active vertex whose value is minimal becomes accepted once and for all. The set of active vertices is then adequately redefined, and the active values are updated.

This update procedure is actually the only very different feature between both versions of the Fast Marching algorithm. Here, it occurs in a situation where two values of T, say $T_i \leq T_j$ are known at two vertices p_i, p_j of a triangle $K = p_i p_j p_k \in \mathcal{T}$, and a trial value $\widetilde{T_k}$ is sought at p_k.

To this end, T is approximated by its piecewise linear interpolate $\pi_K T$ on T from the values T_i, T_j and $\widetilde{T_k}$ at p_i, p_j, p_k respectively. Let $\bar{c}$ be an approximation of c over K (e.g. $\bar{c}$ can be taken as the mean value of c over K). $\widetilde{T_k}$ should be such that:

$$|\nabla (\pi_K T)|^2 = \bar{c}^2.$$

$\widetilde{T_k}$ is then searched as a solution to this quadratic equation which is larger than T_i and T_j, and satisfies some additional properties (related to the causality inherent to equation 1.11 discussed above), which are omitted here. If such a solution $\widetilde{T_k}$ exists, the value T_k is then updated as $T_k = \min(T_k, \widetilde{T_k})$.

Let us eventually mention a potential difficulty in this approach. Depending on the shape of the triangulation $\mathcal{T}$, it may very well happen that, for a certain active point p_k, no triangle of the ball has the other two values accepted, which makes it impossible to rely on the previous local procedure to compute a new trial value $\widetilde{T_k}$ at p_k. This is especially likely to happen when the angles of triangles at p_k are obtuse. A special procedure is required in this case to come back to the previous case.

The exact same construction can be used to generate the distance function to a subset on a triangulated surface of $\mathbb{R}^3$ (this is actually the original setting of the work [194]). Yet, this procedure seems more difficult to extend to the case of a tetrahedral computational mesh of $\mathbb{R}^3$.

1.3.2 The fast sweeping method

The fast sweeping method can be thought of as a speedup of the fast marching method. If $\mathbb{R}^2$ is equipped with a Cartesian grid (and notations from section 1.2.1 are reused), the local update procedures for the computation of new 'trial' values $\widetilde{T_{ij}}$ are identical in both methods. Nevertheless, in the fast sweeping method, the choice of a particular order for enumerating the points of the grid makes it possible to enhance the computational efficiency of the method.

The fast sweeping method rests upon the following heuristics: equations (1.11, 1.12) express a transfer of information from the boundary $\partial\Omega$ of the considered domain to the outer (or inner) medium. This information is conveyed along the *characteristic curves* of the equations, originating at points of $\partial\Omega$. Hence, sweeping the plane in the directions of these characteristic curves should be enough to compute the values of T.

Let us first try out this formal idea when $c \equiv 1$. In this central case, the characteristic curves of (1.11) are straight lines, which can be distributed among four groups, depending on whether they are oriented up-left, up-right, down-left, or down-right (see figure 1.9). Then, all the values T_{ij} at nodes x_{ij} lying on characteristic curves of the up-left kind can be computed within one single iteration, by sweeping the x_{ij} in the order of decreasing i, and increasing j. A similar argument holds for the three other possible orientations. As a result, traveling successively the list of grid nodes in four different orders should allow to compute an accurate approximation of $d(., \partial\Omega)$.

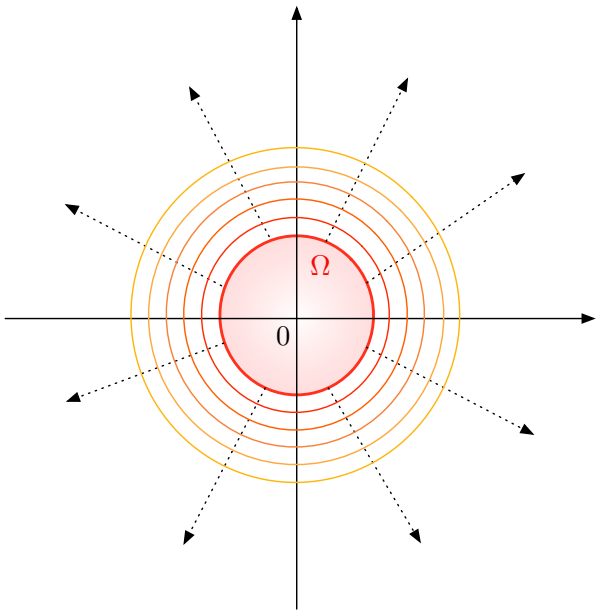


Figure 1.9: Some level sets of the distance function to a domain Ω, and characteristic lines (dotted) of equation (1.11). These lines can be grouped depending on the quarter of the plane corresponding to their directions.

The procedure reads as follows, when performed on a set of nodes $\{x_{ij}\}_{\substack{-I\leq i\leq I\\ -J\leq j\leq J}}$ of a Cartesian grid of $\mathbb{R}^2$:

– **Initialization:**

1. Compute the exact distance function at the nodes of the cells which intersect $\partial\Omega$.
2. The values T_{ij} at the remaining nodes are all set to ∞.

– 2^d **successive loops:** Travel the set $\{x_{ij}\}_{\substack{-I\leq i\leq I\\ -J\leq j\leq J}}$ as:

1. For $i = -I, ...I$, for $j = -J, ..., J$ (up-right sweep),
2. For $i = -I, ...I$, for $j = J, ..., -J$ (down-right sweep),
3. For $i = I, ... - I$, for $j = J, ..., -J$ (up-left sweep),
4. For $i = I, ... - I$, for $j = J, ..., -J$ (down-left sweep),

and, for each loop, update T_{ij} as $T_{ij} = \min\left(T_{ij}, \widetilde{T_{ij}}\right)$, where $\widetilde{T_{ij}}$ is computed owing to the local procedure (1.25).

Note that this idea is readily generalized to the case of d space dimensions, except that 2^d sweeps of the set of processed grid points are necessary.

In terms of computational cost, the fast sweeping method involves a fixed number of travels of the set of processed grid points, and it is easy to see that the fast sweeping algorithm requires about $\mathcal{O}(N)$ operations, where N is the total number of processed grid points.

Of course, when an eikonal equation of the form (1.11,1.12) is considered, with any positive function c, things are not so simple. The characteristic curves of the equation are no longer straight lines, and more than 2^d sweeps may be needed to achieve the convergence of the method (see figure 1.10).

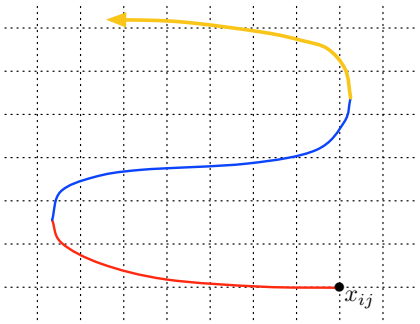


Figure 1.10: Computing an accurate approximation of the distance function to all the grid points lying on the depicted characteristic curve cannot be achieved by sweeping only once in the up-left, up-right, down-left, down-right directions: if the value T_{ij} is initialized to $d(x_{ij}, \partial\Omega)$, the values of T at the grid points lying on the red part of the curve are computed within a sweep of the plane in the up-left direction. Then, those on the blue part are computed using a sweep in the up-right direction. Finally, another sweep in the up-left direction is needed to compute the values of T lying on the yellow part.

However, theoretical and numerical evidence suggest that the fast sweeping method behaves well in a large number of cases of great importance in practice.

Let us eventually mention that this method has been adapted to the case of simplicial computational meshes, in two and three space dimensions, at the cost of higher programming effort (see [262]).

1.3.3 Re-initializing level set functions

We already pinpointed the importance in numerical practice of handling level set functions that are as close to signed distance functions as possible. Unfortunately, even if $\phi(0,.)$ is the signed distance function to $\Omega(0)$, the solution $\phi(t,.)$ to the level set advection equation (1.13) is likely to develop very sharp or loose variations in the vicinity of $\partial\Omega(t)$ as $t > 0$ increases. This may jeopardize the stability of the whole numerical resolution of (1.13).

In numerical applications, it turns out crucial to periodically restore $\phi(t,.)$ as the signed distance function to $\Omega(t)$: this is the purpose of *level set redistancing* (or *re-initialization*).

Let $\Omega \subset \mathbb{R}^d$ a bounded domain, ϕ_0 an associated level set function. Our aim is to generate the signed distance function d_Ω to Ω. The most straightforward idea in this direction consists of course in using one of the aforementioned algorithms for the computation of the (signed) distance function to a domain. Yet, the situation here is slightly different, insofar as a level set function is already available for Ω; in this context, Osher, Sussman and Smereka proposed in [297] to 'regularize' the (possibly very 'irregular') level set function ϕ_0 into a new one, which is close to d_Ω (at least near $\partial\Omega$). To this end, ϕ_0 is used as the initialization of the *redistancing equation*:

$$\left\{ \begin{array}{cl} \frac{\partial\psi}{\partial t}(t,x) + \mathrm{sgn}(\phi_0(x))\,(|\nabla\psi|-1) = 0 & \text{for } (t,x) \in [0,\infty] \times \mathbb{R}^d \\ \psi(0,x) = \phi_0(x) & \text{for } x \in \mathbb{R}^d \end{array} \right. . \tag{1.26}$$

The underlying intuition is that, as the stationary state of (1.26) is obtained, the property $|\nabla\psi| = 1$ is restored (which is formally obtained in the above equation by cancelling the time derivative). The presence of the sign function accounts for the fact that a signed distance function is sought. Of course, this very formal explanation is supported by theoretical properties which will be mentioned in chapter 6.

In practice, the redistancing equation (1.26) is solved using adequate numerical schemes, in the spirit of those presented in section 1.2, relying on a smoothing of the discontinuous function $\mathrm{sgn}(\phi_0)$.

It is eventually worth mentioning that, regardless of the method used to re-initialize $\phi(t,.)$ as $d_\Omega(t)$, this redistancing process is bound to cause a slight perturbation of the interface $\partial\Omega(t)$ handled numerically, unless $\partial\Omega(t)$ is explicitly discretized in the computational mesh.

Remark 1.1. Hitherto, all the operations we have been considering around the level set method have been carried out on a grid (or mesh) of a large computational box. This is quite a pity since, most often, one is only interested in tracking accurately the behavior of the 0 level set of ϕ, without paying much attention to what happens to the other ones. A dramatic increase in efficiency can actually be achieved by using so-called *narrow band methods*, according to which computations using the above numerical schemes are restricted to a neighborhood of the evolving interface $\partial\Omega(t)$ (see [274], chap. 7 for details)

Chapter 2

Shape optimization

Contents

2.1 A quick overview of shape optimization and applications 49
2.1.1 An overview of the main methods 49
2.1.2 Numerical difficulties in shape optimization: non existence of optimal shapes . . . 51
2.2 Shape sensitivity analysis using Hadamard's boundary variation method . . . 53
2.2.1 Hadamard's boundary variation method 53
2.2.2 Shape differentiability and computation of shape derivatives in linear elasticity . . 55
2.2.2.1 Definitions and first shape derivatives 55
2.2.2.2 Material and Eulerian derivatives 56
2.2.2.3 Linear elasticity in a nutschell 58
2.2.2.4 PDE constrained shape optimization 59
2.2.3 Shape optimization using Hadamard's method 62
2.2.3.1 The shape optimization problem 62
2.2.3.2 A generic shape optimization algorithm 63
2.2.3.3 A look at velocity extension and regularization 63
2.3 Shape optimization using the level set method 64

In utter generality, shape optimization is about the search for the 'optimal' shape Ω among a set of *admissible shapes* $\mathcal{U}_{ad}$, with respect to a prescribed criterion, assessed by means of the minimization of a *cost*, or *objective function* $J(\Omega)$:

$$\min_{\Omega \in \mathcal{U}_{ad}} J(\Omega).$$

The first examples of such problems came up a very long time ago. Among others, let us quote the famous isoperimetric problem of Dido, founder and first queen of Carthage (followed several centuries later by the very similar problem of Horatius Cocles). The Aeneid reports that, around 814 BC, Dido landed on the shores of north Africa, and asked a local Berber king, Larbas, a small portion of land to erect a city. King Larbas replied that he would grant her as much land as she could encompass with an oxhide. Dido then slit the oxhide into very thin strips, which she glued together, then used to delimit a region of maximum area, comprising a portion of the seashore, the length of the strip being fixed. Her strip of oxhide reached the shape of an arc of circle meeting the sea, which was to become the boundary of the territory of Byrsa.

Much later, in 1685, Sir Isaac Newton got interested in finding the shape of a body opposing the slightest possible resistance to the motion when immersed in a fluid. Making several drastic reductions in the problem (e.g. supposing the shapes of interest are axisymmetric), he obtained the results depicted in figure 2.1. See

[62], chap. 1, sec. 3 for a more rigorous introduction to this problem, and the website of Mark A. Peletier [251] for recent developments.

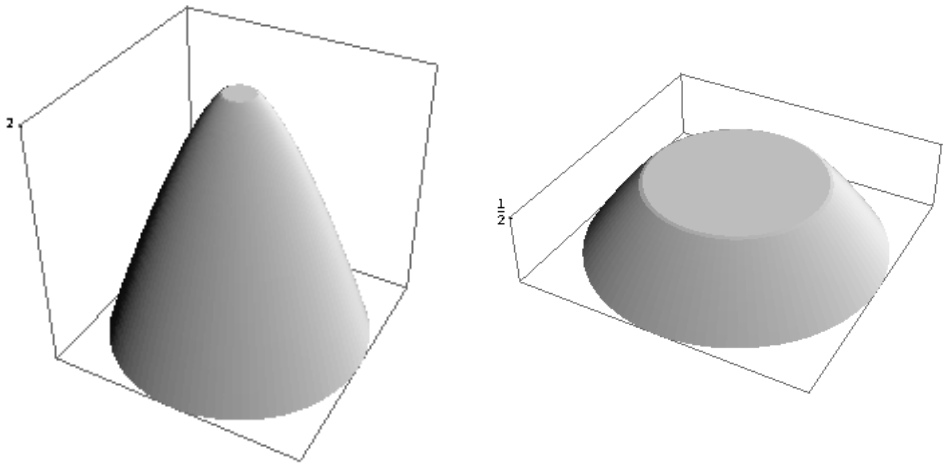


Figure 2.1: Optimal shapes of the nose of the body immersed in a fluid obtained by Newton, corresponding to two different maximal heights (reprinted from [251]).

Let us eventually evoke Lagrange's optimal column problem. In 1773, Lagrange formulated the problem of finding the shape of an axisymmetric column of prescribed volume, which guarantees maximal resistance against buckling when submitted to axial compression efforts. Using mathematical tools from the calculus of variations, he ended up with the conclusion that the cylinder was the optimal shape he was searching for. Unfortunately, he committed several mistakes in his computations, and further developments - notably those of T. Clausen - evidenced that better shapes can be achieved. See [97] for more details about this story.

Since those historical examples, shape optimization has been enjoying numerous developments, both in terms of theoretical and numerical techniques. Yet, a huge amount of issues stay unsolved (see [173]).

Altogether different problems coming from physics, mechanics, biology, etc... can be cast into the framework of shape optimization, and addressed using related techniques. To name a few,

- In [20], the problem of optimal swimming of microorganisms at low Reynolds number is considered: more specifically, the authors are in search for the shape of such microorganism that allows to reach a prescribed displacement while undergoing minimal stress from the surrounding fluid. The shape of the microorganisms is parametrized by means of a few physical parameters, and the sensitivity of the stress exerted by the fluid on the microorganisms with respect to perturbations on these parameters is computed.
- in [67], a system of N electrons with interdependent behaviors is studied. As a precious information on the chemical structure of the system, the authors are in search for the shape Ω which maximizes the criterion:
$$J_\nu(\Omega) = p_\nu(\Omega) - p_\nu^{\text{ind}}(\Omega),$$
where ν is a given integer in $[|0, N|]$, $p_\nu(\Omega)$ is the probability that ν electrons lie in Ω, and $p_\nu^{\text{ind}}(\Omega)$ is the probability that ν electrons would lie in Ω if their behaviors were independent from one another. Their work makes use of the level set method for shape optimization described in section 2.3.
- The papers [189, 71], (and a lot of subsequent ones) tackle the burning topic of image segmentation: from a grey levels image supplied as the unorganized datum of a set of pixels, each pixel being endowed with a light intensity value, one aims at identifying regions corresponding to 'physical shapes' (e.g.

faces, objects, etc...). To achieve this, the authors put the problem under the form of the minimization of an energy functional of the separating curves between different regions.
- In the article [68], an interesting application of shape optimization in the field of electromagnetism is proposed: the repartition of a series of electric wires is sought so that the induced electromagnetic field endows a nearby fluid with a target shape. This study notably relies on previous theoretical investigations around topological sensitivity analyses of Maxwell's equations [218].
- Closer to the topic of this manuscript, shape optimization techniques have been widely used to study the optimal design of bodies immersed in fluids. A famous example of such problems - which is crucial in aeronautic industry - is that of finding the shape of a wing which induces minimal drag, i.e. minimal reaction from the surrounding fluid. See [232] for many other applications of shape optimization concepts in fluid mechanics.
- Last but not least, in this manuscript we shall be mainly concerned with *structural optimization*. This part of computational mechanics which has soared over the last decades is devoted to finding the 'optimal' shape of a mechanical part, with respect to a given mechanical criterion - such as, for instance, the work of external loads, or the internal stress. These problems strongly depend on the physics at play (elasticity, thermoelasticity,...), and on the constraints that should be fulfilled by the shapes.

This chapter is organized as follows: to begin with, section 2.1 presents the high stakes of shape optimization - at least from a computational standpoint - as well as some inherent difficulties to such problems. A particular emphasis is put on the delicate issue of shape description and deformation. Section 2.2 focuses on a more technical description of one particular method - namely *Hadamard's method* - for evaluating the sensitivity of a function with respect to the domain - which implies a notion of differentiation with respect to the shape. The particular case of functions depending on the domain Ω through solutions to the linearized elasticity system posed on Ω is considered. Eventually, section 2.3 discusses a particular numerical framework, that of the *level set method*, which is suitable for an implementation of shape optimization algorithms, and shall be used repeatedly in this manuscript.

2.1 A quick overview of shape optimization and applications

2.1.1 An overview of the main methods

Since the early 60's, when shape optimization techniques were introduced in computational mechanics, many ways of handling structural optimization problems have been introduced, depending on the sought application. These methods mainly differ in the involved ways to represent shapes, and to compute the sensitivity of the objective criterion with respect to the design.

Describing shapes is a thorny problem, for a means of doing so should try and conciliate two antagonist requirements. On the one hand, as we shall see below, shape optimization techniques require to be able to perform mechanical computations on the considered shapes, e.g. by means of finite differences, finite element or finite volume methods, and not all kind of representations lend themselves to such computations. On the other hand, the representation adopted must be versatile enough to allow for a robust account of shapes' deformations.

Among the multiple possible descriptions of shapes, let us mention the following ones:
- The most straightforward (and historically the first) descriptions of shapes that have been used in the context of shape optimization are completely explicit. A surface $\mathcal{S} \subset \mathbb{R}^3$ may be supplied with a parametrization, that is, a covering set of local patches $\{(\sigma_i, U_i)\}_{i\in I}$, such that $\mathcal{S} = \bigcup_{i\in I} U_i$, and each application σ_i maps a portion of $\mathbb{R}^2$ to a portion of $\mathcal{S}$. For instance, the σ_i may be polynomial or rational applications, in which case one talks about *Bézier* or *NURBS* patches. In a similar fashion, shapes may be described by a set of physically relevant parameters (e.g. characteristic lengths, thicknesses), or by the datum of a mesh. In all these cases, shapes are explicitely accounted for by a set of *degrees of*

freedom (e.g. the control points of Bézier patches, the physical parameters, or the nodes of the mesh, depending on the case). See for instance [60, 338] around these issues, or the review article [169], and monograph [255].
- With the development of efficient methods for interface-tracking in various fields such as Computational Fluids Dynamics, implicit methods became very popular in shape optimization - the most famous of them being the *Level Set Method* and the *Phase Field Method*. When using an implicit representation of shapes, a precise description of the boundary is kept under the form of an auxiliary function defined on a large computational domain, but the shape itself is not explicitly represented.
- Eventually, a rather dramatic change in perspectives in structural optimization is embodied by the class of *density methods* in shape optimization. In a broad sense, the starting point of these methods is the works of F. Murat and L. Tartar around the homogenization method, and the optimal design of the microstructure of a material filling a fixed working domain [306], which were followed and completed by (at least) the articles [196, 212] (see also the reference textbooks [8, 41]). These theoretical contributions found an echo in [39]; influenced by these ideas, the authors devised a practical method for topology optimization by changing the very conceptual definition of a shape as an explicit black-and-white design for a 'relaxed' one (in a sense specified in section 2.1.2), as the datum of (at least) a density function $\theta : D \to [0, 1]$ defined over a computational domain D: where θ is close to 0, there is almost only void (or a very 'soft' material, mimicking void), and where θ is close to 1, there is almost only the shape. The problem of finding the 'best' shape is transformed into that of the optimal distribution of a mixture of material and void in a large computational domain. Therefore, the shape optimization problem ends rephrased as a parametric optimization problem. This led to the inception of the so-called SIMP method (see [40], and references therein).

The problem of how to perform perturbations of the considered shapes, and thus of how to compute the sensitivity of the objective function J with respect to the shape, is closely related to this problem of shapes' description. To name a few, the following design-sensitivity analysis methods have been extensively considered in the literature:

- One possibility is to perform a sensitivity analysis of the objective criterion J with respect to perturbations of the boundary of shapes. The Hadamard's and speed methods described in section 2.2.1 belong to this category.
- A completely different alternative consists in performing *topological sensitivity analyses*, according to which the sensitivity of J with respect to the nucleation of infinitesimally small holes inside shapes is evaluated (see for instance [147, 289]). Similar techniques are also widely used in the fields of imaging or inverse problems.
- When shapes are represented as density functions over a computational domain D, J depends on the shape through the values of an associated density function at the nodes of a mesh of D, and a 'classical' parameter sensitivity analysis can be performed.

These methods feature altogether different assets and drawbacks: in a nutshell, the more accurate the description of the shapes, the more accurate the computation of shape sensitivities of the given criterion, but the more acute the problem of representing and deforming them numerically.

Different methods may be coupled, to take the most from each one of them. Thus, in [16, 65], the authors proposed to combine Hadamard's method with information from topological sensitivity analysis: at each iteration, when the current shape, say Ω^n, is updated to the next one Ω^{n+1}, the boundary of Ω^n is deformed to produce that of Ω^{n+1}, according to the shape gradient produced by Hadamard's method, but without any extra ingredient, the topology of shapes are not allowed to change. Hence, from times to times, a topological sensitivity analysis is held, and small holes are nucleated in adequate regions, which are then deformed with the subsequent use of Hadamard's method. In [249], a subsequent preprocessing stage is added, in which the homogenization method is used to deliver a clever initial guess to the shape optimization process.

2.1.2 Numerical difficulties in shape optimization: non existence of optimal shapes

In the previous subsection, we discussed the possibly most obvious difficulties in shape optimization, namely the difficulty of parametrizing the problem and accounting for variations of shapes in a way that yields a satisfying notion of shape sensitivity. Another major difficulty exists on the theoretical side, which underpins many numerical difficulties in shape optimization problems, can greatly influence the choice of a particular shape representation method, and urge to resort to some numerical techniques that are not essential from a theoretical point of view, such as the regularization of shape derivatives which will be discussed hereafter. This problem is that of the non existence of an optimal shape, which is due to a *homogenization phenomenon*.

Let us start the discussion with an example, excerpted from [62] (chap. 4). Let $D \subset \mathbb{R}^d$ a bounded Lipschitz domain. Our goal is to optimize the distribution of two materials within D, one of them being thermally conductive, and the other being thermally insensitive, in such a way that the temperature in D is as close as possible to a constant c when D is heated. Rigorously speaking, one searches for the shape $\Omega \subset D$ (standing for the phase filled with the conductive material) which minimizes the following functional:

$$J(\Omega) = \int_D |u_\Omega - c|^2 \, dx,$$

where c is a constant, and u_Ω is the unique solution in $H_0^1(\Omega)$ to the system:

$$\left\{ \begin{array}{cc} -\Delta u = 1 & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{array} \right.$$

Note that, for the sake of simplicity, both the thermal conductivity of the material in Ω, and the power of the heat source have been set to 1.

The non existence result is the following.

Theorem 2.1. *For $c > 0$ small enough, no Lipschitz domain $\Omega \subset D$ can be a global minimum point of J over $\mathcal{U}_{ad}$.*

Proof. (sketch of the proof) First, it is easily proven that, provided c is small enough the whole domain D is not a global minimum point of the problem (for $J(\emptyset) = c^2$ is then smaller than $J(D)$).

Now, assume that a global minimum point Ω of J exists which is Lipschitz; as $\Omega \subsetneq D$, there exists a point $x_0 \in D \setminus \overline{\Omega}$ lying outside $\overline{\Omega}$, and let $\varepsilon > 0$ such that $d(x_0, \overline{\Omega}) > \varepsilon$ (see figure 2.2). Finally, consider the new phase $\widetilde{\Omega} = \Omega \cup B(x_0, \varepsilon)$ for the conductive material, which is nothing but the previous one, augmented with a small disconnected bubble of material. An explicit expression for $u_{\widetilde{\Omega}}$ can be computed explicitly:

$$\forall x \in \widetilde{\Omega}, \;\; u_{\widetilde{\Omega}}(x) = \left\{ \begin{array}{cc} u_\Omega(x) & \text{if } x \in \Omega \\ \frac{\varepsilon^2 - |x - x_0|^2}{2d} & \text{if } x \in B(x_0, \varepsilon) \\ 0 & \text{if } x \in D \setminus \widetilde{\Omega} \end{array} \right. .$$

Another computation shows then that $J(\widetilde{\Omega}) < J(\Omega)$ for $c > 0$ small enough, in contradiction with Ω being a minimum point for J.

□

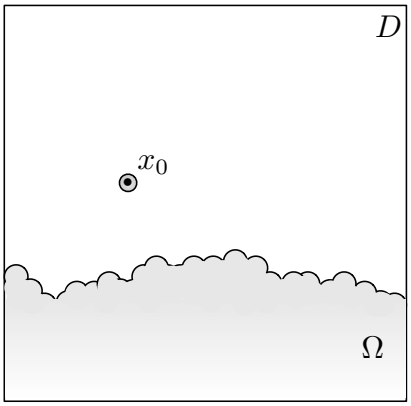


Figure 2.2: An example of non existence of an optimal shape.

How can we understand this result ? Roughly speaking, the adiabatic heating of a small amount of conductive material produces a small positive temperature, and the larger the region, the larger the resulting maximal temperature developed in the thermally conductive phase. Hence, reaching a *small* temperature c requires that the shape of the conductive phase has a large contact surface with the outer medium, and a small area; the 'optimal shape' for Ω would be an infinite collection of infinitesimally small inclusions of conductive material (which is not a Lipschitz domain).

Another interesting example of a shape optimization problem where the optimal shape tends to feature infinitely many holes (with the significant difference that holes carry homogeneous Neumann boundary conditions in this case) is discussed in [9], §6.2.1.

Generally speaking, for a wide class of shape optimization problems (even very simple ones), fractal, or porous structures are preferred over 'plain' structures: the sought optimal shape is not in the investigated class. This fact is not merely theoretical, and has severe practical implications. In particular, it explains why most shape optimization problems are generally very sensitive to the initialization - several local minima exist - or to the mesh size - the finer the computational mesh, the closer to the 'porous' optimal shape the optimization process is allowed to go.

Two main categories of techniques may help in circumventing this difficulty.

- *Relaxation of the original problem:* since the main obstruction to the existence of optimal shapes seems to be that they tend to be 'porous', one idea is to extend the set of admissible shapes so that it encloses such designs. This viewpoint can be mathematically justified owing to the *homogenization theory* (see [8, 306]), and urges to think over the problem of finding the optimal shape as that of finding the optimal distribution (and organization at the infinitesimal scale) of a mixture of material and void within a computational domain. This idea is at the root of density-based methods in structural optimization.
- *Restriction of the original problem:* the converse idea consists in imposing additional constraints on the set of admissible shapes, so as to prevent them from developing too many holes or connected components. Among the possible techniques to achieve this, let us mention the following (see [9, 62, 172] for more details):
  - *Adding a constraint on the perimeter of shapes:* the optimal shape is 'porous' or 'fractal' when the problem at stake urges shapes to maximize the area of their boundary (as in the previous example); hence, imposing an upper bound to the perimeter $P(\Omega)$ of any shape Ω should prevent this behavior. A result in this direction can be found in [21], where a model problem is considered in the thermal conductivity setting sketched above; the authors prove that trading the objective function $J(\Omega)$

for the very close function $J(\Omega) + \tau P(\Omega)$, where $\tau > 0$ is a fixed penalization parameter, leads to existence of optimal shapes in the considered class.

– *Adding a constraint on the regularity of shapes:* for instance, asking the admissible shapes in $\mathcal{U}_{ad}$ to fulfill a *uniform cone condition*, or equivalently to be *uniformly Lipschitz domains* lead to well-posed optimization problems for a wide class of considered objective functions [79].

– *Adding topological constraints on shapes:* the homogenization effect can eventually be prevented by acting directly on the topology of shapes. In this way, it was proved (by Šverak [302] in the context of the diffusion equation, and Chambolle [74] in the linear elasticity setting) proved that, in two space dimensions, imposing an upper bound on the number of connected components of the complementary $D \setminus \overline{\Omega}$ of any shape Ω in a large computational box D can lead to a well-posed optimization problem.

2.2 Shape sensitivity analysis using Hadamard's boundary variation method

Here, we detail one particular method for describing variations of a shape, namely Hadamard's boundary variation method, as well as the inferred notions of differentiation with respect to the domain.

The plan of this section is the following: to begin with, in subsection 2.2.1, we introduce the basic ideas of Hadamard's method and set some notations, that we shall use in the rest of this chapter. Then, subsection 2.2.2 presents the derived notions of differentiation with respect to a domain: the notion of shape derivative of a scalar function of the domain, or of a function which is itself defined on the domain is introduced. The end of the subsection puts the stress on a specific context - that of optimization of linear elastic shapes. Eventually, subsection 2.2.3 outlines the generic shape optimization algorithm derived from these concepts.

2.2.1 Hadamard's boundary variation method

The central idea of Hadamard's boundary variation method dates back to the seminal paper [168] (see also [290]), issued at the beginning of the 20th century. It was then exploited in depth in [234]. Let θ be a displacement field of 'small' amplitude; we consider variations of a given reference shape Ω of the form $(I + \theta)(\Omega)$ (see figure 2.3). Thus, all the considered transformations $(I + \theta)$ are homeomorphisms 'close' to the identity; in particular, all the variations of Ω achieved in this way share the same topology.

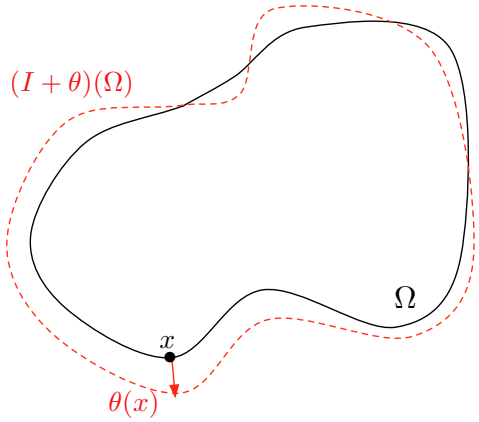


Figure 2.3: Variation $(I + \theta)(\Omega)$ of a reference shape Ω.

Let us make things slightly more rigorous. As far as the deformation vector fields are concerned, the

set of interest is the Banach space $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d) \subset L^\infty(\mathbb{R}^d)^d$ of bounded functions $\theta : \mathbb{R}^d \to \mathbb{R}^d$, whose distributional derivative $\nabla\theta$ belongs to $L^\infty(\mathbb{R}^d)^{d\times d}$, endowed with the natural norm:

$$\forall \theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d),\ \ ||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} := ||\theta||_{L^\infty(\mathbb{R}^d)^d} + ||\nabla\theta||_{L^\infty(\mathbb{R}^d)^{d\times d}}.$$

Equivalently (see section 4.2.3, th. V in [127]), it can be shown that $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ is the space of bounded and Lipschitz functions $\theta : \mathbb{R}^d \to \mathbb{R}^d$, equipped with the norm:

$$\forall \theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d),\ \ ||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} := ||\theta||_{L^\infty(\mathbb{R}^d)^d} + \sup_{\substack{x,y\in\mathbb{R}^d \\ x\neq y}} \frac{|\theta(x)-\theta(y)|}{|x-y|},$$

where $|.|$ stands for the usual Euclidean norm over $\mathbb{R}^d$.

As for the sequel, we will need the following consequence of Picard's fixed point theorem (see lemma 6.13 in [9] for a proof):

Proposition 2.1. *For every deformation field $\theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ such that $||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1$, the application $(I+\theta) : \mathbb{R}^d \to \mathbb{R}^d$ is a Lipschitz homeomorphism with Lispchitz inverse.*

More regularity over the considered deformation fields θ could be of interest; for this reason, for $k \geq 1$, we will sometimes consider the vector space $\mathcal{C}^{k,\infty}(\mathbb{R}^d,\mathbb{R}^d) = \mathcal{C}^k(\mathbb{R}^d,\mathbb{R}^d) \cap W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$, which is also a Banach space when equipped with the norm:

$$\forall \theta \in \mathcal{C}^{k,\infty}(\mathbb{R}^d,\mathbb{R}^d),\ \ ||\theta||_{\mathcal{C}^{k,\infty}(\mathbb{R}^d,\mathbb{R}^d)} := \sum_{l=0}^{k} \sup_{\substack{\alpha\in\mathbb{N}^d \\ |\alpha|=l}} \left|\left| \frac{\partial\theta}{\partial x^\alpha} \right|\right|_{L^\infty(\mathbb{R}^d)^d}.$$

Note that the exact equivalent of proposition 2.1 holds in the context of $\mathcal{C}^{k,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ spaces, with $(I+\theta)$ being a $\mathcal{C}^k$-diffeomorphism instead of a Lipschitz homeomorphism whenever $||\theta||_{\mathcal{C}^{k,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1$.

Notations: till the end of this chapter, $\Omega \subset \mathbb{R}^d$ stands for a fixed domain, which enjoys at least Lispchitz regularity. For any $\theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$, so small that $||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1$, we denote as $\Omega_\theta := (I+\theta)(\Omega)$ the deformed shape with respect to θ.

Hence, variations of a given shape Ω end up parametrized by means of an open subset of a Banach space. As we shall see in the next section, this allows among other things to introduce a notion a differentiability with respect to the shape by rewriting any operation performed on shapes close to Ω in terms of the underlying deformation field θ.

Remark 2.1. Very close in essence to Hadamard's method lies the so-called *speed method*, described in [290]. This method considers variations of a given domain $\Omega \subset \mathbb{R}^d$ by means of *flows* of vector fields, instead of vector fields themselves. More precisely, let $V \in \mathcal{C}([0,\tau], W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d))$ (as previously, V may enjoy additional regularity); for $\tau > 0$ small enough, $T_t(V) \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$, is defined as, for all $t \in [0,\tau]$, $x \in \mathbb{R}^d$, $T_t(V)(x) = X(0,t,x)$, where $t \mapsto X(0,t,x)$ is the solution to the first-order differential equation:

$$\begin{cases} X'(0,t,x) = V(t,X(0,t,x)),\ \ t \in [0,\tau] \\ X(0,0,x) = x \end{cases}.$$

Then, variations of Ω of the form $T_t(V)(\Omega)$ are considered. If we are to infringe a little bit on the forthcoming section, it is not difficult to show that both methods give rise to the same notion of differentiation of a function depending on the domain, in the direction of a given vector field. This is no longer true when it comes to higher order derivatives. However, both points of view are equivalent and formulae allow to switch from one to the other; see [106] for an interesting discussion over this topic. Note also that, in the framework of Hadamard's method, the variations Ω_θ of Ω only depend on the values taken by θ on $\partial\Omega$, whereas the variations $T_t(V)(\Omega)$ involved in the speed method depend on values of V lying outside $\partial\Omega$.

2.2.2 Shape differentiability and computation of shape derivatives in linear elasticity

2.2.2.1 Definitions and first shape derivatives

Let us start with some classical definition and results about differentiation with respect to the domain of *functionals* of the domain $\Omega \mapsto F(\Omega) \in \mathbb{R}$.

Definition 2.1. *Let $F(\Omega)$ a functional of the domain. F is shape differentiable at Ω if the underlying function*

$$\begin{array}{ccc} W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d) & \longrightarrow & \mathbb{R} \\ \theta & \longmapsto & F((I+\theta)(\Omega)) \end{array}$$

is Fréchet differentiable at $\theta = 0$. The associated Fréchet differential - denoted as $F'(\Omega)$ is called the shape derivative of F at Ω. Then, the following asymptotic expansion holds in the vicinity of 0 in $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$:

$$F(\Omega_\theta) = F(\Omega) + F'(\Omega)(\theta) + o(\theta), \quad \textit{where } \frac{o(\theta)}{||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)}} \overset{\theta\to 0}{\longrightarrow} 0. \tag{2.1}$$

Remarks 2.2.

- In the case of a functional F which also depends on other variables than Ω, the partial Fréchet differential with respect to the domain is denoted as $\frac{\partial F}{\partial \Omega}$.
- Actually, we will often require that the vector fields θ describing the variations of the shapes enjoy more regularity than that granted by the space $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ (e.g. $\mathcal{C}^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$-regularity). Anyway, in the following, with some small abuse in terminology, all the corresponding notions of differentiability shall carry the same name.

Note that the above definition is only one possible way for defining the notion of differentiation with respect to the domain: there are actually at least as many as notions of differentiability in a Banach space. In definition 2.1, Fréchet differentiability of the mapping $\theta \mapsto F((I+\theta)(\Omega))$ is required, for it turns out to be the natural setting for several functionals of the domain of interest. Although, Gâteaux differentiability, directional differentiability (to name a few) of this mapping could be considered. We shall see such examples of functionals where the latter notions are more natural ones in the following.

Theorem 2.2. *(Th. 5.2.2. and 5.4.17 in [172]) Let $\Omega \subset \mathbb{R}^d$ be a bounded Lispchitz domain.*

1. *For any function $f \in W^{1,1}(\mathbb{R}^d,\mathbb{R}^d)$, the functional $F(\Omega) = \int_\Omega f(x)\,dx$ is shape differentiable, and its derivative reads:*
$$\forall \theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d),\ F'(\Omega)(\theta) = \int_{\partial\Omega} f(x)\,\theta(x).n(x)\,ds.$$
2. *Assume that Ω is moreover of class $\mathcal{C}^2$, and let $g \in W^{2,1}(\mathbb{R}^d,\mathbb{R}^d)$. Consider the functional $G(\Omega)$ defined as $G(\Omega) = \int_\Gamma g\,ds$. Then G is shape differentiable over $\mathcal{C}^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$, and its shape derivative reads:*
$$\forall \theta \in \mathcal{C}^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d),\ G'(\Omega)(\theta) = \int_\Gamma \left(\frac{\partial g}{\partial n}(x) + \kappa(x) g(x)\right)\,\theta(x).n(x)\,ds.$$

Remark 2.3. An extension of this theorem provides a mathematical version of the so-called *Transport theorem* (or *Liouville's theorem*) in mechanics, for the differentiation of integrals of scalar quantities over evolving-in-time volumes, which is at the root of conservation laws. Indeed, if $f : [0,\tau] \ni t \mapsto f(t) \in L^1(\mathbb{R}^d)$ is differentiable at $t = 0$, $\theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$, and $f(0) \in W^{1,1}(\mathbb{R}^d)$, then $I(t) := \int_{\Omega_{t\theta}} f(t)\,dx$ is derivable at $t = 0$, and (see [172], th. 5.2.2):

$$I'(0) = \int_\Omega f'(0)\,dx + \int_{\partial\Omega} f(0)\,\theta\cdot n\,ds.$$

More involved examples are the so-called *state-constrained systems*, or *distributed systems*, which involve quantities depending on Ω through the solutions of PDE posed on Ω. Before getting into specifics, let us mention the remarkable *Structure Theorem*, which gives, in utter generality, precise information about the form of shape derivatives (see [172], prop. 5.9.1, or [105], Th 9.3.6).

Theorem 2.3. *Let $k \geq 1$, and $\Omega \subset \mathbb{R}^d$ an open, (possibly unbounded) measurable set, with topological boundary Γ. Let*

$$F : \left\{\Omega \subset \mathbb{R}^d \text{ open and measurable}\right\} \longrightarrow \mathbb{R}$$

a functional of the domain such that the underlying mapping $\mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d) \ni \theta \mapsto F((I+\theta)(\Omega))$ is Fréchet differentiable at 0, with shape derivative $F'(\Omega)$. Then,

1. *The restriction of $F'(\Omega)$ to $\mathcal{D}(\mathbb{R}^d, \mathbb{R}^d)$ is a (vector-valued) distribution of order at most $-k$ over $\mathbb{R}^d$, whose support is included in Γ.*
2. *If Ω is an open domain with $\mathcal{C}^1$ boundary, with outer unit normal vector field n, then for any $\theta \in \mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ such that $\theta.n = 0$ on Γ, one has:*

$$F'(\Omega)(\theta) = 0.$$

This result accounts for two very intuitive and important facts: point (1) states that the functional F is insensitive to perturbations of the domain that do not affect its boundary. Point (2) is a bit more subtle, meaning that if the data enjoy enough smoothness, and if θ is a deformation field which only acts on Γ by convection, then at first order, the domain (and any 'reasonable' function of it) stays unchanged.

2.2.2.2 Material and Eulerian derivatives

We are now interested in the differentiation with respect to the domain of functions of the domain $u : \Omega \mapsto u(\Omega) \in \mathcal{W}(\Omega)$ which are themselves defined on the domain - $\mathcal{W}(\Omega)$ being a Banach space depending on Ω. So to set ideas, we follow here the presentation in [290], where the considered domains Ω are bounded domains of class $\mathcal{C}^k$, $k \geq 1$, deformations θ belong to $\mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, and $\mathcal{W}(\Omega)$ is one of the Sobolev spaces $W^{m,p}(\Omega)$ ($m \in [0, k[$, $p \in [1, \infty]$). As in the previous section, the forthcoming notions of *material* and *Eulerian derivatives* may be defined in several slightly different frameworks (e.g. that of functions $u(\Omega)$ belonging to $\mathcal{C}^k(\Omega)$, or a space which does not depend on Ω). When the context is clear, we shall however give to all these notions the same name.

Let then u be an application which, to any domain Ω of class $\mathcal{C}^k$ associates a function $u(\Omega) \in W^{m,p}(\Omega)$. So as to give a precise meaning to the 'derivative of u with respect to Ω', the natural idea is to fix $x \in \Omega$, and look at the derivative of $\Omega \mapsto u_\Omega(x) \in \mathbb{R}$ in the sense of definition 2.1. This makes sense if $x \in \Omega$, for then, for $\theta \in \mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ small enough, $x \in \Omega_\theta$. Yet, we may need to take the derivative of the behavior of $u(\Omega)$ at the boundary $\partial\Omega$, and the previous argument no longer holds for a point $x \in \partial\Omega$. The mathematically convenient point of view consists in transporting all functions $u(\Omega_\theta)$ back to the reference domain Ω, considering $u(\Omega_\theta) \circ (I+\theta) \in W^{m,p}(\Omega)$, then derivating with respect to θ. Only then will the notion of 'pointwise derivative' be inferred.

Definition 2.2. *Function $u : \Omega \mapsto u(\Omega)$ admits a material (or Lagrangian) derivative $\dot{u}(\Omega)$ at a given domain Ω provided the transported function*

$$\mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d) \ni \theta \longmapsto u(\Omega_\theta) \circ (I+\theta) \in W^{m,p}(\Omega),$$

which is defined in the neighborhood of $0 \in \mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, is differentiable at $\theta = 0$.

The derivative $u'(\Omega)(\theta)(x)$ of u with respect to the domain, at a fixed point $x \in \Omega$, is now *defined* as what we would expect from it, using formally the chain-rule:

$$\begin{array}{rcl} \frac{d}{d\theta}\left(u(\Omega_\theta)(x+\theta(x))\right)\big|_{\theta=0} & = & \frac{d}{d\theta}\left(u(\Omega_\theta)(x)\right)\big|_{\theta=0} + \frac{d}{d\theta}\left(u(\Omega)(x+\theta(x))\right)\big|_{\theta=0} \\ \dot{u}(\Omega)(\theta)(x) & = & u'(\Omega)(\theta)(x) + \nabla u(\Omega)(x).\theta(x) \end{array}$$

Definition 2.3. *Function $u : \Omega \mapsto u(\Omega)$ admits a Eulerian derivative $u'(\Omega)(\theta)$ at a given domain Ω in the direction $\theta \in \mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ if it admits a material derivative $\dot{u}(\Omega)(\theta)$ at Ω in direction θ, and $\nabla u(\Omega).\theta \in W^{m,p}(\Omega)$. One defines then:*

$$u'(\Omega)(\theta) = \dot{u}(\Omega)(\theta) - \nabla u(\Omega).\theta \in W^{m,p}(\Omega). \tag{2.2}$$

These notions naturally extend to the case of functions u which, to any (part of the) boundary Γ of a domain of class $\mathcal{C}^k$, associate a function $u(\Gamma) \in W^{m,p}(\Gamma)$ ($m \in [0, k[$, $p \in [1, \infty]$):

Definition 2.4. *Let $u : \Gamma \mapsto u(\Gamma) \in W^{m,p}(\Gamma)$ a function,*

- *u admits a material (or Lagrangian) derivative $\dot{u}(\Gamma)$ at Γ if the transported function*

$$\mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d) \ni \theta \longmapsto u(\Omega_\theta) \circ (I + \theta) \in W^{m,p}(\Gamma)$$

  is differentiable at $\theta = 0$
- *If u admits a material derivative $\dot{u}(\Gamma)$ at Γ, and $\nabla_\Gamma u(\Gamma).\theta \in W^{m,p}(\Gamma)$, then u admits a Eulerian derivative $u'(\Gamma)(\theta)$ in direction $\theta \in \mathcal{C}^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, defined as:*

$$u'(\Gamma)(\theta) = \dot{u}(\Gamma)(\theta) - \nabla_\Gamma u(\Gamma).\theta \in W^{m,p}(\Omega) \tag{2.3}$$

Note that definition (2.3) of the Eulerian derivative in this last case only differs from (2.2) in that the tangential gradient appears in the second definition - which is fortunate since $u(\Gamma)(x)$ does not make sense for points x outside Γ. To emphasize the connection between both formulae, let Γ_0 the boundary of a domain Ω_0 of class $\mathcal{C}^k$ ($k \geq 2$), V a fixed open neighborhood of Γ_0 in $\mathbb{R}^d$, and define a function $\Omega \mapsto \widetilde{u}(\Omega) \in W^{m,p}(V)$ as the unique extension of $u(\partial\Omega)$ to V which is constant along the normal direction to $\partial\Omega$, i.e. $\nabla\widetilde{u}(\Omega) \cdot n = 0$ on V. It can then be shown that the Eulerian derivative of $u(\Gamma)$ at Γ_0 (in the sense of definition 2.4) coincides with the Eulerian derivative of $\widetilde{u}(\Omega)$ at Ω_0 (in the sense of definition 2.3).

Let us illustrate these notions with an example - taken from [234] - that will come in handy in the sequel.

Example 2.1. *Material and Eulerian derivatives of the unit normal vector field.* For any bounded domain Ω of class $\mathcal{C}^1$, let $n_\Omega : \partial\Omega \to \mathbb{S}^1$ be the unit normal vector field, pointing outwards Ω. We consider variations $\theta \in \mathcal{C}^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$. As expected, strictly speaking, the Eulerian derivative of n_Ω has no precise meaning since it is only defined on the (varying) boundary $\partial\Omega$. However, its Lagrangian derivative does. Indeed, for θ small enough, the transported normal vector reads:

$$\forall y \in \partial\Omega, \;\; n_{\Omega_\theta}((I + \theta)(y)) = \frac{\mathrm{com}\,(I + \nabla\theta(y)) \cdot n_\Omega(y)}{|\mathrm{com}\,(I + \nabla\theta(y)) \cdot n_\Omega(y)|},$$

where, for any $d \times d$ matrix $M \in \mathcal{M}_d(\mathbb{R})$, $\mathrm{com}(M)$ stands for the matrix of cofactors of M. Using the matrix identity

$${}^t(I + \nabla\theta(y))\,\mathrm{com}\,(I + \nabla\theta(y)) = \det\,(I + \nabla\theta(y))\,I,$$

differentiating at $\theta = 0$ (which makes sense because all the terms are polynomials in θ), we get:

$$\frac{d}{d\theta}(\mathrm{com}(I + \nabla\theta(y))_{|_{\theta=0}} = \mathrm{div}(\theta)(y)\,I - {}^t\nabla\theta(y).$$

Eventually, the function $\Omega \mapsto n_{\partial\Omega} \in \mathcal{C}(\partial\Omega)$ admits a Lagrangian derivative $\dot{n_{\partial\Omega}}$ whose expression reads, for all $\theta \in \mathcal{C}^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, $y \in \partial\Omega$:

$$\begin{array}{rcl} \dot{n_\Omega}(\theta)(y) & = & \mathrm{div}(\theta)(y)\,n_\Omega(y) - {}^t\nabla\theta(y)n_\Omega(y) - (\mathrm{div}(\theta)(y) - ({}^t\nabla\theta(y)n_\Omega(y)).n_\Omega(y))n_\Omega(y). \\ & = & ({}^t\nabla\theta(y)n_\Omega(y)).n_\Omega(y))n_\Omega(y) - {}^t\nabla\theta(y)n_\Omega(y). \end{array}$$

Thus, the *Eulerian* derivative $n'_\Omega \in \mathcal{C}(\partial\Omega)$ of n_Ω at $y \in \partial\Omega$ is defined as:

$$\begin{array}{rcl} n'_\Omega(\theta)(y) & = & ({}^t\nabla\theta(y)n_\Omega(y)).n_\Omega(y))n_\Omega(y) - {}^t\nabla\theta(y)n_\Omega(y) - \nabla_{\partial\Omega}n_\Omega(y).\theta(y) \\ & = & -\nabla_{\partial\Omega}(\theta.n_\Omega)(y) \end{array}$$

We now turn to shape optimization in the context of a particular mechanical system, that of linearized elasticity.

2.2.2.3 Linear elasticity in a nutschell

Let $\Omega \subset \mathbb{R}^d$ be a bounded domain filled with a homogeneous, isotropic, elastic material. This assumption means that deformations of Ω as a result of an external stress are instantaneous, and that Ω instantaneously returns to its equilibrium state as soon as the stress ceases.

The motion of such a domain Ω is described in terms of the *deformation function* $\varphi : \overline{\Omega} \to \mathbb{R}^d$, with the meaning that each point $x \in \overline{\Omega}$ is moved to the position $\varphi(x)$. Equivalently, one may consider the corresponding *displacement function* $u = \varphi - I$.

So as to measure the induced internal distortion (or *strain*) within Ω, the *Cauchy-Green strain tensor* $C(\varphi)$ and the *Green-Saint-Venant strain tensor* $E(\varphi)$ of the motion are respectively defined as:

$$C(\varphi) = {}^t\nabla\varphi.\nabla\varphi = I + {}^t\nabla u + \nabla u + {}^t\nabla u.\nabla u \ , \ \ E(\varphi) = \frac{1}{2}(C(\varphi) - I).$$

Grossly speaking, $C(\varphi)$ is a measure of the deformation of a curve drawn on Ω entailed by φ, while $E(\varphi)$ quantifies how far φ is from being a rigid-body motion.

The theory of linear elasticity rests upon on the following two fundamental approximations:
- *Small deformation approximation:* the Green-Saint-Venant strain tensor can be approximated by the *linearized strain tensor* $e(u) := \frac{{}^t\nabla u + \nabla u}{2}$, that is: $E(\varphi) \approx e(u)$.
- *Linearity of the material's behavior:* the material's constitutive law is generally defined as the rule which connects the Cauchy stress tensor σ to the strain. Under reasonable assumptions, it is of the following form (see [92], sections 3.6, 3.7):

$$\sigma = 2\mu E + \lambda \mathrm{tr}(E) + o(E),$$

  where λ, μ are the Lamé coefficients of the material. In linear elasticity, it is assumed that the higher order term $o(E)$ can be dropped in the above expression.

All in all, in the linear elasticity approximation, the internal stress $\sigma = \sigma(u)$ within Ω is related to the displacement field by the so-called *Hooke's law*:

$$\sigma(u) = Ae(u) = 2\mu e(u) + \lambda \mathrm{tr}(e(u))I.$$

Of course, these considerations do not hold for a general motion of Ω, and should be reserved to the description of the regime of 'small displacements'.

In this context, if Ω is clamped on a part Γ_D of its boundary $\partial\Omega$, submitted to body forces f, and traction loads g are applied on a part $\Gamma_N := \partial\Omega \setminus \Gamma_D$, the static equilibrium equations for u read:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(\sigma(u)) & = & f & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ \sigma(u)n & = & g & \text{on } \Gamma_N \end{array} \right. \tag{2.4}$$

Let us now outline the classical mathematical setting of the study of the linear elasticity system. The considered domain Ω is assumed to be bounded and Lipschitz. As regards Γ_D, we shall assume that it is of positive $(d-1)$ Hausdorff measure (unless an equilibrium relation is imposed between the body forces and the traction loads). The body forces f are assumed to belong to $H^{-1}(\Omega)^d$, and $g \in H^{-\frac{1}{2}}(\partial\Omega)^d$. One can then prove, using Korn's inequality in combination with the usual Lax-Milgram's lemma, that these equations admit a unique solution $u \in H^1_{\Gamma_D}(\Omega)^d$, where:

$$H^1_{\Gamma_D}(\Omega) := \left\{ v \in H^1(\Omega) \, , v = 0 \text{ on } \Gamma_D \right\}.$$

As is generally the case with elliptic equations, higher regularity results hold in the context of linear elasticity; yet, they are much harder to prove. As an illustration, one has the following regularity theorem in the setting of *pure displacement* in $\mathbb{R}^3$ (see [92], Th. 6.3.6).

Theorem 2.4. *Assume that $\Gamma_D = \partial\Omega$. Let $m \in \mathbb{N}^*$, $p \in]1, \infty[$, $p \geq \frac{6}{5+2m}$; assume the boundary $\partial\Omega$ enjoys $\mathcal{C}^{m+2}$ regularity, and $f \in W^{m,p}(\Omega)^3$. Then, the solution $u \in H^1_{\Gamma_D}(\Omega)^3$ to (2.4) actually belongs to $W^{m+2,p}(\Omega)^3$, and there exists a constant $C > 0$ which only depends on Ω such that:*

$$||u||_{W^{m+2,p}(\Omega)^3} \leq C||f||_{W^{m,p}(\Omega)^3}.$$

A similar regularity result holds in the case of *pure traction*, that is when $\Gamma_N = \partial\Omega$, and the usual equilibrium relation between f and g is satisfied so that there exists a unique solution u to (2.4) (upon higher regularity assumptions on g). However, as in the case of the Laplace operator , these results fail in the case when *mixed boundary conditions* are imposed, that is when Γ_D and Γ_N or both non empty: troubles are likely to appear in the vicinity of points x lying at the change of boundary conditions $x \in \widetilde{\Gamma_D} \cap \widetilde{\Gamma_N}$. Far and apart from these regions (if any), the regularity of u is as high as can be expected.

2.2.2.4 PDE constrained shape optimization

We are now interested in performing shape optimization with respect to objective functionals $J(\Omega)$ which depend on the domain Ω via solutions to the linear elasticity system. Let us sketch the context: we consider shapes, that is bounded Lipschitz domains $\Omega \subset \mathbb{R}^d$, which are clamped on a subset $\Gamma_D \subset \partial\Omega$ of positive $(d-1)$-Hausdorff measure, submitted to body forces $f \in H^1(\mathbb{R}^d)^d$, and traction loads $g \in H^2(\mathbb{R}^d)^d$, to be applied on another part of their boundary $\Gamma_N \subset \partial\Omega$. Neither Γ_D, nor Γ_N but only the *free boundary* $\Gamma := \partial\Omega \setminus (\Gamma_D \cup \Gamma_N)$ is subject to optimization. The displacement field u_Ω of Ω is the unique solution in $H^1_{\Gamma_D}(\Omega)^d$ to the system:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(\sigma(u)) & = & f & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ \sigma(u)n & = & g & \text{on } \Gamma_N \\ \sigma(u)n & = & 0 & \text{on } \Gamma \end{array} \right. . \tag{2.5}$$

According to the previous requirements, the considered set $\mathcal{U}_{ad}$ of *admissible shapes* is defined as:

$$\mathcal{U}_{ad} = \{\Omega \text{ bounded and Lipschitz, } \Gamma_D \cup \Gamma_N \subset \partial\Omega\}, \tag{2.6}$$

so that the set Θ_{ad} of *admissible variations* is:

$$\Theta_{ad} = \left\{\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ \theta = 0 \text{ on } \Gamma_D \cup \Gamma_N\right\}.$$

The first example that comes to mind as an objective function is the *compliance*

$$C(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\, dx = \int_\Omega f \cdot u_\Omega\, dx + \int_{\Gamma_N} g \cdot u_\Omega\, ds, \tag{2.7}$$

which can be equivalently interpreted as the work of external forces applied on Ω, or as the mechanical power spread by Ω in its motion. Another example - used in the device of MEMS (micro electromechanical systems) - is the following: if $u_0 \in H^1_{\Gamma_D}(\Omega)^d$ is a target displacement for u_Ω, $\alpha \geq 2$, and $k \in L^\infty(\mathbb{R}^d)$ is a weight function, we shall consider the least-square criterion

$$D(\Omega) = \left(\int_\Omega k\, |u_\Omega - u_0|^\alpha dx\right)^{\frac{1}{\alpha}}. \tag{2.8}$$

Let us eventually mention a major challenge in structural optimization, that of the device of structures in which the elastic stress is controlled. In this scope, we will encounter functionals of the form

$$S(\Omega) = \int_\Omega k||\sigma(u_\Omega)||^p\, dx, \tag{2.9}$$

where $k \in L^\infty(\mathbb{R}^d)$ is a weight function, $p \geq 2$, and $||.||$ denotes the Frobenius norm over $d \times d$ matrices. This short list is by no means exhaustive; for instance, a wide literature in structural optimization is devoted to criteria involving eigenfrequencies, etc... (see chap. 2 in [40] for numerous interesting optimization problems).

The rigorous computation of the shape derivative of functionals of the domain such as (2.7-2.9) is not an easy task. In practice, we rely on a formal method - much easier to carry out - to obtain the expressions of these shape derivatives, namely *Céa's fast derivation method*, introduced in [72]. It is formal in the sense that all the data at hand - objective functions, domains, deformations - are assumed smooth enough, and above all that the state system u_Ω is differentiable with respect to the shape.

Consequently, unless otherwise specified, we shall thenceforth assume that all the considered shapes $\Omega \in \mathcal{U}_{ad}$ are smooth enough (i.e. of class $\mathcal{C}^k$, for $k \geq 1$ large enough), and similarly for the deformation fields $\theta \in \Theta_{ad}$, and displacement functions u_Ω.

Let us now illustrate the kind of results and techniques we shall meet repeatedly in this manuscript with a representative example, quoted from [14]. Let $j, k : \mathbb{R}^d_x \times \mathbb{R}^d_u \to \mathbb{R}$ two smooth functions, fulfilling adequate growth conditions. Define an objective function $J(\Omega)$ as:

$$\forall \Omega \in \mathcal{U}_{ad}, \;\; J(\Omega) = \int_\Omega j(x, u_\Omega(x))\, dx + \int_{\Gamma \cup \Gamma_N} k(x, u_\Omega(x))\, ds. \tag{2.10}$$

Theorem 2.5. *Functional $J(\Omega)$ defined as (2.10) is shape differentiable at any $\Omega \in \mathcal{U}_{ad}$, and its shape derivative $J'(\Omega)$ reads:*

$$\forall \theta \in \Theta_{ad}, \;\; J'(\Omega)(\theta) = \int_\Gamma \left(j(x, u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega + \frac{\partial (k(x, u_\Omega))}{\partial n} + \kappa k(x, u_\Omega) \right) \theta \cdot n \, ds, \tag{2.11}$$

where n is the unit normal vector field to Ω, κ is the mean curvature of $\partial\Omega$ (oriented so that $\kappa(x)$ is positive when Ω is locally convex near x), and $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is the adjoint state, defined as the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -j'(u_\Omega) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ \sigma(p)n & = & -k'(u_\Omega) & \text{on } \Gamma \cup \Gamma_N \end{array} \right. . \tag{2.12}$$

Outline of proof. A rigorous proof of such a result is generally achieved within two steps.

1. First, one needs to prove the Fréchet differentiability of $\theta \mapsto J(\Omega_\theta)$, which requires a change of variables in the integrals defining J to bring them back to Ω or (a subset of) $\partial\Omega$, with only the integrands depending on θ (see [172], prop. 5.4.3). Doing so yields:

$$\begin{array}{rcl} J(\Omega_\theta) & = & \displaystyle\int_\Omega j(x + \theta(x), u_{\Omega_\theta}(x + \theta(x))) \det(I + \nabla\theta)\, dx \\ & & \displaystyle + \int_{\Gamma \cup \Gamma_N} k(x + \theta(x), u_{\Omega_\theta}(x + \theta(x))) \, |\mathrm{com}(I + \nabla\theta) \cdot n| \, ds \end{array} \tag{2.13}$$

   This expression naturally features the transported functions $u_{\Omega_\theta} \circ (I + \theta)$.

2. The most natural way for computing the derivative $J'(\Omega)$ is then to differentiate directly with respect to θ in (2.13), which brings into play the material derivative $\dot{u_\Omega}$ of $\Omega \mapsto u_\Omega$. Hence, one needs first to study the existence of $\dot{u_\Omega}$, and - if possible - to reach an expression for it. This stage consists in a use of the implicit function theorem in the variational formulation for u_Ω, and supplies a variational problem for $\dot{u_\Omega}$ (see [172], §5.3.3). Using this variational problem and that of the adjoint state p_Ω allows for a simplification in the expression of $J'(\Omega)$.

However perfectly rigorous, this method is not completely satisfactory as far as intuition is concerned. Indeed, the involvement of the adjoint state appears a bit miraculously in this context, and its expression is difficult to guess beforehand, whereas it is a key ingredient when it comes to making the expression of

$J'(\Omega)(\theta)$ completely explicit in θ (which is crucial in numerical practice).

Another method consists in considering the Lagrangian function $\mathcal{L} : \mathcal{U}_{ad} \times H^1_{\Gamma_D}(\mathbb{R}^d)^d \times H^1_{\Gamma_D}(\mathbb{R}^d)^d \ni (\Omega, v, q) \mapsto \mathcal{L}(\Omega, v, q) \in \mathbb{R}$, which incorporates the state equation for u_Ω as a constraint, using a Lagrange multiplier q:

$$\mathcal{L}(\Omega, v, q) = \int_\Omega j(v)\, dx + \int_{\Gamma \cup \Gamma_N} k(v)\, ds + \int_\Omega Ae(v) : e(q)\, dx - \int_\Omega f.q\, dx - \int_{\Gamma_N} g.q\, ds$$

Then, $J(\Omega)$ can be expressed in terms of $\mathcal{L}(\Omega, ., .)$ as:

$$\forall q \in H^1_{\Gamma_D}(\mathbb{R}^d)^d,\ \ J(\Omega) = \mathcal{L}(\Omega, u_\Omega, q). \tag{2.14}$$

Now, we compute the partial differentials of $\mathcal{L}(\Omega, ., .)$ at a given point $(u, p) \in H^1_{\Gamma_D}(\mathbb{R}^d)^d \times H^1_{\Gamma_D}(\mathbb{R}^d)^d$:

– The partial differential of $\mathcal{L}(\Omega, ., .)$ with respect to the q variable reads:

$$\forall q \in H^1_{\Gamma_D}(\mathbb{R}^d)^d,\ \ \frac{\partial \mathcal{L}}{\partial q}(\Omega, u, p)(q) = \int_\Omega Ae(u) : e(q)\, dx - \int_\Omega f \cdot q\, dx - \int_{\Gamma_N} g \cdot q\, ds.$$

Canceling this derivative at (u, p) yields the variational formulation associated to the state system (2.5), and is then equivalent to $u = u_\Omega$.

– Similarly, the partial differential of $\mathcal{L}(\Omega, ., .)$ with respect to the v variable reads:

$$\forall v \in H^1_{\Gamma_D}(\mathbb{R}^d)^d,\ \ \frac{\partial \mathcal{L}}{\partial v}(\Omega, u, p)(v) = \int_\Omega j'(u) \cdot v\, dx + \int_{\Gamma \cup \Gamma_N} k'(u) \cdot v\, ds + \int_\Omega Ae(v) : e(p)\, dx;$$

canceling this derivative is equivalent to the fact that $p = p_\Omega$, the unique solution to system (2.12).

To conclude, differentiating with respect to the domain in (2.14) for a fixed, arbitrary function $p \in H^1_{\Gamma_D}(\mathbb{R}^d)^d$ produces:

$$\forall \theta \in \Theta_{ad},\ \ J'(\Omega)(\theta) = \frac{\partial \mathcal{L}}{\partial \Omega}(\Omega, u_\Omega, p)(\theta) + \frac{\partial \mathcal{L}}{\partial v}(\Omega, u_\Omega, p)(\dot{u_\Omega}(\theta)), \tag{2.15}$$

because $\frac{\partial \mathcal{L}}{\partial q}(\Omega, u_\Omega, p) = 0$. Now evaluating (2.15) at $p = p_\Omega$ allows to cancel the unpleasant last term in the right-hand side, and yields:

$$\forall \theta \in \Theta_{ad},\ \ J'(\Omega)(\theta) = \frac{\partial \mathcal{L}}{\partial \Omega}(\Omega, u_\Omega, p_\Omega)(\theta). \tag{2.16}$$

Eventually, the last partial differential is computed using theorem 2.2 for the differentiation with respect to the domain Ω of integrals of fixed functions over Ω, or Γ, and leads to the desired formula. □

Remark 2.4. As a matter of fact, Céa's method can be rigorously justified in some cases (see [105], chap. 10, §5). Indeed, it is easily seen that $J(\Omega)$ can be expressed as a min-max value of the Lagrangian functional $\mathcal{L}(\Omega, ., .)$:

$$J(\Omega) = \min_{v \in H^1_{\Gamma_D}(\mathbb{R}^d)^d} \max_{q \in H^1_{\Gamma_D}(\mathbb{R}^d)^d} \mathcal{L}(\Omega, v, q).$$

Under certain hypotheses on the objective function $J(\Omega)$, it turns out that, for any $\Omega \in \mathcal{U}_{ad}$, this min-max value is actually a saddle point. Using a theorem for the differentiation of a saddle point with respect to a parameter (see [105], chap. 10, th. 5.1) yields the same conclusion (2.16) as above.

Expression (2.11) lends itself to a physical interpretation. Grossly speaking, the terms $j(x, u_\Omega)$ and $\left(\frac{\partial (k(x,u_\Omega))}{\partial n} + \kappa k(x, u_\Omega)\right)$ correspond to static force fields. The adjoint state defined by (2.12) is a displacement driven by forces pointing towards a decrease in the values of functions j and k. The term $Ae(u_\Omega) : e(p_\Omega)$ is the opposite of the power spread by the virtual displacement p, obtained when submitted to body force j'. Consequently, (2.11) indicates that the increase in performance of a given shape Ω with respect to J is governed by the following trends:

– Ω should 'flee' from where the criteria j and k are high,
– Ω should 'expand' (resp. 'retract') where the power received in a virtual move which improves its performance is positive (resp. negative).

Remarks 2.5.
– The shape derivative of the compliance $C(\Omega)$ has a remarkably simple expression, in the particular case when no body force is applied (i.e $f = 0$). Indeed, it is easy to see that the problem is *self-adjoint*, i.e. $p_\Omega = -u_\Omega$, and:

$$\forall \theta \in \Theta_{ad},\ C'(\Omega)(\theta) = -\int_\Gamma Ae(u_\Omega) : e(u_\Omega)\, \theta \cdot n\, ds.$$

Note that $Ae(u_\Omega) : e(u_\Omega)$ is everywhere non negative, which accounts for the intuitive face that, for the compliance of a shape to be reduced, it should be reinforced everywhere, and first and foremost in the areas where the dissipated elastic energy is high.
– Analogous results hold for other objective functions, e.g. objective functions of the form (2.9).

2.2.3 Shape optimization using Hadamard's method

Let Ω^0 an initial (smooth enough) shape, to be optimized with respect to a given criterion $J(\Omega)$, over a set $\mathcal{U}_{ad}$ of admissible shapes, which accounts for constraints.

2.2.3.1 The shape optimization problem

Up to this point, the only constraint we have been imposing on a shape Ω is that $\Gamma_D \cup \Gamma_N$ should be part of $\partial\Omega$. This constraint is especially easy to enforce, for if a shape Ω is such that $\Gamma_D \cup \Gamma_N \subset \partial\Omega$, and $||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1$ is a deformation field such that $\theta = 0$ on $\Gamma_D \cup \Gamma_N$, then $\Gamma_D \cup \Gamma_N \subset \partial\Omega_\theta$. However, a lot of different constraints may have to be taken into account when dealing with realistic problems. Unfortunately, their very mathematical formulation (let alone incorporation into a shape optimization algorithm) may prove very tedious. To name a few, the following constraints are of tremendous importance in shape optimization of elastic structures:
– One could require the *volume* $V(\Omega) := \int_\Omega dx$ or the *perimeter* $P(\Omega) := \int_{\partial\Omega} ds$ of an admissible shape Ω, to be equal, or lower or equal, to a prescribed upper bound.
– A whole class of particularly demanding constraints (yet crucial in the industrial context) is that of manufacturing constraints. Examples of such are constraints over the curvature of shapes, over the minimum authorized thickness for their members (to ensure the robustness of the associated mold), or the maximum authorized thickness (so that the cooling process is not hindered). These constraints often arise as pointwise contraints, and are numerically very sensitive. See [228] for detailed explanations.
– A huge litterature has been devoted to stress-based constraints, such as the famous Von Mises criterion. These constraints also often arise as pointwise contraints.

Admittedly, the conceptual difference between the cost and constraint functionals lies almost solely in the formulation of the optimization problem. For example, searching for a shape of minimal compliance under a volume constraint, and searching for a shape of minimal volume under a constraint on the compliance are two different optimization problems of equal relevance in practical applications.

In the remainder of this manuscript, we shall limit ourselves with constraints on the volume and perimeter of shapes. They shall be enforced owing to the simplest possible approach: the optimization problem is brought back to that of the constraint-free minimization of a (Lagrangian-like) weighted sum $\mathcal{L}(\Omega)$ of $J(\Omega)$, and $V(\Omega)$ (or $P(\Omega)$), the latter being penalized with a fixed Lagrange multiplier ℓ:

$$\min_{\Omega \in \mathcal{U}_{ad}} \mathcal{L}(\Omega),\ \ \mathcal{L}(\Omega) = J(\Omega) + \ell V(\Omega), \tag{2.17}$$

where $\mathcal{U}_{ad}$ stays defined by (2.6). This simple formulation may seem very crude. Yet, more sophisticated algorithms for handling constraints rely on such a formulation, in combination with an update strategy

for the value of the Lagrange multiplier, so that the constraint is fulfilled in the end (e.g. the quadratic penalty method, the augmented lagrangian method, or the log-barrier method, see [237], chap. 17). More sophisticated (and efficient) algorithms exist, such as the Method of Moving Asymptotes (MMA), described in [300], or the Method of Feasible Directions (MFD) [312], but we shall not deal with them in this manuscript.

2.2.3.2 A generic shape optimization algorithm

We are now in position to introduce a *shape gradient algorithm* for the considered problem (2.17). As suggested by the name, it relies on the knowledge of the first-order shape derivative $\mathcal{L}'(\Omega)$ to produce a descent direction $\theta \in \Theta_{ad}$ for $\mathcal{L}$. Higher order algorithms, relying for instance on the (exact or approximate) shape Hessian of $\mathcal{L}$ could improve considerably the efficiency of this algorithm. Unfortunately, they generally are much more tedious to devise, on both theoretical and numerical sides. Nevertheless, it is worth mentioning the interesting work [177] in this direction.

The Structure Theorem 2.3 states that only the normal component of the deformations undergone by Ω play a role in the shape derivative of 'reasonable' objective functions. Actually, the shape derivatives of all the functionals we shall get interested in enjoy a slightly more precise structure (see theorems 2.2 and 9.1). They are actually of the form,

$$\Theta_{ad} \ni \theta \mapsto \int_\Gamma v\,(\theta.n)\,ds,$$

for a certain scalar field v defined over Γ, which makes the computation of a descent direction for $\mathcal{L}(\Omega)$ very easy: letting

$$\theta = -v\,n$$

in the asymptotic expansion (5.44) yields, for $t > 0$ small enough:

$$\mathcal{L}(\Omega_{t\theta}) = \mathcal{L}(\Omega) - t\int_\Gamma v^2\,ds + o(t) < \mathcal{L}(\Omega),$$

that is, θ is a descent direction for $\mathcal{L}$ at Ω.

A generic algorithm for shape optimization using shape sensitivity analysis is then derived from these considerations as follows:

Starting with an initial guess Ω^0, **for** $n = 0, \ldots$ **till convergence**,

1. Compute the solution u_{Ω^n} to the linear elasticity system (2.5) on Ω^n, and (if need be) the adjoint state p_{Ω^n},
2. Infer a descent direction $\theta^n \in \Theta_{ad}$ for the objective functional at stake, along the lines of section 2.2.2.4,
3. Choose a descent step $\tau^n > 0$ small enough so that $J(\Omega^n_{\tau^n\theta^n}) < J(\Omega^n)$.
4. The new shape is obtained as $\Omega^{n+1} = \Omega^n_{\tau^n\theta^n}$.

2.2.3.3 A look at velocity extension and regularization

The above shape optimization algorithm uses a descent direction under the form of a vector field defined on the free boundary Γ of the current shape Ω. Although it is the only needed piece of information to carry out the theoretical algorithm, we shall see soon that it will be very convenient to work with a velocity field which is consistently extended to the whole space $\mathbb{R}^d$ (or at least a neighborhood of Γ). What's more, this descent direction may prove very irregular (i.e. present sharp variations). On the practical side, this may cause uncontrolled oscillations on the boundary of the considered shapes, and jeopardize the numerical stability of a shape optimization algorithm (see [232], chap. 6).

These are classical issues in gradient-based optimization algorithms. The usual solution to both problems consists in changing scalar products when identifying a descent direction θ from the formula for the shape

derivative (2.2.3.2) [64, 162]. For instance, let us give a hint of how to extend and regularize the scalar field v into another scalar field $\widetilde{v}$ (note that one could also chose to extend and regularize directly the velocity field $\theta = -vn$). Let V be a Hilbert space which is composed of functions enjoying the desired regularity for $\widetilde{v}$, a a coercive bilinear on V form which is close to I (so that $\widetilde{v}$ is hopefully close to v). Usual choices in our context are $V = H^1(\mathbb{R}^d)$, and:

$$\forall \phi, \psi \in V, \;\; a(\phi, \psi) = \alpha^2 \int_{\mathbb{R}^d} \nabla\phi.\nabla\psi \, dx + \int_{\mathbb{R}^d} \phi\psi \, dx,$$

for a small $\alpha > 0$, which can be interpreted as a regularization lengthscale. Then $\widetilde{v}$ is searched as the unique solution in V to the variational problem:

$$\forall \phi \in V, \;\; a(\widetilde{v}, \phi) = \int_\Gamma v\phi \, ds.$$

See [162] for a discussion over the importance of this procedure in the context of shape optimization.

At this point, some numerical issues still need to be solved, the most serious of which being the question of how to couple this theoretical framework with a numerical method for representing and deforming shapes.

2.3 Shape optimization using the level set method

In this section, we dwell on the description of the level set method for shape optimization, as originally proposed by G. Allaire, F. Jouve and A.-M. Toader in 2004 [14], following previous works of J. Sethian and A. Wiegmann [278] and S. Osher and F. Santosa [243] (see also [319]). The reason for doing so is twofold: first, we shall use this method as such in chapters 4 and 5; what's more, the mesh evolution method for shape optimization proposed in chapter 9 is closely modeled on it.

Let D a fixed computational domain, in which all the shapes $\Omega^0, ...\Omega^n, ...$ produced during the shape optimization process are to be included. D is meshed once and for all, e.g. using a Cartesian mesh (which allows for the use of finite difference schemes for operations related to the Level Set Method). Each shape Ω^n is represented as the negative subdomain of a level set function ϕ^n (numerically discretized at the nodes of the Cartesian mesh of D).

The main problem is that, as the mesh of D is fixed, no mesh of the shape Ω^n is available to perform the mechanical analyses needed when it comes to computing the shape gradient $J'(\Omega)$. This difficulty is overcome by approximating the state problem, posed on Ω^n, with another problem in linear elasticity, posed on the whole domain D. The *Ersatz material approach* consists in filling the complementary part $D \setminus \Omega$ of a shape $\Omega \subset D$ with a very soft material of Hooke's tensor εA, $\varepsilon \ll 1$. The resulting approximated problem on D reads then:

$$\left\{ \begin{array}{cc} -\mathrm{div}(A_\Omega e(u)) = f & \text{in } D \\ u = 0 & \text{on } \Gamma_D \\ A_\Omega e(u)n = g & \text{on } \Gamma_N \end{array} \right. , \tag{2.18}$$

where the total Hooke's tensor A_Ω is defined as:

$$\forall x \in D, \;\; A_\Omega(x) = \left\{ \begin{array}{cc} A & \text{if } x \in \Omega \\ \varepsilon A & \text{if } x \in D \setminus \Omega \end{array} \right. .$$

Consistency of this approach can be shown (see [8]).

The shape optimization using the level set method can now be summed up as follows:

Starting with an initial guess Ω^0, **for** $n = 0, \ldots$ **till convergence**,

1. Compute the solution u_{Ω^n} to the linear elasticity system (2.18) on D, and (if need be) the adjoint state p_{Ω^n}, using the Ersatz material approximation.
2. Infer a descent direction $\theta^n \in \Theta_{ad}$ for the objective functional at stake, along the lines of section 2.2.2.4 (which is consistently extended to D), and extend it to a neighborhood of $\partial\Omega^n$, as evoked in section 2.2.3.3.
3. Choose a descent step $\tau^n > 0$, and solve the Hamilton-Jacobi equation:

$$\begin{cases} \frac{\partial \phi}{\partial t} + \theta^n . \nabla \phi = 0 & \text{on } [0, \tau^n] \times D \\ \phi(t = 0, .) = \phi^n & \text{on } D \end{cases}$$

   e.g. using one of the numerical schemes described in chapter 1.
4. The new shape Ω^{n+1} is defined as $\Omega^{n+1} := \{x \in D, \phi(\tau^n, x) < 0\}$.

We end this section by giving some examples, obtained using the level set method, which illustrate each one of the above functionals of the domain (2.7, 2.8, 2.9). The first two are excerpted from [14], whereas the last one stems from [13]. We shall reproduce these examples in different contexts in the next chapters.

Example 2.2. Let us start with the most famous example in structural optimization, namely the benchmark *Cantilever* test case. A cantilever, included in a computational box D of dimensions 2×1, equipped with a Cartesian mesh of 160×80 elements, is clamped at its left side, and a unit vertical load is exerted at the centre of its right side (no body force is applied). The compliance (2.7) of the structure is minimized, under a volume constraint enforced using a fixed Lagrange multiplier $\ell = 150$. 150 iterations of the optimization process are performed, and the results are reported on figure 2.4.

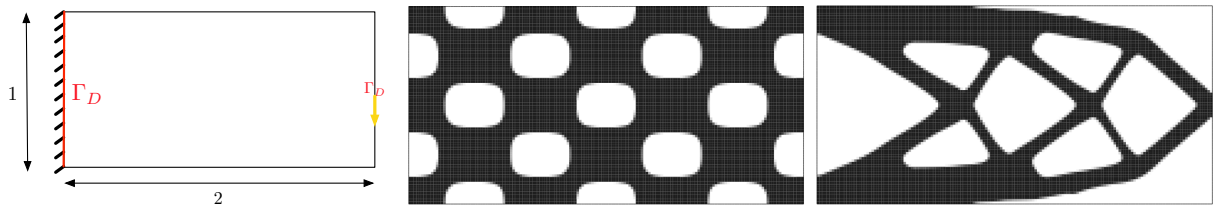


Figure 2.4: *(Left)* Boundary conditions, *(middle)* initialization, and *(right)* resulting shape in the cantilever test case (reprinted from [14]).

Example 2.3. We search for the optimal shape of a gripping mechanism: the problem consists in getting a maximal displacement of the jaws of a grip, as a response to prescribed traction loads. Details on the test case and results are reported on figure 2.5. This test case is especially interesting for at least two reasons: on the mathematical side, the minimization problem of a least-square functional such as (2.8) is not self-adjoint, as is the one of the compliance (2.7) - see remark 2.5. On the numerical side, this test case happens to be very sensitive, notably due to the very thin features which have to develop so as to bestow enough flexibility to the shape.

Example 2.4. Stress reduction has long been a topic of major interest in structural optimization. The last proposed test case consists in minimizing the stress-based criterion (2.9) in the situation depicted on figure 2.6 (left). The localizing weight k is taken equal to 1, except in a small region around the load point and the fixation wall, and several values for the exponent p are considered. Notice that, for large values of p, the reentrant corner - which is the area of highest stress concentration - is rounded up.

Remark 2.6. In section 2.2.1, we hinted at the fact that all the produced shapes by means of the generic algorithm of section 2.2.3.2 should be homeomorphic to one another. The conclusions of examples 2.2 and

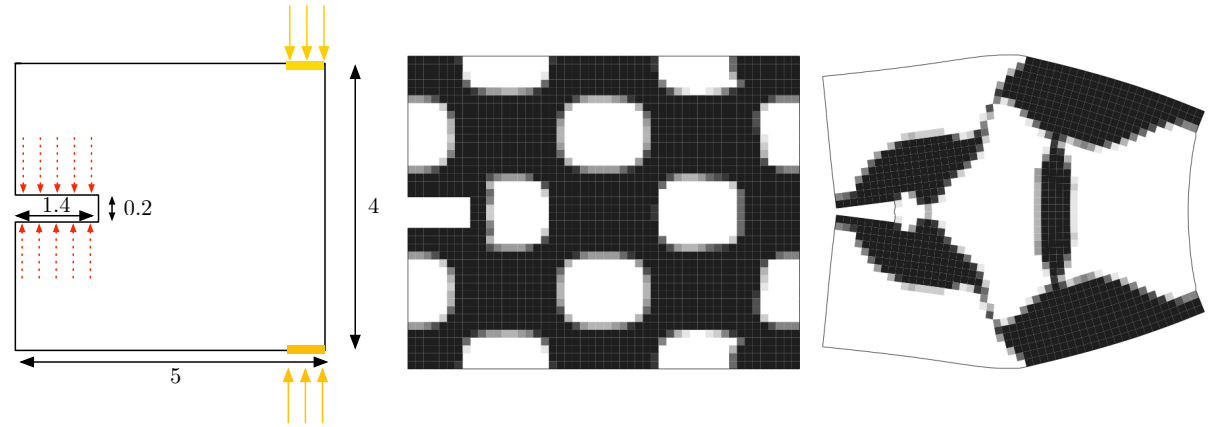


Figure 2.5: *(Left)* Boundary conditions, *(middle)* initialization, and *(right)* resulting shape in the gripping mechanism test case (reprinted from [14]).

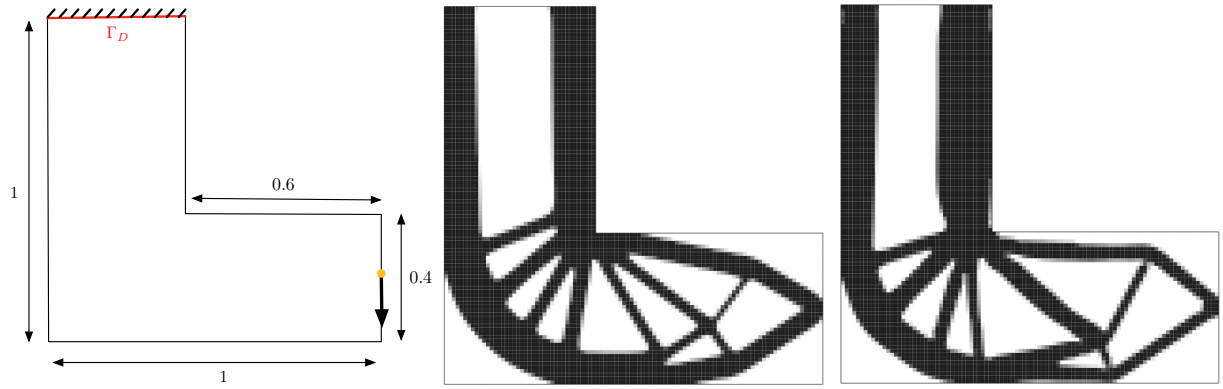


Figure 2.6: *(Left)* Boundary conditions, *(middle-right)* respective final shapes in the L-Beam test case, for values of the exponent $p = 2, 10$ (reprinted from [13]).

2.3 may then seem weird in this regard, since the number of holes of the shapes have changed from the initial to the final stages. This is actually a numerical hack of Hadamard's method: at some iteration n, the descent step τ^n in step (3) of the previous algorithm has not been chosen so small that $(I + \tau^n \theta^n)$ is a Lipschitz diffeomorphism (yet, decreasing of the objective function is assessed).

Chapter 3

Mesh generation, modification and evolution

Contents

3.1 Generalities around meshes: definitions, notations, and useful concepts . . . 68
3.1.1 Definitions and notations . . . 68
3.1.2 Appraising the quality of a mesh . . . 70
3.1.3 The Riemannian paradigm for size and orientation specifications in meshing . . . 71
3.2 Mesh generation techniques . . . 73
3.2.1 Two and three-dimensional 'volume' mesh generation . . . 73
3.2.1.1 Delaunay-based mesh generation . . . 74
3.2.1.1.1 Definition and properties of the Delaunay triangulation of a set of points . . . 74
3.2.1.1.2 Step 1: generation of the Delaunay triangulation of a set of points 75
3.2.1.1.3 Step 2: enforcement of the entities of Σ in the resulting mesh . . 78
3.2.1.1.4 Some post-processing issues . . . 79
3.2.1.2 Advancing front methods for mesh generation . . . 80
3.2.1.3 Meshing implicit domains . . . 82
3.2.2 Surface mesh generation . . . 85
3.2.2.1 Direct methods . . . 85
3.2.2.2 Mesh generation using the parameter space . . . 85
3.2.2.3 Extension to more complex surfaces . . . 86
3.3 Local remeshing . . . 87
3.3.1 Volume remeshing . . . 87
3.3.1.1 Mesh enrichment operators . . . 87
3.3.1.2 Mesh decimation operators . . . 88
3.3.1.3 Connectivity changes . . . 88
3.3.1.4 Vertex relocation . . . 89
3.3.2 Surface remeshing . . . 90
3.3.2.1 Remeshing through parametrization of the surface . . . 90
3.3.2.2 Direct remeshing of the surface . . . 91
3.4 Mesh evolution . . . 92
3.4.1 Purely Lagrangian methods . . . 92
3.4.1.1 Lagrangian deformation of a surface triangulation . . . 93
3.4.1.1.1 Connections with remeshing . . . 93

3.4.1.1.2 Modifying the input velocity field 94
3.4.1.1.3 Resolving intersections 94
3.4.1.2 Deforming a volume mesh together with its surface mesh 95
3.4.2 Hybrid methods 96

Meshing issues lie at the upstream of most numerical considerations: indeed, meshes are commonly used as a means for representing or deforming shapes in computer graphics; on a different note, in the field of numerical simulation of mechanical or physical phenomena, the bulk of techniques (e.g. finite element or finite volume methods) rely on a mesh as a computational support.

This (admittedly verbose) chapter is aimed at providing a non exhaustive overview of the stakes and salient features of three major topics around meshing, namely:

- *Mesh generation:* building a mesh out of a set of numerical data (e.g. a CAD representation) regarding a mechanical part is generally the first step in simulating related phenomena.
- *Mesh deformation:* a mesh accounting for an evolving domain may have to be deformed while tracking the underlying physical process.
- *Mesh modification:* not all meshes are equally suitable as supports of numerical simulations, and 'ill-shaped' meshes may have to be 'improved' in this perspective.

Here we only deal with simplicial meshes (i.e. meshes whose elements are triangles in two dimensions, tetrahedra in three dimensions), mostly from the standpoint of numerical simulation. Besides, we are especially interested in the three-dimensional context, in which meshing features often do not arise as straightforward generalizations of the two-dimensional ones. The material presented below is therefore systematically discussed with the three-dimensional case in mind. We shall however give a hint of some simplifications available in two dimensions, when there are.

In truth, many meshing techniques are application-specific. In this chapter, we limit ourselves to outlining the main facets of the discussed topics. In chapter 8, we shall be focusing extensively on one implementation of a (re)meshing algorithm. Hence, whenever possible, the general idea of a technique will be given in this chapter, referring then to chapter 8 for an illustration in the situation of a particular application.

This chapter is organized as follows: in section 3.1, several definitions and notations are introduced, as well as two important concepts, those of *element quality*, and *Riemannian metric* associated to a size prescription, which we shall encounter throughout a substantial part of this manuscript (chapters 6 7, and 8). Then, in section 3.2, we present the main techniques for generating a tetrahedral mesh, or a surface mesh, depending on the form under which the domain (or surface) to mesh is supplied to the algorithm. Although this manuscript is not concerned so much about mesh generation issues, we indeed saw fit to give a clue of the main methods for they have a lot in common with further topics discussed hereafter. What's more, the difficulties inherent to these methods may explain several biases in the device of the mesh evolution strategy presented in chapter 9. Section 3.3 deals with modification (or optimization) of tetrahedral or (three-dimensional) surface meshes: in particular, the most common operations are detailed. Eventually, section 3.4 is devoted to the topic of mesh evolution; the main goals and difficulties are overviewed, and several particular methods, useful in different contexts, are presented.

3.1 Generalities around meshes: definitions, notations, and useful concepts

3.1.1 Definitions and notations

Credit where credit is due, let us start by defining the objects at stake in this chapter:

Definition 3.1. *Let $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ a bounded, open, polygonal domain. A simplicial mesh $\mathcal{T}$ (triangulation in two dimensions, tetrahedralization in three dimensions) of Ω is a finite collection $(K_i)_{i=1,\dots,N_\mathcal{T}}$ of closed d-simplices (triangles in two dimensions, tetrahedra in three dimensions), such that the following conditions hold :*

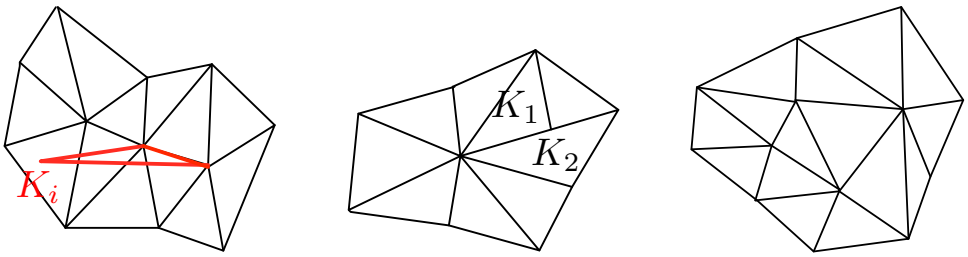
1. *The elements of $\mathcal{T}$ form a covering of Ω in the sense that : $\overline{\Omega} = \bigcup_{i=1}^{N_\mathcal{T}} K_i$.*
2. *Each simplex of $K_i \in \mathcal{T}$ has non empty interior: $\overset{\circ}{K}_i \neq \emptyset$.*
3. *Every two distinct simplices $K_i, K_j \in \mathcal{T}$, $i \neq j$ have disjoint interiors : $\overset{\circ}{K}_i \cap \overset{\circ}{K}_j = \emptyset$.*

These requirements are often supplemented with the following condition:

4. *For every two distinct simplices $K_i, K_j \in \mathcal{T}$, $i \neq j$, the intersection $K_i \cap K_j$ is*
   - *either a point, or a common edge to K_i and K_j in two dimensions,*
   - *either a point, or a common edge or a common (triangular) face to K_i and K_j in three dimensions.*

The vertices of the simplices $K_i \in \mathcal{T}$ are called the vertices of $\mathcal{T}$; likewise, the edges of those simplices are called the edges of $\mathcal{T}$, etc...

Condition (2) above bans from the definition of a mesh those sets of simplices containing flat elements, whereas condition (3) prevents the considered meshes from containing overlapping elements. In the sequel, such sets of simplices will sometimes be referred to as *invalid meshes*. The last condition (4) is called the *conformity assumption*, and is sometimes not required in the definition of a mesh. It roughly states the two-dimensional (resp. three-dimensional) meshes we are interested in have triangles (resp. tetrahedra) matching in an edge-to-edge (resp. face-to-face) fashion. See figure 3.1 for illustrations.



(a) (b) (c)

Figure 3.1: *(a)* Invalid mesh: triangle K_i (in red) overlaps several other triangles; *(b)* non conforming mesh: triangles K_i and K_j have an intersection which is neither a vertex, nor an common edge; *(c)* a conforming mesh, in the sense of definition 3.1.

Most often, we will speak of a *mesh* in $\mathbb{R}^d$ without mentioning the associated polygonal domain. We will also improperly refer to a mesh of a polyhedron 'close' to a non polyhedral domain Ω as a mesh of Ω.

Definition 3.2. *Let $\mathcal{T}$ a mesh in $\mathbb{R}^d$, $x \in \mathcal{T}$ a vertex, $pq \in \mathcal{T}$ an edge.*

- *The ball of point x is the (closed) set $\mathcal{B}(x)$ of simplices $K \in \mathcal{T}$ such that x is a vertex of K.*
- *The shell of edge pq is the (closed) set $\mathcal{S}h(pq)$ of simplices $K \in \mathcal{T}$ such that pq is an edge of K.*

In this manuscript, we will naturally be led to consider meshes that are included in a larger mesh in a conforming fashion:

Definition 3.3. *Let $\mathcal{T}$ a simplicial, conforming mesh in $\mathbb{R}^d$. A submesh of $\mathcal{T}$ is a finite collection $\mathcal{T}\prime$ of closed d-simplices $(K_j)_{j=1,\dots,N_{\mathcal{T}\prime}}$ such that, for any $j = 1, \dots, N_{\mathcal{T}\prime}$, K_j is an element of $\mathcal{T}$.*

Besides, we will also consider meshes of entities of codimension 1, that is curves in $\mathbb{R}^2$, and especially surfaces in $\mathbb{R}^3$. The forthcoming definitions are dedicated to the last case, but are easily adapted to meshes of the boundary of two-dimensional domains.

Definition 3.4. *Let $\Gamma \subset \mathbb{R}^3$ a compact polyhedral surface, with or without boundary. A surface mesh or triangulation $\mathcal{S}$ of Γ is a collection $(T_i)_{i=1,\dots,N_S}$ of closed (two-dimensional) triangles $T_i \subset \mathbb{R}^3$ enjoying the following properties:*

1. *The elements of $\mathcal{S}$ form a covering of Γ in the sense that : $\Gamma = \bigcup_{i=1}^{N_S} T_i$.*
2. *Each triangle of $T_i \in \mathcal{S}$ has non zero two-dimensional Hausdorff measure (i.e. is not 'degenerated to an edge or a point').*
3. *The intersection $T_i \cap T_j$ of every two distinct triangles $T_i, T_j \in \mathcal{S}$, $i \neq j$ is included in the set of edges of triangles of $\mathcal{S}$.*

and occasionnally the following conformity property:

4. *For every two distinct triangles $T_i, T_j \in \mathcal{S}$, $i \neq j$, the intersection $T_i \cap T_j$ is either a point, or a common edge of T_i and T_j.*

Once again, the vertices of the triangles $T_i \in \mathcal{T}$ form the vertices of $\mathcal{S}$, etc...

Like in the case of domains, we shall actually speak of meshes of non polyhedral surfaces.

The notions of *ball of a point*, and *shell of an edge* defined previously in the case of simplicial meshes of domains extend straightforwardly to the case of surface triangulations. However, when dealing with such objects, some specific features may be considered:

Definition 3.5. *Let $\mathcal{S}$ be a surface mesh in $\mathbb{R}^3$.*

- *One says that $\mathcal{S}$ is manifold, provided the associated polygonal surface Γ is a compact submanifold of $\mathbb{R}^3$, with or without boundary.*
- *If $\Gamma \subset \mathbb{R}^3$ is a smooth compact surface, with or without boundary, one says that $\mathcal{S}$ interpolates Γ if every vertex $x \in \mathcal{S}$ belongs to Γ.*

To each simplicial mesh $\mathcal{T}$ in $\mathbb{R}^3$, a natural surface mesh $\mathcal{S}_\mathcal{T}$ is associated, collecting the external (triangular) faces of the tetrahedra of $\mathcal{T}$. The surface mesh constructed in this way accounts for a compact polygonal surface of $\mathbb{R}^3$ without boundary.

Definition 3.6. *Let $\mathcal{T}$ be a mesh in $\mathbb{R}^3$, $\mathcal{S}_\mathcal{T}$ the associated surface mesh, and let $x \in \mathcal{T}$ be a surface vertex, i.e. $x \in \mathcal{S}_\mathcal{T}$. The surface ball $\mathcal{B}_\mathcal{S}(x)$ of x is the set of surface triangles $T \in \mathcal{S}_\mathcal{T}$ sharing x as a vertex.*

This surface mesh $\mathcal{S}_\mathcal{T}$ plays a central role when considering $\mathcal{T}$, insofar as it concentrates all the information about the *geometric approximation* of the underlying continuous geometry.

3.1.2 Appraising the quality of a mesh

Independently of how well a mesh $\mathcal{T}$ approximates the continuous geometry Ω it is intended for - we will come back to this problem of *geometric approximation* of a continuous domain in chapter 8 - the 'numerical performances' of $\mathcal{T}$ are also greatly impacted by the shape of its elements $K \in \mathcal{T}$. Indeed,

- Many classical a priori estimates for the finite element method involve the quality of the computational mesh $\mathcal{T}$ through the *aspect ratio* σ_K of its elements, defined as [91]:

$$\sigma_K = \frac{\rho_K}{h_K}, \tag{3.1}$$

  where ρ_K is the *inradius* of a simplex K - the radius of its inscribed sphere - and h_K is the *diameter* of K i.e. the length of its longest edge. This measure only depends on the shape of the considered simplex and not on its size. From a practical point of view, this implies that the accuracy of a finite element computation performed on $\mathcal{T}$ is of course influenced by the size of its elements, but also by their being 'well-shaped'.

– The accuracy of the $\mathbb{P}^1$ Lagrange interpolation of a given smooth function $f : \mathbb{R}^d \to \mathbb{R}$ on a mesh $\mathcal{T}$ in $\mathbb{R}^d$ is also greatly influenced by the shape of its elements; see the discussion in [282].
– The accuracy of the approximation of geometric quantities (normal vectors, curvatures estimates, etc...) attached to a surface Γ by means of numerical schemes performed on an associated surface mesh $\mathcal{S}$ is also highly dependent on the shape of its elements and its connectivities [227].

Consequently, any mesh operation (generation, modification, deformation) should be performed keeping in mind the concern of imposing, restoring or maintaining high-quality elements in mind.

Actually, many definitions for the quality of a simplex $K \subset \mathbb{R}^d$ could be adopted, all of them being equivalent from the theoretical point of view as long as they allow to discriminate 'ill-shaped', nearly degenerate (flat) elements from 'well-shaped', almost equilateral ones. For instance, in addition to the aspect ratio (3.1), some authors think it better to assess the quality of a d-dimensional simplex K based on the minimum dihedral angle between two of its faces, or on the following ratio:

$$Q(K) = \frac{\mathrm{Vol}(K)}{\left(\sum_{i=1}^{na} \ell(e_i)^2\right)^{\frac{d}{2}}}, \tag{3.2}$$

where $na = d(d+1)/2$ is the number of edges of a d-dimensional simplex, e_i are the edges of K, and $\ell(e_i)$ stands for the length of e_i. A huge literature is devoted to the topic of quality functions for simplices: see for instance [145] §18.2, or [3, 204] for other examples and comparisons.

From the numerical standpoint, one should be very careful to select a quality function which is neither too severe in evaluating a simplex as a 'bad' one (i.e. evaluating an element as 'bad' as soon as it slightly deviates from the ideal shape of an element), for it would prevent almost any operation to be held on meshes, nor too indulgent, for it would allow for meshes with too many 'second-rate' elements.

3.1.3 The Riemannian paradigm for size and orientation specifications in meshing

In various applications, an appreciated increase in accuracy or computational efficiency can be achieved if the computational mesh complies with some user-specified size or orientation information (see the examples of figure 3.2).

Following the lead of the pioneering work [311], a very convenient and elegant way to encode both size and orientation requirements makes use of a Riemannian framework. In particular, it allows for a straightforward (in theory) generalization to the anisotropic case of most of the concepts of this chapter (mostly presented in the isotropic setting), up to an adequate change in the definitions of distance and volume.

Definition 3.7. *Let M be a Riemannian metric over $\mathbb{R}^d$ (i.e. at each point $x \in \mathbb{R}^d$, $M(x)$ is a symmetric, positive definite $d \times d$ matrix); then,*

– *the length $\ell_M(\gamma)$ of a differentiable curve $\gamma : [0,1] \to \mathbb{R}^d$ with respect to M is defined as:*

$$\ell_M(\gamma) = \int_0^1 \sqrt{\langle M(\gamma(t))\gamma'(t), \gamma'(t)\rangle} dt.$$

– *The volume $V_M(K)$ of a simplex K (with respect to M) is:*

$$V_M(K) = \int_K \sqrt{det(M(x))} dx.$$

– *The distance $d_M(x,y)$ between two points $x, y \in \mathbb{R}^d$ in the Riemannian space $\left(\mathbb{R}^d, M\right)$ is defined as:*

$$d_M(x,y) = \inf_{\substack{\gamma \in \mathcal{C}^1([0,1],\mathbb{R}^d) \\ \gamma(0)=x, \gamma(1)=y}} \ell_M(\gamma).$$

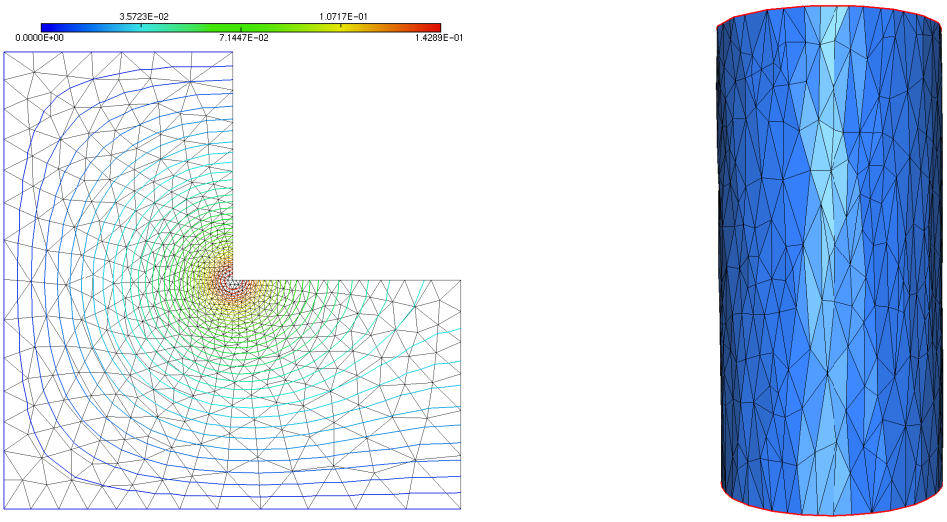


Figure 3.2: *(Left)* A physical phenomenon showing sharp variations near the reentrant corner of a L-shaped domain can be accurately captured using a mesh whose vertex density is concentrated in this region; *(right)* the geometry of a cylinder is optimally described using stretched elements, oriented along its principal axis.

Assume now that a Riemannian metric M is given on $\mathbb{R}^d$. An *adapted mesh* $\mathcal{T}$ in $\mathbb{R}^d$ with respect to M is a *unit* (or, in a more realistic way, *quasi-unit*) mesh with respect to M, that is, all its simplices have edges lengths equal to 1 (resp. lying in $\left[1/\sqrt{2}, \sqrt{2}\right]$), in the sense of definition 3.7 (note that in practice, $M(x)$ is defined only at the nodes of $\mathcal{T}$ - or any background structure - and interpolated from these values [145]).

In the particular case that M is a multiple of the identity matrix, that is, for any $x \in \mathbb{R}^d$, $M(x) = h(x)I$, with $h(x) > 0$, the associated size prescription is said to be isotropic, and h is the associated *size function*.

So as to better understand the connection between this notion of adapted mesh to a Riemannian metric M and a size and orientation prescription, consider the following idealized situation: let $\mathcal{T}$ a unit mesh with respect to M, and x_0 a a vertex of $\mathcal{T}$ such that $M(x) \equiv M$ is almost constant around x_0. Then, every simplex K of $\mathcal{T}$ lying in the ball $\mathcal{B}(x_0)$ of x_0 is inscribed in the ellipsoid $\Phi_M(x_0)$, defined as:

$$\Phi_M(x_0) = \left\{x \in \mathbb{R}^d,\ d_M(x, x_0) = 1\right\} = \left\{x = \sum_{i=1}^{d} x_i\, e_i \in \mathbb{R}^d,\ \lambda_1 x_1^2 + ... + \lambda_d x_d^2 = 1\right\},$$

where the eigenvectors $e_1, ..., e_d$ of M account for the directions of the principal axis of this ellipsoid, while the associated eigenvalues $\lambda_1, ..., \lambda_d$ are linked to the principal radii (or prescribed lengths) $h_1, ..., h_d$ in direction $e_1, ..., e_d$ by : $h_i = \frac{1}{\sqrt{\lambda_i}}$, $i = 1, ..., d$ (see figure 3.3). If the size prescription is *isotropic* (i.e. all the h_i are equal), M is a scalar multiple of the identity matrix: $M = \alpha I_d$, $\alpha > 0$, and only the *size function* α may be considered.

Eventually, note that, especially when the prescribed size feature is *anisotropic*, very stretched elements may be desired (see again figure 3.3), which do not fulfill the standard quality requirements, hinted at in the previous section. These quality functions must be traded for their anisotropic counterparts, obtained e.g. by using the very same expressions (3.1,3.2), except that the distance an volume notions are those supplied by M.

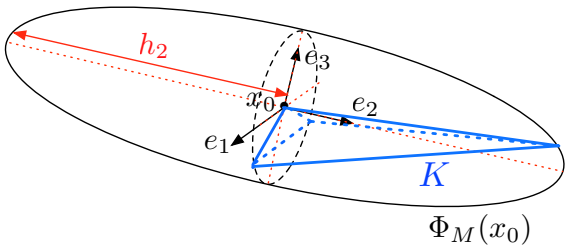


Figure 3.3: The unit ellipsoid $\Phi_M(x_0)$ associated to a (constant) Riemannian metric M, with principal axis e_1, e_2, e_3, and associated principal radii h_1, h_2, h_3. In blue, a unit simplex K with respect to M, sharing x_0 as a vertex, is depicted.

Remark 3.1. The metaphor at the place of honor in this section has been carried further in the work [7], which points out at a duality between a Riemannian structure M over $\mathbb{R}^d$ and a unit mesh of a given domain Ω with respect to M. This notably allows for a particularly elegant understanding of the dependence on the mesh of the interpolation error of a smooth function.

Representative examples of construction (and use) of a size map or a Riemannian metric in the perspective of mesh adaptation will be provided in chapters 6, 7 and 8, and we shall address additional related issues (such as that of *mesh gradation*) in chapter 8.

3.2 Mesh generation techniques

This section is intended as a superficial glimpse of the two intimately linked topics of *volume* and *surface mesh generation*. Few theoretical results are available to assess that a particular method will always succeed - at least with satisfying computational efficiency. For this reason, a great deal of the efficiency of any mesh generation method lies in the resort to some heuristics as well as on the attention paid to numerical implementation (see [145] for further details). In this view, the forthcoming descriptions are mere prototypical outlines, which are hopefully representative of the main features of each method.

3.2.1 Two and three-dimensional 'volume' mesh generation

Constructing a simplicial mesh $\mathcal{T}$ of a polyhedral domain $\Omega \subset \mathbb{R}^d$ ($d = 2$ or 3) in an automatic and robust fashion is possibly the most crucial problem related to meshing in numerical applications. A conceptual gap in difficulty lies between the two- and three-dimensional instances of this issue, which is highlighted by the following theoretical facts:

- In two dimensions, any polygonal domain Ω with non self-intersecting (i.e. manifold) boundary can be endowed with a triangulation whose vertices are exactly those of the boundary polygon. The proof of this fact (see [151], §3.3.3) is moreover constructive. In practice, various and very robust algorithms exist in two-dimensions, which are guaranteed to succeed; see for instance [80], chap. 2, for a more detailed presentation.
- In three dimensions, even very simple polyhedra Ω cannot be meshed without introducing internal points (see figure 3.4). Actually, the problem of deciding whether a given (non convex) polyhedron can be meshed without introducing additional points has been proved to be NP-complete in [272]. In the same vain, for any $k \in \mathbb{N}$, the problem of deciding whether a polyhedron can be meshed by introducing less than k additional vertices (either on the surface, or in the interior) is NP-hard. Conversely, it

has been proved in [76] that a three-dimensional polyhedron with n vertices can always be meshed, provided a (provably optimal) number $\mathcal{O}(n^2)$ of additional vertices is added.

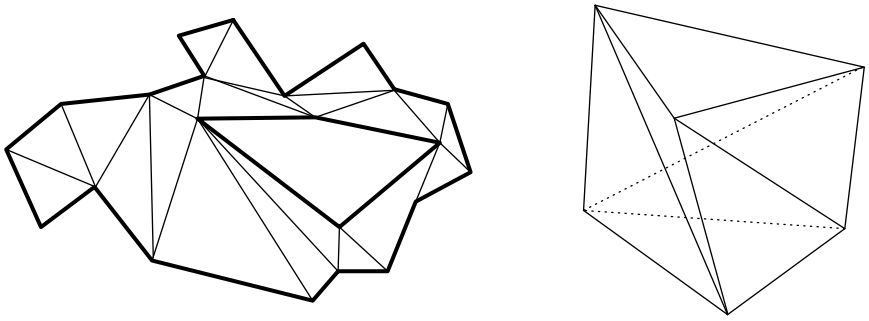


Figure 3.4: *(Left):* In two dimensions, any polygonal domain can be triangulated using only the vertices of its boundary; *(right)* in three dimensions, *Schönardt's polyhedron* is a non-convex polyhedron obtained by twisting a certain partition of a regular prism. It cannot be meshed without introducing any internal point.

In most applications, the domain Ω to be meshed is described via its boundary $\partial\Omega$, which is in turn often supplied as an associated surface mesh (the question of how to construct such a surface mesh is overviewed in section 3.2.2). The two most famous mesh generation methods, presented in sections 3.2.1.1 and 3.2.1.2, assume input data of this kind. However, in section 3.2.1.3, we shall examine a rather different context.

3.2.1.1 Delaunay-based mesh generation

Delaunay-based mesh generation algorithms take as an entry point a surface triangulation $\mathcal{S}$ (a boundary mesh in $2d$), with the goal to mesh the interior polygonal domain Ω. Actually, this problem can be posed in the more general setting of triangulation of *piecewise linear complexes* (i.e. sets of entities such as edges, faces that do not necessarily form a closed surface) [280], but we will not need so much generality in the short forthcoming overview.

Delaunay-based methods are probably among the most popular mesh generation methods, owing to their great robustness and versatility: we will encounter a great part of the numerical tools it involves (and notably the vertex insertion procedure) in other fields related to meshing.

3.2.1.1.1 Definition and properties of the Delaunay triangulation of a set of points The mesh generation method under scrutiny in this section is named after a particular partitioning of the space associated to a given set of vertices, namely its *Delaunay triangulation*.

Definition 3.8. *Let $\mathcal{P} = \{p_i\}_{i=1,\dots,N}$ a finite set of points in $\mathbb{R}^d$.*

- *A triangulation*[1] *of $\mathcal{P}$ is a simplicial mesh $\mathcal{T}$ of the (polygonal) convex hull conv($\mathcal{P}$) of $\mathcal{P}$.*
- *A Delaunay triangulation of $\mathcal{P}$ is a simplicial mesh $\mathcal{T}$ of conv($\mathcal{P}$) which satisfies the empty sphere criterion: for every simplex $K \in \mathcal{T}$, the open circumscribed sphere to K contains no point of $\mathcal{P}$.*

Each set of points $\mathcal{P}$ enjoys at least one Delaunay triangulation, which is moreover 'essentially unique' - it is actually unique when no $d+2$ points of $\mathcal{P}$ lie on a common d-dimensional sphere; if the converse holds, simple transformations allow to pass from one Delaunay triangulation of $\mathcal{P}$ to another [151]. Thus, committing a small abuse in terminology, we will sometimes talk about *the* Delaunay triangulation of a set of points.

The Delaunay triangulation of a set of points $\mathcal{P}$ is especially interesting since it can be considered as the 'best' triangulation of $\mathcal{P}$ from various standpoints, as examplified by the following proposition.

1. In the literature, the terminology 'triangulation' is widely used in this context, irregardless of the space dimension. Other denominations can be found, e.g. that of *tetrahedralization* (in three dimensions), or *simplicial decomposition*.

Proposition 3.1. *Let $\mathcal{P} = \{p_i\}_{i=1,...,N}$ a finite set of points in $\mathbb{R}^d$.*

1. *(In the particular case when $d = 2$) For any triangulation $\mathcal{T}$ of $\mathcal{P}$, denote as $a(\mathcal{T})$ the minimum angle of a triangle of $\mathcal{T}$. Then, among all the triangulations of $\mathcal{P}$, there is a Delaunay triangulation that maximizes $a(\mathcal{T})$.*
2. *(In the particular case when $d = 2$) For any vector $F = (f_1, ..., f_N) \in \mathbb{R}^N$, and any triangulation $\mathcal{T}$ of $\mathcal{P}$, denote as $\pi_{\mathcal{T}}(F)$ the $\mathbb{P}^1$ Lagrange finite element interpolate of F over $\mathcal{T}$, that is the unique such function which fulfills:*
$$\forall n = 1, ..., N, \ \pi_{\mathcal{T}}(F)(p_n) = f_n.$$
Then, for any F, any Delaunay triangulation of $\mathcal{P}$ minimizes the roughness criterion, defined as:
$$r(\mathcal{T}, F) = |\pi_{\mathcal{T}}(F)|_{H^1(conv(\mathcal{P}))},$$
where $|\cdot|_{H^1(conv(\mathcal{P}))}$ stands for the H^1 Sobolev semi-norm.
3. *For any $d \geq 1$, let $f : \mathbb{R}^d \to \mathbb{R}$ be any quadratic function. For any integer $1 \leq p \leq \infty$, the L^p interpolation error $e_p(\mathcal{T})$ of f over a triangulation $\mathcal{T}$ of $\mathcal{P}$,*
$$e_p(\mathcal{T}) = ||f - \pi_{\mathcal{T}}(f)||_{L^p(conv(\mathcal{P}))},$$
where $\pi_{\mathcal{T}}(f)$ is the $\mathbb{P}^1$ Lagrange finite element interpolate of f over $\mathcal{T}$, reaches a minimal value over the set of all the triangulations of $\mathcal{P}$ at a Delaunay triangulation of $\mathcal{P}$.
4. *For any $d \geq 1$, any real $c > 0$, let $\mathcal{C}^{2,c}(\mathbb{R}^d, \mathbb{R})$ the set of $\mathcal{C}^2$ scalar functions over $\mathbb{R}^d$, whose Hessian matrix's spectral radius is uniformly bounded by c. Define the worst-case interpolation error of a triangulation $\mathcal{T}$ of $\mathcal{P}$ as:*
$$wce(\mathcal{T}) = \max_{f \in \mathcal{C}^{2,c}(\mathbb{R}^d, \mathbb{R})} ||f - \pi_{\mathcal{T}}(f)||_{L^\infty(conv(\mathcal{P}))}.$$
Function wce(.) reaches a minimal value over the set of all the triangulations of $\mathcal{P}$ at a Delaunay triangulation of $\mathcal{P}$.

As for proofs, see [280] for points (1) and (4). Property (2) can be found in [266], and (3) was originally announced and proved in two dimensions in [267], then extended to the general case in [225] (with a completely different proof).

Properties (1) and (2) are probably the most relevant, as far as the optimality of a Delaunay triangulation of $\mathcal{P}$ is concerned. While the first one speaks for itself, the second one accounts for the fact that, if we are in search of a triangulation $\mathcal{T}$ of $\mathcal{P}$ to interpolate linearly any data vector $F \in \mathbb{R}^N$ attached to $\mathcal{P}$, a Delaunay triangulation of $\mathcal{P}$ conducts to the 'smoothest' possible graph. Unfortunately, both properties fail in three dimensions, which is an expression of the fact that the Delaunay triangulation is not as 'good' in three dimensions as it is in two dimensions. It is actually likely to contain particularly ill-shaped elements of a particular type, that of the so-called *slivers* (see figure 3.5).

Delaunay-based mesh generation methods classically proceed within two main steps.

3.2.1.1.2 Step 1: generation of the Delaunay triangulation of a set of points This first step is aimed at producing the Delaunay triangulation of the set $\mathcal{P} = \{p_n\}_{n=1,...,N}$ of vertices of the input surface triangulation $\mathcal{S}$. We mainly focus on *Bowyer-Watson*'s incremental approach (sometimes referred to as the *Delaunay kernel*), according to which the points of $\mathcal{P}$ are iteratively inserted, so that a sequence $\mathcal{T}_n$, $n = 1, ..., N$ of Delaunay triangulations is produced, with the property that each triangulation $\mathcal{T}_n$ contains $p_1, ..., p_n$

A usual trick consists in generating actually the Delaunay triangulation of the augmented set of vertices $\widetilde{\mathcal{P}} = \mathcal{P} \cup \mathcal{C}$, where $\mathcal{C}$ is the set of vertices of a large cube D (for simplicity), which encloses all the points of $\mathcal{P}$.

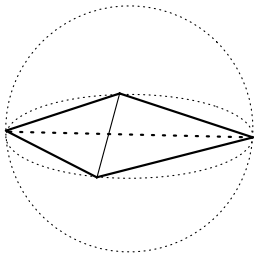


Figure 3.5: A *sliver* (in bold) is an almost flat tetrahedron, whose edges are all of acceptable lengths, and whose circumradius is not excessively large with respect to its dimensions.

The main benefit of this operation is that, starting from a Delaunay triangulation $\mathcal{T}_0$ of D (which is easily produced), inserting each point p_n, $n = 1, ..., N$, in the mesh is eased by the fact that p_n always belongs to one of the simplices of the previous triangulation $\mathcal{T}_{n-1}$.

More accurately, the algorithm operates as follows (see figure 3.6):

- **Initialization:** the mesh $\mathcal{T}_0$ of D is a Delaunay triangulation composed of five tetrahedra.
- **For** $n = 1, ..., N$**,**
  1. Find a simplex $K \in \mathcal{T}_{n-1}$ which contains p_n. Existence of such an element is guaranteed since $p_n \in conv(\{p_1, ..., p_{n-1}\}) = D$.
  2. From K, travel $\mathcal{T}_{n-1}$ by adjacency and build the *cavity* $\mathcal{C}_{p_n}$ of p_n, defined as the set of simplices of $\mathcal{T}_{n-1}$ whose open circumsphere contains p_n. It can be shown that $\mathcal{C}_{p_n}$ is a star-shaped polyhedron with respect to p_n.
  3. Delete all the elements of $\mathcal{C}_{p_n}$ in $\mathcal{T}_{n-1}$, and add the elements of the *ball* $\mathcal{B}_{p_n}$, defined as the simplices formed by joining p_n to the external faces of the cavity $\mathcal{C}_{p_n}$. It can be shown that the resulting triangulation is Delaunay.

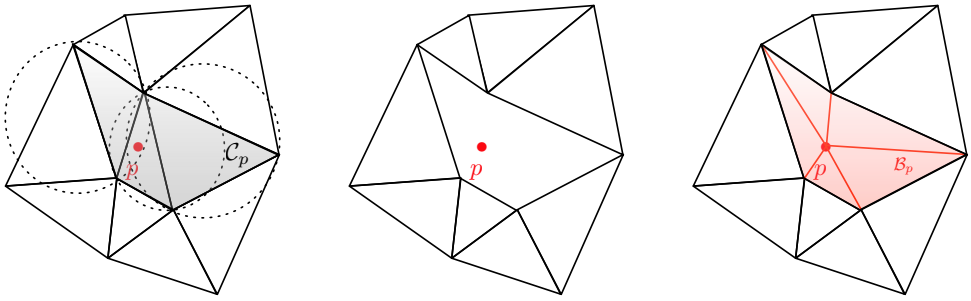
This procedure is often summed up with the schematic equation:

$$\mathcal{T}_n = \mathcal{T}_{n-1} - \mathcal{C}_{p_n} + \mathcal{B}_{p_n}.$$

At this point, it is worth mentioning a numerical difficulty attached to the construction of the cavity $\mathcal{C}_{p_n}$ of the points $p_n \in \mathcal{P}$ (step (2) in the previous algorithm). Whereas a theoretical result guarantees that $\mathcal{C}_{p_n}$ is star-shaped, it may not not necessarily be the case in practice, due to round-off errors. Hence, a correction procedure for the cavity has to be implemented, to assess that it is indeed star-shaped, and repair it if need be; see [151], chap. §2.6.3 for details. We shall come back later to this issue, in a context where the Bowyer-Watson's incremental procedure is used to insert vertices in a mesh which does not comply with the Delaunay requirements (and the cavity is not necessarily star-shaped with respect to the point).

For the sake of completeness, let us eventually mention the other two classical approaches to create the Delaunay triangulation of points:

- *Lawson's algorithm* (also known as the *flipping algorithm*), illustrated on figure 3.7, is a very elegant means for generating the Delaunay triangulation of a set of vertices, which is unfortunately restrained to the two-dimensional case (however a partial extension holds to the three-dimensional case [184]). It relies on the fact that, in two dimensions, any triangulation of conv($\mathcal{P}$) can be transformed into a Delaunay triangulation using only edge swaps, on account of the two following facts:

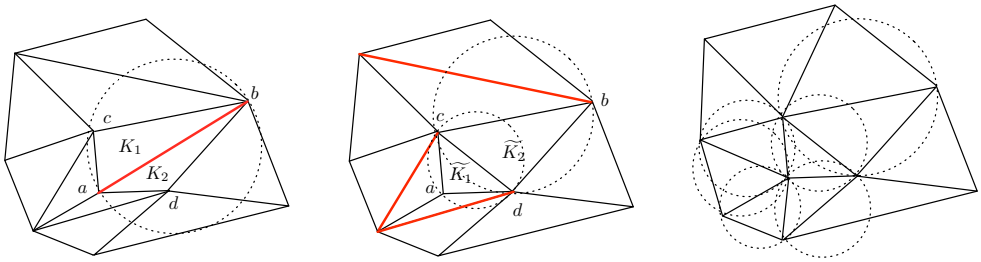


(a) (b) (c)

Figure 3.6: Insertion of a point p in a Delaunay triangulation using Bowyer-Watson's procedure; *(a):* identification of the polygonal cavity $\mathcal{C}_p$ of point p in the current triangulation $\mathcal{T}_n$; *(b):* removal of $\mathcal{C}_p$, and *(c):* reconnection of p with the vertices of $\mathcal{C}_p$: the ball $\mathcal{B}_p$ of p is introduced.

– The *Delaunay lemma* states that the (global) Delaunay property of a triangulation of $\mathcal{P}$ can be checked locally: a triangulation $\mathcal{T}$ of $\mathcal{P}$ fulfills the Delaunay criterion if and only if, for any edge ab of $\mathcal{T}$ shared by two simplices $K_1 = abc$ and $K_2 = abd$, the open circumsphere to K_1 does not contain the fourth vertex d of the configuration. This property actually holds in any dimension.
– The following alternative is only available in two dimensions: for any edge $e = ab$ shared by two simplices $K_1 = abc$ and $K_2 = abd$, either the local Delaunay criterion is satisfied, or the configuration can be swapped: edge e and the two triangles K_1 and K_2 are destroyed, and they are replaced by the alternate configuration, consisting in the diagonal edge $\widetilde{e} = cd$, shared by triangles $\widetilde{K_1} = acd$ and $\widetilde{K_2} = bcd$, the latter configuration satisfying the local Delaunay criterion.

Figure 3.7 shows an example of Lawson's algorithm in motion.



(a) (b) (c)

Figure 3.7: Lawson's algorithm in progress; *(a)* the two triangles sharing the red edge do not satisfy the local Delaunay criterion (the circumcircle of T_1 appears in dotted line), and the edge must be swapped; *(b)* the resulting two triangles from the previous swap now satisfy the local Delaunay criterion, but the configurations near the red edges still do not; *(c)* resulting mesh: all the edges satisfy the local Delaunay criterion, and the global mesh fulfills the Delaunay property.

– The weird *parabolic lifting algorithm* exploits the connection between Delaunay triangulations and convex hulls: roughly speaking, the Delaunay triangulation of a set of vertices $\mathcal{P} = \{p_n\}_{n\in\mathbb{N}} \subset \mathbb{R}^d$ can be seen as the projection onto $\mathbb{R}^d$ of the $(d+1)$-dimensional lower convex hull of the 'lifted set' $\mathcal{P}^+ = \{p_n^+\}_{n\in\mathbb{N}} \subset \mathbb{R}^d$, obtained by projecting the points p_n onto a parabola:

$$\forall n = 1, ..., N,\ \ p_n^+ = (p_n, |p_n|^2).$$

Although this point of view is very appealing - and of great use in theoretical studies, few concrete algorithms based on this property, devoted to generating the Delaunay triangulation of a set of vertices are known (see however [48]).

3.2.1.1.3 Step 2: enforcement of the entities of Σ in the resulting mesh The first step ends with a mesh $\widetilde{\mathcal{T}}$ of D, whose vertices are exactly those of $\mathcal{S}$ (plus the corners of D). Unfortunately, unless Ω is convex, this does not imply that the higher order entities (edges, or faces) of $\mathcal{S}$ appear in $\widetilde{\mathcal{T}}$, and no mesh of Ω can thus be obtained.

Some special treatment has then to be applied to $\widetilde{\mathcal{T}}$, to modify it into a new mesh $\mathcal{T}$ of D, in which all the entities of $\mathcal{S}$ explicitly appear, so that a mesh of Ω exists as a submesh of $\mathcal{T}$, in the sense of definition 3.3. All the usual methods to achieve this purpose rely on the notion of *Steiner point*:

Definition 3.9. *In the context of Delaunay-based mesh generation, a Steiner point is a vertex that has been inserted in the resulting mesh $\mathcal{T}$ of D, which is not a vertex of Σ (neither of D), and has been inserted in order to help the enforcement process of the entities of $\mathcal{S}$ into $\mathcal{T}$.*

Henceforth, the problem of enforcing the entities of $\mathcal{S}$ into $\widetilde{\mathcal{T}}$ can take various context-dependent forms:

- One could require the entities of $\mathcal{S}$ to appear *exactly* in the final mesh $\mathcal{T}$, i.e. the edges of $\mathcal{S}$ are edges of $\mathcal{T}$, and similarly for faces. In the literature, this issue is referred to as the *boundary integrity constraint*. It is a crucial feature in cases when, for instance, the domain to mesh Ω is a subdomain of a larger domain, and its boundary $\mathcal{S}$ is shared by other subdomains.
- On the other hand, one could only ask the entities of $\mathcal{S}$ to appear *weakly* in $\mathcal{T}$, i.e. as a union of entities of $\mathcal{T}$ (e.g. an edge of $\mathcal{S}$ exists as a union of edges of $\mathcal{T}$). This constraint is of course more permissive than the former one.

Whatever the retained acceptation, as for the boundary enforcement constraint, several different strategies can be followed to tackle the problem of enforcing the entities of $\mathcal{S}$ into $\mathcal{T}$:

- A first class of methods - see for instance [235] for a three-dimensional work - cling to getting a Delaunay triangulation $\mathcal{T}$ of D, come what may. The entities of $\mathcal{S}$ are enforced in $\mathcal{T}$ by wisely adding Steiner points to the set of inserted points, in such a way that *any* Delaunay triangulation of this new set of points contains the entities of $\mathcal{S}$ (either exactly or weakly). The resulting mesh $\mathcal{T}$ sometimes bears the name of *conforming Delaunay triangulation* of Ω in the literature. The process of augmenting $\mathcal{P}$ with adequate Steiner points can be thought of *a priori*: in [250], an algorithm is presented which refines $\mathcal{S}$ into a surface triangulation $\widetilde{\mathcal{S}}$ which is *Delaunay admissible*, in the sense that any Delaunay triangulation of the set of vertices of $\widetilde{\mathcal{S}}$ contains exactly the entities of $\widetilde{\mathcal{S}}$. Unfortunately, the refined surface $\widetilde{\mathcal{S}}$ may prove very ill-shaped, due to excessive refinement.
  Generally speaking, aiming at getting a conforming Delaunay triangulation of Ω is a strong requirement, for a lot of Steiner points may have to be inserted to this end. In [47], examples are provided of two-dimensional boundaries $\mathcal{S}$, enjoying m vertices and n edges whose conforming Delaunay triangulations must have at least $\mathcal{O}(mn)$ vertices.
- Following the lead of [85], several authors have proposed to enforce the entities of $\mathcal{S}$ into $\widetilde{\mathcal{T}}$ in the sense of *constrained Delaunay triangulations*. Grossly speaking, a constrained Delaunay triangulation behaves 'almost-everywhere' like a Delaunay triangulation, except for some particular *constrained entities*, that are precisely those we need to enforce. Constrained Delaunay triangulations (which are not Delaunay triangulations, properly speaking) retain most of the good properties of Delaunay triangulations (see [280] for an overview). Three-dimensional algorithms exist to produce a constrained Delaunay triangulation, which contains the entities of $\mathcal{S}$ under weak form. Although no theoretical guarantee assesses that building such constrained Delaunay triangulations requires much fewer Steiner points than their conforming Delaunay triangulations counterparts, this is reported true in numerical practice [281]. This method is used for instance in the work [284], and is complemented by a heuristic

procedure to remove the Steiner points introduced on the entities of $\mathcal{S}$ in the case when these entities are expected to exist exactly in $\mathcal{T}$.

– Eventually, some authors propose to completely drop the Delaunay criterion, and enforce the entities of the boundary $\mathcal{S}$ in $\widetilde{\mathcal{T}}$ using local topological mesh operators, namely *edge swaps* (in combination with a procedure for adding Steiner points to the mesh). In [154], the authors prove that, in two dimensions, all the boundary edges of $\mathcal{S}$ can be exactly enforced in $\widetilde{\mathcal{T}}$ without adding any Steiner point, and an associated numerical procedure based on edge swaps is described (see figure 3.8).

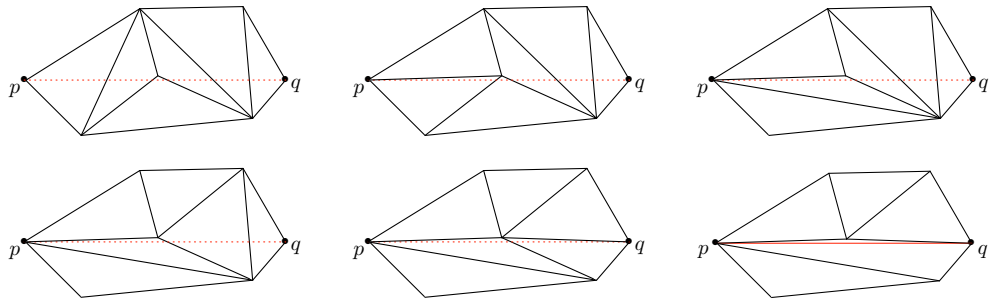


Figure 3.8: Enforcement of the missing edge pq in a mesh, using only edge swaps.

The corresponding three-dimensional procedure is more involved. In [154], a robust algorithm is described for enforcing exactly the entities of $\mathcal{S}$ into $\widetilde{\mathcal{T}}$. This methods relies heavily on edge swaps (see section 3.3.1 for a presentation of this operator), the use of which is enabled by the insertion of several Steiner points. This study is complemented by the work [153], which proposes another method, allowing the insertion of Steiner points on the entities of $\mathcal{S}$, provided their removal in the end of the process is possible. This last method makes it possible to tackle the few cases in which the former fails because of round-off errors, making it impossible to add Steiner points.

Remark 3.2. Because the enforced surface $\mathcal{S}$ is assumed to be the boundary of a polyhedral domain, it is implicitly assumed to be a manifold surface, in particular, it is not self-intersecting (and must be so, for enforcing $\mathcal{S}$ in $\widetilde{\mathcal{T}}$ would prove impossible, should the converse hold). Interestingly enough, these methods allow to detect whether a given surface triangulation is self-intersecting, which is far from a trivial problem.

3.2.1.1.4 Some post-processing issues The boundary enforcement procedure of section 3.2.1.1.3 ends with a mesh $\mathcal{T}$ of D (whether be it a Delaunay triangulation or not), in which the surface mesh $\mathcal{S}$ has been enforced (in whichever sense). Two additional operations are classically performed:

– Strictly speaking, the obtained mesh at this point is not yet a mesh of Ω, but rather a mesh of the larger box D, a submesh of which is a mesh of Ω. To recover a 'true' mesh of Ω, one resorts to a *coloring algorithm* to remove all the 'exterior' tetrahedra of $D \setminus \Omega$ from $\mathcal{T}$: in a nutshell, it consists in starting from one well-identified exterior element $K \in \mathcal{T}$ (e.g. one containing a corner of D as a vertex), then traveling $\mathcal{T}$ from K by adjacency, passing from on element to the other through the triangular faces that are not faces of $\mathcal{S}$. The obtained component is the exterior component $D \setminus \Omega$ and should be removed (see [145] §7.3.4 for further technical details, especially in the case that several connected components of Ω are interlocked) . The underlying idea to this technique is adapted into an algorithm for signing the unsigned distance function to a contour in chapter 6, section 6.4.2.

– The resulting mesh of Ω - still denoted as $\mathcal{T}$ - is bound to enjoy very few internal vertices, those being the Steiner points which have been inserted only with the aim to ease the meshing process, regardless of any element quality criterion. For this reason at least, a Delaunay meshing algorithm is always supplemented with a phase during which internal vertices are inserted, with the ambition to improve the (probably very low) quality of the mesh (see [54] or [145] §7.3.5). As this step is only a component

of the more general issue of *mesh optimization*, which is discussed in section 3.3, we shall not go any further on this topic for the moment.

Remark 3.3. This whole procedure can be extended almost mutatis mutandis (in theory at least) to an anisotropic mesh generation method. Indeed, using the framework of size and orientation prescription through Riemannian metrics sketched in section 3.1.3, only the notions relative to distance should be adapted in the previous construction (which actually confines to adapting the Delaunay criterion of definition 3.8). See [54, 152] for more details.

3.2.1.2 Advancing front methods for mesh generation

Since their early inception [138, 148], advancing front methods have been the most intuitive ones in mesh generation: from a surface triangulation $\mathcal{S}$ of its boundary, the domain to mesh Ω is filled with tetrahedra which are constructed one following the other.

Concretely, a typical advancing front algorithm maintains a *front* $\mathcal{F}$, that is, a list of triangular faces separating the already meshed region from the one yet to be filled. At each stage of the process, a single face of the front is considered, based on which the construction of a new tetrahedron is considered. The process unfolds as follows (see figure 3.9):

- **Initialization:** the mesh $\mathcal{T}$ of Ω does not contain any tetrahedron and the front $\mathcal{F}$ is created as the set of faces of $\mathcal{S}$.
- **While $\mathcal{F}$ is non empty,**
  1. Select a triangular face $T = abc \in \mathcal{F}$,
  2. search for an 'optimal' position for the fourth vertex $\widetilde{p}$ of a tetrahedron $K = abc\widetilde{p}$ based on T, either as an already existing point in $\mathcal{T}$, or as a new point to be added.
  3. Assess the validity of the resulting mesh from the addition of $\widetilde{p}$ (if need be) and K to $\mathcal{T}$. In particular, it should be checked that K does not overlap other tetrahedra of the mesh. Doing so may require one or several travels back and forth to step (2), until an admissible proposal p as for the fourth vertex of a new tetrahedron is reached.
  4. Update $\mathcal{T}$, and the front $\mathcal{F}$: T is removed, and the faces of K which do not already belong to another tetrahedron of the mesh are added.

Such a strategy raises several issues, that are tackled in different ways from one implementation to the other (see [145], chap. 6 for a far more exhaustive description):

- Each iteration of the algorithm begins with the choice of a face $T \in \mathcal{F}$, from which a new tetrahedron is built. The device of an efficient selection strategy of the faces of the front that should be processed in priority proves crucial in practice, since it greatly influences the 'nice behavior' of the front evolution, and the convergence of the whole method. Several choices are available in this regard: the most common one [252] consists in systematically choosing the smallest face (with respect to some measure), so that too large elements do not hinder prematurely the creation of smaller ones; other authors advocate to deal in priority with regions that are particularly pinched. More advanced strategies exist to ease the convergence of the method, which involve several criteria, and notably considerations about the quality of the elements to be created [264]. A structure of priority queue is often used for describing $\mathcal{F}$.
- Assuming that a face $T = abc \in \mathcal{F}$ has been chosen, what is the criterion for computing an optimal point $\widetilde{p}$ for forming the new tetrahedron $K = abc\widetilde{p}$? Once again, each implementation comes along with its own strategy, but common features can be outlined. While some variants of the method [138, 264] propose to create all the internal vertices in a first stage, then to find the optimal connections using the advancing front strategy, most of the implementations attempt to create new vertices, then to connect them on the fly: a size map is provided (defined on a background *control space*, which is often defined as a Cartesian grid, but may be more 'exotic' - e.g. a Delaunay triangulation in [155]), and a

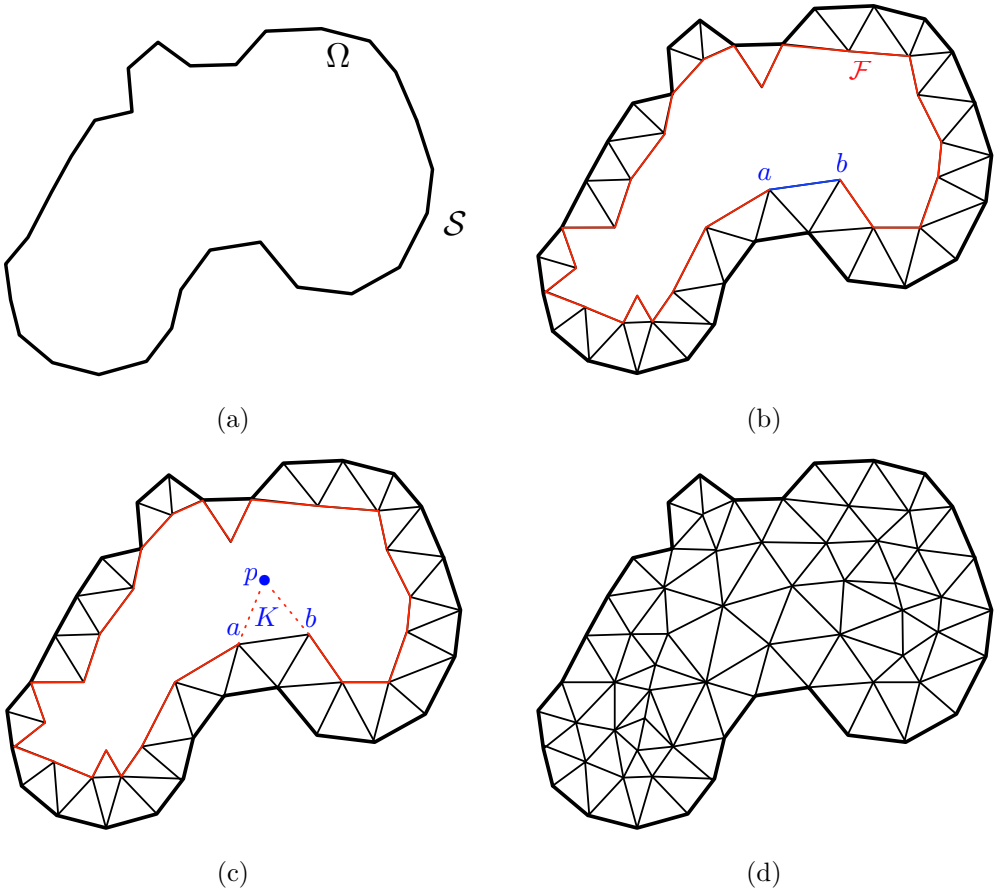


Figure 3.9: Mesh generation using an advancing front algorithm in two dimensions. *(a)* Initialization: $\mathcal{T}$ is empty, and $\mathcal{F}$ contains the edges of Σ; *(b)* at an intermediate step, the front is composed of the colored edges, and the blue edge ab is selected for the next triangle creation; *(c)* an optimal position for a new point p is proposed, and the associated triangle $K = abp$ is created; *(d)* final result of the algorithm.

provisional optimal position $\widetilde{p}$ is computed, which takes into account the local size feature, as well as the quality of the element to be created). This position is then compared to that of 'close points' from $\widetilde{p}$: if a point $q \in \mathcal{T}$ is 'close enough' to $\widetilde{p}$, this position is changed to be q (to avoid very acute 'nest').

– Eventually, the validity of the addition of K has to be tested. K is required not to overlap an already existing element; hence, intersections between K and these elements must be tested in an efficient way. This generally involves the definition of a *neighborhood space*, e.g. a background Cartesian grid (which may differ from the one used for storing the size map), which allows to speed up the tetrahedron to tetrahedron intersection tests, allowing for coarse and very fast rejection tests. An efficient implementation of these two last points is substantially eased by the use of efficient data structures that are detailed in [207].

Advancing front methods enjoy very different assets from Delaunay-based methods:

– The quality of elements lying close to the boundary $\mathcal{S}$ is very high, since as long as two independent parts of the front do not meet (which generally occurs 'far' from the boundary), almost nothing can impede the creation of new elements in an optimal way.

– In the same vein, advancing front techniques prove especially convenient when it comes to generating

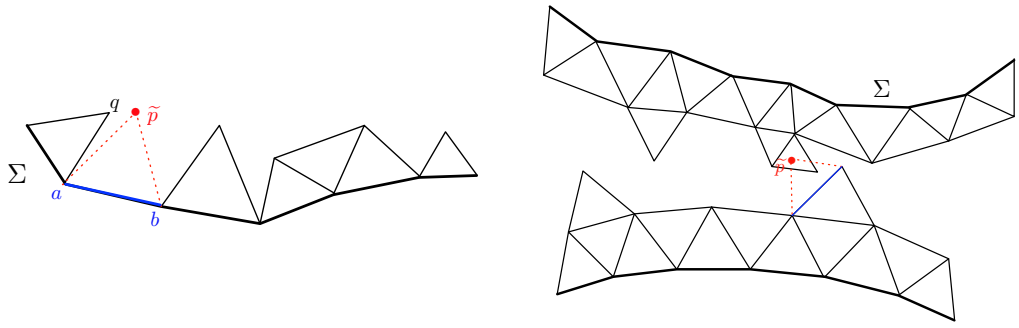


Figure 3.10: *(Left)* $\widetilde{p}$ is the computed optimal position for the third vertex of the triangle built from edge ab, but q is preferred, since creating $ab\widetilde{p}$ would cause difficulties in generating good quality triangles in the next steps; *(right)* the proposed point $\widetilde{p}$ overlaps another triangle of the mesh.

anisotropic meshes, or boundary layer meshes. These concerns were at stake in the early stages of the techniques (see the work [252] for instance).
- The integrity of the surface mesh $\mathcal{S}$, which is desirable in several applications, and is the difficult feature to preserve in Delaunay based methods is automatic.

Unfortunately, advancing front methods are undermined by severe drawbacks, the most critical being:
- they are prone to lack efficiency if not properly implemented (see the previous discussion about the search and intersection tests).
- Even more bothersome is the absence of any theoretical guarantee that the process will successfully converge, especially when it comes to merging colliding parts of the front. To achieve convergence, advancing front algorithms have no option but to rely on heuristics - for instance, a strategy for destroying configurations that are deemed to hamper convergence, which keeps a historical record of the operations held is described in [264] (elaborating on works referenced therein).

Remark 3.4. Delaunay based and advancing front methods enjoy very different features, and several mixed approaches have been thought of, which benefit from their respective assets. In this spirit, in [144], the authors propose to construct the constrained Delaunay triangulation of $\mathcal{S}$, with the smallest number of internal points possible, then to define a front, on account of a quality criterion (the ill-shaped elements, to be removed are part of the front), infer optimal positions for new points, to be connected to faces of the front, then insert the points using the Delaunay procedure described above. In [220], an advancing front strategy is presented, in which points are incrementally inserted in the mesh, using a variation in the Delaunay procedure.

We eventually turn to a rather different problem, that of mesh generation for implicitly-defined domains.

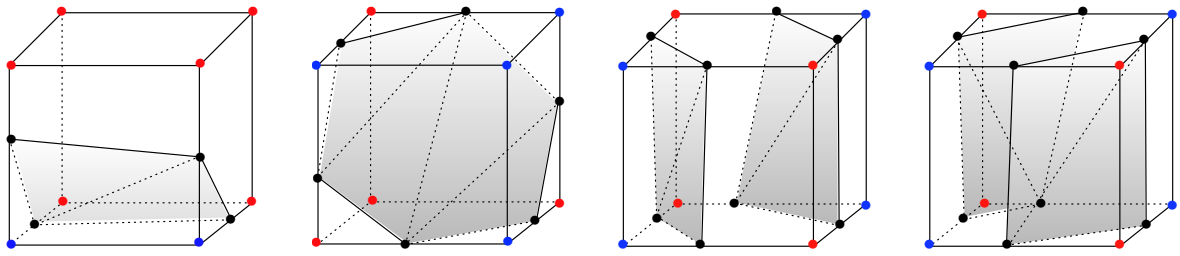
3.2.1.3 Meshing implicit domains

Over the last decades, handling surfaces or domains in an implicit way has become increasingly popular. In biomedical engineering, surfaces or domains of interest (tissues, bones, etc...) are indeed often characterized as regions where one or several measurable quantities are equal to, or lower or equal than a known physiological threshold. For instance, Computed Tomography (CT) techniques measure a relaxation number of the intensity of X-rays spread into the human body; since air, bone, water have different behaviors with respect to this number, these entities can be separately observed by looking at the level sets of the relaxation. A similar philosophy presides over MRI and PECT techniques. Rather differently, we have seen in chapter 1 that a more and more convenient framework for addressing free and moving boundary problems is that of the level set method, which features implicitly-defined domains. Hence, obtaining a mesh of the resulting

domain of such an evolution process naturally involves a mesh generation process for implicit geometries (see for instance section 3.4.2, and Chapter 8).

The most intuitive way to get a mesh of an implicitly-defined domain is possibly the famous *Marching Cubes* algorithm [208] (or one of its numerous variants), which we briefly describe now. Let $D \subset \mathbb{R}^3$ a computational box, equipped with a uniform grid of $n_x \times n_y \times n_z$ nodes, denoted as $\{x_{ijk}\}_{1\leq i\leq n_x,\, 1\leq j\leq n_y\, 1\leq k\leq n_z}$ (see [49, 283] for space adaptive versions of the algorithm).

Let ϕ be a scalar quantity, known as the discrete set $\{\phi_{ijk}\}_{1\leq i\leq n_x,\, 1\leq j\leq n_y\, 1\leq k\leq n_z}$ of its values at the nodes of the grid. This set will also be referred to as ϕ for the sake of simplicity. We aim at meshing the negative subdomain Ω, and 0 isosurface $\Gamma = \partial\Omega$ of ϕ (in the language of Chapter 1, ϕ is a level set function associated to Ω). To achieve this, each grid cell Q where the sign of ϕ changes is processed independently from the others. The intersections of Γ with the edges of Q are computed by assuming e.g. a linear variation of ϕ along them. The algorithm then relies on templates for inferring the corresponding piece of triangulated surface $\Gamma \cap Q$ from those data. A smart use of the symmetries between the $2^8 = 256$ possible configurations as regards the signs of ϕ at the vertices of Q allows to bring their number down to 15; see figure 3.11 for examples.



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

Figure 3.11: *(a-b)* Two patterns for the marching cube algorithm. The red (resp. blue) nodes are associated to positive (resp. negative) values of ϕ, and the reconstructed isosurface is greyed; *(c-d)* two possible patterns associated to the same configuration for the sign of ϕ, leading to an ambiguity in the marching cubes algorithm.

Unfortunately, this alone is not enough to guarantee a 'fine' construction of Γ. Indeed, the shape of the portion $\Sigma \cap Q$ of Γ enclosed in Q is not uniquely determined by the intersections of Γ with the edges of Q (see figure 3.11, (c,d)). If no particular attention is paid, the reconstructed surface Γ may show 'cracks', or 'holes'. Several strategies exist in the literature to alleviate these ambiguities:

- A first approach consists in inventing a smooth piece of surface inside Q from the values of ϕ at its vertices (or approximated higher-order information), which could be deemed as representative of the behavior of Γ, then in deciding accordingly as for the 'correct' topology for $\Gamma \cap Q$. A very simple and efficient approach relying on this philosophy is presented in [219]: $\Gamma \cap Q$ is approximated as the 0 level set of the $\mathbb{Q}^1$ Lagrange finite element interpolate $\pi_Q(\phi)$ of ϕ over Q, that is, the unique $\mathbb{Q}^1$ function $\pi_Q(\phi)$ such that $\pi_Q(\phi)(x) = \phi(x)$ for each vertex x of Q. A close study of some elementary properties of such surfaces allows then to decide in a rigorous way as for the 'good' topology to retain for $\Gamma \cap Q$.
- Many authors [49, 141] propose on the contrary to capture the 'correct' behavior of Γ inside a grid cell Q by subdividing Q into twelve tetrahedra. This calls for the definition of a new, artificial vertex x_0 at the centre of Q, and of a consistent value for ϕ at x_0 (this can be done e.g. by using $\mathbb{Q}^1$ interpolation inside Q). Then, for each thus obtained tetrahedron K, the portion of surface $\Gamma \cap K$ is approximated by using a $\mathbb{P}^1$ linear approximation of ϕ inside K, without any possible ambiguity.

This last point certainly deserves further comments. The variant of the marching cubes algorithm unfolding on a tetrahedral computational support - which is sometimes referred to as the *marching tetrahedra*

algorithm [117] - turns out more convenient than its Cartesian counterpart, given that it lends itself to a natural and unambiguous reconstruction of Γ. On a different note, the use of a simplicial computational support allows to use mesh adaptivity. In this respect, see the interesting work [197] for a (possibly adaptive) marching tetrahedra algorithm with provable bounds on the quality of the final mesh.

Note that, in the first place, the method described above is rather devoted to an implicitly-defined *surface* Γ than to a domain Ω - this is actually the context of its original introduction. However, an easy modification of the templates for reconstructing the piece of surface lying in a grid cell Q from its intersection with the edges of Q enables the construction of a tetrahedral mesh of the implicit domain Ω [141].

Before putting an end to this section, we should mention the mesh generation method for implicitly-defined domains introduced by Persson in [253, 254], as an interesting alternative to the marching cubes algorithm. Keeping the notations of the previous paragraphs, this method requires the function ϕ to be the *signed distance function* to the domain $\Omega \subset \mathbb{R}^d$, which may be supplied analytically, or defined on a background grid. The method proceeds along the lines of the following scheme (see figure 3.12 for an illustration obtained using the `Matlab` code available on P.-O. Persson's webpage [2]:

1. At first, a set $\mathcal{P}$ of points is spread within Ω, with a density related to a prescribed size function h (also defined on a background grid). Note that whether a point $x \in \mathbb{R}^d$ belongs to Ω or ${}^c\Omega$ can be easily tested by merely looking at the sign of $\phi(x)$.
2. The Delaunay triangulation of $\mathrm{conv}(\mathcal{P})$ is generated, and each simplex whose circumcenter lies outside Ω is removed from the mesh. This phase produces an intermediate simplicial mesh $\widetilde{\mathcal{T}}$, whose vertices are exactly the elements of $\mathcal{P}$.
3. The vertices of $\widetilde{\mathcal{T}}$ are relocated with the purpose to improve the mesh quality. To this end, an analogy between edges of a mesh and bars of a truss is used. Besides, the vertices of the boundary of $\widetilde{\mathcal{T}}$ are projected to the exact boundary $\partial\Omega$, by making good use of the analytical expression of the projection operator $p_{\partial\Omega} : \mathbb{R}^d \to \partial\Omega$ in terms of the signed distance function to Ω (see Chapter 4).

Note that stages (2), (3) may have to be repeated several times until a fine mesh $\mathcal{T}$ of Ω is eventually reached.

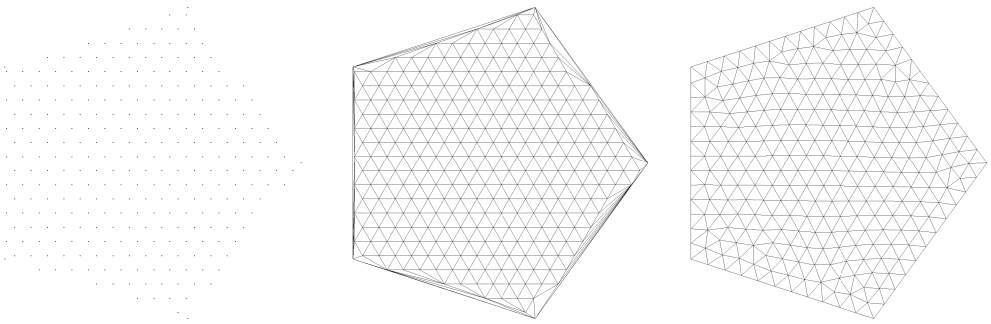


Figure 3.12: Illustration of Persson's method; *(left)* initial set of points $\mathcal{P}$, *(middle)* Delaunay triangulation of $\mathrm{conv}(\mathcal{P})$, and *(right)* final mesh after node relocations.

Remark 3.5. This brief and biased summary of some of the most famous methods for volume mesh generation should definitely not be considered as exhaustive. For example, we did not even made mention to the class of *quadtree* (or *octree* in three space dimensions) *methods* [279], which propose to generate a mesh of a domain supplied by means of a description of its surface, relying on a philosophy which shares a lot of features with the marching cubes method.

2. http://persson.berkeley.edu/

3.2.2 Surface mesh generation

Surface meshing is quite a peculiar topic; while having a lot in common with two-dimensional volume meshing (the elements considered being mere triangles), it also inherently exhibits tenuous and specific connections with differential geometry of surfaces.

Let $\Gamma \subset \mathbb{R}^3$ be a surface; a surface mesh $\mathcal{S}$ of Γ should fulfill two independent functions:

1. $\mathcal{S}$ should consist of well-shaped triangles, whose sizes are adapted to a user-defined prescription.
2. $\mathcal{S}$ should be a 'close geometric approximation' of the original surface Γ.

Point (2) is obviously the original requirement to the setting of surface meshing, and it could be given several different meanings, depending on the considered application; for instance, $\mathcal{S}$ could be expected to be close to Γ in terms of Hausdorff distance, or in terms of their first-order geometric behaviors, i.e their normal vector fields could be asked to be close from one another, so that the surface mesh does not show parasitic 'folds'. See [143] for a more exhaustive discussion around this topic, which we will come back to in Chapter 8.

The methods available for meshing a surface Γ also strongly depend on the structure under which the related information is known. In this section, we limit ourselves to the prevailing case in numerical applications, namely that of parametrized surfaces - we already mentioned in section 3.2.1.3 how to deal with another very important class of implicit surfaces.

In sections 3.2.2.1 and 3.2.2.2 below, Γ is assumed to be a single *parametric patch*, i.e. it is described through the datum of an open domain $U \subset \mathbb{R}^2$, and of a smooth, one-to-one and onto mapping $\sigma : U \to \Gamma$ (see figure 3.13, left). In Computer-Aided Design (CAD), U is generally a very simple domain, e.g. a rectangle, and σ is a bivariate polynomial application. Section 3.2.2.3 will eventually explain how this simplified setting can be used to address the general problem of surface meshing.

Surface mesh generation methods fall into two categories:

3.2.2.1 Direct methods

Direct methods are so named because they act rather at the level of the surface Γ itself than at the one of the parametric space U. They generally imitate two-dimensional volume mesh generation methods, taking into account the fact that the object to mesh has now a non trivial geometry.

For instance, [206] proposes an extension of the advancing front method to surface mesh generation, in a slightly different context than that of this section however; the overall strategy of the advancing front method presented in section 3.2.1.2 is retained, except that new points are proposed directly on the surface. Several additional ingredients are added to overcome specific issues such as the fact that the meeting of two parts of the front is now a purely three-dimensional situation.

Delaunay based methods have also been extended to the context of surface mesh generation: in [86], an algorithm is proposed, which starts from a very coarse mesh of Γ, e.g. one connecting straightly the points of its boundary curve, then to insert iteratively vertices on the surface, using an adapted Delaunay kernel.

3.2.2.2 Mesh generation using the parameter space

Recall that, for now, Γ is a single parametric patch, described by a smooth mapping $\sigma : \mathbb{R}^2 \supset U \to \Gamma$.

The converse idea to that of section 3.2.2.1 consists in constructing first a mesh $\mathcal{T}'$ of the parameter space U (which is nothing but generating a mesh of a two-dimensional domain), then using the application σ to send this planar triangulation back to Γ: the vertices x of $\mathcal{T}'$ are relocated to their corresponding positions $\sigma(x) \in \Gamma$ without any other alteration of $\mathcal{T}'$.

This approach suffers from one main drawback: the application σ accounts for a (possibly severe) distortion, inherent to the fact that it maps a portion of the plane to a (possibly very) curved surface Γ. Consequently, even a very well-shaped mesh $\mathcal{T}'$ of U is likely to transform into a dramatically twisted - and possibly self-intersecting - mesh $\mathcal{S}$ of Γ during the final stage of the algorithm.

Of course, this problem can be alleviated if a sufficiently small mesh size is chosen as for $\mathcal{T}'$, but this size should be selected wisely, so that the resulting surface triangulation $\mathcal{S}$ does not enjoy too many elements, and is well-shaped.

The solution, as proposed in [55] (see also [145] §15.3.3 for details), converts the problem of generating a (even isotropic) mesh of Γ into that of generating an anisotropic mesh of the parameter space. More precisely, it consists in using an interpolation error estimate for the mapping σ to devise a Riemannian metric M (or a size map h if an isotropic mesh of U is preferred) on U, so that a quasi-unit mesh of U with respect to M is mapped to a well-shaped mesh of Γ. This Riemannian metric inherently encodes information about the second-order behavior of Γ, which is, as evoked above, the key feature in measuring the distortion of the mapping σ.

3.2.2.3 Extension to more complex surfaces

Assuming the considered surface Γ can be described with a single, smooth parametric patch $\sigma : U \to \Gamma$ - as we have been doing so far - seems unrealistic with respect to concrete applications:

- Even very simple surfaces - e.g. spheres (and actually all the compact submanifolds of $\mathbb{R}^3$) ! - cannot be described using a single parametric patch, for obvious topological reasons.
- Many surfaces - especially those accounting for mechanical parts - present sharp features (e.g. ridge edges, corners, etc...), as exemplified in figure 3.13, right. Describing such surfaces by means of a single *smooth* mapping σ seems then hopeless.

Actually, in concrete applications (notably those involving CAD modeling), a surface Γ is provided as a set $\{(U_i, \sigma_i)\}_{i=1,\dots,n}$ of patches similar to those described above, such that the images $V_i = \sigma_i(U_i)$ are disjoint open subsets of Γ and covering Γ: $\Gamma = \bigcup_{i=1}^n V_i$ (see figure 3.13, right). The intersections between the subsets $\overline{V_i}$ describe a web of curves $\{\Sigma_j\}_{j=1,\dots,m}$, wherein the sharp features of Γ lie.

Now, generating a mesh for Γ requires first to mesh the curves Σ_j, then to get a mesh $\mathcal{S}_i$ of each one of the patches U_i, along the lines of section 3.2.2.2, respecting the discretization of ∂U_i imposed by that of the intersecting curves Σ_j so that the final mesh $\mathcal{S} = \bigcup_{i=1}^n \mathcal{S}_i$ of Γ is conforming. See [151], §11.3 for technical details.

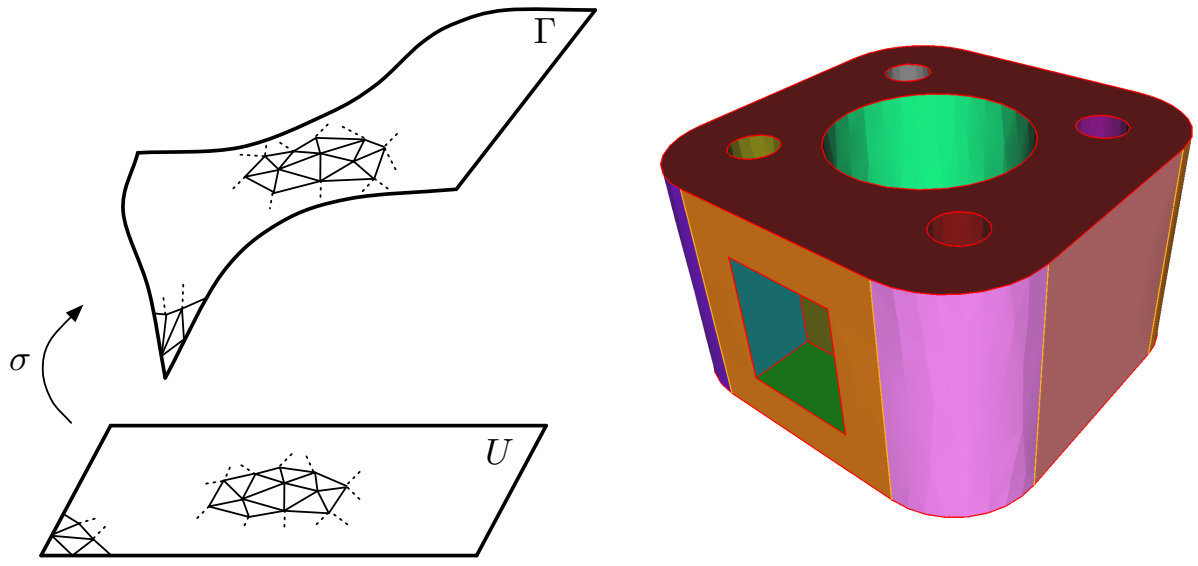


Figure 3.13: *(Left)* A smooth parametric patch; *(right)* decomposition of a surface into several smooth parametric patches along its sharp features.

3.3 Local remeshing

For various reasons, it is very usual that, in a context of numerical simulation, a mesh is not perfectly amenable for computations. For instance, we already hinted at the fact that meshes that directly result from a mesh generation procedure are likely to suffer from poor quality elements. From another angle, it may be desirable to adapt the size of the elements of a mesh $\mathcal{T}$ to a user-specified prescription, e.g. linked to an a posteriori error analysis of a previous computation held on $\mathcal{T}$.

These concerns express the need for *mesh optimization*, or *remeshing*[3] methods: let $\mathcal{T}$ a *valid* mesh (resp. valid surface mesh) of a domain $\Omega \subset \mathbb{R}^d$ (resp. a surface $\Gamma \subset \mathbb{R}^d$) in the sense of section 3.1.1 - we do not address here the case of invalid meshes, which appeals for *mesh repairing* techniques, see [58], chap. 8 - which is possibly ill-shaped, and not adapted to a given size prescription. The aim is to modify $\mathcal{T}$ (resp. $\mathcal{S}$) into a new mesh $\widetilde{\mathcal{T}}$ (resp. $\widetilde{\mathcal{S}}$) of the same domain Ω (resp. the same surface Γ), which is well-shaped and adapted to the prescribed size. Note that optimizing (or remeshing) a mesh is generally achieved as a series of *local* operations, i.e. which affect a configuration of few elements: indeed, modifying large parts of the mesh at one fell swoop would somehow boil down to a true mesh generation problem.

As in mesh generation, *volume remeshing* procedures, aimed at remeshing a three-dimensional tetrahedral mesh (or a two-dimensional triangular mesh), and *boundary remeshing* procedures, for remeshing a three-dimensional surface triangulation (or a two-dimensional mesh of a curve), should be considered separately for they enjoy very different stakes. Indeed, whereas a typical volume remeshing procedure of a volume mesh $\mathcal{T}$ involves more combinatorial analysis since the situations considered then are 'genuinely' three-dimensional, the concerns of enforcing, or preserving, a fine description of the surface accounted for by the initial mesh should be at the core of any surface remeshing method.

Once again, the issue of local volume remeshing being much more intricate in three dimensions as in two dimensions, we focus the contents of this section on the former case.

3.3.1 Volume remeshing

In all this section, let $\mathcal{T}$ be a tetrahedral mesh, whose elements' average quality may be to improve, or whose size ought to be made conform to a given size prescription (supplied e.g. under the form of a size function h, or a Riemannian metric M).

Most of the remeshing strategies discussed in the literature rely on a combination of four elementary operations. For the sake of simplicity, all the meshes obtained throughout the iterative remeshing process are still denoted as $\mathcal{T}$. Here is a very sketchy description of the mesh operators; we shall present their use in a particular implementation with more details in Chapter 8 (see also [145] for more details).

3.3.1.1 Mesh enrichment operators

Mesh $\mathcal{T}$ may have to be enriched either because some of its regions are undersampled (i.e. the number of vertices is insufficient with respect to the local feature size), or with the aim to improve the quality of the affected elements. Generally speaking, $\mathcal{T}$ is enriched by adding vertices one by one, and numerous possibilities exist as for the insertion of a new vertex into $\mathcal{T}$:

– the simplest (and most robust) way consists in introducing a new point m on a edge pq of $\mathcal{T}$: pq is split into two new edges pm and mq, and each tetrahedron K of the shell $\mathcal{S}h(pq)$ of pq is divided into two tetrahedra (see figure 3.14, left). This procedure is likely to cause very ill-shaped elements to appear in the mesh - especially in three dimensions, where tetrahedra are much more prone to degeneracy than two-dimensional triangles - if it is not controlled properly. Some strategies exist to do so: *bisection*

3. In the literature, depending on authors, *remeshing* may either refer to as the collection of methods for (iteratively) improving an input mesh, or on the contrary indicate that the input mesh is downright abandoned, and a new mesh of the corresponding domain is generated.

algorithms have been constructed, which iteratively insert the midpoint of a judiciously chosen set of simplices in the mesh (until a desired size is reached), and can be proved to produce meshes with good quality (see [205, 268]). Anyway, this operator comes in handy as an enrichment operator in a general remeshing strategy involving several operators (see the other sections, and Chapter 8).

– So as to mitigate the quick degeneracy of elements entailed by edge splitting procedures, some authors advocate to insert new points in $\mathcal{T}$ using the Delaunay kernel discussed in section 3.2.1.1.2, even if $\mathcal{T}$ is not a Delaunay triangulation [115, 152]. The reason to do so is that the Delaunay kernel alters a whole local configuration, and - except the possible appearance of slivers - leads to better-shaped elements. Of course, as the considered mesh $\mathcal{T}$ is not a Delaunay triangulation at each stage of the process, there is no theoretical guarantee that the cavity of each inserted point should be star-shaped, and this property ought to be enforced (see section 3.2.1.1.2).

– Eventually, some authors propose to initialize and maintain $\mathcal{T}$ as a *constrained Delaunay triangulation* of the associated domain of interest; see the survey in [280], chap. 6, 7, about *Delaunay refinement algorithms*. The mesh $\mathcal{T}$ is iteratively refined by inserting (using the Delaunay kernel) the centre of the circumsphere of the simplices whose ratio circumradius-to-shortest edge is larger than a given threshold, which depends e.g. on a size prescription (see figure 3.14, right). The maximum circumradius-to-shortest edge ratio of a triangle in the mesh is automatically decreased by doing so; this procedure lends itself to provably good mesh refinement algorithms in two dimensions [87, 273]. In three dimensions, slivers are likely to appear, and have to be taken care of.

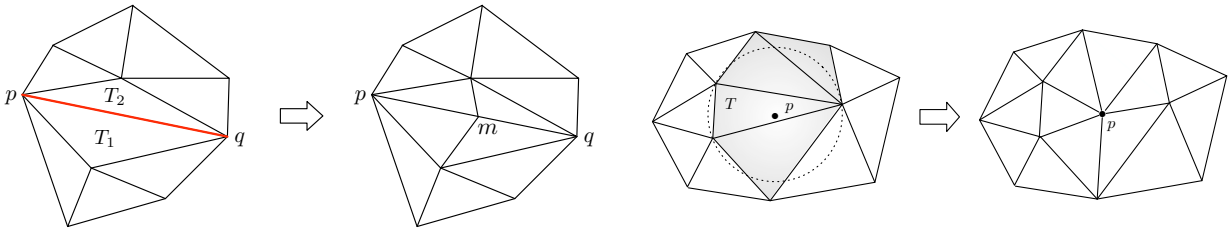


Figure 3.14: In two dimensions, *(left)* Splitting an edge pq leads to the formation of four triangles, *(right)* illustration of the Delaunay refinement procedure: the centre p of the circumcircle of T is inserted in $\mathcal{T}$, using the Delaunay kernel.

3.3.1.2 Mesh decimation operators

Mesh decimation is the exact converse operation to mesh enrichment, and allows to remove vertices from $\mathcal{T}$ that are deemed 'unnecessary' - for instance because the desired mesh size for (some region of) $\mathcal{T}$ is larger than the initial one. *Edge collapse* is the chief operator to decimate $\mathcal{T}$: let pq be a 'too short' edge of $\mathcal{T}$ to be collapsed, for instance in the sense that p is collapsed onto q. The edge collapse operator consists in deleting all the elements of the shell $\mathcal{S}h(pq)$ of pq, then to update the other simplices of the ball $\mathcal{B}(p)$ of p by trading their vertex p with q (see figure 3.15). A practical use of this operator raises several numerical issues - for instance, several validity checks must be performed - detailed in [145], that we shall discuss with more details in Chapter 8.

As suggested by their names, both enrichment and decimation operations are aimed at reaching a fine 'sampling' in the mesh, and generally do not allow by themselves to reach very good quality meshes. This becomes the purpose of the two forthcoming operations.

3.3.1.3 Connectivity changes

A first way to improve the quality of $\mathcal{T}$ consists in acting on its connectivities: the positions of the vertices of $\mathcal{T}$ is held constant, and only the binds (edges or elements) between them are changed. To serve

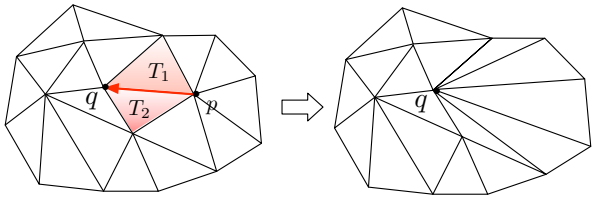


Figure 3.15: Point p is collapsed onto q; the shell $\mathcal{S}h(pq)$ vanishes in the process (in red), and the other triangles of $\mathcal{B}(p)$ are updated.

this purpose, the already introduced *swap operator* is used in two dimensions. Unfortunately, this operator does not lend itself to an easy generalization in three dimensions, and two operators are generally used, which are considered as key ingredients in building good quality meshes, for their ability to get rid of nasty configurations (including slivers). Their are more extensively described in [139, 150, 114], and in Chapter 8.

- the *face swap* operator (figure 3.16, left) applies to a configuration of two tetrahedra *pabc* and *qabc* sharing a common face *abc*. This common face is erased and the opposite edge *pq* is created in the mesh, so that the initial configuration is replaced by one featuring three new tetrahedra, namely *pqab*, *pqbc* and *pqac* (see figure 3.16, left). Of course, several validity checks are in order, so that the resulting mesh stays valid.
- The *edge swap* acts as the converse of the previous one, to some extent: let pq be an edge in $\mathcal{T}$, whose shell is denoted as $\mathcal{S}h(pq) = \{K_i = pqa_ia_{i+1}\}_{i=1,...,N}$ (indices are taken modulo N). The vertices a_i form a pseudo two-dimensional polygon $\mathcal{P}$, which can be triangulated (in a non unique way) as $\mathcal{P} = \{T_j\}_{j=1,...,M}$ without introducing any additional point (see section 3.2.1). Deleting edge pq, then introducing the tetrahedra formed by basis T_j, and fourth vertex p or q yields a swapped configuration (see figure 3.16, right). Of course, here again, checks ought to be performed, so that the resulting mesh remains valid.

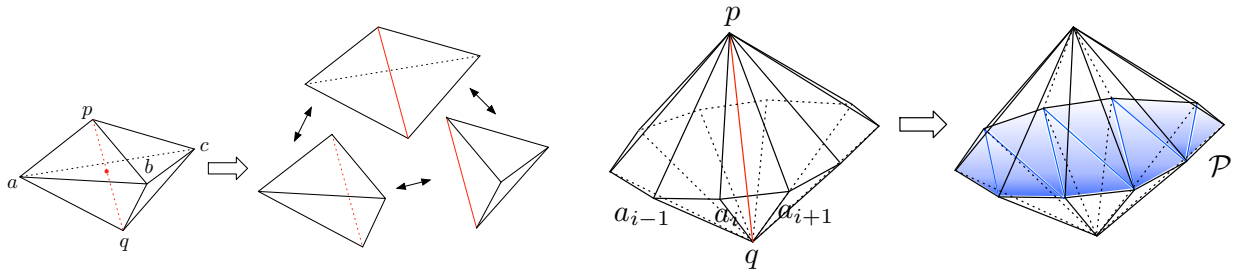


Figure 3.16: *(Left)* Swap of the common face *abc* to the two tetrahedra *abcp*, *abcq*; *(right)* swap of edge pq, associated to one of the possible triangulations of the pseudo-polygon delimited by the a_i.

3.3.1.4 Vertex relocation

Last but not least, at the core of almost any mesh optimization strategy stands the *vertex relocation* operator, which simply consists in changing the actual position of a particular vertex p of $\mathcal{T}$ for an *optimal position* p^* without altering the connectivities of the mesh (provided the motion is admissible). Only the simplices of the ball $\mathcal{B}(p)$ of p are thus affected by the operation.

Several choices are available as for the optimal position p^*. The most celebrated one leads to the so-called *Laplacian smoothing* algorithm [134]: denoting as $a_1, ..., a_N$ the vertices of $\mathcal{T}$ which are connected to p, one

may propose the following formula, which is very reminiscent of a numerical scheme for the Laplace equation on a Cartesian grid:

$$p^* = \frac{1}{N} \sum_{n=1}^{N} a_n.$$

Several improvements have been devised for this numerical scheme. To name a few (see [317]), each point a_n in the previous formula may be assigned a weight $w_n > 0$, so that the optimal position p^* reads:

$$p^* = \frac{\sum_{n=1}^{N} w_n \, a_n}{\sum_{n=1}^{N} w_n}.$$

These weights may be related to the volumes of the simplices sharing edge pa_n, to their qualities, etc... Furthermore, one could consider *relaxing* the optimal position p^*: p is not relocated to the optimal position p^*, but rather to an intermediate position $(1-\alpha)p + \alpha p^*$ (for some $\alpha \in (0,1)$) between its actual position p and p^*. In the context that each point of the mesh is processed several times in the course of a vertex relocation stage, relaxing the optimal positions proposed by the Laplacian smoothing procedure generally turns out to produce better quality meshes.

However very simple, these Laplacian smoothing procedures remain heuristic, and may not improve the mesh quality, depending on the criterion of interest. This is especially the case in three dimensions, where Laplacian smoothing - which aims at relocating p so that all the edges pa_n have similar lengths - may produce slivers. More sophisticated procedures exists for proposing a 'good' optimal position p^*; for instance, in [139], the worst quality in the elements of the ball of p is expressed as a (minimum, thus nonsmooth) function of the position of p. Techniques from nonsmooth optimization are then used to find an optimal position which explicitly increases the worst quality. The same philosophy motivates the work in [3] which relies on the particular shape of the objective function to find geometrically an optimal position. See [22] for other examples.

So far, we have been focusing on the physical part of the remeshing process - i.e. the description of the local operators involved. Actually, a great deal of the efficiency of remeshing algorithms lies in the device of a successful strategy, that is a way to steer and intertwine the operators (see for instance the discussion in Chapter 8). As examples of such, let us mention the work [179], which casts a problem of surface mesh optimization as that of minimizing a quadratic energy aggregating two terms, one of them assessing the quality of the mesh, the other one assessing the fidelity of the remeshed model to the initial triangulated surface. On a completely different note, [195] presents a strategy oriented to getting rid of the worst elements within a tetrahedral mesh.

3.3.2 Surface remeshing

Surface remeshing algorithms are expected to modify an input triangulation $\mathcal{S}$ of a surface $\Gamma \subset \mathbb{R}^3$ into a new one, say $\widetilde{\mathcal{S}}$, which is at the same time well-shaped, and a satisfactory geometric approximation of Γ, with all the implied shades of difference between the possible acceptations of this notion (see section 3.2.2).

Additional duties for a surface remeshing algorithm could be dreamt up, among other things denoising, or reconstruction of sharp features on $\mathcal{S}$... see [18] for an overview of these topics.

A surface remeshing algorithm could take two different paths:

3.3.2.1 Remeshing through parametrization of the surface

Inspired by the mesh generation method of section 3.2.2.2, a first idea consists in performing the remeshing of $\mathcal{S}$ from a parameter space $U \subset \mathbb{R}^2$, then map back the obtained two-dimensional mesh to the ambient three-dimensional space. The immediate problem of this approach is that the supplied surface is assumed to be fully discrete - i.e. no parametrization, such as one stemming from CAD modeling, is readily available.

Nevertheless, recent advances in the domain of surface parametrization make it possible to compute numerically such a parametrization, using for instance the theories on harmonic mappings [121], conformal mappings [200] - see [136] for a more exhaustive overview of the available techniques. They allow then to get a (e.g. piecewise linear) mapping $\tau : \mathcal{S} \to U$, where U is a simple domain in $\mathbb{R}^2$ (see figure 3.17). See [265, 215] for important technical details, limitations and solutions for these techniques.

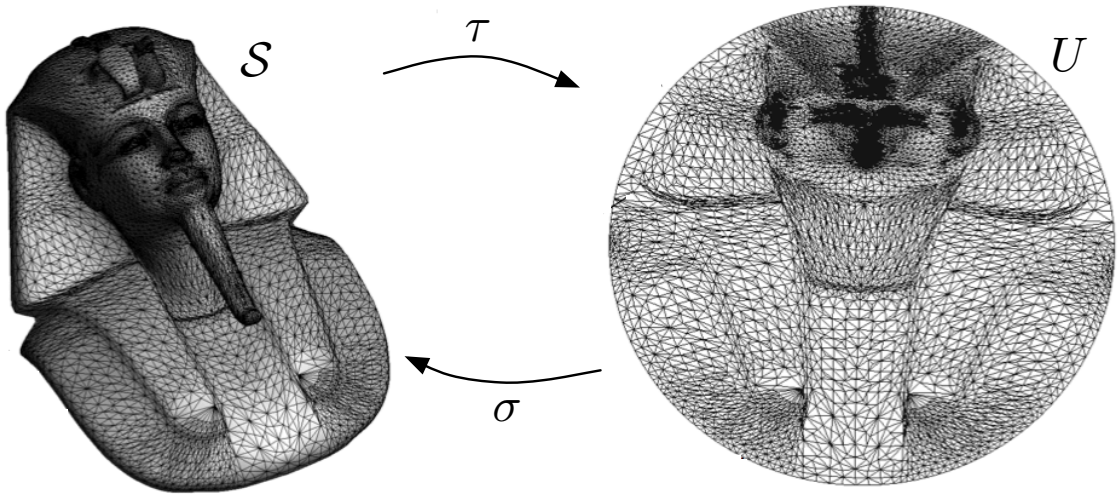


Figure 3.17: Computation of a parametrization $\sigma : U \to \mathcal{S}$ of a triangulated surface $\mathcal{S}$ using discrete harmonic mappings (reprinted from [265]).

Once such a parametrization τ, and meshed parametric domain U are computed, the discussion of section 3.2.2.2 is easily put into the context of remeshing to yield a surface remeshing algorithm.

3.3.2.2 Direct remeshing of the surface

The converse approach consists in remeshing $\mathcal{S}$ by local manipulations of the discrete triangulation $\mathcal{T}$ [140, 295]. The operators involved are exactly those described in section 3.3.1 (in the two dimensional setting). Nevertheless, they are more drastically monitored, so that the remeshing operations, performed either for reaching a fine sampling, or for elements' quality related reasons do not jeopardize with the geometric approximation of Γ. We shall linger over these questions about discrete surface remeshing in Chapter 8; let us however provide haphazardly some clues about the control of the local operators in surface remeshing:

- Inserting a new vertex p into $\mathcal{S}$ poses no particular difficulty. Because the Delaunay kernel is not easily generalized to surface triangulations, most of the authors propose to insert p into $\mathcal{S}$ by splitting one of its edges. p is generally positioned by computing a local model for a continuous surface from the discrete data around the split edge [142].
- The edge collapse operator is probably the most delicate to control. If no attention is paid, it can severely degrade the geometric approximation of Γ, or provoke folds on $\mathcal{S}$. The work [52] proposes a strategy of mesh decimation based on an upper bound on the gap in terms of Hausdorff distance (computed by geometric considerations), and on the normal deviation of triangles entailed by an edge collapse.
- The swap operator should similarly be strictly controlled, e.g. in terms of the deviation of the normal vectors to the triangles of the resulting configuration with respect to those at their nodes [318, 38].

– Eventually, the vertex relocation operator can cause a progressive deviation of the remeshed surface $\mathcal{S}$ from the initial model; in [38], the optimal position for relocating a point p is computed in its tangent plane, then, a (costly) stage of projection to the real surface is performed. In [140], the computation of optimal position p^* for a point p relies on the definition of a local quadratic model as for the behavior of Γ near p, and p is prudently moved to p^*. Quite the same approach is retained in [295], where the local reconstruction of Γ is performed in a more global fashion, and a whole strategy is based on a maintenance of a system of overlapping local patches.

Remarks 3.6.

– This short presentation is by no means exhaustive, and many remeshing strategies are based upon completely different principles. For instance, the work [327] exploits the structure of *Centroidal Voronoi Tessellation*, and *Lloyd's relaxation algorithm* for their construction, to remesh an input surface triangulation into a high-quality mesh.

– Of course, both volume and surface remeshing techniques can be worked out in concert, as complimentary parts of a *domain remeshing* algorithm, that is, an algorithm for remeshing at the same time the internal and surface parts of a tetrahedral mesh. Such a remeshing method is the cornerstone of the mesh deformation method described in [323], and is precisely what we shall be trying to carry out in Chapter 8.

3.4 Mesh evolution

This last section is dedicated to the topic of *mesh evolution*, which naturally comes up in the investigation of evolving domains (e.g. when it comes to modeling a transient physical phenomenon). Certainly, numerous celebrated *Eulerian* techniques make it possible to describe the motion of an evolving domain without relying on a meshed description (for instance, the level set method, or the volume of fluid method). However, in the study of numerically sensitive models (e.g. multiphase flows, or multiphase linear elasticity, see also the discussion in Chapter 4, §4.3.2) may take advantage of an explicit representation of the domain(s) or interface(s) involved.

The generic mesh evolution problem can be informally formulated as follows. To save notations, in this section, we do not distinguish the continuous and discrete settings; let $\Omega \subset \mathbb{R}^d$ be a domain (resp. $\Gamma \subset \mathbb{R}^d$ a surface), equipped with a simplicial mesh $\mathcal{T}$ (resp. a surface triangulation $\mathcal{S}$), and let $u : \mathbb{R}^d \to \mathbb{R}^d$ a displacement field, numerically discretized on $\mathcal{T}$ (or $\mathcal{S}$) or on a background grid, transforming Ω into $\widetilde{\Omega}$ (resp. Γ into $\widetilde{\Gamma}$). From this knowledge, how can we build a mesh $\widetilde{\mathcal{T}}$ for $\widetilde{\Omega}$ (resp. $\widetilde{\mathcal{S}}$ for $\widetilde{\Gamma}$) ?

This acute problem has been addressed by (at least) two communities whose needs and requirements differ utterly, namely computer graphics, and computational mechanics or physics.

3.4.1 Purely Lagrangian methods

Purely Lagrangian mesh deformation methods stick to the intuitive idea of an evolving domain: each vertex x of $\mathcal{T}$ (or $\mathcal{S}$) is relocated to its deformed position $x + u(x)$, the connectivities of the mesh being unaltered as far as possible (see figure 3.18, left). In the context of purely Lagrangian mesh evolution methods, we shall refer to this operation as *advecting* $\mathcal{T}$ (or $\mathcal{S}$) along u.

The problem of whether only a surface mesh, or a whole volume mesh (together with its surface mesh) should be deformed is not that simple. Paradoxically, it does not depend so much on whether the problem involves only an interface or a domain, as on the precise numerical context of the study. It is generally dictated by the context. For instance, in [309, 158], only a mesh of an interface Γ between two (or several) domains filled with different fluids is maintained, whereas its velocity is governed by fluid equations posed on whole domains. This is because these equations are actually solved on a background Cartesian grid using high-order finite difference schemes; the mesh of the interface is used to obtain a fine approximation of the interface conditions (e.g. the surface tension in [309]). On the contrary, there are purely surface

models (e.g. the mean curvature flow, evoked in chapter 1) that are dealt with by embedding the surface $\mathcal{S}$ in a volume mesh $\mathcal{T}$ - in the sense that the faces of $\mathcal{S}$ exist as faces of simplices of $\mathcal{T}$ - and all the mesh $\mathcal{T}$ is deformed (reasons motivating this approach are explained hereafter) [231]. Surface and volume mesh Lagrangian deformation exhibit different characteristics, which are summed up below.

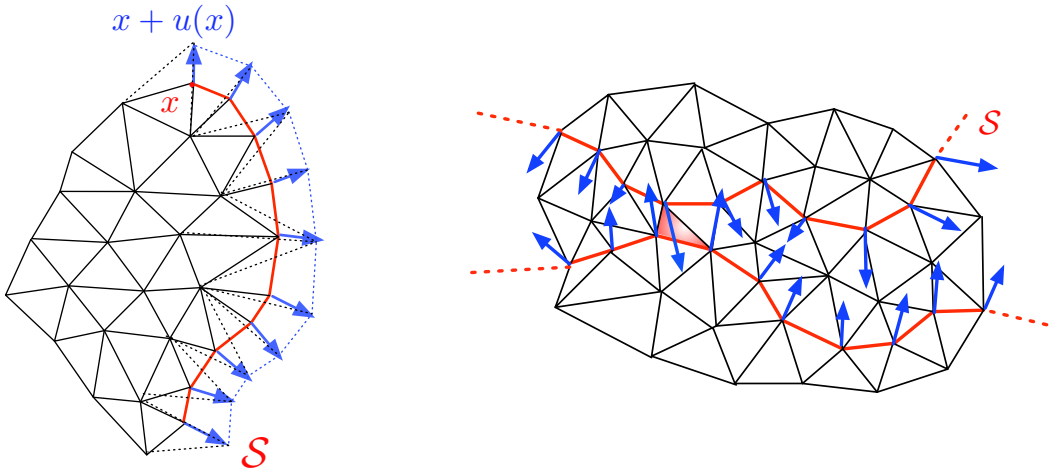


Figure 3.18: In two dimensions, *(left):* advection of the nodes of a volume mesh $\mathcal{T}$; *(right):* an interface $\mathcal{S}$ (red segments) is embedded in a volume mesh $\mathcal{T}$. The trend of $\mathcal{S}$ to develop self-intersections in the course of its advection along a velocity field u (in blue) can be detected by the inversion of the red element.

3.4.1.1 Lagrangian deformation of a surface triangulation

At first glance, deforming a triangulated surface $\mathcal{S}$ according to a vector field u (e.g. defined at its nodes) may seem reasonably easier than deforming a volume mesh $\mathcal{T}$. Unfortunately, quite the opposite is true: relocating each vertex x of $\mathcal{S}$ to its deformed position $x+u(x)$, leaving the connectivities of $\mathcal{S}$ unchanged may lead to a severely distorted surface mesh $\widetilde{\mathcal{S}}$. Indeed, even if $\mathcal{S}$ is well-shaped, with uniform size distribution, the deformation imposed by u may trigger high or low concentration of vertices on some regions of $\widetilde{\mathcal{S}}$ (see figure 3.19, left). Even worse, not even broaching the (common) case that u may express a change of topology from Γ to $\widetilde{\Gamma}$, even very 'smooth' vector fields u may entail self-intersections of $\widetilde{\mathcal{S}}$ depending on the initial distribution of vertices on $\mathcal{S}$ (figure 3.19, right).

Note that the sole detection of whether a triangulation surface presents self-intersections is not an easy task, and can be computationally expensive if no particular care is paid to the implementation. Actually, it is very similar to the front-checking operation in advancing-front mesh generation methods, and like then, authors generally rely on a background Cartesian grid to make the process computationally affordable [309].

Several solutions have been thought up to address all these critical issues, which we now briefly outline.

3.4.1.1.1 Connections with remeshing Almost all the works around mesh deformation incorporate at some point a (local) remeshing stage; the reasons are twofold:

- first, as evoked above, advecting a (surface) mesh $\mathcal{S}$ along a deformation field u dramatically jeopardizes the quality of the resulting mesh $\widetilde{\mathcal{S}}$ (let alone the fact that it may develop self-intersections), which may undermine the accuracy of numerical operation performed on it (see section 3.1.2).
- Second, the fact that $\mathcal{S}$ present a fine quality of elements may, if not prevent, at least postpone the appearance of 'folds' or self-intersections in the resulting surface $\widetilde{\mathcal{S}}$, and thus facilitate an iterative process in which the whole deformation account for by u is performed within several substeps of incomplete yet safe mesh deformation.

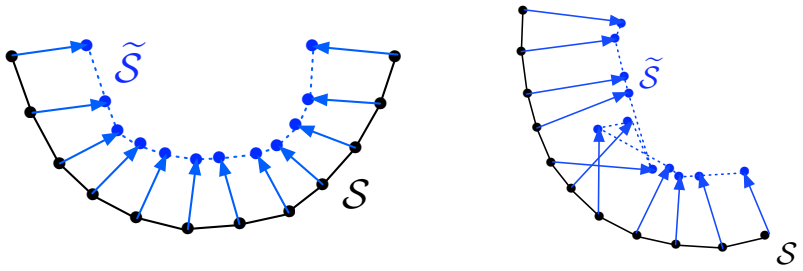


Figure 3.19: In two dimensions, a 'reasonable' deformation field u transforms a uniformly meshed contour $\mathcal{S}$ into *(left)* a new contour $\widetilde{\mathcal{S}}$ presenting a very high concentration of vertices; *(right)* a self-intersecting contour $\widetilde{\mathcal{S}}$.

However, the 'amount' of remeshing that should be performed within a mesh deformation process is quite unclear, and possibly application-dependent. For instance, the works [324, 330] in the field of computer graphics intertwine mesh advection stages and remeshing stages, and use the whole kit of local operators described in section 3.3 to maintain a surface mesh composed of high quality triangles, amenable for accurate numerical computations. On the contrary, the works [309] to a lesser extent, and [158] are devoted to a high-accuracy simulation of multiphase flows, and highlight that remeshing the evolving triangulated surface requires to interpolate the attached state values. Hence, they advocate not to remesh this surface to often, lest that is should cause excessive numerical diffusion.

3.4.1.1.2 Modifying the input velocity field Let us now look into how the considered evolving surface could be prevented from developing self-intersections. A first possible method consists in anticipating self-intersections on the advected mesh $\widetilde{\mathcal{S}}$, and modifying the values of the displacement field u into new values $\widetilde{u}$, in such a way that advecting $\mathcal{S}$ along $\widetilde{u}$ yields a topologically valid surface mesh.

As an example, in [183], a procedure is presented which strives to mimic the 'causality principle' attached to surface evolution discussed in chapter 1: in a first stage, the initial surface mesh $\mathcal{S}$ is advected according to the velocity field u, to produce a possibly invalid surface triangulation $\mathcal{S}'$. In a second stage, for each vertex $x \in \mathcal{S}$, the images of the triangles of the ball of x in $\mathcal{S}'$ are analyzed, and a new deformed position $x + \widetilde{u}(x)$ for x is found as the intersection (in the least squares sense) to the supporting planes of these triangles. This new position is possibly modified using a curvature analysis of the resulting surface, and a tangential motion term may be added to redistribute the vertices on the deformed surface $\widetilde{\mathcal{S}}$ (as a substitute to a remeshing procedure).

A slightly different method is proposed in [61], where a *continuous collision detection* algorithm makes it possible to detect colliding elements in the course of the advection from $\mathcal{S}$ to $\widetilde{\mathcal{S}}$. The deformation field u around the vertices of the colliding triangles are modified by deleting their component along the 'collision direction'.

3.4.1.1.3 Resolving intersections The converse viewpoint consists in acknowledging whether -and where - the advected surface $\widetilde{\mathcal{S}}$ is self-intersecting, and trying to remedy this problem. This work can be done prior (i.e. by an analysis of $\mathcal{S}$ and u) or after (i.e. by an analysis of $\widetilde{\mathcal{S}}$) the intersection did occur. Once two close regions on $\mathcal{S}$ (leading to a self-intersection on $\widetilde{\mathcal{S}}$), or two intersecting regions of $\widetilde{\mathcal{S}}$ are identified, the following procedures can be used:

– in [61], a method is proposed, which applies to a configuration of two 'close' edges on the initial surface $\mathcal{S}$, say e and e', whose vicinities have self-intersecting images on $\widetilde{\mathcal{S}}$. The two regions are merged on $\mathcal{S}$ by replacing the configurations of the four surface triangles sharing e or e' as edges with a new one, where the four triangles are removed, and the boundaries of the resulting holes are joined by a 'pipe'

made of eight triangles, provided this new configuration yields a topologically valid surface.
- in [330], two intersecting areas are identified in the advected surface mesh $\widetilde{\mathcal{S}}$ (using a background Cartesian grid to speed up the detection). The (polygonal) intersection line $\Sigma \subset \widetilde{\mathcal{S}}$ between those regions is explicitly discretized in this mesh by refining accordingly the intersecting triangles, leading to a new surface mesh $\widetilde{\mathcal{S}}'$. Eventually, a simple analysis of the surface properties allows to decide which of the two parts of the resulting non manifold, yet conforming, surface triangulation $\widetilde{\mathcal{S}}'$ should be removed to provide a final manifold surface mesh.
- in [324], $\mathcal{S}$ is once again embedded in a Cartesian grid of a computational box, and self-intersections on $\widetilde{\mathcal{S}}$ (or possibly very close parts that may want to merge, or very thin parts that may want to separate on $\mathcal{S}$) are looked for, inside each grid cell. In whichever case, a pattern is used, in the spirit of the marching cubes method, to propose a new discretization of the part of the surface inside the grid cell, which is topologically manifold.

All these methods prove highly combinatorial, and difficult to implement. As we have just noticed, the main difficulty in tracking an evolving surface $\mathcal{S}$ is the difficulty in identifying close parts on $\mathcal{S}$. For this reason, it is generally acknowledged that equipping $\mathcal{S}$ with a volumetric structure may prove significantly helpful in assessing (or preventing) that $\mathcal{S}$ is on the verge of becoming - or has already become - invalid. In this spirit, a deformation method is presented in [334] in which an evolving surface is connected to a graph structure whose inversions betray self-intersections on $\mathcal{S}$.

3.4.1.2 Deforming a volume mesh together with its surface mesh

The Lagrangian deformation of a tetrahedral mesh $\mathcal{T}$ undoubtedly brings about more constraints than that of a sole surface mesh $\mathcal{S}$. Conversely, these additional constraints can be seen as subsequent controls over the ongoing mesh advection procedure; indeed, detecting whether a tetrahedron $K \in \mathcal{T}$ ends up degenerated (or worse, inverted) at some point is far easier than detecting self-intersections of a surface mesh (it only requires comparing the orientation of $K \in \mathcal{T}$ with that of its advected counterpart $\widetilde{K} \in \widetilde{\mathcal{T}}$); see figure 3.18, right.

For this reason, even when only the deformation of a surface triangulation $\mathcal{S}$ is investigated, some authors [115, 231] deem fit to embed $\mathcal{S}$ into a larger tetrahedral mesh $\mathcal{T}$ - in the sense that the faces of $\mathcal{S}$ arise as faces of tetrahedra of $\mathcal{T}$ - then to proceed to the whole deformation of $\mathcal{T}$.

In the remainder of this section, let $\mathcal{T}$ be a tetrahedral mesh, whose evolution along a displacement field u is of interest, and $\mathcal{S}_\mathcal{T}$ the associated surface mesh. The forthcoming descriptions can be straightforwardly adapted to the setting of an evolving surface triangulation embedded in a tetrahedral mesh hinted at above.

A first important point is that u may be defined only at the vertices $x \in \mathcal{S}_\mathcal{T}$, and not at the internal ones - shape optimization (see Chapters 2 and 9) is only one example of such setting among many. One could of course try and deform $\mathcal{T}$ by the sole displacement of the vertices of the boundary mesh [231]. However, following [115, 29], extending consistently the displacement field u to the internal vertices of the mesh may ease the process dramatically. The work [29] proposes to generate a displacement field $\widetilde{u}$ on the whole mesh $\mathcal{T}$ of the considered domain Ω as the solution to a linear elasticity system posed on Ω, with imposed Dirichlet boundary conditions u on $\partial\Omega$ (i.e. at the vertices of $\mathcal{S}_\mathcal{T}$.

Once a displacement value has been assigned to each vertex $x \in \mathcal{T}$ (even if it implies that $u(x) = 0$ at the internal vertices $x \in \mathcal{T}$), the advection of $\mathcal{T}$ along u can be carried out. Of course, doing so is very likely to cause overlappings of the resulting mesh $\widetilde{\mathcal{T}}$. This situation can be easily detected by comparing the orientations of the elements of $\widetilde{\mathcal{T}}$ with those of $\mathcal{T}$.

Now, to get past this difficulty, the works [29, 115, 231] propose to rely on a smart intertwining of incomplete advection stages of the evolving mesh and remeshing stages, which is illustrated in figure 3.20, and about which we now provide a coarse, schematic description. If by any chance $\widetilde{\mathcal{T}}$ happens to be valid

(i.e. non overlapping), the procedure stops. Otherwise, $\mathcal{T}$ is advected as long as it stays valid, meaning that a real number $\alpha \in (0,1)$ is found (e.g. by dichotomy) such that the advected mesh $\mathcal{T}'$ along $\alpha\, u$ is valid, but 'on the verge of becoming invalid' (e.g. one or several elements are nearly degenerate). Then, the nearly degenerate configurations of $\mathcal{T}'$ are improved by triggering a remeshing procedure of $\mathcal{T}'$ (edge swaps prove especially useful in getting rid of nearly flat elements). A new, hopefully well-shaped, partially advected mesh $\widetilde{\mathcal{T}'}$ is obtained, and the algorithm starts back with the advection of $\widetilde{\mathcal{T}'}$ along the remaining displacement $(1-\alpha)u$.

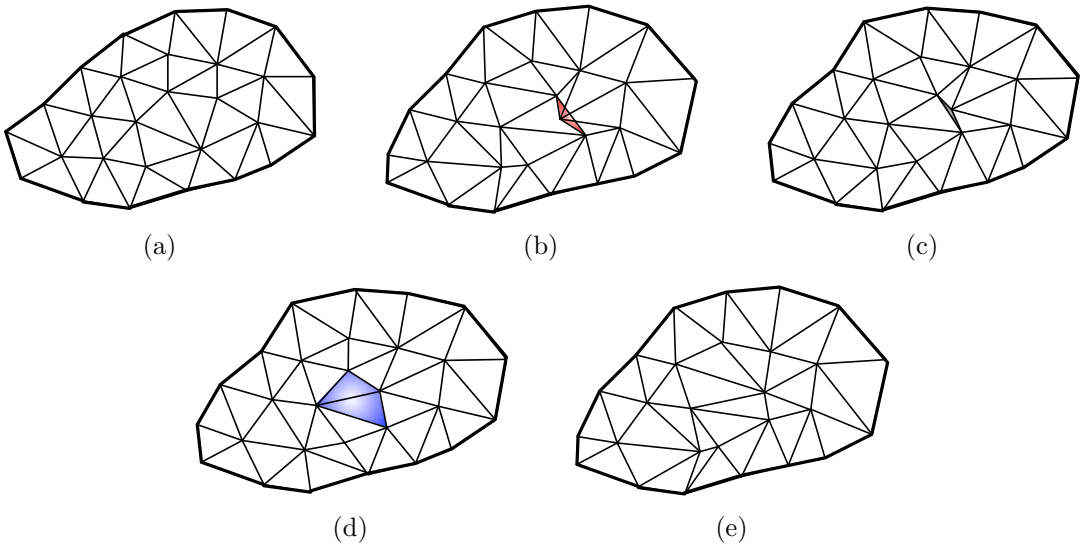


Figure 3.20: An example of mesh deformation procedure; *(a)* Initial mesh $\mathcal{T}$, *(b)* the invalid, advected mesh $\widetilde{\mathcal{T}}$ (the element in red has been inverted), *(c)* the partially advected mesh $\mathcal{T}'$; the greyed element is on the cusp of inversion, *(d)* the remeshed, partially advected configuration $\widetilde{\mathcal{T}'}$; en edge swap has made it possible to resolve the nearly degenerate configuration (in blue); *(e)* the final configuration $\widetilde{\mathcal{T}}$.

Note that, in the course of its deformation, an evolving domain $\mathcal{T}$ may of course develop self-intersections. This typically happens when two independent parts of $\mathcal{S}_{\mathcal{T}}$ tend to merge, and cannot be prevented by other methods than those evoked in section 3.4.1.1. The fundamental difficulty is that, once again, no 'volume structure' exists to prevent such collisions. For this reason, in [231], the authors to embed any evolving triangulated surface $\mathcal{S}$ or domain $\mathcal{T}$ into a mesh of a larger computational box D. In the last case, the fact that $D \setminus \mathcal{T}$ is also meshed enables an easy detection of any kind of self-intersection of $\mathcal{T}$, at the expense of more constraints on the deformation.

Remark 3.7. Several strategies, referred to as *Arbitrary Lagrangian-Eulerian* methods (see the introductory material [118, 181]), have been devised in the field of numerical simulation in which the displacement of the vertices of the mesh may be decoupled from the movement of the considered domain, so that the movement is always possible and the quality of the mesh is never too much degraded. A correspondance between the actual mesh, referred to as the *reference mesh* - which may not exactly account for the domain - and the real domain itself, must be kept, and the underlying equations of the simulation involved must be written in terms of the *reference coordinates*.

3.4.2 Hybrid methods

The difficulties in carrying out mesh evolution methods in a purely Lagrangian way, especially when it comes to describing motions during which numerous topological changes occur, urged many authors to

free themselves from tracking deformations explicitly. A non exhaustive and biased selection of alternative methods is now provided:

- The method proposed in [33] features an explicit discretization $\mathcal{S}$ of an evolving surface Γ (we do not make explicit the dependance of Γ with respect to time for simplicity) at each time step of the evolution, but describes its deformation in a purely Eulerian way: a background (adaptive) Cartesian grid of a computational box D is maintained, and the interface Γ is described from the standpoint of the level set method. The surface evolution between any two iterations of the process is performed using the level set method (see chapter 1, section 1.2.3 for a description of Strain's numerical scheme, which is the one used here), and at each iteration, a *contouring step* extracts a piecewise affine representation of Γ at the current state, using a variant of the marching cubes method. The resulting mesh of the interface is very ill-shaped, but the mesh is only extracted for visualization purposes, a case in which it does not pose any problem. Note that this technique also allows for an easy transfer of physical properties of the surface from one iteration to the other (in this particular case, the color).
  A similar approach is used in [221]: a polygonal contour is deformed under the motion of its vertices, and at each time, the (possibly invalid) contour is resampled from a intersecion with a background grid, and a marching cubes reconstruction analysis.
- The method for mesh evolution proposed by Persson in [254], §5.3 shares many characteristics with the one proposed in chapter 9: it is interested in a shape optimization problem, namely the *Cantilever* test case (see Chapter 2, §2.3, where computing the displacement field u for the evolving domain Ω requires solving a linear elasticity equation on Ω. To this end, Ω is tracked by relying on the level set method (performed on a Cartesian grid), and at each time step, Persson's method for generating a mesh associated to an implicitly-defined domain (see section 3.2.1.3) to get a mesh of the actual shape Ω, which makes it possible to compute the new displacement field for Ω.
- Eventually, the method proposed in [260] is of an altogether different type. It proposes to embed a mesh $\mathcal{T}$ of an evolving domain Ω into a mesh of a larger box D. At each iteration, the mesh $\mathcal{T}$ is advected in a purely Lagrangian way, and becomes potentially invalid. Yet, it allows to get an 'print' of the advected domain $\widetilde{\Omega}$ (e.g. by removing the inverted elements). Besides, its also provides a (non connected) set of vertices $\widetilde{\mathcal{P}}$ for building a mesh $\widetilde{\mathcal{T}}$ of $\widetilde{\Omega}$. This operation is achieved by taking the *restricted Delaunay triangulation* of $\widetilde{\mathcal{P}}$ (in D) to the available 'print' of $\widetilde{\Omega}$.

Part II

Two problems in shape optimization

Chapter 4

Multi-phase optimization via a level set method

Contents

4.1 Introduction 102
4.2 Around the shape differentiability of the signed distance function 104
4.2.1 Some facts around the signed distance function 104
4.2.2 Shape derivative of the signed distance function 106
4.2.3 Another expression for these derivatives 111
4.2.4 More differentiability results for the signed distance function 113
4.2.4.1 The signed distance function is not differentiable as a L^∞_{loc}-valued function of the domain 113
4.2.4.2 Shape differentiability of the projection application $p_{\partial\Omega}$ 114
4.2.4.3 Application to shape differentiability results for the signed distance function in higher regularity spaces 117
4.2.4.4 The conclusion of proposition 4.8 does not extend when D intersects $\overline{\Sigma}$. 119
4.2.5 Some words around the notion of minimum thickness of a domain 120
4.2.5.1 The reach of a set 120
4.2.5.2 Sets with minimum thickness 121
4.3 Sharp-interface formulation in a fixed mesh framework 124
4.3.1 Description of the problem 124
4.3.2 Shape-sensitivity analysis of the sharp-interface problem 125
4.4 Shape derivative in the smoothed-interface context 131
4.4.1 Description of the problem 131
4.4.2 Shape derivative of the compliance in the multi-materials setting 131
4.4.3 Approximate formulas for the shape derivative 133
4.4.4 Convergence of the smoothed-interface shape optimization problem to the sharp-interface problem 134
4.5 Extension to more than 2 materials 136
4.6 Discussion and comparison with previous formulae in the literature 138
4.7 Numerical results 140
4.7.1 Level-set representation 140
4.7.2 Two materials in the sharp interface context 141
4.7.3 Two materials in the smoothed-interface context 142
4.7.4 Four phases in the smoothed interface context 145

4.7.4.1 Short cantilever using two materials and void . . . 145
4.7.4.2 Short cantilever using three materials and void . . . 145
4.7.4.3 3-force bridge using two materials and void . . . 147
4.7.4.4 3-force bridge using three materials and void . . . 148
4.7.4.5 Medium cantilever using three materials and void . . . 148
4.7.4.6 Long cantilever using two materials and void . . . 149
4.8 Appendix: convergence of the smoothed-interface shape optimization problem to its sharp-interface equivalent . . . 150
4.8.1 A model problem in the context of thermal conductivity . . . 150
4.8.1.1 Study of the asymptotic behavior of u_ε as $\varepsilon \to 0$. . . . 151
4.8.1.2 Convergence of the shape gradient associated to the smoothed-interface problem as $\varepsilon \to 0$. . . . 156
4.8.2 Extension to the context of linearized elasticity . . . 159

This chapter is a joint work with Grégoire Allaire, Gabriel Delgado and Georgios Michailidis, and was submitted for publication as the shorter article

G. ALLAIRE, C. DAPOGNY, G. DELGADO AND G. MICHAILIDIS, *Multi-phase optimization via a level set method*, submitted (2013).

which omits certain technical details around the signed distance function, presented here in section 4.2, as well as the proofs of most of the presented theoretical results (notably the convergence proof in section 4.8).

4.1 Introduction

As exemplified by the recent enthusiasm encountered by the study of composite materials, the general problem of finding the optimal distribution of several materials in a fixed working domain, in order to minimize a criterion related to the overall mechanical behavior or cost of the phases mixture, is of great relevance in material science and industry.

A crucial issue in the modeling of this problem is the parametrization of the phases mixture. While the exact formulation require the material properties, or the global Hooke's tensor, to be discontinuous at the interfaces between two materials, it is often convenient, for numerical purposes, to devise an appropriate material interpolation scheme to smoothen the coefficients or equivalently to replace sharp interfaces by diffuse ones. This diffuse or smeared interface approach has its own interest when one is interested in the optimization of functionally graded materials [66], [203], [296], [308], [314].

There is already a vast literature about multiphase optimization and various methods have been proposed to address the problem. The Hadamard method of geometric shape optimization, as described in [105], [172], [234] (see also the brief reminder in chapter 2), was used, for example, in [171] for optimal composite design. The homogenization method [8], [82], [305] was the main tool in the multiphase problem studied in [11] for the optimal reloading of nuclear reactors (sequential laminates were shown to be optimal composite materials). In the framework of the SIMP (Solid Isotropic Material with Penalization) method, several interpolation schemes have been proposed for the mathematical formulation of the Hooke's tensor of the mixture [39], [303], [328]. In general, material interpolation schemes can be quite involved [328] and one may design such a model in order to favor certain phases [303]. Applications range from the design of materials with extreme or unusual thermal expansion behavior [287] to multi-material actuators [285], through conductivity optimization for multi-phase microstructural materials [336]. In the framework of the phase-field method, a generalized Cahn-Hilliard model of multiphase transition was implemented in [335] to perform multi-material structural optimization.

The first publications on multi-phase optimization, using the level set method, are [223] and [321] (see also [224], [320], [322]). Following an idea of Vese and Chan [315], the authors in [223], [321] used m level set functions to represent up to $n = 2^m$ materials: we shall adhere to this setting (see section 4.5). The level set functions are advected through eikonal Hamilton-Jacobi equations in which the normal velocity is given by the shape derivative of the objective function. Unfortunately, the shape derivatives, derived in [223] and [321], are not correct in full mathematical rigor as we explain in section 4.6. Fortunately, they are however approximations of the correct formula upon various assumptions. A first goal of the present chapter is to clarify the issue of shape differentiability of a multi-phase optimization problem. In section 4.3 we give the correct shape derivative in the setting of a sharp interface between phases (see Proposition 4.10). It was first obtained in [15] for a problem of damage and fracture propagation but, in a scalar setting, previous contributions can be found in [175], [46], [248]. Because the phase properties are discontinuous through the interfaces, the transmission conditions imply that only the elastic displacement and the normal stress are continuous at the interfaces, leaving the tangential stress and the normal strain discontinuous. These discontinuities yield obvious difficulties which must be handled carefully. The exact or continuous shape derivative turns out to be somehow inadequate for numerical purposes since it involves jumps of strains and stresses through the interfaces, quantities which are notably hard to evaluate with continuous finite elements. Therefore, Proposition 4.11 gives a discrete variant of this shape derivative which does not involve any jumps and is similar to the result of [223] and [321]. The idea is to consider a finite element approximation of the elasticity system, the solution of which has no derivative jumps through the interface, implying that the shape derivative is much easier to compute.

Another delicate issue in multiphase optimization using the level set method is that the interface is inevitably diffused and its thickness may increase, thus deterioring the peformance of the analysis and eventually of the optimization. Note that, for most objective functions, it is always advantageous to introduce intermediate values of the material properties, so that the interface spreading is produced by the optimization process itself and not merely by the numerical diffusion. In [223] the authors introduced a penalization term to control the width of the interpolation zone between the materials. In [321] the level set functions are re-initialized to become signed distance functions, which permits a more explicit control of the interpolation width.

A second goal of the present chapter is to propose a smoothed-interface setting which guarantees a fixed thickness of the interface without any increase in its width (as it is already the case in the standard single material level set method for shape and topology optimization). This setting relies on the notion of signed distance function to a domain, whose related features have already been extensively used in image processing [329], or shape optimization [108]. In the course of a long digression in section 4.2, we are interested in the behavior of the signed distance function with respect to variations of the considered domain, as well as in other related properties of independent interest. In section 4.4 we describe a regularization of the interface which relies on the signed distance function to the interface. Note that the signed distance function has nothing to do with the level set function which is used in numerical simulations. Indeed, the solution of the advection Hamilton-Jacobi equation (with a velocity given by the shape derivative) is usually not the signed distance function (which explains why reinitialization is often used in practice). In such a smoothed interface setting our main result is Theorem 4.8 which gives the shape derivative of the objective function. It requires several intermediate technical results, notably finding the shape derivative of the distance function (first obtained in [108]) and using a coarea formula to reduce a volume integral to a product integral on the interface and along normal rays. Once again, we show in section 4.4.3 that, when the regularization parameter (or the thickness of the diffuse interface) is vanishingly small, the exact shape derivative can be approximated by the formula already obtained in Proposition 4.11 which corresponds to the result of [223] and [321] too.

Section 4.4.4 and the appendix in section 4.8 explain how the smoothed interface model converges to the sharp interface problem as the regularization parameter goes to zero. Since, for simplicity, all the previous theoretical results were stated in the case of a single interface between two phases, we explain how to generalize our smoothed interface setting to more materials in section 4.5. Section 4.6 is devoted to a comparison with [223] and [321]. Finally, in section 4.7 we show several 2-d results and make comparisons

between the different settings and formulas for the shape derivatives. Some optimal designs obtained by our approach are compared to those previously computed in [321] and [322]: ours are more symmetric and sometimes slightly different. We believe it is due to our use of a correct shape derivative instead of an approximate one.

4.2 Around the shape differentiability of the signed distance function

4.2.1 Some facts around the signed distance function

The main purpose of this section is to study the *signed distance function* to a domain.

Definition 4.1.

– *Let $A \subset \mathbb{R}^d$ be a closed set. The Euclidean distance function to A, denoted as $d(.,A)$, is defined as:*

$$\forall x \in \mathbb{R}^d,\ d(x,A) = \min_{a \in A} |x-a|. \tag{4.1}$$

– *Let $\Omega \subset \mathbb{R}^d$ be a Lipschitz domain. The signed distance function d_Ω to Ω is defined as:*

$$\forall x \in \mathbb{R}^d,\ d_\Omega(x) = \begin{cases} -d(x,\partial\Omega) & \text{if} \quad x \in \Omega \\ 0 & \text{if} \quad x \in \partial\Omega \\ d(x,\partial\Omega) & \text{if} \quad x \in {}^c\overline{\Omega} \end{cases}. \tag{4.2}$$

Throughout section 4.2, Ω stands for a bounded open set in $\mathbb{R}^d$ which is only assumed to be Lipschitz for the moment. Note that some of the hereafter presented concepts and results would hold in a more general settings, but we shall not require so much generality.

Let us start by collecting some definitions related to the signed distance function to Ω (see Figure 4.1 for a geometric illustration).

Definition 4.2. *Let $\Omega \subset \mathbb{R}^d$ be a Lipschitz bounded open set.*

– *For any $x \in \mathbb{R}^d$, $\Pi_{\partial\Omega}(x) := \{y_0 \in \partial\Omega$ such that $|x-y_0| = \inf_{y\in\partial\Omega} |x-y|\}$ is the set of projections of x on $\partial\Omega$. It is a closed subset of $\partial\Omega$. When $\Pi_{\partial\Omega}(x)$ reduces to a single point, this point is denoted as $p_{\partial\Omega}(x)$, and is called the projection of x onto $\partial\Omega$.*

– *$\Sigma := \left\{x \in \mathbb{R}^d$ such that $(d_\Omega)^2$ is not differentiable at $x\right\}$ is the skeleton of $\partial\Omega$ (or sometimes Ω by a small abuse in terminology).*

– *For any $x \in \partial\Omega$, $\mathrm{ray}_{\partial\Omega}(x) := \{y \in \mathbb{R}^d$ such that d_Ω is differentiable at y and $p_{\partial\Omega}(y) = x\}$ is the ray emerging from x. Equivalently, $\mathrm{ray}_{\partial\Omega}(x) = p^{-1}_{\partial\Omega}(x)$.*

Note that these definitions are easily generalized to the case of an arbitrary closed set $A \subset \mathbb{R}^d$. For instance, one may define the set of projections of $x \in \mathbb{R}^d$ onto A as: $\Pi_A(x) := \{y_0 \in A$ such that $|x-y_0| = \inf_{y\in A} |x-y|\}$.

We now recall some classical results (see [105], chapter 7, theorems 3.1, 3.3 and [172], prop. 5.4.14 for proofs).

Proposition 4.1.

1. *Let $x \in \mathbb{R}^d \setminus \partial\Omega$ and $y \in \Pi_{\partial\Omega}(x)$. If $\partial\Omega$ enjoys $\mathcal{C}^1$ regularity in a neighbourhood of y, then:*

$$\frac{x-y}{d_\Omega(x)} = n(y),$$

where $n(y)$ is the unit normal vector to Ω at y, pointing outward.

2. *A point $x \notin \partial\Omega$ has a unique projection $p_{\partial\Omega}(x)$ onto $\partial\Omega$ if and only if $x \notin \Sigma$. In such a case, it satisfies $d(x, \partial\Omega) = |p_{\partial\Omega}(x) - x|$ and the gradient of d_Ω at x reads*

$$\nabla d_\Omega(x) = n(p_{\partial\Omega}(x)) = \frac{x - p_{\partial\Omega}(x)}{d_\Omega(x)}.$$

3. *As a consequence of Rademacher's theorem ([126], section 3.1.2), Σ has zero Lebesgue measure in $\mathbb{R}^d$. Furthermore, when Ω is $\mathcal{C}^2$, $\overline{\Sigma}$ has zero Lebesgue measure too [214].*
4. *For any $x \in \mathbb{R}^d$, $p \in \Pi_{\partial\Omega}(x)$, $\alpha \in [0,1]$, denoting $x_\alpha := p + \alpha(x - p)$ the points of the $\mathrm{ray}_{\partial\Omega}(x)$ lying between p and x, we have $d_\Omega(x_\alpha) = \alpha d_\Omega(x)$ and $\Pi_{\partial\Omega}(x_\alpha) \subset \Pi_{\partial\Omega}(x)$.*
5. *If Ω is of class $\mathcal{C}^k$, for $k \geq 2$, then d_Ω is $\mathcal{C}^k$ too in a tubular neighborhood of $\partial\Omega$. In that case, d_Ω is differentiable at every point $x \in \partial\Omega$, and at such a point: $\nabla d_\Omega(x) = n(x)$.*

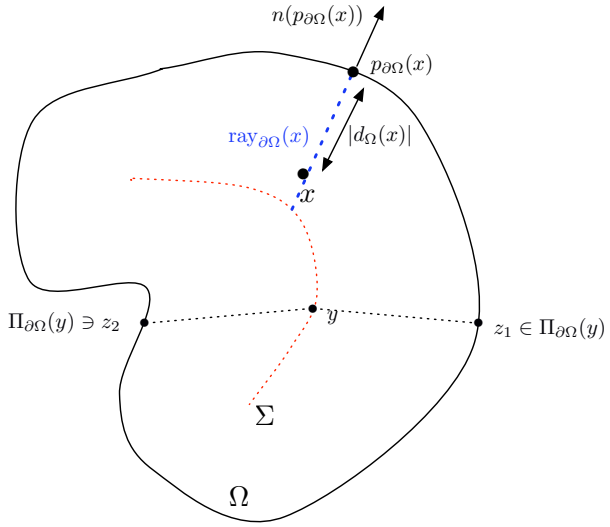


Figure 4.1: Unique projection point $p_{\partial\Omega}(x)$ and line segment $\mathrm{ray}_{\partial\Omega}(x)$ of a point x lying outside the skeleton Σ of Ω. For a point $y \in \Sigma$, at least two points z_1, z_2 belong to the set of projections $\Pi_{\partial\Omega}(y)$.

When the boundary $\partial\Omega$ of the considered domain enjoys additional regularity, sharper results become available. Suppose now that Ω is a bounded domain of class $\mathcal{C}^2$. For $x \in \partial\Omega$, denote $\kappa_i(x)$, $i = 1, ..., d-1$ the principal curvatures of the smooth submanifold $\partial\Omega$, and $e_i(x)$ the associated principal directions. These curvatures are oriented with the convention that $\kappa_i(x) \geq 0$ if $\partial\Omega$ is locally convex near x, in the normal section defined by $e_i(x)$ (see figure 4.2).

The following proposition is proved in [69].

Proposition 4.2. *For every $x \in \mathbb{R}^d$, and every $y \in \Pi_{\partial\Omega}(x)$, one has*

$$\forall i = 1, ..., d-1, \quad -\kappa_i(y) d_\Omega(x) \leq 1.$$

Furthermore, denote Γ the set of points $x \notin \Sigma$ such that the above equality holds for some i. Then we have:

$$\overline{\Sigma} = \Sigma \cup \Gamma.$$

If $x \notin \overline{\Sigma}$, then d_Ω is twice differentiable at x and

$$-\kappa_i(p_{\partial\Omega}(x)) d_\Omega(x) < 1 \quad ; \quad \mathcal{H} d_\Omega(x) = \sum_{i=1}^{d-1} \frac{\kappa_i(p_{\partial\Omega}(x))}{1 + \kappa_i(p_{\partial\Omega}(x)) d_\Omega(x)} e_i(p_{\partial\Omega}(x)) \otimes e_i(p_{\partial\Omega}(x))$$

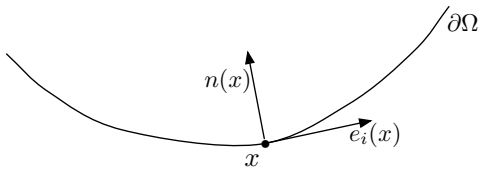


Figure 4.2: Case of a *negative* curvature : $\partial\Omega$ is locally concave near x, in the normal section associated to $e_i(x)$.

Remark 4.1. From this result, a standard bootstrap argument allows to conclude that d_Ω is actually 'as regular as the boundary', outside $\overline{\Sigma}$. The projection application $p_{\partial\Omega}$ is indeed related to d_Ω and ∇d_Ω by:

$$\forall x \in \mathbb{R}^d \setminus \Sigma,\ p_{\partial\Omega}(x) = x - d_\Omega(x)\nabla d_\Omega(x).$$

Let us end this section with a result around the behavior of the skeleton of a series of domains which 'smoothly' converge to a given domain Ω. To this end, we need the following definition:

Definition 4.3. *($\mathcal{C}^k$ topology on the space of domains) Let $\mathcal{E}_k$ the set of all bounded domains in $\mathbb{R}^d$ with $\mathcal{C}^k$ boundary. The $\mathcal{C}^k$ topology on $\mathcal{E}_k$ is the topology spanned by the sets*

$$\mathcal{U}_{\varepsilon,\Omega_0} := \left\{\Omega \in \mathcal{E}_k\text{: there exists a surjective embedding } \phi : \Omega_0 \to \Omega,\ ||\phi - id||_{\mathcal{C}^k(\Omega_0)} < \varepsilon\right\},$$

where $||.||_{\mathcal{C}^k(\Omega_0)}$ is the usual norm on the set of class $\mathcal{C}^k$ mappings.

The following proposition, is proved only in two space dimensions in [69]; yet, the proof extends *mutatis mutandis* to the d-dimensional case.

Proposition 4.3. *Let Ω_n a sequence of bounded connected domains of $\mathbb{R}^d$ with $\mathcal{C}^2$ boundary, and for any $n \in \mathbb{N}$, denote Σ_n the corresponding skeleton. If Ω_n converges to a domain Ω in the sense of the $\mathcal{C}^2$ topology, then $\overline{\Sigma_n}$ converges to $\overline{\Sigma}$ in the sense of the Hausdorff distance between compacts subsets of $\mathbb{R}^d$.*

4.2.2 Shape derivative of the signed distance function

Let Ω be a bounded domain with Lipschitz boundary. So as to account for variations of Ω, we rely on *Hadamard's boundary variation method*: we look for variations of Ω of the form

$$\Omega_\theta := (I + \theta)(\Omega),\ \ \theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ \ ||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} \leq 1.$$

In the following, we will be led to impose higher regularity over Ω and the allowed deformations θ, which will be specified when the time comes.

Let us start our study of the dependence of the signed distance function on the domain with the following elementary result:

Lemma 4.1.

1. *Let A, B two closed subsets of $\mathbb{R}^d$, and $x \in \mathbb{R}^d$. The Euclidean distance function fulfills the following Lipschitz-like inequality:*
$$|d(x, A) - d(x, B)| \leq d^H(A, B),$$
where $d^H(A, B) := \max(\rho(A, B), \rho(B, A))$ is the Hausdorff distance between A an B, introducing the notation $\rho(A, B) := \max_{x\in A} d(x, B)$.

2. *Let $x \in \mathbb{R}^d$ be a fixed point. The Euclidean distance functions $\theta \mapsto d(x, \partial\Omega_\theta)$, $\theta \mapsto d(x, \overline{\Omega_\theta})$ and $\theta \mapsto d(x, {}^c\Omega_\theta)$, from $B(0,1) \subset W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ into $\mathbb{R}$ are 1-Lipschitz.*
3. *Let $x \in \mathbb{R}^d$ be given. The signed distance function $\theta \mapsto d_{\Omega_\theta}(x)$ to x is 1-Lipschitz as a function from $B(0,1) \subset W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ into $\mathbb{R}$.*

Proof. 1. Without loss of generality, assume $d_A(x) \geq d_B(x)$. Then, there exists $b \in B$ such that, for all $a \in A$, one has :

$$|d(x,A) - d(x,B)| = d(x,A) - d(x,B) \leq d(x,a) - d(x,b) \leq d(a,b).$$

Since this is true for every $a \in A$, it comes from the definitions that:

$$|d(x,A) - d(x,B)| \leq d(b,A) \leq \max_{b \in B} d(b,A) = \rho(B,A),$$

whence the result.

2. Let us prove the result for $B(0,1) \ni \theta \mapsto d(x, \partial\Omega_\theta)$. The other points will follow in the same way. For θ_1, θ_2 small enough so that $(I + \theta_1)$, $(I + \theta_2)$ are Lipschitz diffeomorphisms of $\mathbb{R}^d$, using point (1), we have :
$$|d(x, \partial\Omega_{\theta_1}) - d(x, \partial\Omega_{\theta_2})| \leq d^H(\partial\Omega_{\theta_1}, \partial\Omega_{\theta_2}).$$
Now, any point $y \in \partial\Omega_{\theta_1}$ can be written under the form $y = z + \theta_1(z)$, for some $z \in \partial\Omega$. Thus,
$$\rho(\partial\Omega_{\theta_1}, \partial\Omega_{\theta_2}) = \sup_{z \in \partial\Omega} d(z + \theta_1(z), \partial\Omega_{\theta_2}) \leq \sup_{z \in \partial\Omega} |z + \theta_1(z) - (z + \theta_2(z))| \leq ||\theta_1 - \theta_2||_{W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)}.$$
Estimating $\rho(\partial\Omega_{\theta_2}, \partial\Omega_{\theta_1})$ the same way, point (2) is proved.
3. First, note that, as a direct consequence of point (2), and of the fact that $d_\Omega(x) = d(x, \overline{\Omega}) - d(x, {}^c\Omega)$ for any open domain Ω, the application $\theta \mapsto d_{\Omega_\theta}(x)$ is 2-Lipschitz (hence continuous). Let θ_1, θ_2 be two elements in $B(0,1) \subset W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$. Two cases arise:
– If $d_{\Omega_{\theta_1}}(x)$ and $d_{\Omega_{\theta_2}}(x)$ have same sign, one simply has:
$$|d_{\Omega_{\theta_1}}(x) - d_{\Omega_{\theta_2}}(x)| = |d(x, \partial\Omega_{\theta_1}) - d(x, \partial\Omega_{\theta_2})|,$$
and the result follows from point (2).
– On the contrary, if $d_{\Omega_{\theta_1}}(x)$ and $d_{\Omega_{\theta_2}}(x)$ have different signs, there exists $t_0 \in [0,1]$ such that $d_{\Omega_{\theta_1 + t(\theta_2 - \theta_1)}}(x) = 0$. Then,
$$\begin{aligned} |d_{\Omega_{\theta_1}}(x) - d_{\Omega_{\theta_2}}(x)| &= d_{\Omega_{\theta_1}}(x) + d_{\Omega_{\theta_2}}(x) \\ &= \left(d_{\Omega_{\theta_1}}(x) - d_{\Omega_{\theta_1 + t_0(\theta_2 - \theta_1)}}(x)\right) + \left(d_{\Omega_{\theta_2}}(x) - d_{\Omega_{\theta_1 + t_0(\theta_2 - \theta_1)}}(x)\right), \\ &\leq t_0 ||\theta_2 - \theta_1||_{W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)} + (1 - t_0) ||\theta_2 - \theta_1||_{W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)} \end{aligned}$$
where the last inequality follows from point (2). This ends the proof.

□

Recall the following result over the *differentiation through a minimum* [105] (Chap 10, th. 2.1):

Theorem 4.1. *Let X be an arbitrary set, $\tau > 0$ and $G : [0,\tau] \times X \to \mathbb{R}$ a functional. Denote, for every $t \in [0,\tau]$, $g(t) := \inf_X G(t,.)$. Assume that the four conditions below are fulfilled :*

1. *For every $t \in [0,\tau]$, the set $X(t) := \{x \in X \ , \ G(t,x) = \inf_X G(t,.)\}$ is nonempty.*
2. *G is differentiable with respect to t at every $(t,x) \in [0,\tau] \times X$.*
3. *for every $x \in X(0)$, the map $t \mapsto \frac{\partial G}{\partial t}(t,x)$ is upper semicontinuous at $t = 0$.*

4. (sequential semi-continuity for the set-valued function) There exists a topology $\{\mathcal{U}_i\}_{i\in I}$ over X such that, for any sequence $\{t_n\} \subset [0,\tau]$, $t_n \to 0$, there exists $x_0 \in X(0)$, a subsequence $\{t_{n_k}\}$ of $\{t_n\}$, and elements $x_{n_k} \in X(t_{n_k})$ such that :
- *$x_{n_k} \to x_0$ for the $\{\mathcal{U}_i\}_{i\in I}$ - topology over X.*
- *$\lim \inf_{\substack{k\to\infty \\ t\downarrow 0}} \dfrac{\partial G}{\partial t}(t, x_{n_k}) \geq \dfrac{\partial G}{\partial t}(0, x_0)$*

then there exists $x_0 \in X(0)$ such that

$$\frac{dg}{dt}(0^+) = \inf_{x\in X} \frac{\partial G}{\partial t}(0,x) = \frac{\partial G}{\partial t}(0,x_0).$$

The shape differentiability of the signed distance function is now assessed by the following proposition.

Proposition 4.4. *Assume Ω is Lipschitz, and let $\theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ a deformation direction. Fix a point $x \in \mathbb{R}^d$. Then the function $\mathbb{R}_+ \ni t \mapsto d_{\Omega_{t\theta}}(x)$ is right-differentiable at $t = 0$, and, if $x \in \Omega$,*

$$\frac{d}{dt}\left(d_{\Omega_{t\theta}}(x)\right)\Big|_{t=0^+} = \sup_{y\in\Pi_{\partial\Omega}(x)} \left(\left(\frac{y-x}{d_\Omega(x)}\right)\cdot\theta(y)\right). \tag{4.3}$$

whereas, if $x \in \mathbb{R}^d \setminus \overline{\Omega}$,

$$\frac{d}{dt}\left(d_{\Omega_{t\theta}}(x)\right)\Big|_{t=0^+} = \inf_{y\in\Pi_{\partial\Omega}(x)} \left(\left(\frac{y-x}{d_\Omega(x)}\right)\cdot\theta(y)\right). \tag{4.4}$$

Proof. We first prove the right-differentiability of the square signed distance function $t \mapsto d_{\Omega_{t\theta}}(x)^2$. To this end, define a functional $G : [0,\tau]\times\partial\Omega \to \mathbb{R}$ as :

$$\forall t \in [0,\tau], y \in \partial\Omega,\ \ G(t,y) = |x-(I+t\theta)(y)|^2.$$

It is clear that $d_{\Omega_{t\theta}}(x)^2 = \inf_{y\in\partial\Omega} G(t,y)$; we will apply theorem 4.1 to G, and must check that hypotheses $(1-4)$ are fulfilled:
- Condition (1) is immediate since $\partial\Omega$ is bounded.
- Condition (2) is also easy, and we have, for every $(t,y) \in [0,\tau]\times\partial\Omega$:

$$\frac{\partial G}{\partial t}(t,y) = -2\left(x-(y+t\theta(y))\right).\theta(y), \tag{4.5}$$

  where $\cdot$ stands for the usual scalar product over $\mathbb{R}^d$.
- Condition (3) holds because of expression (4.5).
- Endow $\partial\Omega$ with the topology induced by that of $\mathbb{R}^d$, and let $\{t_n\}$ be a sequence of real numbers decreasing to 0. For each n, let $y_n \in \partial\Omega$ be any element $y_n \in X(t_n)$. As $\partial\Omega$ is compact, one can extract a subsequence $\{t_{n_k}\}$, with $y_{n_k} \to y_0 \in \partial\Omega$.

  Now assume $y_0 \notin X(0) = \Pi_{\partial\Omega}(x)$, and let $z_0 \in X(0)$. Then, there exists $\alpha > 0$ such that:

$$\inf_{y\in\partial\Omega}|x-y| = |x-z_0| \leq |x-y_0| - 2\alpha$$

  By continuity, for $t > 0$ small enough:

$$\inf_{y\in\partial\Omega}|x-(I+t\theta)(y)| \leq |x-(I+t\theta)(z_0)| \leq |x-(I+t\theta)(y_0)| - \alpha. \tag{4.6}$$

  On the other hand, for k large enough, and by definition of y_{n_k},

$$\begin{array}{rcl} |x-(I+t_{n_k}\theta)(y_0)| & \leq & |x-(I+t_{n_k}\theta)(y_{n_k})| + |y_{n_k}-y_0| + t_{n_k}|\theta(y_{n_k})-\theta(y_0)| \\ & \leq & \inf_{y\in\partial\Omega}|x-(I+t_{n_k}\theta)(y)| + \left(1+t_{n_k}||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)}\right)|y_{n_k}-y_0| \end{array}$$

and since the last term in the right hand side goes to 0 as $k \to \infty$, this is in contradiction with (4.6). Thus, $y_0 \in X(0)$.

It remains to show that $\liminf_{\substack{k\to\infty \\ t\downarrow 0}} \frac{\partial G}{\partial t}(t, y_{n_k}) \geq \frac{\partial G}{\partial t}(0, y_0)$, which in our case amounts to:

$$\liminf_{\substack{k\to\infty \\ t\to 0}} -2\left(x - (y_{n_k} + t\theta(y_{n_k}))\right) \cdot \theta(y_{n_k}) \geq -2(x - y_0) \cdot \theta(y_0).$$

and this is a direct consequence of the fact that $y_{n_k} \to y_0$, and of the continuity of θ.
Thus, Theorem 4.1 allows to conclude that:

$$\frac{d}{dt}\left(d^2_{\Omega_{t\theta}}(x)\right)\Big|_{t=0^+} = 2 \inf_{y\in\Pi_{\partial\Omega}(x)} (y - x) \cdot \theta(y).$$

The other expressions follow easily. □

In case Ω enjoys at least $\mathcal{C}^1$ regularity, this result can be given a more convenient form, using Proposition 4.1, (1):

Corollary 4.1. *Assume that Ω is of class $\mathcal{C}^1$. For any $x \in \mathbb{R}^d$, the function $t \mapsto d_{\Omega_{t\theta}}(x)$ is right-differentiable at $t = 0$, and,*

- *if $x \in \Omega$, $\frac{d}{dt}\left(d_{\Omega_{t\theta}}(x)\right)\big|_{t=0^+} = -\inf_{y\in\Pi_{\partial\Omega}(x)} \theta(y) \cdot n(y)$.*
- *if $x \in {}^c\overline{\Omega}$, $\frac{d}{dt}\left(d_{\Omega_{t\theta}}(x)\right)\big|_{t=0^+} = -\sup_{y\in\Pi_{\partial\Omega}(x)} \theta(y) \cdot n(y)$.*

If furthermore $x \notin \Sigma$, then $\theta \mapsto d_{\Omega_\theta}(x)$ is Gâteaux-differentiable at $\theta = 0$, and its derivative $d'_\Omega(\theta)(x)$ reads:

$$\forall \theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ \ d'_\Omega(\theta)(x) = -\theta(p_{\partial\Omega}(x)) \cdot n(p_{\partial\Omega}(x)).$$

Remarks 4.2.

- These results were already observed in [107] and used (in a completely different situation than that of this chapter) in [108], without complete proof (although they are admittedly not very involved).
- The signed distance function can also be seen as a solution of the following Hamilton-Jacobi equation

$$\begin{cases} |\nabla d_\Omega(x)| = 1 & \text{in } D, \\ d_\Omega(x) = 0 & \text{on } \partial\Omega. \end{cases}$$

  The behavior of the variations of d_Ω with respect to the domain can be retrieved by a formal computation. Indeed, assuming that d_Ω is shape differentiable, taking (formally) the derivative in the above system yields that the directional shape derivative $d'_\Omega(\theta)$ satisfies

$$\begin{cases} \nabla d_\Omega(x) \cdot d'_\Omega(\theta)(x) = 0 & \text{in } D, \\ d'_\Omega(\theta)(x) = -\theta(x) \cdot n(x) & \text{on } \partial\Omega. \end{cases}$$

- In proposition 4.4 and corollary 4.1, we did not consider the case $x \in \partial\Omega$. Actually, it seems unclear what happens then. The proof of proposition 4.4 does not extend as is, and the situation seems more complicated (it is for instance difficult to assess whether x belongs to $\Omega_{t\theta}$ or not). Anyway, we will not need this in the sequel.
- Let us catch a glimpse of the case that $\partial\Omega$ is not smooth near a projection point $y \in \Pi_{\partial\Omega}(x)$. Suppose for instance that y is a 'reentrant corner', which defines a *fan* as regards its influence over the signed distance function (see figure 4.3). Formulae (4.3,4.4) express that the derivative of the signed distance function at x depends on $\theta(y)$ not through its normal component at y (which has no meaning), but rather through its component along the segment $[xy]$.

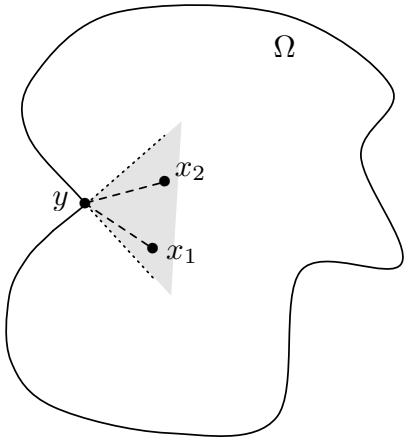


Figure 4.3: The reentrant corner y creates a 'fan' of influence (in grey) over the signed distance function.

We now turn to the differentiability with respect to the domain of some functionals that depend on Ω through the associated signed distance function d_Ω. To this end, we consider a large (bounded) 'computational' domain D which contains Ω and all its variations. We have the following result:

Proposition 4.5. *Assume Ω is a bounded domain of class $\mathcal{C}^1$, and $j : \mathbb{R}^d_x \times \mathbb{R}_s \to \mathbb{R}$ a function of class $\mathcal{C}^1$. Define the functional $J(\Omega)$ as:*

$$J(\Omega) = \int_D j(x, d_\Omega(x))\, dx. \tag{4.7}$$

The application $\theta \mapsto J\left((I+\theta)(\Omega)\right)$, from $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ into $\mathbb{R}$, is Gâteaux-differentiable at $\theta = 0$ and its derivative at Ω reads:

$$J'(\Omega)(\theta) = -\int_D \frac{\partial j}{\partial s}(x, d_\Omega(x))\, \theta(p_{\partial\Omega}(x)).n(p_{\partial\Omega}(x))dx \tag{4.8}$$

Proof. This is merely a use of Lebesgue's dominated convergence theorem. Note that because the skeleton Σ of Ω is of null Lebesgue measure, $p_{\partial\Omega}$ is well-defined at almost every point of D and the above expression does make sense.

To prove the result, since formula (4.8) accounts for a continuous linear form on $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, we only have to prove that, for every sequence of positive real numbers $t_n \to 0$, the ratio $\frac{J(\Omega_{t_n\theta}) - J(\Omega)}{t_n}$ converges to (4.8). From the previous analysis, it is clear that, for almost any point $x \in D$ (actually, for $x \in D \setminus \Sigma$), one has:

$$\frac{j(x, d_{\Omega_{t_n\theta}}(x)) - j(x, d_\Omega(x))}{t_n} \longrightarrow -\frac{\partial j}{\partial s}(x, d_\Omega(x))\theta(p_{\partial\Omega}(x)) \cdot n(p_{\partial\Omega}(x)).$$

Moreover, the domination hypothesis is fulfilled since, for any $x \in D$:

$$\begin{array}{rcl} \left|\dfrac{j(x, d_{\Omega_{t_n\theta}}(x)) - j(x, d_\Omega(x))}{t_n}\right| & \leq & C\, \dfrac{\left|d_{\Omega_{t_n\theta}}(x) - d_\Omega(x)\right|}{t_n} \\ & \leq & 2C||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} \end{array},$$

where the constant C appearing in the first line is a Lipschitz constant for j over the (bounded) set $\left\{(x, d_{\Omega_{t\theta}(x)}) \in D \times \mathbb{R},\ t \text{ small enough}\right\}$, and the second line is a consequence of lemma 4.1, (3).

Hence, Lebesgue's dominated convergence theorem can be applied, and the desired result follows. □

Remark 4.3. The very same proof shows that the signed distance function is Gâteaux-differentiable with respect to the domain when seen as a $L^p_{loc}(\mathbb{R}^d)$-valued function, with $1 \leq p < +\infty$. More precisely, for such p, and for a fixed bounded domain D, and a fixed $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, the function $\theta \mapsto d_{\Omega_\theta} \in L^p(D)$ is Gâteaux-differentiable at 0. As we shall see soon, this happens to be wrong in the critical case $p = \infty$.

Using the same recipe as in the proof of proposition 4.4, one can differentiate other geometric functionals with respect to the domain, e.g. the *diameter* functional:

Proposition 4.6. *Suppose $\Omega \subset \mathbb{R}^d$ is a bounded domain with (only) Lipschitz boundary. Denote*

$$\mathrm{diam}(\Omega) := \sup_{x,y\in\Omega} |x-y| = \sup_{x,y\in\partial\Omega} |x-y|$$

the diameter of Ω. The function $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \ni \theta \mapsto \mathrm{diam}(\Omega_\theta)$ is directionally differentiable at $\theta = 0$, and its directional derivative reads for any and $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$:

$$\frac{d}{dt}\left(\mathrm{diam}(\Omega_{t\theta})\right)\Big|_{t=0+} = \sup_{\substack{(y,z)\in\partial\Omega^2 \\ ||y-z||=diam(\Omega)}} (\theta(y)\cdot n(y) + \theta(z)\cdot n(z)).$$

4.2.3 Another expression for these derivatives

The purpose of this section is to present another expression of the shape derivatives produced by proposition 4.5, which will come in handy later on, in section 4.4.2, in the search for descent directions for functionals which depend on the domain through the signed distance function. In this section, we are especially interested in integrals of the form:

$$\int_D \varphi(x)\, dx,$$

where $D \subset \mathbb{R}^d$ is a large (bounded) domain enclosing the (still bounded and Lipschitz) domain Ω under consideration, and φ presents 'simple' variations along the rays $\mathrm{ray}_{\partial\Omega}(y)$, $y \in \partial\Omega$ (compare formula (4.8) with proposition 4.1).

Let us start with the following definition:

Definition 4.4. *Let $m \geq n$, and $f : \mathbb{R}^m \to \mathbb{R}^n$ a differentiable function at a point $x \in \mathbb{R}^m$. The Jacobian $\mathrm{Jac}(f)(x)$ of f at x is defined as*

$$\mathrm{Jac}(f)(x) := \sqrt{det(\nabla f(x)\, \nabla f(x))^T}$$

Note that $\mathrm{Jac}(f)(x) > 0$ if an only if $\nabla f(x)$ is of maximum rank, that is n, i.e. if and only if the differential of f at x is a surjective map from $\mathbb{R}^m$ to $\mathbb{R}^n$.

The key ingredient in the present discussion is a coarea formula (see [75]):

Theorem 4.2. *Let X, Y be two smooth Riemannian manifolds of respective dimensions $m \geq n$, and $f : X \to Y$ a surjective map of class $\mathcal{C}^1$, whose differential $d_x f : T_x X \to T_{f(x)} Y$ is surjective for almost every $x \in X$. Let φ an integrable function over X. Then:*

$$\int_X \varphi(x)dx = \int_Y \left(\int_{z\in f^{-1}(y)} \varphi(z) \frac{1}{\mathrm{Jac}(f)(z)}\, dz \right) dy$$

Remark 4.4. From Sard's theorem, almost every $y \in Y$ is a regular value of f, and $f^{-1}(y)$ is a submanifold of X, so that the right-hand side of the above expression is well-defined.

We intend to apply this formula in our context to $X = \Omega$, $Y = \partial\Omega$ and $f = p_{\partial\Omega}$. To do so, we need $p_{\partial\Omega}$ to be smooth enough. This is the purpose of the following lemma.

Lemma 4.2. *Assume that Ω enjoys $\mathcal{C}^2$ regularity, and let $x \in \mathbb{R}^d \backslash \overline{\Sigma}$. The projection map $p_{\partial\Omega}$ is differentiable at x and, in the orthonormal basis $\{e_1, ..., e_{d-1}, n\}\,(p_{\partial\Omega}(x))$ of $\mathbb{R}^d$ (see Figure 4.4), its gradient is the $d \times d$ diagonal matrix*

$$\nabla p_{\partial\Omega}(x) = \begin{pmatrix} 1 - \frac{d_\Omega(x)\kappa_1(p_{\partial\Omega}(x))}{1+d_\Omega(x)\kappa_1(p_{\partial\Omega}(x))} & 0 & ... & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & 1 - \frac{d_\Omega(x)\kappa_{d-1}(p_{\partial\Omega}(x))}{1+d_\Omega(x)\kappa_{d-1}(p_{\partial\Omega}(x))} & 0 \\ 0 & ... & 0 & 0 \end{pmatrix}. \tag{4.9}$$

Proof. The proof starts from the characterization of the projection map when $x \in D \setminus \Sigma$ (see Lemma 4.1)

$$p_{\partial\Omega}(x) = x - d_\Omega(x)\nabla d_\Omega(x).$$

This last equality can then be differentiated once more for $x \in D \setminus \overline{\Sigma}$, leading to:

$$\nabla p_{\partial\Omega}(x) = I - \nabla d_\Omega(x)\, \nabla d_\Omega(x)^T - d_\Omega(x)\mathcal{H}d_\Omega(x). \tag{4.10}$$

Since $\nabla d_\Omega(x) = n(p_{\partial\Omega}(x))$, a simple calculation ends the proof. □

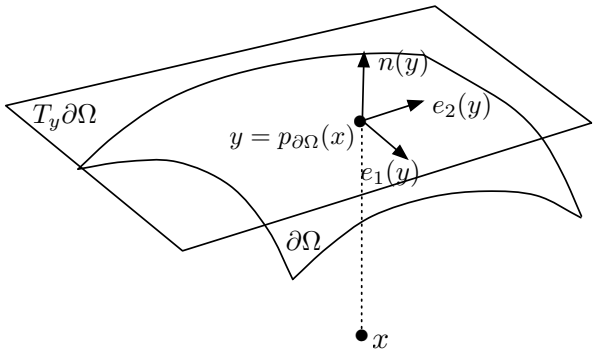


Figure 4.4: Principal directions, normal vector at the projection point $y = p_{\partial\Omega}(x)$ of $x \in \mathbb{R}^d$.

We are now in position to get to the main result of this section.

Corollary 4.2. *Let $\Omega \subset D$ be a $\mathcal{C}^2$ bounded domain, and let φ an integrable function over D. Then,*

$$\int_D \varphi(x)dx = \int_{\partial\Omega} \left(\int_{\text{ray}_{\partial\Omega}(y)\cap D} \varphi(z) \prod_{i=1}^{d-1} (1 + d_\Omega(z)\kappa_i(y))dz \right) dy, \tag{4.11}$$

where z denotes a point in the ray emerging from $y \in \partial\Omega$ and dz is the line integration along that ray.

Proof. Since $\overline{\Sigma}$ is of null Lebesgue measure, we have

$$\int_D \varphi(x)dx = \int_{D\setminus\overline{\Sigma}} \varphi(x)dx.$$

Applying Lemmas 4.2 and 4.2, $p_{\partial\Omega}$ is a surjective and differentiable map from $D \setminus \overline{\Sigma}$ into $\partial\Omega$, with a positive finite Jacobian for any $x \in D \setminus \overline{\Sigma}$

$$\mathrm{Jac}\left(p_{\partial\Omega}\right)(x) = \frac{1}{\prod_{i=1}^{d-1}\Big(1 + d_\Omega(x)\kappa_i(p_{\partial\Omega}(x))\Big)}.$$

Proposition 4.2 then yields the desired result. □

Remark 4.5. The result of corollary 4.2 echoes to other very similar formulae in the literature (see for instance [133]), but does not restrict to tubular neighborhoods of $\partial\Omega$ as for the domain of integration D. Actually, in this chapter (and manuscript), we shall only use corollary 4.2 in this restricted context. See nevertheless [228] for a more advanced use.

This point marks the end of the material about the signed distance function which we shall need from section 4.4, and the remainder of section 4.2 is devoted to several brief topics that may find an interest in themselves, and which are used - for one part - in [228].

4.2.4 More differentiability results for the signed distance function

Proposition 4.5, together with the subsequent remark, claims that the signed distance function is Gâteaux-differentiable as a L^p_{loc}-valued function of the domain ($1 \leq p < \infty$), namely, Ω being a Lipschitz bounded domain, for any fixed relatively compact open set $D \subset\subset \mathbb{R}^d$, the application

$$\begin{array}{ccc} W^{1,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right) & \longrightarrow & L^p(D) \\ \theta & \longmapsto & d_{\Omega_\theta} \end{array}$$

is Gâteaux-differentiable at 0. This naturally raises the question of whether this result stands in higher regularity spaces than L^p_{loc}, $p < \infty$. in this section, we are going to see that, in utter generality, the answer is negative.

4.2.4.1 The signed distance function is not differentiable as a L^∞_{loc}-valued function of the domain

Consider the following counterexample: $\Omega = B(0,1) \subset \mathbb{R}^2$ is the unit open ball in the plane, whose skeleton is simply $\Sigma = \{0\}$. As for deformation field θ, define

$$\forall z = (x,y) \in \mathbb{R}^2,\ \theta(z) = (0,1).$$

Choose eventually any computational domain $D \subset\subset \mathbb{R}^2$ that encloses Σ. For $t > 0$ sufficiently small, the signed distance function $d_{\Omega_{t\theta}}$ to $\Omega_{t\theta}$ can be computed explicitly:

$$\forall z = (x,y) \in \mathbb{R}^2,\ d_{\Omega_{t\theta}}(z) = \sqrt{x^2 + (y-t)^2} - 1.$$

For any point $z \notin \Sigma$, we know from proposition 4.4 that $t \mapsto d_{\Omega_{t\theta}}(z) \in \mathbb{R}$ is differentiable at 0 and:

$$\left.\frac{d}{dt}\left(d_{\Omega_{t\theta}}(z)\right)\right|_{t=0^+} = -\theta(p_{\partial\Omega(z)}) \cdot n(p_{\partial\Omega}(z)).$$

Then $t \mapsto d_{\Omega_{t\theta}} \in \mathbb{R}$ is *not* right-differentiable as a $L^\infty(D)$-valued function, for if it was, the quotient

$$r(t,z) := \frac{d_{\Omega_{t\theta}}(z) - d_\Omega(z) + t\,\theta(p_{\partial\Omega}(z)) \cdot n(p_{\partial\Omega}(z))}{t}$$

would converge to 0 uniformly on D as $t \to 0$, which is not the case, as we are going to see now.

For $z \notin \Sigma$, as simple computation yields:

$$\begin{aligned} r(t,z) &= \frac{\sqrt{x^2+(y-t)^2}-\sqrt{x^2+y^2}}{t} + \theta(p_{\partial\Omega}(z)).n(p_{\partial\Omega}(z)) \\ &= \frac{\sqrt{x^2+(y-t)^2}-\sqrt{x^2+y^2}}{t} + \frac{y}{\sqrt{x^2+y^2}} \end{aligned}$$

Intuitively, the lack of uniform convergence near 0 is due to the fact that, for any $t > 0$ - however small - there are some points $z \in D$ which do not lie 'on the same side' of the skeletons of Ω and $\Omega_{t\theta}$, and then whose projection $p_{\partial\Omega_{t\theta}}(z)$ change suddenly. As there are 'few' such points, they are unseen by the L^p norms, for $p < \infty$, but impede the uniform norm of $r(t,.)$ to converge to 0 as $t \to 0$.

Concretely, consider the sequence of points $z_n := (0, \frac{t_n}{2})$ (see figure 4.5). One has, for any $n \in \mathbb{N}$:

$$\begin{aligned} ||r(t_n,.)||_{L^\infty(\mathcal{D})} &\geq |r(t_n, z_n)| \\ &= \frac{\sqrt{\frac{(-t_n)^2}{4}}-\sqrt{\frac{t_n^2}{4}}}{t_n} + 1 \\ &= 1 \end{aligned}$$

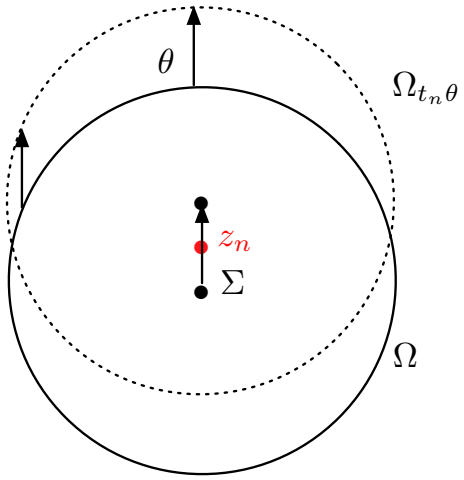


Figure 4.5: Non-differentiability of the signed distance function as a L^∞_{loc}-valued function

Actually, we will see that, upon some regularity assumptions on the considered domain Ω and allowed deformations θ, this phenomenon of 'crossing of the skeleton' is the only obstruction to the derivation of the signed distance function in more regular functional spaces.

4.2.4.2 Shape differentiability of the projection application $p_{\partial\Omega}$

Let us now study the shape differentiability of the projection application $\Omega \mapsto p_{\partial\Omega}(x)$, for a *fixed* point $x \in \mathbb{R}^d \setminus \Sigma$. The main result of this paragraph is the following:

Proposition 4.7. *Let $\Omega \subset \mathbb{R}^d$ a bounded domain of class $\mathcal{C}^2$, and consider a smooth variation of Ω, $\theta \in \mathcal{C}^{2,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right) := \mathcal{C}^2\left(\mathbb{R}^d, \mathbb{R}^d\right) \cap W^{2,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right)$. Let $x \in \mathbb{R}^d \setminus \overline{\Sigma}$, where Σ denotes the skeleton of Ω.*

Then there exists $\alpha > 0$ such that the function $t \mapsto p_{\partial\Omega_{t\theta}}(x)$ is well-defined and $\mathcal{C}^1$-differentiable over $(0, \alpha)$. Besides,

$$\frac{d}{dt}\left(p_{\partial\Omega_{t\theta}}(x)\right)\Big|_{t=0} = (\theta(z)\cdot n(z))\, n(z) + d_\Omega(x)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\nabla_{\partial\Omega}(\theta.n)(z),$$

where $z := p_{\partial\Omega}(x)$, and $\nabla_{\partial\Omega}$ stands for the tangential gradient on $\partial\Omega$.

Remark 4.6. Thanks to proposition 4.2, we know that for $x \notin \overline{\Sigma}$, $-\kappa_i(p_{\partial\Omega}(x))d_\Omega(x) < 1$, $i = 1, ..., d$. Hence, because the Hessian matrix of d_Ω at any point of the boundary matches with the matrix of the second fundamental form of $\partial\Omega$, the matrix $(I + d_\Omega(x)\mathcal{H}d_\Omega(z))$ is invertible and the above expression makes sense.

Proof. The point $x \in \mathbb{R}^d \setminus \overline{\Sigma}$, and $\theta \in \mathcal{C}^{2,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right)$ being fixed, denote as $z = p_{\partial\Omega}(x)$. Moreover, to keep expressions as simple as possible in this proof, we take the shortcuts $d_t := d_{\Omega_{t\theta}}$, $p_t = p_{\partial\Omega_{t\theta}}$ and $n_t = n_{\Omega_{t\theta}}$, the unit normal vector field to $\Omega_{t\theta}$, pointing outward.

Recall that, because Ω is of class $\mathcal{C}^2$, proposition 4.1 claims that there exists a tubular neighborhood W of $\partial\Omega$, such that ∇d_Ω is a well-defined, unitary vector field of class $\mathcal{C}^1$ on W, such that $\forall y \in \partial\Omega$, $\nabla d_\Omega(y) = n(y)$ (a similar operation holds for $\Omega_{t\theta}$). In the following, this unit extension of the normal vector field is still denoted as $n : W \to \mathcal{S}^1$.

Owing to proposition 4.1, we may restrict t (say $|t| < \alpha$) so that, for any such t, $\partial\Omega_{t\theta} \subset W$.

Eventually, recall that, if $y \in \partial\Omega$, because of the regularity assumptions made on Ω and θ, the normal vector field n_t at $(I + t\theta)(y)$ reads (see [234], sec. 4.3.3) :

$$n_t((I + t\theta)(y)) = \frac{\mathrm{com}\left(I + t\nabla\theta(y)\right) n(y)}{\left|\mathrm{com}\left(I + t\nabla\theta(y)\right) n(y)\right|} \tag{4.12}$$

• *First step : existence and differentiability of $t \mapsto p_t(x)$, for t small enough.*

Consider the function $F :]-\alpha, \alpha[\times W \to \mathbb{R}^d$ defined as :

$$F(t, y) = y + d_t(x)n_t(y) - x.$$

Because of the regularity assumptions we have made on the data at hand, F is a function of class $\mathcal{C}^1$ (even if it means decreasing $\alpha > 0$). Indeed, because of proposition 4.3, for t close to 0, x does not belong to the skeleton of $\Omega_{t\theta}$, the function $t \mapsto d_t(x)$ is continuously differentiable in a vicinity of 0, and $t \mapsto p_t(x)$ is well-defined (i.e. $\Pi_{\partial\Omega_{t\theta}}$ is a singleton).

Our purpose is to characterize $p_t(x)$ as the unique zero of $F(t, .)$ over W, then to apply the implicit function theorem to get the desired differentiability result. Proposition 4.1 (1) guarantees that, if $y \in \Pi_{\Omega_{t\theta}}(x)$, then $F(t, y) = 0$. However, the converse may not hold, since $F(t, y) = 0$ does not necessarily imply that $y \in \partial\Omega_{t\theta}$.

For now, let us apply the implicit function theorem to F at point $(0, p_{\partial\Omega}(x))$. The Jacobian matrix of the partial application $y \mapsto F(0, y)$ at $p_{\partial\Omega}(x)$ reads :

$$\nabla_y F(0, p_{\partial\Omega}(x)) = I + d_\Omega(x)\mathcal{H}d_\Omega(p_{\partial\Omega}(x)).$$

Since $x \in \mathbb{R}^d \setminus \overline{\Sigma}$, this matrix is invertible (see remark 4.6). As a consequence, there exists a neighborhood $V \subset \mathbb{R}^d$ of $p_{\partial\Omega}(x)$, a real $\alpha > 0$ (maybe smaller than the previous one), as well as a $\mathcal{C}^1$ function $g : (-\alpha, \alpha) \to V$ such that :

$$\forall t \in]-\alpha, \alpha[,\ y \in V,\ F(t, y) = 0 \Leftrightarrow y = g(t).$$

Of course, $g(0) = p_{\partial\Omega}(x)$. Now, we need to identify $g(t)$ with $p_t(x)$. To this end, we make use of the following elementary lemma:

Lemma 4.3. *Let Ω a bounded domain, Σ its skeleton. Let $x \in \mathbb{R}^d \setminus \Sigma$, $p_{\partial\Omega}(x)$ its projection point. Let $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$. Then, for any open neighborhood V of $p_{\partial\Omega}(x)$, there exists $\alpha > 0$ such that*

$$\forall |t| < \alpha,\ \Pi_{\partial\Omega_{t\theta}}(x) \subset V.$$

Proof. This is a direct consequence of the continuity with respect to the domain of the signed distance function (whose minimum is achieved by projection points). Suppose the contrary holds, that is, there exists a sequence $t_n \to 0$, and $p_n \in \Pi_{\partial\Omega_{t_n\theta}}(x)$, $p_n \notin V$. Up to a subsequence (still denoted t_n), p_n converges towards a point $q \in \partial\Omega$, $q \notin V$. Moreover,

$$d^2_{t_n}(x) = |x - p_n|^2.$$

As $d^2_{t_n}(x) \to d^2_\Omega(x)$, and $|x - p_n|^2 \to |x - q|^2$, we conclude that q is a projection point of x onto $\partial\Omega$, which is impossible because the unique such point belongs to V. □

Remark 4.7. Lemma 4.3 actually is a manifestation of a more general phenomenon (which is proved using exactly the above argument), namely the fact that the set of minimum points $M(t) := \{y_0,\ G(t, y_0) = \inf_y G(t, y)\}$ of a continuous function $G(t, y)$ is *upper semi-continuous* as a set-valued function.

Thus, decreasing α if need be, we may assume that for $|t| < \alpha$, $\Pi_{\partial\Omega_{t\theta}}(x) \subset V$, but we have seen that, for any $|t| < \alpha$, there is exactly one point $y \in V$ such that $F(t, y) = 0$, which we called $g(t)$, so that $p_t(x) = g(t)$. As a consequence, for $|t| < \alpha$, the mapping $t \mapsto p_t(x)$ is of class $\mathcal{C}^1$.

• *Second step : computation of $\frac{d}{dt}(p_t(x))$.*

We know that, for t small enough : $F(t, p_t(x)) = 0$. As F and $t \mapsto p_t(x)$ are regular enough, differentiating at $t = 0$ yields :

$$\partial_t F(0, z) + \nabla_y F(0, z) \cdot \frac{d}{dt}(p_t(x))\Big|_{t=0} = 0.$$

Now, for a point $y \in \partial\Omega$, we compute $\partial_t F(0, y)$:

$$\begin{array}{rcl} \partial_t F(0,y) & = & \frac{d}{dt}(d_t(x))\big|_{t=0}\ n(y) + d_\Omega(x)\ \frac{d}{dt}(n_t(y))\big|_{t=0} \\ & = & -\left(\theta(z).n(z)\right) n(y) + d_\Omega(x)\ \frac{d}{dt}(n_t(y))\big|_{t=0}. \end{array}$$

Now, the term $\frac{d}{dt}(n_t(y))\big|_{t=0}$ is a bit tedious to deal with. It corresponds to the *Eulerian* derivative of the (extended) normal vector field. It is more convenient to deal first with its *Lagrangian* derivative (remember the discussion in chapter 2, §2.2.2.2). More precisely, for t sufficiently small:

$$n_t((I + t\theta)(y)) = \frac{\mathrm{com}\,(I + t\nabla\theta(y))\, n(y)}{|\mathrm{com}\,(I + t\nabla\theta(y))\, n(y)|}. \tag{4.13}$$

Using the matrix identity

$$(I + t\nabla\theta(y))^T \cdot \mathrm{com}\,(I + t\nabla\theta(y)) = \det(I + t\nabla\theta(y))\, I,$$

differentiating at $t = 0$ (which makes sense because all the terms are polynomials in t), we get:

$$\frac{d}{dt}(\mathrm{com}(I + t\nabla\theta(y))\Big|_{t=0} = \mathrm{div}(\theta)(y)\, I - \nabla\theta(y)^T.$$

After some computation, the following expression for the *Lagrangian* derivative of the normal vector field at y follows:

$$\frac{d}{dt}\left(n_t((I+t\theta)(y))\right)\Big|_{t=0} = \left(\nabla\theta(y)^T\, n(y)\right)\cdot n(y))\, n(y) - \nabla\theta(y)^T\, n(y)$$

Thus, the *Eulerian* derivative of the normal at y reads, using the chain rule:

$$\frac{d}{dt}\left(n_t(y)\right)\Big|_{t=0} = \left(\nabla\theta(y)^T\, n(y)\right)\cdot n(y))n(y) - \nabla\theta(y)^T\, n(y) - \nabla n(y)\cdot\theta(y),$$

where the gradient $\nabla n(y)$ has to be understood as that of the smooth extension of the normal to a tubular neighborhood of $\partial\Omega$. Recall eventually that, by definition of the tangential gradient:

$$\begin{array}{rcl} \nabla_{\partial\Omega}(\theta\cdot n)(y) & = & \nabla(\theta\cdot n)(y) - (\nabla(\theta\cdot n)(y)\cdot n(y))n(y) \\ & = & \nabla\theta(y)^T.n(y) + \nabla n(y)^T\cdot\theta(y) - \left((\nabla\theta(y)^T.n(y))\cdot n(y) + (\nabla n(y)^T.\theta(y))\cdot n(y)\right) n(y) \\ & = & \nabla\theta(y)^T.n(y) + \nabla n(y)^T\cdot\theta(y) - \left((\nabla\theta(y)^T\cdot n(y))\cdot n(y)\right) n(y), \end{array} \tag{4.14}$$

because we chose an extension of the normal vector that has a symmetric gradient with $\nabla n(y).n(y) = 0$. We then end up with:

$$\frac{d}{dt}\left(n_t(y)\right)\Big|_{t=0} = -\nabla_{\partial\Omega}(\theta\cdot n)(y), \tag{4.15}$$

and:

$$\partial_t F(0,y) = -\left(\theta(z)\cdot n(z)\right) n(y) - d_\Omega(x)\nabla_{\partial\Omega}(\theta\cdot n)(y).$$

Finally, the derivative of the projection point reads:

$$\begin{array}{rcl} \frac{d}{dt}(p_t(x))\big|_{t=0} & = & -\left(\nabla_y F(0,z)\right)^{-1}\left(\partial_t F(0,z)\right) \\ & = & \left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\left[(\theta\cdot n)(z)n(z) + d_\Omega(x)\nabla_{\partial\Omega}(\theta\cdot n)(z)\right] \\ & = & (\theta\cdot n)(z)\, n(z) + d_\Omega(x)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\nabla_{\partial\Omega}(\theta\cdot n)(z). \end{array} \tag{4.16}$$

□

4.2.4.3 Application to shape differentiability results for the signed distance function in higher regularity spaces

As a first application of proposition 4.7, we can compute the shape derivative of the gradient of the signed distance function.

Corollary 4.3. *Still under the assumption that $\Omega\subset\mathbb{R}^d$ is a bounded domain of class $\mathcal{C}^2$, let $\theta\in\mathcal{C}^{2,\infty}\left(\mathbb{R}^d,\mathbb{R}^d\right)$. Let also $x\in\mathbb{R}^d\setminus\overline{\Sigma}$. Then, function $t\mapsto\nabla d_{\Omega_{t\theta}}(x)$ enjoys $\mathcal{C}^1$ regularity over some interval $(-\alpha,\alpha)$, and its derivative at $t=0$ reads, introducing $z = p_{\partial\Omega}(x)$:*

$$\frac{d}{dt}(\nabla d_{\Omega_{t\theta}}(x))\Big|_{t=0} = \left[-I + d_\Omega(x)\mathcal{H}d_\Omega(z)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\right]\nabla_{\partial\Omega}(\theta\cdot n)(z). \tag{4.17}$$

Proof. Here again, we use the abbreviations of the proof of proposition 4.7: $z = p_{\partial\Omega}(x)$, $d_t := d_{\Omega_{t\theta}}$, $p_t = p_{\partial\Omega_{t\theta}}$ and $n_t = n_{\Omega_{t\theta}}$. Owing to proposition 4.7, the projection application $t\mapsto p_t(x)$ is well-defined over some interval $(-\alpha,\alpha)$. Then, so is $t\mapsto\nabla u_t(x)$ (see proposition 4.1). We know also that, for such values of t:

$$\nabla u_t(x) = n_t(p_t(x));$$

using formulae (4.15) and (4.16), it comes:

$$\begin{array}{rcl} \frac{d}{dt}(\nabla u_t(x))\big|_{t=0} & = & \frac{d}{dt}\left(n_t(z)\right)\big|_{t=0} + \nabla n(z)\cdot\frac{d}{dt}(p_t(x))\big|_{t=0} \\ & = & -\nabla_{\partial\Omega}(\theta\cdot n)(z) + d_\Omega(x)\nabla n(z)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\nabla_{\partial\Omega}(\theta.n)(z) \\ & = & \left[-I + d_\Omega(x)\mathcal{H}d_\Omega(z)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\right]\nabla_{\partial\Omega}(\theta\cdot n)(z). \end{array}$$

□

Remark 4.8. Actually, we could get a little bit more regularity at this point. Indeed, putting $x \notin \overline{\Sigma}$ as a parameter in the application of the implicit function theorem, in the proof of proposition 4.7, we could get that $(t, y) \mapsto \nabla u_t(y)$ is actually differentiable in a vicinity of $(0, x)$.

From the last corollary, a higher differentiability result for d_Ω can be obtained. To this end, we will need the following lemma, which is a nothing but 'Schwartz property' for the derivatives of d_t, with respect to t and the space variable $x \notin \overline{\Sigma}$:

Lemma 4.4. *Under the hypothesis of corollary 4.3, if $x \in \mathbb{R}^d \setminus \overline{\Sigma}$, function $\mathbb{R}^d \ni x \mapsto \frac{d}{dt}(u_t(x))\big|_{t=0}$ is differentiable at x and its gradient reads :*

$$\nabla\left(\frac{d}{dt}(u_t(x))\bigg|_{t=0}\right) = \frac{d}{dt}(\nabla u_t(x))\bigg|_{t=0}.$$

Proof. One last time, we use here the abbreviations of the proof of proposition 4.7. Proposition 4.4 asserts that, for $y \in \mathbb{R}^d$ close to x, function $\mathbb{R}^d \ni y \mapsto \frac{d}{dt}(u_t(y))\big|_{t=0}$ is well-defined and that the following expression holds:

$$\frac{d}{dt}(u_t(y))\bigg|_{t=0} = -\theta(p_{\partial\Omega}(y)).n(p_{\partial\Omega}(y));$$

thus, in order to prove the desired result, we have to differentiate this expression at x, and compare it with (4.17). We get :

$$\begin{aligned}
\nabla\left(\tfrac{d}{dt}(u_t(x))\big|_{t=0}\right) &= -\nabla(\theta(p_{\partial\Omega}(x)))^T \cdot n(z) - \nabla(n(p_{\partial\Omega}(x)))^T \cdot \theta(z) \\
&= -\nabla p_{\partial\Omega}(x)^T\left(\nabla\theta(z)\, n(z) + \nabla n(z).\theta(z)\right) \\
&= -\nabla p_{\partial\Omega}(x)^T\left(\nabla_{\partial\Omega}(\theta.n)(z) + \left((\nabla\theta(z)^T\, n(z)\right)\cdot n(z)\right) n(z)\right),
\end{aligned}$$

where we used formula (4.14) for the tangential gradient. Recall now the expression for the gradient of the projection application (4.10), from which we get easily that $\nabla p_{\partial\Omega}(x)^T \cdot n(z) = 0$ so that the second term in the above expression vanishes. Using also that $\nabla_{\partial\Omega}(\theta.n)(z) \cdot n(z) = 0$, and the fact that expression (4.9) reads, in compact form:

$$\nabla p_{\partial\Omega}(x) = \left[-I + d_\Omega(x)\mathcal{H}d_\Omega(z)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\right],$$

we get eventually:

$$\nabla\left(\frac{d}{dt}(u_t(x))\bigg|_{t=0}\right) = \left[-I + d_\Omega(x)\mathcal{H}d_\Omega(z)\left(I + d_\Omega(x)\mathcal{H}d_\Omega(z)\right)^{-1}\right]\nabla_{\partial\Omega}(\theta.n)(z),$$

which is the sought expression. □

We now end up with the result of interest:

Proposition 4.8. *Let $\Omega \subset \mathbb{R}^d$ a bounded domain of class $\mathcal{C}^2$, with skeleton Σ, and $\theta \in \mathcal{C}^{2,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right)$. For any computational domain $D \subset\subset \mathbb{R}^d \setminus \overline{\Sigma}$, and any $1 \leq p < \infty$, the signed distance function $\theta \mapsto d_{\Omega_\theta}$, from $\mathcal{C}^{2,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ into $W^{1,p}(D)$ is differentiable at $\theta = 0$.*

Proof. In view of proposition 4.4 and the subsequent remarks, the only thing there is to prove is that, for a fixed $\theta \in \mathcal{C}^{2,\infty}(\mathbb{R}^d, \mathbb{R}^d)$:

$$\frac{1}{t}\left|\left|\nabla d_{\Omega_{t\theta}} - \nabla d_\Omega - t\nabla\left(\frac{d}{dt}d_{\Omega_{t\theta}}\bigg|_{t=0}\right)\right|\right|_{L^p(D)} \xrightarrow{t\to 0} 0.$$

Using lemma 4.4, one has:

$$\frac{1}{t}\left|\left|\nabla d_{\Omega_{t\theta}} - \nabla d_\Omega - t\nabla\left(\frac{d}{dt}d_{\Omega_{t\theta}}\bigg|_{t=0}\right)\right|\right|_{L^p(D)} = \left|\left|\frac{\nabla d_{\Omega_{t\theta}} - \nabla d_\Omega}{t} - \frac{d}{dt}(\nabla d_{\Omega_{t\theta}}(x))\bigg|_{t=0}\right|\right|_{L^p(D)}.$$

As hinted at in remark 4.8, function $(t,x) \mapsto \nabla d_{\Omega_{t\theta}}(x)$ is of class $\mathcal{C}^1$ in $(-\alpha,\alpha)\times D$. Hence, the mean value theorem ensures the existence of a constant M such that for t small enough,

$$\forall x \in D,\ \left|\frac{\nabla d_{\Omega_{t\theta}}(x) - \nabla d_{\Omega}(x)}{t}\right| \leq M.$$

This guarantees that the domination hypothesis in Lebesgue's dominated convergence theorem is satisfied, and allows to conclude. □

4.2.4.4 The conclusion of proposition 4.8 does not extend when D intersects $\overline{\Sigma}$

At this point, one could wonder what happens when the integration domain D does intersect the closure $\overline{\Sigma}$ of the skeleton of Ω. In this case, because of the counter-example at the beginning of this section, and owing to Sobolev's embeddings, one cannot hope that $\theta \mapsto d_{\Omega_\theta}$ could be differentiable as a $W^{1,p}(D)$-valued function for any $p \geq d$, which leaves the case $p < d$ open for discussion.

Actually, the counterexample of section 4.2.4.1 as for the lack of shape differentiability of d_Ω as a L^∞-valued function does not allow to conclude in this case. Consider the situation depicted in figure 4.6 below (which is presented with a non-bounded open set Ω, but could easily be brought down to this case).

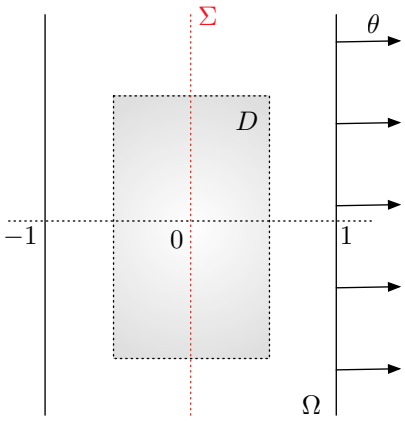


Figure 4.6: Non-differentiability of the signed distance function as a $W^{1,p}_{loc}$-valued function when the domain of integration meets the skeleton.

Take $\Omega := \left\{z=(x,y)\in\mathbb{R}^2,\ -1<x<1\right\}$, so that $\overline{\Sigma} = \left\{z=(x,y)\in\mathbb{R}^2,\ x=0\right\}$, and take for instance $\mathcal{D} = \left\{z=(x,y)\in\mathbb{R}^2,\ -\frac{1}{2}<x<\frac{1}{2},\ -1<y<1\right\}$, with a displacement field θ such that $\theta(z)=(0,0)$ if $x=-1$ and $\theta(z)=(1,0)$ if $x=1$. Intuitively speaking, the skeleton of Ω_θ is translated to the right together with the right part of the boundary.

Elementary computations show that, for $t \geq 0$ and $z \in D$:

$$d_{\Omega_{t\theta}}(z) = \left\{\begin{array}{ccc} x-1-t & \text{if} & x \geq \frac{t}{2} \\ -1-x & \text{if} & x < \frac{t}{2} \end{array}\right. \quad ; \quad \nabla d_{\Omega_{t\theta}}(z) = \left\{\begin{array}{ccc} \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \text{if} & x > \frac{t}{2} \\ \begin{pmatrix} -1 \\ 0 \end{pmatrix} & \text{if} & x < \frac{t}{2} \end{array}\right.$$

Thus, for $z \notin \overline{\Sigma}$, $\frac{d}{dt}(\nabla d_{\Omega_{t\theta}}(z))\big|_{t=0} = 0$, whereas:

$$\frac{\nabla d_{\Omega_{t\theta}}(z) - \nabla d_\Omega(z)}{t} = \begin{cases} 0 & \text{if} \quad x > \frac{t}{2} \text{ or } x < 0 \\ \begin{pmatrix} -\frac{2}{t} \\ 0 \end{pmatrix} & \text{if} \quad 0 < x < \frac{t}{2} \end{cases}$$

Hence,

$$\begin{aligned} \left|\left| \frac{\nabla d_{\Omega_{t\theta}} - \nabla d_\Omega}{t} - \frac{d}{dt}(\nabla d_{\Omega_{t\theta}})\Big|_{t=0} \right|\right|_{L^p(D)} &= \int_{-1}^{1} \left(\int_0^{\frac{t}{2}} \frac{2^p}{t^p} dx \right) dy \\ &= \frac{2^p}{t^{p-1}}, \end{aligned}$$

and this quantity does not converge to 0 as $t \to 0^+$, for $p \geq 1$.

Remark 4.9. This last, negative result indicates that, in 'most cases', functionals $\Omega \mapsto \int_D j(n_\Omega)\, dx$ depending on the domain Ω through the extension of the normal vector field n_Ω given by ∇d_Ω are not differentiable with respect to Ω if the domain D of integration goes as far as $\overline{\Sigma}$ (even if this extension makes sense almost everywhere in D).

4.2.5 Some words around the notion of minimum thickness of a domain

This subsection is essentially independent from the rest of section 4.2, and studies domains $\Omega \subset \mathbb{R}^d$ that are 'not too thin' or whose different parts are 'not too close' from one another. Such requirements around shapes appear naturally in industrial applications of structural optimization; indeed, mechanical structures exhibiting very thin parts may prove very hard and costly to manufacture. See [228] for further motivations and explanations.

To investigate further on this topic, the key notion is that of the *reach* of a set, which we briefly recall now.

4.2.5.1 The reach of a set

Let A an arbitrary *closed* subset of $\mathbb{R}^d$. If $h > 0$, denote $A_h := \left\{ x \in \mathbb{R}^d \,,\, d(x, A) < h \right\}$. The following definition stems from [133].

Definition 4.5. *Let $h > 0$. A closed set $A \subset \mathbb{R}^d$ is said to have reach $\geq h$ provided every point $x \in A_h$ has a unique projection point on A (i.e. the infimum in (4.1) is achieved at a unique point). The supremum of such h is called the reach of A, and denoted as* $\mathrm{reach}(A)$.

Grossly speaking, the *reach* of a set A is the largest value such that each A_h is homeomorphic to A for all $0 \leq h < \mathrm{reach}(A)$. This concept of *reach* is actually closely related to the so-called *uniform exterior ball condition*.

Definition 4.6. *Let $h > 0$. A set $A \subset \mathbb{R}^d$ is said to comply with the exterior ball condition of radius h at point $x \in \partial A$ if there exists a unit vector $\xi \in \mathbb{R}^d$ such that $B(x + h\xi, h) \cap A = \emptyset$. It is said to comply with the uniform exterior ball condition of radius h if the exterior ball condition of radius h is satisfied at each $x \in \partial A$. Symmetrically, A is said to comply with the interior ball condition of radius h at $x \in \partial A$ (resp. uniform interior ball condition of radius h) if $\overline{\mathbb{R}^d \setminus A}$ complies with the exterior ball condition of radius h at x (resp. uniform exterior ball condition of radius h) .*

We easily infer the following property:

Proposition 4.9. *Let $A \subset \mathbb{R}^d$ a closed set, with* $\mathrm{reach}(A) = h > 0$. *Then A satisfies the uniform exterior ball condition with radius h.*

Proof. This proof is a rephrasing of the arguments stated in [94] (section 2 and corollary 4.15).
For some given $r > 0$, $x \in \partial A$, a simple computation shows that a unit vector $\xi \in \mathbb{R}^d$ is such that $B(x + r\xi, r) \cap A = \emptyset$ if and only if it satisfies the so-called *proximal normal inequality*:

$$\forall y \in A,\ \xi \cdot (y - x) \leq \frac{1}{2r}|y - x|^2. \tag{4.18}$$

Now, consider a sequence u_i of points in $\mathbb{R}^d \setminus A$ converging to x, at which $d(., A)$ is differentiable (this is possible because $d(., A)$ is differentiable almost everywhere). Then, for each i, $\Pi_A(u_i) = \{x_i\}$ for some $x_i \in \partial A$, and of course $x_i \to x$. Because $\mathrm{reach}(A) = h$, th. 6.2, (iv) of chap. 6 in [105] (which also follows from a simple computation, reasoning by contradiction) implies that, for every $y \in A$, we have:

$$(u_i - x_i) \cdot (y - x_i) \leq \frac{|u_i - x_i||y - x_i|^2}{2h}. \tag{4.19}$$

Because $u_i \notin A$, up to extraction, one may assume that $x_i \to x$, $u_i \to x$ and $\frac{u_i - x_i}{|u_i - x_i|} \to \xi$, for some unit vector ξ (this argument is the one for the existence of a so-called *proximal normal* at each point of the boundary, in the finite-dimensional context). Then, taking the limit in (4.19), we end up with :

$$\forall y \in A,\ \xi \cdot (y - x) \leq \frac{1}{2h}|y - x|^2,$$

which is exactly saying $B(x + h\xi, h) \cap A = \emptyset$. □

Actually, a far deeper result states that some converse happens to be true in the case when the set A is compact and *epi-lipschitz* (see below), in the more general context of *φ-convexity* [238]. However, we will not need it for our purposes.

4.2.5.2 Sets with minimum thickness

Thanks to this concept, we can now define the intuitive notions of *minimum thickness* and *minimum distance between members* of a domain Ω (see figure 4.7).

Definition 4.7. *Let $\Omega \subset \mathbb{R}^d$ a bounded domain with Lipschitz boundary.*

- *We call minimum thickness of Ω, and denote* $\mathrm{e}(\Omega)$ *the reach of the complementary of Ω, that is*

$$\mathrm{e}(\Omega) = \mathrm{reach}({}^{\mathrm{c}}\Omega).$$

- *We call minimum distance between members of Ω, and denote* $\mathrm{md}(\Omega)$ *the reach of the closure of Ω,*

$$\mathrm{md}(\Omega) = \mathrm{reach}(\overline{\Omega}).$$

We are now interested in domains which answer constraints of minimum thickness and minimum distance between members. To this end, for $h > 0$, define

$$\mathcal{E}_h = \{\Omega \text{ open }, \ \Omega \subset D \ , \ \mathrm{e}(\Omega) \geq h \ , \ \mathrm{md}(\Omega) \geq h\}.$$

We are going to see those sets are compact, in a far stronger way than the sets of domains with uniformly bounded perimeter.

Recall first the following definitions from [172]:

Definition 4.8. – *For $x \in \mathbb{R}^d$, $\xi \in \mathbb{S}^{d-1}$ and $\varepsilon > 0$, the (open) cone $C(x, \xi, \varepsilon)$ of apex x, direction ξ and width ε is defined as*

$$C(x, \xi, \varepsilon) := \left\{y \in \mathbb{R}^d \ , \ \xi \cdot (y - x) > \cos(\varepsilon)|y - x| \ , \ |y - x| < \varepsilon\right\}.$$

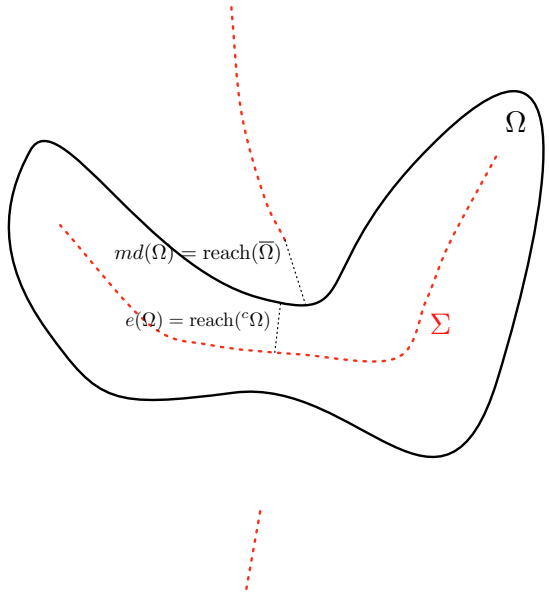


Figure 4.7: Definition of the minimum thickness and distance between members functions

- *An open set $\Omega \subset \mathbb{R}^d$ is said to enjoy the uniform ε-cone property provided for each $x_0 \in \partial\Omega$, there exists a unitary vector $\xi_{x_0} \in \mathbb{R}^d$ such that, for every $x \in \Omega \cap B(x_0, \varepsilon)$, $C(x, \xi_{x_0}, \varepsilon) \subset \Omega$.*
- *An open set $\Omega \subset \mathbb{R}^d$ is said to be epi-lipschitz with constants (r, a, L) if, for every $x_0 \in \partial\Omega$, there exists a local orthonormal frame $\{e_1, ..., e_d\}$ with origin x_0, and a Lipschitz function $\varphi : B_{d-1}(x_0, r) \to (-a, a)$ (where $B_{d-1}(x_0, r)$ is the ball of radius r in $\text{span}\{e_1, ..., e_d\}$ around x_0) with $\varphi(0) = 0$ and Lipschitz constant $\leq L$, such that, denoting $K(r, a) = B_{d-1}(x_0, r) \times (-a, a)$,*

$$\partial\Omega \cap K(r, a) = \left\{(x', x_d) \in \mathbb{R}^{d-1} \times \mathbb{R}\ ,\ x_d = \varphi(x')\right\},$$

$$\Omega \cap K(r, a) = \left\{(x', x_d) \in \mathbb{R}^{d-1} \times \mathbb{R}\ ,\ x_d > \varphi(x')\right\}.$$

Actually, these two notions are equivalent. This is the meaning of the following result (see [172], th. 2.4.7 and the subsequent remark), which quantifies the correspondence between the values of the different constants appearing in the definitions above.

Theorem 4.3. *Let $\Omega \subset \mathbb{R}^d$ a bounded domain. Then Ω enjoys the ε-cone property for some $\varepsilon > 0$ if and only if Ω is epi-lipschitz with some constants (r, a, L). Moreover,*

- *if Ω enjoys the ε-cone property, then r, a, L can be chosen equal to, respectively $\nu\tan(\nu), \nu, \frac{1}{\tan(\nu)}$ for any $\nu < \frac{\varepsilon}{2}$, $\tan^2(\nu) \leq 1$.*
- *conversely, if Ω is epi-lipschitz with constants (r, a, L), the constant ε in the corresponding ε-cone condition can be chosen as $\varepsilon = \min(\frac{r}{2}, \frac{a}{2}, \arctan(\frac{1}{L}))$.*

Consider now a large (yet bounded) computational domain D, and denote:

$$\mathcal{O}_\varepsilon = \{\Omega \text{ open }\ ,\ \Omega \subset D\ ,\ \Omega \text{ enjoys the } \varepsilon\text{-cone property}\}.$$

These sets of domains are well known to enjoy many crucial compactness properties (in the sense of the Hausdorff metric, of γ-convergence,...) [172].

The following theorem expresses the compactness which is brought by the notion of *reach* (see [105], sec. 6, th. 6.6). Note that, in the following, we will only need the fact that the set of domains with uniformly positive reach is closed for the Hausdorff distance, which is actually the difficult part in the demonstration of theorem 4.4.

Theorem 4.4. *Let $h > 0$. The set $\left\{ A \subset \mathbb{R}^d \text{ closed}, \emptyset \neq A \subset \overline{D}, \text{reach}(A) \geq h \right\}$ is compact for the topology induced by the Hausdorff distance between compact subsets of $\mathbb{R}^d$.*

To achieve compactness of the sets $\mathcal{E}_h$, it is then enough to include them in some $\mathcal{O}_\varepsilon$. Actually, 'well-known' results assess that domains enjoying both a uniform and an exterior ball criterion (with some positive radius) are of class $\mathcal{C}^{1,1}$. However, we need to control the uniform cone constant of such an $\Omega \in \mathcal{E}_h$ by the constant h, *independently from* Ω. Reading along the lines of the proof of [30] (th. 5.1.13), the following quantitative result is proved.

Theorem 4.5. *If $\Omega \subset \mathbb{R}^d$ is a bounded domain which satisfies both the uniform exterior and interior ball conditions with radius $h > 0$, then Ω is a domain of class $\mathcal{C}^{1,1}$. In particular, Ω is epi-lipschitz, and the constants r, a, L that appear in definition 4.8 can be chosen as:*

$$r = \lambda h, \quad a = h, \quad L = \sqrt{\left(\left((\sqrt{2} - \lambda)^2 - 1 \right)^{-2} - 1 \right)}$$

for some absolute constant $\lambda > 0$, that depends neither on r, nor on Ω.

Recall that if $\Omega_1, \Omega_2 \subset D$ are open sets, one defines the (complementary) Hausdorff distance between Ω_1 and Ω_2 as

$$d_H(\Omega_1, \Omega_2) := d^H(\overline{D} \setminus \Omega_1, \overline{D} \setminus \Omega_2),$$

where $d^H(K_1, K_2)$ stands for the usual Hausdorff distance between the compact sets K_1, K_2 (see [172], chap. 2 for properties of this metric).

The following result is now a consequence of theorems 4.3, 4.4, and 4.5.

Theorem 4.6. *For every $h > 0$, there exists $\varepsilon > 0$ such that $\mathcal{E}_h$ is a closed subset of $\mathcal{O}_\varepsilon$, for the topology induced by the Hausdorff distance over open subsets of D.*

Proof. First, let us prove the inclusion $\mathcal{E}_h \subset \mathcal{O}_\varepsilon$, for a certain $\varepsilon > 0$. Let $\Omega \in \mathcal{E}_h$; as $\text{reach}(\overline{\Omega}) \geq h$, and $\text{reach}({}^c\Omega) \geq h$, proposition 4.9 implies that Ω satisfies both a uniform interior and exterior ball conditions, with radius h. Consequently, theorem 4.5 implies that Ω is an epi-Lipschitz domain with constants r, a, L only depending on h (through explicit formulae). Thus, because of theorem 4.3, Ω belongs to $\mathcal{O}_\varepsilon$, for some constant ε depending on h through explicit formulae.

Now, let us turn to the closedness of $\mathcal{E}_h$ for the topology induced by the Hausdorff metric over open subsets of D. Let $\Omega_n \in \mathcal{E}_h$ be a sequence of domains converging to a domain $\Omega \subset \mathbb{R}^d$ in the sense of the complementary Hausdorff distance. As each element of the sequence Ω_n lies in $\mathcal{O}_\varepsilon$, using theorem 2.4.10 in [172], one can extract a subsequence Ω_{n_k} such that Ω_{n_k} converges to Ω for the Hausdorff distance between open subsets of D, and $\overline{\Omega_{n_k}}$ and $\partial\Omega_{n_k}$ converge respectively to $\overline{\Omega}$ and $\partial\Omega$ for the Hausdorff distance between compact sets. Because $\text{reach}(\overline{\Omega_{n_k}}) \geq h$, theorem 4.4 implies that $\text{reach}(\overline{\Omega}) \geq h$. On the other hand, the very definition of Hausdorff convergence for open subsets of D means that $\overline{D} \setminus \Omega_{n_k}$ converges to $\overline{D} \setminus \Omega$. Choose a big ball B (or any other bounded convex set) such that $D \subset\subset B$. The properties of Hausdorff distance entail that for any open sets $\Omega_1, \Omega_2 \subset D$, $d^H(\overline{D} \setminus \Omega_1, \overline{D} \setminus \Omega_2) = d^H(\overline{B} \setminus \Omega_1, \overline{B} \setminus \Omega_2)$. Hence, $\overline{B} \setminus \Omega_{n_k}$ also converges to $\overline{B} \setminus \Omega$. What's more, as $\Omega_{n_k} \subset\subset B$ for any k, one gets $\partial\left(\overline{B} \setminus \Omega_{n_k}\right) = \partial B \cup \partial\Omega_{n_k}$, and B being convex,

$$\text{reach}(\overline{B} \setminus \Omega_{n_k}) = \text{reach}({}^c\Omega_{n_k}).$$

Using theorem 4.4, one finally finds that $\text{reach}(\overline{B} \setminus \Omega) = \text{reach}({}^c\Omega)$ is greater than h, hence the desired result. □

As an illustration to how such a result allows to obtain existence results of optimal shapes under constraints of minimum thickness and minimum distance between members, we use here the framework of [172].

Let $D \subset \mathbb{R}^d$ a bounded computational domain, and $f \in L^2(D)$ a source term. For any domain $\Omega \subset D$ with Lipschitz boundary, denote as $u_\Omega \in H_0^1(\Omega)$ the unique solution to the Dirichlet problem:

$$\text{Find } u \in H_0^1(\Omega) \text{ s.t. } \left\{ \begin{array}{rcll} -\Delta u & = & f & \text{in } \Omega \\ u & = & 0 & \text{on } \partial\Omega \end{array} \right. \tag{4.20}$$

Then, the following result (see [172], th. 4.3.1) holds:

Theorem 4.7. *Let $j : D \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ a measurable function such that the partial application $j(x,.,.) : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ is continuous for almost every $x \in D$, and that there exists a constant $C > 0$ with*

$$\forall x \in D,\ r \in \mathbb{R},\ p \in \mathbb{R}^d,\ |j(x,r,p)| \leq C(1 + r^2 + ||p||^2).$$

For any domain $\Omega \subset D$, define the functional

$$J(\Omega) = \int_\Omega j\left(x, u_\Omega(x), \nabla u_\Omega(x)\right) dx.$$

If $\mathcal{O}$ is any subset of $\mathcal{O}_\varepsilon$, for some $\varepsilon > 0$, which is closed for the topology of Hausdorff distance between open subsets of D, then J has a global minimizer over $\mathcal{O}$.

As a consequence of this theorem and of the results of this section, for any $h > 0$, J admits a minimizer over $\mathcal{E}_h$.

4.3 Sharp-interface formulation in a fixed mesh framework

After the fairly long aside of section 4.2 around the signed distance function to a domain, we now turn to the main topic of this chapter, to wit the modeling of multi-phase shape optimization problems.

To simplify the exposition in the first sections we limit ourselves to the case of two materials. Of course, the proposed approach extends to more phases and the corresponding details are given in section 4.5.

4.3.1 Description of the problem

The general purpose of this chapter is to optimize the position of the interface Γ between two linear elastic materials, hereafter labeled as 0 and 1, with respective Hooke's law A_0, A_1. These materials fill two respective subdomains Ω^0, Ω^1 of a (bounded) working domain D of $\mathbb{R}^d$, ($d = 2$ or 3) which accounts for the resulting structure of the optimal distribution of materials, i.e. $D = \Omega^0 \cup \Gamma \cup \Omega^1$. To avoid mathematical technicalities, we assume that Γ is a smooth surface without boundary and strictly included in D, that is, $\Gamma \cap \partial D = \emptyset$. We refer to Ω^1 as the *exterior* subdomain, so that $\partial\Omega^0 = \Gamma$ (see Figure 4.8). Thus, the shape of the interface Γ is altogether conditionned by that of Ω^0, and conversely. In the sequel, the variable of shape optimization is denoted either by Γ or Ω^0, without distinction.

The structure D is clamped on a part $\Gamma_D \subset \partial D$ of its boundary, and is submitted to *body forces* and *surface loads*, to be applied on a part $\Gamma_N \subset \partial D$, which are given as two vector-valued functions defined on D, respectively $f \in L^2(D)^d$, and $g \in H^1(D)^d$.

Perhaps the most natural and physical way to model such a distribution of two materials among a fixed working domain is the so-called *sharp-interface formulation*. More specifically, the total Hooke's law on D

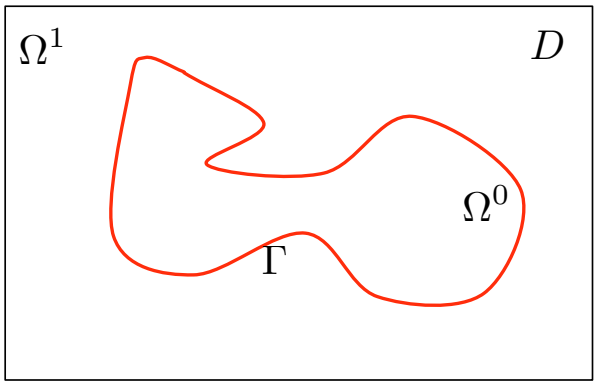


Figure 4.8: Fixed working domain D occupied by two distinct materials Ω^0 and Ω^1 separated by a smooth interface Γ.

is defined as $A_\chi := A_0\chi_0 + A_1\chi_1$, where χ_i stands for the characteristic function of the phase Ω^i. In this context, the *displacement field* u is the unique solution in $H^1(D)^d$ to the linearized elasticity system

$$\left\{\begin{array}{rcll} -\mathrm{div}\,(A_\chi\, e(u)) & = & f & \text{in } D \\ u & = & 0 & \text{on } \Gamma_D \\ (A_1\, e(u))\, n & = & g & \text{on } \Gamma_N, \end{array}\right. \tag{4.21}$$

where $e(u) = \frac{\nabla u^T + \nabla u}{2}$ is the strain tensor, and n stands for the outer unit normal vector to ∂D.

Our purpose is to minimize an objective function of the interface Γ, which is rather expressed as a function $J(\Omega^0)$ of the interior subdomain,

$$J(\Omega^0) = \int_D j(x,u)\,dx + \int_{\Gamma_N} k(x,u)\,ds, \tag{4.22}$$

where $j(x,u)$ and $k(x,u)$ are smooth functions satisfying adequate growth conditions. A typical example is the *compliance* of the structure D (the work done by the loads), which reads

$$J(\Omega^0) = \int_D f\cdot u\,dx + \int_{\Gamma_N} g\cdot u\,ds = \int_D A_\chi(x)e(u) : e(u)dx. \tag{4.23}$$

Of course, the minimization of (4.22) or (4.23) is complemented by a volume constraint on the phase A_0. In particular, it is often a requirement in order to avoid obvious designs made of only one phase.

We do not discuss the well-posedness of this optimal design problem. Let us simply recall that the minimization of (4.22) or (4.23) usually does not admit a solution in the class of open subsets $\Omega_0 \subset D$. Existence of an optimal shape is rather obtained with some additional smoothness or geometrical or topological constraints (for example, imposing a uniform bound on the perimeter of Ω_0, i.e., on the measure of the interface Γ); see for instance the discussion in chapter 2, §2.1.2.

4.3.2 Shape-sensitivity analysis of the sharp-interface problem

There exists a vast literature on the Hadamard method for computing derivatives with respect to the exterior boundary (see e.g. [9], [105], [172], [234] and references therein) but relatively few works on the derivation with respect to an interface between two regions. In the conductivity context (i.e. replacing (4.21)

by a scalar equation), derivatives with respect to an interface have been obtained in [175], [46], [248]. These results were extended to the elasticity setting in [15]. Let us also mention the works [188], [239] where similar results are obtained for a stratified media (where the interfaces are flat and parametrized by a single scalar parameter).

In the present context, we go on relying on Hadamard's method for describing variations of Γ (or equivalently Ω^0). For a smooth open subset $\Omega^0 \subset D$, we consider variations of the type

$$\big(Id+\theta\big)(\Omega^0) := \big\{x+\theta(x) \text{ for } \ x \in \Omega^0\big\}\,,$$

with $\theta \in W^{1,\infty}(D;\mathbb{R}^d)$ such that θ is tangential on ∂D (this last condition ensures that $D = (Id+\theta)D$). It is well known that, for sufficiently small θ, $(Id+\theta)$ is a diffeomorphism in D.

The notion of derivation with respect to the domain of interest is now the following:

Definition 4.9. *The shape derivative of a function $J(\Omega^0)$ is defined as the Fréchet derivative in $W^{1,\infty}(D;\mathbb{R}^d)$ at 0 of the application $\theta \to J\big(\big(Id+\theta\big)\Omega^0\big)$, i.e.*

$$J\big(\big(Id+\theta\big)\Omega^0\big) = J(\Omega^0) + J'(\Omega^0)(\theta) + o(\theta) \quad \textit{with} \quad \lim_{\theta\to 0} \frac{|o(\theta)|}{\|\theta\|_{W^{1,\infty}}} = 0\,,$$

where $J'(\Omega^0)$ is a continuous linear form on $W^{1,\infty}(D;\mathbb{R}^d)$.

As noticed in [15] and [248], the essential ingredients that must be considered in the calculation of the shape derivative of a problem such as (4.21) are the transmission conditions and the differentiability of the solution u with respect to the interface Γ. Furthermore, when a numerical implementation is sought, an additional element must be taken into account: the way in which the transmission conditions (continuity of the displacement and continuity of the normal stress across the interface) are interpreted by finite element methods in a fixed mesh framework. In general these methods either partially preserve the transmission conditions (e.g. classical Lagrange finite elements method) or exactly preserve the transmission conditions (e.g. extended finite elements XFEM [293], adapted interface meshing, which is one of the main topics of this manuscript).

It is known [15], [248] that the solution $u \in H^1(D)$ of (4.21) is not shape differentiable with respect to the interface Γ. The reason is that some spatial derivatives of u are discontinuous across the interface because of the jump of the material elastic properties. Note however that the transported (or pull-back) function $u_\theta := u\circ(Id+\theta)$ is indeed differentiable with respect to θ (this is the difference between the material derivative in the latter case and the shape derivative in the former case, see [9], [172]). It is not necessary to use the concept of material derivative for computing the shape derivative of the objective function. One can stay in a Eulerian framework and use Céa's formal Lagrangian method [72] to find the correct formula for the shape derivative $J'(\Omega^0)(\theta)$. In order to circumvent the non-differentiability of u, the idea is to introduce the restrictions of u on Ω^0 and Ω^1, denoted by $u^0 := u|_{\Omega^0}$ and $u^1 := u|_{\Omega^1}$.

We recall the result of [15] for the shape derivation of the objective function (4.22). We need to introduce some notations about jumps through the interface Γ. For any quantity s which is discontinuous across Γ, taking values s^0 (resp. s^1) on Ω^0 (resp. Ω^1), denote as $[s] = s^1 - s^0$ the *jump* of s. We also introduce at each point of Γ the local basis obtained by gathering the unit normal vector n (pointing outward Ω^0) and a collection of of unit tangential vectors, denoted by τ, such that (τ, n) is an orthonormal frame. For a symmetric $d\times d$ matrix $\mathcal{M}$, written in this basis, we introduce the notation

$$\mathcal{M} = \begin{pmatrix} \mathcal{M}_{\tau\tau} & \mathcal{M}_{\tau n} \\ \mathcal{M}_{n\tau} & \mathcal{M}_{nn} \end{pmatrix}$$

where $\mathcal{M}_{\tau\tau}$ stands for the $(d-1)\times(d-1)$ minor of $\mathcal{M}$, $\mathcal{M}_{\tau n}$ is the vector of the $(n-1)$ first components of the n-th column of $\mathcal{M}$, $\mathcal{M}_{n\tau}$ is the row vector of the $(n-1)$ first components of the n-th row of $\mathcal{M}$, and

$\mathcal{M}_{nn}$ the (n,n) entry of $\mathcal{M}$. Finally, we define the adjoint problem

$$\left\{\begin{array}{rcll} -\mathrm{div}\,(A_\chi\, e(p)) & = & -j'(x,u) & \text{in } D,\\ p & = & 0 & \text{on } \Gamma_D,\\ (A_1\, e(p))\, n & = & -k'(x,u) & \text{on } \Gamma_N, \end{array}\right. \tag{4.24}$$

where the symbol $'$ denotes differentation with respect to u.

Proposition 4.10. *The shape derivative of the cost function J, defined in (4.22), reads*

$$J'(\Omega^0)(\theta) = -\int_\Gamma \mathcal{D}(u,p)\,\theta\cdot n\,ds,$$

$$\mathcal{D}(u,p) = -\sigma(p)_{nn} : [e(u)_{nn}] - 2\sigma(u)_{n\tau} : [e(p)_{n\tau}] + [\sigma(u)_{\tau\tau}] : e(p)_{\tau\tau}. \tag{4.25}$$

where $[\cdot] = \cdot^1 - \cdot^0$ denotes the jump through Γ, $n = n^0 = -n^1$ and $\sigma(v) = A_\chi\, e(v)$.

Remark 4.10. To better appreciate the expression (4.25) where some terms have jumps and others not, we recall that the tangential strain tensors $e(u)_{\tau\tau}$ and $e(p)_{\tau\tau}$ are continuous through the interface Γ while the normal components $e(u)_{nn}$, $e(u)_{n\tau}$, $e(p)_{nn}$ and $e(p)_{n\tau}$ are discontinuous. On the contrary, the normal components of the stress tensors $\sigma(u)_{nn}$, $\sigma(u)_{n\tau}$, $\sigma(p)_{nn}$ and $\sigma(p)_{n\tau}$ are continuous through Γ while their tangential parts $\sigma(u)_{\tau\tau}$ and $\sigma(p)_{\tau\tau}$ are discontinuous.

Proof. We merely sketch the proof that can be found in [15]. In order to apply Céa's Lagrangian method [72] (see the sketch of the method in chapter 2, §2.2.2.4), we first introduce the restrictions of u on Ω^0 and Ω^1, denoted by $u^0 := u|_{\Omega^0}$ and $u^1 := u|_{\Omega^1}$, which satisfy the transmission problem:

$$\left\{\begin{array}{rcll} -\mathrm{div}\left(A_1\, e(u^1)\right) & = & f & \text{in } \Omega^1\\ u^1 & = & 0 & \text{on } \Gamma_D\cap\partial\Omega^1\\ \left(A_1\, e(u^1)\right) n & = & g & \text{on } \Gamma_N\cap\partial\Omega^1\\ u^1 & = & u^0 & \text{on } \Gamma\\ (A_0 e(u^0))n^0 + (A_1 e(u^1))n^1 & = & 0 & \text{on } \Gamma, \end{array}\right. \tag{4.26}$$

and

$$\left\{\begin{array}{rcll} -\mathrm{div}\left(A_0\, e(u^0)\right) & = & f & \text{in } \Omega^0\\ u^1 & = & u^0 & \text{on } \Gamma\\ (A_0 e(u^0))n^0 + (A_1 e(u^1))n^1 & = & 0 & \text{on } \Gamma. \end{array}\right. \tag{4.27}$$

Of course, (4.21) and (4.26)-(4.27) are equivalent. Note that, by standard regularity theory [222], u is smooth on each subdomain, namely $u^0 \in H^2(\Omega^0)$ and $u^1 \in H^2(\Omega^1)$. Then, we define the Lagrangian

$$\begin{aligned} \mathcal{L}(\theta, v^1, v^0, q^1, q^0) &= \sum_{i=0,1}\left(\int_{(Id+\theta)\Omega^i} j(x,v^i)dx + \int_{\Gamma_N} k(x,v^i)ds\right) \\ &+ \sum_{i=0,1}\left(\int_{(Id+\theta)\Omega^i} A_i e(v^i) : e(q^i)dx - \int_{(Id+\theta)\Omega^i} f\cdot q^i dx - \int_{\Gamma_N} g\cdot q^i ds\right) \\ &+ \frac{1}{2}\int_{(Id+\theta)\Gamma} (\sigma^1(v^1) + \sigma^0(v^0))n\cdot(q^1-q^0)ds \\ &+ \frac{1}{2}\int_{(Id+\theta)\Gamma} (\sigma^1(q^1) + \sigma^0(q^0))n\cdot(v^1-v^0)ds, \end{aligned} \tag{4.28}$$

where the last two surface integrals account for the transmission conditions. Differentiating $\mathcal{L}$ with respect to q^1, q^0 yields the state equations (4.26)-(4.27), while differentiating with respect to v^1, v^0 leads to the adjoint equation (4.24). Then a standard, albeit nasty, computation (see [15] for full details) shows that

$$J'(\Omega^0)(\theta) = \frac{\partial\mathcal{L}}{\partial\theta}(0, u^1, u^0, p^1, p^0)(\theta),$$

which yields the result. □

Remark 4.11. Proposition 4.10 can be extended in several ways. For example, if the integrand j depend on χ, namely if the objective function is

$$J(\Omega^0) = \int_D j_\chi(x,u)dx + \int_{\Gamma_N} k(x,u)ds := \sum_{i=0,1} \int_{\Omega^i} j_i(x,u)dx + \int_{\Gamma_N} k(x,u)dx,$$

we obtain a shape derivative which is

$$J'(\Omega^0)(\theta) = -\int_\Gamma \Big([j_\chi(x,u)] + \mathcal{D}(u,p) \Big) \theta \cdot n \, ds,$$

with the same expression (4.25) for $\mathcal{D}(u,p)$.

Although formula (4.25) for the shape derivative makes perfect sense in a continuous setting, its numerical discretization is not obvious. Indeed, (4.25) involves jumps through the interface which are difficult to evaluate from a numerical point of view if the interface is not exactly meshed. Let us explain the difficulty by making some specific discretization choices, keeping in mind that any other numerical method will feature similar drawbacks. Suppose D is equipped with a conforming simplicial mesh $D_h = \bigcup_{i=1}^N K_i$ with N elements K_i of maximal size h. Let $\Pi_1(D_h)$ and $\Pi_0(D_h)$ be the finite-dimensional spaces of Lagrange $\mathbb{P}^1$, respectively $\mathbb{P}^0$, finite element functions. Define $u_h, p_h \in \Pi_1(D_h)$ the internal approximations of u and p respectively, i.e.,

$$\int_D A_\chi e(u_h) : e(v_h) \, dx = \int_D f \cdot v_h \, dx + \int_{\Gamma_N} g \cdot v_h \, ds, \quad \forall v_h \in \Pi_1(D_h), \tag{4.29}$$

and

$$\int_D A_\chi e(p_h) : e(v_h) \, dx = -\int_D j'(x,u_h) \cdot v_h \, dx - \int_{\Gamma_N} k'(x,u_h) \cdot v_h \, ds, \quad \forall v_h \in \Pi_1(D_h). \tag{4.30}$$

Since the discrete strain tensors $e(v_h)$ are constant in each cell K_i, we can replace A_χ in the above internal approximate variational formulation by its $\mathbb{P}^0$ interpolate A^* defined by

$$A^*|_K = \rho A^0 + (1-\rho)A^1, \quad \text{with } \rho = \int_K \chi \, dx.$$

Within this discretized framework the naive evaluation of the jumps in (4.25) has no meaning. Indeed, consider the generic case of an element K cut in its interior by the interface Γ (see Figure 4.9). For $\mathbb{P}^1$ Lagrange finite elements the strain tensors $e(v_h)$, for $v_h = u_h, p_h$, are constant in K, thus yielding a zero jump. Similarly, if the stress tensors are evaluated as $\sigma_h = A^* e(v_h)$, they are constant in K and their jump is again zero, leading to a vanishing shape derivative if formula (4.25) is used with these values ! There is an alternative formula for the stress tensor which is $\sigma_h = A_\chi e(v_h)$: it yields a non-vanishing jump $[A]e(v_h)$ and the discretization of (4.25) would be

$$\big(\mathcal{D}(u,p)\big)_h = \big([A]e(u)\big)_{\tau\tau} : e(p)_{\tau\tau}, \tag{4.31}$$

which is different from the discrete formula (4.33) by lack of any normal components. On the same token, note that the theoretical continuity of the normal stress through Γ does not hold when this approximation is used, for $\sigma_h = A_\chi e(v_h)$ with $v_h = u_h, p_h$ since

$$[\sigma_h \cdot n] = ([A]e(v_h)) \cdot n \neq 0.$$

Henceforth some special care is required for the numerical approximation of (4.25). A complicated process was proposed in [15] for computing the jump of a discontinuous quantity s_h, based on the diffuse interface approximation

$$[s_h] \approx \Big((1-\chi)s_h - \chi s_h \Big). \tag{4.32}$$

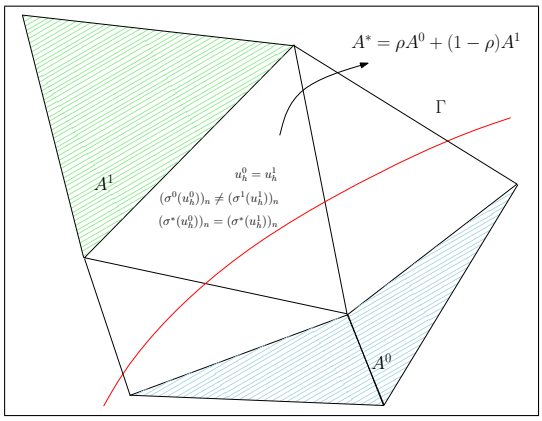


Figure 4.9: Transmission condition in a fixed mesh framework.

Notwithstanding this approximation seems to work well when the contrast between the two elastic phases is very large (as is the case in damage or fracture models, see [15]), more general numerical experiments for comparable elastic moduli indicate a much worse behavior of this approximation, up to the point that (4.25) does not any longer provide a proper descent direction to minimize (4.22) (see section 4.7.2).

This difficulty in the numerical evaluation of the shape derivative (4.25) is just another example of the well-known paradigm 'should we differentiate first and then discretize or *vice versa* ?' as already studied in [240]. In order to get around this issue it is tempting, and we do so now, to investigate the case when we first discretize and then differentiate. In other words we consider the objective function

$$J_h(\Omega^0) = \int_D j(x, u_h)\, dx + \int_{\Gamma_N} k(x, u_h)\, ds,$$

where $u_h \in \Pi_1(D_h)$ is the discrete solution of (4.29).

Proposition 4.11. *Assume that the interface Γ generically cut the mesh D_h, namely that it is never aligned with part of a face of any cell K_i. Then, the solution u_h of (4.29) is shape differentiable and the shape derivative of the cost function J_h is given by*

$$J_h'(\Omega^0)(\theta) = -\int_\Gamma [A_\chi] e(u_h) : e(p_h)\, \theta \cdot n\, ds, \tag{4.33}$$

where $[\cdot]$ denotes the jump through Γ and p_h is the solution of (4.30).

Remark 4.12. Note that Proposition 4.11 holds true for most finite elements discretization and not merely $\mathbb{P}^1$ Lagrange finite elements. The assumption on the interface Γ is necessary in the sense that, if a face of an element K of the mesh is embedded in Γ, then neither u_h nor J_h are shape differentiable (in the most favorable case, there would be two directional derivatives corresponding to Γ moving on one side or on the other of this face of K). However, if instead of Lagrange finite elements, we use Hermite finite elements which ensure that $e(u_h)$ is continuous on D, then the results of Proposition 4.11 hold true without any assumption on Γ.

Proof. Let us denote by $\phi_i(x)$ the basis functions of the finite element space $\Pi_1(D_h)$. The solution $u_h \in \Pi_1(D_h)$ is decomposed as

$$u_h(x) = \sum_i U_i^h \phi_i(x),$$

and the vector U^h of components U_i^h is the solution of the linear system

$$K^h U^h = F^h,$$

where the stiffness matrix K^h and the right hand side F^h are defined as

$$K_{i,j}^h = \int_D A_\chi e(\phi_i) : e(\phi_j)\, dx, \text{ and } F_i^h = \int_D f \cdot \phi_i\, dx + \int_{\Gamma_N} g \cdot \phi_i\, ds.$$

The basis functions ϕ_i are independent of Γ so the shape differentiability of the function u_h reduces to that of the vector U^h and thus of the stiffness matrix K^h. Since the quantity $e(\phi_i) : e(\phi_j)$ is piecewise constant on each element K, we need our assumption that Γ does not overlap any face of K. In such a case we obtain

$$\left(K_{ij}^h\right)'(\Gamma)(\theta) = \int_\Gamma [A_\chi] e(\phi_i) : e(\phi_j)\, \theta \cdot n\, ds$$

and thus, the shape differentiability of U^h is a simple consequence of the implicit function theorem. We end up with:

$$u_h'(\Gamma)(\theta) = \sum_i \left(U_i^h\right)'(\Gamma)(\theta)\phi_i, \text{ where } \left(U^h\right)'(\Gamma)(\theta) = -(K^h)^{-1}\left(K^h\right)'(\Gamma)(\theta)U^h.$$

Once u_h is shape differentiable, it is not necessary anymore to consider a complicated Lagrangian like (4.28), taking into account the transmission conditions through Γ (which, by the way, do not hold true for u_h). Therefore we define a discrete Lagrangian as

$$\mathcal{L}_h(\theta, v_h, q_h) = \int_D j(x, v_h)\, dx + \int_{\Gamma_N} k(x, v_h)\, ds + \int_D A_{(Id+\theta)\chi} e(v_h) : e(q_h)\, dx - \int_D f \cdot q_h\, dx - \int_{\Gamma_N} g \cdot q_h\, ds,$$

to which it is easy to apply Céa's method. Note that the adjoint problem obtained by differentiating $\mathcal{L}_h$ with respect to v_h is exactly (4.30) which was a discretization of the continuous adjoint. Therefore we deduce

$$J_h'(\Omega^0)(\theta) = \frac{\partial \mathcal{L}_h}{\partial \theta}(0, u_h, p_h)(\theta),$$

which yields the desired result. □

There is a clear difference between the discrete derivative (4.33) and the continuous one (4.25). Even if the continuous derivative is further discretized as suggested in (4.31), there is still a difference between (4.33) and (4.31) which is that the last one is restricted to the tangential components of the stress and strain tensors.

There is however one case where both formulas coincide which is when one of the phases is void. Indeed, assume (formally) that $A_0 = 0$ (and similarly that $f = 0$ and $j = 0$ in Ω^0 so that no loads are applied to the void region). Then, in the domain Ω^0 we have

$$\sigma(p)_{nn} = 0,\ \sigma(p)_{n\tau} = 0,\ \sigma(u)_{nn} = 0 \text{ and } \sigma(u)_{n\tau} = 0.$$

Thus, the continuous derivative (4.25) becomes

$$J'(\Omega^0)(\theta) = -\int_\Gamma \sigma(u^1)_{\tau\tau} : e(p^1)_{\tau\tau}\, \theta \cdot n\, ds,$$

which, upon discretization, coincides with the discrete derivative (4.33)

$$J_h'(\Omega^0)(\theta) = -\int_\Gamma A^1 e(u_h) : e(p_h)\, \theta \cdot n\, ds,$$

since $\sigma(u^1)_{nn} = \sigma(u^1)_{n\tau} = 0$ on Γ.

The above study shows that the numerical discretization of the sharp-interface problem should be handled carefully when a standard finite element method is used for solving the state and adjoint systems (4.21) and (4.24) in a fixed mesh setting. The main reason of this difficulty lies in the difference of regularity of the exact and approximated solutions through the interface. The discrete derivative (4.33) proves very efficient in numerical practice. Many examples are given in [110] in the context of optimal design of laminated composite panels.

4.4 Shape derivative in the smoothed-interface context

4.4.1 Description of the problem

We now present an alternative approach to that of section 4.3 which can be coined as smoothed or diffuse interface approach. It can be seen as a mathematically convenient approximation of the sharp-interface problem but, as explained in the introduction, it has its own merits for some problems in material science which feature physically thick transition zones [66], [296], [308], [314]. More precisely, either for a mathematical approximation or for physical reasons, it may be desirable to model the interface Γ between Ω^0 and Ω^1 as a thin layer of (small) width $2\varepsilon > 0$ rather than as a sharp interface. In this context, we rely on the notion of *signed distance function*, described in section 4.2.

The material properties in D are defined as a smooth interpolation between A_0 and A_1 in the layer of width 2ε around Γ, so that the resulting Hooke tensor $A_{\Omega^0,\varepsilon}$ reads

$$A_{\Omega^0,\varepsilon}(x) = A_0 + h_\varepsilon(d_{\Omega^0}(x))(A_1 - A_0), \quad \forall\, x \in D, \tag{4.34}$$

where $h_\varepsilon : \mathbb{R} \to \mathbb{R}$ is a smooth approximation of the Heaviside function, that is, a smooth monotone function enjoying the properties : $h_\varepsilon(t) = 0$ for $t < -\epsilon$, $h_\varepsilon(t) = 1$ for $t > \epsilon$. In the sequel, we chose the $\mathcal{C}^2$ function

$$\forall\, t \in \mathbb{R},\ \ h_\varepsilon(t) = \begin{cases} 0 & \text{if} \quad t < -\varepsilon \\ \frac{1}{2}\left(1 + \frac{t}{\varepsilon} + \frac{1}{\pi}\sin(\frac{\pi t}{\varepsilon})\right) & \text{if} \quad -\varepsilon \le t \le \varepsilon \\ 1 & \text{if} \quad t > \varepsilon \end{cases} \tag{4.35}$$

Remark 4.13. Formula (4.35) expresses a simple choice for the interpolation of the material properties between the two materials, and of course, one could think of different interpolation rules. Moreover, the interpolation function could also contain parameters that are themselves subject to optimization (e.g. the layer width ε) and both a geometric and parametric optimization could be combined using a method of alternating directions.

We modify (4.21) so that the elastic displacement now solves

$$\begin{cases} -\mathrm{div}\left(A_{\Omega^0,\varepsilon}\, e(u)\right) &=& f & \text{in } D \\ u &=& 0 & \text{on } \Gamma_D \\ (A_1\, e(u))\, n &=& g & \text{on } \Gamma_N. \end{cases} \tag{4.36}$$

The objective function does not change and we still minimize (4.22) which depends on d_{Ω^0} through (4.34). In order to compute its shape derivative, we shall use the preliminary material of section 4.2.

4.4.2 Shape derivative of the compliance in the multi-materials setting

With the results of section 4.2, we have all the necessary ingredients to differentiate the cost function (4.22) with respect to the domain. We keep the geometrical assumptions of section 4.3, namely for a given bounded open set $D \subset \mathbb{R}^d$ which is partitioned in two subdomains $\Omega^0, \Omega^1 \subset D$, Ω^0 is a strict subset of D in

the sense that its boundary Γ, as well as its thick approximation, does not touch ∂D (see figure 4.8) and Γ is smooth.

We define the adjoint problem

$$\left\{\begin{array}{rcll} -\mathrm{div}\left(A_{\Omega^0,\varepsilon}\, e(p)\right) &=& -j'(x,u) & \text{in } D, \\ p &=& 0 & \text{on } \Gamma_D, \\ (A_1\, e(p))\, n &=& -k'(x,u) & \text{on } \Gamma_N, \end{array}\right. \tag{4.37}$$

where the symbol $'$ denotes differentation with respect to u.

Our main result is the following.

Theorem 4.8. *The objective function (4.22) is shape differentiable in the sense of Gâteaux, namely $\theta \mapsto J((Id+\theta)\Omega^0)$ admits a Gâteaux derivative at $\theta = 0$, which is*

$$J'(\Omega^0)(\theta) = -\int_\Gamma \theta(x)\cdot n(x)\Big(f_0(x)+f_1(x)\Big)dx, \quad \forall\, \theta \in W^{1,\infty}(D,\mathbb{R}^d), \tag{4.38}$$

where n is the outer unit normal to Ω^0 and f_0, f_1 are scalar functions defined by

$$f_0(x) = \int_{\mathrm{ray}_\Gamma(x)\cap\Omega^0} h'_\varepsilon\left(d_{\Omega^0}(z)\right)(A_1-A_0)e(u)(z):e(p)(z)\prod_{i=1}^{d-1}(1+d_{\Omega^0}(z)\kappa_i(x))dz,$$

$$f_1(x) = \int_{\mathrm{ray}_\Gamma(x)\cap\Omega^1} h'_\varepsilon\left(d_{\Omega^0}(z)\right)(A_1-A_0)e(u)(z):e(p)(z)\prod_{i=1}^{d-1}(1+d_{\Omega^0}(z)\kappa_i(x))dz,$$

where z denotes a point in the ray emerging from $x \in \Gamma$.

Proof. The rigorous proof of existence of the shape derivative stems from classical arguments (typically the implicit function theorem) similar to those invoked in [234] or chapter 5 in [172]. We rather focus on the actual computation of the shape derivative and use once again the formal Lagrangian method of Céa [72]. As the computation unfolds very similarly to that in the proof of Theorem 3.6 in [14], we limit ourselves to the main arguments.

Define first the functional space $V := \{v \in H^1(D)^d \text{ such that } v = 0 \text{ on } \Gamma_D\}$, in which are sought the solution of the state equation (4.36) and of the adjoint equation (4.37). We introduce the Lagrangian $\mathcal{L} : W^{1,\infty}\left(D,\mathbb{R}^d\right)\times V\times V \to \mathbb{R}$, defined by

$$\mathcal{L}(\theta,v,q) = \int_D j(x,v)\,dx + \int_{\Gamma_N} k(x,v)\,ds + \int_D A_{(Id+\theta)\Omega^0,\varepsilon}e(v):e(q)\,dx - \int_D f\cdot q\,dx - \int_{\Gamma_N} g\cdot q\,ds. \tag{4.39}$$

Here, q is intended as the Lagrange multiplier associated to the enforcement of the state equation. As usual, stationarity of the Lagrangian provides the optimality conditions for the minimization problem. At $\theta = 0$, cancelling the partial derivative of $\mathcal{L}$ with respect to q yields the variational formulation of the state u. In the same way, the nullity of the partial derivative of $\mathcal{L}$ with respect to v leads to the variational formulation of the adjoint p.

Eventually, the shape derivative of the objective function is the partial derivative of $\mathcal{L}$ with respect to θ, evaluated at u and p

$$J'(\Omega^0)(\theta) = \frac{\partial\mathcal{L}}{\partial\theta}(0,u,p)(\theta).$$

Some elementary algebra yields, using proposition 4.5:

$$\begin{array}{rcl} J'(\Omega^0)(\theta) &=& \displaystyle\int_D \left(A_{(Id+\theta)\Omega^0,\varepsilon}\right)'(\theta)\, e(u):e(p)\,dx \\ &=& \displaystyle -\int_D h'_\varepsilon(d_{\Omega^0}(x))\left(\theta(p_\Gamma(x))\cdot n(p_\Gamma(x))\right)(A_1-A_0)e(u):e(p)\,dx, \end{array} \tag{4.40}$$

where $\left(A_{(Id+\theta)\Omega^0,\varepsilon}\right)'(\theta)$ is the directional shape derivative of $A_{(Id+\theta)\Omega^0,\varepsilon}$ while h'_ε is the standard derivative of the real function h_ε. It remains to transform this expression by the coarea formula in order to deduce a boundary integral. Using formula (4.11) for (4.40), we get:

$$J'(\Omega^0)(\theta) = -\int_\Gamma \theta(x)\cdot n(x)\left(\int_{\mathrm{ray}_\Gamma(x)\cap D} h'_\varepsilon(d_{\Omega^0}(z))(A_1-A_0)e(u)(z):e(p)(z)\prod_{i=1}^{d-1}(1+d_{\Omega^0}(z)\kappa_i(x))dz\right)dx.$$

Now decomposing the above integral over Ω^0 and Ω^1 readily yields the desired result. □

Remark 4.14. Theorem 4.8 provides a simple way of choosing a descent direction for a shape gradient based algorithm. Indeed it is enough to perturb the interface Γ by choosing the vector field

$$\theta(x) = \Big(f_0(x) + f_1(x)\Big)n(x),$$

which ensures that the directional derivative (4.38) is negative and thus yields a decrease of the objective function (4.23). This is in sharp contrast with Proposition 4.5 which provided formula (4.8) for the shape derivative. However it was impossible to extract directly from (4.8) an explicit value of θ which was a guaranteed descent direction.

Remark 4.15. In the case of compliance minimization, namely for the objective function (4.23), we have $j' = f$, $k' = g$ and thus $p = -u$. If we assume that material 1 is stronger than material 0, in the sense that $A^1 \geq A^0$ as positive definite tensors, we deduce from the formulas of Theorem 4.8 that both f_0 and f_1 are non-positive because $1 + \kappa_i(x)d_{\Omega^0}(z) \geq 0$ by virtue of Lemma 4.2. Thus, a descent direction is obtained by choosing θ such that $\theta(x)\cdot n(x) < 0$ on Γ, namely we expand Ω^1. This is in accordance with the mechanical intuition that a more robust mixture of the two materials is achieved when A^1 prevails over A^0. Of course, for the problem to be reasonable, a volume constraint is imposed on the phases.

4.4.3 Approximate formulas for the shape derivative

Although formula (4.38) is satisfying from a mathematical point of view, its numerical evaluation is not completely straightforward. There are two delicate issues. First, one has to compute the principal curvatures $\kappa_i(x)$ for any point $x \in \Gamma$ on the interface. Second, one has to perform a 1-d integration along the rays of the energy-like quantity $[A]e(u):e(p)$. This is a classical task in the level-set framework [274] but, still, it is of interest to devise a simpler approximate formula for the shape derivative.

A first approximate formula is to assume that the interface is roughly plane, namely to assume that the principal curvatures κ_i vanish. In such a case we obtain a 'Jacobian-free' approximate shape derivative

$$\begin{aligned} J'(\Omega^0)(\theta) &= -\int_\Gamma \theta(x)\cdot n(x)\Big(f_0(x)+f_1(x)\Big)dx \\ f_i(x) &= \int_{\mathrm{ray}_\Gamma(x)\cap\Omega^i} h'_\varepsilon\left(d_{\Omega^0}(z)\right)(A_1-A_0)e(u)(z):e(p)(z)dz. \end{aligned} \tag{4.41}$$

A second approximate formula is obtained when the smoothing parameter ε is small. Note that, since the support of the function h'_ε is of size 2ε, the integral in formula (4.38) is confined to a tubular neighborhood of Γ of width 2ε. Therefore, if ε is small, one may assume that the functions depending on z are constant along each ray, equal to their value at $x \in \Gamma$. In other words, for small ε we assume

$$e(u)(z) \approx e(u)(x), \quad e(p)(z) \approx e(p)(x) \ \text{ and } d_{\Omega^0}(z) \approx d_{\Omega^0}(x) = 0,$$

which yields the approximate formulas, for $x \in \Gamma$,

$$f_0(x) \approx (A_1-A_0)e(u)(x):e(p)(x)\int_{ray_\Gamma(x)\cap\Omega^0} h'_\varepsilon\left(d_{\Omega^0}(z)\right)dz,$$

$$f_1(x) \approx (A_1 - A_0)e(u)(x) : e(p)(x) \int_{ray_\Gamma(x)\cap\Omega^1} h'_\varepsilon\left(d_{\Omega^0}(z)\right)\, dz.$$

Furthermore, most rays have a length larger than 2ε so that

$$\int_{ray_\Gamma(x)\cap\Omega^0} h'_\varepsilon\left(d_{\Omega^0}(z)\right)\, dz + \int_{ray_\Gamma(x)\cap\Omega^1} h'_\varepsilon\left(d_{\Omega^0}(z)\right)\, dz = h_\varepsilon(\varepsilon) - h_\varepsilon(-\varepsilon) = 1.$$

In turn we obtain the following approximate formula for (4.38)

$$J'(\Omega^0)(\theta) \approx -\int_\Gamma (A_1 - A_0)e(u) : e(p)\, \theta \cdot n\, dx, \tag{4.42}$$

which is nothing but the discrete shape derivative (4.33) that we obtained in the sharp-interface case. This computation seems a bit miraculous but makes sense as a kind of commutation property between interface regularization and optimization.

Our numerical results show that the latter simplification (4.42), which we shall refer to as the *approximate shape derivative*, works very well in practice for problems of compliance minimization. Formula (4.42) is also used by other authors in their numerical simulations [321].

4.4.4 Convergence of the smoothed-interface shape optimization problem to the sharp-interface problem

When the smoothed-interface setting is used as an approximation of the sharp-interface case, it is a natural task to prove that this approximation is mathematically consistent. In this section, we present a result in this direction, and outline the main ideas of the proof, referring to the appendix (section 4.8) for all the technical details.

More specifically, for a given regular interface Γ, we prove that the shape gradient obtained in Theorem 4.8 for a smoothed transition layer of width 2ε converges, as ε goes to 0, to the corresponding shape gradient in the sharp-interface context, recalled in Proposition 4.10.

To set ideas, let us limit ourselves to the case of compliance minimization, the case of a general objective function such as (4.22) being no different in principle. In order to make explicit the dependence on the half-thickness ε of the smoothed transmission area, the solution of the state system (4.36) is denoted u_ε in this section. Similarly the stress tensor is $\sigma(u_\varepsilon) = A_{\Omega^0,\varepsilon}\, e(u_\varepsilon)$ and the compliance is

$$J_\varepsilon(\Omega^0) = \int_D \sigma(u_\varepsilon) : e(u_\varepsilon)\, dx.$$

The solution of the state system (4.21) in the sharp-interface case is still denoted as u, and the associated compliance as $J(\Omega^0)$.

To find the limit of $J'_\varepsilon(\Omega_0)$, as $\varepsilon \to 0$, requires some knowledge of the asymptotic behavior of $e(u_\varepsilon)$ and $\sigma(u_\varepsilon)$ in the vicinity of the interface Γ. Unfortunately, one cannot expect all the components of $e(u_\varepsilon)$ and $\sigma(u_\varepsilon)$ to converge toward their counterpart in $e(u)$ and $\sigma(u)$ in any space of smooth enough functions. Indeed, for fixed ε, $e(u_\varepsilon)$ is smooth over D (because so is the associated Hooke's tensor), whereas we recalled in Remark 4.10 that $e(u)_{\tau n}$ and $e(u)_{nn}$ are discontinuous across Γ, as imposed by the transmission conditions. However, some of the components of $e(u_\varepsilon)$ and $\sigma(u_\varepsilon)$ do behave well as $\varepsilon \to 0$. This is the purpose of the following proposition, which is a consequence of rather classical results in elliptic regularity theory.

Proposition 4.12. *Assuming Γ is a $\mathcal{C}^2$ interface, there exists a tubular neighborhood $V \subset\subset D$ of Γ such that one can define a smooth extension in V of the normal n and of a set of tangentials and orthonormal vectors τ. Then, the following strong convergences hold true*

$$\begin{array}{lcll} e(u_\varepsilon)_{\tau\tau} & \overset{\varepsilon\to 0}{\longrightarrow} & e(u)_{\tau\tau} & \text{in } H^1(V)^{(d-1)^2} \text{ strong,} \\ \sigma(u_\varepsilon)_{\tau n} & \overset{\varepsilon\to 0}{\longrightarrow} & \sigma(u)_{\tau n} & \text{in } H^1(V)^d \text{ strong,} \\ \sigma(u_\varepsilon)_{nn} & \overset{\varepsilon\to 0}{\longrightarrow} & \sigma(u)_{nn} & \text{in } H^1(V) \text{ strong.} \end{array} .$$

Remark 4.16. The components of the strain and stress tensors which converge in Proposition 4.15 are exactly those which are continuous through the interface Γ as explained in Remark 4.10.

We are now in a position to state the main result of the present section which implies that the shape derivative of the smoothed-interface objective function is a consistent approximation of the corresponding shape derivative in the sharp-interface case.

Theorem 4.9. *Under the above assumptions, we have*

$$\lim_{\varepsilon\to 0} J'_\varepsilon(\Omega^0)(\theta) = J'(\Omega^0)(\theta) \quad \forall\, \theta \in W^{1,\infty}(D,\mathbb{R}^d).$$

Sketch of the proof. We limit ourselves with an outline of the main steps, referring to section 4.8 for details. The goal is to pass to the limit $\varepsilon \to 0$ in formula (4.38), for a fixed $\theta \in W^{1,\infty}(D,\mathbb{R}^d)$. To achieve this, the rays $\mathrm{ray}_\Gamma(x)\cap\Omega^0$ and $\mathrm{ray}_\Gamma(x)\cap\Omega^1$ are expressed as integrals over the segment $(0,1)$. Therefore, (4.38) becomes

$$J'_\varepsilon(\Omega^0)(\theta) = -\int_\Gamma \theta(x)\cdot n(x)\Big(f_0^\varepsilon(x)+f_1^\varepsilon(x)\Big)dx,$$

where $f_0^\varepsilon, f_1^\varepsilon \in L^1(\Gamma)$ are defined as

$$f_0^\varepsilon(x) = \int_{-1}^{0} h'_\varepsilon(s\varepsilon)(A_1-A_0)e(u_\varepsilon)(x+s\varepsilon n(x)) : e(u_\varepsilon)(x+s\varepsilon n(x))\, k_\varepsilon(x,s)\, ds, \tag{4.43}$$

$$f_1^\varepsilon(x) = \int_{0}^{1} h'_\varepsilon(s\varepsilon)(A_1-A_0)e(u_\varepsilon)(x+s\varepsilon n(x)) : e(u_\varepsilon)(x+s\varepsilon n(x))\, k_\varepsilon(x,s)\, ds, \tag{4.44}$$

with

$$k_\varepsilon(x,s) = \prod_{i=1}^{d-1}\left(1+s\varepsilon\kappa_i(x)\right).$$

Since $h'_\varepsilon(s\varepsilon)$ does not depend on ε, to pass to the limit in (4.43) and (4.82) requires merely the following simple technical convergence result:

$$\int_0^1 v(s)\, f_\varepsilon(x+s\varepsilon n(x)) g_\varepsilon(x+s\varepsilon n(x))\, ds \xrightarrow{\varepsilon\to 0} \left(\int_0^1 v(s)\, ds\right) f(x)g(x) \quad \text{ in } L^1(\Gamma) \tag{4.45}$$

for a smooth function $v(s)$ and any sequences $f_\varepsilon, g_\varepsilon \in H^1(D)$, which converge strongly in $H^1(D)$ to f, g respectively. In order to apply (4.45) we rewrite expressions (4.43) and (4.82) in terms of the components $e(u_\varepsilon)_{\tau\tau}$ and $\sigma(u_\varepsilon)_{\tau n}$, $\sigma(u_\varepsilon)_{nn}$ of the strain and stress tensors, which have a fine behavior at the limit $\varepsilon\to 0$ as guaranteed by Proposition 4.15. After some algebra, we obtain the following rearrangement for the integrand in f_0^ε and f_1^ε:

$$\begin{aligned}
(A_1-A_0)e(u_\varepsilon) : e(u_\varepsilon)(x+s\varepsilon n(x)) &= \mu'(s)\left(e(u_\varepsilon)_{\tau\tau} : e(u_\varepsilon)_{\tau\tau}\right)(x+s\varepsilon n(x)) \\
&+ \frac{\mu'(s)}{\mu(s)^2}\left(\sigma^\varepsilon(u_\varepsilon)_{\tau n}\cdot\sigma^\varepsilon(u_\varepsilon)_{\tau n}\right)(x+s\varepsilon n(x)) \\
&+ \frac{4\mu^2(s)\lambda'(s)+2\mu'(s)\lambda^2(s)}{(2\mu(s)+\lambda(s))^2}\,\mathrm{tr}(e(u_\varepsilon)_{\tau\tau})^2(x+s\varepsilon n(x)) \\
&+ \frac{2\mu'(s)+\lambda'(s)}{(2\mu(s)+\lambda(s))^2}\,\sigma^\varepsilon(u_\varepsilon)_{nn}^2(x+s\varepsilon n(x)) \\
&+ \frac{4\mu(s)\lambda'(s)-4\mu'(s)\lambda(s)}{(2\mu(s)+\lambda(s))^2}\left(\sigma^\varepsilon(u_\varepsilon)_{nn}\,\mathrm{tr}(e(u_\varepsilon)_{\tau\tau})\right)(x+s\varepsilon n(x))
\end{aligned},$$

with

$$\lambda(s) = \lambda_0 + h_\varepsilon(s\varepsilon)(\lambda_1-\lambda_0), \quad \mu(s) = \mu_0 + h_\varepsilon(s\varepsilon)(\mu_1-\mu_0),$$

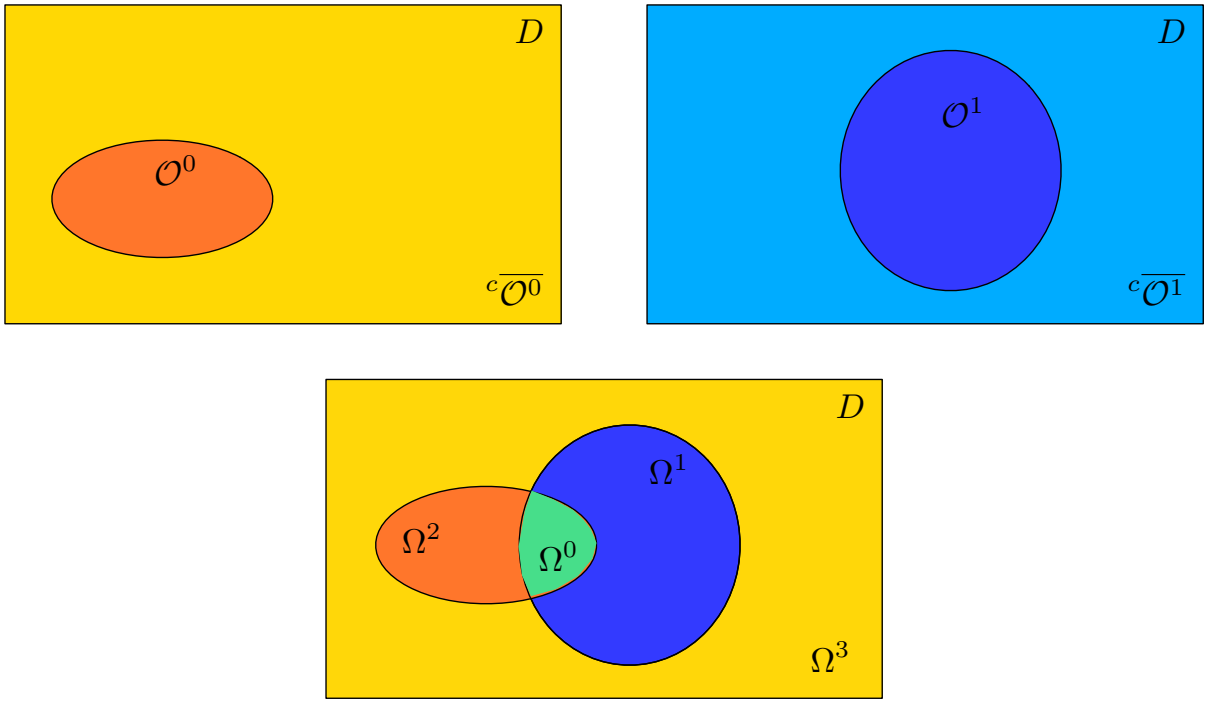


Figure 4.10: Two subdomains of D *(top)* and the four phase domains derived by combining them together *(down)*.

where λ_0, μ_0 and λ_1, μ_1 are the Lamé coefficients of materials 0, 1 respectively. Note that all the functions of s involving $\lambda(s)$ and $\mu(s)$ appearing in the above expression arise as exact derivatives of functions of $\lambda(s)$ and $\mu(s)$. Passing to the limit in the above expression using (4.45) leads to

$$(f_0^\varepsilon + f_1^\varepsilon) \to \mathcal{D}(u,u) \quad \text{ in } L^1(\Gamma),$$

where $\mathcal{D}(u,u)$ is defined as

$$\begin{aligned} \mathcal{D}(u,u)(x) \;=\; & 2\,[\mu]\, e(u)_{\tau\tau}(x) : e(u)_{\tau\tau}(x) - \left[\tfrac{1}{\mu}\right] \sigma(u)_{\tau n}(x) \cdot \sigma(u)_{\tau n}(x) \\ & + \left[\tfrac{2\lambda\mu}{(2\mu+\lambda)}\right] \operatorname{tr}(e(u)_{\tau\tau}(x))^2 - \left[\tfrac{1}{2\mu+\lambda}\right] \sigma(u)(x)^2_{nn} \\ & + \left[\tfrac{2\lambda}{2\mu+\lambda}\right] \sigma(u)_{nn}(x)\, \operatorname{tr}(e(u)_{\tau\tau}(x)) \end{aligned} \quad ,$$

which after some algebra rewrites as (4.25). This completes the proof. □

4.5 Extension to more than 2 materials

The methods presented in sections 4.3 and 4.4 for two phases can be extended to the case of several materials to be optimally placed in the domain D, following a classical idea in the level-set framework [315], [321].

Hitherto, we considered a single subdomain $\Omega^0 \subset D$, which allows to account for two separate phases within D, occupying respectively the domains Ω^0 and $\Omega^1 := {}^c\overline{\Omega^0}$ (where ^{c} denotes the complementary part in D). To consider more phases, we introduce m subdomains $\mathcal{O}^0, ..., \mathcal{O}^{m-1} \subset D$ which are not subject to any

geometrical constraints (they can intesect, or not, and they don't need to cover D). These m subdomains allow us to treat up to 2^m distinct phases, filling respectively the phase domains $\Omega^0, ..., \Omega^{2^m-1} \subset D$, defined as (see Figure 4.5)

$$\left\{\begin{array}{lcl} \Omega^0 & = & \mathcal{O}^0 \cap \mathcal{O}^1 \cap ... \cap \mathcal{O}^{m-1}, \\ \Omega^1 & = & {}^c\overline{\mathcal{O}^0} \cap \mathcal{O}^1 \cap ... \cap \mathcal{O}^{m-1}, \\ & \vdots & \\ \Omega^{2^m-1} & = & {}^c\overline{\mathcal{O}^0} \cap {}^c\overline{\mathcal{O}^1} \cap ... \cap {}^c\overline{\mathcal{O}^{m-1}}. \end{array}\right. \tag{4.46}$$

Note that $\Omega^0, ..., \Omega^{2^m-1}$ is a partition of D. To simplify the exposition, from now on we take $m = 2$, meaning that we consider four different materials, with respective Hooke's law A_0, A_1, A_2, A_3. Two subdomains $\mathcal{O}^0, \mathcal{O}^1$ of D are then introduced, and each material A_i fills an area $\Omega^i \subset D$, defined through formula (4.46).

For the sharp-interface problem, the definition of the mixture Hooke's tensor A_χ is standard. Introducing χ_0 and χ_1 the characteristic functions of $\mathcal{O}^0$ and $\mathcal{O}^1$, respectively, we define

$$A_\chi(x) := \chi_0(x)\chi_1(x)A_0 + (1-\chi_0(x))\,\chi_1(x)A_1 + \chi_0(x)\,(1-\chi_1(x))\,A_2 + (1-\chi_0(x))\,(1-\chi_1(x))\,A_3. \tag{4.47}$$

For the smoothed-interface problem, we propose a formula inspired from (4.47)

$$\begin{array}{rcl} A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}(x) & = & (1-h_\varepsilon(d_{\mathcal{O}^0}(x)))(1-h_\varepsilon(d_{\mathcal{O}^1}(x)))A_0 + h_\varepsilon(d_{\mathcal{O}^0}(x))(1-h_\varepsilon(d_{\mathcal{O}^1}(x)))A_1 \\ & + & (1-h_\varepsilon(d_{\mathcal{O}^0}(x)))h_\varepsilon(d_{\mathcal{O}^1}(x))A_2 + h_\varepsilon(d_{\mathcal{O}^0}(x))h_\varepsilon(d_{\mathcal{O}^1}(x))A_3, \end{array} \tag{4.48}$$

where h_ε is the smooth approximation (4.35) of the Heaviside function and $d_{\mathcal{O}^0}, d_{\mathcal{O}^1}$ are the signed distance functions to $\mathcal{O}^0$ and $\mathcal{O}^1$ respectively. Of course, there are other interpolation formulas and any alternative choice which, as (4.48), satisfies the following consistency

$$A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}(x) = \left\{\begin{array}{l} A_0 \quad \text{if} \quad d_{\mathcal{O}^0}(x) < -\varepsilon \quad \text{and} \quad d_{\mathcal{O}^1}(x) < -\varepsilon, \\ A_1 \quad \text{if} \quad d_{\mathcal{O}^0}(x) > +\varepsilon \quad \text{and} \quad d_{\mathcal{O}^1}(x) < -\varepsilon, \\ A_2 \quad \text{if} \quad d_{\mathcal{O}^0}(x) < -\varepsilon \quad \text{and} \quad d_{\mathcal{O}^1}(x) > +\varepsilon, \\ A_3 \quad \text{if} \quad d_{\mathcal{O}^0}(x) > +\varepsilon \quad \text{and} \quad d_{\mathcal{O}^1}(x) > +\varepsilon, \\ \text{a smooth interpolation between } A_0, A_1, A_2, A_3 \text{ otherwise,} \end{array}\right. \tag{4.49}$$

will do. In particular, for applications in material science where the thick interface has a clear physical interpretation, one could choose a physically relevant choice of the interpolant Hooke's law for the mixture of A_0, A_1, A_2, A_3 in the intermediate areas, like a sequential laminate or another microstructure achieving Hashin and Shtrikman bounds [229]. On the other hand, if the smoothed-interface problem is merely a mathematical approximation of the sharp-interface case, then it is a consistent approximation since, as the regularizing parameter ε goes to 0, the smooth tensor $A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}$ converges to the discontinuous one A_χ.

In the multiphase case, the definition of the objective function (4.22) does not change

$$J(\mathcal{O}^0, \mathcal{O}^1) = \int_D j(x,u)\,dx + \int_{\Gamma_N} k(x,u)\,ds, \tag{4.50}$$

and the state or adjoint equations are the same, up to changing the previous Hooke's tensor by $A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}$. There are now two variable subdomains, $\mathcal{O}^0, \mathcal{O}^1$, as design variables for the optimization problem. Accordingly, we introduce two separate vector fields $\theta_0, \theta_1 \in W^{1,\infty}\left(D, \mathbb{R}^d\right)$ in order to vary the subdomains $\mathcal{O}^0, \mathcal{O}^1$.

According to Theorem 4.8, the partial shape derivative of the objective function (4.50) with respect to $\mathcal{O}^0$ and $\mathcal{O}^1$, which we shall denote as $\frac{\partial J}{\partial \mathcal{O}^0}$ and $\frac{\partial J}{\partial \mathcal{O}^1}$ respectively, in the direction of θ^0 and θ^1, respectively, are

$$\frac{\partial J}{\partial \mathcal{O}^0}(\mathcal{O}^0, \mathcal{O}^1)(\theta_0) = \int_D \theta_0(p_{\partial\mathcal{O}^0}(x)) \cdot n_0(p_{\partial\mathcal{O}^0}(x)) \frac{\partial A}{\partial d_{\mathcal{O}^0}}(d_{\mathcal{O}^0}, d_{\mathcal{O}^1})e(u) : e(p)\,dx, \tag{4.51}$$

$$\frac{\partial J}{\partial \mathcal{O}^1}(\mathcal{O}^0, \mathcal{O}^1)(\theta_1) = \int_D \theta_1(p_{\partial\mathcal{O}^1}(x)) \cdot n_1(p_{\partial\mathcal{O}^1}(x)) \frac{\partial A}{\partial d_{\mathcal{O}^1}}(d_{\mathcal{O}^0}, d_{\mathcal{O}^1})e(u) : e(p)\,dx, \tag{4.52}$$

where $A(d_{\mathcal{O}^0}, d_{\mathcal{O}^1}) = A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}$, defined in (4.49). Of course, one can apply Theorem 4.8 to simplify (4.51) and (4.52) and transform them in surface integrals on $\partial\mathcal{O}^0$ and $\partial\mathcal{O}^1$.

Remark 4.17. In the sharp interface context one could compute shape derivatives of the objective function J with respect to $\mathcal{O}^0$ and $\mathcal{O}^1$ too, thus recovering formulas similar to (4.51) and (4.52). However, it is possible only if we assume that the boundary of $\mathcal{O}^0$ and $\mathcal{O}^1$ do not superpose. Indeed if, for example, $\partial\mathcal{O}^0 = \partial\mathcal{O}^1$, then moving $\mathcal{O}^0$ inside $\mathcal{O}^1$, or vice versa, implies that one phase or another one appears. This means that a topology change is occuring which cannot be handled by Hadamard's method. At most, one can expect to compute two different directional derivatives (inward and outward) which clearly shows that there is no differentiability in this case. Note that there is no such difficulty in the smoothed interface setting: formulas (4.51) and (4.52) hold true for any geometrical situation of $\mathcal{O}^0$ and $\mathcal{O}^1$ since $A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}$ is a smooth function of x in D.

4.6 Discussion and comparison with previous formulae in the literature

To our knowledge, the first works on multi-phase optimization using a level-set method are [223] and [321]. Further references include [224], [320], [322]. In all these works the computation of the shape derivative is not mathematically rigorous and the obtained formulas are not strictly correct. Indeed, either the shape differentiation is performed in the sharp-interface case and then the non-differentiable character of the solution of (4.21) is ignored (as explained in section 4.3.2), or the shape derivative is evaluated in the smoothed-interface case and then the derivative of the signed distance function is not taken into account. Fortunately, the shape derivative formulas in [223] and [321] coincide with what we called our approximate shape derivatives obtained in Proposition 4.11 for a discretization of the sharp-interface case and in (4.42) for a very thin smoothed interface. A third possibility for interpreting these works is to consider that the regularization of the interface is made with the help of the level set function ψ (used in numerical practice for representing and advecting the shape, see section 4.7 below) rather than with the signed distance function d_Ω. Then the differentiation is performed with respect to ψ rather than with respect to the shape Ω. It alleviates all the technical details of section 4.4 but it has one major flaw that we now describe.

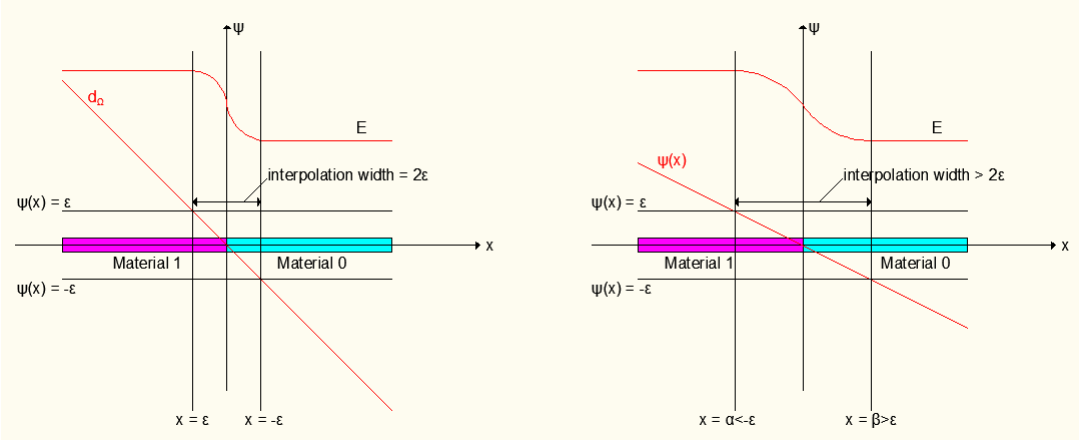


Figure 4.11: Intermediate zone for regularization with the signed distance function *(left)* or with an arbitrary level set function *(right)*.

Indeed, in the context of section 4.4 on the smoothed interface approach, one could be tempted to replace

the regularization formula (4.34) by a similar one

$$A_{\Omega^0,\varepsilon}(x) = A_0 + h_\varepsilon(\psi(x))(A_1 - A_0), \quad \forall\, x \in D, \tag{4.53}$$

where the signed distance function d_Ω has simply been replaced by the level set function ψ. Then, as is done in [223] and [224], one may differentiate the objective function with respect to ψ. A serious problem that rises directly from this choice, is that the interpolation zone, where $A_{\Omega^0,\varepsilon}$ takes intermediate values between A_0 and A_1, can thicken during the optimization process, especially if the level set function ψ is not frequently reinitialized towards the signed distance function to the boundary (see Figure 4.11). The reason is that the interpolation zone corresponds to some kind of homogenized material made of A_0 and A_1, which is known to be more advantageous than pure phases in most problems [9]. The optimization process therefore does not only move the interface location but also flatten the level set function ψ so that the interpolation zone gets thicker. Even when the level set function is reinitialized, there remains a difficulty in the sense that the value of the objective function may change before and after reinitialization. A partial remedy to this inconvenient, as suggested in [223], is to add to the objective function a penalization term to control the enlargement.

The computation of the shape derivative is slightly different in [321]: the authors carry out the derivation with the level set function ψ but in the resulting formula they assume that ψ coincides with the signed distance function to the interface d_Ω. More precisely, following the notations of Proposition 4.5, they consider a functional

$$J(\Omega) = \int_D m(x, \psi(x))\, dx, \tag{4.54}$$

where ψ is a solution of the Hamilton-Jacobi equation

$$\frac{\partial \psi}{\partial t} + \theta \cdot n\, |\nabla \psi| = 0.$$

Then, the authors claim that the shape derivative is

$$J'(\Omega)(\theta) = -\int_D \frac{\partial m}{\partial \psi}(x, \psi(x))\, \theta(x) \cdot n(x)\, dx. \tag{4.55}$$

Note the difference with our formula (4.8), which involves the projection $p_\Gamma(x)$ of x on the boundary $\Gamma = \partial\Omega$, and that we recall as

$$J'(\Omega)(\theta) = -\int_D \frac{\partial m}{\partial \psi}(x, d_\Omega(x))\, \theta(p_\Gamma(x)) \cdot n(p_\Gamma(x))\, dx.$$

Unfortunately, there is no a priori guarantee that the transported signed distance function to the boundary $\partial\Omega$ remains the signed distance function to the transported boundary $(Id + \theta)\partial\Omega$. Therefore, the shape derivative $d'_\Omega(\theta)(x)$ cannot be replaced by the expression $\frac{\partial \psi}{\partial t} = -\theta \cdot n\, |\nabla \psi|$ coming from the Hamilton-Jacobi equation, as it is done in [223] and [321], without making any further assumptions. For example, in [160] it is shown that the transported level set function remains the signed distance function (at least for a small time) if the advection velocity remains constant along the normal, namely $(\theta \cdot n)(x) = (\theta \cdot n)(p_\Gamma(x))$.

A difficulty with (4.55) is that it does not satisfy the Hadamard structure theorem (see e.g. [9], [105], [172], [234] and references therein) since it does not depend solely on the normal trace $\theta \cdot n$ on the interface $\Gamma = \partial\Omega$. In fact, assuming that the support of $\frac{\partial m}{\partial \psi}$ is concentrated around Γ, formula (4.55) would be similar to what we called earlier 'approximate shape derivative', obtained in Proposition 4.11 for a discretization of the sharp-interface case and in (4.42) for the smoothed-interface case when the regularization parameter ε is small. In any case, (4.55) does not guarantee a descent direction in general, unless $\frac{\partial m}{\partial \psi}$ keeps a constant sign along the normal, at least for the width of the intermediate zone.

4.7 Numerical results

4.7.1 Level-set representation

Following the lead of [14], [15], we represent the moving and optimizable interfaces by level set functions [245] defined on a fixed mesh in an Eulerian framework. According to Section 4.5, using m level set functions we can represent up to 2^m separate phases.

When there are only two phases to optimize, it suffices to use one level set function to represent the interface Γ between two complementary sub-domains Ω^0 and Ω^1 of the working domain D. The level set function ψ (see figure 4.26) complies with the properties:

$$\begin{cases} \psi(x) = 0 & \text{for } x \in \Gamma = \partial\Omega_0, \\ \psi(x) < 0 & \text{for } x \in \Omega^0, \\ \psi(x) > 0 & \text{for } x \in \Omega^1. \end{cases}$$

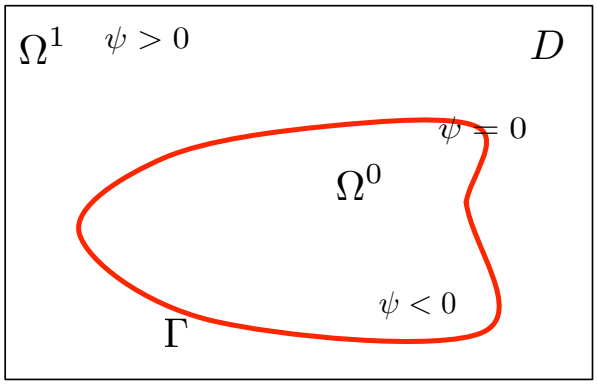


Figure 4.12: Level-set representation of the domains Ω^0 and Ω^1.

During the optimization process the shape is advected with a scalar velocity field $V(x)$ in the direction of the outer normal vector field $n(x)$ to Ω^0 ($V = \theta \cdot n$). For instance, in the smoothed-interface setting of section 4.4, the value of this scalar field is derived from the shape sensitivity analysis of Theorem 4.8. More precisely, the choice

$$V(x) = f_0(x) + f_1(x),$$

where f_0, f_1 are defined by (4.38), clearly gives a descent direction for $\theta = V\,n$. The functions f_0 and f_1 are defined for all points $x \in \Gamma$ as integrals along rays in the normal direction. Since the interface Γ is not explicitly discretized, f_0 and f_1 are evaluated at the nodes of the elements that are crossed by the zero level-set. The normal vector is computed for each of these nodes, which defines the direction of the rays and a simple quadrature formula is used for the numerical approximation of f_0 and f_1. This computation is done only in a band of thickness 2ε around the interface, where h'_ε is non-zero, and as long as the skeleton of Γ (see Definition 4.2) is not detected (recall that the rays end up at the skeleton). When integrating along a ray the skeleton is identified as the region where the signed-distance function loses its monotonicity.

The advection is described in the level set framework by introducing a pseudo time $t \in \mathbb{R}_+$ and solving the Hamilton-Jacobi equation over D

$$\frac{\partial \psi}{\partial t} + V|\nabla \psi| = 0, \tag{4.56}$$

using an explicit upwind scheme [274] (see also chapter 1, §1.2.1). However, the scalar field V is a priori defined only on the boundary of the shape and therefore it is necessary to extend it to the whole domain

in order to be able to perform multiple iterations of the transport equation (4.56) for each finite element analysis. Moreover, it is numerically advantageous to regularize the advection velocity in order to assure some smoothness required by sensitivity analysis [162] (see chapter 1, §2.2.3.3). One way to extend and regularize at the same time V is to solve the variational formulation for $Q \in H^1(D)$

$$\int_D \left(\alpha^2 \nabla Q \cdot \nabla V + Q\, V\right) dx = J'(\Omega)(V\, n) \quad \text{for any } V \in H^1(D), \tag{4.57}$$

where $\alpha > 0$ is a positive parameter that controls the regularization width (typically α is of the order of the mesh size). Then, choosing $V = -Q$, we find

$$J'(\Omega)(-Q\, n) = -\int_D \left(\alpha^2 |\nabla Q|^2 + Q^2\right) dx,$$

which guarantees again that $-Q\, n$ is a descent direction for J.

In order to describe up to four distinct phases, two level-set functions ψ_0 and ψ_1 are defined such that

$$\begin{cases} \psi_0(x) = 0 & \text{for } x \in \partial\mathcal{O}^0, \\ \psi_0(x) < 0 & \text{for } x \in \mathcal{O}^0, \\ \psi_0(x) > 0 & \text{for } x \in {}^c\overline{\mathcal{O}^0}, \end{cases} \quad \text{and} \quad \begin{cases} \psi_1(x) = 0 & \text{for } x \in \partial\mathcal{O}^1, \\ \psi_1(x) < 0 & \text{for } x \in \mathcal{O}^1, \\ \psi_1(x) > 0 & \text{for } x \in {}^c\overline{\mathcal{O}^1}, \end{cases}$$

following the notations of Figure 4.5. Then, each level set function ψ_i, $i = 0, 1$, is transported independently solving (4.56), where V_i, $i = 0, 1$ results from the formulas (4.51) and (4.52).



Figure 4.13: Boundary conditions for the long cantilever.

4.7.2 Two materials in the sharp interface context

We work in the context of Section 4.3, namely in a sharp interface framework. We compare the two shape derivatives: the continuous formula furnished by Proposition 4.10 and the discrete formula given in Proposition 4.11. The numerical implementation of the continuous formula of the shape derivative in Proposition 4.10 is achieved according to the scheme proposed in [15] for computing the jump approximation (4.32). We consider a long cantilever of dimensions 2×1, discretized by 100×50 $\mathbb{P}1$ elements, clamped at its left side and submitted to a unit vertical load at the middle of its right side (see Figure 4.13). The domain is filled by two isotropic materials 0 and 1, with different Young's moduli, respectively $E^0 = 0.5$ and $E^1 = 1$ (material 1 is stiffer than material 0) but with the same Poisson ratio $\nu = 0.3$. We minimize the compliance (4.23) with a constraint of fixed volume for the two phases. The computations are done with the FreeFem++ package [259].

For all the numerical examples in this chapter, an augmented Lagrangian method is applied to handle the constraints. Following the approach in [237], supposing that our problem contains m equality constraints

of the type $c_i(\Omega^0) = 0 \quad (i = 1, ..., m)$, an augmented Lagrangian function is constructed as

$$L(\Omega^0, \ell, \mu) = J(\Omega^0) - \sum_{i=1}^{m} \ell_i c_i(\Omega^0) + \sum_{i=1}^{m} \frac{\mu_i}{2} c_i^2(\Omega^0),$$

where $\ell = (\ell_i)_{i=1,...,m}$ and $\mu = (\mu_i)_{i=1,...,m}$ are Lagrange multipliers and penalty parameters for the constraints. The Lagrange multipliers are updated at each iteration n according to the optimality condition $\ell_i^{n+1} = \ell_i^n - \mu_i c_i(\Omega_n^0)$. The penalty parameters are augmented every 5 iterations. With such an algorithm the constraints are enforced only at convergence (see for example Figure 4.15). Of course, other (and possibly more efficient) optimization algorithms could be used instead.

The results are displayed on Figure 4.14. As usual the strong phase 1 is black and the weak phase 0 is white. The design obtained with the discrete formula is quite similar to the one exposed in Figure 4.16 (c). However the continuous formula gives a different optimal shape which is worse in terms of the objective function than the one obtained with the discrete formula (see Figure 4.15).

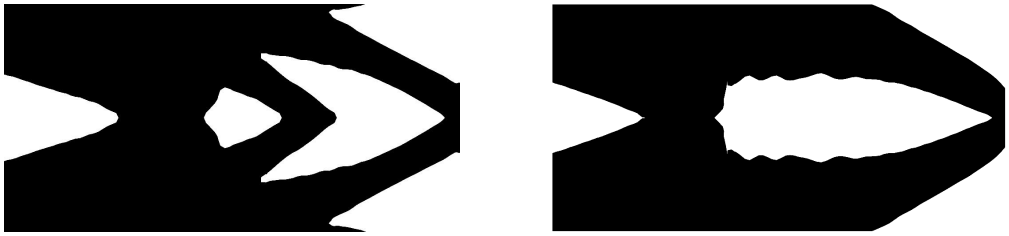


Figure 4.14: Optimal shapes for the long cantilever using the discrete shape gradient *(left)* and the continuous formula *(right)*.

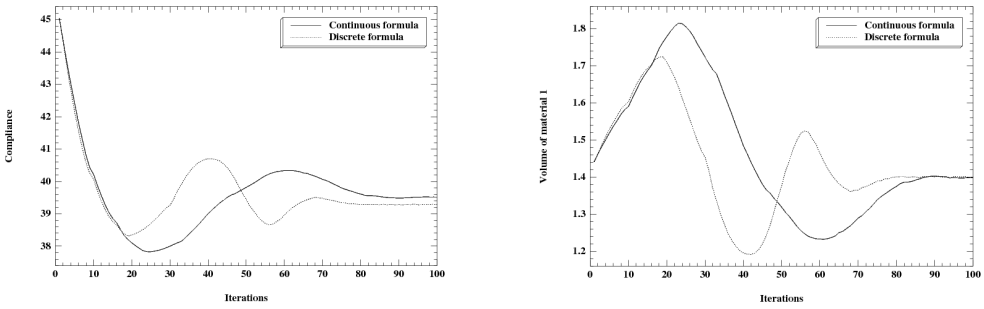


Figure 4.15: Convergence history of the compliance *(left)* and the volume *(right)* for the sharp interface results displayed on Figure 4.14.

4.7.3 Two materials in the smoothed-interface context

We now switch to the smoothed-interface setting as described in Section 4.4. We perform the same test case, with the same parameter values, as in Section 4.7.2. All computations are performed in Scilab. A first goal is to compare the smoothed-interface approach to the sharp-interface one. A second goal is to compare the various formulas for the shape derivative obtained in Section 4.4.

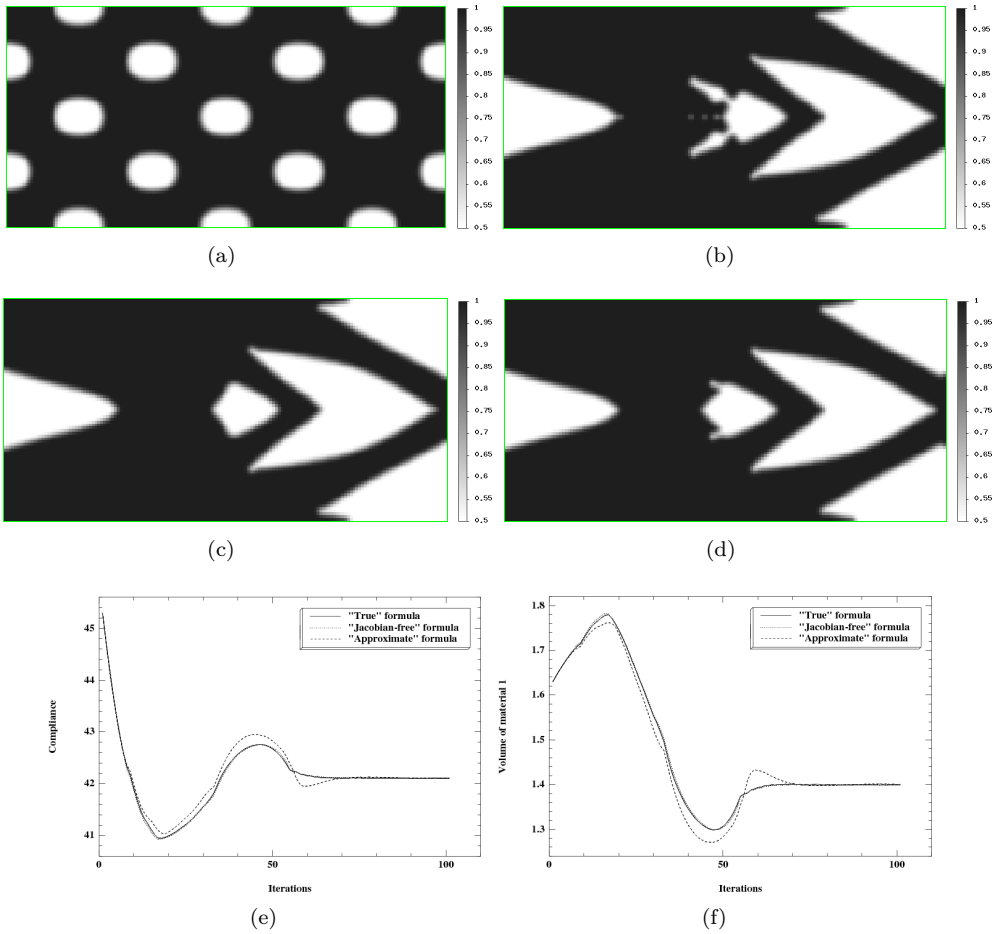


Figure 4.16: Long cantilever using two phases with $V_T = 0.7|D|$ and a small smoothing parameter $\varepsilon = 2\Delta x$; *(a)* initialization, *(b)* optimized shape using the 'true' formula, *(c)* optimized shape using the 'Jacobian-free' formula, *(d)* optimized shape using the 'approximate' formula, *(e)* convergence of the compliance, (f) convergence of the volume.

We minimize again the compliance (4.23) with a constraint of fixed volume for the two phases which is written

$$\int_D h_\varepsilon(d_{\Omega^0}(x))dx = V_T,$$

where V_T is the target volume of the strong phase occupying Ω^1.

We test three different formulas for the shape gradient. The first one is the 'true' formula given by (4.38) (see also (4.51) and (4.52) in the case of more than two phases). The second one, called 'Jacobian-free', is (4.41) which is obtained from (4.38) by neglecting the Jacobian term. The reason for this choice is that the curvature is not precisely calculated using a fixed mesh and therefore we may introduce a significant approximation error. In any case, it amounts to neglecting a positive factor, thus it remains a descent direction. The third one is the 'approximate' formula (4.42) obtained for a very thin smoothing zone around the interface.

First, we consider the case of a 'thin' interface. The interpolation width is chosen as $\varepsilon = 2\Delta x$, where Δx is the uniform mesh size. The results for $V_T = 0.7|D|$ are shown in Figure 4.16. We plot the Young modulus

distribution (black being the strong material A_1 and white the weak material A_0). The convergence histories are almost identical for the 'true' and 'Jacobian-free' formulas of the shape derivative. It is slightly more oscillating for the 'approximate' formula although it converges to almost the same value of the objective function. The resulting optimal designs are very similar.

For a larger interpolation width $\varepsilon = 8\Delta x$ ('thick' interface), the results are shown in Figure 4.17. We clearly see a difference for the optimal shape obtained using the "true" formula of the shape derivative: in this case, the algorithm produces a very long and oscillating interface in such a way that the overall structure is almost like a composite structure. This is due to the fact that the intermediate zone inside the interface is very favorable compared to the pure phases. Nevertheless, despite the differences in the final shapes, the values of the compliance are almost the same for the 'true' and 'Jacobian-free' formulas, slightly worse for the "approximate" formula of the shape gradient.

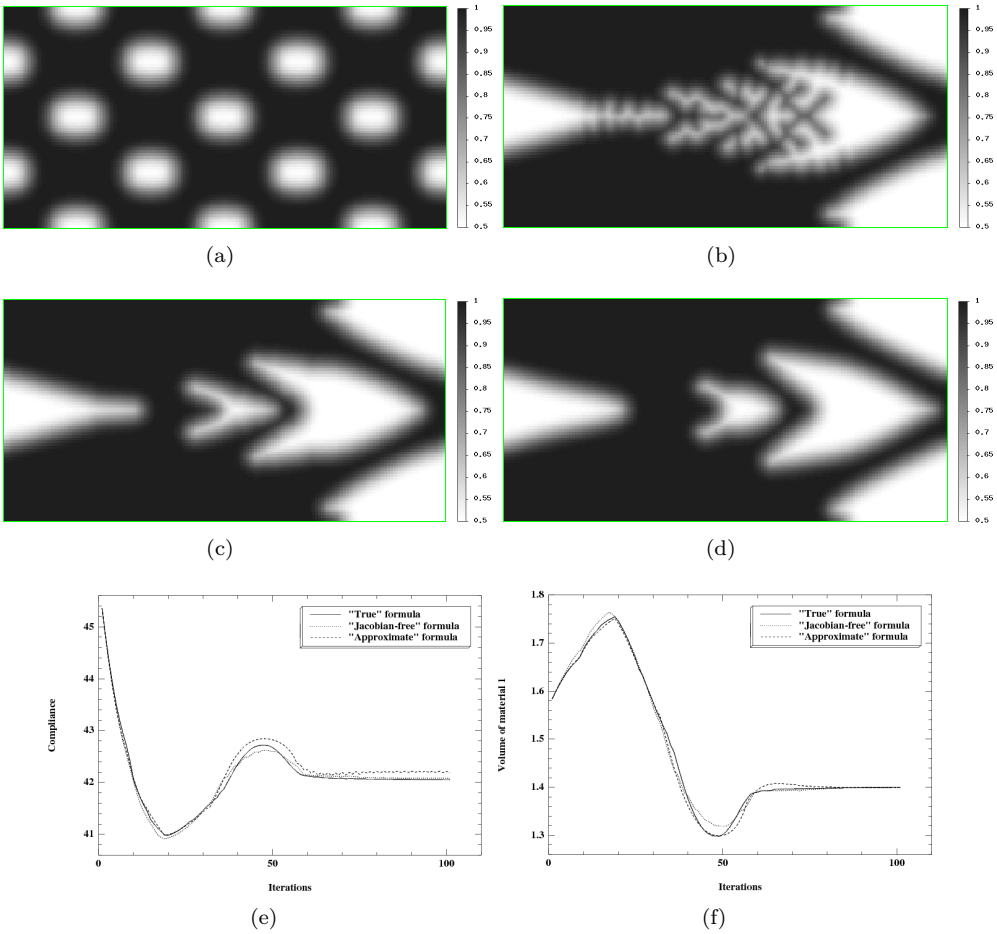


Figure 4.17: Long cantilever using two phases with $V_T = 0.7|D|$ and a large smoothing parameter $\varepsilon = 8\Delta x$; *(a)* initialization, *(b)* optimized shape using the 'true' formula, *(c)* optimized shape using the 'Jacobian-free' formula, *(d)* optimized shape using the 'approximate' formula, *(e)* convergence of the compliance, (f) convergence of the volume.

4.7.4 Four phases in the smoothed interface context

We consider now the case of using up to four phases and consequently two level set functions. A smoothed approximation of the characteristic function of each phase can be constructed using combinations of the functions h_ε, defined in equation (4.35), as follows

$$\begin{cases} \chi_0 = (1 - h_\varepsilon(d_{\mathcal{O}^0}))(1 - h_\varepsilon(d_{\mathcal{O}^1})), \\ \chi_1 = h_\varepsilon(d_{\mathcal{O}^0})(1 - h_\varepsilon(d_{\mathcal{O}^1})), \\ \chi_2 = (1 - h_\varepsilon(d_{\mathcal{O}^0}))h_\varepsilon(d_{\mathcal{O}^1}), \\ \chi_3 = h_\varepsilon(d_{\mathcal{O}^0})h_\varepsilon(d_{\mathcal{O}^1}), \end{cases} \tag{4.58}$$

and the global Hooke's tensor in given by (4.48). The optimization problem now reads

$$\begin{aligned} \min_{\mathcal{O}^0,\mathcal{O}^1 \in \mathcal{U}_{ad}} J(\mathcal{O}^0, \mathcal{O}^1) &= \int_D A_{\mathcal{O}^0,\mathcal{O}^1,\varepsilon}(x) e(u) : e(u)\, dx \\ \text{s.t.} \quad \int_D \chi_i\, dx &= V_T^i,\ i = 0, ..., 3\,, \end{aligned} \tag{4.59}$$

where V_T^i is the target volume for the phase i (they sum up to the volume of D). As previously, an augmented Lagrangian algorithm is applied to enforce the constraints. In this section we work with a 'thin' interface, namely $\varepsilon = 2\Delta x$. For all test cases, we checked numerically that the three formulas of the shape gradient give very similar optimal shapes, as expected. The results presented in the sequel have been obtained using the 'Jacobian-free' formula.

We test our method with several benchmark examples presented in [321] and [322]. Since the initial design are different, as well as the numerical methods, it is hard to make a quantitative comparison and we satisfy ourselves with a qualitative comparison.

4.7.4.1 Short cantilever using two materials and void

In this paragraph we consider only three phases, made of two materials and void. The first structure to be optimized is a two-dimensional short cantilever, of dimensions 1×2, discretized using 80×160 $\mathbb{Q}^1$ elements. The left part of the structure is clamped and a unitary vertical force is applied at the mid point of its right part (see Figure 4.18). The Young moduli of the four phases are defined as $E^0 = 0.5, E^1 = 10^{-3}, E^2 = 1$ and $E^3 = 10^{-3}$, where both phases 1 and 3 represent void. The target volumes for phases 0 and 2 are set to $V_T^0 = 0.2|D|$ and $V_T^2 = 0.1|D|$. Remark that phases 1 and 3 are the same, corresponding to void. The fact that the void zone is represented by two different characteristic functions has no influence on the numerical results (at least in all our numerical experiments). The initial and the optimal shape (obtained after 200 iterations) are shown in Figure 4.19 (a) and (b). We plot the Young modulus with a grey scale: dark stands for the stronger phase, white for void and grey for the intermediate phase.

This test case was previously studied in [321] (see figures 7 and 8 therein for two different initializations). Our results are roughly similar to those in [321] and even slightly better since the design of Figure 4.19 (b) is symmetric (as expected), contrary to the results in [321].

4.7.4.2 Short cantilever using three materials and void

The same example as in the previous paragraph is considered here with an additional phase: half of the volume of material 0 is replaced by a weaker material 1. More precisely, the Young moduli of the four phases are defined as $E^0 = 0.5, E^1 = 0.25, E^2 = 1$ and $E^3 = 10^{-3}$, while the target volumes for the three materials 0, 1 and 2 are set to $V_T^0 = V_T^1 = V_T^2 = 0.1|D|$. The initial and optimal shapes (after 200 iterations) are displayed on Figure 4.19 (c) and (d).

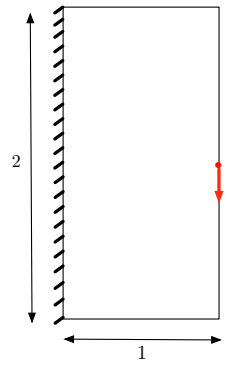


Figure 4.18: Boundary conditions and initialization for the short cantilever.

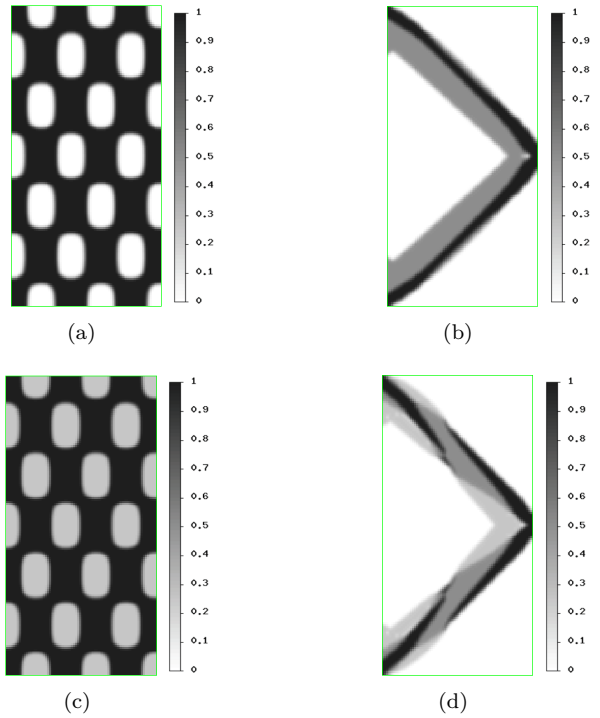


Figure 4.19: Short cantilever using two or three phases and void; *(a)* initialization for two phases and void, *(b)* optimal shape for two phases and void, *(c)* initialization for three phases and void, *(d)* optimal shape for three phases and void.

This test case was also studied in [321] (see figures 11 and 12 therein for two different initializations). Our result differs notably from these previous ones. Indeed, in [321] the strong material 2 always forms a two-bar truss which is further reinforced by the other materials. On the contrary, in our Figure 4.19 (d) the strong phase is disconnected and the intermediate material 0 plays a more active role in the transfer of the load to the fixed wall.

4.7.4.3 3-force bridge using two materials and void

A bridge-type structure of dimensions 2×1 is discretized by 160×80 $\mathbb{Q}^1$ elements. Both the horizontal and vertical displacement are fixed at the lower left part as well as the vertical displacement of the lower right part. Three equally spaced forces are applied at the lower part (see Figure 4.20). The value of F is set to 1. The Young moduli of the four phases are set to $E^0 = 0.5, E^1 = 10^{-3}, E^2 = 1$ and $E^3 = 10^{-3}$ and the target volumes for phases 0 and 2 are set to $V_T^0 = 0.2|D|$ and $V_T^2 = 0.1|D|$. The initial and optimal designs (after 250 iterations) are shown in Figure 4.21 (a) and (b).

Once again this test case was performed in [321] (see figure 13 therein). Our result is quite different. First, our design in Figure 4.21 (b) is symmetric, as it should be. Second, a major difference occurs in the use of the strong phase. In our design, the strong material is used in the lower part of the 'radial' bars whereas it was absent in figure 13 of [321] (and rather used in the upper 'arch').

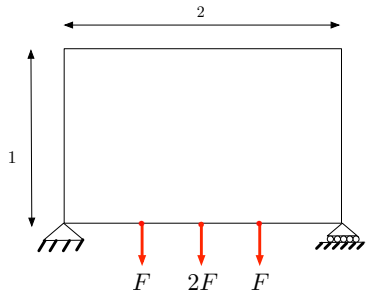


Figure 4.20: Boundary conditions for the 3-force bridge.

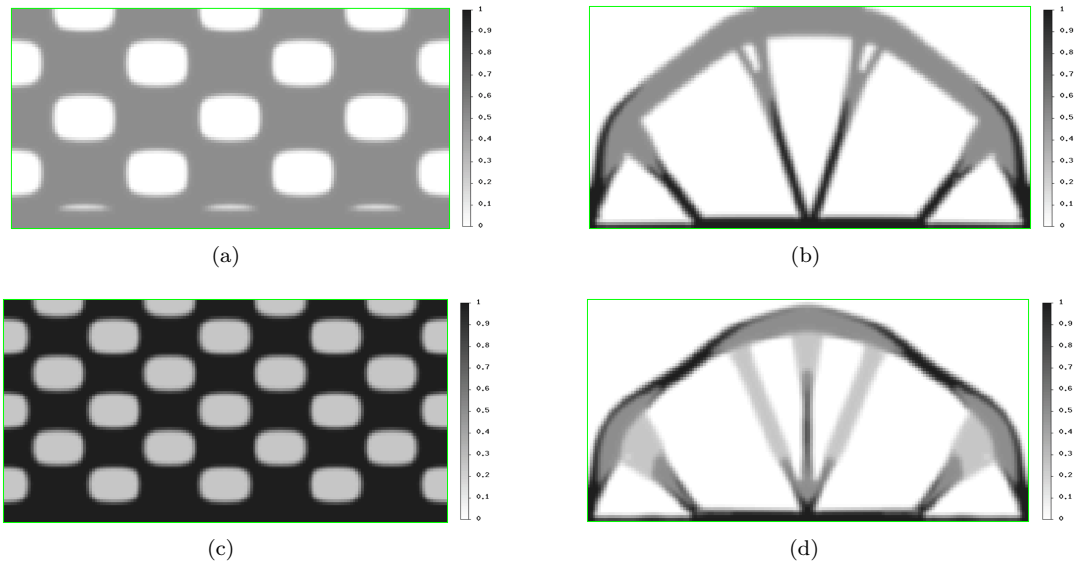


Figure 4.21: 3-force bridge using two or three phases and void; *(a)* initialization for two phases and void, *(b)* optimal shape for two phases and void, *(c)* initialization for three phases and void, *(d)* optimal shape for three phases and void.

4.7.4.4 3-force bridge using three materials and void

The same example as in the previous paragraph is considered here with an additional phase: half of the volume of material 0 is replaced by a weaker material 1. The Young moduli of the four phases are defined as $E^0 = 0.5, E^1 = 0.25, E^2 = 1$ and $E^3 = 10^{-3}$, while the target volumes for phases 0, 1 and 2 are set to $V_T^0 = V_T^1 = V_T^2 = 0.1|D|$. The initial and optimal designs (after 250 iterations) are displayed on Figure 4.19 (c) and (d).

This test case can be found in [321] (figure 14) too, and again our result is quite different.

4.7.4.5 Medium cantilever using three materials and void

The next structure is a medium cantilever of dimensions 3.2×2, discretized using 120×75 $Q1$ elements. The left part of the structure is clamped and a unitary vertical force is applied at the bottom of its right part (see Figure 4.22). The Young moduli of the four phases are again set to $E^0 = 0.5, E^1 = 0.25, E^2 = 1$ and $E^3 = 10^{-3}$, while the target volumes for phases 0, 1 and 2 are $V_T^0 = V_T^1 = V_T^2 = 0.1|D|$. The initial and optimal shapes (after 250 iterations) are shown in Figure 4.23.

This test case was also performed in [322] (see figure 7 therein). Our optimal design has a more complex topology and a different layout of the three materials. However, the final volumes of the three materials in [322] are not the same as ours and thus a comparison is not easy to establish.

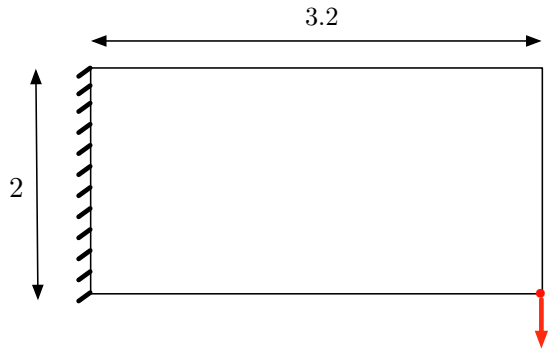
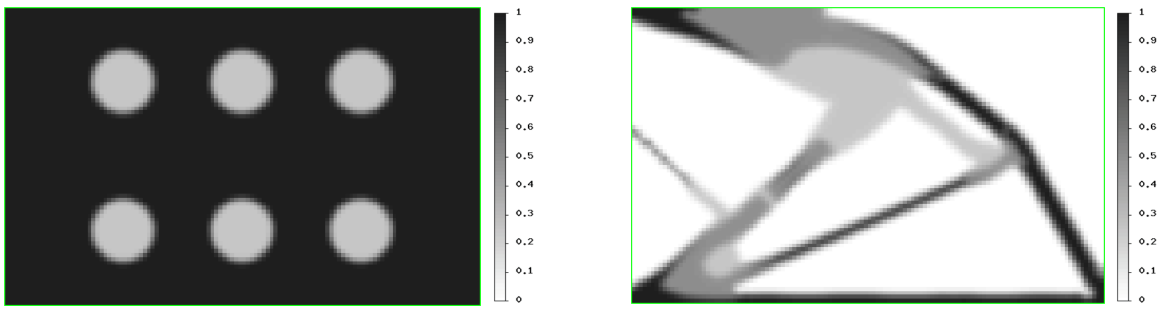


Figure 4.22: Boundary conditions and initialization for the medium cantilever.



(a) (b)

Figure 4.23: *Medium cantilever using three materials and void; (a) initialization, (b) optimal shape.*

4.7.4.6 Long cantilever using two materials and void

The goal of this last paragraph is twofold. First, we consider again the 2×1 long cantilever, as in Figure 4.13, but with four phases, defined by their Young moduli $E^0 = 0.5, E^1 = 10^{-3}, E^2 = 1$ and $E^3 = 10^{-3}$. Second, we switch to an unconstrained optimization algorithm. We do not impose equality constraints for the volume of each phase. Rather, we fix Lagrange multipliers and we minimize an objective function $J(\mathcal{O}^0, \mathcal{O}^1)$, which reads

$$J(\mathcal{O}^0, \mathcal{O}^1) = \int_D A(d_{\mathcal{O}^0}, d_{\mathcal{O}^1}) e(u) : e(u) dx + \sum_{i=0}^{3} \ell^i \int_D \chi_i(x) dx. \tag{4.60}$$

We then carry out a standard constraint-free steepest descent algorithm in order to minimize J.

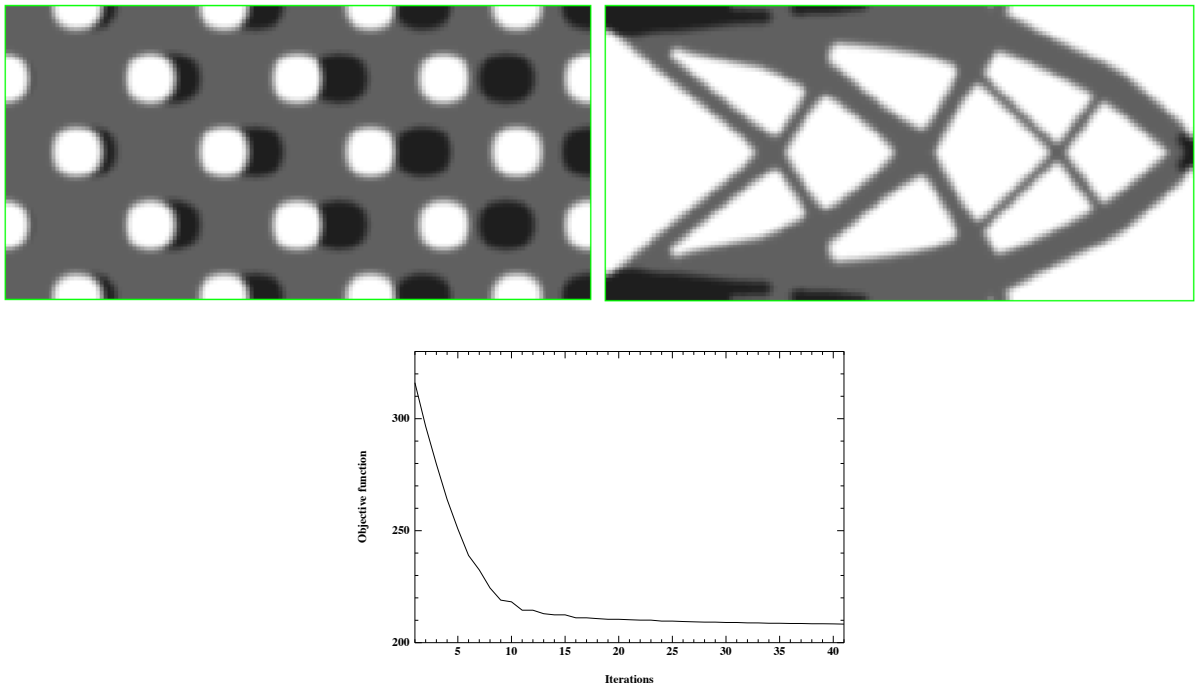


Figure 4.24: Initialization with two materials *(top left)*, optimal shape *(top right)* and convergence history of the objective function *(bottom)*.

A small tolerance parameter $tol > 0$ (in the example below, we used $tol = 0.02$) over acceptance of the produced shapes is introduced so as to ease the occurrence of topological changes and is then turned off after some iterations. More accurately, in the course of the optimization process, a step $\mathcal{O}_n^0 \to \mathcal{O}_{n+1}^0$ and $\mathcal{O}_n^1 \to \mathcal{O}_{n+1}^1$ is accepted provided:

$$J(\mathcal{O}_{n+1}^0, \mathcal{O}_{n+1}^1) < (1 + tol) J(\mathcal{O}_n^0, \mathcal{O}_n^1).$$

For the results shown in Figure 4.24, the Lagrange multipliers in (4.60) are set to $\ell^0 = 100$, $\ell^1 = 0$, $\ell^2 = 200$, $\ell^3 = 0$. As can be expected the strong material is distributed at the areas of high stress, while the weak material completes the shape of the optimal cantilever.

It is interesting to see the optimal subdomains $\mathcal{O}^0$ and $\mathcal{O}^1$ (defined in Section 4.5) in Figure 4.25. Recall that it is the intersections of these two subdomains and their complementaries which give rise to the phase domains in the optimal design of Figure 4.24. Nevertheless, $\mathcal{O}^0$ and $\mathcal{O}^1$ are important from a numerical point since, rather than the phase domains, they are advected by the shape gradients and represented by the level set functions.



Figure 4.25: Final subdomains $\mathcal{O}^0$ *(left)* and $\mathcal{O}^1$ *(right)*.

4.8 Appendix: convergence of the smoothed-interface shape optimization problem to its sharp-interface equivalent

In this appendix, we focus on the demonstration of the convergence result of section 4.4.4. As a first step, in section 4.8.1, we investigate into the case of a simplified model in thermal conductivity, before turning, in section 4.8.2, to the case of the linearized elasticity system, which is very similar in our considerations, yet involving more tedious algebra.

The results and proofs concentrated in this section do not seem to exist as is in the literature, however very reminiscent of previous works they are. They rely (in a probably non optimal way) of classical methods in the theory of elliptic equations.

4.8.1 A model problem in the context of thermal conductivity

Consider the situation depicted in figure 4.26: Γ is a $\mathcal{C}^2$ compact submanifold of $\mathbb{R}^d$, delimiting two complementary subdomains Ω^0 and Ω^1 in a larger (bounded) computational domain D. We assume that $\Omega^0 \subset\subset D$, so that $\partial\Omega^0 \cap \partial D = \emptyset$.

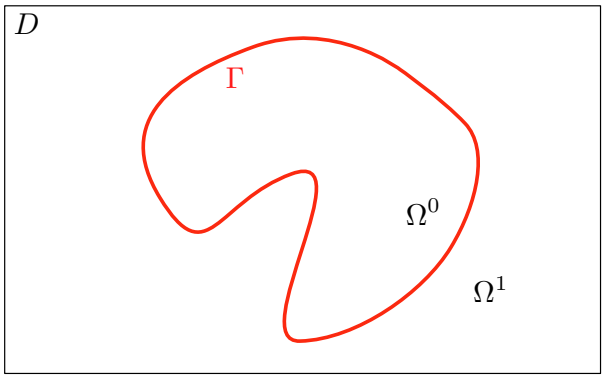


Figure 4.26: The considered situation: two distinct phases are separated by an interface Γ.

For small enough $\varepsilon > 0$, one defines $u_\varepsilon \in H^1_0(D)$ as the solution to the *smoothed-interface problem*:

$$\left\{ \begin{array}{rcl} -\mathrm{div}(A_\varepsilon \nabla u_\varepsilon) = f & \text{in} & D \\ u_\varepsilon = 0 & \text{on} & \partial D \end{array} \right. , \tag{4.61}$$

and $u \in H_0^1(D)$ as the solution to the *sharp-interface problem*:

$$\left\{ \begin{array}{ccc} -\mathrm{div}(A\nabla u) = f & \text{in} & D \\ u = 0 & \text{on} & \partial D \end{array} \right. , \tag{4.62}$$

where the source term f belongs to $L^\infty(D)$, the conductivity matrix $A \in L^\infty(D)^{d^2}$ is discontinuous across the interface Γ, say $A = A_0\chi_0 + A_1\chi_1$ -χ_i standing for the characteristic function of the phase Ω^i-, and A_ε is a smooth interpolation between A_0 and A_1 in the layer of (small) width 2ε around Γ:

$$A_\varepsilon(x) = A_0 + h_\varepsilon(d_{\Omega^0}(x))(A_1 - A_0). \tag{4.63}$$

where $h_\varepsilon : \mathbb{R} \to \mathbb{R}$ is the smooth approximation of the Heaviside function defined by (4.35). Remark that notations have been slightly altered with respect to those of the previous sections; in particular, the dependence of A and A_ε on Ω^0 has been dropped since variations of Γ are not considered in this section.

Remark 4.18. At first glance, the setting of the problem could seem a bit restrictive because of the choice of Dirichlet's homogeneous boundary conditions on ∂D. Actually, imposing other kinds of boundary conditions would not change a word in the forthcoming analysis, since we are solely interested in what happens in the neighborhood of Γ, which lies 'far' from ∂D, and all the regularity results of interest (e.g. Meyer's theorem 4.10) are local.

Consider now the compliance of the resulting structure D from the distribution of materials accounted for by the partition of D into Ω^0 and Ω^1, in the smoothed-interface context:

$$J_\varepsilon(\Omega^0) = \int_D f\, u_\varepsilon\, dx.$$

The conclusions of this section would hold in the case of any 'reasonable' objective function of the domain, up to some additional regularity assumptions on the data at hand, but, for the sake of simplicity, we will only focus on this case.

In section 4.4.2, we computed the sensitivity of J_ε with respect to the shape of this partitioning of D (in truth, we performed the equivalent computation in the linear elasticity setting); it is described by the following shape gradient:

$$\forall \theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ \ J'_\varepsilon(\Omega^0)(\theta) = \int_\Gamma \left[f_0^\varepsilon(x) + f_1^\varepsilon(x)\right](\theta(x).n(x))\ ds(x), \tag{4.64}$$

where for all $x \in \Gamma$, $n(x)$ stands for the unit normal vector to Γ (pointing outward Ω^0), and:

$$f_0^\varepsilon(x) = \int_{p_\Gamma^{-1}(x)\cap\Omega_0} h'_\varepsilon\left(d_{\Omega^0}(y)\right) \prod_{i=1}^{d-1} \left(1 + d_{\Omega^0}(y)\kappa_i(x)\right) (A_1 - A_0)\, \nabla u_\varepsilon(y).\nabla u_\varepsilon(y)\, dy,$$

$$f_1^\varepsilon(x) = \int_{p_\Gamma^{-1}(x)\cap\Omega_1} h'_\varepsilon\left(d_{\Omega^0}(y)\right) \prod_{i=1}^{d-1} \left(1 + d_{\Omega^0}(y)\kappa_i(x)\right) (A_1 - A_0)\, \nabla u_\varepsilon(y).\nabla u_\varepsilon(y)\, dy.$$

The purpose of this appendix is to study the convergence of $J'_\varepsilon(\Omega^0)(\theta)$, for fixed Ω^0 and θ, as the thickness ε of the smooth interpolation layer between the two different conductivities goes to 0.

4.8.1.1 Study of the asymptotic behavior of u_ε as $\varepsilon \to 0$.

Let us first recall the following regularity results arounf u_ε and u, under the above hypotheses :

– Since the conductivity matrix A_ε is smooth (and uniformly coercive) over D, for any $\varepsilon > 0$, the solution u_ε to (4.61) belongs to $H^2(D)$ (and even to $W^{2,p}(D)$ for $1 \leq p < \infty$). This stems from the 'classical' elliptic regularity theory [57].
– The solution u to (4.62) is less regular : actually, one can show [222] that $u|_{\Omega^0} \in W^{2,p}(\Omega^0)$, and similarly, that $u|_{\Omega^1} \in W^{2,p}(\Omega^1)$, for any $1 \leq p < \infty$.

The first thing to study is in which extent u_ε converges to u, when $\varepsilon \to 0$. To this end, we will use repeatedly Meyer's theorem (see [41], chap. 1, sec. 4):

Theorem 4.10. *Let $\Omega \subset \mathbb{R}^d$ a bounded open domain of class $\mathcal{C}^2$. Let $0 < \alpha < \beta$ two real numbers, and $A \in L^\infty(\Omega)^{d \times d}$ a matrix-valued application on Ω, such that*

$$\forall x \in \Omega, \forall \xi \in \mathbb{R}^d, \ \alpha|\xi|^2 \leq A(x)(\xi,\xi) \leq \beta|\xi|^2.$$

For $f \in H^{-1}(\Omega)$, let u be the unique solution in $H_0^1(\Omega)$ of :

$$\left\{ \begin{array}{ccc} -\mathrm{div}(A\nabla u) = f & in & \Omega \\ u = 0 & on & \partial\Omega \end{array} \right. ,$$

There exists a number $p > 2$ and a constant $C > 0$ which both depend only on α, β and Ω, such that, if f belongs to $W^{-1,p}(\Omega)$, then u belongs to $W_0^{1,p}(\Omega)$ and satisfies:

$$||u||_{W_0^{1,p}(\Omega)} \leq C||f||_{W^{-1,p}(\Omega)}.$$

Introducing ellipticity $\alpha > 0$ and continuity $\beta > 0$ constants available for A_0 and A_1, we denote thenceforth as $p > 2$ various exponents depending only on D (and possibly on fixed subdomains of D), α, β, supplied by repeated uses of Meyer's theorem. Now we can prove:

Lemma 4.5. *The solution u_ε to (4.61) converges towards the solution u to (4.62) in $H_0^1(D)$.*

Proof. First note that, as a consequence of Lebesgue's dominated convergence theorem, one has:

$$\forall p < \infty, \ A_\epsilon \overset{\varepsilon \to 0}{\longrightarrow} A \text{ in } L^p(D)^{d^2}, \tag{4.65}$$

which obviously fails to happen in $L^\infty(D)$, since A is not even continuous. Now, for any function $v \in H_0^1(D)$,

$$\int_D A_\varepsilon(x)\nabla u_\varepsilon(x) \cdot \nabla v(x)dx = \int_D f(x)\, v(x)dx \quad ; \quad \int_D A(x)\nabla u(x) \cdot \nabla v(x)\, dx = \int_D f(x)\, v(x)dx.$$

Consequently, we get:

$$\int_D A_\varepsilon(x)\nabla(u_\varepsilon - u)(x) \cdot \nabla v(x)dx = \int_D (A - A_\varepsilon)(x)\nabla u(x) \cdot \nabla v(x)dx. \tag{4.66}$$

Putting $v = u - u_\varepsilon$ in (4.66) and using Hölder's inequality yields:

$$\alpha||u - u_\varepsilon||^2_{H_0^1(D)} \leq ||A - A_\varepsilon||_{L^r(D)}\, ||u||_{W_0^{1,p}(D)}\, ||u - u_\varepsilon||_{H_0^1(D)},$$

for some $r > 0$ such that $\frac{1}{r} + \frac{1}{p} + \frac{1}{2} = 1$ (which is possible since $p > 2$). Hence, there exists a constant $C > 0$ such that:

$$\alpha||u - u_\varepsilon||_{H_0^1(D)} \leq C\, ||A - A_\varepsilon||_{L^r(D)}\, ||f||_{W^{-1,p}(D)},$$

and the result follows. □

Actually, the situation is a bit better than that, owing to the following lemma:

Lemma 4.6. *Let $\Omega \subset \mathbb{R}^d$ any open domain, u_ε a sequence of functions of some $L^p(\Omega)$, for $p > 2$, $u \in L^p(\Omega)$, and suppose:*

– *u_ε converges to u strongly in $L^2(\Omega)$.*
– *there exists $C>0$ such that $||u_\varepsilon||_{L^p(\Omega)} \leq C$ for every ε.*
Then u_ε converges to u strongly in $L^q(\Omega)$, for any $2 \leq q < p$.

Proof. Define $\theta \in (0,1)$ such that $q = 2\theta + p(1-\theta)$. Then one has :

$$\int_\Omega |u-u_\varepsilon|^q dx = \int_\Omega \left(|u-u_\varepsilon|^2\right)^\theta \left(|u-u_\varepsilon|^p\right)^{1-\theta} dx.$$

Now, using Hölder's inequality between $\left(|u-u_\varepsilon|^2\right)^\theta \in L^{\frac{1}{\theta}}(\Omega)$ and $\left(|u-u_\varepsilon|^p\right)^{1-\theta} \in L^{\frac{1}{1-\theta}}(\Omega)$, we get :

$$\int_\Omega |u-u_\varepsilon|^q dx \leq \left(\int_\Omega |u-u_\varepsilon|^2 dx\right)^{\frac{1}{\theta}} \left(\int_\Omega |u-u_\varepsilon|^p dx\right)^{\frac{1}{1-\theta}},$$

whence

$$||u-u_\varepsilon||_{L^q(\Omega)} \leq ||u-u_\varepsilon||_{L^2(\Omega)}^{\frac{2}{q\theta}} ||u-u_\varepsilon||_{L^p(\Omega)}^{\frac{p}{q(1-\theta)}}.$$

Eventually, $||u-u_\varepsilon||_{L^2(\Omega)}^{\frac{2}{q\theta}}$ goes to 0 as $\varepsilon \to 0$, while $||u-u_\varepsilon||_{L^p(\Omega)}^{\frac{p}{q(1-\theta)}}$ is bounded, owing to the second hypothesis, which ends the proof. □

Putting together lemmas 4.5 and 4.6, and another use of Meyer's theorem 4.10 shows that actually, u_ε goes to u in every $W_0^{1,q}(D)$, for $2 \leq q < p$.

Now, we investigate convergence of the higher-order derivatives of u_ε. Actually, we are only interested in what happens very close to Γ. Since Γ is of class $\mathcal{C}^2$, let us introduce a tubular neighborhood $\Gamma \subset V \subset D$ as in proposition 4.1, (5). As we have seen, one can then introduce an extension of the normal vector field $n \in \mathcal{C}^1(\Gamma)$ to V, still denoted n, as:

$$\forall x \in V,\ n(x) = \nabla d_{\Omega^0}(x),$$

and similarly, one can construct an extension (still denoted as τ) of any vector field $\tau \in T\Gamma$ on Γ to V by considering $\tau(p_{\partial\Omega}(x))$.

The result of interest is the following:

Proposition 4.13. *Let $\tau \in T\Gamma$ be any $\mathcal{C}^1$ vector field on Γ. The following strong convergence results hold true:*

$$\begin{array}{ccccc} \frac{\partial u_\varepsilon}{\partial \tau} & \overset{\varepsilon\to 0}{\longrightarrow} & \frac{\partial u}{\partial \tau} & \text{in } H^1(V) \text{ strong}, \\ (A_\varepsilon \nabla u_\varepsilon)\cdot n & \overset{\varepsilon\to 0}{\longrightarrow} & (A\nabla u)\cdot n & \text{in } H^1(V) \text{ strong} \end{array}.$$

Proof. The proof unfolds exactly as that of theorem 9.25 in [57], except that there is no need to use the method of difference quotients to perform integration by parts (since we already know that the handled functions enjoy enough regularity), and that a priori estimates resulting from Meyer's theorem 4.10 are used to get the desired convergence result.

Before passing to the proof of proposition 4.13 itself, let us operate several simplifications in the problem, owing to a standard argument of partitions of unity. Let $Q = [-1,1]^d \subset \mathbb{R}^d$, and denote:

$$\begin{array}{rcl} Q_+ & = & \{x = (x_1,...,x_d) \in Q,\ x_d > 0\}, \\ Q_- & = & \{x = (x_1,...,x_d) \in Q,\ x_d < 0\}, \\ Q_0 & = & \{x = (x_1,...,x_d) \in Q,\ x_d = 0\} \end{array}.$$

As Γ is compact, it can be covered by finitely many open sets $W_j \subset D$, $j = 1,...,J$, such that, for each index j, there exists a $\mathcal{C}^2$ diffeomorphism $\phi_j : Q \to W_j$ with the properties:

$$\phi_j(Q_-) = W_j \cap \Omega^0,\ \phi_j(Q_+) = W_j \cap \Omega^1,\ \phi_j(Q_0) = W_j \cap \Gamma.$$

Considering a partition of unity $\{\alpha_j\}_{j=1,\dots,J}$ associated to this covering, that is, for every j, α_j is a $\mathcal{C}^2$ function with compact support included in W_j, and one has: $\sum_j \alpha_j = 1$ on D. It is then enough to prove proposition 4.13 with χu_ε and χu instead of u_ε and u respectively, where χ stands for any of the α_j above, associated to $W = W_j$ (see figure 4.27).

Now, we restrict (the variational formulations of) equations (4.61-4.62) by expressing the problems solved

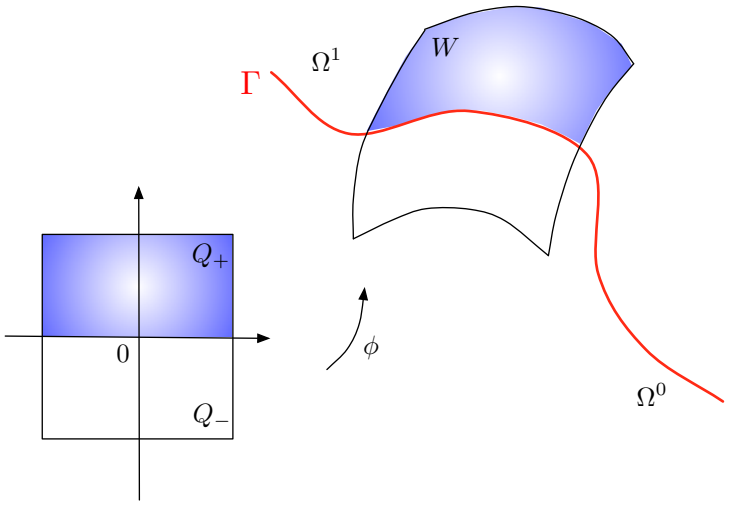


Figure 4.27: Reduction in the proof of proposition 4.13.

by $v_\varepsilon := \chi u_\varepsilon$, and $v := \chi u$. We find, after some computation, that, for all test functions $\varphi \in H_0^1(W)$:

$$\int_V A_\varepsilon \nabla v_\varepsilon \cdot \nabla \varphi \, dx = \int_V g_\varepsilon \varphi \, dx + \int_V h_\varepsilon \cdot \nabla \varphi \, dx, \tag{4.67}$$

$$\int_V A \nabla v \cdot \nabla \varphi \, dx = \int_V g \varphi \, dx + \int_V h \cdot \nabla \varphi \, dx, \tag{4.68}$$

where we have defined:

$$\left\{ \begin{array}{rcl} g_\varepsilon & = & \chi f - A_\varepsilon \nabla u_\varepsilon \cdot \nabla \chi \\ g & = & \chi f - A \nabla u \cdot \nabla \chi \end{array} \right. , \quad \left\{ \begin{array}{rcl} h_\varepsilon & = & u_\varepsilon A_\varepsilon \nabla \chi \\ h & = & u A \nabla \chi \end{array} \right. .$$

We then pull equations (4.67-4.68) back to Q, using a change of variables. Again, after some computations, we find that $\widetilde{v_\varepsilon} := v_\varepsilon \circ \phi$ and $\widetilde{v} := v \circ \phi$ are respectively solution to: for any $\psi \in H_0^1(Q)$,

$$\int_Q \widetilde{A_\varepsilon} \nabla \widetilde{v_\varepsilon} \cdot \nabla \psi \, dx = \int_Q \widetilde{g_\varepsilon} \psi \, dx + \int_Q \widetilde{h_\varepsilon} \cdot \nabla \psi \, dx, \tag{4.69}$$

$$\int_Q \widetilde{A} \nabla \widetilde{v} \cdot \nabla \psi \, dx = \int_Q \widetilde{g} \psi \, dx + \int_Q \widetilde{h} \cdot \nabla \psi \, dx, \tag{4.70}$$

where

$$\widetilde{g_\varepsilon} := |\det(\nabla \phi)| \, g_\varepsilon \circ \phi, \;\; \widetilde{g} := |\det(\nabla \phi)| \, g \circ \phi,$$
$$\widetilde{h_\varepsilon} := |\det(\nabla \phi)| \, h_\varepsilon \circ \phi, \;\; \widetilde{h} := |\det(\nabla \phi)| \, h \circ \phi,$$

and:

$$\widetilde{A_\varepsilon} := |\det(\nabla \phi)| \, \nabla \phi^{-1} \, (A_\varepsilon \circ \phi) \, \nabla \phi^{-T}, \;\; \widetilde{A} := |\det(\nabla \phi)| \, \nabla \phi^{-1} \, (A \circ \phi) \, \nabla \phi^{-T}.$$

With this done, we have, as far as the quantity $(\widetilde{v_\varepsilon} - \widetilde{v})$ is concerned, for any $\psi \in H_0^1(Q)$:

$$\int_Q \widetilde{A_\varepsilon}\nabla\left(\widetilde{v_\varepsilon} - \widetilde{v}\right)\cdot\nabla\psi\, dx = \int_Q \left(\widetilde{g_\varepsilon} - \widetilde{g}\right)\psi\, dx + \int_Q \left(\widetilde{h_\varepsilon} - \widetilde{h}\right)\cdot\nabla\psi\, dx + \int_Q \left(\widetilde{A} - \widetilde{A_\varepsilon}\right)\nabla\widetilde{v}\cdot\nabla\psi\, dx. \quad (4.71)$$

• *We are now in position to prove the first point of proposition 4.13.* Note that, because of the linearity of $\tau \mapsto \frac{\partial u_\varepsilon}{\partial \tau}$, it is enough to prove that the result holds for any of the vector fields τ_i $(i = 1, ..., d-1)$ on V defined as:

$$\forall y \in V,\ \ \tau_i(y) := \nabla\phi(\phi^{-1}(y))e_i,$$

where $e_1, ..., e_d$ are the vectors of the canonical basis of $\mathbb{R}^d$.

Because $\frac{\partial\widetilde{v_\varepsilon}}{\partial x_i} = \frac{\partial v_\varepsilon}{\partial \tau_i}$ (and similarly with $\widetilde{v}$ and v instead of $\widetilde{v_\varepsilon}$ and v_ε), we show that

$$\forall i = 1, ..., d-1,\ \ \frac{\partial\widetilde{v_\varepsilon}}{\partial x_i} \longrightarrow \frac{\partial\widetilde{v}}{\partial x_i} \text{ strongly in } H_0^1(Q).$$

Let $\psi \in \mathcal{C}_c^\infty(Q)$; applying (4.71) with $\frac{\partial\psi}{\partial x_i}$ as a test function, then integrating by parts leads to:

$$\begin{aligned}
\int_Q \widetilde{A_\varepsilon}\nabla\left(\frac{\partial\widetilde{v_\varepsilon}}{\partial x_i} - \frac{\partial\widetilde{v}}{\partial x_i}\right)\cdot\nabla\psi\, dx \;=\; & -\int_Q \left[\frac{\partial\left|\det\left(\nabla\phi\right)\right|}{\partial x_i}\nabla\phi^{-1}\left(A_\varepsilon\circ\phi\right)\nabla\phi^{-T}\right]\nabla\left(\widetilde{v_\varepsilon} - \widetilde{v}\right)\cdot\nabla\psi\, dx \\
& -\int_Q \left[\frac{\partial\nabla\phi^{-1}}{\partial x_i}\left(A_\varepsilon\circ\phi\right)\nabla\phi^{-T}\right]\nabla\left(\widetilde{v_\varepsilon} - \widetilde{v}\right)\cdot\nabla\psi\, dx \\
& -\int_Q \left[\nabla\phi^{-1}\left(A_\varepsilon\circ\phi\right)\frac{\partial\nabla\phi^{-T}}{\partial x_i}\right]\nabla\left(\widetilde{v_\varepsilon} - \widetilde{v}\right)\cdot\nabla\psi\, dx \\
& -\int_Q \left(\widetilde{g_\varepsilon} - \widetilde{g}\right)\frac{\partial\psi}{\partial x_i}\, dx + \int_Q \left(\frac{\partial\widetilde{h_\varepsilon}}{\partial x_i} - \frac{\partial\widetilde{h}}{\partial x_i}\right)\cdot\nabla\psi\, dx \\
& +\int_Q \left(\widetilde{A} - \widetilde{A_\varepsilon}\right)\nabla\left(\frac{\partial\widetilde{v}}{\partial x_i}\right)\cdot\nabla\psi\, dx
\end{aligned} \quad , \quad (4.72)$$

where we used the facts that $\widetilde{v_\varepsilon} \in H^2(Q), \widetilde{v} \in H^2(Q_+) \cap H^2(Q_-)$, and that $\frac{\partial\widetilde{A_\varepsilon}}{\partial x_i} = \frac{\partial\widetilde{A}}{\partial x_i} = 0$. Note also that, because $u_\varepsilon \to u$ in some $W^{1,p}(D)$, this equality also holds for $\psi \in H_0^1(Q)$ owing to a standard density argument. Eventually, letting $\psi = \left(\frac{\partial\widetilde{v_\varepsilon}}{\partial x_i} - \frac{\partial\widetilde{v}}{\partial x_i}\right)$ in (4.72) and applying Hölder's inequality as above leads to:

$$\alpha\left|\left|\frac{\partial\widetilde{v_\varepsilon}}{\partial x_i} - \frac{\partial\widetilde{v}}{\partial x_i}\right|\right|_{H_0^1(Q)} \leq C\left(||\widetilde{v_\varepsilon} - \widetilde{v}||_{W_0^{1,p}(Q)} + ||\widetilde{g_\varepsilon} - \widetilde{g}||_{L^2(Q)} + \left|\left|\widetilde{h_\varepsilon} - \widetilde{h}\right|\right|_{L^2(Q)^d} + \left|\left|\widetilde{A_\varepsilon} - \widetilde{A}\right|\right|_{L^r(Q)^{d^2}}\right),$$

for some $r > 0$ large enough, and some constant C which does not depend on ε. Applying once again Hölder's inequality to deal with the terms $||\widetilde{g_\varepsilon} - \widetilde{g}||_{L^2(Q)}$ and $\left|\left|\widetilde{h_\varepsilon} - \widetilde{h}\right|\right|_{L^2(Q)^d}$ eventually yields the result. Note that, owing to Meyer's theorem, this last conclusion even holds in some space $W_0^{1,p}(Q)$, for some $p > 2$

• *We now prove that* $(A_\varepsilon\nabla u_\varepsilon)\cdot n \to (A\nabla u)\cdot n$ *strongly in* $H^1(V)$, which boils down to showing that $\left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon}\right)\cdot e_d \to \left(\widetilde{A}\nabla\widetilde{v}\right)\cdot e_d$ strongly in $H_0^1(Q)$. Still in the spirit of [57], the previous point already made it clear that, for $i = 1, ..., d-1$,

$$\frac{\partial}{\partial x_i}\left(\left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon}\right)\cdot e_d\right) \to \frac{\partial}{\partial x_i}\left(\left(\widetilde{A}\nabla\widetilde{v}\right)\cdot e_d\right) \text{ strongly in } L^2(Q),$$

so that we are left with proving:

$$\frac{\partial}{\partial x_d}\left(\left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon}\right)\cdot e_d\right) \to \frac{\partial}{\partial x_d}\left(\left(\widetilde{A}\nabla\widetilde{v}\right)\cdot e_d\right) \text{ strongly in } L^2(Q).$$

To achieve this, we return to the original variational formulations (4.69,4.70) for $\widetilde{v_\varepsilon}$ and $\widetilde{v}$, and make appear the term of interest; for any $\psi \in H_0^1(Q)$, one has:

$$\int_Q \left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon} - \widetilde{A}\nabla\widetilde{v}\right).e_d \,\frac{\partial\psi}{\partial x_d}\,dx = -\sum_{i=1}^{d-1}\int_Q \left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon} - \widetilde{A}\nabla\widetilde{v}\right).e_i \,\frac{\partial\psi}{\partial x_i}\,dx + \int_Q (\widetilde{g_\varepsilon} - \widetilde{g})\,\psi\,dx + \int_Q \left(\widetilde{h_\varepsilon} - \widetilde{h}\right)\cdot\nabla\psi\,dx.$$

We can now conclude as for the first point: performing integration by parts an using the conclusions of the first point leads to, for any $\psi \in H_0^1(Q)$,

$$\begin{aligned}\int_Q \frac{\partial}{\partial x_d}\left(\left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon} - \widetilde{A}\nabla\widetilde{v}\right).e_d\right)\psi\,dx \;=\; & -\sum_{i=1}^{d-1}\int_Q \frac{\partial}{\partial x_i}\left(\left(\widetilde{A_\varepsilon}\nabla\widetilde{v_\varepsilon} - \widetilde{A}\nabla\widetilde{v}\right).e_i\right)\psi\,dx - \int_Q (\widetilde{g_\varepsilon} - \widetilde{g})\,\psi\,dx \\ & + \int_Q \operatorname{div}\left(\widetilde{h_\varepsilon} - \widetilde{h}\right)\psi\,dx\end{aligned},$$

and using the a priori estimates at our disposal in the same way as before leads to the desired conclusion, which ends the proof. □

4.8.1.2 Convergence of the shape gradient associated to the smoothed-interface problem as $\varepsilon \to 0$.

Our purpose is now to pass to the limit $\varepsilon \to 0$ in expression (4.64). To achieve this, we can first assume that ε is so small that every point $x \in D$ with $|d_{\Omega^0}(x)| < \varepsilon$ has a unique projection over Γ (with the language of section 4.2.5, $\varepsilon < \text{reach}(\Gamma)$). Then,

$$\forall x \in \Gamma,\;\; f_1^\varepsilon(x) = \int_0^\varepsilon h'_\varepsilon(s)\,k_\varepsilon(x,s)\,(A_1 - A_0)\,\nabla u_\varepsilon(x + sn(x))\cdot\nabla u_\varepsilon(x + sn(x))\,ds,$$

where we have introduced $k_\varepsilon(x,s) = \prod_{i=1}^{d-1}(1 + s\kappa_i(x))$, and a similar expression holds for f_0^ε.

Now, the idea consists in using the convergence results of section 4.8.1.1 for $\frac{\partial u_\varepsilon}{\partial\tau}$ and $(A_\varepsilon\nabla u_\varepsilon).n$. It requires decomposing the gradient ∇u_ε into its tangential and normal components, bringing into play the components that behave 'nicely' in the limit $\varepsilon \to 0$, in light of proposition 4.13.

For the sake of simplicity, we will dwell on the case when these tensors are *isotropic*, that is

$$A = \alpha I_2,\;\; \alpha(x) = \alpha_i \text{ if } x \in \Omega^i,\;\text{ for } \alpha_i > 0 \text{ fixed},$$

although the whole argument would hold in the general case, but yield to more complex expressions as far as the limit of $J'_\varepsilon(\Omega^0)(\theta)$ is concerned. Accordingly, we shall also denote $A_\varepsilon = \alpha_\varepsilon I$, with obvious signification.

We need a small technical result to perform our study:

Proposition 4.14. *Let $\Omega \subset \mathbb{R}^d$ a bounded open domain of class $\mathcal{C}^2$, n the corresponding unit outer normal vector field, and $\varepsilon < \varepsilon_0 < reach(\partial\Omega)$, so that for any $x \in \partial\Omega$, and any $0 < s < \varepsilon$, $x - sn(x) \in \Omega$. Let $v : [0,1] \to \mathbb{R}$ a continuous function, and f_ε, $g_\varepsilon \in H^1(\Omega)$ two sequences of functions such that:*

$$f_\varepsilon \overset{\varepsilon\to 0}{\longrightarrow} f,\;\; g_\varepsilon \overset{\varepsilon\to 0}{\longrightarrow} g \text{ strongly in } H^1(\Omega).$$

Then, the sequence $I_\varepsilon \in L^1(\partial\Omega)$, defined as

$$\forall x \in \partial\Omega,\;\; I_\varepsilon(x) = \int_0^1 v(s)\,f_\varepsilon(x - s\varepsilon n(x))g_\varepsilon(x - s\varepsilon n(x))\,ds,$$

converges in $L^1(\partial\Omega)$ to $I(x) := \left(\int_0^1 v(s)ds\right)f(x)g(x)$.

Proof. First, because of the trace theorem, $f_\varepsilon \to f$ and $g_\varepsilon \to g$ in $L^2(\partial\Omega)$. It is then enough to prove that:

$$I_\varepsilon - \left(\int_0^1 v(s)ds\right) f_\varepsilon g_\varepsilon \longrightarrow 0 \text{ in } L^1(\partial\Omega).$$

This arises in turn as a consequence of the following lemma, whose proof is postponed:

Lemma 4.7. *Under the assumptions of proposition 4.14, for any continuous function $v : [0,1] \to \mathbb{R}$, and any functions $f, g \in H^1(\Omega)$, define $J_\varepsilon : \partial\Omega \to \mathbb{R}$ as:*

$$\forall x \in \partial\Omega,\ \ J_\varepsilon(x) = \left(\int_0^1 v(s)\, f(x - s\varepsilon n(x)) g(x - s\varepsilon n(x)) ds\right) - \left(\int_0^1 v(s) ds\right) f(x) g(x).$$

Then, there exists a constant $C > 0$, which depends only on $\partial\Omega$, v and ε_0, such that, for any two functions $f, g \in H^1(\Omega)$, and any $\varepsilon < \varepsilon_0$, one has :

$$||J_\varepsilon||_{L^1(\partial\Omega)} \leq C \left(||f||^2_{H^1(\partial\Omega_\varepsilon)} + ||g||^2_{H^1(\partial\Omega_\varepsilon)} \right),$$

where $\partial\Omega_\varepsilon = \{x \in \Omega,\ d(x, \partial\Omega) < \varepsilon\}$.

Lemma 4.7 implies the existence of a constant C such that:

$$\begin{array}{rcl} \left|\left| I_\varepsilon - \left(\int_0^1 v(s)ds\right) f_\varepsilon g_\varepsilon \right|\right|_{L^1(\partial\Omega)} & \leq & C \left(||f_\varepsilon||^2_{H^1(\partial\Omega_\varepsilon)} + ||g_\varepsilon||^2_{H^1(\partial\Omega_\varepsilon)} \right) \\ & \leq & C \left(||f - f_\varepsilon||^2_{H^1(\Omega)} + ||f||^2_{H^1(\partial\Omega_\varepsilon)} + ||g - g_\varepsilon||^2_{H^1(\Omega)} + ||g||^2_{H^1(\partial\Omega_\varepsilon)} \right), \end{array}$$

and eventually, the right-hand side goes to 0 as $\varepsilon \to 0$, because of the assumptions on $f_\varepsilon, g_\varepsilon$ and f, g, □

We can now state and prove the main convergence result of this section:

Theorem 4.11. *Under the above assumptions, the following convergence result holds:*

$$\lim_{\varepsilon\to 0} J'_\varepsilon(\Omega^0)(\theta) = (\alpha_1 - \alpha_0) \int_\Gamma \nabla_\Gamma u \cdot \nabla_\Gamma u\, (\theta \cdot n)\, ds - (\alpha_1^{-1} - \alpha_0^{-1}) \int_\Gamma \left(\alpha \frac{\partial u}{\partial n}\right) \left(\alpha \frac{\partial u}{\partial n}\right) (\theta \cdot n)\, ds. \qquad (4.73)$$

Proof. First of all, recall that, denoting as $u^i \in H^1(\Omega^i)$ the restriction of u to Ω^i, the *transmission conditions* on Γ read:

$$\nabla_\Gamma u^0(x) = \nabla_\Gamma u^1(x)\ ,\ \ \left(\alpha_0 \frac{\partial u^0}{\partial n}\right)(x) = \left(\alpha_1 \frac{\partial u^1}{\partial n}\right)(x),\ \text{ a.e. } x \in \Gamma,$$

so that the quantities $\nabla_\Gamma u$ and $\left(\alpha \frac{\partial u}{\partial n}\right)$ are well-defined on Γ.

From a simple change of variables and the definition of h, we get:

$$\forall x \in \Gamma,\ \ f_1^\varepsilon(x) = \int_0^1 k_\varepsilon(x,s)\, m'(s)\, \nabla u_\varepsilon(x + s\varepsilon n(x)) \cdot \nabla u_\varepsilon(x + s\varepsilon n(x))\, ds,$$

where m is the function defined as :

$$m(s) = A_\varepsilon(x + s\varepsilon n(x)) = \alpha_0 + \frac{1}{2}\left(1 + s + \frac{1}{\pi}\sin(\pi s)\right)(\alpha_1 - \alpha_0).$$

This is actually the expression in which we have to pass to the limit. To achieve this, we rely on proposition 4.14, separating the tangential and normal components of $\nabla_\Gamma u_\varepsilon$ appearing in the dot product.

$$\begin{array}{rcl} \forall x \in \Gamma,\ \ f_1^\varepsilon(x) & = & \displaystyle\int_0^1 k_\varepsilon(x,s)\, m'(s)\, \nabla_\Gamma u_\varepsilon(x + s\varepsilon n(x)) \cdot \nabla_\Gamma u_\varepsilon(x + s\varepsilon n(x))\, ds \\ & & \displaystyle + \int_0^1 k_\varepsilon(x,s)\, \frac{m'(s)}{m^2(s)} \left(\alpha \frac{\partial u_\varepsilon}{\partial n}\right)(x + s\varepsilon n(x)) \left(\alpha \frac{\partial u_\varepsilon}{\partial n}\right)(x + s\varepsilon n(x))\, ds \end{array}$$

Finally, we end up with $f_1^\varepsilon \to f_1$ in $L^1(\Gamma)$, where f_1 is defined as :

$$\forall x \in \Gamma,\ \ f_1(x) = \left(\int_0^1 m'(s)\, ds\right) \nabla_\Gamma u(x) \cdot \nabla_\Gamma u(x) + \left(\int_0^1 \frac{m'(s)}{m^2(s)}\, ds\right) \left(\alpha \frac{\partial u}{\partial n}\right)(x) \left(\alpha \frac{\partial u}{\partial n}\right)(x),$$

and after some easy computation:

$$\forall x \in \Gamma,\ \ f_1(x) = \left(\alpha_1 - \frac{\alpha_0 + \alpha_1}{2}\right) \nabla_\Gamma u(x) \cdot \nabla_\Gamma u(x) - \left(\frac{1}{\alpha_1} - \frac{2}{\alpha_0 + \alpha_1}\right) \left(\alpha \frac{\partial u}{\partial n}\right)(x) \left(\alpha \frac{\partial u}{\partial n}\right)(x).$$

Similarly, one shows that $f_0^\varepsilon \to f_0$ in $L^1(\Gamma)$, where f_0 is defined as:

$$\forall x \in \Gamma,\ \ f_0(x) = \left(\frac{\alpha_0 + \alpha_1}{2} - \alpha_0\right) \nabla_\Gamma u(x) \cdot \nabla_\Gamma u(x) - \left(\frac{2}{\alpha_0 + \alpha_1} - \frac{1}{\alpha_0}\right) \left(\alpha \frac{\partial u}{\partial n}\right)(x) \left(\alpha \frac{\partial u}{\partial n}\right)(x).$$

Eventually, combining the two latter expressions results in the desired formula. □

Remark 4.19. Fortunately, expression (4.73) is exactly the formula obtained in [248] for the shape gradient of the objective function for the sharp interface problem.

We end this short note with the proof of the missing link in proposition 4.14.

proof of lemma 4.7. By a standard density argument, it is enough to tackle the particular case when f and g are smooth functions, namely $f, g \in \mathcal{C}^\infty(\overline{\Omega})$. In this setting, for any $x \in \partial\Omega$, one has:

$$\begin{array}{rcl} J_\varepsilon(x) & = & \dfrac{1}{\varepsilon}\displaystyle\int_0^\varepsilon v\left(\frac{s}{\varepsilon}\right) f(x - sn(x))\, (g(x - sn(x) - g(x))\, ds + \frac{1}{\varepsilon}\int_0^\varepsilon v\left(\frac{s}{\varepsilon}\right) (f(x - sn(x)) - f(x))\, g(x)\, ds \\ & = & -\dfrac{1}{\varepsilon}\displaystyle\int_0^\varepsilon v\left(\frac{s}{\varepsilon}\right) f(x - sn(x)) \left(\int_0^s \nabla g(x - tn(x)).n(x) dt\right) ds \\ & & -\dfrac{1}{\varepsilon}\displaystyle\int_0^\varepsilon v\left(\frac{s}{\varepsilon}\right) \left(\int_0^s \nabla f(x - tn(x)).n(x) dt\right) g(x) ds \end{array},$$

whence,

$$\begin{array}{rcl} ||J_\varepsilon||_{L^1(\partial\Omega)} & \leq & \dfrac{||v||_{L^\infty(0,1)}}{\varepsilon} \displaystyle\int_{\partial\Omega} \left(\int_0^\varepsilon |f(x - sn(x))|\, ds\right) \left(\int_0^\varepsilon |\nabla g(x - tn(x))| dt\right) dx \\ & & + \dfrac{||v||_{L^\infty(0,1)}}{\varepsilon} \displaystyle\int_{\partial\Omega} \int_0^\varepsilon \left(\int_0^\varepsilon |\ \nabla f(x - tn(x))| dt\right) |g(x)| ds\, dx \\ & \leq & \dfrac{||v||_{L^\infty(0,1)}}{2\varepsilon} \left(\displaystyle\int_{\partial\Omega} \left(\int_0^\varepsilon |f(x - sn(x))| ds\right)^2 dx + \int_{\partial\Omega} \left(\int_0^\varepsilon |\nabla g(x - sn(x))| ds\right)^2 dx\right) \\ & & + \dfrac{||v||_{L^\infty(0,1)}}{2\varepsilon} \left(\displaystyle\int_{\partial\Omega} \left(\int_0^\varepsilon |g(x)|\ ds\right)^2 dx + \int_{\partial\Omega} \left(\int_0^\varepsilon |\nabla f(x - sn(x))| ds\right)^2 dx\right) \\ & \leq & \dfrac{||v||_{L^\infty(0,1)}}{2} \left(\displaystyle\int_{\partial\Omega} \int_0^\varepsilon |f(x - sn(x))|^2 ds dx + \int_{\partial\Omega} \int_0^\varepsilon |\nabla f(x - sn(x))|^2 ds dx\right) \\ & & + \dfrac{||v||_{L^\infty(0,1)}}{2} \left(\displaystyle\int_{\partial\Omega} \int_0^\varepsilon |g(x)|^2\, ds\, dx + \int_{\partial\Omega} \int_0^\varepsilon |\nabla g(x - sn(x))|^2 ds dx\right) \end{array}.$$

Now, using back the coarea formula (see theorem 4.2) yields the existence of a constant $C > 0$, which only depends on $\partial\Omega$ (through its principal curvatures), v and ε_0 such that :

$$||J_\varepsilon||_{L^1(\partial\Omega)} \leq C \left(||f||^2_{H^1(\partial\Omega_\varepsilon)} + ||g||^2_{H^1(\partial\Omega_\varepsilon)}\right),$$

which is the expected estimate. □

4.8.2 Extension to the context of linearized elasticity

This section is devoted to extending the results of the previous section to the isotropic linearized elasticity setting. Let us first set some notations.

We are still interested in a situation like that depicted in figure 4.26. For $\varepsilon > 0$ sufficiently small, define $u \in H^1(D)^d$ as the solution to the *sharp-interface problem*:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(u)) & = & f & \text{in } D \\ u & = & 0 & \text{on } \Gamma_D \\ Ae(u)n & = & g & \text{on } \Gamma_N \end{array} \right. . \tag{4.74}$$

Here, A is a discontinuous Hooke's tensors across the interface $\Gamma = \partial\Omega^0 : A = A_0\chi_0 + A_1\chi_1$, where A_0, A_1 are isotropic Hooke's tensors, with respective Lamé coefficients λ_0, μ_0 and λ_1, μ_1, i.e.

$$\forall \xi \in \mathcal{S}^d(\mathbb{R}),\ \ A_i\xi = 2\mu_i\xi + \lambda_i \mathrm{tr}(\xi)I.$$

Once again, as an approximation to (4.74), we consider the *smoothed-interface problem* with parameter $\varepsilon > 0$, bringing into play the displacement $u_\varepsilon \in H^1(D)^d$ as the solution to

$$\left\{ \begin{array}{rcll} -\mathrm{div}(A_\varepsilon e(u_\varepsilon)) & = & f & \text{in } D \\ u_\varepsilon & = & 0 & \text{on } \Gamma_D \\ A_\varepsilon e(u_\varepsilon)n & = & g & \text{on } \Gamma_N \end{array} \right. , \tag{4.75}$$

with the following expression for A_ε:

$$A_\varepsilon(x) = A_0 + h_\varepsilon(d_{\Omega^0}(x))(A_1 - A_0). \tag{4.76}$$

In the smoothed-interface setting, the compliance $J_\varepsilon(\Omega^0)$ of the total structure D, subdivided as $D = \Omega^0 \cup \Gamma \cup \Omega^1$ reads

$$J_\varepsilon(\Omega^0) = \int_D f \cdot u_\varepsilon \, dx + \int_{\Gamma_N} g \cdot u_\varepsilon \, ds,$$

and we have proved with theorem 4.8 that, for any $\varepsilon > 0$ small enough, the shape derivative of J_ε reads:

$$\forall \theta \in W^{1,\infty}(\mathbb{R}^2, \mathbb{R}^2),\ \ J_\varepsilon\prime(\Omega^0)(\theta) = \int_\Gamma \left[f_0^\varepsilon(x) + f_1^\varepsilon(x)\right](\theta(x).n(x))\ ds(x), \tag{4.77}$$

where $\forall x \in \Gamma$, we have, denoting $\kappa_1(x), ..., \kappa_{d-1}(x)$ the principal curvatures of Γ at x:

$$f_0^\varepsilon(x) = \int_{p_\Gamma^{-1}(x)\cap\Omega_0} \prod_{i=1}^{d-1} (1 + d_{\Omega^0}(y)\kappa_i(x))\, h_\varepsilon'\left(d_{\Omega^0}(y)\right)(A_1 - A_0)e(u_\varepsilon)(y) : e(u_\varepsilon(y))\, dy, \tag{4.78}$$

$$f_1^\varepsilon(x) = \int_{p_\Gamma^{-1}(x)\cap\Omega_1} \prod_{i=1}^{d-1} (1 + d_{\Omega^0}(y)\kappa_i(x))\, h_\varepsilon'\left(d_{\Omega^0}(y)\right)(A_1 - A_0)e(u_\varepsilon)(y) : e(u_\varepsilon(y))\, dy. \tag{4.79}$$

On the other hand, as mentioned in section 4.3, the shape derivative of the compliance $J(\Omega^0)$ associated to the sharp-interface problem reads as:

$$\forall \theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ \ J'(\Omega^0)(\theta) = \int_\Gamma \mathcal{D}(u,u)(\theta.n)\, ds, \tag{4.80}$$

where $\mathcal{D}(u,u)$ is defined by formula (4.10), or equivalently by the following formula which will turn out more convenient in our context (see [15]):

$$\begin{array}{rcl} \forall x \in \Gamma,\ \ \mathcal{D}(u,u)(x) & = & 2\left[\mu\right] e(u(x))_{\tau,\tau} : e(u(x))_{\tau,\tau} - \left[\frac{1}{\mu}\right] \sigma(u(x))_{n,\tau} \cdot \sigma(u(x))_{n,\tau} \\ & & + \left[\frac{2\lambda\mu}{(2\mu+\lambda)}\right] \mathrm{tr}(e(u(x))_{\tau,\tau})^2 - \left[\frac{1}{2\mu+\lambda}\right] \sigma(u(x))_{n,n}^2 \\ & & + \left[\frac{2\lambda}{2\mu+\lambda}\right] \sigma(u(x))_{n,n}\, \mathrm{tr}(e(u(x))_{\tau,\tau}) \end{array} , \tag{4.81}$$

where $[\alpha] = \alpha^1 - \alpha^0$ still denotes the jump of a discontinuous quantity α across the interface Γ.

Our purpose is to show that formula (4.77) for the shape derivative of the compliance in the smoothed-interface context converges towards its equivalent (4.80) in the sharp-interface context in the limit $\varepsilon \to 0$.

The proof of this fact unfolds almost exactly as its counterpart in the scalar conductivity setting: using information about the derivatives of u_ε which behave 'well' in the neighborhood as $\varepsilon \to 0$, expressions (4.78,4.79) are decomposed so that they feature u_ε only through these derivatives. Eventually, the convergence result of Proposition 4.14 is used. According to this sketch, we have to start with the following proposition, whose proof arises exactly as that of Proposition 4.13 (with the tedious algebra of linear elasticity, see for instance the convergence results in [222]):

Proposition 4.15. *Assuming Γ is a $\mathcal{C}^2$ interface, there exists a tubular neighborhood $V \subset\subset D$ of Γ such that one can define a smooth extension in V of the normal n and of a set of tangentials and orthonormal vectors τ. Then, the following strong convergences hold true*

$$\begin{array}{lcll} e(u_\varepsilon)_{\tau\tau} & \overset{\varepsilon\to 0}{\longrightarrow} & e(u)_{\tau\tau} & \text{in } H^1(V)^{(d-1)^2} \text{ strong,} \\ \sigma(u_\varepsilon)_{\tau n} & \overset{\varepsilon\to 0}{\longrightarrow} & \sigma(u)_{\tau n} & \text{in } H^1(V)^d \text{ strong,} \\ \sigma(u_\varepsilon)_{nn} & \overset{\varepsilon\to 0}{\longrightarrow} & \sigma(u)_{nn} & \text{in } H^1(V) \text{ strong.} \end{array} .$$

We can now state and prove the result of interest:

Theorem 4.12. *Under the above assumptions, we have:*

$$\lim_{\varepsilon\to 0} J'_\varepsilon(\Omega^0)(\theta) = J'(\Omega^0)(\theta) \quad \forall\, \theta \in W^{1,\infty}(D, \mathbb{R}^d).$$

Proof. Here again, assume that ε is so small that $\varepsilon < \text{reach}(\Gamma)$. We are interested in the asymptotic behavior of f_1^ε as $\varepsilon \to 0$ (the case of f_0^ε being identical). One can write:

$$\forall x \in \Gamma,\ \ f_1^\varepsilon(x) = \int_0^\varepsilon \prod_{i=1}^{d-1} (1 + s\kappa_i(x))\, h'_\varepsilon(s)(A_1 - A_0)e(u_\varepsilon(x + sn(x))) : e(u_\varepsilon(x + sn(x)))\, ds.$$

A straightforward change of variables provides:

$$\forall x \in \Gamma,\ \ f_1^\varepsilon(x) = \int_0^1 k_\varepsilon(x, s)\, (\varepsilon h'_\varepsilon(s\varepsilon)(A_1 - A_0)e(u_\varepsilon(x + s\varepsilon n(x)))) : e(u_\varepsilon(x + s\varepsilon n(x)))\, ds. \tag{4.82}$$

For any $s \in (0, 1)$, and any symmetric matrix $\xi \in \mathcal{S}^d(\mathbb{R})$, note that:

$$\varepsilon h'_\varepsilon(s\varepsilon)(A_1 - A_0)\xi = 2\mu'(s)\xi + \lambda'(s)\text{tr}(\xi) =: M'(s)\xi,$$

where we have introduced: $M(s) = A_0 + h_\varepsilon(s\varepsilon)(A_1 - A_0)$ the isotropic linear elasticity tensor with Lamé coefficients

$$\left\{ \begin{array}{rcccl} \lambda(s) & = & \lambda_0 + h_\varepsilon(s\varepsilon)(\lambda_1 - \lambda_0) & = & \lambda_0 + \frac{1}{2}\left(1 + \frac{y}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi y}{\epsilon})\right)(\lambda_1 - \lambda_0) \\ \mu(s) & = & \mu_0 + h_\varepsilon(s\varepsilon)(\mu_1 - \mu_0) & = & \mu_0 + \frac{1}{2}\left(1 + \frac{y}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi y}{\epsilon})\right)(\mu_1 - \mu_0) \end{array} \right.$$

Let us now simplify a bit notations. In the forthcoming (algebraic) computations, let $x_\varepsilon(s) = x + s\varepsilon n(x)$, $e^\varepsilon(s) = e(u_\varepsilon(x_\varepsilon(s))$ and $\sigma^\varepsilon(s) = \sigma(u_\varepsilon(x_\varepsilon(s))$. We also denote $k_\varepsilon(s)$ instead of $k_\varepsilon(x, s)$. What's more, for any $x \in \Gamma$, introduce an orthonormal basis $\{\tau_i\}_{i=1,\dots,d-1}$ of vectors of the tangent plane to Γ, collectively denoted as τ, and let n the unit normal vector to Γ, pointing outward Ω^0, in such a way that (τ, n) is an orthonormal basis of $\mathbb{R}^d$.

We have to modify expression (4.82), so that it involves only those quantities that enjoy a 'nice' behavior near Γ as $\varepsilon \to 0$, as expressed by Proposition 4.14. To this end, the following relationship will come in handy:

$$\left\{\begin{array}{rcl} \sigma^\varepsilon(s)_{n,n} &=& 2\mu(s)e^\varepsilon(s)_{n,n} + \lambda(s)\left(\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s)) + e^\varepsilon(s)_{n,n}\right) \\ \sigma^\varepsilon(s)_{n,\tau_i} &=& 2\mu(s)e^\varepsilon(s)_{n,\tau_i},\ \forall i = 1,...,d-1 \\ \sigma^\varepsilon(s)_{\tau_i,\tau_j} &=& 2\mu(s)e^\varepsilon(s)_{\tau_i,\tau_j} + \lambda(s)\left(\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s)) + e^\varepsilon(s)_{n,n}\right)\delta_{i,j}\ \forall i,j = 1,...,d-1 \end{array}\right. , \tag{4.83}$$

where $\delta_{i,j}$ stands for the Kronecker symbol, and we have denoted $\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s)) = \sum_{i=1}^{d-1} e^\varepsilon(s)_{\tau_i,\tau_i}$. These relations can be inverted to produce the following formulae, where the right-hand sides only depend on quantities whose passing to the limit $\varepsilon \to 0$ 'unfolds well' near Γ:

$$\left\{\begin{array}{rcl} e^\varepsilon(s)_{n,n} &=& \frac{1}{2\mu(s)+\lambda(s)}\left(\sigma^\varepsilon(s)_{n,n} - \lambda(s)\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\right) \\ e^\varepsilon(s)_{n,\tau_i} &=& \frac{1}{2\mu(s)}\sigma^\varepsilon(s)_{n,\tau} \\ \sigma^\varepsilon(s)_{\tau_i,\tau_j} &=& 2\mu(s)e^\varepsilon(s)_{\tau_i,\tau_j} + \left(\frac{2\mu(s)\lambda(s)}{2\mu(s)+\lambda(s)}\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s)) + \frac{\lambda(s)}{2\mu(s)+\lambda(s)}\sigma^\varepsilon(s)_{n,n}\right)\delta_{i,j} \end{array}\right. . \tag{4.84}$$

Now, expression (4.82) for $f_1^\varepsilon(x)$ rewrites, in condensed form:

$$f_1^\varepsilon(x) = \int_0^1 k^\varepsilon(s)M'(s)e^\varepsilon(s) : e^\varepsilon(s)\, ds.$$

Expanding the symmetric tensor $M'(s)e^\varepsilon(s)$ yields:

$$M'(s)e^\varepsilon(s) = 2\mu'(s)e^\varepsilon(s) + \lambda'(s)\mathrm{tr}(e^\varepsilon(s))I,$$

whence:

$$\begin{array}{rcl} M'(s)e^\varepsilon(s)e^\varepsilon(s) &=& 2\mu'(s)e^\varepsilon(s) : e^\varepsilon(s) + \lambda'(s)\mathrm{tr}(e^\varepsilon(s))\mathrm{tr}(e^\varepsilon(s)), \\ &=& 2\mu'(s)\left(e^\varepsilon(s)_{\tau,\tau} : e^\varepsilon(s)_{\tau,\tau} + 2e^\varepsilon(s)_{n,\tau}\cdot e^\varepsilon(s)_{n,\tau} + e^\varepsilon(s)_{n,n}\, e^\varepsilon(s)_{n,n}\right) \\ && +\lambda'(s)\left(\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s)) + e^\varepsilon(s)_{n,n}\right)^2 \\ &=& 2\mu'(s)e^\varepsilon(s)_{\tau,\tau} : e^\varepsilon(s)_{\tau,\tau} + \frac{4\mu'(s)}{4\mu(s)^2}\sigma^\varepsilon(s)_{n,\tau}\cdot\sigma^\varepsilon(s)_{n,\tau} \\ && +\frac{2\mu'(s)}{(2\mu(s)+\lambda(s))^2}\left(\sigma^\varepsilon(s)_{n,n} - \lambda(s)\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\right)^2 \\ && +\lambda'(s)\left(\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s)) + \frac{1}{2\mu(s)+\lambda(s)}\left(\sigma^\varepsilon(s)_{n,n} - \lambda(s)\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\right)\right)^2 \end{array}$$

where we have been using relations (4.84). Now, rearranging things a bit, we obtain:

$$\begin{array}{rcl} M'(s)e^\varepsilon(s)e^\varepsilon(s) &=& 2\mu'(s)e^\varepsilon(s)_{\tau,\tau} : e^\varepsilon(s)_{\tau,\tau} + \frac{\mu'(s)}{\mu(s)^2}\sigma^\varepsilon(s)_{n,\tau}\cdot\sigma^\varepsilon(s)_{n,\tau} \\ && +\frac{2\mu'(s)+\lambda'(s)}{(2\mu(s)+\lambda(s))^2}\left(\sigma^\varepsilon(s)_{n,n} - \lambda(s)\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\right)^2 + \lambda'(s)\left(\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\right)^2 , \\ && +\frac{2\lambda'(s)}{2\mu(s)+\lambda(s)}\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\left(\sigma^\varepsilon(s)_{n,n} - \lambda(s)\mathrm{tr}(e^\varepsilon_{\tau,\tau}(s))\right) \end{array}$$

and, after some computation:

$$\begin{array}{rcl} M'(s)e^\varepsilon(s)e^\varepsilon(s) &=& 2\mu'(s)e^\varepsilon(s)_{\tau,\tau} : e^\varepsilon(s)_{\tau,\tau} + \frac{\mu'(s)}{\mu(s)^2}\sigma^\varepsilon(s)_{n,\tau}\cdot\sigma^\varepsilon(s)_{n,\tau} \\ && +\frac{4\mu^2(s)\lambda'(s)+2\mu'(s)\lambda^2(s)}{(2\mu(s)+\lambda(s))^2}\mathrm{tr}(e^\varepsilon_{\tau,\tau})^2 + \frac{2\mu'(s)+\lambda'(s)}{(2\mu(s)+\lambda(s))^2}\sigma^\varepsilon(s)^2_{n,n} , \\ && +4\frac{\mu(s)\lambda'(s)-\mu'(s)\lambda(s)}{(2\mu(s)+\lambda(s))^2}\sigma^\varepsilon(s)_{n,n}\,\mathrm{tr}(e^\varepsilon_{\tau,\tau}) \end{array}$$

We are now in position to apply proposition 4.14, noticing that:

$$\left\{\begin{array}{rcl} 2\mu'(s) &=& (2\mu(s))' \\ \frac{\mu'(s)}{\mu(s)^2} &=& \left(-\frac{1}{\mu(s)}\right)' \\ \frac{4\mu^2(s)\lambda'(s)+2\mu'(s)\lambda^2(s)}{(2\mu(s)+\lambda(s))^2} &=& \left(\frac{2\lambda(s)\mu(s)}{(2\mu(s)+\lambda(s))}\right)' \\ \frac{2\mu'(s)+\lambda'(s)}{(2\mu(s)+\lambda(s))^2} &=& \left(-\frac{1}{2\mu(s)+\lambda(s)}\right)' \\ 4\frac{\mu(s)\lambda'(s)-\mu'(s)\lambda(s)}{(2\mu(s)+\lambda(s))^2} &=& \left(\frac{2\lambda(s)}{2\mu(s)+\lambda(s)}\right)' \end{array}\right.$$

We then have that $f_1^\varepsilon \to f_1$ in $L^1(\Gamma)$, where f^1 is defined as:

$$\begin{aligned}\forall x \in \Gamma,\ \ f_1(x) \quad = \quad & [2\mu(s)]_0^1\, e(u(x))_{\tau,\tau} : e(u(x))_{\tau,\tau} - \left[\tfrac{1}{\mu(s)}\right]_0^1 \sigma(u(x))_{n,\tau} \cdot \sigma(u(x))_{n,\tau} \\ & + \left[\tfrac{2\lambda(s)\mu(s)}{(2\mu(s)+\lambda(s))}\right]_0^1 \mathrm{tr}(e(u(x))_{\tau,\tau})^2 - \left[\tfrac{1}{2\mu(s)+\lambda(s)}\right]_0^1 \sigma(u(x))_{n,n}^2 \\ & + \left[\tfrac{2\lambda(s)}{2\mu(s)+\lambda(s)}\right]_0^1 \sigma(u(x))_{n,n}\, \mathrm{tr}(e(u(x))_{\tau,\tau})\end{aligned} \quad .$$

Doing the exact same job for the convergence of f_0^ε as $\varepsilon \to 0$ eventually shows that the limit of the integrand $[f_0^\varepsilon + f_1^\varepsilon]$ in formula (4.64) reads, for all $x \in \Gamma$:

$$\begin{aligned}f_0(x) + f_1(x) \quad = \quad & 2\,[\mu]\, e(u(x))_{\tau,\tau} : e(u(x))_{\tau,\tau} - \left[\tfrac{1}{\mu}\right] \sigma(u(x))_{n,\tau} \cdot \sigma(u(x))_{n,\tau} \\ & + \left[\tfrac{2\lambda\mu}{(2\mu+\lambda)}\right] \mathrm{tr}(e(u(x))_{\tau,\tau})^2 - \left[\tfrac{1}{2\mu+\lambda}\right] \sigma(u(x))_{n,n}^2 \\ & + \left[\tfrac{2\lambda}{2\mu+\lambda}\right] \sigma(u(x))_{n,n}\, \mathrm{tr}(e(u(x))_{\tau,\tau})\end{aligned} \quad ,$$

which is exactly formula (4.81) for the integrand in the formula for the shape derivative of the sharp-interface problem. □

Chapter 5

A linearized approach to worst-case design in parametric and geometric shape optimization

Contents

5.1 Introduction 164
5.2 General setting and main notations 165
5.3 Worst-case design in parametric optimization 167
5.3.1 Description of the model problem 167
5.3.2 Worst-case design of an elastic plate under perturbations on the body forces 168
5.3.3 Extension to a worst-case optimization problem, with respect to a perturbation on surface loads 176
5.3.4 Parametric optimization of a worst-case scenario problem under geometric uncertainty 177
5.3.5 Worst-case design with uncertainties over the elastic material's properties 181
5.4 Worst-case design in shape optimization 183
5.4.1 Description of the model problem 183
5.4.2 Worst-case design in shape optimization under uncertainties over the applied body forces 184
5.4.3 Worst-case design in shape optimization under uncertainties on the Lamé moduli of the material 188
5.4.4 Worst-case design in shape optimization under geometric uncertainties 190
5.5 Numerical results 198
5.5.1 Worst-case optimization problems in parametric structural optimization 198
5.5.1.1 Uncertainties around the applied body forces in parametric optimization 198
5.5.1.2 Geometric uncertainties in parametric optimization 199
5.5.2 Examples of shape optimization problems under uncertainties 200
5.5.2.1 Details around the numerical implementation 200
5.5.2.2 Shape optimization under uncertainties about the applied loads 203
5.5.2.3 Shape optimization under uncertainties on the material's properties . . . 208
5.5.2.4 Shape optimization under geometric uncertainties 211
5.5.3 General tools 218
5.5.4 Several Green's formulae 219

The purpose of this chapter is to propose a deterministic method for optimizing a structure when a 'small' uncertainty exists over some of its features, with respect to its worst possible behavior. The main idea of the method is to linearize all the dependences of the considered cost function with respect to the uncertain parameters, then to consider the supremum function of the obtained linear approximation, which can be rewritten as a more 'classical' function of the design, owing to classical adjoint techniques from optimal control theory. This formal approach can be legitimated in some particular cases, and is very general. In particular, it allows to address several problems of considerable importance in both parametric and shape optimization of elastic structures, in a unified framework.

This chapter is a joint work with Grégoire Allaire.

5.1 Introduction

As idealized visions of reality, most optimization frameworks assume a complete knowledge of the parameters of the experiments they represent. Unfortunately, for a lot of reasons, physical parameters involved in realistic applications are hardly ever known with such exactness, and the feasibility and optimality of the solution to an optimization problem can be tremendously jeopardized by slight variations in the attached parameters. In this spirit, an example of the less compliant microstructure for an elastic composite material submitted to a particular set of traction loads is given in [83], which is infinitely compliant when submitted to any load with a different orientation (see also [42] for a discussion about the sensitivity of linear programming problems with uncertain parameters).

In the more specific context of shape optimization of elastic structures, which is at stake in this paper, optimization problems may be plagued by (at least) three very different types of uncertainties:

- *Uncertainties about the location, magnitude and orientation of the body forces or surface loads exerted on shapes:* not to mention the fact that they are generally known through error-prone measurements, these external stresses are affected by the outer medium, which may itself undergo unknown perturbations.
- *Uncertainties about the elastic material's properties:* changes in the conditions in the considered medium (temperature, humidity, etc...) may alter the material's stiffness. On a different note, the material's properties could also be perturbed during its manufacturing process, in which small inclusions of 'parasitic' bodies may accidentally occur.
- *Uncertainties about the geometry of the shape itself:* due to wear, or in the course of the manufacturing process, the geometry of the shape may cease to be (or may not be from the beginning) completely in keeping with the initially forecast design.

Robust design has been paid much attention in shape and topology optimization (and in optimization in general). Depending on the available information regarding the uncertainties, the question has been studied from two distinct perspectives.

On the one hand, many authors assume the knowledge of a probabilistic distribution (which is often obtained via statistical studies) as for the behavior of perturbations around an unperturbed state; see for instance [137, 190] and references therein. Then, the mean value of the considered objective criterion [95, 186], or a weighted sum of its mean value and standard deviation are minimized to guarantee a fine performance, which is relatively independent of the perturbations. Likewise, *reliability-based* approaches (see the overview in [88]) put the emphasis on guaranteeing that constraints stay satisfied even in the perturbed configurations, and add for instance upper bound constraints on failure probabilities. Regardless of the particular considered model, such probabilistic methods generally prove very costly, since they imply repeated evaluations of the mean value or standard deviation of the objective function, or probabilities of violation of constraint functions. This is generally achieved by (expensive) sampling methods (e.g. Monte-Carlo methods).

On the other hand, when no information is available on the expected perturbations but for bounds on their magnitude, so-called *worst-case designs* approaches are preferred. The problem can then be rephrased either as that of minimizing the worst value of the objective function among all the perturbed designs [163],

or guaranteeing that constraints are fulfilled by every such design [165] (the latter approach being called *confidence* optimization). Such problems are generally formulated as *bilevel* 'min-max' problems [166]: a first problem consists in finding the worst-case perturbation for a given design, then a second one is about finding the optimal design with respect to this supremum criterion. Of course, worst-case design problems are very difficult and computationally expensive in utter generality (see for instance [263] for a reduced-basis method adapted to a worst-case design problem). They cannot be solved without resorting to approximations, except in a few very specific cases; see the very interesting works [84, 163] about finding the less compliant shape under uncertainties on the body forces. On the theoretical side, worst-case functionals have recently been studied, for instance in [174], where conditions are given for such min-max problems to admit at least one solution. In [178], it is shown that the concept of topological derivative is robust when the linear elasticity system at play undergoes 'small' perturbations.

The aim of this chapter is to propose a unified framework for the worst-case design of elastic structures, with respect to 'small' perturbations on the applied body forces and surface loads, on the material's properties and on the geometries of the structures. More precisely, a formal and rather inexpensive approach is presented for the minimization of the maximum value of a given criterion under the assumption of 'small amplitude' perturbations. Note that we do not attempt to tackle the so-called 'confidence worst-case design' approach, which would ask constraints to be satisfied for all the perturbed designs, but claim that the same philosophy would allow to deal with them. The starting ingredient of our approach is to take advantage of the smallness of the perturbations and thus to linearize the cost function with respect to perturbations around the unperturbed configuration. The maximum value of the resulting linear functional among all possible perturbations can be explicitly computed, and is then optimized. This idea is quite natural and it has already been used in some specific cases (for example, [165] considers the case of compliance minimization under geometric uncertainties). Our approach is however systematic and very general. It is formal, for there is absolutely no guarantee that the supremum of the linearized cost function should be close to the real worst value of the criterion. Yet, we shall see that, in some cases, it can be legitimated and admits a physical interpretation. Besides, in general, it should provide valuable help in discerning 'trends' towards robustness with respect to perturbations of various kinds.

This chapter is organized as follows: Section 5.2 opens the discussion with a presentation of the general philosophy of the proposed method in a formal, abstract framework. This method is then carried out in Section 5.3, in the simpler situation of (parametric) optimization of the thickness of an elastic plate, which already features almost all the salient features of the approach; then, it is used in shape optimization in Section 5.4. Then, some numerical examples and discussions are proposed in Section 5.5 to appraise the efficiency of the proposed method, and some technical details are supplied concerning the proposed implementation.

5.2 General setting and main notations

This very informal section presents the generic worst-case optimization problem addressed in this chapter, and exposes the main ideas of the proposed approach to deal with it. In the meantime, some notations are introduced, which are used throughout this chapter.

Let $\mathcal{H}$ be a set of admissible designs among which the 'best' element is sought, with respect to a prescribed criterion or *cost function* $\mathcal{C}(u(h))$, depending on $h \in \mathcal{H}$ via the solution (or *state*) $u(h)$ to a system where h acts as a parameter, say:

$$\mathcal{A}(h)u(h) = b. \tag{5.1}$$

In the applications ahead, $\mathcal{H}$ will stand for either the space $L^\infty(\Omega)$ of thickness functions for an elastic plate with fixed cross-section Ω (Section 5.3), or a space of linear elastic structures (section 5.4).

The system (5.1) may undergo perturbations that affect the state $u(h)$, thus spoiling the performance of the corresponding design $h \in \mathcal{H}$. The set $\mathcal{P}$ of such perturbations is assumed to be a Banach space, with norm $||.||_{\mathcal{P}}$, and we only assume that the expected perturbations have 'small' maximum norm $m > 0$.

To keep things simple, let us assume that these perturbations only affect the right-hand side of (5.1), i.e. $b = b(\delta)$, $\delta \in \mathcal{P}$. As we shall see, this is the case when the optimal thickness (resp. shape) of an elastic plate (resp. structure) is sought under uncertainties over the applied external body forces or traction loads. The state $u = u(h, \delta)$ is now solution to:

$$\mathcal{A}(h)u(h,\delta) = b(\delta). \tag{5.2}$$

By convention, the case $\delta = 0$ accounts for the unperturbed situation, and we shall indifferently denote by $u(h, 0)$ or $u(h)$ the corresponding state when the context is clear. The associated *worst-case optimization problem* consists in minimizing the functional $\mathcal{J} : \mathcal{H} \to \mathbb{R}$, defined as the maximum value reached by the cost function $\mathcal{C}(u(h, \delta))$ for all the potential perturbations $\delta \in \mathcal{P}$, i.e.

$$\forall h \in \mathcal{H},\ \ \mathcal{J}(h) = \sup_{\substack{\delta\in\mathcal{P} \\ ||\delta||_{\mathcal{P}} \leq m}} \mathcal{C}(u(h,\delta)).$$

Taking advantage of the assumption that the amplitude m of the exerted perturbations is small, and since we claimed that this problem is difficult to solve as such in general, we propose to trade it for that of minimizing another functional $\widetilde{\mathcal{J}}$, obtained from $\mathcal{J}$ by linearizing the dependence of $\mathcal{C}(u(h, \delta))$ on δ before evaluating the supremum:

$$\forall h \in \mathcal{H},\ \ \widetilde{\mathcal{J}}(h) = \sup_{\substack{\delta\in\mathcal{P} \\ ||\delta||_{\mathcal{P}} \leq m}} \left(\mathcal{C}(u(h)) + \frac{d\mathcal{C}}{du}(u(h))\frac{\partial u}{\partial \delta}(h,0)(\delta)\right), \tag{5.3}$$

where $\frac{d\mathcal{C}}{du}$ stands for the (total) differential of $\mathcal{C}$ with respect to u, and $\frac{\partial u}{\partial \delta}$ is the (partial) differential of u with respect to δ. Note that the application $\delta \mapsto \left(\mathcal{C}(u(h)) + \frac{d\mathcal{C}}{du}(u(h))\frac{\partial u}{\partial \delta}(h,0)(\delta)\right)$ is linear by construction, since it only involves differentials of functions. Now, the supremum in formula (5.3) can be computed explicitly. Indeed, assuming that $\mathcal{P}$ arises as the dual of another Banach space, say $\mathcal{Q}$, it comes, using lemma 5.6:

$$\forall h \in \mathcal{H},\ \ \widetilde{\mathcal{J}}(h) = \mathcal{C}(u(h)) + m\left|\left|\frac{d\mathcal{C}}{du}(u(h))\frac{\partial u}{\partial \delta}(h,0)\right|\right|_{\mathcal{Q}}.$$

The resulting expression is still not explicit in terms of h, for it involves the sensitivity $\frac{\partial u}{\partial \delta}(h, 0)$ of the solution to the state equation with respect to perturbations. However, classical techniques in optimal control theory allow to make it explicit, up to the introduction of an *adjoint state* $p(h)$, which arises as the solution to an *adjoint system* very similar to (5.1):

$$\mathcal{A}(h)p(h) = \mathcal{C}'(u(h)),$$

where the notation $\mathcal{C}'$ stands for the identification of the differential application $\frac{d\mathcal{C}}{du}$ with respect to some dual pairing between Banach spaces (here, we implicitly assumed that $\mathcal{A}(h)$ is a linear, self-adjoint operator). Anyhow, $\widetilde{\mathcal{J}}$ rewrites as:

$$\widetilde{\mathcal{J}}(h) = \mathcal{C}(u(h)) + m||p(h)||_{\mathcal{Q}}. \tag{5.4}$$

This last expression can be interpreted as follows: the approximate cost function $\widetilde{\mathcal{J}}(h)$ is an aggregated sum of the unperturbed cost function $\mathcal{C}(u(h))$ and a penalization of the perturbations $||p(h)||_{\mathcal{Q}}$, the penalization parameter m being precisely the expected magnitude of perturbations.

Of course, the above argument is very formal, since the involved expressions mix blithely the spaces associated to perturbations, state variables, etc... As we shall see, a significant part of the work consists in giving a precise meaning to this rough sketch.

Remark 5.1. It is no surprise that formula (5.4) for functional $\widetilde{\mathcal{J}}$ features the adjoint state $p(h)$; it is indeed well-known that the adjoint measures the sensitivity of the cost function $\mathcal{C}$ with respect to perturbations on the state equation (5.1) (see e.g. [9], rem. 4.14 and 5.21).

The previous analysis leaves us with a more classical state-constrained optimization problem, save that the functional to be minimized itself brings an adjoint state into play. The computation of the derivative of $\widetilde{\mathcal{J}}$ can however be carried out as follows:

- Derivating the part $\mathcal{C}(u(h))$ in (5.4) does not pose any further difficulty: it is merely the unperturbed cost function of the considered problem. It involves the already introduced adjoint state $p(h)$, which accounts for the sensitivity of $\mathcal{C}$ with respect to its argument.
- Derivating the second part $||p(h)||_{\mathcal{Q}}$ is a little bit more tricky, and brings into play two further adjoint states $q(h)$ and $z(h)$. The first one $q(h)$ has nothing to do with $p(h)$ and expresses the sensitivity of the new 'objective function' $||.||_{\mathcal{Q}}$ with respect to its argument. As we shall see, even when the unperturbed optimization problem is self-adjoint (i.e. $p(h) = \pm u(h)$), $q(h)$ differs from $\pm u(h)$. The second one $z(h)$ describes the sensitivity of $p(h)$ - that is, in some way, of $\mathcal{C}'$ - and typically involves second order derivatives of $\mathcal{C}$ with respect to its argument. When the unperturbed problem is self-adjoint, $z(h)$ happens to be equal to $\pm q(h)$.

Remark 5.2. For simplicity of the exposition, we did not evoke the possibility that the problem of minimizing $\mathcal{J}$ (or $\widetilde{\mathcal{J}}$) may be constrained. Actually, in the remainder of this chapter, we will only be considering constraints on the volumes of structures, which is especially easy to enforce in our context (see Section 5.5). In this regard, one could wonder over the possibility to tackle worst-case design problems with *confidence* constraints, i.e. problems of the form:

$$\min_{h\in\mathcal{H}} \max_{\delta\in\mathcal{P}} \mathcal{C}(u(h,\delta)) \text{ s.t. } \max_{\delta\in\mathcal{P}} c_i(u(h,\delta)) \leq 0, \ \forall i = 1, ..., N,$$

where c_i are scalar-valued functions of the state u.

Note that, all things considered, the proposed method is nothing but a formal and approximate way to differentiate supremal functionals of the form $\mathcal{H} \ni h \mapsto \max_{\delta\in\mathcal{P}} \mathcal{C}(u(h,\delta))$. Hence, since most algorithms for constrained optimization rely at some point on a linearization of the cost and constraint functions (the Method of Moving Asymptotes [300], or the Method of Feasible Directions [312] to name a few), we believe that the methods and computations of this chapter can be used in this context.

Notations. Throughout this chapter, we consistently denote as $\mathcal{C}$ the various cost functions under consideration. Note that, contrary to the basic setting presented above, such cost functions may themselves depend on the perturbations (which is not subsequently problematic). We will denote as $\mathcal{J}$ the associated worst-case design functional, and as $\widetilde{\mathcal{J}}$ the approximate worst-case design functional, obtained by the aforementioned linearization of the cost function.

5.3 Worst-case design in parametric optimization

5.3.1 Description of the model problem

Throughout Section 5.3, we consider a thin linear elastic plate in plane stress situation, with Lipschitz cross-section $\Omega \subset \mathbb{R}^d$ ($d = 2$ in concrete applications) and positive thickness $h \in L^\infty(\Omega)$ (see figure 5.1).

From the plane stress assumption, the equilibrium equation of the plate can be written as a d-dimensional system posed on Ω. More specifically, assume the plate is clamped on a part of its boundary, associated to the subset Γ_D of $\partial\Omega$. Surface loads are applied on the complementary part of the plate, which are equivalent to transversal loads $g \in L^2(\Gamma_N)^d$, where $\Gamma_N := \partial\Omega \setminus \Gamma_D$. Denote also as $f \in L^2(\Omega)^d$ the d-dimensional equivalent body force to the whole body force exerted on the plate.

The transversal displacement function u belongs to the space $H^1_{\Gamma_D}(\Omega)^d$, where we have defined:

$$H^1(\Omega) \supset H^1_{\Gamma_D}(\Omega) := \left\{ u \in H^1(\Omega), u = 0 \text{ on } \Gamma_D \right\}.$$

It arises as the unique solution to the d-dimensional linear elasticity system on Ω:

$$\left\{ \begin{array}{rcll} -\text{div}(hAe(u)) & = & f & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ hAe(u)n & = & g & \text{on } \Gamma_N \end{array} \right. , \tag{5.5}$$

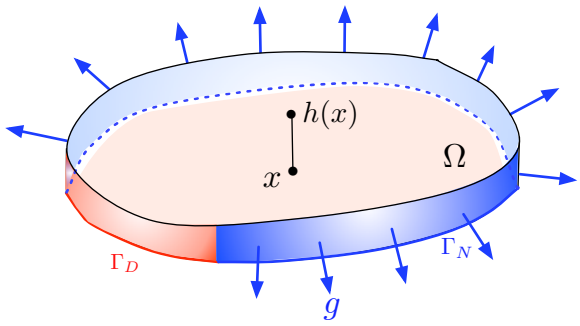


Figure 5.1: Thin plate with cross section Ω and thickness h, clamped on a part of its boundary corresponding to $\Gamma_D \subset \partial\Omega$, and submitted to transversal loads, applied on a part associated to $\Gamma_N \subset \partial\Omega$.

where $e(u) := \frac{\nabla u^T + \nabla u}{2}$ is the strain tensor, A is the material Hooke's law, with Lamé coefficients λ, μ, i.e.

$$\forall e \in \mathcal{S}(\mathbb{R}^d),\ Ae = 2\mu e + \lambda \mathrm{tr}(e) I,$$

where $\mathcal{S}(\mathbb{R}^d)$ is the set of $d \times d$ symmetric matrices with real coefficients, and $n : \partial\Omega \to \mathbb{S}^{d-1}$ is the unit outer normal vector field to Ω.

Our general purpose is to optimize the thickness h of the considered plate among a set $\mathcal{U}_{ad} \subset L^\infty(\Omega)$ of *admissible* thickness functions, with respect to a criterion yet to be specified . Throughout Section 5.3, we shall simply work with:

$$\mathcal{U}_{ad} = \{h \in L^\infty(\Omega),\ \text{s.t. a.e. } x \in \Omega,\ \ h_{min} \leq h(x) \leq h_{max}\}\,, \tag{5.6}$$

where $0 < h_{min} < h_{max}$ are prescribed lower and upper bounds for thickness functions.

Notations:

- If $J : \mathcal{U}_{ad} \to \mathbb{R}$ is any (Fréchet differentiable) functional of the thickness, $J'(h)$ stands for the Fréchet derivative of J with respect to h.
- In the following subsections, we shall be considering various (smooth enough) integrand functions j of the form: $j : \mathbb{R}^d_f \times \mathbb{R}^d_u \times \mathbb{R}^d_p \to \mathbb{R}$, where f stands for the perturbation variable, u stands for the solution to a 'state' linear elasticity system of the form (5.5), and p stands for the solution to an 'adjoint' linear elasticity system (see below). These functions may generally depend also on the space variable $x \in \mathbb{R}^d$, but, to keep notations as light as possible, this dependence will be systematically omitted (and does not change anything in any of the forthcoming formulae). The partial gradients of j with respect to the f, u, p variables are denoted respectively as: $\nabla_f j, \nabla_u j, \nabla_p j \in \mathbb{R}^d$.

The remainder of Section 5.3 is now dedicated to illustrating the general guideline of Section 5.2 on a series of model problems, which hopefully better embody the difficulties that may arise when carrying it out than the diversity of situations it may allow to tackle.

5.3.2 Worst-case design of an elastic plate under perturbations on the body forces

Let us start our study with the optimization of the thickness of the considered plate in the worst-case scenario when the applied body forces f are perturbed as $f + \xi$, for small $\xi \in L^2(\Omega)^d$.

Let $j : \mathbb{R}^d_f \times \mathbb{R}^d_u \to \mathbb{R}$ be a function of class $\mathcal{C}^2$, which complies with the following growth conditions:

$$\forall f \in \mathbb{R}^d,\ u \in \mathbb{R}^d,\ \left\{ \begin{array}{l} |j(f,u)| \leq C(|f|^2+|u|^2) \\ |\nabla_f j(f,u)| \leq C(|f|+|u|),\ |\nabla_u j(f,u)| \leq C(|f|+|u|) \\ |\nabla^2_f j(f,u)| \leq C,\ |\nabla_f \nabla_u j(f,u)| \leq C,\ |\nabla^2_u j(f,u)| \leq C \end{array} \right. , \tag{5.7}$$

for a large enough constant $C > 0$. Note that $\nabla_u j(f,u)$ and $\nabla_f j(f,u)$ are vectors in $\mathbb{R}^d$, and $\nabla^2_f j(f,u)$, $\nabla_f \nabla_u j(f,u) = \nabla^2_u j(f,u)$ are $d \times d$ matrices.

For any admissible thickness function $h \in \mathcal{U}_{ad}$, any body force term $f \in L^2(\Omega)^d$, and any traction loads $g \in L^2(\Gamma_N)^d$, denote as $u_{h,f} \in H^1_{\Gamma_D}(\Omega)^d$ the unique solution to problem (5.5) using these parameters.

The *cost* of such a plate is then defined as:

$$\mathcal{C}(h,f) = \int_\Omega j(f, u_{h,f})\, dx. \tag{5.8}$$

To set ideas, we have only assumed this cost to depend on $u_{h,f}$ - and not on its gradient-, as an integral expression on Ω - not on its boundary -, but it would be easy to generalize the discussion ahead to such cases (see sections 5.3.3 and 5.4.4 for instance).

Let us now fix $f \in L^2(\Omega)^d$ and $g \in L^2(\Gamma_N)^d$. As in Section 5.2, when no confusion is possible, we will denote indifferently $u_h = u_{h,f}$ the solution to the unperturbed problem. The worst-case design problem in our situation reads:

$$\min_{h \in \mathcal{U}_{ad}} \mathcal{J}(h),\ \text{where } \mathcal{J}(h) = \sup_{\substack{\xi \in L^2(\Omega)^d \\ ||\xi||_{L^2(\Omega)^d} \leq m}} \mathcal{C}(h, f+\xi). \tag{5.9}$$

Following the general guideline of Section 5.2, we propose to trade problem (5.9) for a new one where $\mathcal{C}(h,f)$ has been linearized with respect to f, namely:

$$\min_{h \in \mathcal{U}_{ad}} \widetilde{\mathcal{J}}(h),\ \text{where } \widetilde{\mathcal{J}}(h) = \sup_{\substack{\xi \in L^2(\Omega)^d \\ ||\xi||_{L^2(\Omega)^d} \leq m}} \left(\mathcal{C}(h,f) + \frac{\partial \mathcal{C}}{\partial f}(h,f)(\xi) \right). \tag{5.10}$$

We now aim at devising a gradient algorithm for problem (5.10), which requires to compute the gradient of functional $\widetilde{\mathcal{J}}$ with respect to the thickness $h \in L^\infty(\Omega)$.

The result of interest is the following:

Theorem 5.1. *The functional $\widetilde{\mathcal{J}}$, defined as (5.10) rewrites as:*

$$\forall h \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(h) = \int_\Omega j(f,u_h)\, dx + m\, ||\nabla_f j(f,u_h) - p_h||_{L^2(\Omega)^d},$$

where $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the first adjoint state, defined as the unique solution to

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(p)) & = & -\nabla_u j(f,u_h) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ hAe(p)n & = & 0 & \text{on } \Gamma_N \end{array} \right. . \tag{5.11}$$

Let $h \in \mathcal{U}_{ad}$ such that $\nabla_f j(f,u_h) - p_h \neq 0$ in $L^2(\Omega)^d$. Then $\widetilde{\mathcal{J}}$ is Fréchet differentiable at h, and its derivative reads:

$$\forall s \in L^\infty(\Omega),\ \widetilde{\mathcal{J}}'(h)(s) = \int_\Omega v(u_h, p_h, q_h, z_h)\, s\, dx, \tag{5.12}$$

where:

$$v(u_h,p_h,q_h,z_h) := \left(Ae(u_h) : e(p_h) + \frac{m}{2\,||\nabla_f j(f,u_h) - p_h||_{L^2(\Omega)^d}} \left(Ae(u_h) : e(z_h) + Ae(p_h) : e(q_h) \right) \right).$$

In the last formula, the second, and third adjoint states $q_h, z_h \in H^1_{\Gamma_D}(\Omega)^d$ are defined as the unique solutions to, respectively:

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & -2\left(p_h - \nabla_f j(f,u_h)\right) & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ hAe(q)n & = & 0 & on\ \Gamma_N \end{array}\right. \tag{5.13}$$

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(z)) & = & -2\,\nabla_f\nabla_u j(f,u_h)^T\left(\nabla_f j(f,u_h) - p_h\right) - \nabla^2_u j(f,u_h) q_h & in\ \Omega \\ z & = & 0 & on\ \Gamma_D \\ hAe(z)n & = & 0 & on\ \Gamma_N \end{array}\right. \tag{5.14}$$

Proof. First and foremost, note that we have been a little sloppy in writing expression (5.12) for the derivative of $\widetilde{\mathcal{J}}$ at any point $h \in \mathcal{U}_{ad}$, whereas, rigorously speaking, it only makes sense at $h \in \mathring{\mathcal{U}}_{ad}$. Actually, $\widetilde{\mathcal{J}}$ can be evaluated over a larger set of designs than the sole $\mathcal{U}_{ad}$ (see the definition (5.6)), and its differentiation over the whole set $\mathcal{U}_{ad}$ is not a difficulty. We shall repeatedly commit this minor abuse in notations in the following.

The differentiability issues raised by theorem 5.1 being postponed to lemma 5.1, we proceed within two steps.

• *Expression of $\widetilde{\mathcal{J}}$ as a function of h using an adjoint state.*

As evoked in Section 5.2, this first step consists in encoding the dependence of j on the body force term into an associated adjoint state. We start from the very definition of $\widetilde{\mathcal{J}}$:

$$\widetilde{\mathcal{J}}(h) = \int_\Omega j(f,u_h)dx + \sup_{\substack{\xi \in L^2(\Omega)^d \\ ||\xi||_{L^2(\Omega)^d} \leq m}} \left(\int_\Omega \nabla_f j(f,u_h)\cdot \xi\, dx + \int_\Omega \nabla_u j(f,u_h)\cdot \frac{\partial u_{h,f}}{\partial f}(\xi)\, dx \right). \tag{5.15}$$

Dealing with the first term in the supremum in (5.15) does not pose any problem. As for the second, let us use the variational formula for the derivative $\frac{\partial u_{h,f}}{\partial f}(\xi)$ obtained by differentiating that of $u_{h,f}$ with respect to f (which is possible since we already know that $f \mapsto u_{h,f}$ is differentiable). We get:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega hAe(u_{h,f}) : e(v)\, dx = \int_\Omega f\cdot v\, dx + \int_{\Gamma_N} g\cdot v\, ds,$$

whence:

$$\forall \xi \in L^2(\Omega)^d, \forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega hAe\left(\frac{\partial u_{h,f}}{\partial f}(\xi)\right) : e(v)\, dx = \int_\Omega \xi\cdot v\, dx.$$

Now introduce the first adjoint state $p_h \in H^1_{\Gamma_D}(\Omega)^d$ as the unique solution to the system (5.11). It comes from the associated variational formulation that:

$$\int_\Omega -\nabla_u j(f,u_{h,})\cdot \frac{\partial u_{h,f}}{\partial f}(\xi)\, dx = \int_\Omega hAe(p_h) : e\left(\frac{\partial u_{h,f}}{\partial f}(\xi)\right)\, dx = \int_\Omega p_h\cdot \xi\, dx.$$

In this view, (5.15) becomes:

$$\begin{array}{rcl} \widetilde{\mathcal{J}}(h) & = & \displaystyle\int_\Omega j(f,u_h)\, dx + \sup_{\substack{\xi \in L^2(\Omega)^d \\ ||\xi||_{L^2(\Omega)^d} \leq m}} \left(\int_\Omega \nabla_f j(f,u_h)\cdot \xi dx - \int_\Omega p_h\cdot \xi\, dx \right) \\ & = & \displaystyle\int_\Omega j(f,u_h)\, dx + m\, ||\nabla_f j(f,u_h) - p_h||_{L^2(\Omega)^d} \end{array}, \tag{5.16}$$

where lemma 5.6 has been used from the first line to the second.

Note that a similar approach to the one used in the proof of lemma 5.2 (using Céa's method) could have been used here, instead of directly differentiating with respect to f in the defining problem for $u_{h,f}$.

Céa's method offers a more systematic and understandable derivation of the adjoint equation, and is well-suited when the dependence on f (the variable with respect to which sensitivity is evaluated in general) of the state equation is complex. Both (formal) viewpoints are of course equivalent (see the discussion in chapter 1).

• *Differentiation of $\widetilde{\mathcal{J}}$ with respect to the thickness h.*

At this point, we are brought back to the more classical problem of minimizing a functional $\widetilde{\mathcal{J}}$ with respect to h, and the whole problem of worst-case design with respect to an uncertainty on the body forces has been concealed in the adjoint state p_h.

As expected, the first term $\int_\Omega j(f,u_h)\,dx$ in equation (5.16) is the unperturbed cost function; its differentiation is a classical result recalled in lemma 5.2 below and we obtain:

$$\forall s \in L^\infty(\Omega),\ \ \frac{\partial}{\partial h}\left(\int_\Omega j(f,u_h)\,dx\right)(s) = \int_\Omega sAe(u_h) : e(p_h)\,dx. \tag{5.17}$$

Remark that the same adjoint state p_h is used here to express a sensitivity with respect to a perturbation on the body force term f, while it is used in the first step to express a perturbation on the thickness h. This is indeed natural since p_h actually describes the way j depends on its argument (which is the quantity depending on h and f).

As for differentiating the second term

$$||\nabla_f j(f,u_h) - p_h||_{L^2(\Omega)^d} = \sqrt{\int_\Omega |\nabla_f j(f,u_h) - p_h|^2\,dx},$$

we have to assume that h is such that $\nabla_f j(f,u_h) - p_h \neq 0$ in $L^2(\Omega)^d$. This is a reasonable hypothesis, meaning that the considered plate with thickness h is not indifferent to a small perturbation on the applied body forces. Excluding such a case, and using again Lemma 5.2 with function $\ell : \mathbb{R}^d \times \mathbb{R}^d \ni (u,p) \mapsto \ell(u,p) = |\nabla_f j(f,u) - p|^2$, an elementary calculation yields:

$$\frac{\partial}{\partial h}\left(||\nabla_f j(f,u_h) - p_h||_{L^2(\Omega)^d}\right)(s) = \frac{1}{2\,||\nabla_f j(f,u_h) - p_h||_{L^2(\Omega)^d}} \int_\Omega s(Ae(u_h) : e(z_h) + Ae(p_h) : e(q_h))\,dx, \tag{5.18}$$

where q_h and z_h are defined in (5.13) and (5.14).

Eventually, combining expressions (5.17) and (5.18) delivers the desired formula. □

In the course of the proof, we made use of the following lemma around the regularity of the dependency of the solution $u_{h,f}$ to (5.5) with respect to the source term f. This is a classical result in optimal control theory (see e.g. [9], or [172], chap. 5 in a harder case), which results from a use of the implicit function theorem. In the remaining of this chapter, we shall not dwell too much on these issues (nor on the necessary regularity assumptions that should be put on Ω and f so that u is smooth enough with respect to h and f) and generally content ourselves with formal computations.

Lemma 5.1. *Still denoting as $u_{h,f} \in H^1_{\Gamma_D}(\Omega)^d$ the solution to (5.5) with parameters h, f, the mapping $f \mapsto u_{h,f}$, from $L^2(\mathbb{R}^d)^d$ into $H^1_{\Gamma_D}(\Omega)^d$, is of class $\mathcal{C}^\infty$.*

We also made use of the following general lemma for differentiating functionals depending on h via u_h and the adjoint state p_h, solution to (5.11):

Lemma 5.2. *For $h \in \mathcal{U}_{ad}$, denote as $u_h \in H^1_{\Gamma_D}(\Omega)^d$ the unique solution to problem (5.5).*

(i) Let $j : \mathbb{R}^d \to \mathbb{R}$ be a function of class $\mathcal{C}^1$, which fulfills the corresponding growth conditions in (5.7). Consider the functional:

$$K(h) = \int_\Omega j(u_h)\,dx.$$

Then K is Fréchet differentiable at any $h \in \mathcal{U}_{ad}$ and its derivative reads:

$$\forall s \in L^\infty(\Omega),\ K'(h)(s) = \int_\Omega sAe(u_h) : e(p_h)\, dx.$$

where the adjoint state $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(p)) & = & -\nabla_u j(u_h) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ hAe(p)n & = & 0 & on\ \Gamma_N \end{array} \right. , \tag{5.19}$$

(ii) Suppose moreover that j is of class $\mathcal{C}^2$, let $\ell : \mathbb{R}^d_u \times \mathbb{R}^d_p \to \mathbb{R}$ a function of class $\mathcal{C}^1$ (both of them comply with the relevant growth conditions in (5.7)), and consider the functional:

$$L(h) = \int_\Omega \ell(u_h, p_h)\, dx,$$

where p_h is defined by (5.19). Then L is Fréchet differentiable at any $h \in \mathcal{U}_{ad}$ and its derivative reads:

$$\forall s \in L^\infty(\Omega),\ L'(h)(s) = \int_\Omega s\left(Ae(u_h) : e(z_h) + Ae(p_h) : e(q_h)\right)\, dx.$$

where $q_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & -\nabla_p \ell(u_h, p_h) & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ hAe(q)n & = & 0 & on\ \Gamma_N \end{array} \right. , \tag{5.20}$$

and $z_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(z)) & = & -\nabla_u \ell(u_h, p_h) - \nabla^2_u j(u_h) q_h & in\ \Omega \\ z & = & 0 & on\ \Gamma_D \\ hAe(z)n & = & 0 & on\ \Gamma_N \end{array} \right. , \tag{5.21}$$

Proof. (i): This is a classical result in optimal control theory. For the sake of completeness, we briefly recall its derivation using *Cea's method* [72] (see also chapter 2, §2.2.2.4 in the context of shape optimization). Note that, since we already know that $h \mapsto u_h$ is differentiable (by an analogous result to lemma 5.1), the forthcoming argument is not only formal, as in most cases where Céa's method generally comes in handy. Introduce the Lagrangian $\mathcal{L} : \mathcal{U}_{ad} \times H^1_{\Gamma_D}(\Omega)^d \times H^1_{\Gamma_D}(\Omega)^d \to \mathbb{R}$, defined as:

$$\mathcal{L}(s, \widehat{u}, \widehat{p}) = \int_\Omega j(\widehat{u})\, dx + \int_\Omega sAe(\widehat{u}) : e(\widehat{p})\, dx - \int_\Omega f \cdot \widehat{p}\, dx - \int_{\Gamma_N} g \cdot \widehat{p}\, ds, \tag{5.22}$$

and let us search for the points (u, p) where the partial derivatives of $\mathcal{L}(h, ., .)$ cancel, for a given $h \in \mathcal{U}_{ad}$.

First, the cancellation of the derivative of $\mathcal{L}$ at $(h, u, p) \in \mathcal{U}_{ad} \times H^1_{\Gamma_D}(\Omega)^d \times H^1_{\Gamma_D}(\Omega)^d$ with respect to $\widehat{p}$ reads:

$$\forall \widehat{p} \in H^1_{\Gamma_D}(\Omega)^d,\ \frac{\partial \mathcal{L}}{\partial \widehat{p}}(h, u, p)(\widehat{p}) = \int_\Omega hAe(u) : e(\widehat{p})\, dx - \int_\Omega f \cdot \widehat{p}\, dx - \int_{\Gamma_N} g \cdot \widehat{p}\, ds = 0, \tag{5.23}$$

which is just the variational formulation for problem (5.5); hence $u = u_h$, solution to (5.5).

Next, cancelling the derivative of $\mathcal{L}$ with respect to $\widehat{u}$ leads to the variational formulation for the adjoint state:

$$\forall \widehat{u} \in H^1_{\Gamma_D}(\Omega)^d,\ \frac{\partial \mathcal{L}}{\partial \widehat{u}}(h,u,p)(\widehat{u}) = \int_\Omega \nabla_u j(u)\cdot \widehat{u}\,dx + \int_\Omega hAe(\widehat{u}) : e(p)\,dx = 0, \tag{5.24}$$

which readily gives that $p = p_h$ defined by (5.19).

By definition of u_h, we now have:

$$\forall s \in L^\infty(\Omega),\ \forall \widehat{p} \in H^1_{\Gamma_D}(\Omega)^d,\quad K(s) = \mathcal{L}(s,u_s,\widehat{p}).$$

Thus, differentiating this last expression with respect to s, evaluating it at point h yields:

$$\forall \widehat{p} \in H^1_{\Gamma_D}(\Omega)^d\ \ K'(h)(s) = \frac{\partial \mathcal{L}}{\partial s}(h,u_h,\widehat{p})(s) + \frac{\partial \mathcal{L}}{\partial \widehat{u}}(h,u_h,\widehat{p})(\frac{\partial u_h}{\partial h}(s)).$$

Now taking $\widehat{p} = p_h$ in the previous expression and using (5.24) lead to:

$$K'(h)(s) = \frac{\partial \mathcal{L}}{\partial s}(h,u_h,p_h)(s).$$

Eventually, differentiating (5.22) with respect to s leads to the desired formula:

$$\forall s \in L^\infty(\Omega),\ \ K'(h)(s) = \int_\Omega sAe(u_h) : e(p_h)\,dx.$$

(ii): Similarly, note that in this case, the application $h \mapsto p_h$ is differentiable (see again Lemma 5.1). Let us introduce the (different) Lagrangian $\mathcal{L} : \mathcal{U}_{ad} \times H^1_{\Gamma_D}(\Omega)^d \times H^1_{\Gamma_D}(\Omega)^d \to \mathbb{R}$, defined as:

$$\mathcal{L}(s,\widehat{p},\widehat{q}) = \int_\Omega \ell(u_s,\widehat{p})\,dx + \int_\Omega sAe(\widehat{p}) : e(\widehat{q})\,dx + \int_\Omega \nabla_u j(u_s)\cdot \widehat{q}\,dx.$$

Searching for a point $(p,q) \in H^1_{\Gamma_D}(\Omega)^d \times H^1_{\Gamma_D}(\Omega)^d$ at which the derivative of $\mathcal{L}(h,.,.)$ with respect to $\widehat{q}$ vanishes yields:

$$\forall \widehat{q} \in H^1_{\Gamma_D}(\Omega)^d,\ \ \frac{\partial \mathcal{L}}{\partial \widehat{q}}(h,p,q)(\widehat{q}) = \int_\Omega hAe(p) : e(\widehat{q})\,dx + \int_\Omega \nabla_u j(u_h)\cdot \widehat{q}\,dx = 0, \tag{5.25}$$

and we find $p = p_h$, the solution to (5.19).

Now cancelling the derivative of $\mathcal{L}$ with respect to $\widehat{p}$ one finds:

$$\forall \widehat{p} \in H^1_{\Gamma_D}(\Omega)^d,\ \ \frac{\partial \mathcal{L}}{\partial \widehat{p}}(h,p,q)(\widehat{p}) = \int_\Omega \nabla_p \ell(u_h,p)\cdot \widehat{p}\,dx + \int_\Omega hAe(\widehat{p}) : e(q)\,dx = 0, \tag{5.26}$$

and since $p = p_h$, we identify $q = q_h$ the unique solution to (5.20).

Consequently, we have, for any $s \in \mathcal{U}_{ad}$, and any $\widehat{q} \in H^1_{\Gamma_D}(\Omega)^d$: $L(s) = \mathcal{L}(s,p_s,\widehat{q})$. As in the previous point, differentiating this expression with respect to s, evaluating it at a particular point $h \in \mathcal{U}_{ad}$, then taking $\widehat{q} = q_h$, we end up with:

$$L'(h)(s) = \frac{\partial \mathcal{L}}{\partial s}(h,p_h,q_h)(s).$$

Now, we are left with computing this partial derivative. Since $\mathcal{L}$ depends on $s \in \mathcal{U}_{ad}$ via u_s, we use point (i): for a fixed $h \in L^\infty(\Omega)$, define the $\mathcal{C}^1$ function $m : \mathbb{R} \to \mathbb{R}$ as:

$$\forall u \in \mathbb{R},\ \ m(u) = \ell(u,p_h) + \nabla_u j(u)\cdot q_h.$$

Note that m actually also depends on the space variable $x \in \mathbb{R}^d$, and this dependence is omitted. Applying the first point to function m as an integrand, and introducing $z_h \in H^1_{\Gamma_D}(\Omega)^d$ as the unique solution to (5.21), we end up with:

$$\begin{array}{rcl} L'(h)(s) & = & \dfrac{\partial}{\partial s}\left(\displaystyle\int_\Omega \ell(u_s,p_h)\,dx - \int_\Omega -\nabla_u j(u_s).q_h\,dx\right)\Big|_h(s) + \displaystyle\int_\Omega sAe(p_h):e(q_h)\,dx \\ & = & \displaystyle\int_\Omega sAe(u_h):e(z_h)\,dx + \int_\Omega sAe(p_h):e(q_h)\,dx \end{array},$$

which ends the proof.

□

Remark 5.3. The differentiation of $\widetilde{\mathcal{J}}$ in the second step, using Cea's formal method, is by no means the easiest way to proceed in this particular case. Indeed, expressing the variational problems satisfied by the different derivatives with respect to the thickness h, then introducing the associated adjoint states from the according variational formulation is rather straightforward in this case (see for instance the first step of the proof of Theorem 5.1). Yet, such an easy expression of these derivatives no longer holds when it comes to shape optimization. To put the stress on the similarities between both settings, we thought it better to prove Lemma 5.2 using Cea's method, which will be the convenient tool in Section 5.4.

Example 5.1. For the sake of simplicity, suppose that no surface loads are applied - $g = 0$ (however, the argument would adapt *mutatis mutandis* to the general case)-, and that we are interest in the *compliance* as a cost function, i.e. $j(f,u) = f \cdot u$ in (5.8). The various derivatives of j are:

$$\nabla_u j(f,u) = f,\ \ \nabla_f j(f,u) = u,\ \ \nabla^2_u j(u) = 0,\ \ \nabla_f \nabla_u j(f,u) = I.$$

For any $h \in \mathcal{U}_{ad}$, $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to problem (5.11), and in this case $p_h = -u_h$ the unperturbed problem is self-adjoint, as is well-known in this case). Then, $\widetilde{\mathcal{J}}$ has the following expression:

$$\forall h \in \mathcal{U}_{ad},\ \ \widetilde{\mathcal{J}}(h) = \int_\Omega f \cdot u_h\,dx + 2m\,||u_h||_{L^2(\Omega)^d}. \tag{5.27}$$

Furthermore, $q_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to the problem:

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & 4u_h & \text{in } \Omega \\ q & = & 0 & \text{on } \Gamma_D \\ hAe(q)n & = & 0 & \text{on } \Gamma_N \end{array}\right.,$$

and from (5.14), one acknowledges that $z_h = -q_h$, which gives the straightforward expression for the gradient of $\widetilde{\mathcal{J}}(h)$, for any $h \in \mathcal{U}_{ad}$ such that $u_h \neq 0$:

$$\forall s \in L^\infty(\Omega),\ \ \widetilde{\mathcal{J}}'(h)(s) = -\int_\Omega s\left(Ae(u_h):e(u_h) + \frac{m}{2\,||u_h||_{L^2(\Omega)^d}} Ae(u_h):e(q_h)\right)dx.$$

Remarks 5.4.

- Interestingly enough, formula (5.27) expresses the fact the - at first order - optimizing the worst-case scenario compliance when uncertainties around body forces are expected translates into a penalization (with fixed weight m equal to the magnitude of the anticipated perturbations) of the unperturbed compliance by the norm of the displacement u_h of the structure.
- As we already mentioned in the introduction, in this particular case where the cost function $\mathcal{C}$ is the compliance, the study of the *exact* worst-case functional $\mathcal{J}$ defined in (5.9) can be addressed without linearization of $\mathcal{C}$, resorting to more involved techniques [84, 163].

Adapting in a straightforward way the proofs of Theorem 5.1 and Lemma 5.2 allows to derive analogous results around very close models to that of Example 5.1.

Example 5.2 (Localization of perturbations)**.** Still assuming that $g = 0$, let $\chi \in L^\infty(\Omega)$ be a fixed function on Ω, meant to localize the area where perturbations on the body forces are exerted (for example, χ may be the characteristic function of a subdomain of Ω); i.e. only perturbed configurations associated to body forces of the form $(f + \chi\xi)$, $\xi \in L^2(\Omega)^d$, $||\xi||_{L^2(\Omega)^d} \leq m$ are considered.

Consider once again the compliance as an objective function, meaning that the cost of a plate of thickness $h \in \mathcal{U}_{ad}$ and submitted to body forces $f \in L^2(\Omega)^d$ is $\mathcal{C}(h, f) = \int_\Omega f \cdot u_h \, dx$. The worst-case functional of interest is in this case:

$$\forall h \in \mathcal{U}_{ad}, \;\; \widetilde{\mathcal{J}}(h) = \sup_{\substack{\xi \in L^2(\Omega)^d \\ ||\xi||_{L^2(\Omega)^d} \leq m}} \mathcal{C}(h, f + \chi\xi). \tag{5.28}$$

A mere rephrasing of the proofs of lemma 5.2 and theorem 5.1 allows to derive the following result:

Theorem 5.2. *Consider the functional* $\widetilde{\mathcal{J}} : \mathcal{U}_{ad} \ni h \mapsto \widetilde{\mathcal{J}}(h) = \sup_{\substack{\xi \in L^2(\Omega)^d \\ ||\xi||_{L^2(\Omega)^d} \leq m}} \Big(\mathcal{C}(h, f) + \frac{\partial \mathcal{C}}{\partial f}(h, f)(\chi\xi)\Big)$, *obtained by replacing* $\mathcal{C}$ *by its linear approximation at* (h, f) *in (5.28). Then* $\widetilde{\mathcal{J}}$ *rewrites:*

$$\widetilde{\mathcal{J}}(h) = \int_\Omega f.u_h \, dx + 2m \, ||\chi u_h||_{L^2(\Omega)^d} .$$

Let $h \in \mathcal{U}_{ad}$ *such that* $\chi u_h \neq 0$ *in* $L^2(\Omega)^d$*. Then* $\widetilde{\mathcal{J}}$ *is Fréchet differentiable at* h*, and its differential reads:*

$$\forall s \in L^\infty(\Omega), \;\; \widetilde{\mathcal{J}}'(h)(s) = -\int_\Omega s \left(Ae(u_h) : e(u_h) + \frac{m}{2\,||\chi u_h||_{L^2(\Omega)^d}} Ae(u_h) : e(q_h) \right) dx.$$

where the adjoint state $q_h \in H^1_{\Gamma_D}(\Omega)^d$ *is the unique solution to:*

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & 4\chi u_h & \text{in } \Omega \\ q & = & 0 & \text{on } \Gamma_D \\ hAe(q)n & = & 0 & \text{on } \Gamma_N \end{array} \right.$$

Example 5.3 (Localization and restriction of the direction of perturbations)**.** Eventually, still assuming that $g = 0$, let $\eta \in L^\infty(\Omega)^d$ be a fixed vector field on Ω; the most important case we have in mind is when η is 0 everywhere except on a small portion of Ω, where it is a constant unit direction. The underlying idea is to investigate perturbed body forces of the kind $(f + \xi\eta)$, $\xi \in L^2(\Omega)$, $||\xi||_{L^2(\Omega)} \leq m$.

Still considering the compliance as an objective function, the worst-case functional $\mathcal{J}$ of interest is in this case:

$$\forall h \in \mathcal{U}_{ad}, \;\; \mathcal{J}(h) = \sup_{\substack{\xi \in L^2(\Omega) \\ ||\xi||_{L^2(\Omega)} \leq m}} \mathcal{C}(h, f + \xi\eta), \tag{5.29}$$

where $\mathcal{C}$ is still defined as (5.8). We then have:

Theorem 5.3. *Consider the functional* $\widetilde{\mathcal{J}} : \mathcal{U}_{ad} \ni h \longmapsto \widetilde{\mathcal{J}}(h) = \sup_{\substack{\xi \in L^2(\Omega) \\ ||\xi||_{L^2(\Omega)} \leq m}} \Big(\mathcal{C}(h, f) + \frac{\partial \mathcal{C}}{\partial f}(h, f)(\xi\eta)\Big)$*. Then* $\widetilde{\mathcal{J}}$ *rewrites:*

$$\widetilde{\mathcal{J}}(h) = \int_\Omega f \cdot u_h \, dx + 2m \, ||\eta \cdot u_h||_{L^2(\Omega)} . \tag{5.30}$$

Let $h \in \mathcal{U}_{ad}$ such that $\eta \cdot u_h \neq 0$ in $L^2(\Omega)$. Then $\widetilde{\mathcal{J}}$ is Fréchet-differentiable at h, and its differential reads:

$$\forall s \in L^\infty(\Omega),\ \widetilde{\mathcal{J}}'(h)(s) = -\int_\Omega s\left(Ae(u_h) : e(u_h) + \frac{m}{2\,||\eta \cdot u_h||_{L^2(\Omega)}} Ae(u_h) : e(q_h)\right) dx.$$

where the adjoint state $q_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to:

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & 4(\eta \cdot u_h)\eta & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ hAe(q)n & = & 0 & on\ \Gamma_N \end{array}\right.$$

5.3.3 Extension to a worst-case optimization problem, with respect to a perturbation on surface loads

The analysis of section 5.3.2 adapts in a straightforward way to the case of perturbations concerning the traction loads exerted on the boundary Γ_N of the plate. For any thickness function $h \in \mathcal{U}_{ad}$, body forces $f \in L^2(\Omega)^d$, and any surface loads term $g \in L^2(\Gamma_N)^d$, denote by $u_{h,g} \in H^1_{\Gamma_D}(\Omega)^d$ the unique solution to problem (5.5) using these parameters.

Let $j : \mathbb{R}^d \to \mathbb{R}$ and $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be two functions of class $\mathcal{C}^2$, and define the *cost* of the design with thickness h, when submitted to surface loads g:

$$\mathcal{C}(h, g) = \int_\Omega j(u_{h,g})\, dx + \int_{\Gamma_N} k(g, u_{h,g})\, ds. \tag{5.31}$$

Now, consider fixed body forces $f \in L^2(\Omega)^d$ and surface loads $g \in L^2(\Gamma_N)^d$; we are interested in perturbations on the surface loads of the form $(g + \xi)$, $\xi \in L^2(\Gamma_N)^d$, $||\xi||_{L^2(\Gamma_N)^d} \leq m$, for some parameter $m > 0$. As in the previous section, we shall denote by $u_h = u_{h,g_0}$, the solution to the unperturbed problem.

The worst-case scenario optimization problem for the cost (5.31) reads:

$$\min_{h \in \mathcal{U}_{ad}} \mathcal{J}(h),\ \text{where } \mathcal{J}(h) = \sup_{\substack{\xi \in L^2(\partial\Omega)^d \\ ||\xi||_{L^2(\partial\Omega)^d} \leq m}} \mathcal{C}(h, g + \xi),$$

which is, as in section 5.3.1 traded for the approximate problem:

$$\min_{h \in \mathcal{U}_{ad}} \widetilde{\mathcal{J}}(h),\ \text{where } \widetilde{\mathcal{J}}(h) = \sup_{\substack{\xi \in L^2(\partial\Omega)^d \\ ||\xi||_{L^2(\partial\Omega)^d} \leq m}} \left(\mathcal{C}(h, g) + \frac{\partial \mathcal{C}}{\partial g}(h, g)(\xi)\right). \tag{5.32}$$

This last problem lends itself to an easier analysis, owing to the following result, whose proof is an easy rephrasing of that of Theorem 5.1 and Lemma 5.2, and is then omitted.

Theorem 5.4. *The functional $\widetilde{\mathcal{J}} : \mathcal{U}_{ad} \to \mathbb{R}$, defined in (5.32) rewrites:*

$$\forall h \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(h) = \int_\Omega j(u_{h,g})\, dx + \int_{\Gamma_N} k(g, u_{h,g})\, ds + m\, ||\nabla_g k(g, u_h) - p_h||_{L^2(\Gamma_N)^d},$$

where $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the first adjoint state, defined as the unique solution to

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(p)) & = & -\nabla_u j(u_h) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ hAe(p)n & = & -\nabla_u k(g, u_h) & on\ \Gamma_N \end{array}\right. .$$

Let $h \in \mathring{\mathcal{U}}_{ad}$ such that $\nabla_g k(g,u_h) - p_h \neq 0$ in $L^2(\Gamma_N)^d$. Then $\widetilde{\mathcal{J}}$ is Fréchet differentiable at h, and its differential reads:

$$\forall s \in L^\infty(\Omega),\ \widetilde{\mathcal{J}}'(h)(s) = \int_\Omega s\, v(u_h, p_h, q_h, z_h)\, dx,$$

where

$$v(u_h,p_h,q_h,z_h) = Ae(u_h) : e(p_h) + \frac{m}{2\,||\nabla_g k(g,u_h) - p_h||_{L^2(\Gamma_N)^d}} \left(Ae(u_h) : e(z_h) + Ae(p_h) : e(q_h)\right),$$

and the second and third adjoint states $q_h, z_h \in H^1_{\Gamma_D}(\Omega)^d$ are defined as the unique solutions to, respectively:

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & 0 & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ hAe(q)n & = & -2\,(p_h - \nabla_g k(g,u_h)) & on\ \Gamma_N \end{array}\right.$$

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(z)) & = & -\nabla_u^2 j(u_h) q_h & in\ \Omega \\ z & = & 0 & on\ \Gamma_D \\ hAe(z)n & = & -2\,\nabla_g \nabla_u k(g_0,u_h)^T\,(\nabla_g k(g,u_h) - p_h) & on\ \Gamma_N \end{array}\right.$$

5.3.4 Parametric optimization of a worst-case scenario problem under geometric uncertainty

We now investigate perturbations of a different nature, searching for the optimal thickness $h \in L^\infty(\Omega)$ of the considered plate when robustness is expected with respect to uncertainties over the thickness of the plate itself. As we have already mentioned in the introduction, such a problem typically occurs in the case of mechanical parts which are likely to undergo high stress during their use, thus to wear out, or of mechanical parts whose manufacturing process is especially error-prone.

More accurately, let $f \in L^2(\Omega)^d$, and $g \in L^2(\Gamma_N)^d$ be body forces and surface loads terms, and let $j, k : \mathbb{R}^d \to \mathbb{R}$ be two functions of class $\mathcal{C}^2$; for any $h \in \mathcal{U}_{ad}$, denote as u_h the solution to problem (5.5) when h is the considered thickness function.

Introduce the *cost* of the design with thickness h:

$$\mathcal{C}(h) := \int_\Omega j(u_h)\, dx + \int_{\Gamma_N} k(u_h)\, ds.$$

Modeling uncertainties over the geometry (i.e. thickness) of the plate itself demands first to address an important issue around the perturbed designs: if $h \in \mathcal{U}_{ad}$ and $s \in L^\infty(\Omega)$ is a 'small' perturbation over h, the thickness $(h+s)$ of the corresponding perturbed design may not belong to $\mathcal{U}_{ad}$ (although it is still uniformly bounded away from 0 and ∞). However, we believe this is part of the modeling, for designs generally end up perturbed in an accidental way, and there is no particular reason that a perturbed design should still fulfill any imposed constraint. Furthermore, not enforcing that $(h+s)$ should belong to $\mathcal{U}_{ad}$ allows for an easier mathematical study.

Let $m < h_{min}$ the maximum expected amplitude of the uncertainty over the thickness h. We consider the optimization problem:

$$\min_{h \in \mathcal{U}_{ad}} \mathcal{J}(h),\ \text{where } \mathcal{J}(h) = \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \leq m}} \mathcal{C}(h+s). \tag{5.33}$$

As alluded to in section 5.3.1, this problem is very difficult to tackle in such form. However, in the particular situation where the cost $\mathcal{C}(h)$ is the compliance of the structure, it turns out almost trivial, meaning that the worst case in (5.33) can be found analytically:

Proposition 5.1. *Suppose that the cost function $\mathcal{C}(h)$ is the compliance, that is:*

$$\forall h \in \mathcal{U}_{ad},\ \mathcal{C}(h) = \int_\Omega hAe(u_h) : e(u_h)\, dx = \int_\Omega f \cdot u_h\, dx + \int_{\Gamma_N} g \cdot u_h\, ds.$$

Then, the exact worst-case functional $\mathcal{J}$ equals:

$$\mathcal{J}(h) = \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \leq m}} \mathcal{C}(h+s) = \mathcal{C}(h-m).$$

Simply put, the worst case with respect to the compliance, when there is an uncertainty of maximum amplitude m over h, is the corresponding structure with thickness $(h-m)$, which is the less rigid (thinner) perturbed structure.

Proof. This is a simple consequence of the elastic energy minimization principle. One has:

$$\mathcal{C}(h) = -2 \inf_{u \in H^1_{\Gamma_D}(\Omega)^d} \left(\frac{1}{2} \int_\Omega hAe(u) : e(u)\, dx - \int_\Omega f.u\, dx - \int_{\Gamma_N} g.u\, ds \right).$$

Hence,

$$\begin{aligned}
\mathcal{J}(h) &= \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \leq m}} \sup_{u \in H^1_{\Gamma_D}(\Omega)^d} \left(2\int_\Omega f.u\, dx + 2\int_{\Gamma_N} g.u\, ds - \int_\Omega (h+s)Ae(u) : e(u)\, dx \right) \\
&= \sup_{u \in H^1_{\Gamma_D}(\Omega)^d} \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \leq m}} \left(2\int_\Omega f.u\, dx + 2\int_{\Gamma_N} g.u\, ds - \int_\Omega (h+s)Ae(u) : e(u)\, dx \right), \\
&= \sup_{u \in H^1_{\Gamma_D}(\Omega)^d} \left(2\int_\Omega f.u\, dx + 2\int_{\Gamma_N} g.u\, ds - \int_\Omega (h-m)Ae(u) : e(u)\, dx \right)
\end{aligned}$$

which allows to conclude. □

Remark 5.5. The situation is more complex (and cannot be dealt with analytically) if a constraint on the volume of the perturbed shapes is incorporated into the modeling, e.g. if we are to assume that $\int_\Omega s\, dx = 0$ for any potential perturbation $s \in L^\infty(\Omega)$ over the thickness of shapes.

In the general, non trivial setting (i.e. when $\mathcal{C}(h)$ is not the compliance), we propose to reformulate our optimization problem according to the general principle of Section 5.2:

$$\min_{h \in \mathring{\mathcal{U}}_{ad}} \widetilde{\mathcal{J}}(h),\ \text{where } \widetilde{\mathcal{J}}(h) = \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \leq m}} \left(\mathcal{C}(h) + \frac{\partial \mathcal{C}}{\partial h}(h)(s) \right), \tag{5.34}$$

and the following result makes it possible to build a gradient-based algorithm for this simplified minimization problem.

Theorem 5.5. *The functional $\widetilde{\mathcal{J}}$, defined as (5.34) rewrites:*

$$\forall h \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(j) = \int_\Omega j(u_h)\, dx + \int_{\Gamma_N} k(u_h)\, ds + m\, ||Ae(u_h) : e(p_h)||_{L^1(\Omega)},$$

where $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the first adjoint state, defined as the unique solution to

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(p)) & = & -\nabla_u j(u_h) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ hAe(p)n & = & -\nabla_u k(u_h) & on\ \Gamma_N \end{array} \right. . \tag{5.35}$$

Moreover, $\widetilde{\mathcal{J}}$ is differentiable at any $h \in \mathcal{U}_{ad}$ such that the set

$$E_h := \{x \in \Omega,\ Ae(u_h) : e(p_h) = 0\} \tag{5.36}$$

has zero Lebesgue measure, and its differential at such a point reads:

$$\forall s \in L^\infty(\Omega),\ \widetilde{\mathcal{J}}'(h)(s) = \int_\Omega s\left(Ae(u_h) : e(p_h) + m\left(Ae(p_h) : e(q_h) + Ae(u_h) : e(z_h)\right)\right) dx,$$

the second and third adjoint states $q_h, z_h \in H^1_{\Gamma_D}(\Omega)^d$ being respectively defined as the unique solutions to:

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(q)) & = & \mathrm{div}(\lambda Ae(u_h)) & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ hAe(q)n & = & -\lambda Ae(u_h)n & on\ \Gamma_N \end{array}\right., \tag{5.37}$$

$$\left\{\begin{array}{rcll} -\mathrm{div}(hAe(z)) & = & -\nabla_u^2 j(u_h)q_h + \mathrm{div}(\lambda Ae(p_h)) & in\ \Omega \\ z & = & 0 & on\ \Gamma_D \\ hAe(z)n & = & -\lambda Ae(p_h)n - \nabla_u^2 k(u_h)q_h & on\ \Gamma_N \end{array}\right., \tag{5.38}$$

and $\lambda \in L^\infty(\Omega)$ is defined as $\lambda = \mathrm{sgn}\left(Ae(u_h) : e(p_h)\right)$.

Proof. The derivation of this result is close in essence to that of Theorem 5.1, and we illustrate another way to get the desired expressions, without relying on Céa's method (see Remark 5.3). Knowing that $h \mapsto u_h$ and $h \mapsto p_h$ are differentiable as functions from $\mathcal{U}_{ad}$ into $H^1_{\Gamma_D}(\Omega)^d$ (see lemma 5.1), we achieve a variational formulation for their derivatives by differentiating the ones of (5.5) and (5.35), then introducing the adjoint states p_h, q_h and z_h with simple algebraic manipulations (in the parametric setting).

• *Expression of $\widetilde{\mathcal{J}}$ as a function of h using an adjoint state.*

From the very definition of the cost $\mathcal{C}$, we get, for any $h \in \mathring{\mathcal{U}}_{ad}$:

$$\widetilde{\mathcal{J}}(h) = \int_\Omega j(u_h)\,dx + \int_{\Gamma_N} k(u_h)\,ds + \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \le m}} \left(\int_\Omega \nabla_u j(u_h) \cdot \left(\frac{\partial u_h}{\partial h}(s)\right) dx + \int_{\Gamma_N} \nabla_u k(u_h) \cdot \left(\frac{\partial u_h}{\partial h}(s)\right) ds\right).$$

Then, using the variational formula for the adjoint state p_h, defined as the solution to (5.35), it comes:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega hAe(p_h) : e(v)\,dx = -\int_\Omega \nabla_u j(u_h).v\,dx - \int_{\Gamma_N} \nabla_u k(u_h).v\,dx,$$

and this yields:

$$\widetilde{\mathcal{J}}(h) = \int_\Omega j(u_h)\,dx + \int_{\Gamma_N} k(u_h)\,ds + \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \le m}} \left(-\int_\Omega hAe(p_h) : e\left(\frac{\partial u_h}{\partial h}(s)\right) ds\right).$$

On the other hand, differentiating in the variational formulation of (5.5), one has:

$$\forall s \in L^\infty(\Omega),\ \forall v \in H^1_{\Gamma_D}(\Omega)^d\ \int_\Omega sAe(u_h) : e(v)\,dx = -\int_\Omega hAe\left(\frac{\partial u_h}{\partial h}(s)\right) : e(v)\,dx.$$

Eventually,

$$\widetilde{\mathcal{J}}(h) = \int_\Omega j(u_h)\,dx + \int_{\Gamma_N} k(u_h)\,ds + \sup_{\substack{s \in L^\infty(\Omega) \\ ||s||_{L^\infty(\Omega)} \le m}} \left(\int_\Omega sAe(u_h) : e(p_h))\,ds\right).$$

This, used in combination with Lemma 5.6 and Theorem 5.11 delivers the desired expression for $\widetilde{\mathcal{J}}$.

• *Computation of the Fréchet derivative of* $\widetilde{\mathcal{J}}$.

Assume now that the set E_h defined by (5.36) has zero Lebesgue measure, and introduce $\lambda := \mathrm{sgn}\,(Ae(u_h) : e(p_h)) \in L^\infty(\Omega)$. We then have, for all $s \in L^\infty(\Omega)$,

$$\begin{aligned}\widetilde{\mathcal{J}}'(h)(s) &= \frac{\partial}{\partial h}\left(\int_\Omega j(u_h)\,dx + \int_{\Gamma_N} k(u_h)\,ds\right)\Big|_h (s) \\ &\quad + m\left(\int_\Omega \lambda\, Ae\left(\frac{\partial u_h}{\partial h}(s)\right) : e(p_h)\,dx + \int_\Omega \lambda\, Ae(u_h) : e\left(\frac{\partial p_h}{\partial h}(s)\right)\,dx\right),\end{aligned}$$

where we used Lemma 5.7. As usual, the first term actually rewrites, from the definition of p_h by (5.35),

$$\frac{\partial}{\partial h}\left(\int_\Omega j(u_h)\,dx + \int_{\Gamma_N} k(u_h)\,ds\right)\Big|_h (s) = \int_\Omega sAe(u_h) : e(p_h)\,dx.$$

As for the second term, differentiating with respect to h directly in the variational formulation satisfied by p_h, we get, for all $s \in L^\infty(\Omega)$, and any $v \in H^1_{\Gamma_D}(\Omega)^d$,

$$\begin{aligned}\int_\Omega hAe\left(\frac{\partial p_h}{\partial h}(s)\right) : e(v)\,dx &= -\int_\Omega sAe(p_h) : e(v)\,dx - \int_\Omega \left(\nabla^2_u j(u_h)\frac{\partial u_h}{\partial h}(s)\right)\cdot v\,dx \\ &\quad -\int_{\Gamma_N}\left(\nabla^2_u k(u_h)\frac{\partial u_h}{\partial h}(s)\right)\cdot v\,ds\end{aligned}.$$

Thus, introducing $q_h \in H^1_{\Gamma_D}(\Omega)^d$ as the unique solution to (5.37), we get:

$$\begin{aligned}\int_\Omega \lambda\, Ae(u_h) : e\left(\frac{\partial p_h}{\partial h}(s)\right)\,dx &= -\int_\Omega h\, Ae(q_h) : e\left(\frac{\partial p_h}{\partial h}(s)\right)\,dx \\ &= \int_\Omega sAe(p_h) : e(q_h)\,dx + \int_\Omega \left(\nabla^2_u j(u_h)\frac{\partial u_h}{\partial h}(s)\right)\cdot q_h\,dx \\ &\quad +\int_{\Gamma_N}\left(\nabla^2_u k(u_h)\frac{\partial u_h}{\partial h}(s)\right)\cdot q_h\,ds\end{aligned}.$$

Now introduce one last adjoint state $z_h \in H^1_{\Gamma_D}(\Omega)^d$ as the unique solution to (5.38). This yields, for all $s \in L^\infty(\Omega)$:

$$\begin{aligned}&\int_\Omega \lambda\, Ae\left(\frac{\partial u_h}{\partial h}(s)\right) : e(p_h)\,dx \\ &+\int_\Omega \left(\nabla^2_u j(u_h)\frac{\partial u_h}{\partial h}(s)\right)\cdot q_h\,dx + \int_{\Gamma_N}\left(\nabla^2_u k(u_h)\frac{\partial u_h}{\partial h}(s)\right)\cdot q_h\,ds &= -\int_\Omega hAe(z_h) : e\left(\frac{\partial u_h}{\partial h}(s)\right)\,dx\ , \\ &= \int_\Omega sAe(z_h) : e(u_h)\,dx\end{aligned}$$

thus ending the proof. □

Example 5.4. Consider once again the model case of the minimization of the compliance, assuming $g = 0$ (for simplicity). Then, $j(u) = f.u$, $k(u) = 0$, and one has:

$$\nabla_u j(u) = f \quad , \quad \nabla^2_u j(u) = 0.$$

Since $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to (5.35), one easily finds that $p_h = -u_h$. Then, the approximate worst-case functional $\widetilde{\mathcal{J}}$ rewrites:

$$\forall h \in \mathring{\mathcal{U}}_{ad},\ \ \widetilde{\mathcal{J}}(h) = \int_\Omega f\cdot u_h\,dx + m\,||Ae(u_h) : e(u_h)||_{L^1(\Omega)}\,.$$

Consider a value of the thickness $h \in \mathcal{U}_{ad}$ such that the elastic energy density function $Ae(u_h) : e(u_h)$ does not vanish on Ω except possibly on a subset of null Lebesgue measure. One has:

$$\lambda = \mathrm{sgn}\left(-Ae(u_h) : e(u_h)\right) = -1.$$

In this view, it is then easy to show that the second and third adjoint states actually amount to:

$$q_h = u_h \quad , \quad z_h = -u_h.$$

Eventually, the differential of $\widetilde{\mathcal{J}}$ reads:

$$\forall s \in L^\infty(\Omega),\ \widetilde{\mathcal{J}}'(h)(s) = -(1+2m)\int_\Omega sAe(u_h) : e(u_h)\, dx.$$

Remark 5.6. This expression can be given a mathematical legitimacy when compared to Proposition 5.1, which claims that, in the setting of Example 5.4, the exact worst-case functional $\mathcal{J}$ reads:

$$\forall h \in \mathcal{U}_{ad},\ \mathcal{J}(h) = \mathcal{C}(h-m),$$

whence, for a function $h \in \mathring{\mathcal{U}}_{ad}$,

$$\forall s \in L^\infty(\Omega),\ \mathcal{J}'(h)(s) = \frac{\partial \mathcal{C}}{\partial h}(h-m)(s) = -\int_\Omega sAe(u_{h-m}) : e(u_{h-m})\, dx.$$

Then, it is easy to check that, (still formally) linearizing the above expression with respect to m as in the proofs of Theorems 5.1 and 5.5, the following asymptotic expansion holds, for a given $h \in \mathcal{U}_{ad}$:

$$\forall s \in L^\infty(\Omega),\ \mathcal{J}'(h)(s) = \widetilde{\mathcal{J}}'(h)(s) + \mathcal{O}(m^2).$$

5.3.5 Worst-case design with uncertainties over the elastic material's properties

We eventually investigate one last potential source of uncertainties, namely perturbations on the mechanical behavior of the constituent material of the plate. Suppose the plate is submitted to body forces $f \in L^2(\Omega)^d$, and traction loads $g \in L^2(\Gamma_N)^d$, and that the mechanical properties of the constituent material is subject to variations.

Dealing with this case requires slight changes in the previous notations. For functions $\lambda, \mu \in L^\infty(\Omega)$ such that:

$$\lambda(x), \mu(x) \geq \gamma > 0,\ \text{a.e } x \in \Omega,$$

denote by $A_{\lambda,\mu}e = 2\mu e + \lambda \mathrm{tr}(e)$, the Hooke's tensor with (possibly inhomogeneous) Lamé coefficients λ, μ, and as $u_{h,\lambda,\mu} \in H^1_{\Gamma_D}(\Omega)^d$ the unique solution to the linear elasticity system (5.5) when h is the thickness of the plate and the Lamé moduli of the constituent material are λ, μ.

Let $j : \mathbb{R}^d \to \mathbb{R}$ and $k : \mathbb{R}^d \to \mathbb{R}$ be two functions of class $\mathcal{C}^2$. The cost of the associated plate is defined by:

$$\mathcal{C}(h, \lambda, \mu) = \int_\Omega j(u_{h,\lambda,\mu})\, dx + \int_{\Gamma_N} k(u_{h,\lambda,\mu})\, ds.$$

We investigate perturbations $(\lambda + \alpha, \mu + \beta)$ over the Lamé coefficients of the material of magnitude (i.e. of L^∞-norm) $m < \gamma$, around a reference state (λ, μ), which is thenceforward assumed to be fixed. These coefficients are confessedly not the physically relevant properties of the material over which perturbations should be considered: for instance, the impact of an increase in temperature on the material's properties is certainly better transcribed in terms of the Young's modulus and Poisson ratio. Nevertheless, we will focus on perturbations on the Lamé coefficients, so to keep expressions as light as possible; Young's modulus and Poisson ratio being analytical functions of the Lamé coefficients of the material, this last case would be no more difficult.

In the following, we shall denote as $u_h := u_{h,\lambda,\mu}$, and $A := A_{\lambda,\mu}$, when no ambiguity is possible.

The considered objective function is, in this context:

$$\forall h \in \mathcal{U}_{ad},\ \ \mathcal{J}(h) = \sup_{\substack{\alpha,\beta \in L^\infty(\Omega) \\ ||\alpha||_{L^\infty(\Omega)} \leq m \\ ||\beta||_{L^\infty(\Omega)} \leq m}} \mathcal{C}(h, \lambda + \alpha, \mu + \beta).$$

Note that, an analogous result to proposition 5.1 holds in this case: if the chosen cost function $\mathcal{C}$ is the compliance of the plate, the worst-case functional $\mathcal{J}$ can be computed exactly as:

$$\forall h \in \mathcal{U}_{ad},\ \ \mathcal{J}(h) = \mathcal{C}(h, \lambda - m, \mu - m). \tag{5.39}$$

In the general case, of course, no such expression holds, and functional $\mathcal{J}$ is as usual replaced by the approximated worst-case functional:

$$\widetilde{\mathcal{J}}(h) = \mathcal{C}(h, \lambda, \mu) + \sup_{\substack{\alpha,\beta \in L^\infty(\Omega) \\ ||\alpha||_{L^\infty(\Omega)} \leq m \\ ||\beta||_{L^\infty(\Omega)} \leq m}} \left(\frac{\partial \mathcal{C}}{\partial \lambda}(h, \lambda, \mu)(\alpha) + \frac{\partial \mathcal{C}}{\partial \mu}(h, \lambda, \mu)(\beta) \right). \tag{5.40}$$

We have the following theorem:

Theorem 5.6. *The functional $\widetilde{\mathcal{J}}$ defined by (5.40) reduces to:*

$$\widetilde{\mathcal{J}}(h) = \int_\Omega j(u_h)\, dx + \int_{\Gamma_N} k(u_h)\, ds + 2m||he(u_h) : e(u_h)||_{L^1(\Omega)} + m||h\mathrm{div}(u_h)\mathrm{div}(p_h)||_{L^1(\Omega)}, \tag{5.41}$$

where $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the first adjoint state, defined as the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(hAe(p)) & = & -\nabla_u j(u_h) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ hAe(p)n & = & -\nabla_u k(u_h) & on\ \Gamma_N \end{array} \right. . \tag{5.42}$$

Moreover, $\widetilde{\mathcal{J}}$ is differentiable at any $h \in \mathring{\mathcal{U}}_{ad}$ such that the set

$$E_h := \{x \in \Omega,\ \ Ae(u_h) : e(p_h) = 0 \ or\ \mathrm{div}(u_h)\mathrm{div}(p_h) = 0\}$$

is of null Lebesgue measure, and its differential at such a point reads:

$$\forall s \in L^\infty(\Omega),\ \ \widetilde{\mathcal{J}}'(h)(s) = \int_\Omega s\, v(u_h, p_h, q_h, z_h)\, dx,$$

where

$$v(u_h, p_h, q_h, z_h) = Ae(u_h) : e(p_h) + m\left(2\kappa_e e(u_h) : e(p_h) + \kappa_d \mathrm{div}(u_h)\mathrm{div}(p_h) + Ae(p_h) : e(q_h) + Ae(u_h) : e(z_h)\right),$$

the second and third adjoint states $q_h, z_h \in H^1_{\Gamma_D}(\Omega)^d$ being respectively defined as the unique solutions to some boundary value problems which are best formulated in terms of the associated variational equations:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \ \int_\Omega hAe(q_h) : e(v)\, dx = -\int_\Omega \left(2h\kappa_e e(u_h) : e(v) + h\kappa_d \mathrm{div}(u_h)\mathrm{div}(v)\right)\, dx,$$

$$\begin{array}{rl} \forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \ \displaystyle\int_\Omega hAe(z_h) : e(v)\, dx = & -\displaystyle\int_\Omega \left(2h\kappa_e e(p_h) : e(v) + h\kappa_d \mathrm{div}(p_h)\mathrm{div}(v) + \left(\nabla^2_u j(u_h) q_h\right) \cdot v\right)\, dx \\ & -\displaystyle\int_{\Gamma_N} \left(\nabla^2_u k(u_h) q_h\right) \cdot v\, ds \end{array},$$

and $\kappa_e, \kappa_d \in L^\infty(\Omega)$ are respectively defined as $\kappa_e = \mathrm{sgn}\left(e(u_h) : e(p_h)\right)$, $\kappa_d = \mathrm{sgn}\left(\mathrm{div}(u_h)\mathrm{div}(p_h)\right)$.

Proof. The proof is in essence identical to that of theorem 5.3, and the derivation of formula (5.41) makes use of the general lemma 5.5. □

Example 5.5. As an example, let us specify Theorem 5.6 in the case that the plate is only submitted to body forces (i.e. $g = 0$), and the considered functional is the *compliance* of the structure, that is: $j(u) = f.u$, $k(u) = 0$, and:

$$\nabla_u j(u) = f \quad , \quad \nabla^2_u j(u) = 0.$$

Since $p_h \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to (5.42), one easily finds that $p_h = -u_h$, so that $\widetilde{\mathcal{J}}$ can be rewritten:

$$\forall h \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(h) = \int_\Omega f \cdot u_h \, dx + 2m||he(u_h) : e(u_h)||_{L^1(\Omega)} + m||h\mathrm{div}(u_h)^2||_{L^1(\Omega)}.$$

Now, assuming that $e(u_h)$ and $\mathrm{div}(u_h)$ do not vanish, except possibly on a subset of Ω of null Lebesgue measure, one has: $\kappa_e = \kappa_d = -1$. $q_h \in H^1_{\Gamma_D}(\Omega)^d$ is then defined as the unique solution to the following variational problem:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega hAe(q_h) : e(v)\, dx = \int_\Omega (2he(u_h) : e(v) + h\mathrm{div}(u_h)\mathrm{div}(v))\ dx,$$

$z_h = -q_h$, and easy computations yield:

$$\forall s \in L^\infty(\Omega), \widetilde{\mathcal{J}}'(h)(s) = \int_\Omega s\,(-Ae(u_h) : e(u_h) + m\,(2e(u_h) : e(u_h) + \mathrm{div}(u_h)\mathrm{div}(u_h) - 2Ae(u_h) : e(q_h)))\ dx.$$

This expression can be drawn closer to the exact expression (5.39) for $\mathcal{J}$ in this case. After some computations, we find, as in Remark 5.6, that for any $h \in \mathcal{U}_{ad}$:

$$\forall s \in L^\infty(\Omega),\ \mathcal{J}'(h)(s) = \widetilde{\mathcal{J}}'(h)(s) + \mathcal{O}(m^2).$$

Of course, the whole argument still holds (involving more computations) when $g \neq 0$ and $k(u) = g \cdot u$.

5.4 Worst-case design in shape optimization

5.4.1 Description of the model problem

We now get interested in *shapes*, that is bounded domains $\Omega \subset \mathbb{R}^d$, with at least Lipschitz regularity. Every considered shape Ω is submitted to body forces $f \in H^1(\mathbb{R}^d)^d$. It is moreover clamped on a part $\Gamma_D \subset \partial\Omega$ of its boundary, and subject to traction loads $g \in H^2(\mathbb{R})^d$, applied on another part $\Gamma_N \subset \partial\Omega$. Neither of these parts is subject to optimization, and in this view, contrary to the previous setting of section 5.3, we introduce the *free boundary* $\Gamma := \partial\Omega \setminus (\Gamma_D \cup \Gamma_N)$, which is the only optimizable part of $\partial\Omega$.

The displacement of this shape arises then as the unique solution in $H^1_{\Gamma_D}(\Omega)^d$ to the linear elasticity system posed on Ω:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(u)) & = & f & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ Ae(u)n & = & g & \text{on } \Gamma_N \\ Ae(u)n & = & 0 & \text{on } \Gamma \end{array}\right. , \tag{5.43}$$

where A is the material's Hooke's law. In accordance with this setting, the set $\mathcal{U}_{ad}$ of *admissible domains* is:

$$\mathcal{U}_{ad} = \left\{\Omega \subset \mathbb{R}^d \text{ is open, Lipschitz and bounded, } \Gamma_D \cup \Gamma_N \subset \partial\Omega\right\}.$$

As for representing variations of shapes, we rely once more on Hadamard's boundary variation method (see the outline in Chapter 2): for a shape $\Omega \subset \mathbb{R}^d$, we consider variations of the form:

$$\Omega_\theta := (I + \theta)(\Omega),\ \theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ ||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1.$$

This leads to the following notion of differentiation with respect to the domain:

Definition 5.1. *A functional $J(\Omega)$ of the domain is shape differentiable at Ω if the underlying function $\theta \mapsto J((I+\theta)(\Omega))$, from $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ into $\mathbb{R}$ is Fréchet differentiable at $\theta = 0$. The shape derivative $J'(\Omega)$ of J at Ω is then the corresponding Fréchet differential, so that the following asymptotic expansion holds in the vicinity of $0 \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$:*

$$J(\Omega_\theta) = J(\Omega) + J'(\Omega)(\theta) + o(\theta), \quad \textit{where } \frac{|o(\theta)|}{||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)}} \xrightarrow{\theta\to 0} 0. \tag{5.44}$$

To guarantee that all the considered variations of shapes belong to $\mathcal{U}_{ad}$, the set of deformations of shapes is restricted from $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ to $\Theta_{ad} \subset W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$, where:

$$\Theta_{ad} = \left\{\theta \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d) \text{ s.t. } \theta(x) = 0 \text{ a.e. } x \in \Gamma_D \cup \Gamma_N\right\}.$$

Notation: As in Section 5.3, we will be considering several functions as integrands of our objective functions. The possible dependences of these functions on the space variable x are consistently omitted. If $j : \mathbb{R}^d_f \times \mathbb{R}^d_u \times \mathbb{R}^d_p \to \mathbb{R}$ is any smooth enough function, its partial gradients with respect to the f, u, p variables are still denoted respectively: $\nabla_f j, \nabla_u j, \nabla_p j \in \mathbb{R}^d$.

5.4.2 Worst-case design in shape optimization under uncertainties over the applied body forces

In this section, we aim at optimizing the shape of a structure Ω with respect to the greatest value reached by a given objective function of the domain when small perturbations are expected on the body force term. This problem is the simplest of all the ones we are going to investigate in the shape optimization setting. To keep things simple, we only limit the forthcoming presentation to the simplest case as regards the form of the cost function, which is easily generalized to more complex situations (as well as to ones where perturbations are also expected over the surface loads term).

Let $j : \mathbb{R}^d_f \times \mathbb{R}^d_u \to \mathbb{R}$ be a function of class $\mathcal{C}^2$. For any admissible domain $\Omega \in \mathcal{U}_{ad}$, any body force term $f \in H^1(\mathbb{R}^d)^d$ and any surface loads $g \in H^2(\mathbb{R}^d)$, let $u_{\Omega,f}$ be the corresponding displacement of Ω, solution to problem (5.43) when it is submitted to this set of forces.

The *cost* associated to this configuration is:

$$\mathcal{C}(\Omega, f) = \int_\Omega j(f, u_{\Omega,f})\, dx.$$

We now fix a particular body force term $f \in H^1(\mathbb{R}^d)^d$, and introduce the worst-case optimization problem at stake in the section:

$$\min_{\Omega\in\mathcal{U}_{ad}} \mathcal{J}(\Omega), \text{ where } \mathcal{J}(\Omega) = \sup_{\substack{\xi\in L^2(\mathbb{R}^d)^d \\ ||\xi||_{L^2(\mathbb{R}^d)^d}\le m}} \mathcal{C}(\Omega, f+\xi). \tag{5.45}$$

As proposed in Section 5.2, the following linearized version of problem (5.45) lends itself to a far easier analysis:

$$\min_{\Omega\in\mathcal{U}_{ad}} \widetilde{\mathcal{J}}(\Omega), \text{ where } \widetilde{\mathcal{J}}(\Omega) = \sup_{\substack{\xi\in L^2(\mathbb{R}^d)^d \\ ||\xi||_{L^2(\mathbb{R}^d)^d}\le m}} \left(\mathcal{C}(\Omega, f) + \frac{\partial \mathcal{C}}{\partial f}(\Omega, f)(\xi)\right). \tag{5.46}$$

We now intend to compute the shape gradient of $\widetilde{\mathcal{J}}$. To achieve this, we follow the steps of Section 5.3, and start with the following lemma (which is an equivalent for Lemma 5.2 in the shape optimization context):

Lemma 5.3. *For any $\Omega \in \mathcal{U}_{ad}$, denote by $u_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ the unique solution to (5.43).*

(i) Let $j, k : \mathbb{R}^d \to \mathbb{R}$ be two functions of class $\mathcal{C}^1$, and define a functional K of the domain Ω as:

$$\forall \Omega \in \mathcal{U}_{ad},\ \ K(\Omega) = \int_\Omega j(u_\Omega)\, dx + \int_{\Gamma \cup \Gamma_N} k(u_\Omega)\, dx.$$

Then K is shape differentiable at any $\Omega \in \mathcal{U}_{ad}$, and its derivative reads

$$\forall \theta \in \Theta_{ad},\ \ K'(\Omega)(\theta) = \int_\Gamma \left(j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - p_\Omega.f + \frac{\partial (k(u_\Omega))}{\partial n} + \kappa\, (k(u_\Omega)) \right) (\theta \cdot n)\, ds,$$

where κ is the mean curvature of $\partial\Omega$ (oriented so that it is positive when Ω is locally convex around x), and the adjoint state $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -\nabla_u j(u_\Omega) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ Ae(p)n & = & -\nabla_u k(u_\Omega) & on\ \Gamma \cup \Gamma_N \end{array} \right. . \tag{5.47}$$

(ii) Let $b, c : \mathbb{R}^d \to \mathbb{R}^d$ and $\ell : \mathbb{R}^d_u \times \mathbb{R}^d_p \to \mathbb{R}$ be three functions of class $\mathcal{C}^1$. Note that $\nabla_u b$ and $\nabla_u c$ are $d \times d$ matrices, and that $\nabla_u \ell$ and $\nabla_p \ell$ are vectors in $\mathbb{R}^d$. Introduce the functional L, defined as:

$$\forall \Omega \in \mathcal{U}_{ad},\ \ L(\Omega) = \int_\Omega \ell(u_\Omega, p_\Omega)\, dx,$$

where $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is defined as the unique solution to

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -b(u_\Omega) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ Ae(p)n & = & -c(u_\Omega) & on\ \Gamma \cup \Gamma_N \end{array} \right. . \tag{5.48}$$

L is then shape differentiable at any $\Omega \in \mathcal{U}_{ad}$, and its shape derivative reads

$$\begin{array}{rcl} \forall \theta \in \Theta_{ad},\ \ L'(\Omega)(\theta) & = & \displaystyle\int_\Gamma \left(\ell(u_\Omega, p_\Omega) + b(u_\Omega).q_\Omega + Ae(u_\Omega) : e(z_\Omega) + Ae(p_\Omega) : e(q_\Omega) - z_\Omega.f \right) (\theta \cdot n)\, ds \\ & & \displaystyle + \int_\Gamma \left(\frac{\partial (c(u_\Omega) \cdot q_\Omega)}{\partial n} + \kappa\, (c(u_\Omega) \cdot q_\Omega) \right) (\theta \cdot n)\, ds \end{array},$$

where $q_\Omega, z_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ are respectively defined as the unique solutions to the systems:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(q)) & = & -\nabla_p \ell(u_\Omega, p_\Omega) & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ Ae(q)n & = & 0 & on\ \Gamma \cup \Gamma_N \end{array} \right. , \tag{5.49}$$

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(z)) & = & -\nabla_u b(u_\Omega) q_\Omega - \nabla_u \ell(u_\Omega, p_\Omega) & in\ \Omega \\ z & = & 0 & on\ \Gamma_D \\ Ae(z)n & = & -\nabla_u c(u_\Omega) q_\Omega & on\ \Gamma \cup \Gamma_N \end{array} \right. . \tag{5.50}$$

Proof. (i): this is a particular case of (the proof of) theorem 3.6 in [14].

(ii): Let us once again rely on Céa's method. Introduce the Lagrangian $\mathcal{L} : \mathcal{U}_{ad} \times H^1_{\Gamma_D}(\mathbb{R}^d)^d \times H^1_{\Gamma_D}(\mathbb{R}^d)^d \to \mathbb{R}$, defined as:

$$\mathcal{L}(\Omega, \widehat{p}, \widehat{q}) = \int_\Omega \ell(u_\Omega, \widehat{p})\, dx + \int_\Omega Ae(\widehat{p}) : e(\widehat{q})\, dx + \int_\Omega b(u_\Omega) \cdot \widehat{q}\, dx + \int_{\Gamma \cup \Gamma_N} c(u_\Omega) \cdot \widehat{q}\, ds,$$

and let us search for the points $(p,q) \in H^1_{\Gamma_D}(\mathbb{R}^d)^d \times H^1_{\Gamma_D}(\mathbb{R}^d)^d$ where the partial derivatives of $\mathcal{L}(\Omega,.,.)$ vanish, for a given, arbitrary shape $\Omega \in \mathcal{U}_{ad}$.

• The partial derivative of $\mathcal{L}$ with respect to $\widehat{q}$ at (Ω,p,q) reads:

$$\forall \widehat{q} \in H^1_{\Gamma_D}(\mathbb{R}^d)^d,\ \frac{\partial \mathcal{L}}{\partial \widehat{q}}(\Omega,p,q)(\widehat{q}) = \int_\Omega Ae(p) : e(\widehat{q})\, dx + \int_\Omega b(u_\Omega)\cdot \widehat{q}\, dx + \int_{\Gamma\cup\Gamma_N} c(u_\Omega)\cdot \widehat{q}\, ds.$$

Canceling this expression against test functions with compact support in Ω, say $\widehat{q} \in C^\infty_c(\Omega)$, implies that p is a solution to: $-\mathrm{div}(Ae(p)) = -b(u_\Omega)$ in Ω. Then, using test functions $\widehat{q}$ with null trace on Γ_D yields the boundary condition: $Ae(p)n = -c(u_\Omega)$ on $\Gamma\cup\Gamma_N$. Eventually, since by essence $p \in H^1_{\Gamma_D}(\mathbb{R}^d)$, we readily obtain that $p = p_\Omega$, the 'adjoint' state defined by (5.48).

• The derivative of $\mathcal{L}$ with respect to $\widehat{p}$ evaluated at (Ω,p,q) reads:

$$\forall \widehat{p} \in H^1_{\Gamma_D}(\mathbb{R}^d)^d,\ \frac{\partial \mathcal{L}}{\partial \widehat{p}}(\Omega,p,q)(\widehat{p}) = \int_\Omega \nabla_p \ell(u_\Omega,p)\cdot \widehat{p}\, dx + \int_\Omega Ae(\widehat{p}) : e(q)\, dx.$$

Canceling this expression against test functions with compact support in Ω, $\widehat{p} \in C^\infty_c(\Omega)$, we get that q is a solution to: $-\mathrm{div}(Ae(q)) = -\nabla_p \ell(u_\Omega,p_\Omega)$ in Ω. As above, using test functions $\widehat{p}$ with null trace on Γ_D yield the boundary condition: $Ae(q)n = 0$ on $\Gamma\cup\Gamma_N$, and the last condition $q = 0$ on Γ_D is naturally recovered from the definition $q \in H^1_{\Gamma_D}(\mathbb{R}^d)$. Eventually, $q = q_\Omega$, defined by (5.49).

• Eventually, one has, for any $\Omega \in \mathcal{U}_{ad}$, and for any $\widehat{q} \in H^1_{\Gamma_D}(\mathbb{R}^d)$, $L(\Omega) = \mathcal{L}(\Omega,p_\Omega,\widehat{q})$. As usual, differentiating this relation with respect to Ω, then taking $\widehat{q} = q_\Omega$ in the resulting expression yields:

$$\forall \theta \in \Theta_{ad},\ L'(\Omega)(\theta) = \frac{\partial \mathcal{L}}{\partial \Omega}(\Omega,p_\Omega,q_\Omega)(\theta).$$

This last (partial) shape derivative can now be computed using point (i), since it depends on Ω only via u_Ω, the solution to (5.43). Introducing the third adjoint state $z_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ as the unique solution to (5.50), one has:

$$\begin{array}{rcl} \forall \theta \in \Theta_{ad},\ L'(\Omega)(\theta) & = & \displaystyle\int_\Gamma \left(\ell(u_\Omega,p_\Omega) + b(u_\Omega)\cdot q_\Omega + Ae(u_\Omega) : e(z_\Omega) + Ae(p_\Omega) : e(q_\Omega) - z_\Omega \cdot f\right)(\theta\cdot n)\, ds \\ & & \displaystyle + \int_\Gamma \left(\frac{\partial (c(u_\Omega)\cdot q_\Omega)}{\partial n} + \kappa\,(c(u_\Omega)\cdot q_\Omega)\right)(\theta\cdot n)\, ds \end{array},$$

which is the desired formula. □

We are now in position to carry out the shape sensitivity analysis of functional (5.46):

Theorem 5.7. *The functional $\widetilde{\mathcal{J}}$ defined in (5.46) rewrites:*

$$\forall \Omega \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(\Omega) = \int_\Omega j(f,u_\Omega)\, dx + m||\nabla_f j(f,u_\Omega) - p_\Omega||_{L^2(\Omega)^d}, \tag{5.51}$$

where $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is the first adjoint state, defined as the unique solution to

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -\nabla_u j(f,u_\Omega) & in\ \Omega \\ p & = & 0 & on\ \Gamma_D \\ Ae(p)n & = & 0 & on\ \Gamma\cup\Gamma_N \end{array} \right. . \tag{5.52}$$

Moreover, $\widetilde{\mathcal{J}}$ is differentiable at any $\Omega \in \mathcal{U}_{ad}$ such that $\nabla_f j(f,u_\Omega) - p_\Omega \neq 0$ in $L^2(\Omega)^d$ and its derivative at such a point reads:

$$\forall \theta \in \Theta_{ad}, \quad \begin{array}{rcl} \widetilde{\mathcal{J}}'(\Omega)(\theta) & = & \displaystyle\int_\Gamma \left(j(f,u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - p_\Omega \cdot f\right)(\theta\cdot n)\, ds \\ & & + \frac{m}{2||\nabla_f j(f,u_\Omega)-p_\Omega||_{L^2(\Omega)^d}} \left(\displaystyle\int_\Gamma \left(|\nabla_f j(f,u_\Omega) - p_\Omega|^2 + \nabla_u j(f,u_\Omega).q_\Omega - z_\Omega.f \right)(\theta\cdot n)\, ds \right) \\ & & + \frac{m}{2||\nabla_f j(f,u_\Omega)-p_\Omega||_{L^2(\Omega)^d}} \left(\displaystyle\int_\Gamma \left(Ae(u_\Omega) : e(z_\Omega) + Ae(p_\Omega) : e(q_\Omega) \right)(\theta\cdot n)\, ds \right). \end{array}$$

where the second and third adjoint states $q_\Omega, z_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ are respectively defined as the unique solutions to the following systems:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(q)) & = & -2\left(p_\Omega - \nabla_f j(f,u_\Omega)\right) & in\ \Omega \\ q & = & 0 & on\ \Gamma_D \\ Ae(q)n & = & 0 & on\ \Gamma_N \end{array} \right. . \tag{5.53}$$

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(z)) & = & -\nabla_u^2 j(f,u_\Omega) q_\Omega - 2\nabla_f \nabla_u j(f,u_\Omega)^T (\nabla_f j(f,u_\Omega) - p_\Omega) & in\ \Omega \\ z & = & 0 & on\ \Gamma_D \\ Ae(z)n & = & 0 & on\ \Gamma_N \end{array} \right. . \tag{5.54}$$

Proof. Once again, the (formal) proof follows the general outline sketched above, and is divided into two steps.

• *Rearrangement of the expression of $\widetilde{\mathcal{J}}$ as a function of Ω.*

To achieve an explicit expression of $\widetilde{\mathcal{J}}$ in terms of Ω, we introduce an adjoint state, whose expression may be found by differentiating the variational formula for (5.43) with respect to f. Let us start with:

$$\forall \xi \in L^2(\Omega)^d;\ \frac{\partial \mathcal{C}}{\partial f}(\Omega,f)(\xi) = \int_\Omega \left(\nabla_f j(f,u_\Omega).\xi + \nabla_u j(f,u_\Omega) \cdot \left(\frac{\partial u_{\Omega,f}}{\partial f}(\xi) \right) \right) dx.$$

As usual, the first part $\nabla_f j(f,u_\Omega).\xi$ of the integrand does not pose any problem since it leads to an explicit expression with respect to Ω. As for the second one, the variational formulation associated to $u_{\Omega,f}$ reads, for any source term $f \in L^2(\Omega)^d$:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega Ae(u_{\Omega,f}) : e(v)\, dx = \int_\Omega f\cdot v\, dx + \int_{\Gamma_N} g\cdot v\, ds,$$

whence

$$\forall \xi \in L^2(\Omega)^d,\ \forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega Ae\left(\frac{\partial u_{\Omega,f}}{\partial f}(\xi) \right) : e(v)\, dx = \int_\Omega f\cdot \xi\, dx.$$

On the other hand, from the defining system (5.52) for $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$, one gets:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega Ae(p_\Omega) : e(v)\, dx = -\int_\Omega \nabla_u j(f,u_\Omega)\cdot v\, dx.$$

Combining those two relations produces:

$$\forall \xi \in L^2(\Omega)^d,\ \int_\Omega \nabla_u j(f,u_\Omega) \cdot \left(\frac{\partial u_{\Omega,f}}{\partial f}(\xi) \right) dx = -\int_\Omega Ae(p_\Omega) : e\left(\frac{\partial u_{\Omega,f}}{\partial f}(\xi) \right) dx = -\int_\Omega f\cdot p_\Omega\, dx.$$

Hence, using one more time lemma 5.6 delivers the sought expression for $\widetilde{\mathcal{J}}(\Omega)$.

• *Shape sensitivity analysis of* $\widetilde{\mathcal{J}}$.

Differentiating the first term in (5.56) with respect to the domain is a straightforward application of lemma (5.3), (i). Its shape derivative reads:

$$\forall \theta \in \Theta_{ad},\ \frac{d}{d_\Omega}\left(\int_\Omega j(f,u_\Omega)\,dx\right)(\theta) = \int_\Gamma \left(j(f,u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - p_\Omega \cdot f\right)(\theta\cdot n)\,ds.$$

As for the second term, we use lemma (5.3), (ii), with $\ell(u,p) = |\nabla_f j(f,u) - p|^2$, so that

$$\ell'(u,p) = 2\,\nabla_f \nabla_u j(u)^T \cdot (\nabla_f j(f,u) - p) \quad , \quad \nabla_p \ell(u,p) = 2(p - \nabla_f j(f,u)).$$

Doing so entails, for all $\theta \in \Theta_{ad}$:

$$\begin{aligned}\frac{d}{d_\Omega}\left(\int_\Omega |\nabla_f j(f,u_\Omega) - p_\Omega|^2\,dx\right)(\theta) &= \int_\Gamma \left(|\nabla_f j(f,u_\Omega) - p_\Omega|^2 + \nabla_u j(f,u_\Omega)\cdot q_\Omega - z_\Omega \cdot f\right)(\theta\cdot n)\,ds \\ &\quad + \int_\Gamma \left(Ae(u_\Omega) : e(z_\Omega) + Ae(p_\Omega) : e(q_\Omega)\right)(\theta\cdot n)\,ds\end{aligned}$$

where $q_\Omega, z_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ are defined by (5.53) and (5.54). These two identities lead to the desired formula for $\widetilde{\mathcal{J}}'(\Omega)(\theta)$. □

Example 5.6. As an illustration of Theorem 5.7, assume that $g = 0$ and consider the case of the compliance as a cost function. We have: $j(f,u) = f \cdot u$, therefore:

$$\nabla_f j(f,u) = u\ ,\ \nabla_u j(f,u) = f\ ,\ \nabla_f \nabla_u j(f,u) = I\ ,\ \nabla^2_u j(f,u) = 0.$$

In this context, it is easy to see that $p_\Omega = -u_\Omega$. Thus, $\widetilde{\mathcal{J}}$ admits the following expression:

$$\forall \Omega \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(\Omega) = \int_\Omega f.u_\Omega\,dx + 2m||u_\Omega||_{L^2(\Omega)^d}.$$

Furthermore, if $u_\Omega \neq 0$ in $L^2(\Omega)^d$, $\widetilde{\mathcal{J}}$ is shape differentiable at Ω. Easy computations allow then to deduce that the adjoint state $q_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is the unique solution to the system:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(q)) &=& 4u_\Omega & \text{in } \Omega \\ q &=& 0 & \text{on } \Gamma_D \\ Ae(q)n &=& 0 & \text{on } \Gamma \cup \Gamma_N \end{array}\right. ,$$

and that $z_\Omega = -q_\Omega$. The shape derivative of $\widetilde{\mathcal{J}}$ at Ω then takes the form:

$$\begin{aligned}\forall \theta \in \Theta_{ad},\ \widetilde{\mathcal{J}}'(\Omega)(\theta) &= \int_\Gamma \left(2u_\Omega.f - Ae(u_\Omega) : e(u_\Omega)\right)(\theta\cdot n)\,ds \\ &\quad + \tfrac{m}{4||u_\Omega||_{L^2(\Omega)^d}}\int_\Gamma \left(4|u_\Omega|^2 + 2q_\Omega.f - 2Ae(u_\Omega) : e(q_\Omega)\right)(\theta\cdot n)\,ds.\end{aligned}$$

5.4.3 Worst-case design in shape optimization under uncertainties on the Lamé moduli of the material

This section is intended as the mirror image of Section 5.3.5 in the context of shape optimization, namely, we investigate into the worst-case design of an objective functional of the domain with respect to perturbations on the Lamé coefficients of the elastic material filling Ω.

Let us first recollect some notations. As in Section 5.3.5, for functions $\lambda, \mu \in L^\infty(\mathbb{R}^d)$ such that

$$\exists \gamma > 0,\ \forall x \in \mathbb{R}^d,\ \lambda(x) > \gamma,\ \mu(x) > \gamma, \tag{5.55}$$

denote by $A_{\lambda,\mu} e = 2\mu e + \lambda \mathrm{tr}(e)$, $e \in \mathcal{S}(\mathbb{R}^d)$ the linear elasticity tensor with Lamé coefficients λ, μ, and as $u_{\Omega,\lambda,\mu} \in H^1_{\Gamma_D}(\Omega)^d$ the solution to problem (5.43) posed on a shape Ω filled with such a material.

Let $j : \mathbb{R}^d \to \mathbb{R}$ and $k : \mathbb{R}^d \to \mathbb{R}$ be two functions of class $\mathcal{C}^2$. For any functions $\lambda, \mu \in L^\infty(\mathbb{R}^d)$ satisfying (5.55), the *cost* of the structure Ω filled with a material with such Lamé coefficients is defined as:

$$\mathcal{C}(\Omega, \lambda, \mu) = \int_\Omega j(u_{\Omega,\lambda,\mu})\, dx + \int_{\Gamma \cup \Gamma_N} k(u_{\Omega,\lambda,\mu})\, ds.$$

Let us now fix Lamé coefficients λ, μ. For the sake of simplicity, when the context is clear, we still denote $u_\Omega = u_{\Omega,\lambda,\mu}$ and $A = A_{\lambda,\mu}$. Considering perturbations of magnitude $m < \gamma$ over λ, μ, the corresponding worst-case objective function $\mathcal{J}$ is then:

$$\forall \Omega \in \mathcal{U}_{ad},\ \ \mathcal{J}(\Omega) = \sup_{\substack{\alpha,\beta \in L^\infty(\mathbb{R}^d) \\ ||\alpha||_{L^\infty(\mathbb{R}^d)} \leq m \\ ||\beta||_{L^\infty(\mathbb{R}^d)} \leq m}} \mathcal{C}(\Omega, \lambda + \alpha, \mu + \beta),$$

and the considered approximated objective function $\widetilde{\mathcal{J}}$ then reads:

$$\forall \Omega \in \mathcal{U}_{ad},\ \ \widetilde{\mathcal{J}}(\Omega) = \mathcal{C}(\Omega, \lambda, \mu) + \sup_{\substack{\alpha,\beta \in L^\infty(\mathbb{R}^d) \\ ||\alpha||_{L^\infty(\mathbb{R}^d)} \leq m \\ ||\beta||_{L^\infty(\mathbb{R}^d)} \leq m}} \left(\frac{\partial \mathcal{C}}{\partial \lambda}(\Omega, \lambda, \mu)(\alpha) + \frac{\partial \mathcal{C}}{\partial \mu}(\Omega, \lambda, \mu)(\beta) \right).$$

We have the following result, whose proof is omitted:

Theorem 5.8. *The considered functional $\widetilde{\mathcal{J}}$ rewrites, for any $\Omega \in \mathcal{U}_{ad}$:*

$$\widetilde{\mathcal{J}}(\Omega) = \int_\Omega j(u_\Omega)\, dx + \int_{\Gamma \cup \Gamma_N} k(u_\Omega)\, dx + 2m||e(u_\Omega) : e(p_\Omega)||_{L^1(\Omega)} + m||\mathrm{div}(u_\Omega)\mathrm{div}(p_\Omega)||_{L^1(\Omega)}, \tag{5.56}$$

where $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is the first adjoint state, defined as the unique solution to

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -\nabla_u j(u_\Omega) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ Ae(p)n & = & -\nabla_u k(u_\Omega) & \text{on } \Gamma \cup \Gamma_N \end{array} \right. . \tag{5.57}$$

Moreover, $\widetilde{\mathcal{J}}$ is differentiable at any shape $\Omega \in \mathcal{U}_{ad}$ such that the set

$$E_\Omega := \{x \in \Omega,\ \ (e(u_\Omega) : e(p_\Omega))(x) = 0 \text{ or } (\mathrm{div}(u_\Omega)\mathrm{div}(p_\Omega))(x) = 0\}$$

is of null of null Lebesgue measure, and its shape derivative at such a point is:

$$\forall \theta \in \Theta_{ad},\ \ \widetilde{\mathcal{J}}'(\Omega)(\theta) = \int_\Gamma v(u_h, p_h, q_h, z_h)\ (\theta \cdot n)\, ds,$$

where:

$$\begin{array}{rcl} v(u_h, p_h, q_h, z_h) & = & j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - p_\Omega \cdot f \\ & & +m\,(2|e(u_\Omega) : e(p_\Omega)| + |\mathrm{div}(u_\Omega)\mathrm{div}(p_\Omega)| + \nabla_u j(u_\Omega).q_\Omega + Ae(p_\Omega) : e(q_\Omega) + Ae(u_\Omega) : e(z_\Omega)) \\ & & + \frac{\partial(k(u_\Omega))}{\partial n} + \kappa\,(k(u_\Omega)) + m\left(\frac{\partial(\nabla_u k(u_\Omega) \cdot q_\Omega)}{\partial n} + \kappa\,(\nabla_u k(u_\Omega) \cdot q_\Omega) \right) \end{array}$$

where the second and third adjoint states $q_\Omega, z_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ are respectively defined as the unique solutions to the variational equations:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega Ae(q_\Omega) : e(v)\, dx = -\int_\Omega \left(2\kappa_e e(u_\Omega) : e(v) + \kappa_d \mathrm{div}(u_\Omega)\mathrm{div}(v)\right)\, dx,$$

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \int_\Omega Ae(z_\Omega) : e(v)\, dx = -\int_\Omega \left(2\kappa_e e(p_\Omega) : e(v) + \kappa_d \mathrm{div}(p_\Omega)\mathrm{div}(v) + \left(\nabla^2_u j(u_\Omega) q_\Omega\right) \cdot v\right)\, dx - \int_{\Gamma\cup\Gamma_N} \left(\nabla^2_u k(u_\Omega) q_\Omega\right) \cdot v\, ds,$$

and $\kappa_e, \kappa_d \in L^\infty(\Omega)$ are respectively defined as: $\kappa_e = \mathrm{sgn}\left(e(u_\Omega) : e(p_\Omega)\right)$, and $\kappa_d = \mathrm{sgn}\left(\mathrm{div}(u_\Omega)\mathrm{div}(p_\Omega)\right)$.

Remark 5.7. The observation of Remark 5.5 can be extended from the parametric to the shape optimization setting (up to some extra computations): when the cost function $\mathcal{C}$ is the compliance, i.e. $j(u) = f \cdot u$ and $k(u) = g \cdot u$, the exact worst-case functional $\mathcal{J}$ reads:

$$\forall \Omega \in \mathcal{U}_{ad},\ \mathcal{J}(\Omega) = \mathcal{C}(\Omega, \lambda - m, \mu - m),$$

and the following asymptotic expansion holds at any $\Omega \in \mathcal{U}_{ad}$:

$$\forall \theta \in \Theta_{ad},\ \mathcal{J}'(\Omega)(\theta) = \widetilde{\mathcal{J}}'(\Omega)(\theta) + \mathcal{O}(m^2).$$

5.4.4 Worst-case design in shape optimization under geometric uncertainties

The purpose of this section is to compute a shape derivative for a given functional of the domain $J(\Omega)$ which is robust with respect to uncertainties on the boundaries of shapes themselves.

Let us first specify what we intend by *shape optimization under geometric uncertainty*. We assume that perturbations only affect the free boundary Γ of Ω; more precisely, let $\chi : \mathbb{R}^d \to \mathbb{R}$ be a cutoff function, and $\mathcal{O}_1 \subsetneq \mathcal{O}_2$ be two open neighborhoods of $\Gamma_D \cup \Gamma_N$ in $\mathbb{R}^d$, enjoying the following properties:

$$\chi \text{ is smooth and nonnegative over } \mathbb{R}^d,\ \begin{cases} \chi(x) = 0 & \text{for} \quad x \in \mathcal{O}_1, \\ \chi(x) \in (0,1] & \text{for} \quad x \in \mathcal{O}_2 \setminus \overline{\mathcal{O}_1} \\ \chi(x) = 1 & \text{for} \quad x \in \mathcal{O}_2 \end{cases} \tag{5.58}$$

If $m > 0$ is the expected magnitude of perturbations over the geometry, we are interested in perturbations of $\Omega \in \mathcal{U}_{ad}$ of the form (see Figure 5.2):

$$(I + \chi V)(\Omega),\ V \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ ||V||_{L^\infty(\mathbb{R}^d)^d} \leq m. \tag{5.59}$$

Note that an other way to describe this problem consists in assuming perturbations of Ω of the form:

$$(I + \chi v n)(\Omega),\ v \in W^{1,\infty}(\mathbb{R}^d),\ ||v||_{L^\infty(\mathbb{R}^d)} \leq m, \tag{5.60}$$

where $n = n_\Omega$ denotes (an extension to $\mathbb{R}^d$ of) the normal vector field to $\partial\Omega$ (the $_\Omega$ - index is meant to emphasize its dependence on Ω and will be omitted when the situation is clear).

As we shall observe, both descriptions are equivalent as far as we are concerned. However, the former one (5.59), which is the one retained in the following, proves more convenient from a mathematical viewpoint, since it features independent sets for admissible shapes and admissible perturbations of them; on a different note, perturbed shapes in the sense of (5.60) are 'less regular' than the unperturbed one.

Remark 5.8. The chosen description for perturbations over the geometry slightly differs from that adopted in other contributions on the topic, for instance:

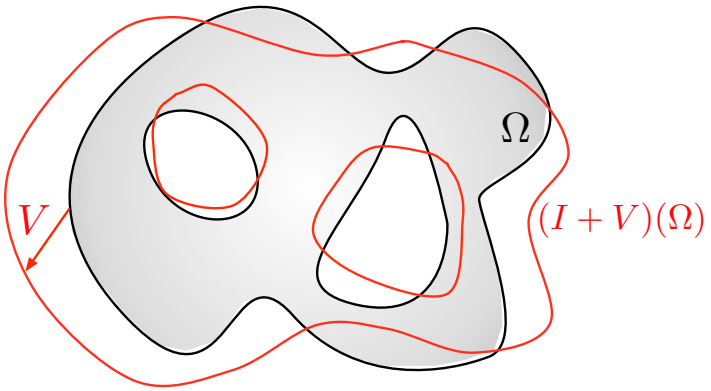


Figure 5.2: Perturbation $(I + \chi V)(\Omega)$ of a domain Ω

- in [286], the only retained possibility is that Ω may suffer from a (small) uniform 'shrinking' or 'thickening', i.e. perturbations of shapes are of the form (5.60), with constant v; a filtering approach is used to incorporate this uncertainty into the objective function.
- Closer to the present work, in [77], the authors also perturbations of the form (5.60) with a scalar field v varying in a random fashion over the boundary of the shape, following a Gaussian probability distribution with 0 mean value.
- Eventually, in [165], perturbations of a shape Ω are of the form (5.60), with v being bounded in $L^2(\mathbb{R}^d)$-norm, and 'small' in the sense that the discrepancy between the volumes of the perturbed and unperturbed shapes is 'small'.

We believe that the above setting is well-suited to simulate 'realistic' uncertainties over manufacturing constraints, but the approaches considered in [165, 286] could be tackled owing to similar computations to those presented in this section.

Actually, the proposed approach in this chapter almost requires to differentiate J *twice* with respect to the domain. Hence, the formulae derived below could also be used in the context of a second-order algorithm for shape optimization.

Let us start with a fairly informal discussion to emphasize the connection between both topics. For any $\Omega \in \mathcal{U}_{ad}$, denote as $u_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ the unique solution to problem (5.43). Let $J(\Omega)$ be a functional of the domain (e.g. one of those we have been considering hitherto) whose minimization is under scrutiny. Under reasonable regularity assumptions on the data, it is well-known (and we have seen several such examples) that the shape derivative of J can be put under the generic form:

$$\forall \theta \in \Theta_{ad}, \quad J'(\Omega)(\theta) = \int_\Gamma j(x, u_\Omega, e(u_\Omega))\,(\theta \cdot n_\Omega)\,ds, \tag{5.61}$$

for some function j.

On the other hand, the proposed approach in this note deals with functionals $\widetilde{\mathcal{J}}$ of the domain cooked by linearizing the functional J under consideration around each shape with respect to the expected perturbations, then taking the supremum of the resulting linear function over all possible perturbations of prescribed maximum amplitude m. Consequently, when perturbations over the geometry of the shape itself are expected, the method described in section 5.2 naturally brings about functionals of the form:

$$\widetilde{\mathcal{J}}(\Omega) = \int_\Gamma k(x, u_\Omega, e(u_\Omega))\,ds, \tag{5.62}$$

for some function k, closely related to j. Devising a numerical algorithm for minimizing $\widetilde{\mathcal{J}}$ naturally demands to differentiate in (5.62). Actually, doing so almost boils down to differentiating expressions such as (5.61), that is to differentiating J *twice* with respect to the domain. Indeed, the general forms of (5.61) and (5.62) only differ from one another because of the factor $(\theta \cdot n_\Omega)$. Yet, provided Ω is smooth enough, the *Eulerian* derivative of the normal vector field n_Ω reads (see [234], or the computation in chapter 4, §4.2.4):

$$\forall x \in \partial\Omega,\ \frac{d}{dt}\left(n_{(I+t\theta)(\Omega)}(x)\right)\Big|_{t=0} = -\nabla_{\partial\Omega}(\theta \cdot n_\Omega)(x).$$

Thus, applying (formally) the chain rule entails, for all $\theta, \xi \in \Theta_{ad}$:

$$\begin{aligned} J''(\Omega)(\theta,\xi) &:= \left(J'(\Omega)(\theta)\right)'(\xi) \\ &= \frac{d}{d_\Omega}\left(\int_\Gamma j(x,u_\Omega,e(u_\Omega))v\,ds\right)\Big|_{v=\theta\cdot n_\Omega}(\xi) - \int_\Gamma j(x,u_\Omega,e(u_\Omega))\ (\theta\cdot\nabla_{\partial\Omega}(\xi\cdot n_\Omega))\ ds\ . \end{aligned}$$

Now, if we constrain θ to be a *normal* vector field, that is $\theta = (\theta \cdot n)n$ (at least in a neighborhood of Γ) - which is very natural because of the structure theorem for shape derivatives - see [105], Th. 9.3.6 or Theorem 2.3 in Chapter 2 of this manuscript -, the second term vanishes, and the computation of $J''(\Omega)$ amounts to differentiating in an expression of the form (5.62). Hence, both problems of computing the shape Hessian of J, and analyzing the worst-case design of J with respect to geometric uncertainties are very much akin, and some ideas in the remainder of this section could be prove useful in the device of a second order shape optimization algorithm (see [109] for further remarks around second-order shape derivatives).

Let us now get into the heart of the matter. Let $\mathcal{C}(\Omega)$ be a cost functional; the associated worst-case scenario functional is:

$$\mathcal{J}(\Omega) = \sup_{\substack{V\in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)\\ ||V||_{L^\infty(\mathbb{R}^d)^d}\leq m}} \mathcal{C}((I+\chi V)(\Omega)),$$

where χ is the cutoff function defined by (5.58).

In what follows, we will focus on several particular cases as regards the form of $\mathcal{C}(\Omega)$; the presented techniques could easily be generalized to different problems. The first investigated example will be that of the *compliance* $C(\Omega)$ of a shape Ω:

$$C(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\,dx = \int_\Omega f\cdot u_\Omega\,dx + \int_{\Gamma_N} g\cdot u_\Omega\,ds; \tag{5.63}$$

then we will turn to a functional $J(\Omega)$, which depends only on u_Ω (not on $e(u_\Omega)$) by means of a smooth enough function $j : \mathbb{R}^d \to \mathbb{R}$ (we have in mind the least-square discrepancy criterion, with respect to a target displacement):

$$J(\Omega) = \int_\Omega j(u_\Omega)\,dx. \tag{5.64}$$

Finally, we will consider the case of a functional $S(\Omega)$, which depends only on the stress tensor $\sigma(u_\Omega) := Ae(u_\Omega)$, through a smooth function $j : \mathcal{S}(\mathbb{R}^d) \to \mathbb{R}$ (we have in mind a L^p-norm of the stress, as studied in [13]):

$$S(\Omega) = \int_\Omega j(\sigma(u_\Omega))\,dx, \tag{5.65}$$

where $\sigma(u_\Omega) := Ae(u_\Omega)$ is the stress tensor associated to the displacement u_Ω.

As in the corresponding context of parametric optimization (see section 5.3.4), the case of the compliance as a cost function is especially simple, as confirmed by the following proposition, whose proof unrolls along the lines of Proposition 5.1.

Proposition 5.2. *Assume that no body forces are applied to the structures under optimization: $f = 0$, and that the cost function $\mathcal{C}(\Omega)$ is the compliance, that is:*

$$\mathcal{C}(\Omega) = C(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\, dx = \int_{\Gamma_N} g.u_\Omega\, ds.$$

Then, for any shape $\Omega \in \mathcal{U}_{ad}$, the exact worst-case functional $\mathcal{J}$ reads:

$$\mathcal{J}(\Omega) = \sup_{\substack{V \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d) \\ ||V||_{L^\infty(\mathbb{R}^d)^d} \leq m}} C((I + \chi V)(\Omega)) = C((I - m\chi n_\Omega)(\Omega)).$$

Simply put, the most compliant shape among all the perturbed designs of a shape Ω according to (5.59) is the thinnest of all.

However, for a general objective function, the worst-case functional $\mathcal{J}(\Omega)$ is not explicit and we approximate it by $\widetilde{\mathcal{J}}(\Omega)$, defined as:

$$\forall \Omega \in \mathcal{U}_{ad},\ \widetilde{\mathcal{J}}(\Omega) = \sup_{\substack{V \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d) \\ ||V||_{L^\infty(\mathbb{R}^d)^d} \leq m}} (\mathcal{C}(\Omega) + \mathcal{C}'(\Omega)(\chi V)).$$

From now on, we assume that the data f, g are smooth enough, and that the sets $\mathcal{U}_{ad}$ of admissible shapes and Θ_{ad} of admissible variations of shapes incorporate enough smoothness in their definitions, so that all the state and adjoint functions u_Ω, p_Ω, q_Ω and z_Ω appearing in the forthcoming formulae are also smooth enough.

Before stating the results of interest, let us set some more notations. If $\Omega \in \mathcal{U}_{ad}$ is any shape, we denote as τ a local basis of tangent vectors to Γ, so that (τ, n) is a local orthonormal frame of $\mathbb{R}^d$. Any matrix $\mathcal{M} \in \mathcal{S}(\mathbb{R}^d)$ can be decomposed into this basis as:

$$\mathcal{M} = \begin{pmatrix} \mathcal{M}_{\tau\tau} & \mathcal{M}_{\tau n} \\ \mathcal{M}_{n\tau} & \mathcal{M}_{nn} \end{pmatrix}$$

where $\mathcal{M}_{\tau\tau}$ stands for the $(d-1) \times (d-1)$ tangential minor of $\mathcal{M}$, $\mathcal{M}_{\tau n}$ is the vector of the $(n-1)$ first tangential components of the normal column $\mathcal{M}n$, $\mathcal{M}_{n\tau}$ is the row vector of the $(n-1)$ first tangential components of the normal row $n^T\mathcal{M}$, and $\mathcal{M}_{nn} = \mathcal{M}n \cdot n$. We eventually denote as div_Γ the tangential divergence operator defined on Γ.

We start with a technical lemma concerning our example (5.64) of a cost function.

Lemma 5.4. *1. For any $\Omega \in \mathcal{U}_{ad}$, the shape derivative of the functional $J(\Omega)$, defined by (5.64), reads:*

$$\forall \theta \in \Theta_{ad},\ J'(\Omega)(\theta) = \int_\Gamma (j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega)\, (\theta \cdot n)\, ds,$$

where the adjoint state $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is defined as the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -\nabla_u j(u_\Omega) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ Ae(p)n & = & 0 & \text{on } \Gamma \cup \Gamma_N \end{array} \right. . \tag{5.66}$$

2. Let $\ell : \mathbb{R}^d_u \times \mathbb{R}^d_p \times \mathbb{R}_e \to \mathbb{R}$ be any smooth enough function which vanishes in a neighborhood of $\Gamma_D \cup \Gamma_N$, and define the functional $L(\Omega)$ as:

$$L(\Omega) = \int_\Gamma \ell(u_\Omega, p_\Omega, Ae(u_\Omega) : e(p_\Omega))\, ds, \tag{5.67}$$

where p_Ω is defined by system (5.66). Then L is shape differentiable, and its shape derivative reads:

$$\forall \theta \in \Theta_{ad},\ \ L'(\Omega)(\theta) = \int_\Gamma w(u_\Omega, p_\Omega, q_\Omega, z_\Omega)\ (\theta \cdot n)\ ds, \tag{5.68}$$

where we defined:

$$w(u_\Omega, p_\Omega, q_\Omega, z_\Omega) = \left(\frac{\partial}{\partial n} + \kappa\right)\left(\ell(u_\Omega, (\sigma(u_\Omega)_{\tau\tau} : (e(p_\Omega)_{\tau\tau}))\right) + Ae(p_\Omega) : e(q_\Omega) + Ae(u_\Omega) : e(z_\Omega) - f \cdot z_\Omega, \tag{5.69}$$

and the second an third adjoint states $q_\Omega, z_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ are respectively defined as the unique solutions to:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(q)) & = & 0 & \text{in } \Omega \\ q & = & 0 & \text{on } \Gamma_D \\ Ae(q)n & = & 0 & \text{on } \Gamma_N \\ Ae(q)n & = & -\nabla_p \ell(u_\Omega, p_\Omega, Ae(u_\Omega) : e(p_\Omega)) + \mathrm{div}_\Gamma\left(\frac{\partial \ell}{\partial e}(u_\Omega, p_\Omega, Ae(u_\Omega) : e(p_\Omega))(\sigma(u_\Omega))_{\tau\tau}\right) & \text{on } \Gamma \end{array}\right., \tag{5.70}$$

and:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(z)) & = & -\nabla^2_u j(u_\Omega) q_\Omega & \text{in } \Omega \\ z & = & 0 & \text{on } \Gamma_D \\ Ae(z)n & = & 0 & \text{on } \Gamma_N \\ Ae(z)n & = & -\nabla_u \ell(u_\Omega, p_\Omega, Ae(u_\Omega) : e(p_\Omega)) + \mathrm{div}_\Gamma\left(\frac{\partial \ell}{\partial e}(u_\Omega, p_\Omega, Ae(u_\Omega) : e(p_\Omega))(\sigma(p_\Omega))_{\tau\tau}\right) & \text{on } \Gamma \end{array}\right.. \tag{5.71}$$

Proof. (1): This is a very classical result in shape optimization (see e.g. [14]).

(2): Here, we need to assume that u_Ω and p_Ω enjoy more regularity than the sole 'natural' $H^1(\Omega)^d$ regularity, e.g $u_\Omega \in H^2(\Omega)^d \cap H^1_{\Gamma_D}(\Omega)^d$ and $p_\Omega \in H^2(\Omega)^d \cap H^1_{\Gamma_D}(\Omega)^d$, so that the very definition of $L(\Omega)$ makes sense. This is typically the case when the data Ω, f, g are assumed smooth enough (see the above assumptions).

As we are about to see, the problem of differentiating $L(\Omega)$ is not that simple. Indeed, if we attempt to carry out Cea's formal method as usual, we will get stuck by a problem of loss of regularity in the derived variational formulations for the adjoint states q_Ω and z_Ω, which feature traces on Γ of first order derivatives of test functions (which should enjoy only $H^1(\Omega)^d$ regularity).

We introduce the Lagrangian $\mathcal{L} : \mathcal{U}_{ad} \times \left(H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d\right)^4 \to \mathbb{R}$, defined by:

$$\begin{array}{rcl} \mathcal{L}(\Omega, \widehat{u}, \widehat{z}, \widehat{p}, \widehat{q}) & = & \displaystyle\int_\Gamma \ell(\widehat{u}, \widehat{p}, Ae(\widehat{u}) : e(\widehat{p}))\, ds + \int_\Omega Ae(\widehat{u}) : e(\widehat{z})\, dx - \int_\Omega f \cdot \widehat{z}\, dx - \int_{\Gamma_N} g \cdot \widehat{z}\, ds \\ & & \displaystyle + \int_\Omega Ae(\widehat{p}) : e(\widehat{q})\, dx - \int_\Omega -\nabla_u j(\widehat{u}) \cdot \widehat{q}\, dx \end{array}.$$

As usual, we look for the points $(u, z, p, q) \in \left(H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d\right)^4$ where the partial derivatives of $\mathcal{L}(\Omega, ., .)$ vanish, for a particular shape $\Omega \in \mathcal{U}_{ad}$.

• As before, canceling the partial derivative of $\mathcal{L}$ with respect to z at (Ω, u, z, p, q) imposes that u should satisfy:

$$\forall \widehat{z} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d,\ \ \int_\Omega Ae(u) : e(\widehat{z})\, dx = \int_\Omega f \cdot \widehat{z}\, dx + \int_{\Gamma_N} g \cdot \widehat{z}\, ds.$$

As $H^2(\mathbb{R}^d)$ is dense in $H^1(\mathbb{R}^d)$, this is equivalent to the fact that $u = u_\Omega$, the unique solution to (5.43).

• Similarly, canceling the derivative of $\mathcal{L}$ with respect to q at (Ω, u, z, p, q) imposes that p should satisfy:

$$\forall \widehat{q} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d, \ \int_\Omega Ae(p) : e(\widehat{q})\, dx = -\int_\Omega \nabla_u j(u_\Omega) \cdot \widehat{q}\, dx.$$

For the same reason, this implies that $p = p_\Omega$, the unique solution to (5.66).

• Let us now study the partial derivative of $\mathcal{L}$ with respect to p at (Ω, u, z, p, q). It reads, for all $\widehat{p} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d$:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p}(\Omega, u, z, p, q)(\widehat{p}) &= \int_\Gamma \left(\nabla_p \ell(u, p, Ae(u) : e(p)) \cdot \widehat{p} + \frac{\partial \ell}{\partial e}(u, p, Ae(u) : e(p)) Ae(u) : e(\widehat{p}) \right) ds \\ &\quad + \int_\Omega Ae(\widehat{p}) : e(q)\, dx \end{aligned} . \quad (5.72)$$

Under this form, this last expression does not lend itself to an unambiguous definition of q by means of a variational formulation over the space $H^1_{\Gamma_D}(\mathbb{R}^d)^d$, because the term

$$\widehat{p} \mapsto \int_\Gamma \left(\frac{\partial \ell}{\partial e}(u, p, Ae(u) : e(p)) Ae(u) : e(\widehat{p}) \right) ds$$

is not a continuous linear form over $H^1_{\Gamma_D}(\mathbb{R}^d)^d$. The trick consists in noticing that we already identified u as u_Ω. In particular, u complies with Neumann homogeneous boundary conditions $Ae(u)n = 0$ over Γ. This allows for a convenient simplification of the nasty term in (5.72):

$$\forall \widehat{p} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d, \ \ Ae(u) : e(\widehat{p}) = (Ae(u))_{\tau\tau} : e(\widehat{p})_{\tau\tau} \ \text{ a.e. on } \Gamma.$$

Using this information in (5.72) together with an integration by parts on Γ yields, for all $\widehat{p} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d$:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p}(\Omega, u, z, p, q)(\widehat{p}) &= \int_\Gamma \left(\nabla_p \ell(u, p, Ae(u) : e(p_\Omega)) \cdot \widehat{p} - \mathrm{div}_\Gamma \left(\frac{\partial \ell}{\partial e}(u, p, Ae(u) : e(p))(Ae(u))_{\tau\tau} \right) \cdot \widehat{p} \right) ds \\ &\quad + \int_\Omega Ae(\widehat{p}) : e(q)\, dx \end{aligned} .$$

Under the previous assumption that $u = u_\Omega$ is smooth enough, canceling this last expression yields a well-defined variational problem for q, which admits as unique solution in $H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d$ (owing to the regularity theory for linear elasticity, see [92]) $q = q_\Omega$, defined by (5.70).

• The study of the partial derivative of $\mathcal{L}$ with respect to u at (Ω, u, z, p, q) unrolls in the same way. It reads, for all $\widehat{u} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d$:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial u}(\Omega, u, z, p, q)(\widehat{u}) &= \int_\Gamma \left(\nabla_u \ell(u, p, Ae(u) : e(p)) \cdot \widehat{u} + \frac{\partial \ell}{\partial e}(u, p, Ae(u) : e(p))(Ae(p))_{\tau\tau} : e(\widehat{u})_{\tau\tau} \right) ds \\ &\quad + \int_\Omega Ae(\widehat{u}) : e(z)\, dx + \int_\Omega \left(\nabla^2_u j(u) q \right) \cdot \widehat{u}\, dx \\ &= \int_\Gamma \left(\nabla_u \ell(u, p, Ae(u) : e(p)) \cdot \widehat{u} - \mathrm{div}_\Gamma \left(\frac{\partial \ell}{\partial e}(u, p, Ae(u) : e(p))(Ae(p))_{\tau\tau} \right) \cdot \widehat{u} \right) ds \\ &\quad + \int_\Omega Ae(\widehat{u}) : e(z)\, dx + \int_\Omega \left(\nabla^2_u j(u) q \right) \cdot \widehat{u}\, dx \end{aligned} ,$$

where the third line stems again from integration by parts on Γ, with $\ell \equiv 0$ on $\partial\Gamma$. Under the previous assumption that $u = u_\Omega$ is smooth enough, canceling this last expression yields a well-defined variational problem for z, which admits $z = z_\Omega$ as unique solution in $H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d$, defined by (5.71).

Eventually, for any domain $\Omega \in \mathcal{U}_{ad}$, and any fixed functions $\widehat{q}, \widehat{z} \in H^2(\mathbb{R}^d)^d \cap H^1_{\Gamma_D}(\mathbb{R}^d)^d$, one has:

$$L(\Omega) = \mathcal{L}(\Omega, u_\Omega, \widehat{z}, p_\Omega, \widehat{q}),$$

whence, differentiating this expression with respect to Ω, and evaluating at $\widehat{q} = q_\Omega$ and $\widehat{z} = z_\Omega$,

$$\forall \theta \in \Theta_{ad},\ \ L'(\Omega)(\theta) = \frac{\partial \mathcal{L}}{\partial \Omega}(\Omega, u_\Omega, z_\Omega, p_\Omega, q_\Omega)(\theta),$$

and the desired formula (5.68) follows. □

Using this technical result, we are now in position to prove the following theorem:

Theorem 5.9. *Consider the worst-case design functional $\widetilde{\mathcal{J}}(\Omega)$, defined as:*

$$\forall \Omega \in \mathcal{U}_{ad},\ \ \widetilde{\mathcal{J}}(\Omega) = J(\Omega) + \sup_{\substack{V \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \\ ||V||_{L^\infty(\mathbb{R}^d)^d} \leq m}} J'(\Omega)(\chi V).$$

Then $\widetilde{\mathcal{J}}$ rewrites:

$$\widetilde{\mathcal{J}}(\Omega) = \int_\Omega j(u_\Omega)\, dx + m \int_\Gamma \chi\, |j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega|\ ds, \tag{5.73}$$

where the adjoint state $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is defined as the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -\nabla_u j(u_\Omega) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ Ae(p)n & = & 0 & \text{on } \Gamma \cup \Gamma_N \end{array} \right. . \tag{5.74}$$

Furthermore, $\widetilde{\mathcal{J}}$ is shape differentiable at any $\Omega \in \mathcal{U}_{ad}$ such that the set

$$E_\Omega := \{x \in \Gamma,\ \ (j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega)\,(x) = 0\}$$

is of null (surface) Lebesgue measure. At such a point, its shape derivative reads:

$$\begin{array}{rcl} \forall \theta \in \Theta_{ad},\ \ \widetilde{\mathcal{J}}'(\Omega)(\theta) & = & \displaystyle\int_\Gamma \chi\, (j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega)\, (\theta \cdot n)\ ds \\ & & \displaystyle + m \int_\Gamma \left(\left(\frac{\partial}{\partial n} + \kappa \right) (\chi\, |j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega|) \right)\ (\theta \cdot n)\ ds\ , \\ & & \displaystyle + m \int_\Gamma (Ae(p_\Omega) : e(q_\Omega) + Ae(u_\Omega) : e(z_\Omega) - f \cdot z_\Omega)\ (\theta \cdot n)\ ds \end{array} \tag{5.75}$$

where the second and third adjoint states q_Ω, z_Ω are respectively defined as the unique solutions in $H^1_{\Gamma_D}(\Omega)^d$ to the systems:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(q)) & = & 0 & \text{in } \Omega \\ q & = & 0 & \text{on } \Gamma_D \\ Ae(q)n & = & 0 & \text{on } \Gamma_N \\ Ae(q)n & = & \varepsilon\chi\, f + \mathrm{div}_\Gamma\, (\varepsilon\chi\, (\sigma(u_\Omega))_{\tau\tau}) & \text{on } \Gamma \end{array} \right. , \tag{5.76}$$

and:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(z)) & = & -\nabla_u^2 j(u_\Omega) q_\Omega & \text{in } \Omega \\ z & = & 0 & \text{on } \Gamma_D \\ Ae(z)n & = & 0 & \text{on } \Gamma_N \\ Ae(z)n & = & -\varepsilon\chi\, \nabla_u j(u_\Omega) + \mathrm{div}_\Gamma\, (\varepsilon\chi\, (\sigma(p_\Omega))_{\tau\tau}) & \text{on } \Gamma \end{array} \right. , \tag{5.77}$$

where $\varepsilon := \mathrm{sgn}\,(j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega)$.

Proof. First, using lemma (5.4) (i), we know that J is shape differentiable at any $\Omega \in \mathcal{U}_{ad}$, with shape derivative:

$$\forall \theta \in \Theta_{ad},\ \ J'(\Omega)(\theta) = \int_\Gamma \left(j(u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega\right)(\theta \cdot n)\, ds.$$

Using this expression together with theorem 5.11 readily gives rise to formula (5.73).

Then, using Lemma 5.4, (3), with $\ell(u,p,e) = \chi\,|j(u) + e - f \cdot p|$ produces formula (5.75). □

We conclude this tour with the study of the stress-based cost function $S(\Omega)$ defined by (5.65). The following result is proved in the exact same way as Theorem 5.9, and the proof is omitted:

Theorem 5.10. *Let $\widetilde{\mathcal{J}}(\Omega)$ the worst-case design functional defined as:*

$$\forall \Omega \in \mathcal{U}_{ad},\ \ \widetilde{\mathcal{J}}(\Omega) = S(\Omega) + \sup_{\substack{V \in W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d) \\ ||V||_{L^\infty(\mathbb{R}^d)^d} \leq m}} S'(\Omega)(\chi V).$$

Then $\widetilde{\mathcal{J}}$ rewrites:

$$\widetilde{\mathcal{J}}(\Omega) = \int_\Omega j(\sigma(u_\Omega))\, ds + m \int_\Gamma \chi\ \left|j(\sigma(u_\Omega)) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega\right|\ ds, \tag{5.78}$$

where the adjoint state $p_\Omega \in H^1_{\Gamma_D}(\Omega)^d$ is defined as the unique solution to:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & \mathrm{div}(A\frac{\partial j}{\partial \sigma}(\sigma(u_\Omega))) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ Ae(p)n & = & -A\frac{\partial j}{\partial \sigma}(\sigma(u_\Omega))n & \text{on } \Gamma \cup \Gamma_N \end{array}\right. . \tag{5.79}$$

Besides, $\widetilde{\mathcal{J}}$ is shape differentiable at any $\Omega \in \mathcal{U}_{ad}$ such that the set

$$E_\Omega := \{x \in \Gamma,\ \ (j(\sigma(u_\Omega)) + Ae(u_\Omega) : e(p_\Omega) - f.p_\Omega)\,(x) = 0\}$$

is of zero (surface) Lebesgue measure. At such a point, its shape derivative reads, for all $\theta \in \Theta_{ad}$:

$$\begin{array}{rcl} \forall \theta \in \Theta_{ad},\ \ \widetilde{\mathcal{J}}'(\Omega)(\theta) & = & \displaystyle\int_\Gamma \chi\,(j(\sigma(u_\Omega)) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega)\,(\theta \cdot n)\ ds \\ & & \displaystyle + m\int_\Gamma \left(\left(\frac{\partial}{\partial n} + \kappa\right)(\chi\,|j(\sigma(u_\Omega)) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega|)\right)\ (\theta \cdot n)\ ds\ , \\ & & \displaystyle + m\int_\Gamma (Ae(p_\Omega) : e(q_\Omega) + Ae(u_\Omega) : e(z_\Omega) - f \cdot z_\Omega)\ (\theta \cdot n)\ ds \end{array} \tag{5.80}$$

where the second and third adjoint states q_Ω, z_Ω are respectively defined as the unique solutions in $H^1_{\Gamma_D}(\Omega)^d$ to the following variational problems:

$$\forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \ \int_\Omega Ae(q) : e(v)\, dx = -\int_\Gamma \varepsilon\chi\,((\sigma(u_\Omega))_{\tau\tau} : e(v)_{\tau\tau} - f \cdot v)\ ds, \tag{5.81}$$

and:

$$\begin{array}{rcl} \forall v \in H^1_{\Gamma_D}(\Omega)^d,\ \ \displaystyle\int_\Omega Ae(z) : e(v)\, dx & = & \displaystyle -\int_\Gamma \varepsilon\chi\left(\left(\left(A\frac{\partial j}{\partial \sigma}(\sigma(u_\Omega))\right)_{\tau\tau}\right) : e(v)_{\tau\tau} + (\sigma(p_\Omega))_{\tau\tau} : e(v)_{\tau\tau}\right)\ ds, \\ & & \displaystyle -\int_\Omega \left(\frac{\partial^2 j}{\partial \sigma^2}(\sigma(u_\Omega))Ae(q_\Omega)\right) : Ae(v)\, dx \end{array} \tag{5.82}$$

where $\varepsilon := \mathrm{sgn}\,(j(\sigma(u_\Omega)) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega)$.

5.5 Numerical results

5.5.1 Worst-case optimization problems in parametric structural optimization

Let us start with the parametric structural optimization setting, and test the derivatives computed in Section 5.3. We reuse the notations introduced then: in every case, a cost functional $\mathcal{C}$ of the thickness (and of perturbation parameters) is considered, and the corresponding approximate worst-case functional $\widetilde{\mathcal{J}}$ is minimized using either a steepest-descent algorithm, or an augmented Lagrangian algorithm (see [237], §17.4 or chapter 4 sec. 4.7 for a short description).

In both examples below, the imposed bounds over admissible thickness functions are $h_{min} = 0.1$ and $h_{max} = 1$, and the initial design of the plate is described by a uniform thickness $h = 0.5$. The elastic material filling the plate is characterized by its (normalized) Young's modulus and Poisson ration ν given by:

$$E = 1, \quad \nu = 0.3. \tag{5.83}$$

Both examples are performed within the `FreeFem++` environment [259].

5.5.1.1 Uncertainties around the applied body forces in parametric optimization

This first example illustrates the results of Section 5.3.2, and more accurately those of Example 5.3. The situation is as depicted in Figure 5.3: the plate is clamped on its bottom-left and bottom-right sides, and its cost, when its thickness is h and when submitted to body forces f is its compliance:

$$\mathcal{C}(h, f) = \int_\Omega f \cdot u_{h,f}\, dx,$$

(no surface loads are applied). The plate is equipped with a triangular computational mesh, which is worth 10128 vertices (thus, twice as many triangles). The unperturbed state is associated to the following distribution of forces: $f = (0, -1)$ near the centre of the bottom side of the plate (red spot on Figure 5.3, top), and $f = (0, 0)$ elsewhere. Vertical perturbations $(0, \xi) \in L^2(\Omega)^2$ of maximum amplitude $||\xi||_{L^2(\Omega)^2} \leq m$ are expected, which are located on the bottom side on the plate, between the regions where it is clamped, and that where body forces are applied on the unperturbed shape (grey areas on Figure 5.3, top).

The approximate worst-case functional $\widetilde{\mathcal{J}}$ defined by (5.30) is considered for minimization, and so that the problem is not trivial, a volume constraint is added, using a fixed Lagrange multiplier $\ell = 5.10^{-4}$. The considered minimization problem thus becomes:

$$\min_{h \in \mathcal{U}_{ad}} \left(\widetilde{\mathcal{J}}(h) + \ell \int_\Omega h\, dx \right).$$

For increasing values of m, 100 iterations of a gradient-based steepest descent algorithm are performed and the resulting shapes and convergence histories are reported in Figures 5.3 and 5.4.

Predictably, this simple setting does not really allow to compare the obtained shapes with one another: since the Lagrange multiplier used to enforce a volume constraint is always the same regardless of the value of m (which actually acts as a penalization parameter as we have discussed), it does not always expresses the same volume constraint, and shapes show a trend towards thickening as the amplitude of perturbation grows.

Nevertheless, the results show interesting changes in behaviors as m increases. To better capture this phenomenon, we turn to a more 'realistic' context, where a volume constraint

$$\mathrm{Vol}(h) := \int_\Omega h\, dx = V_T,$$

is enforced owing to an augmented Lagrangian method in the course of minimizing $\widetilde{\mathcal{J}}$. The same test case is run with a target volume $V_T = 0.7$; 150 iterations prove necessary to achieve convergence of the algorithm,

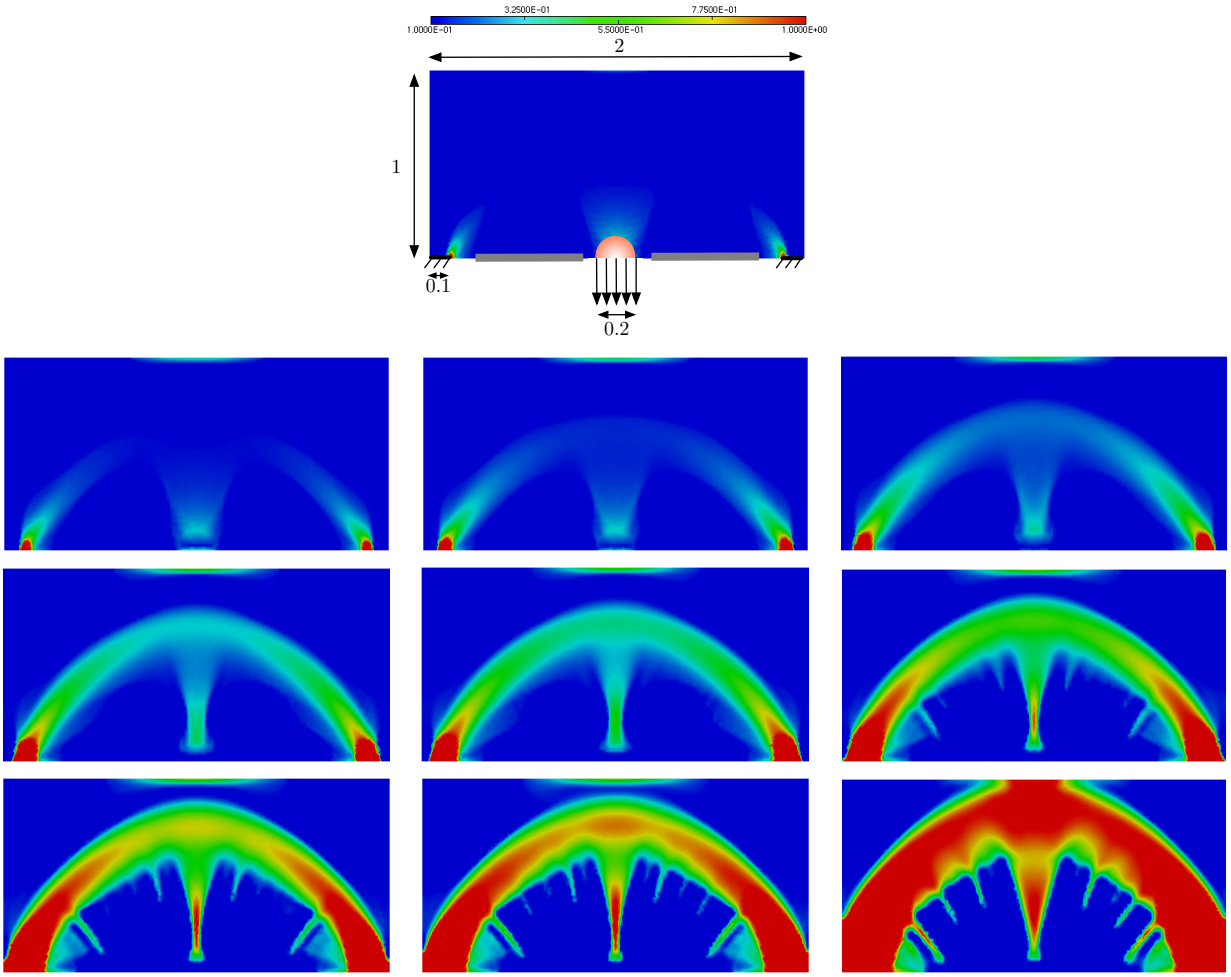


Figure 5.3: Minimization of the compliance with uncertainties over the body forces; from left to right, top to bottom, $m = 0$ (with boundary conditions), $0.05, 0.1, 0.2, 0.35, 0.5, 1, 1.5, 2, 5$.

and each computation takes about $6 - 7$ minutes (except for the one associated to the value $m = 0$, which does not involve any computation of an adjoint state). Results and convergence histories are reported on Figures 5.5 and 5.6.

5.5.1.2 Geometric uncertainties in parametric optimization

We now illustrate the proposed model for dealing with geometric uncertainty in parametric optimization of section 5.3.4. Recall that, in this context, the case of the compliance as a cost function is almost trivial, as assessed by Proposition 5.1.

Hence, let us consider another example, depicted in Figure 5.7: the considered plate is fixed on a part $\Gamma_D \subset \partial\Omega$, and submitted to surface loads $g \in L^2(\Gamma_N)^2$ on another part $\Gamma_N \subset \partial\Omega$. g equals $(0, -1)$ on the upper part of Γ_N, and $(0, 1)$ on its lower part. The plate is endowed with a triangular mesh of 12382 vertices.

In this case, we chose as cost function $\mathcal{C}(h)$ a least-square discrepancy criterion between the solution u_h

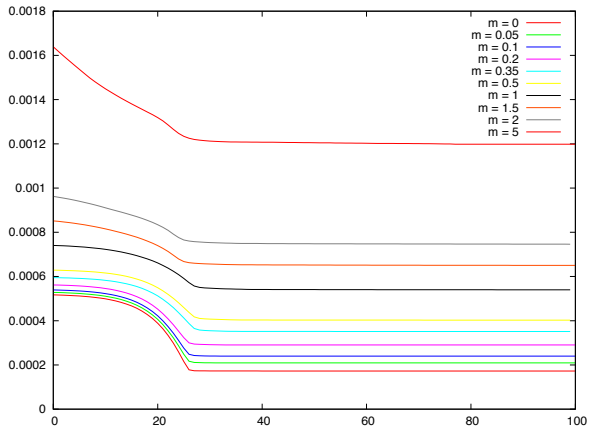


Figure 5.4: Convergence histories for the problem of compliance minimization under uncertainties over the body forces, using the same Lagrange multiplier for all examples.

to (5.5), and a target displacement u_0, that is:

$$\forall h \in \mathcal{U}_{ad},\ \mathcal{C}(h) = \int_{\Gamma_T} |u_h - u_0|^2\, ds,$$

where Γ_T is another non optimizable subset of $\partial\Omega$, disjoint from Γ_D and Γ_N. We chose $u_0 = (0, -1)$ on the upper part of Γ_T, and $u_0 = (0, 1)$ on its lower part.

To help the optimization algorithm of the associated approximate worst-case functional $\widetilde{\mathcal{J}}$ to reach a connected optimal shape, a very small volume constraint is imposed by using a fixed Lagrange multiplier $\ell = 0.0003$, which is a mere numerical token (the cost criterion does not vary monotonically with the volume of the structure in this case).

For increasing values of m, 100 iterations of a steepest-descent algorithm based on the conclusions of Theorem 5.5 are performed, and the results are depicted in Figures 5.7 and 5.8; each computation takes about 10 minutes (except for the one associated to $m = 0$, which only involves the computation of one adjoint state at each iteration of the process, whereas the others involve three). As in the previous test-case, one observes that the performances of the obtained shapes in their unperturbed states worsen as m grows (which is coherent, since the larger the value of m, the lower the importance of this unperturbed problem in the balance expressed by $\widetilde{\mathcal{J}}$ between perturbed and unperturbed states). An interesting 'topological' change in trends among the optimal shapes is also to be noted at $m \approx 0.0365$.

5.5.2 Examples of shape optimization problems under uncertainties

5.5.2.1 Details around the numerical implementation

As far as numerical simulations are concerned, shape optimization of elastic structures differs from its parametric counterpart mainly regarding the difficulty to account for the evolutions of shapes during the process. To deal with this issue, we rely on the level set method, as was originally suggested in [14, 319] (see chapter 1 for a description), which roughly speaking consists in describing every shape $\Omega \subset \mathbb{R}^d$ by means of a scalar function $\phi : \mathbb{R}^d \to \mathbb{R}$ enjoying the properties:

$$\forall x \in \mathbb{R}^d,\ \begin{cases} \phi(x) < 0 & \text{if } x \in \Omega \\ \phi(x) = 0 & \text{if } x \in \partial\Omega \\ \phi(x) > 0 & \text{if } x \in {}^c\overline{\Omega} \end{cases}.$$

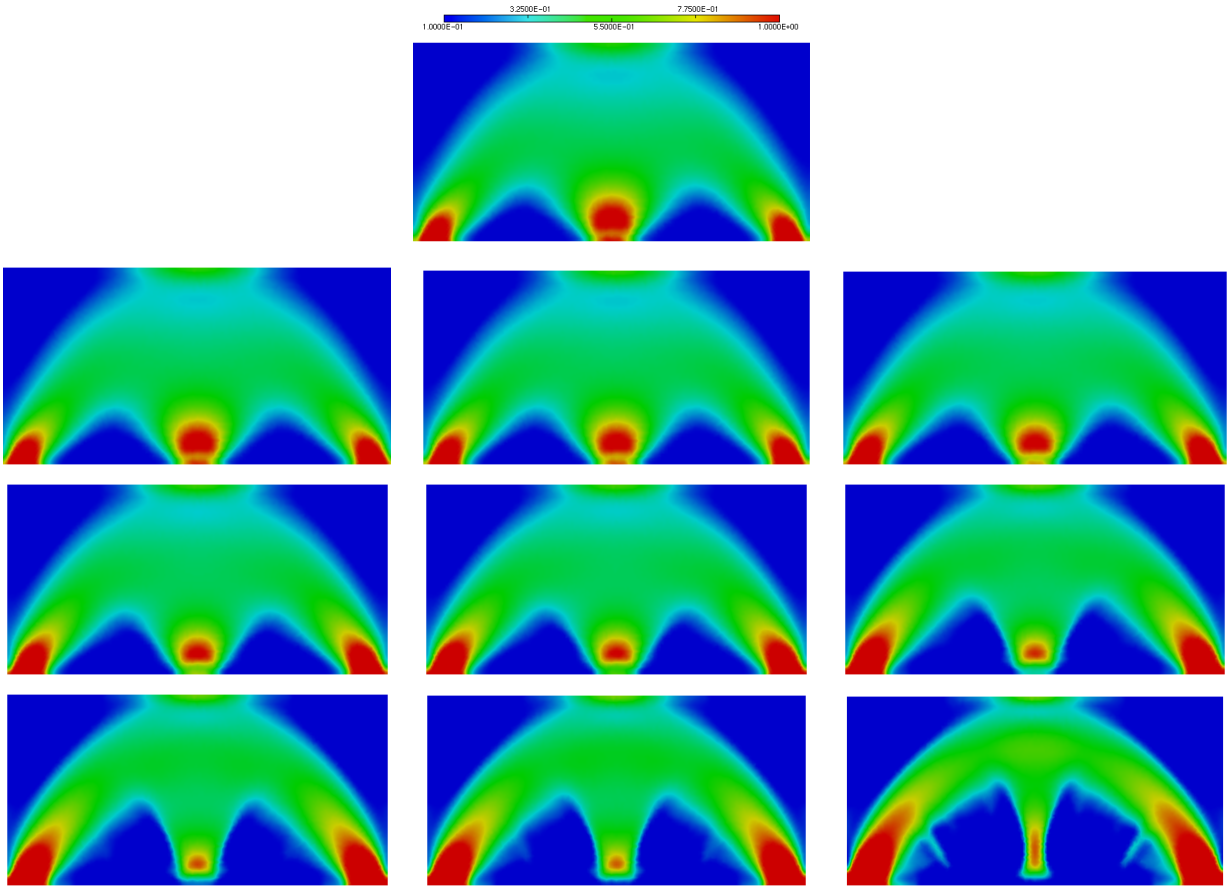


Figure 5.5: Minimization of the compliance with uncertainties over the body forces, with an imposed volume $V_T = 0.7$; from left to right, top to bottom, $m = 0, 0.05, 0.1, 0.2, 0.35, 0.5, 1, 1.5, 2, 5$, with a target volume $V_T = 0.7$.

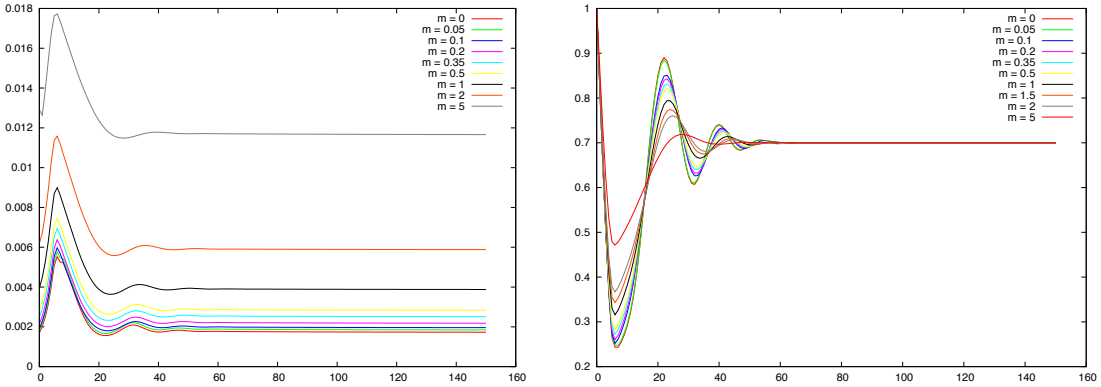


Figure 5.6: Convergence histories for the approximate worst-case compliance (left) and for the volume (right) of the plate when uncertainties over body forces are considered.

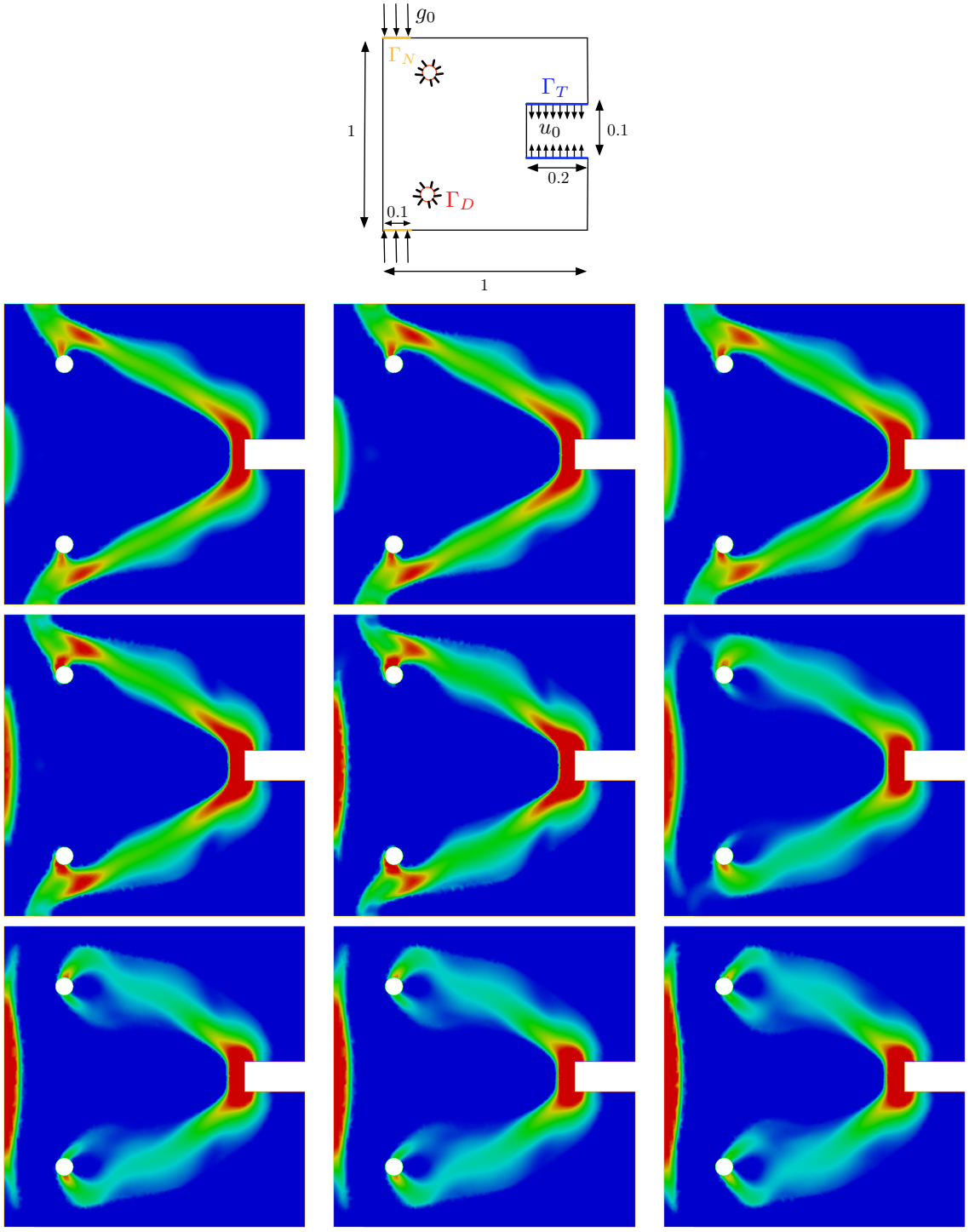


Figure 5.7: Parametric optimization of a plate under uncertainties over its thickness. From left to right, top to bottom, details of the test case, and obtained shapes for $m = 0, 0.01, 0.02, 0.03, 0.036, 0.0365, 0.038, 0.04, 0.05$.

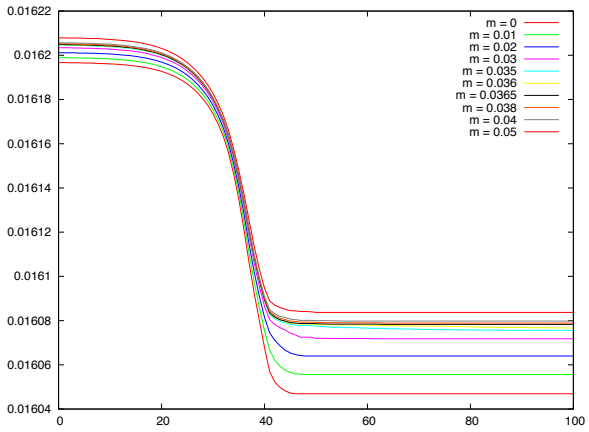


Figure 5.8: Convergence histories for the parametric optimization example under uncertainties over the thickness.

The main asset of this change in perspectives lies in that the motion of a domain $\Omega(t)$, $t \in [0,T]$ evolving in time, driven by a normal velocity field $V(t,x)\, n_{\Omega(t)}(x)$ is translated in terms of a corresponding level set function $\phi(t,.)$ into the following Hamilton-Jacobi equation:

$$\frac{\partial \phi}{\partial t} + V|\nabla \phi| = 0 \text{ on } [0,T] \times \mathbb{R}^d. \tag{5.84}$$

In the situation of Section 5.4 (whose notations are reused here), the minimization of a functional $\widetilde{\mathcal{J}}(\Omega)$ of the domain is considered, whose shape derivative is of the form:

$$\forall \theta \in \Theta_{ad},\ \ \widetilde{\mathcal{J}}'(\Omega)(\theta) = \int_\Gamma v(u_\Omega, p_\Omega, q_\Omega, z_\Omega)\, (\theta \cdot n)\, ds,$$

for some algebraic combination $v(u_\Omega, p_\Omega, q_\Omega, z_\Omega)$ of solutions to (state and adjoints) elasticity systems posed on Ω. V is then set to 0 on $\Gamma_D \cup \Gamma_N$ and to $-v(u_\Omega, p_\Omega, q_\Omega, z_\Omega)$ on the free boundary Γ.

In numerical practice, the whole space $\mathbb{R}^d$ is reduced to a large working domain $D \subset \mathbb{R}^d$, which encloses all the considered shapes, and comes equipped with a *fixed* simplicial mesh $\mathcal{T}$. The (state or adjoints) linear elasticity systems posed on a given shape Ω, involved in the expressions of V, cannot be computed exactly since Ω is only known by means of an associated level set function (i.e. no mesh of Ω is available). The Ersatz material approach [14] is then used to transfer a linear elasticity system posed on Ω to one posed on D by filling the void $D \backslash \Omega$ with a very soft material with Hooke's law εA, $\varepsilon \ll 1$ ($\varepsilon = 1e^{-3}$ in our examples).

All the numerical operations in the sequel are performed using the `FreeFem++` package [259], except for the routines for solving (5.84) and *redistancing* ϕ, which come from the works described in chapters 6 and 7. For the sake of completeness, the computational times of two representative computations are provided, in Sections 5.5.2.2 (whose model involves one adjoint state) and 5.5.2.4 (whose model involves three adjoint states).

5.5.2.2 Shape optimization under uncertainties about the applied loads

Let us start by illustrating the conclusions of Section 5.4.2, and more accurately of Example 5.6. The cost $\mathcal{C}(\Omega, f)$ of a shape $\Omega \subset \mathbb{R}^d$, when submitted to body forces $f \in L^2(\Omega)^d$ and traction loads $g \in L^2(\Gamma_N)^d$ is its compliance:

$$\mathcal{C}(\Omega, f) = \int_\Omega f \cdot u_{\Omega,f}\, dx + \int_{\Gamma_N} g \cdot u_{\Omega,f}\, ds.$$

We first consider the situation depicted in Figure 5.9: a mast is clamped on a part Γ_D of its boundary and traction loads $g = (0, -1)$ are applied on Γ_N, near the bottom-left and bottom-right parts of its arms. In the unperturbed state, no body forces are applied ($f = 0$). Perturbations are expected as vertical body forces $(0, \xi)$, of amplitude $||\xi||_{L^2(\mathbb{R}^d)} \leq m$, which are located on near the bottom of the arms of the mast (blue areas in Figure 5.9).

We first minimize the corresponding worst-case scenario functional $\widetilde{\mathcal{J}}$ with respect to the shape for different values of parameter m, using a fixed Lagrange multiplier $\ell = 1$ to impose a volume constraint. 200 iterations of a gradient algorithm are used, and results are displayed on Figure 5.9.

One observes that, once again, the shapes tend to thicken as m grows, but also notices interesting changes in trends in the layout of the structure. Once again, to better appraise this feature, we run the very same example, using an augmented Lagrangian method to enforce a volume constraint $\mathrm{Vol}(\Omega) = V_T$, where V_T is a target volume (in this example, $V_T = 2000$). Each computation (except for the one associated to $m = 0$) takes about 25 minutes, for a computational mesh composed of 11257 vertices. The results are reported on Figure 5.10, and confirm our initial guess (see also Figure 5.11 for convergence histories).

The exact same sequence of operations is applied on another model, namely the benchmark *optimal bridge* test case, as described in Figure 5.12 (top): a bridge is clamped on two sides of its boundary, and vertical body forces $f = (0, -10)$ are applied at the middle of the bottom of the structure (yellow box). Vertical perturbations of amplitude lower than m are expected to occur on the blue areas. First, a minimization procedure is carried out, using a fixed Lagrange multiplier $\ell = 0.2$ for the volume constraint, for several values of m, and results are to be seen on Figure 5.12. The results of the subsequent step, to get optimal shapes with the same target volume $V_T = 0.75$ are displayed on Figure 5.13

Remark 5.9. Some of the 'optimal' shapes displayed turn out to be non symmetric, whereas the setting of the corresponding test case is. This is mainly because no particular attention has been paid about this feature; in particular, the meshes of the (symmetric) bounding boxes are triangular, and not symmetric.

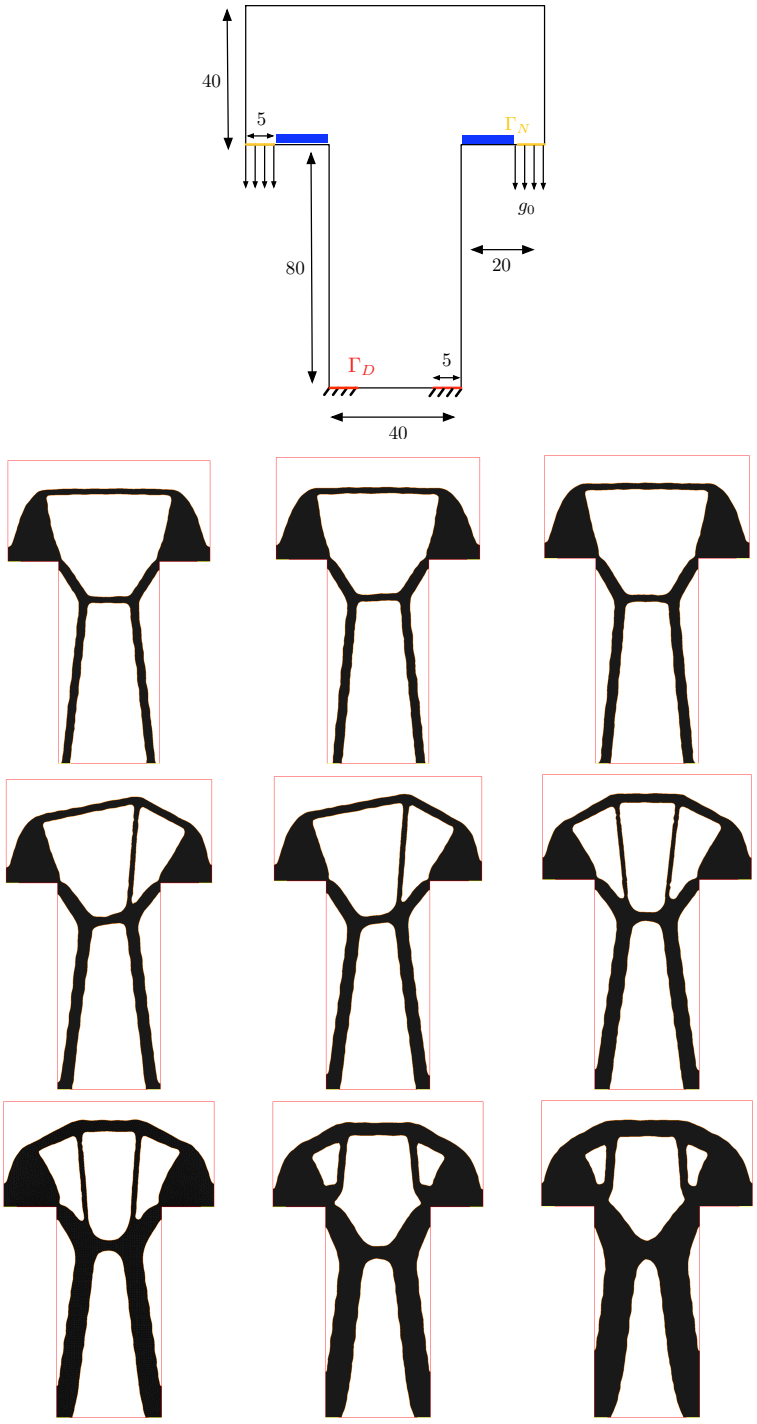


Figure 5.9: *(From left to right, top to bottom):* Setting of the test case, and optimal shapes of a mast for compliance minimization, with uncertainties on the body forces of amplitude $m = 0,\ 0.1, 0.25, 0.4, 0.6, 1, 2, 3, 5$. The same Lagrange multiplier for the volume constraint is used in all cases.

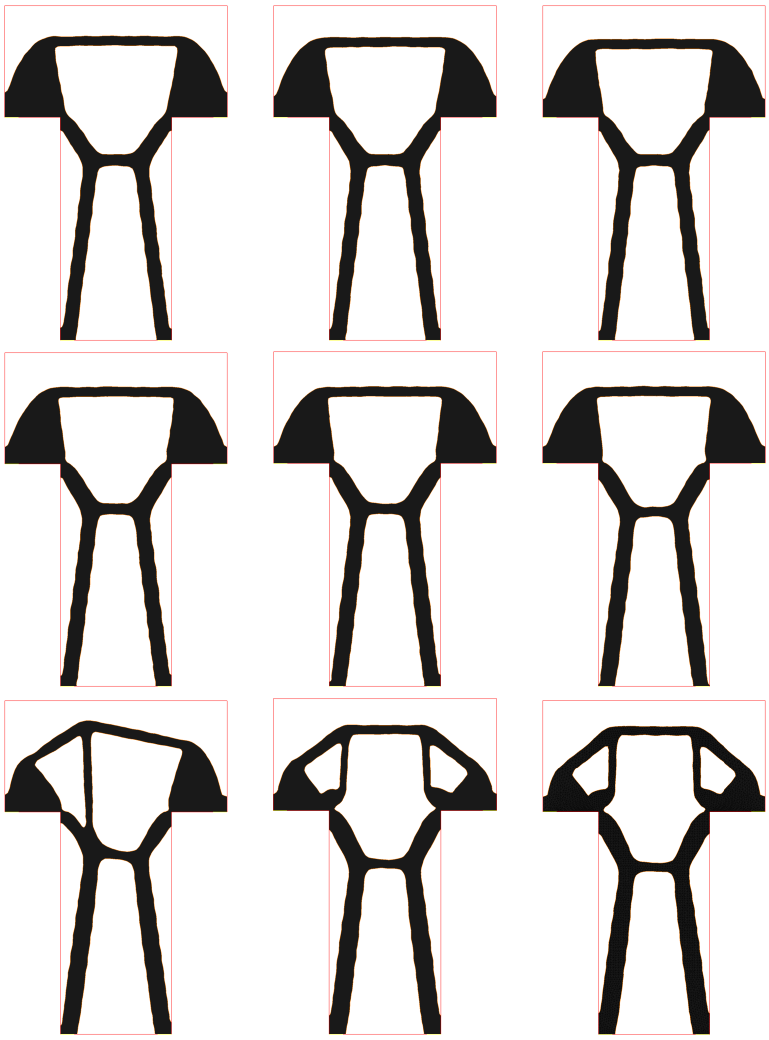


Figure 5.10: *(From left to right, top to bottom):* optimal shapes of a mast for compliance minimization, under uncertainties on the body forces of amplitude $m = 0,\ 0.1, 0.25, 0.4, 0.6, 1, 2, 3, 5$; all the shapes have the some volume $V_T = 2000$.

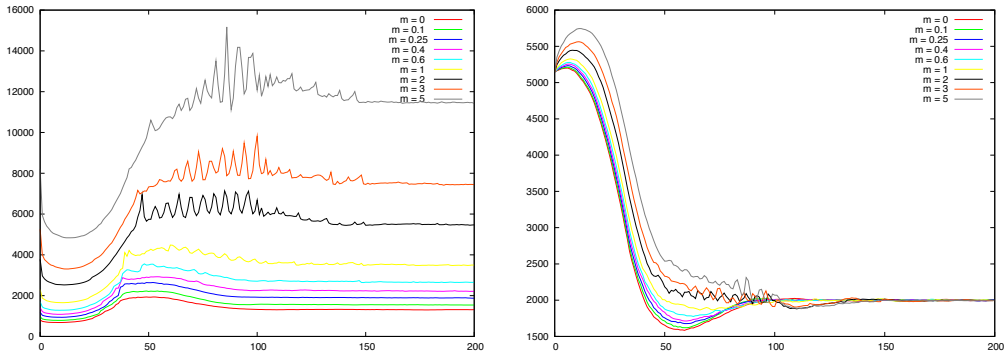


Figure 5.11: Convergence history for the approximate worst-case compliance (left) and for the volume (right) in the (worst-case) optimal mast test case.

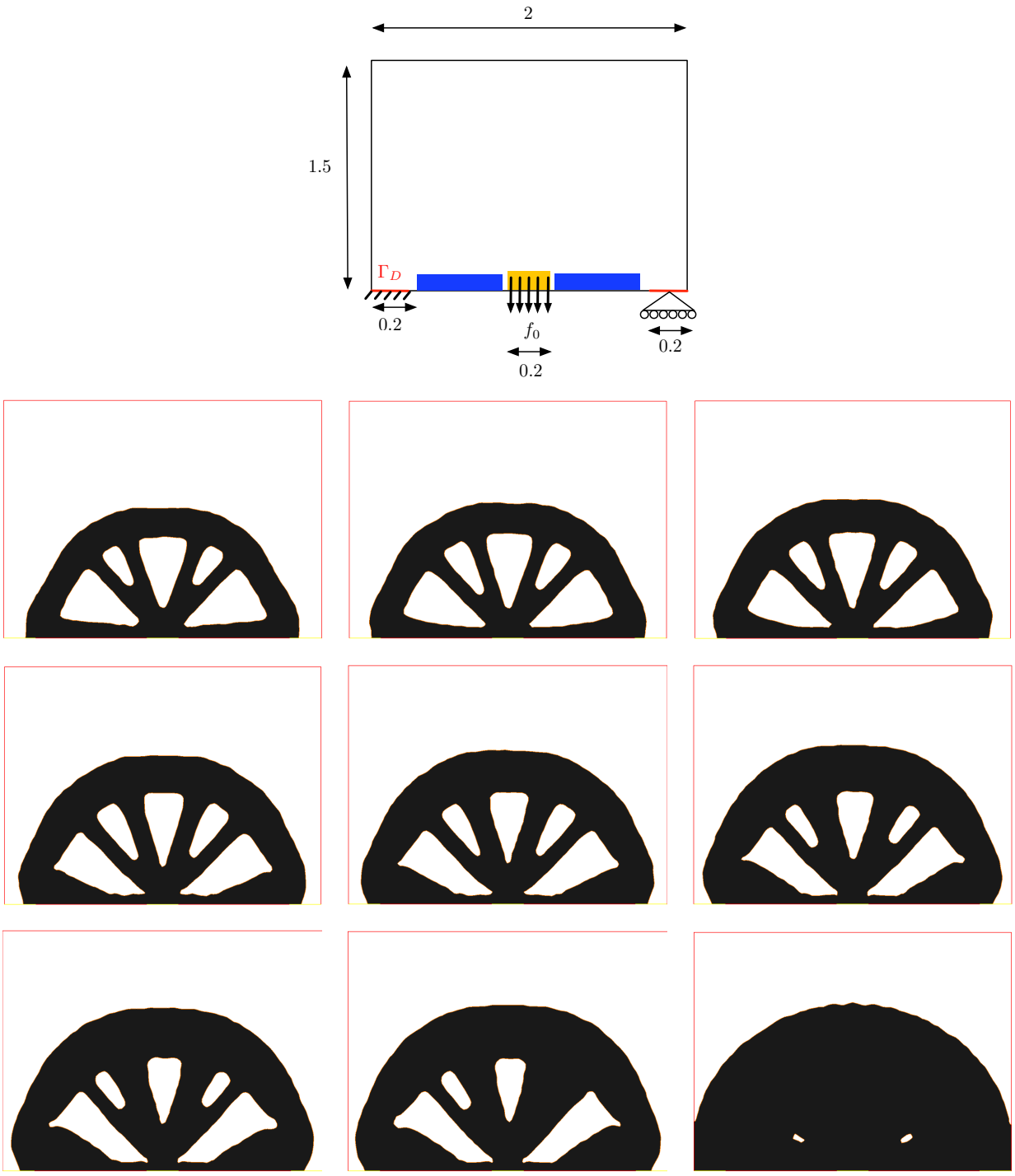


Figure 5.12: *(From left to right, top to bottom):* Optimal shape of a bridge under perturbations over the body forces of amplitude $m = 0,\ 0.1, 0.2, 0.5, 0.7, 1, 1.2, 1.5, 2$. The same Lagrange multiplier for the volume constraint is used in all cases.

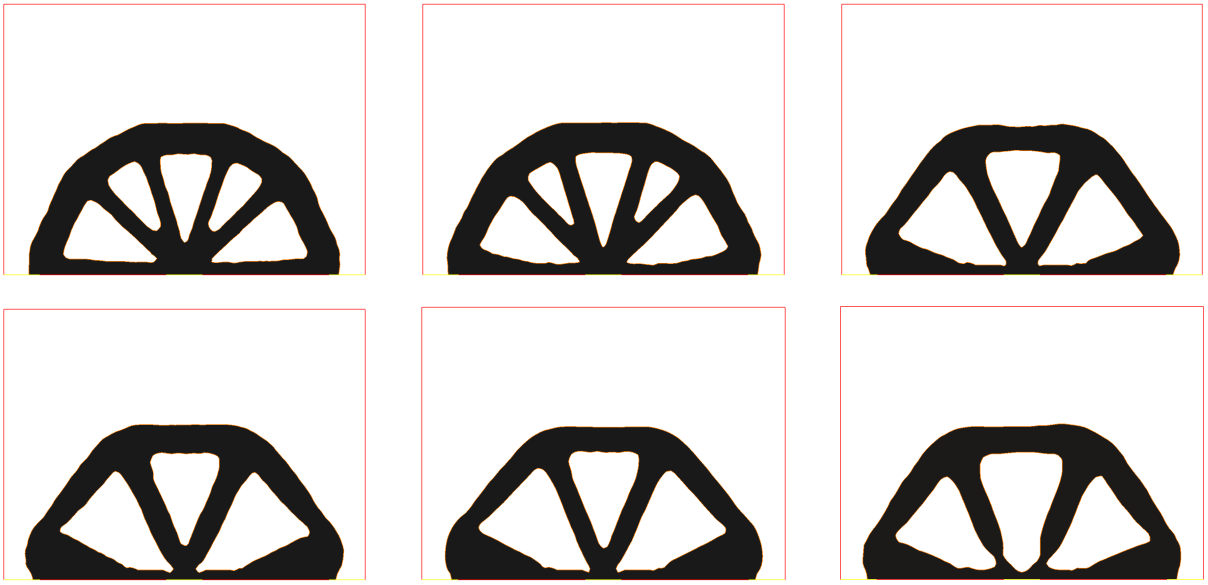


Figure 5.13: *(From left to right, top to bottom):* optimal shape for the worst-case optimal bridge example, for $m = 0,\ 0.2, 0.5, 1, 1.5, 2$. The same volume constraint $V = V_T = 0.75$ is imposed in all six cases.

5.5.2.3 Shape optimization under uncertainties on the material's properties

The proposed approach is evaluated in the context of Section 5.4.3; we seek the optimal design of a force inverter: the considered shapes are clamped on the upper and lower parts of their left-hand side, a surface load $g = (-0.1, 0)$ is applied at the centre of this left-hand side, and should exhibit a prescribed displacement $u_0 = (1, 0)$ in a (non optimizable) area located at the centre of their right-hand side (see the details on Figure 5.14).

In this context, the cost of a shape $\Omega \in \mathcal{U}_{ad}$, when filled with a material with Lamé coefficients λ, μ is:

$$\mathcal{C}(\Omega, \lambda, \mu) = \int_\Omega k(x)|u_{\Omega,\lambda,\mu} - u_0|^2 \, dx,$$

where k is the characteristic function of the area where the target displacement should be reached.

We are in search of a shape $\Omega \in \mathcal{U}_{ad}$ which minimizes this cost, when perturbations $|\alpha| \leq m$, $|\beta| \leq m$ are expected over the 'reference' Lamé coefficients λ, μ associated to (5.83), and this leads us to consider the functional $\widetilde{\mathcal{J}}$ of formula (5.56).

As was the case in the example of section 5.5.1.2, the performance of a shape has nothing to do with its weight. Nevertheless, for purely numerical purposes, we add a very small penalization, with respective parameters $\ell_v = 5.e^{-3}$ and $\ell_c = 0.02$ on the volume and compliance of the shapes to the expression of $\widetilde{\mathcal{J}}$. The first additional term helps in removing the small 'islands' (i.e. disconnected parts obtained after topological changes occurred), while the second one makes it easier to obtained a connected structure (which is difficult, since shapes tend to develop very small parts in the course of the process in order to gain flexibility).

Figure 5.14 shows the shapes obtained after 400 iterations of a gradient algorithm, for several values of m, and Figure 5.15 displays the corresponding displaced shapes. The convergence histories for these computations are reported on Figure 5.16.

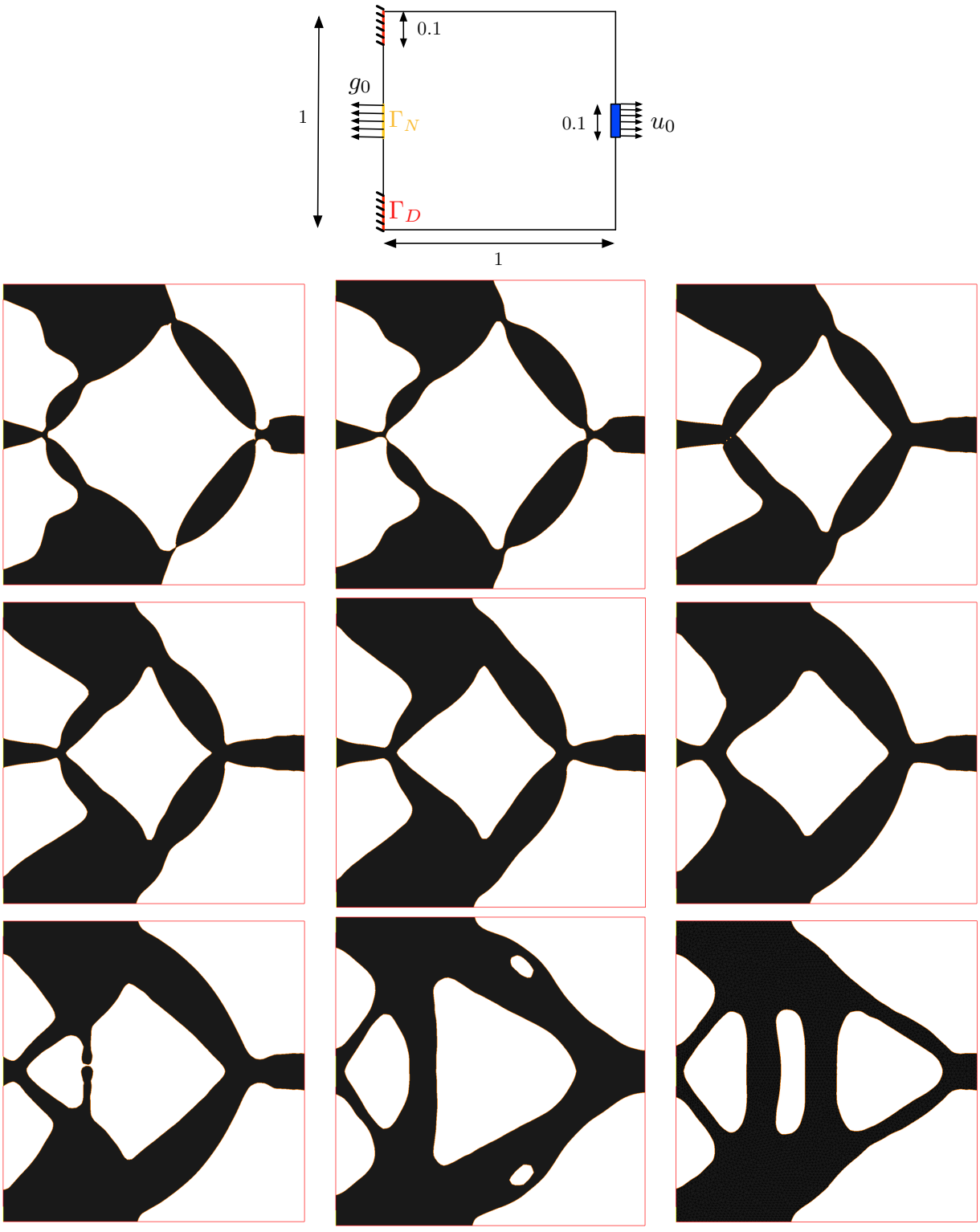


Figure 5.14: *(From left to right, top to bottom):* details of the test-case, optimal shape for the worst case force inverter test case, with perturbations over the Lamé coefficients of the material of magnitude $m = 0, 0.001, 0.002, 0.003, 0.0045, 0.0075, 0.01, 0.02, 0.1$.

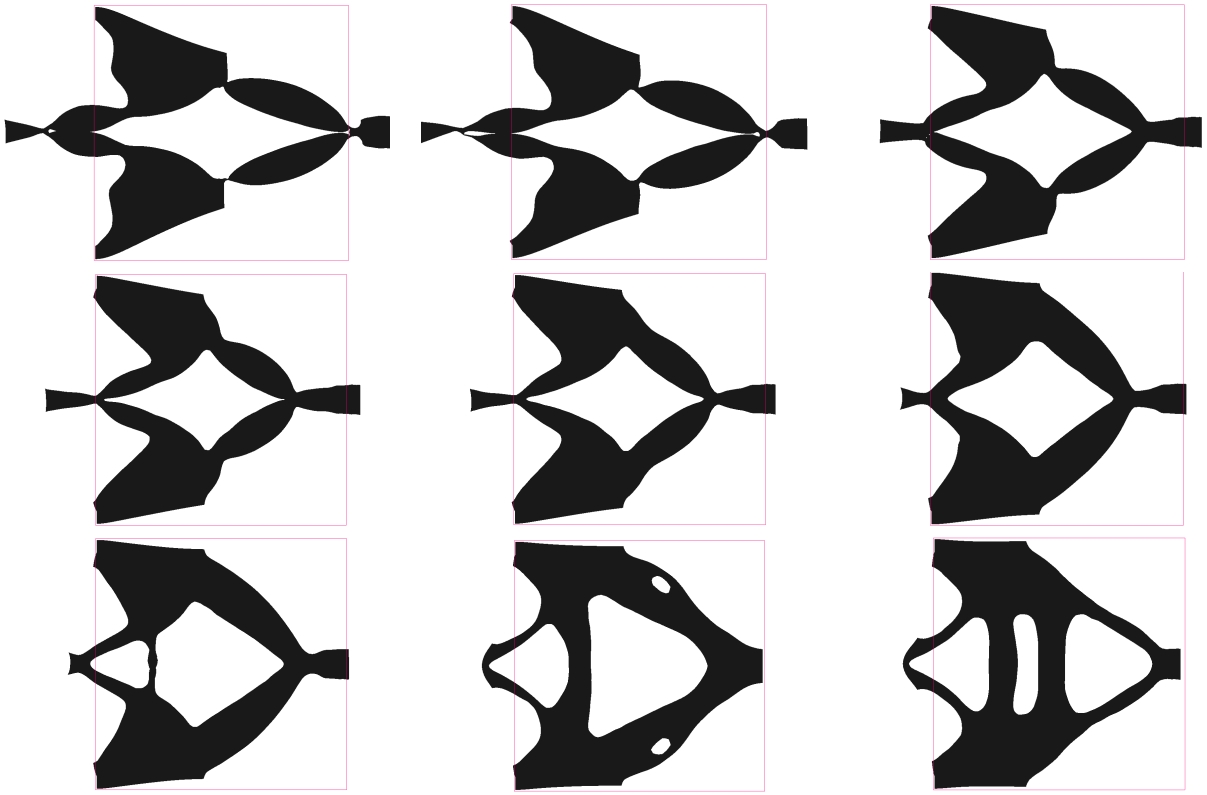


Figure 5.15: *(From left to right, top to bottom):* deformed configurations of the optimal shapes of Figure 5.14, with $m = 0, 0.001, 0.002, 0.003, 0.0045, 0.0075, 0.01, 0.02, 0.1$. The bounding box of the optimal shapes is displayed in red.

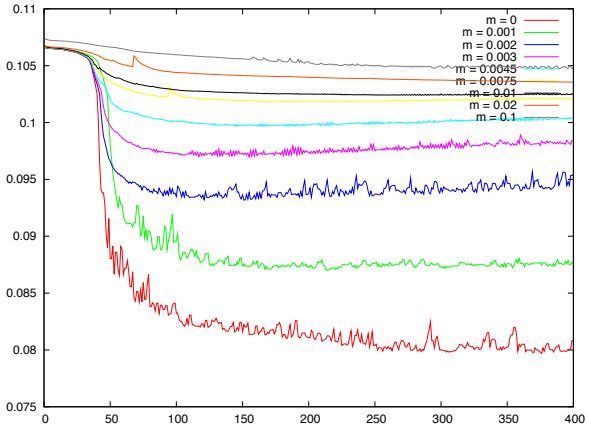


Figure 5.16: Convergence histories for the (worst-case) force inverter test case.

5.5.2.4 Shape optimization under geometric uncertainties

We eventually look into the setting of Section 5.4.4, where geometric uncertainties are considered, first in the context of the device of a gripping mechanism, as illustrated in Figure 5.17: the shapes of interest are clamped on the top and bottom parts of their left-hand side, and a small horizontal force $g = (0.1, 0)$ is applied at the centre of this side, with the hope that the jaws (corresponding to the blue area in Figure 5.17) will comply with a target displacement u_0, equalling $(0, -0.2)$ on the upper part, and $(0, 0.2)$ on the lower part. The cost $\mathcal{C}(\Omega)$ of a shape Ω reads:

$$\forall \Omega \in \mathcal{U}_{ad},\ \mathcal{C}(\Omega) = \int_\Omega k(x)|u_\Omega - u_0|^2\, dx,$$

where k is the characteristic function of the area near the jaws.

As perturbations of magnitude m on the geometry of shapes are expected, we aim at optimizing the approximate worst-case functional $\widetilde{\mathcal{J}}$ associated to this problem, defined by formula (5.73). Small constraints over the volume and compliance of shapes are incorporated using fixed Lagrange multipliers $\ell_v = 0.003$, and $\ell_c = 1$, serving the same purposes as in the force inverter test case, and 200 iterations of the usual gradient-based algorithm are performed.

Several results are displayed on Figure 5.17, corresponding to different values of m. Each computation takes about 35 minutes, except for the one associated to $m = 0$. The corresponding displacements are shown on Figure 5.18, and it is easily seen that, as expected, the performances of the unperturbed shapes are less and less efficient as m increases.

The proposed approach for addressing geometric uncertainties is eventually applied to a case where the stress of structures is at stakes. Our shapes are now L-shaped beams, clamped on their upper part, and submitted to traction loads $g = (0, -1)$ on a portion of their right-hand side (see the details on Figure 5.19). The cost $\mathcal{C}(\Omega)$ of a shape Ω is now related to the stress $\sigma(u_\Omega)$ induced by its displacement as:

$$\forall \Omega \in \mathcal{U}_{ad},\ \mathcal{C}(\Omega) = \int_\Omega k(x)||\sigma(u_\Omega)||^p\, dx, \tag{5.85}$$

where $p \geq 2$, k is a characteristic function which equals 1 everywhere on the working domain except near the area Γ_N where loads are applied, and $||.||$ is the Frobenius norm for matrices.

The worst-case design associated to this cost function is investigated, when uncertainties over the geometry of the shape of maximum amplitude m are expected. The approximate worst-case function $\widetilde{\mathcal{J}}$ defined by (5.78) is minimized, using parameter $p = 2$.

At first, a volume constraint is enforced by means of a fixed Lagrange multiplier $\ell = 3$, and several examples are shown in figure 5.19, associated to various values of m. An augmented Lagrangian method is then used to impose a target volume $V_T = 0.8$ on shapes and confirm the changes in trends caused by uncertainties over the geometry: see the results on Figure 5.20, the stress distribution in the resulting structures in Figure 5.22, and the convergence histories in Figure 5.21.

Eventually, the same procedure is applied for the value $p = 5$ (and still increasing values for m). As expected, the resulting optimal shapes are more 'rounded' in the vicinity of the reentrant corner, where a stress singularity develops. See Figure 5.23 for results, Figure 5.24 for the stress distribution in the shapes, and Figure 5.25 for convergence histories.

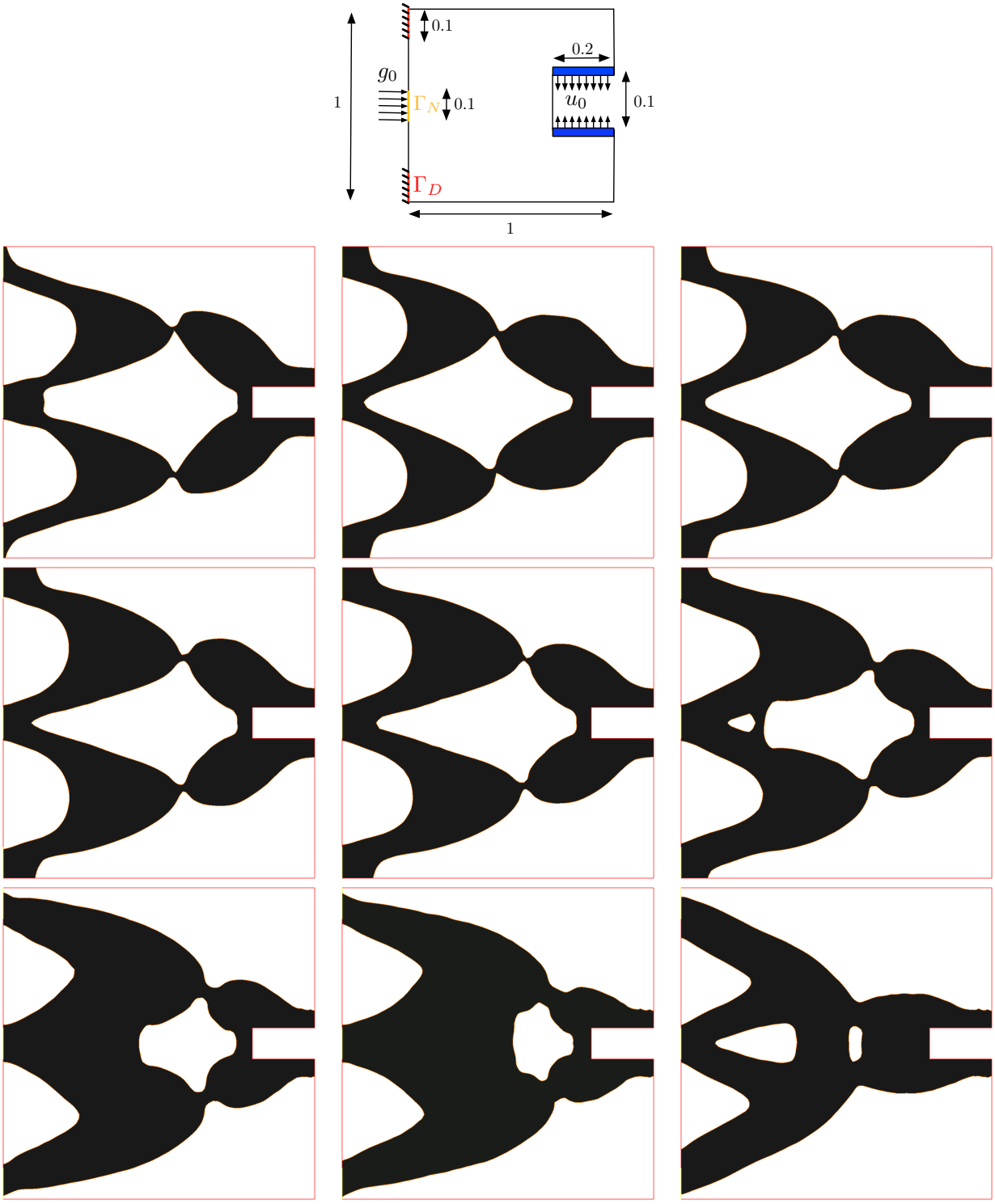


Figure 5.17: *(From left to right, top to bottom):* details of the test-case, optimal shape for the worst-case gripping mechanism test case, with $m = 0, 0.001, 0.002, 0.004, 0.005, 0.007, 0.009, 0.01, 0.02$.

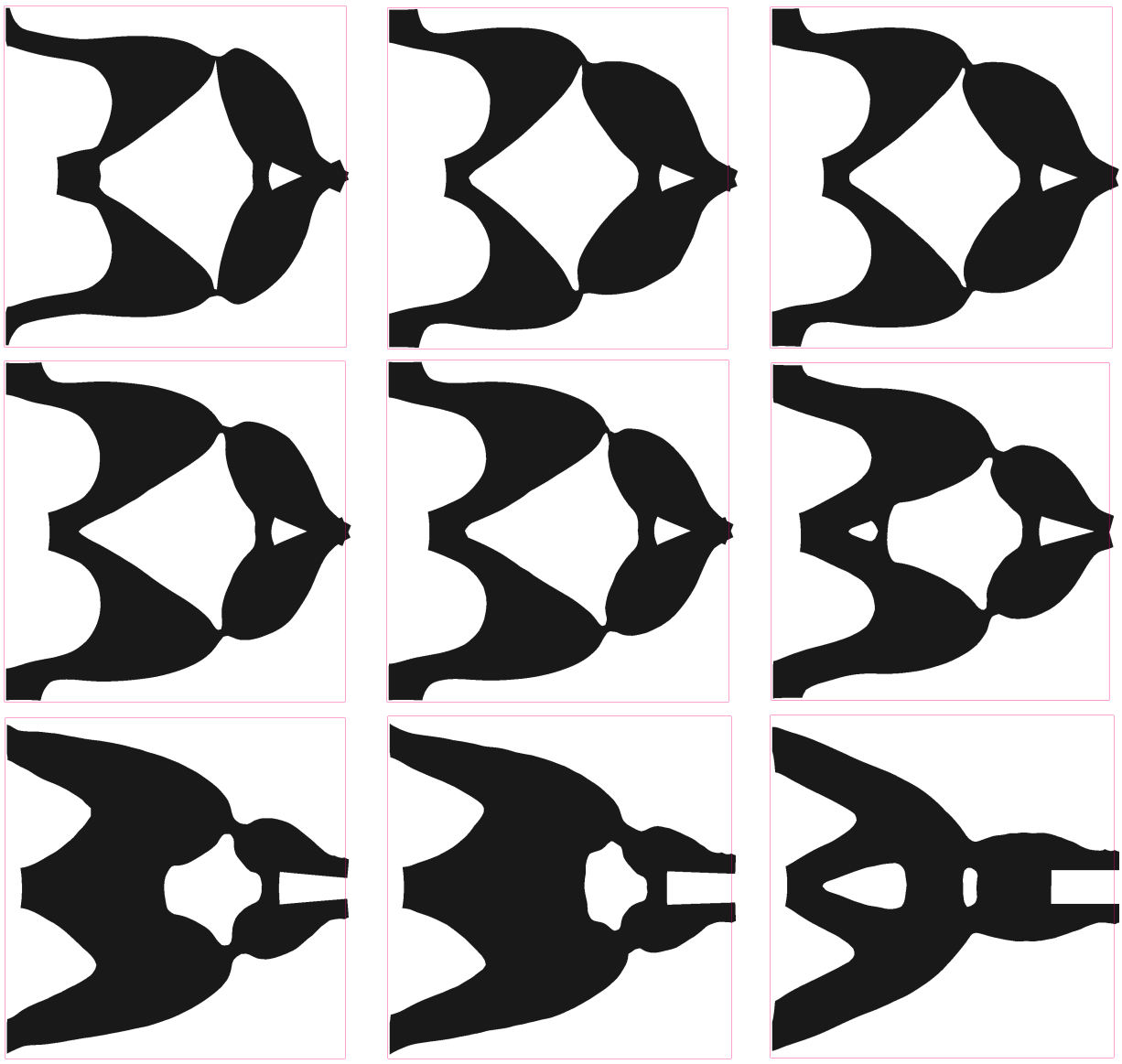


Figure 5.18: *(From left to right, top to bottom):* deformed configurations of the optimal shapes for the worst-case gripping mechanism test case, with $m = 0, 0.001, 0.002, 0.004, 0.005, 0.007, 0.009, 0.01, 0.02$ (the bounding box of the optimal shapes is displayed in red).

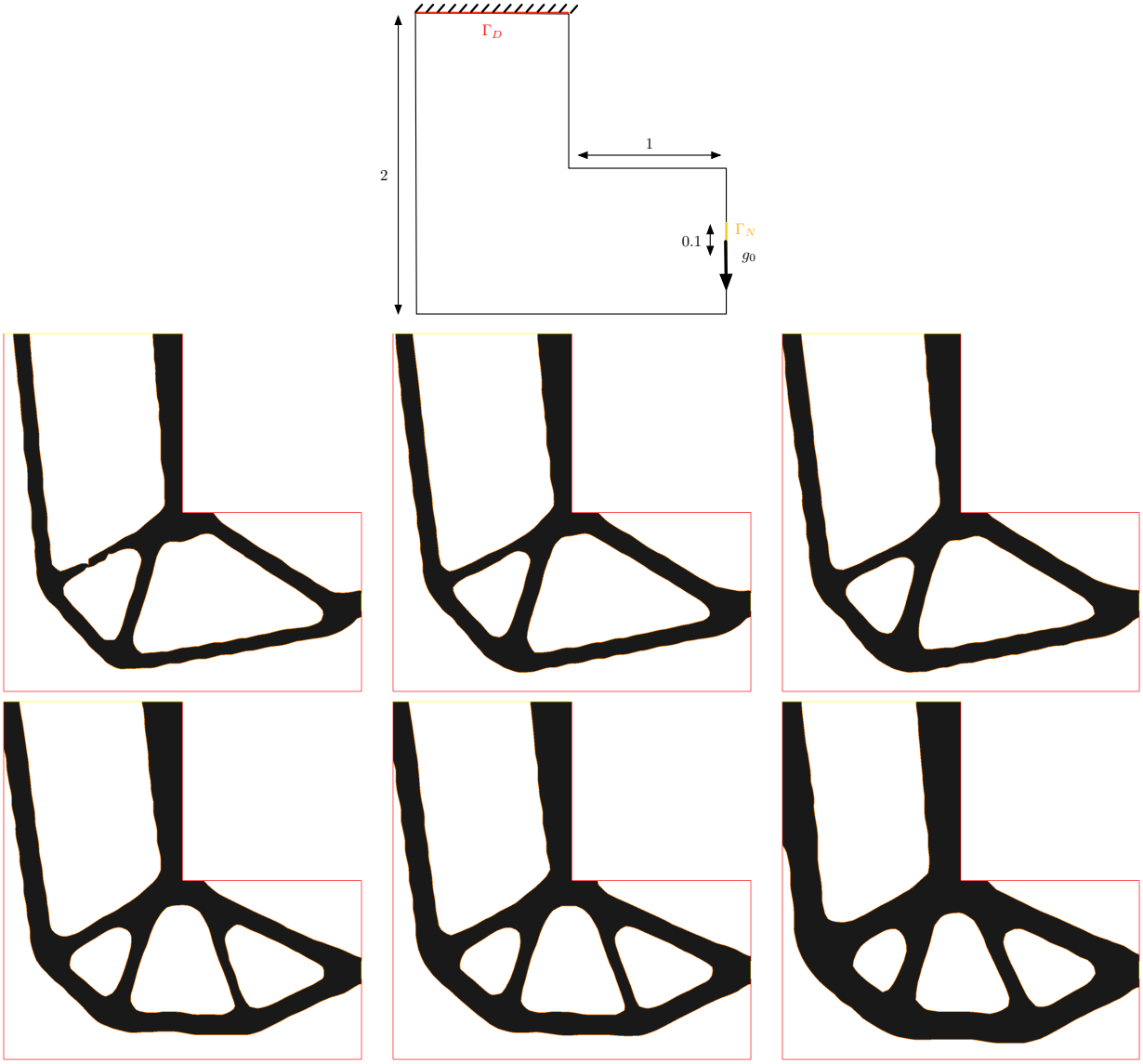


Figure 5.19: *(From left to right, top to bottom):* details of the test-case, optimal shape in the (worst-case) L-Beam test case, for geometric perturbations of amplitude $m = 0, 0.005, 0.01, 0.015, 0.02, 0.05$.

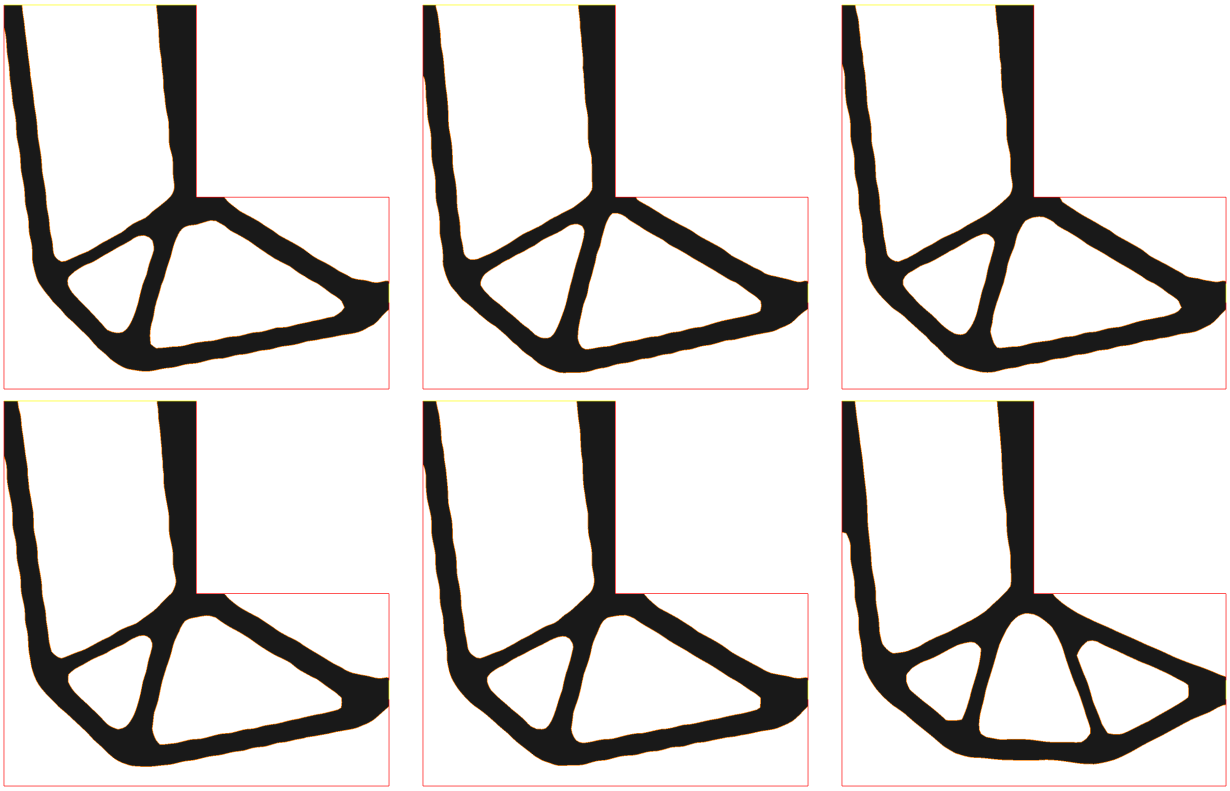


Figure 5.20: *(From left to right, top to bottom):* details of the test-case, optimal shape for $m = 0, 0.005, 0.01, 0.015, 0.02, 0.05$, for the (worst-case) L-Beam example, using $p = 2$.

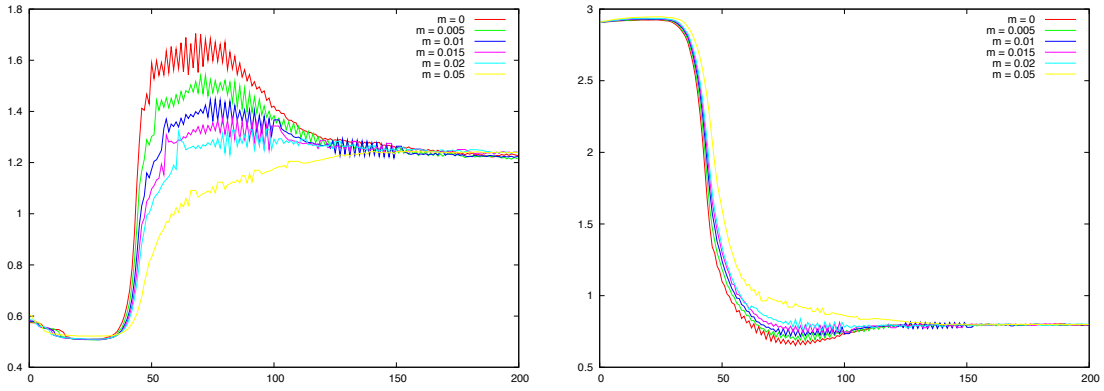


Figure 5.21: Convergence history for *(left):* the stress $\int_\Omega k||\sigma(u_\Omega)||^p\,dx$ and *(right):* the volume, in the (worst-case) L-Beam example, using $p = 2$.

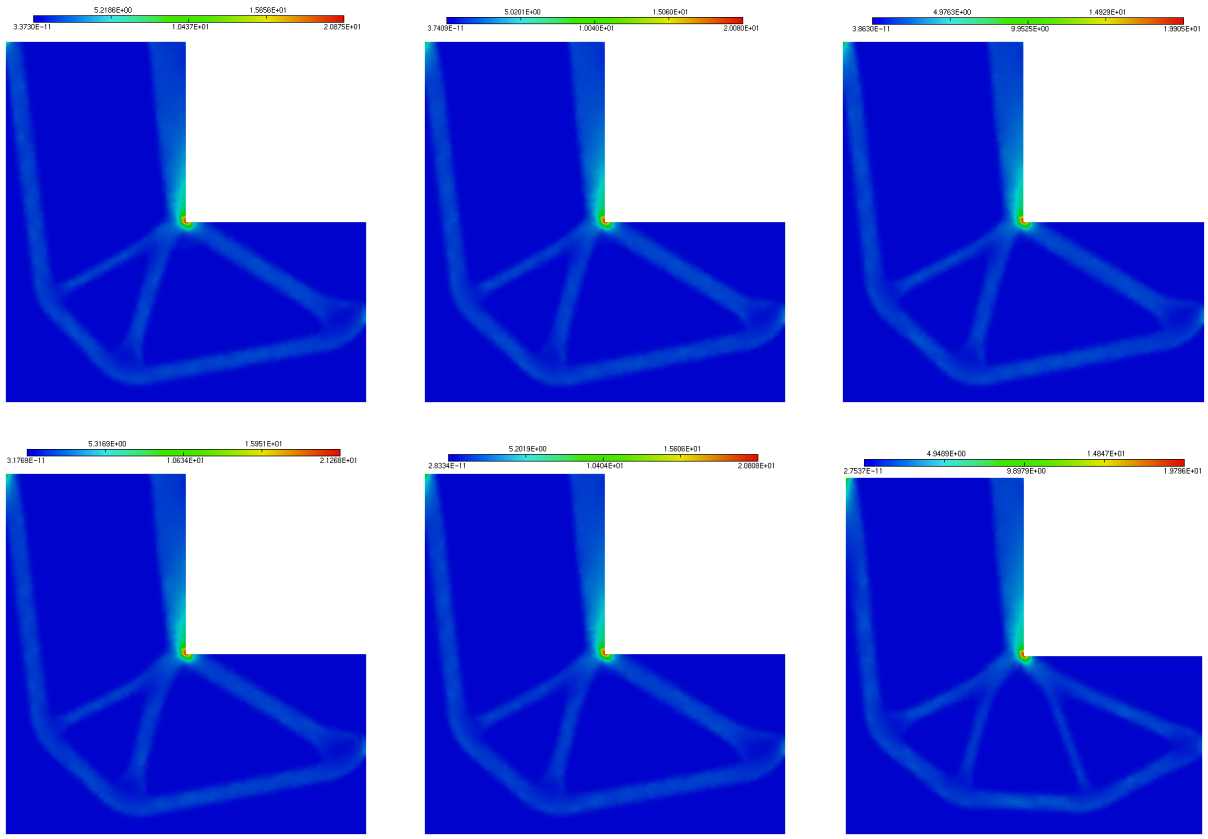


Figure 5.22: *(From left to right, top to bottom):* stress distribution ($||\sigma||^2$) for $m = 0, 0.005, 0.01, 0.015, 0.02, 0.05$ in the optimal L-Beams displayed on Figure 5.20, using $p = 2$.

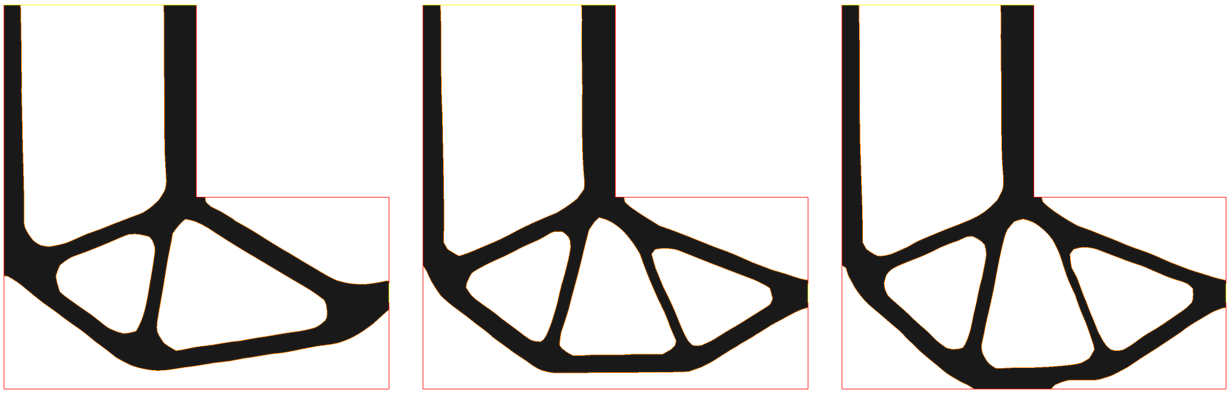


Figure 5.23: *(From left to right):* optimal shape for $m = 0, 0.01, 0.02$, for the L-Beam example, using $p = 5$.

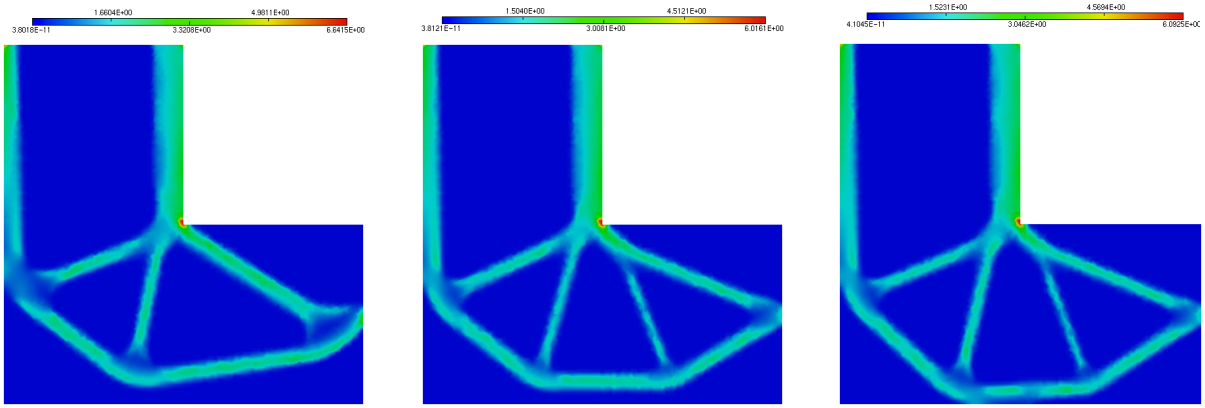


Figure 5.24: *(From left to right):* stress distribution ($||\sigma||^2$) for $m = 0, 0.01, 0.02$ for the L-Beam example, using $p = 5$.

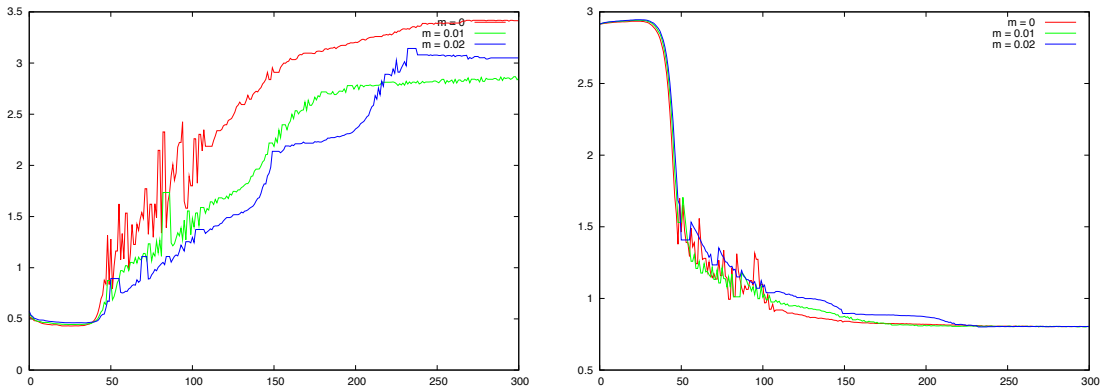


Figure 5.25: Convergence history for the volume for the L-Beam example, using $p = 5$.

Appendix: some useful technical tools

5.5.3 General tools

For the reader's convenience, this section reproduces several well-known results from the general theory of linear and Banach spaces that are repeatedly used throughout this chapter. First of all, the following basic fact allows for the identification of the dual norm of a product of vector spaces:

Lemma 5.5. *Let E, F two vector spaces. Then the application $\phi : (E \times F)^* \to E^* \times F^*$ defined as:*

$$\forall \ell \in (E \times F)^*, \ \phi(\ell) = (\ell(.,0), \ell(0,.))$$

is a linear isomorphism, which is an isometry when $E^* \times F^*$ is equipped with the norm $||.||_1$, defined as:

$$\forall (\ell_E, \ell_F) \in E^* \times F^*, \ ||(\ell_E, \ell_F)||_1 = ||\ell_E||_{E^*} + ||\ell_F||_{F^*}.$$

Recall the following corollary of Hahn-Banach theorem:

Lemma 5.6. *Let $(E, ||.||)$ any normed vector space, and E^* its topological dual. Then, for all $x \in E$, one has:*

$$||x|| = \sup_{\substack{\varphi \in E^* \\ ||\varphi||_{E^*} \leq 1}} < \varphi, x >_{E^*,E} = \max_{\substack{\varphi \in E^* \\ ||\varphi||_{E^*} \leq 1}} < \varphi, x >_{E^*,E}.$$

The following theorem can be found in [57]:

Theorem 5.11. *Let $\Omega \subset \mathbb{R}^d$ any open set, $\varphi \in L^1(\Omega)^*$. Then, there exists a unique function $u \in L^\infty(\Omega)$ such that:*

$$\forall f \in L^1(\Omega), \ \langle \varphi, f \rangle_{L^1(\Omega)^*, L^1(\Omega)} = \int_\Omega uf \, dx.$$

What's more, one has: $||u||_{L^\infty(\Omega)} = ||\varphi||_{L^1(\Omega)^*}$.

We shall also need the following easy result:

Lemma 5.7. *Let $\Omega \subset \mathbb{R}^d$ an open, bounded domain, and denote as $f : L^1(\Omega) \to \mathbb{R}$ the L^1 norm function:*

$$\forall u \in L^1(\Omega), \ f(u) = \int_\Omega |u(x)| \, dx.$$

f is then convex, and its subgradient $\partial f(u) \subset L^\infty(\Omega)$ at any point $u \in L^1(\Omega)$ reads:

$$\lambda \in \partial f(u) \ \Leftrightarrow \ \left\{ \begin{array}{cc} \lambda(x) = 1 & \text{if } u(x) > 0 \\ \lambda(x) = -1 & \text{if } u(x) < 0 \\ \lambda(x) \in [-1, 1] & \text{if } u(x) = 0 \end{array} \right. .$$

As a consequence, if $u \in L^1(\Omega)$ is such that $\{x \in \Omega, \ u(x) = 0\}$ is of null Lebesgue measure, f is Fréchet-differentiable at u, and its differential $df(u)$ reads:

$$\forall v \in L^1(\Omega), \ df(u)(v) = \int_\Omega \mathrm{sgn}(u(x)) v(x) \, dx.$$

5.5.4 Several Green's formulae

Notations: Let $\Gamma \subset \mathbb{R}^d$ a compact, oriented $\mathcal{C}^2$ submanifold of dimension $d-1$, and $V \in \mathcal{C}^1(\Gamma, \mathbb{R}^d)$ a vector field on Γ. The *tangential divergence* $\mathrm{div}_\Gamma(V) \in \mathcal{C}^0(\Gamma)$ of V is defined as:

$$\mathrm{div}_\Gamma(V) = \mathrm{div}(\widetilde{V}) - \nabla \widetilde{V} n \cdot n,$$

where $\widetilde{V}$ stands for *any* $\mathcal{C}^1$ extension of V to a tubular neighborhood of Γ. One can actually see [172] that this definition is independent on the choice of such an extension, and that, denoting as $D_\Gamma V$ the $d \times d$ matrix whose i-th line equals $\nabla_\Gamma V_i$,

$$\mathrm{div}_\Gamma(V) = \mathrm{tr}\left(D_\Gamma V\right).$$

There is another equivalent point of view as regards the tangential divergence of a vector field, which is completely intrinsic to the submanifold Γ. Γ may be viewed as a Riemannian manifold, when equipped with the Euclidean metric on $\mathbb{R}^d$. Let ∇ be the associated Levi-Civita connection on Γ. Then, if V is a vector field on Γ (i.e. for this definition, we need that, for all $x \in \Gamma$, $V(x) \in T_x\Gamma$), for any $x \in \Gamma$ $\mathrm{div}_\Gamma(V)(x)$ may be defined as the trace of the linear operator

$$T_x\Gamma \ni \xi \longmapsto \nabla_\xi V(x) \in T_x\Gamma.$$

Now, let $\sigma \in \mathcal{C}^0(\Gamma, \mathcal{S}(\mathbb{R}^d))$ be a continuous symmetric matrix-valued function. The *tangential part* $\sigma_{\tau\tau}$ of σ is the symmetric bilinear form over $T\Gamma$ (or equivalently the associated symmetric matrix-valued function) defined as:

$$\forall x \in \Gamma, \forall v, w \in T_x\Gamma, \;\; \sigma_{\tau\tau}(x)(v,w) = \sigma(x)(v,w).$$

Roughly speaking, $\sigma_{\tau\tau}(x)$ is the restriction of $\sigma(x)$ to the tangent plane $T_x\Gamma$ - i.e. σ reads:

$$\sigma = \begin{pmatrix} \sigma_{\tau\tau} & \sigma_{\tau n} \\ \sigma_{n\tau} & \sigma_{nn} \end{pmatrix}$$

in an orthonormal basis of $\mathbb{R}^d$ obtained by gathering $(d-1)$ tangent vectors of Γ (collectively denoted as τ).

Similarly, if $\sigma \in \mathcal{C}^1(\Gamma, \mathcal{S}(\mathbb{R}^d))$, one defines its *tangential divergence* $\mathrm{div}_\Gamma : \Gamma \to \mathbb{R}^d$ as:

$$\forall i = 1, ..., d, \;\; \mathrm{div}_\Gamma(\sigma)_i = \left[\mathrm{div}_\Gamma\left((\sigma_{i,j})_{j=1,...,d}\right)\right]_\Gamma,$$

where $[.]_\Gamma$ denotes the projection of a vector field onto $T\Gamma$.

Let us start with the following Green's formula on a submanifold of $\mathbb{R}^d$:

Proposition 5.3. *Let $\Gamma \subset \mathbb{R}^d$ a compact, oriented $\mathcal{C}^2$ submanifold of dimension $d-1$, with (possibly empty) boundary Σ. Let $n_\Sigma : \Sigma \to \mathbb{S}^{d-1}$ the outer unit normal vector to Σ in Γ. For any $\mathcal{C}^1$ function $u : \mathbb{R}^d \to \mathbb{R}$, and any $\mathcal{C}^1$ vector field $\tau \in T\Gamma$, one has:*

$$\int_\Gamma \frac{\partial u}{\partial \tau}\, ds = \int_\Sigma u\tau.n_\Sigma \, d\ell - \int_\Gamma \mathrm{div}_\Gamma(\tau) u \, ds, \tag{5.86}$$

where ds and $d\ell$ stand for the volume forms on Γ and Σ respectively.

Proof. First, note that, owing to a standard argument involving partitions of unity, it is enough to show that formula (5.86) holds locally, i.e. with U and $U \cap \Sigma$ instead of Γ and Σ respectively, where U is an arbitrarily small open subset of Γ.

Now, we can assume that Γ amounts to a single local chart, and introduce local coordinates $x := (x_1, ..., x_{d-1}) : \Gamma \to W \subset \mathbb{R}^{d-1}$ on Γ, that is, x is a $\mathcal{C}^2$ diffeomorphism. Let $(\frac{\partial}{\partial x_1}, ..., \frac{\partial}{\partial x_{d-1}})$ the associated basis of the tangent bundle $T\Gamma$, and $(dx_1, ..., dx_{d-1})$ the dual basis of $T\Gamma^*$. The decomposition of τ into this

local basis of the tangent bundle then reads $\tau = \sum_{i=1}^{d-1} \tau_i \frac{\partial}{\partial x_i}$, and the expression of the volume form ds in terms of these local coordinates reads:

$$ds = \sqrt{\det(g)}\, dx_1 \wedge ... \wedge dx_{d-1},$$

where g is the $(d-1) \times (d-1)$ matrix

$$\forall p \in \Gamma, \forall i, j = 1, ..., d-1, \;\; g_{i,j}(p) = \langle \frac{\partial}{\partial x_i}(p), \frac{\partial}{\partial x_j}(p) \rangle_p.$$

One computes:

$$\begin{aligned}
\int_\Gamma \frac{\partial u}{\partial \tau}\, ds &= \sum_{i=1}^{d-1} \int_\Gamma \frac{\partial u}{\partial x_i} \tau_i \, ds \\
&= \sum_{i=1}^{d-1} \int_\Gamma \frac{\partial u}{\partial x_i} \tau_i \sqrt{\det(g)}\, dx_1 \wedge ... \wedge dx_{d-1} \\
&= \sum_{i=1}^{d-1} \int_\Gamma \frac{\partial}{\partial x_i} \left(u \tau_i \sqrt{\det(g)} \right) dx_1 \wedge ... \wedge dx_{d-1} - \sum_{i=1}^{d-1} \int_\Gamma \frac{\partial}{\partial x_i} \left(\tau_i \sqrt{\det(g)} \right) u dx_1 \wedge ... \wedge dx_{d-1}
\end{aligned}$$

Now, using the expression of the tangential divergence in terms of local coordinates:

$$\mathrm{div}_\Gamma(\tau) = \frac{1}{\sqrt{\det(g)}} \sum_{i=1}^{d-1} \frac{\partial}{\partial x_i} \left(\sqrt{\det(g)} \tau_i \right),$$

the second term amounts to:

$$\sum_{i=1}^{d-1} \int_\Gamma \frac{\partial}{\partial x_i} \left(\tau_i \sqrt{\det(g)} \right) u dx_1 \wedge ... \wedge dx_{d-1} = \int_\Gamma \mathrm{div}_\Gamma(\tau) u \, ds.$$

As for the first term, using Stokes' theorem (Th. XII.2.1 in [198]), it rewrites:

$$\sum_{i=1}^{d-1} \int_\Sigma (-1)^{i-1} \tau_i \, u \sqrt{\det(g)}\, dx_1 \wedge ... \wedge dx_{i-1} \wedge dx_{i+1} \wedge ... \wedge dx_{d-1}.$$

Hence, the only remaining thing to prove is that:

$$\forall i = 1, ..., d-1, \;\; (-1)^{i-1} \sqrt{\det(g)}\, dx_1 \wedge ... \wedge dx_{i-1} \wedge dx_{i+1} \wedge ... \wedge dx_{d-1} = (n_\Sigma)_i d\ell.$$

To see this, note that because $d\ell$ is a volume form on Σ, there exists a function $\alpha \in \mathcal{C}^\infty(\Gamma)$ such that $(-1)^{i-1} \sqrt{\det(g)} dx_1 \wedge ... \wedge dx_{i-1} \wedge dx_{i+1} \wedge ... \wedge dx_{d-1} = \alpha d\ell$. We are left with the problem of identifying α; this can be carried out in a pointwise fashion. Thus, let $p \in \Sigma$, and $(e_1, ..., e_{d-2})$ a direct orthonormal basis of $T_p\Sigma$ (that is, ordered so that $(n_\Sigma, e_1, ..., e_{d-2})$ is a direct orthonormal basis of $T_p\Gamma\,|_\Sigma$). By definition, one has:

$$d\ell(e_1, ..., e_{d-2}) = 1.$$

On the other hand,

$$dx_1 \wedge ... \wedge dx_{i-1} \wedge dx_{i+1} \wedge ... \wedge dx_{d-1}(e_1, ..., e_{d-2}) = \begin{vmatrix} dx_1(e_1) & ... & dx_1(e_{d-2}) \\ \vdots & \vdots & \vdots \\ dx_{i-1}(e_1) & ... & dx_{i-1}(e_{d-2}) \\ dx_{i+1}(e_1) & ... & dx_{i+1}(e_{d-2}) \\ \vdots & \vdots & \vdots \\ dx_{d-1}(e_1) & ... & dx_{d-1}(e_{d-2}) \end{vmatrix}.$$

This last expression arises as the $(i,1)$ minor of $\det(M)$, where M is the $(d-1)\times(d-1)$ matrix defined as:

$$M = \begin{pmatrix} dx_1(n_\Sigma) & dx_1(e_1) & ... & dx_1(e_{d-2}) \\ \vdots & \vdots & \vdots & \vdots \\ dx_i(n_\Sigma) & dx_i(e_1) & ... & dx_i(e_{d-2}) \\ \vdots & \vdots & \vdots & \vdots \\ dx_{d-2}(n_\Sigma) & dx_{d-1}(e_1) & ... & dx_{d-1}(e_{d-2}) \end{pmatrix}.$$

Note that M is the passing matrix between the two bases $(n_\Sigma, e_1, ..., e_{d-2})$ (which is orthonormal) and $(\frac{\partial}{\partial x_1}, ..., \frac{\partial}{\partial x_{d-1}})$ of $T_p\Gamma$. Hence,

$$(-1)^{i-1} dx_1 \wedge ... \wedge dx_{i-1} \wedge dx_{i+1} \wedge ... \wedge dx_{d-1}(e_1, ..., e_{d-2}) = \frac{1}{\det(M)} \left({}^tM^{-1}\right)_{i,1}.$$

Now, it is not difficult to see that $\det(M) = \sqrt{\det(g)}$ (see e.g. [198], prop. XV.1.1). What's more, $\left({}^tM^{-1}\right)_{i,1}$ is the coefficient of $\frac{\partial}{\partial x_i}$ along n_Σ in the orthonormal basis $(n_\Sigma, e_1, ..., e_{d-2})$, that is $n_{\Sigma i}$, which ends the proof. □

We shall have use of this result under the following form:

Proposition 5.4. *Let $\Gamma \subset \mathbb{R}^d$ a compact, oriented submanifold of class $\mathcal{C}^2$, with (possibly empty) boundary Σ. For any function $u \in \mathcal{C}^1(\mathbb{R}^d)^d$, and any symmetric matrix-valued function $\sigma \in \mathcal{C}^1(\mathcal{S}(\mathbb{R}^d))$, one has:*

$$\int_\Gamma \sigma_{\tau\tau} : e(u)_{\tau\tau}\, ds = \int_\Sigma [u]_\Gamma.(\sigma_{\tau\tau}.n_\Sigma)\, d\ell - \int_\Gamma [u]_\Gamma.\mathrm{div}_\Gamma(\sigma)\, ds.$$

Proof. This formula is a consequence of the following identity:

$$\mathrm{div}_\Gamma(\sigma_{\tau\tau}.[u]_\Gamma) = \sigma_{\tau\tau} : e(u)_{\tau\tau} + [u]_\Gamma.\mathrm{div}_\Gamma(\sigma). \tag{5.87}$$

Indeed, if this equality holds, then using proposition (5.3) with $\tau = \sigma_{\tau\tau}.[u]_\Gamma$, $u = 1$ leads to the desired result.

As for proving (5.87), consider any fixed point $p \in \Gamma$, and chose local *normal* coordinates $(x_1, ..., x_{d-1})$ on Γ at p. In particular, (see [116], chap. 3, ex. 14), one has:

- The basis $(\frac{\partial}{\partial x_1}(p), ..., \frac{\partial}{\partial x_{d-1}}(p))$ of $T_p\Gamma$ is orthonormal.
- The Christoffel symbols associated to this basis of $T\Gamma$ vanish at p, that is:

$$\forall i,j = 1, ..., d-1,\ \nabla_{\frac{\partial}{\partial x_i}} \frac{\partial}{\partial x_j}\ (p) = 0, \tag{5.88}$$

  where ∇ stands for the Levi-Civita connection on Γ when equipped with the metric induced by the Euclidean scalar product of $\mathbb{R}^d$.
- For any vector field X on Γ, whose coordinates in system $(x_1, ..., x_{d-1})$ read: $X = \sum_{i=1}^{d-1} X_i \frac{\partial}{\partial x_i}$, one has:

$$\mathrm{div}_\Gamma(X)(p) = \sum_{i=1}^{d-1} \frac{\partial X_i}{\partial x_i}(p). \tag{5.89}$$

Now writing $\sigma_{\tau\tau}.[u]_\Gamma$ in coordinates $(x_1, ..., x_{d-1})$, and using formulae (5.88 - 5.89) yields the desired result. □

Remark 5.10. Propositions 5.3 and 5.4 were stated and proved in the context of a smooth function $u \in \mathcal{C}^1(\mathbb{R}^d)$. Obviously, the results extend owing to the standard density argument to the case of functions $u \in H^2(\mathbb{R}^d)$.

Part III

Level set methods on unstructured meshes; connections with mesh adaptation

Chapter 6

Computation of the signed distance function to a discrete contour on adapted simplicial mesh.

Contents

6.1 Introduction 226
6.2 Some preliminaries about the signed distance function 226
6.3 A short study of some properties of the solution to the unsteady Eikonal equation 228
6.4 A numerical scheme for the signed distance function approximation 230
6.4.1 Extending the signed distance function from the boundary 230
6.4.2 Initialization of the signed distance function near $\partial\Omega$ 231
6.5 Mesh adaptation for a sharper approximation of the signed distance function 233
6.5.1 Anisotropic mesh adaptation 234
6.5.2 Computation of a metric tensor associated to the minimization of the $\mathbb{P}^1$ interpolation error 235
6.5.3 Mesh adaptation for a geometric reconstruction of the 0 level set of a function . . 236
6.6 A remark about level-set redistancing 239
6.7 Extension to the computation of the signed distance function in a Riemannian space 240
6.8 Numerical Examples 240

In this chapter, we present a numerical method for computing the signed distance function to a discrete domain, at the vertices of an arbitrary simplicial computational mesh (i.e. composed of triangles in two dimensions, and tetrahedra in three dimensions). It mainly relies on the use of some theoretical properties of the unsteady Eikonal equation. Then we propose a way to adapt the mesh on which computations are held to enhance at the same time the accuracy of the approximation of the signed distance function and the approximation of the initial discrete contour by the piecewise affine reconstruction as the 0 level set set of its signed distance function, which is crucial when using this signed distance function in a context of level set methods. Several two- and three-dimensional examples are presented to appraise our analyses.

This chapter is a joint work with Pascal Frey. Its contents have been published under the reference:

C. Dapogny and P. Frey, *Computation of the signed distance function to a discrete contour on adapted triangulation*, Calcolo, Vol. 49, Issue 3, (2012), pp. 193-219.

6.1 Introduction

The knowledge of the signed distance function to a domain $\Omega \subset \mathbb{R}^d$ ($d = 2, 3$ in our applications) has proved to be a valuable information in various fields such as collision detection [146], shape reconstruction from an unorganized cloud of points [333],[93], and of course in the setting of the level set methods of Sethian and Osher [245] (see also [274] or [242], and chapter 1 for a presentation and an overview of the main features of the method), where ensuring the property of unitary gradient of the level set function is especially relevant.

In this chapter, we intend to devise a numerical algorithm that computates the signed distance function to a *polyhedral* domain Ω, supplied with minimal information. We only rely on the knowledge of a triangulation $\mathcal{S}$ of its boundary $\partial\Omega$, which we suppose to be orientable (so that Ω is well-defined on the sole basis of its boundary). With this assumption, we imply that we do not rely on any knowledge of the outer normal to the considered surface, nor do we suppose the mesh $\mathcal{S}$ to be conforming. For the moment, we do not want to make any particular assumption on the simplicial background mesh $\mathcal{T}$ on which computations are performed either, even if, of course, the quality of the solution may depend on it, as will be seen.

Regarding the computation of the signed distance function, there exist mainly two types of approaches (see also the overview in chapter 1), both of them being based on an approximation of the solution of the Eikonal equation (6.1). The first one advocates to treat this equation as a stationary boundary value problem, and to start from the knowledge of the distance in the elements of the computational mesh which are close to the interface $\partial\Omega$, then to propagate the information throughout the whole domain. The most notorious methods - such as the Fast Marching Method [275], [276], [194] or the Fast Sweeping Method [332], [262] - belong to this category, and rely on a local solver for the Eikonal equation, and on a marching method, meant to enforce the natural causality embedded in the equation. Another way of addressing the problem is to consider it as an unsteady problem and then to devise a propagation method for extending the signed distance field from the boundary of Ω - see [292] [81] [298]. This is the point of view retained in this chapter. It leads to an efficient, easy to implement, and easy to parallelize method for computing the signed distance function to Ω. This method is presented in dimensions 2 and 3, but naturally extends to the general case.

Furthermore, in a context where the signed distance function is used as an implicit function defining the domain Ω, the quality of the approximation of $\partial\Omega$ by its piecewise affine reconstruction as the 0 level set of the computed signed distance function is a crucial point. Unfortunately, the discrete interface thus obtained can prove quite different from the true interface $\partial\Omega$ (this is especially likely to happen if the computational mesh is too coarse). Following the work presented in [119], we investigate into an adaptation scheme, based on the signed distance function, to produce a background mesh adapted to the boundary $\partial\Omega$ so as to improve in the meantime the computation of the signed distance function to Ω - at least in the areas of the computational domain where it is relevant - and the associated numerical reconstruction of Ω.

The remainder of this chapter is organized as follows. In section 6.2, we collect the general properties of the signed distance function which are useful in this work, and among other things recall how it can be seen as the stationnary state of the solution of the unsteady Eikonal equation. This urges us to study the dynamics of this equation in section 6.3. From this study, we infer a numerical scheme for approximating the signed distance function in section 6.4. We then show in section 6.5.1 how the background computational mesh can be adapted so that both the approximation of the signed distance function and the discrete isosurface resulting from the process can be controlled and improved. We briefly discuss two interesting extensions of this work, to the problem of reinitialization of a level set function in section 6.6, and to the computation of the signed distance function to a domain in a Riemannian space in section 6.7. Numerical examples are eventually provided in section 6.8 to emphasize the main features of the presented approach.

6.2 Some preliminaries about the signed distance function

For the sake of clarity, let us start this work by recalling the facts around the signed distance function to a domain that we shall need in our study.

Definition 6.1. *Let $\Omega \subset \mathbb{R}^d$ a bounded domain. The signed distance function to Ω is the function $\mathbb{R}^d \ni x \mapsto d_\Omega(x)$ defined by:*

$$d_\Omega(x) = \begin{cases} -d(x, \partial\Omega) & \text{if } x \in \Omega \\ 0 & \text{if } x \in \partial\Omega \\ d(x, \partial\Omega) & \text{if } x \in {}^c\overline{\Omega} \end{cases},$$

where $d(., \partial\Omega)$ denotes the usual Euclidean distance function to the set $\partial\Omega$.

When studying the distance to such a bounded domain Ω, the notion of *skeleton* plays a major role:

Definition 6.2. *The skeleton Σ of $\partial\Omega$ is the set of points $x \in \mathbb{R}^d$ such that the minimum in*

$$d(x, \partial\Omega) = \min_{y \in \partial\Omega} |x - y|$$

is achieved for at least two distinct points of $\partial\Omega$.

Being 1-lipschitz, and owing to Rademacher's theorem [126], the function d_Ω is almost everywhere differentiable. Actually, the following interesting proposition delivers a geometric characterization of the regularity of d_Ω [105].

Theorem 6.1. *Let $\Omega \subset \mathbb{R}^d$ be a $\mathcal{C}^2$ bounded domain; then there exists a tubular neighborhood V of $\partial\Omega$ such that d_Ω is of class $\mathcal{C}^2$ on V. Moreover, for any point $x \in \mathbb{R}^d$,*

- *either $x \in \partial\Omega$, and then d_Ω is differentiable at x, with*

$$\nabla d_\Omega(x) = n(x), \text{ the outer unit normal vector to } \partial\Omega \text{ at } x,$$

- *or $x \in \mathbb{R}^d \setminus \partial\Omega$, then for any projection point p_x of x onto $\partial\Omega$ (i.e. for any point $p_x \in \partial\Omega$ such that $d(x, \partial\Omega) = |x - p_x|$), one has:*

$$\frac{x - p_x}{d_\Omega(x)} = n(p_x).$$

Furthermore, d_Ω is differentiable at x if and only if x belongs to the complementary $\mathbb{R}^d \setminus \Sigma$ of the skeleton of $\partial\Omega$. In such a case, there exists a unique projection point of x onto $\partial\Omega$ - denoted as $p_{\partial\Omega}(x)$ - and the gradient $\nabla d_\Omega(x)$ reads:

$$\nabla d_\Omega(x) = \frac{x - p_{\partial\Omega}(x)}{d_\Omega(x)}.$$

In particular, d_Ω satisfies the Eikonal equation at every point at which it is differentiable:

$$\begin{cases} |\nabla d_\Omega(x)| = 1 & \text{a.e. } x \in \mathbb{R}^d \\ d_\Omega(x) = 0 & \text{for } x \in \partial\Omega \end{cases}. \tag{6.1}$$

Unfortunately, theory happens to be scarce as for functions being solutions of a PDE almost everywhere. For this reason - and many others - it is much more convenient from a theoretical point of view to see d_Ω as a *viscosity solution* of the Eikonal equation (see [126] again, and chapter 1).

Proposition 6.1. *d_Ω is the unique solution to the Eikonal equation (6.1) in the sense of viscosity.*

Another way of thinking of d_Ω consists in seeing it as the result of a propagation by means of an evolution equation: suppose Ω is implicitly known as

$$\Omega = \left\{x \in \mathbb{R}^d,\, d_0(x) < 0\right\} \text{ and } \partial\Omega = \left\{x \in \mathbb{R}^d,\, d_0(x) = 0\right\}, \tag{6.2}$$

where d_0 is a continuous function. Note that, in the theoretical framework, such a function u_0 exists and is quite easy to construct by means of partitions of unity techniques. Then the function d_Ω can be considered as the steady state of the so-called *redistancing equation* or *unsteady Eikonal equation*

$$\begin{cases} \dfrac{\partial d}{\partial t} + \mathrm{sgn}(d_0)(||\nabla d|| - 1) = 0 & \forall t > 0, x \in \mathbb{R}^d \\ d(t = 0, x) = d_0(x) & \forall x \in \mathbb{R}^d \end{cases} \tag{6.3}$$

Formally speaking, this equation starts with an arbitrary (continuous) function d_0 implicitly defining the domain Ω and straightens it into the 'best' function which suits that purpose - d_Ω. For this reason, this equation was first introduced in [299] for redistancing a very stretched level set function arising in the course of a computation. The following important theorem makes these statements more precise (a proof can be found in [26] and [182]):

Theorem 6.2. *Let Ω a bounded domain of $\mathbb{R}^d$ implicitly defined by a continuous function d_0 - i.e. (6.2) holds. Define function d, for any $x \in \mathbb{R}^d$ and $t \in \mathbb{R}_+$ by the following formula:*

$$d(t,x) = \begin{cases} \mathrm{sgn}(d_0(x)) \inf\limits_{||y|| \le t} (sgn(d_0(x)) d_0(x+y) + t) & \text{if } t \le d(x, \partial\Omega) \\ \mathrm{sgn}(d_0(x))\, d(x, \partial\Omega) & \text{if } t > d(x, \partial\Omega) \end{cases} \tag{6.4}$$

Let $T \in \mathbb{R}_+$. Then d is the unique uniformly continuous viscosity solution of (6.3) such that, for all $0 \le t \le T$, $d(t,x) = 0$ on $\partial\Omega$.

Note that the exact formula (6.4) expresses the idea of a propagation of information from the boundary with constant unit speed. This feature will later be exploited in the device of our numerical algorithm for computing the signed distance function to Ω.

6.3 A short study of some properties of the solution to the unsteady Eikonal equation

So as to build a resolution scheme for the unsteady Eikonal equation (6.3), let us take a closer look to its dynamics in view of the previous section.

The main idea is to start with a function d_0 which implicitly defines domain Ω in the sense that (6.2) holds and that we suppose continuous over $\mathbb{R}^d$, then to regularize it thanks to equation (6.3), considering the exact solution d, provided by formula (6.4), for the resulting Cauchy problem. For the sequel, it will prove convenient to assume moreover that the initial function d_0 is an *overestimation* of the signed distance function d_Ω to Ω, except on a tubular neighbourhood V of $\partial\Omega$, where it is exactly d_Ω, that is:

$$\forall x \in V,\ d_0(x) = d_\Omega(x)\ ;\ \forall x \in \mathbb{R}^d \setminus V\ ,\ |d_0(x)| \ge |d_\Omega(x)| \tag{6.5}$$

We then have the following small result:

Lemma 6.1. *Assume Ω is a bounded open set, with boundary $\partial\Omega$ of class $\mathcal{C}^2$. For any small enough time step $dt > 0$, denote $t^n = ndt$, $n \in \mathbb{N}$. Suppose the initial function d_0 satisfies (6.5) and denote d the solution of equation (6.3) provided by theorem 6.2. Then*

$$\begin{array}{lll} \forall x \in \mathbb{R}^d \setminus \Omega \quad d(t^{n+1}, x) &=& \inf\limits_{|z| \le dt} (d(t^n, x+z) + dt) \\ \forall x \in \Omega \qquad\quad d(t^{n+1}, x) &=& \sup\limits_{|z| \le dt} (d(t^n, x+z) - dt) \end{array} \tag{6.6}$$

Proof. First, thanks to the exact formula (6.4), it is easily checked that assumption (6.5) on d_0 implies that for all $t > 0$ and $x \in \mathbb{R}^d$

$$|d(t,x)| \geq |d_\Omega(x)|. \tag{6.7}$$

By symmetry, we restrict ourselves to showing the lemma in the case of a point $x \in \mathbb{R}^d \setminus \Omega$.

The result is proved by induction on $n \in \mathbb{N}$. Since it is clear for $n = 0$, assume the conclusion holds for some $n \in \mathbb{N}$. Two situations must then be distinguished:

• Assume first that $t^{n+1} < d(x, \partial\Omega)$. Then,

$$\begin{array}{rcl} d(t^{n+1},x) & = & \inf\limits_{|y|\leq t^{n+1}} \left(d_0(x+y) + t^{n+1}\right) \\ & = & \inf\limits_{|y|\leq t^n} \inf\limits_{|z|\leq dt} \left(d_0(x+y+z) + t^n + dt\right) \\ & = & \inf\limits_{|z|\leq dt} \left(d(t^n, x+z) + dt\right). \end{array}$$

the last equality holding because $d(x, \partial\Omega) > t^{n+1}$ implies that for all $|z| \leq dt$, $d(x+z, \partial\Omega) > t^n$.

• Suppose now that $t^{n+1} \geq d(x, \partial\Omega)$, and suppose dt is smaller than the width of the tubular neighborhood V of $\partial\Omega$ (on which d_Ω may be assumed smooth owing to theorem 6.1). We then have:

$$\begin{array}{rcl} \inf\limits_{|z|\leq dt} \left(d(t^n, x+z) + dt\right) & \geq & \inf\limits_{|z|\leq dt} \left(d_\Omega(x+z) + dt\right) \\ & = & d_\Omega(x) \\ & = & d(t^{n+1}, x). \end{array},$$

where the first line holds because of (6.7), and the last one because $t^{n+1} \geq d(x, \partial\Omega)$.

We are now left with the converse inequality. Because dt is smaller than the thickness of V, there exists $|z_0| \leq dt$ such that:

$$d(x + z_0, \partial\Omega) = d(x, \partial\Omega) - dt,$$

(take for instance $z_0 = -dt\, n(p)$, where p is a projection point of x over $\partial\Omega$). Then, $t^{n+1} \geq d(x, \partial\Omega)$ implies that $d(x + z_0, \partial\Omega) \leq t^n$. Hence:

$$\begin{array}{rcl} \inf\limits_{|z|\leq dt} \left(d(t^n, x+z) + dt\right) & \leq & d(t^n, x+z_0) + dt \\ & = & d(x+z_0, \partial\Omega) + dt \\ & = & d(x, \partial\Omega) - dt + dt \\ & = & d_\Omega(x) \end{array},$$

which concludes the proof. □

Suppose our computation is restricted to a bounded domain D enclosing Ω. We aim at computing function $d(t^N, .)$ over D for N large enough so that $t^N \geq \sup_{x\in D} |d_\Omega(x)|$. To this end, we use the iterative formulae supplied by the previous lemma, rearranging them a bit so as to make the decreasing property explicit, using the overestimation inequality (6.7): introduce, for $n = 0, 1, ...$, the sequence $\widetilde{d^n}$ of continuous functions over D, iteratively defined by $\widetilde{d^0} = d_0$, and

$$\forall x \in D \setminus \Omega\,,\ \widetilde{d^{n+1}}(x) = \inf\left(d(t^{n+1}, x), \widetilde{d^n}(x)\right) \tag{6.8}$$

$$\forall x \in \Omega\,,\ \widetilde{d^{n+1}}(x) = \sup\left(d(t^{n+1}, x), \widetilde{d^n}(x)\right) \tag{6.9}$$

From the above arguments, it follows that $\widetilde{d^n}$ is a sequence of continuous functions over D which converges pointwise to d_Ω. Furthermore, from its definition, it is clear that for $x \in D \setminus \Omega$, $\widetilde{d^n}(x)$ *decreases* from $d_0(x)$ to $d_\Omega(x)$ and that for $x \in \Omega$, $\widetilde{d^n}(x)$ *increases* from $d_0(x)$ to $d_\Omega(x)$. This sequence of functions - computed thanks to the iterative process expressed in lemma 6.1 - is the one we will try and approximate in the next section so as to end up with the desired signed distance function.

6.4 A numerical scheme for the signed distance function approximation

In this section, we propose a numerical method for computing the signed distance function to a polyhedral domain. The algorithm consists of two steps; the first one is purely geometric and amounts to identifying the simplices of the background mesh which intersect contour $\partial\Omega$, and the second one is purely analytical and is based on an explicit numerical scheme for solving the unsteady Eikonal equation (6.3).

6.4.1 Extending the signed distance function from the boundary

Given a polyhedral domain Ω, supplied by means of a simplicial mesh $\mathcal{S}$ of its boundary $\partial\Omega$, we intend to compute the signed distance function to Ω at every node x of a simplicial mesh $\mathcal{T}$ of a bounding box D.

More accurately, call $\mathcal{K} := \{K \in \mathcal{T}, K \cap \partial\Omega \neq \emptyset\}$ the set of the simplices of $\mathcal{T}$ which intersect the boundary $\partial\Omega$. We suppose that the signed distance function is initialized to its exact value in the nodes of those simplices $K \in \mathcal{K}$, and to a large value in the other nodes so that the initialization of the algorithm satisfies (6.5), at least in a discrete way; the implementation of this initialization is described in the next section 6.4.2. We then expect to devise an iterative numerical scheme on basis of formulae (6.6) to extend this signed distance function to the whole bounding domain.

To achieve this, we propose to approximate d_Ω by means of a $\mathbb{P}^1$ finite element function on mesh $\mathcal{T}$. Let dt be a time-step, $t^n = ndt$ $(n = 0, ...)$ and d^n be a $\mathbb{P}^1$ function intended as an approximation of the unique viscosity solution d to equation (6.3) at time t^n. We then iteratively compute d^n thanks to algorithm (1).

Algorithm 1 Extending the signed distance function field

1: Initialize the signed distance function d^0 with:

$$\begin{cases} d^0(x) &= \text{exact signed distance function to } \Omega \text{ if } x \text{ belongs to a simplex of } \mathcal{K} \\ d^0(x) &= d_{MAX} \text{ otherwise} \end{cases}$$

2: **for** $n = 1, ...$ until convergence **do**
3: $d^n(x) = d^{n-1}(x)$ for each node x of $\mathcal{T}$
4: **for** each simplex T of $\mathcal{T}$ **do**
5: **for** each node x of T which does not belong to a simplex in $\mathcal{K}$ **do**
6: **if** $x \notin \Omega$ **then**
7:

$$d^n(x) = \min\left(d^n(x), d^{n-1}\left(x - \frac{\nabla(d^{n-1}|_T)}{|\nabla(d^{n-1}|_T)|}dt\right)\right) + dt \tag{6.10}$$

8: **else**
9:

$$d^n(x) = \max\left(d^n(x), d^{n-1}\left(x + \frac{\nabla(d^{n-1}|_T)}{|\nabla(d^{n-1}|_T)|}dt\right)\right) - dt \tag{6.11}$$

10: **end if**
11: **end for**
12: **end for**
13: **end for**
14: **return** d^n

This algorithm needs some clarifying comments: for each step $t^n \to t^{n+1}$, we intend to mimic formulae (6.6), except that the infimum and supremum appearing there are difficult to compute in a discrete way.

Assuming time step dt to be small enough, given for example a node x of $\mathcal{T}$, $x \in D \setminus \Omega$ and $n \geq 0$ the simplest approximation of $\inf_{|z| \leq dt} (d^n(x+z) + dt)$ is achieved considering the gradient of d^n at x. However, this gradient is a constant-per-simplex vector, and is obviously discontinuous at the interface between two adjacent simplices of $\mathcal{T}$. In particular, it is irrelevant to talk about its value at a node x of $\mathcal{T}$. The most natural way to discretize formula (6.6) is then, taking into account the numerical discretization of the handled quantities:

$$\inf_{|z| \leq dt} (d^n(x+z) + dt) \approx \inf_{T \in \mathcal{B}(x)} d^n \left(x - \frac{\nabla (d^{n-1}|_T)}{|\nabla (d^{n-1}|_T)|} dt \right) + dt, \tag{6.12}$$

where $\mathcal{B}(x)$ is the set of simplices of mesh $\mathcal{T}$ containing x as vertex.

The rest of algorithm (1) is merely a discrete version of formulae (6.8) and (6.9). See figure 6.1 for a visual intuition of the process.

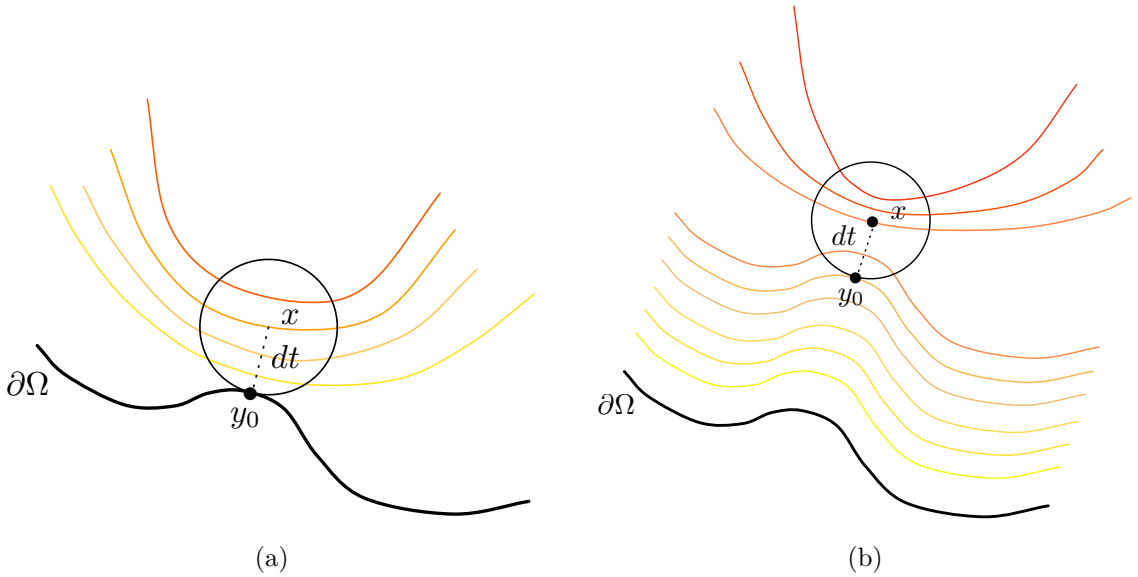
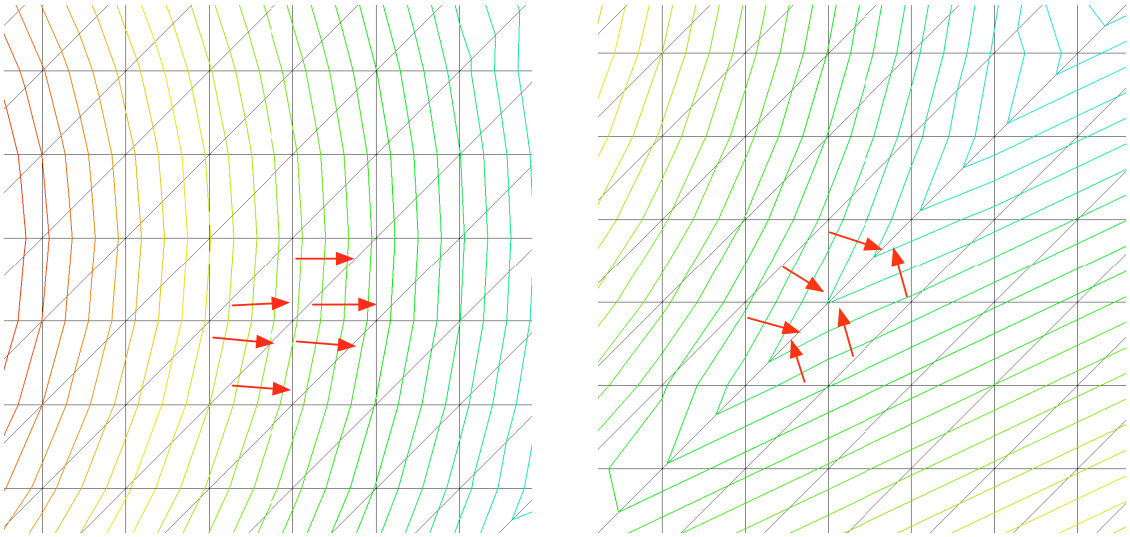


Figure 6.1: At a given iteration n, the proposed numerical scheme amounts to 'straighten up' the value of d^n at point x from its value at point y_0 in such a way that $d^n(y_0) = \inf_{y \in B(x,dt)} d^n(y)$ with the property of unitary gradient, *(a)* e.g. for a point x at distance dt from $\partial\Omega$, $d^1(x) = d_0(y_0) + dt = dt = d(x, \partial\Omega)$. *(b)* The property of unit gradient 'propagates' from the boundary $\partial\Omega$, near which values of d^n are 'regularized' at an early stage.

Remark 6.1. Actually, a formal study of the characteristic curves of equations (6.1) and (6.3) would have brought more or less the same numerical scheme. In that scope, the points $x \in D$ where d_Ω fails to be smooth can be interpreted as the crossing points of different characteristic curves of Eikonal equation (6.1); see figure (6.2). At nodes x of mesh $\mathcal{T}$ close to such kinks of the signed distance function, the discretization (6.12) expresses the idea that each one of these crossing characteristic curve is backtracked.

6.4.2 Initialization of the signed distance function near $\partial\Omega$

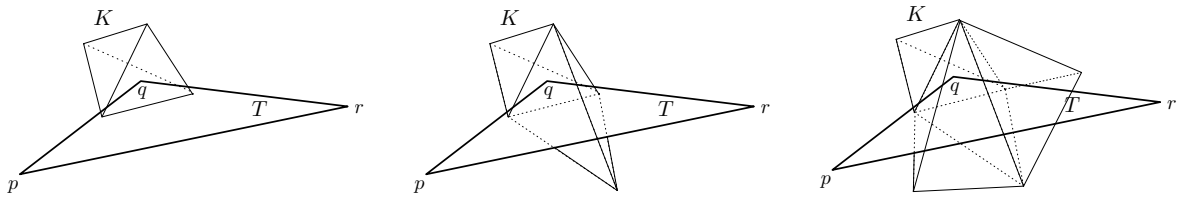
Before extending the signed distance function from the boundary $\partial\Omega$ of the input polyhedral domain, we need first to detect those simplices $K \in \mathcal{T}$ intersecting $\partial\Omega$. This is achieved by scanning each surface triangle $T \in \mathcal{S}$ in three dimensions (segment in two dimensions), storing a background mesh simplex $K \in \mathcal{T}$



(a) (b)

Figure 6.2: *(a)* Continuity of the gradient in the areas where the signed distance function is regular; *(b)* discontinuity of the gradient at a node close to the skeleton.

containing one of the three nodes of T, then travelling the background mesh from K by adjacency, advancing only through faces which intersect T. Figure 6.3 illustrates this step. A computationally efficient algorithm for the three-dimensional triangle-triangle overlap test, relying only on algebraic predicates, developed in [164] is used to this end.



(a) (b) (c)

Figure 6.3: Identification of the background simplices intersecting a triangle $T = (pqr) \in \mathcal{S}$: *(a)* one starts with a simplex K of $\mathcal{T}$ containing one of the points p, q or r then *(b)* marches through the faces of K which intersect T and *(c)* goes on, stopping when there is no more simplex of K to add.

Then, at each node of K which belongs to such an intersecting simplex, we initialize the (still unsigned) distance function to its exact value. See [185] for an efficient algorithm computing a point-triangle distance in three dimensions. This can be done at a relatively cheap cost since a node x of $\mathcal{K}$ lies 'close' to $\partial\Omega$ and only very few triangles of $\mathcal{S}$ have to be processed to identify the one with respect to which $d(x, \partial\Omega)$ is reached. In all the other nodes of $\mathcal{T}$, we assign an arbitrary large value (e.g. larger than the domain size).

This leaves us with initializing the sign. Surprisingly enough, this stage happens to be the most tedious one of the initialization process, all the more so as it is barely considered in the literature (see nevertheless [292] for another approach, based on an octree grid refinement). We propose here a purely logical algorithm

(2) based on a progression by 'layers', which relies on two piles *Layer* and *Boundary*, and an integer *Sign*. It is very similar to the classical coloring techniques used to recover the connected components of a configuration in a Delaunay meshing context [145]; see figure 6.4.

Algorithm 2 Signing the unsigned initial distance field

1: Initialize the pile *Layer* with a simplex $K \in \mathcal{T}$ which is in the 'most external' connected component of the configuration (for instance a simplex in the corner of the bounding box).
Initialize *Sign* to $+1$.
2: **while** an element K in pile *Layer* as not been inspected **do**
3: Consider every neighbour K' of K: if K' has not yet been inspected, and does not belong to $\mathcal{K}$, put K' in *Layer*; if K' belongs to $\mathcal{K}$, put it into pile *Boundary*.
4: **end while**
5: **while** an element K in *Boundary* has not yet been inspected **do**
6: Consider every neighbour K' of K: if K' has not yet been inspected, and belongs to $\mathcal{K}$, put K' in pile *Boundary* (this step ensures pile *Boundary* contains *all* the boundary elements corresponding to the ongoing layer).
7: **end while**
8: Attribute the current sign of *Sign* in each vertex of each element of pile *Stratum*.
9: $Sign \leftarrow -Sign$.
10: Clean pile *Layer*.
11: **for** K is in pile *Boundary* **do**
12: Consider every neighbour K' of K: if K' has not yet been inspected, put it into pile *Layer*: then we get all the new starting points of the new layer (possibly associated to different connected components of Ω.
13: **end for**
14: Clean pile *Boundary*.
15: Go back to step 2.

So as to enhance the numerical efficiency of the proposed method, several improvements may be proposed. First, note that the propagating scheme of Algorithm 1 is inherently parallel: at a given step, the operations carried out in a background simplex K are independant from those carried out in another such simplex. Furthermore, the time step dt must be chosen small enough at the beginning of the process, so that going back along the characteristic lines does not lead to crossing the interface $\partial\Omega$ and picking irrelevant values. But after a certain amount of iterations, we can obviously increase this time step. Eventually, note that Theorem 6.2 expresses the fact that the information propagates from $\partial\Omega$ to the whole space with a unit velocity - and that, on the theoretical framework, at a given point $x \in \mathbb{R}^d$, $d(s,x) = d_\Omega(x)$ at any time $s > d(x, \partial\Omega)$. According to this observation, we decided to consider the values computed at a node x fixed when these values are smaller than the current total time of the propagation (with a security margin).

6.5 Mesh adaptation for a sharper approximation of the signed distance function

The proposed numerical method in section 6.4 approximates the signed distance function to Ω by means of a $\mathbb{P}^1$ finite elements function; therefore it seems only natural to attempt to decrease the interpolation error of this function on the background mesh $\mathcal{T}$. Moreover, if we intend to use the signed distance function to Ω as an implicit function defining Ω in a context inscribed in an evolution process described using the level set method, we may want the 0 level set of the approximate signed distance function (which is intended to be close to the $\mathbb{P}^1$ interpolant of this function) to match the true boundary $\partial\Omega$ in the best possible way. Thus

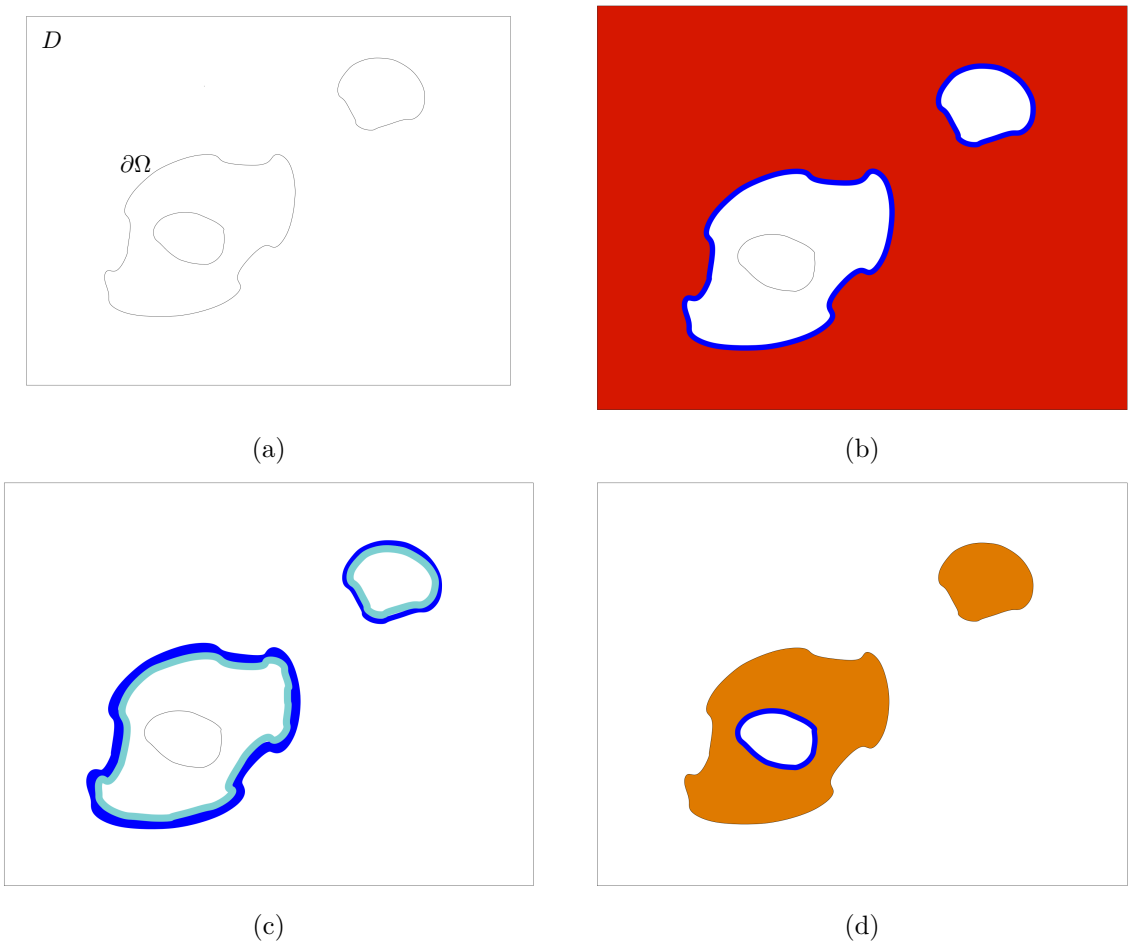


Figure 6.4: Signing the unsigned initial distance field; *(a)* a contour $\partial\Omega$, *(b)* start from a triangle of the computational mesh $\mathcal{T}$ that is known to be in the exterior to the domain Ω, travel $\mathcal{T}$ by adjacency and recover the first (outer) layer (in red), and the simplices of $\mathcal{T}$ intersecting the part of $\partial\Omega$ connected to this first layer (in blue) *(c)* Get all the triangles of $\mathcal{T}$ (in pale blue) that are the different starting points for the new (now interior) layer, *(d)* travel again by adjacency to get this new layer (in orange), as well as the new simplices of $\mathcal{T}$ intersecting the part of $\partial\Omega$ connected to this layer (in blue).

it can be interesting to couple the computation of the signed distance function with a process of adaptation of the background mesh $\mathcal{T}$. Actually, we will see that adapting $\mathcal{T}$ in the same way leads to an improvement as regards both problems.

6.5.1 Anisotropic mesh adaptation

Mesh adaptation basically lies on the fact that the most efficient way to refine a mesh so as to increase the efficiency of computations is to adjust the directions and sizes of its elements in agreement with the variations of the functions under consideration. Thus, significant improvements can be achieved in accuracy, while the cost of the computations (which is in close relation with the total number of elements of the mesh) is kept minimal, and conversely.

Numerous methods exist as regards the very popular topic of mesh adaptation, some of them relying on the concepts of Riemannian metric [311]: the local desired size, shape and orientation related information at a node x of mesh $\mathcal{T}$ is stored in a metric tensor field $M(x)$, prescribed by an error indicator or an error estimate which can arise from various possible preoccupations: a posteriori geometric error estimates, analytic error estimates, etc... (see for instance [6], [25], [180]). For the sake of convenience, we now briefly recall the useful features of this paradigm in the rest of this chapter.

Given a metric tensor field $M(x)$ defined at each point $x \in \mathbb{R}^d$, (notice that in practice, $M(x)$ is defined only at the nodes of $\mathcal{T}$ and then interpolated from these values [145]) we consider respectively the *length* $\ell_M(\gamma)$ of a curve $\gamma : [0,1] \to \mathbb{R}^d$ and the *volume* $V_M(K)$ of a simplex K in the Riemannian space $\left(\mathbb{R}^d, M\right)$:

$$\ell_M(\gamma) = \int_0^1 \sqrt{\langle M(\gamma(t))\gamma'(t), \gamma'(t)\rangle} dt \ , \ V_M(K) = \int_K \sqrt{det(M(x))} dx$$

and aim at modifying the mesh $\mathcal{T}$ so as to make it *quasi-unit* with respect to the metric $M(x)$, that is to say all its simplices K have edges lengths lying in $\left[\frac{1}{\sqrt{2}}, \sqrt{2}\right]$ and an *anisotropic quality measure*:

$$\mathcal{Q}_M(K) := \alpha_d \frac{V_M(K)^2}{\left(\sum_{i=1}^{na} \ell_M(e_i)^2\right)^d}$$

close to 1 (where $na = d(d+1)/2$ is the number of edges of a d-dimensional simplex, e_i are the edges of K and α_d is a normalization factor). For instance, in the particular case when $M(x)$ is constant over $\mathbb{R}^d$, it can be characterized by the ellipsoid:

$$\Phi_M(1) = \left\{v \in \mathbb{R}^d, \ \ell_M(v) = 1\right\}$$

of unit vectors with respect to M, and a simplex K with unit edges is simply a simplex which can be inscribed in this ellipsoid.

Several techniques have been devised for generating anisotropic meshes according to a metric tensor field, that can be roughly classified into two categories. On the one hand, global methods, such as Delaunay-based methods and advancing-front methods, perform the same kind of operations as in the classical case with adapted notions of length and volume. On the other hand, local mesh modification methods [115] start from an existing non-adapted mesh and adapt it so that it fits at best the above conditions (see the outline of some of the main methods for mesh generation in chapter 3). The approach used in this paper falls into the second category.

6.5.2 Computation of a metric tensor associated to the minimization of the $\mathbb{P}^1$ interpolation error

Let us recall the following general result (whose proof lies in [103] [24]) about the L^∞ interpolation error of a smooth function ϕ on a simplicial mesh by means of a $\mathbb{P}^1$ Lagrange finite element function.

Theorem 6.3. *Let $\mathcal{T}$ be a simplicial mesh of large computational mesh $D \subset \mathbb{R}^d$ ($d = 2$ or 3) and ϕ a $\mathcal{C}^2$ function on D. Let $\mathcal{C}(D)$ the set of continuous functions over D, $V_\mathcal{T} \subset \mathcal{C}(D)$ the space of continuous functions on D whose restriction to every simplex of $\mathcal{T}$ is a $\mathbb{P}^1$ function, and denote by $\pi_\mathcal{T} : \mathcal{C}(D) \to V_\mathcal{T}$ the usual $\mathbb{P}^1$ finite element interpolation operator. Then for every simplex $K \in \mathcal{T}$,*

$$||\phi - \pi_\mathcal{T}\phi||_{L^\infty(K)} \leq \frac{1}{2}\left(\frac{d}{d+1}\right)^2 \max_{x\in K} \max_{y,z\in K} \langle |\mathcal{H}(\phi)|(x)yz, yz\rangle,$$

where $\langle .,. \rangle$ *stands for the usual scalar product on* $\mathbb{R}^d$*, and, for a symmetric matrix* $S \in \mathcal{S}_d(\mathbb{R})$*, which admits the following diagonal shape in orthonormal basis* $S = P \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_d \end{pmatrix} P^T$*, we denote*

$$|S| := P \begin{pmatrix} |\lambda_1| & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & |\lambda_d| \end{pmatrix} P^T.$$

Relying on this theorem, we can build a metric tensor field $M(x)$ on D that allows for an accurate control of the L^∞ interpolation error of any smooth enough function ϕ (recall that the main feature of this interpolation error is that it is an upper bound for the approximation error of ϕ by means of a finite elements calculus with the space $V_{\mathcal{T}}$, at least in the case of elliptic problems [91]): so as to get

$$\forall K \in \mathcal{T},\ ||\phi - \pi_{\mathcal{T}}\phi||_{L^\infty(K)} \leq \varepsilon$$

for a prescribed margin $\varepsilon > 0$, we urge the shape of an element K of $\mathcal{T}$ to behave in such a way that:

$$\max_{y,z \in K} \langle \widetilde{\mathcal{H}} yz, yz \rangle \leq c\varepsilon$$

where c is a constant which only depends on the dimension, stemming from theorem (6.3) and $\widetilde{\mathcal{H}}$ is the mean value (or an approximation) of the metric tensor $|\mathcal{H}(\phi)|$ over element K. This leads to defining the desired metric tensor $M(x)$ at each node x of $\mathcal{T}$ by (see [5] [6]):

$$M(x) = P(x) \begin{pmatrix} \widetilde{\lambda_1} & 0 & 0 \\ 0 & \widetilde{\lambda_2} & 0 \\ 0 & 0 & \widetilde{\lambda_3} \end{pmatrix} P(x)^T \tag{6.13}$$

where

$$\widetilde{\lambda_i} = \min(\max(\frac{c|\lambda_i|}{\varepsilon}, \frac{1}{{h_{max}}^2}), \frac{1}{{h_{min}}^2}), \quad \widetilde{|\mathcal{H}(\phi)|}(x) = P(x) \begin{pmatrix} |\lambda_1| & 0 & 0 \\ 0 & |\lambda_2| & 0 \\ 0 & 0 & |\lambda_3| \end{pmatrix} {}^tP(x)$$

being an approximation of the Hessian of u around node x, written here in diagonal form in an orthonormal basis, h_{min} (resp. h_{max}) being the smallest (resp. largest) size allowed for an element in any direction, and c being the above constant.

This principle can be applied to the signed distance function $\phi = d_\Omega$, at least in a formal way (we saw during chapter 4 that d_Ω is smooth except at the skeleton of Ω). This produces a mesh adaptation procedure thanks to which the interpolation error of d_Ω is decreased, and hopefully the numerical scheme presented in section 6.4 leads to a closer approximation of d_Ω.

6.5.3 Mesh adaptation for a geometric reconstruction of the 0 level set of a function

In this section, we consider a bounded domain $\Omega \subset \mathbb{R}^d$, implicitly defined by a function $\phi : \mathbb{R}^d \to \mathbb{R}$ (which does not necessarily match with d_Ω for the moment), that is:

$$\Omega = \left\{x \in \mathbb{R}^d,\ \phi(x) < 0\right\}\ ;\ \partial\Omega = \left\{x \in \mathbb{R}^d,\ \phi(x) = 0\right\}\ ;\ {}^c\overline{\Omega} = \left\{x \in \mathbb{R}^d,\ \phi(x) > 0\right\}, \tag{6.14}$$

and we want to adapt the background mesh $\mathcal{T}$ so that the 0 level set of the function $\pi_{\mathcal{T}}\phi$ obtained from u by $\mathbb{P}^1$ finite elements interpolation, say $\partial\Omega_{\mathcal{T}}$, is as close as possible to the 0 level set of the true function ϕ, in terms of the Hausdorff distance (for a review of the various methods for discretizing an implicit surface, see [161]). Then we will apply these results to the particular case when u is the signed distance function to Ω. Throughout this section, we follow the work [119].

Definition 6.3. *Let K_1, K_2 two compact subsets of $\mathbb{R}^d$. For all $x \in \mathbb{R}^d$, denote $d(x, K_1) = \inf_{y \in K_1} d(x, y)$ the Euclidean distance from x to K_1. We define:*

$$\rho(K_1, K_2) := \sup_{x \in K_1} d(x, K_2)$$

and the Hausdorff distance between K_1 and K_2 , denoted by $d^H(K_1, K_2)$, is the quantity:

$$d^H(K_1, K_2) := \max(\rho(K_1, K_2), \rho(K_2, K_1)) .$$

The following lemma proves useful when it comes to measuring the distance to Ω relying on *any* implicitly-defining function ϕ in the sense that (6.14) holds:

Lemma 6.2. *Let ϕ a $\mathcal{C}^1$ function on $\mathbb{R}^d$. Assume there exists a tubular neighborhood $V = \left\{x \in \mathbb{R}^d,\ |\phi(x)| < \alpha\right\}$ for some $\alpha > 0$, such that ϕ does not exhibit any critical point in V. Then, $\partial\Omega$ is a submanifold of $\mathbb{R}^d$, and Ω is a bounded subdomain of $\mathbb{R}^d$ with $\mathcal{C}^1$ boundary. For any point $x \in V$ we have the estimate:*

$$d(x, \partial\Omega) \leq \frac{\sup_{z \in V} |\nabla\phi(z)|}{\inf_{z \in V} |\nabla\phi(z)|^2} |\phi(x)| \tag{6.15}$$

Proof. Without loss of generality, we may limit ourselves to considering any point $x \in V \cap {}^c\overline{\Omega}$.

The proof consists in 'going backwards' following the velocity field $\nabla\phi$ until going against $\partial\Omega$, then in estimating the distance between x and the contact point with $\partial\Omega$. To achieve this, introduce the integral curve γ associated of $\nabla\phi$ emerging from x, defined as a solution to the ODE:

$$\begin{cases} \gamma(0) = x \\ \gamma'(s) = -\nabla\phi(\gamma(s)) \end{cases} \tag{6.16}$$

It is easily seen that the curve $s \mapsto \gamma(s)$ is defined (at least) over a period of the form $[0, s_1)$, for some $s_1 > 0$.and there exists a real number $0 < s_0 < s_1$, such that $y := \gamma(s_0)$ belongs to $\partial\Omega$. We then have:

$$\begin{aligned} \phi(x) &= \phi(x) - \phi(y) \\ &= \int_{s_0}^{0} \langle \nabla\phi(\gamma(s)), \gamma'(s) ds \rangle \\ &= \int_{s_0}^{0} |\nabla\phi(\gamma(s))|^2 ds \end{aligned}$$

and thus obtain

$$|s_0| \inf_{z \in V} |\nabla\phi(z)|^2 \leq |\phi(x)| \tag{6.17}$$

On the other hand, we also get

$$x - y = -\int_{s_0}^{0} \nabla\phi(\gamma(s)) ds$$

and

$$|x - y| \leq |s_0| \sup_{z \in V} |\nabla\phi(z)|. \tag{6.18}$$

Eventually, with (6.17) and (6.18), we conclude that:

$$d(x, \partial\Omega) \leq |x - y| \leq \frac{\sup_{z \in V} |\nabla\phi(z)|}{\inf_{z \in V} |\nabla\phi(z)|^2} |\phi(x)|.$$

□

Note that formula (6.16) expresses the idea that the closer ϕ is to the signed distance function to Ω (or a fixed multiple of it), the more reliable the evaluation of ϕ as an estimate of the Euclidean distance to the boundary $\partial\Omega$.

We are now ready to derive a formal estimate of the Hausdorff distance between $\partial\Omega$ and $\partial\Omega_{\mathcal{T}}$. Take a point $x \in \partial\Omega_{\mathcal{T}}$, which belongs to a simplex K of the background mesh $\mathcal{T}$. Using lemma 7.2 formally, we have:

$$\begin{aligned} d(x,\partial\Omega) &\leq \frac{\sup\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|}{\inf\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|^2} |\phi(x)| \\ &= \frac{\sup\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|}{\inf\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|^2} |\phi(x) - \pi_{\mathcal{T}}\phi(x)| \end{aligned}$$

and it yields, thanks to theorem 6.3

$$d(x,\partial\Omega) \leq c \frac{\sup\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|}{\inf\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|^2} \max_{x\in K} \max_{y,z\in K} \langle |\mathcal{H}(\phi)|(x)yz, yz\rangle$$

where c is a scalar constant which only depends on d. Thus, we find

$$\rho(\partial\Omega_{\mathcal{T}},\partial\Omega) \leq c \frac{\sup\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|}{\inf\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|^2} \max_{K\in\mathcal{T}} \max_{x\in K} \max_{y,z\in K} \langle |\mathcal{H}(\phi)|(x)yz, yz\rangle.$$

By the same token, formally applied to a point $x \in \partial\Omega$, since $\partial\Omega_{\mathcal{T}} = \left\{x \in \mathbb{R}^d,\ \pi_{\mathcal{T}}\phi(x) = 0\right\}$, we have

$$\rho(\partial\Omega,\partial\Omega_{\mathcal{T}}) \leq c \frac{\sup\limits_{K\in\mathcal{T}} |\nabla(\pi_{\mathcal{T}}\phi)|_K|}{\inf\limits_{K\in\mathcal{T}} |\nabla(\pi_{\mathcal{T}}\phi)|_K|^2} \max_{K\in\mathcal{T}} \max_{x\in K} \max_{y,z\in K} \langle |\mathcal{H}(\phi)|(x)yz, yz\rangle.$$

Eventually, neglecting the discrepancy between $|\nabla\phi|$ and $|\nabla\pi_{\mathcal{T}}\phi|$ yields:

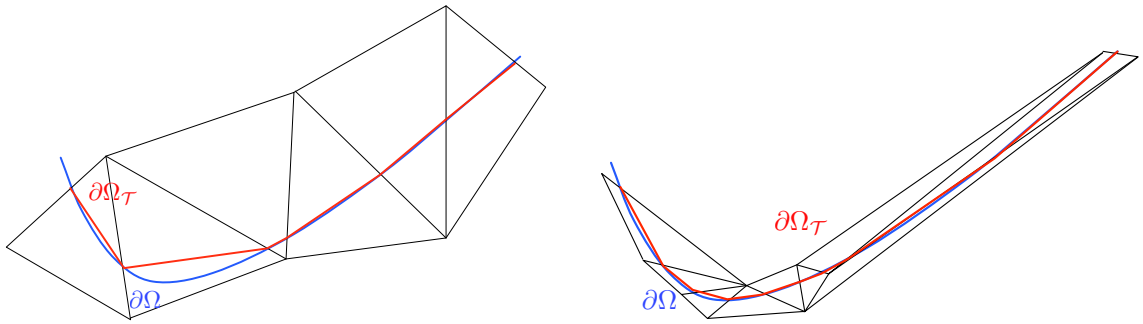
$$d^H(\partial\Omega,\partial\Omega_{\mathcal{T}}) \leq c \frac{\sup\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|}{\inf\limits_{y\in\mathbb{R}^d} |\nabla\phi(y)|^2} \max_{K\in\mathcal{T}} \max_{x\in K} \max_{y,z\in K} \langle |\mathcal{H}(\phi)|(x)yz, yz\rangle.$$

We observe that this estimate is very similar to the result given by theorem 6.3, and especially that it involves the Hessian matrix of ϕ; thus, with the same metric tensor field (7.21) as the one associated to the control of the $\mathbb{P}^1$-interpolation error of d_Ω on $\mathcal{T}$, we achieve control of the Hausdorff distance between the exact boundary $\partial\Omega$ and its piecewise affine reconstruction $\partial\Omega_{\mathcal{T}}$ as the 0 level set of the interpolate d_Ω. Of course, depending on where we need the accuracy, we can restrict ourselves to prescribing metric (7.21) only in certain areas of $\mathbb{R}^d$ (e.g. in a level set context, in a narrow band near the boundary, or in the vicinity of a particular level set of ϕ).

This result admits a rather interesting geometric interpretation in the case when $\phi = d_\Omega$, i.e. when ϕ is the signed distance function to Ω. In that case, it is well-known [43] the second fundamental form (oriented in the sense that it is positive definite at a point $x \in \partial\Omega$ near which $\partial\Omega$ is locally convex) reads, for any point $x \in \partial\Omega$:

$$\forall \xi \in T_x\partial\Omega,\ \ \mathbf{II}_x(\xi,\xi) = \langle \mathcal{H}d_\Omega(x)\xi, \xi\rangle$$

where $\langle .,. \rangle$ denotes the usual Euclidean scalar product of $\mathbb{R}^d$. Hence, the eigenvalues of $\mathcal{H}d_\Omega(x)$ are the two principal curvatures of the surface $\partial\Omega$, associated with the two principal directions at point x. In such case, the above estimates mean that, understandably enough, so as to get the best reconstruction of $\partial\Omega$, we have to align the circumscribed ellipsoids of the simplices of the background mesh $\mathcal{T}$ with the curvature of this surface (see Figure (6.5)).



(a) (b)

Figure 6.5: Piecewise affine reconstruction $\partial\Omega_\mathcal{T}$ of $\partial\Omega$ with *(a)* a regular background mesh $\mathcal{T}$ and *(b)* an adapted anisotropic background mesh $\mathcal{T}$.

6.6 A remark about level-set redistancing

When tracking the evolution of an interface by means of the level set method - e.g. when dealing with multi-phase flow systems - a crucial issue is to initialize and maintain the level set function under consideration $\phi^n = \phi(t^n, .)$ ($t^n = ndt$, where dt is the time step of the process) as close as possible to the signed distance function to the zero level set $\partial\Omega^n$ so defined, while it tends to get very far from it within very few iterations (see e.g. [217] about the relevance of this feature in the context of multi-phase flows). To achieve this, a so-called *redistancing* step must be carried out [89]. Unfortunately, it is worth emphasizing this problem is ill-posed since it is impossible to 'regularize' function ϕ^n to make it close to a distance function $\widetilde{\phi^n}$ - i.e. $|\nabla\widetilde{\phi^n}| \approx 1$ - without altering the zero level-set which then becomes $\widetilde{\partial\Omega^n} := \left\{x \in \mathbb{R}^d;\ \widetilde{\phi^n}(x) = 0\right\}$. Several approaches exist to address this issue, depending on the applications they are suited for and thus on the features of the interface $\partial\Omega^n$ which must be retained (see e.g. [236] [299] [298] [242]).

Given an iteration n when this process is carried out, most of the existing approaches consist in solving the *unsteady Eikonal equation* (6.3), with initial data ϕ^n and over a short time period, so that the obtained solution enjoys the desired unit gradient property, at least in a neighborhood of the tracked interface. To this end, in [244] [298] [81], an approximation of the sign function appearing in equation (6.3) by a smooth, steep function is introduced. The overall process is very fast, since the only performed operations are a few iterations of an often explicit scheme for equation (6.3). In particular, this trick of approximating the sign function enables not to regenerate an exact distance function near the boundary similarly to what we explained in section 6.4.2, which can be costly if the computational mesh is large (see section 6.8). The drawback is that the shift in the considered interface is not controlled.

Hence, we limit ourselves to the case of an adaptation process, where at each step, the background mesh $\mathcal{T}$ is adapted to the zero level-set of function ϕ^n, by prescribing the metric tensor field (7.21) at least in a narrow band near the interface (see [63] for an example). Here, implementing the redistancing step $\phi^n \to \widetilde{\phi^n}$ by taking for $\widetilde{\phi^n}$ the signed distance function to Ω^n, computed by means of the algorithm devised

in section (6.4), provides a true signed distance function $\widetilde{\phi^n}$, while ensuring the shift of the interface is controlled by:

$$d^H(\partial\Omega^n, \widetilde{\partial\Omega^n}) \leq \varepsilon.$$

Of course, this process is slower than the one discussed above, but we believe it can be of interest when an close control of the change in interfaces $\partial\Omega^n \to \widetilde{\partial\Omega^n}$ is sought.

6.7 Extension to the computation of the signed distance function in a Riemannian space

The proposed method admits a straightforward extension in the case we want to compute the signed distance function d_Ω^M from a subdomain Ω of $\mathbb{R}^d$, $\mathbb{R}^d$ *being endowed with a Riemannian metric* M, that is,

$$d_\Omega^M(x) = \begin{cases} -d_M(x, \partial\Omega) & if\ x \in \Omega \\ 0 & if\ x \in \partial\Omega \\ d_M(x, \partial\Omega) & if\ x \in^c \overline{\Omega} \end{cases},$$

where

$$d_M(x,y) = \inf\left\{\ell_M(\gamma), \gamma : [0,1] \to \mathbb{R}^d \text{ piecewise differentiable }, \gamma(0) = x\ ;\ \gamma(1) = y\right\}$$

is the *distance* from x to y in the space $(\mathbb{R}^d, M)$ and $d_M(x, \partial\Omega) = \inf_{y\in\partial\Omega} d_M(x,y)$ is the *(unsigned) distance function from* x *to* $\partial\Omega$. For another approach, based on the Fast Marching Method, see [226] [194].

Indeed, the function d_Ω^M is a solution to the *Eikonal equation in the Riemannian space* $(\mathbb{R}^d, M)$ in the sense of viscosity [214]:

$$\sqrt{\langle M^{-1}(x)\nabla d(x), \nabla d(x)\rangle} = 1$$

Considering a continuous function d_0 which implicitly defines Ω in the sense that (6.2) holds, we have the corresponding unsteady equation:

$$\begin{cases} \dfrac{\partial d}{\partial t}(t,x) + \operatorname{sgn}(d_0(x))(\sqrt{\langle M^{-1}(x)\nabla d(t,x), \nabla d(t,x)\rangle} - 1) = 0 \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad d(t=0,x) = d_0(x) \end{cases}$$

and the same study as in section (6.3) yields the following approximation formulae for computing the solution d to this equation; for $t > 0$, a small time step dt, and $x \in {}^c\overline{\Omega}$:

$$d(t+dt,x) \approx d\left(t, x - dt\frac{M^{-1}(x)\nabla d(t,x)}{\sqrt{\langle M^{-1}(x)\nabla d(t,x), \nabla d(t,x)\rangle}}\right) + dt$$

and, by symmetry, for $x \in \Omega$:

$$d(t+dt,x) \approx d\left(t, x + dt\frac{M^{-1}(x)\nabla d(t,x)}{\sqrt{\langle M^{-1}(x)\nabla d(t,x), \nabla d(t,x)\rangle}}\right) - dt$$

which raises a numerical scheme similar to the scheme presented in section 6.4 (see Figure 6.9 for an example).

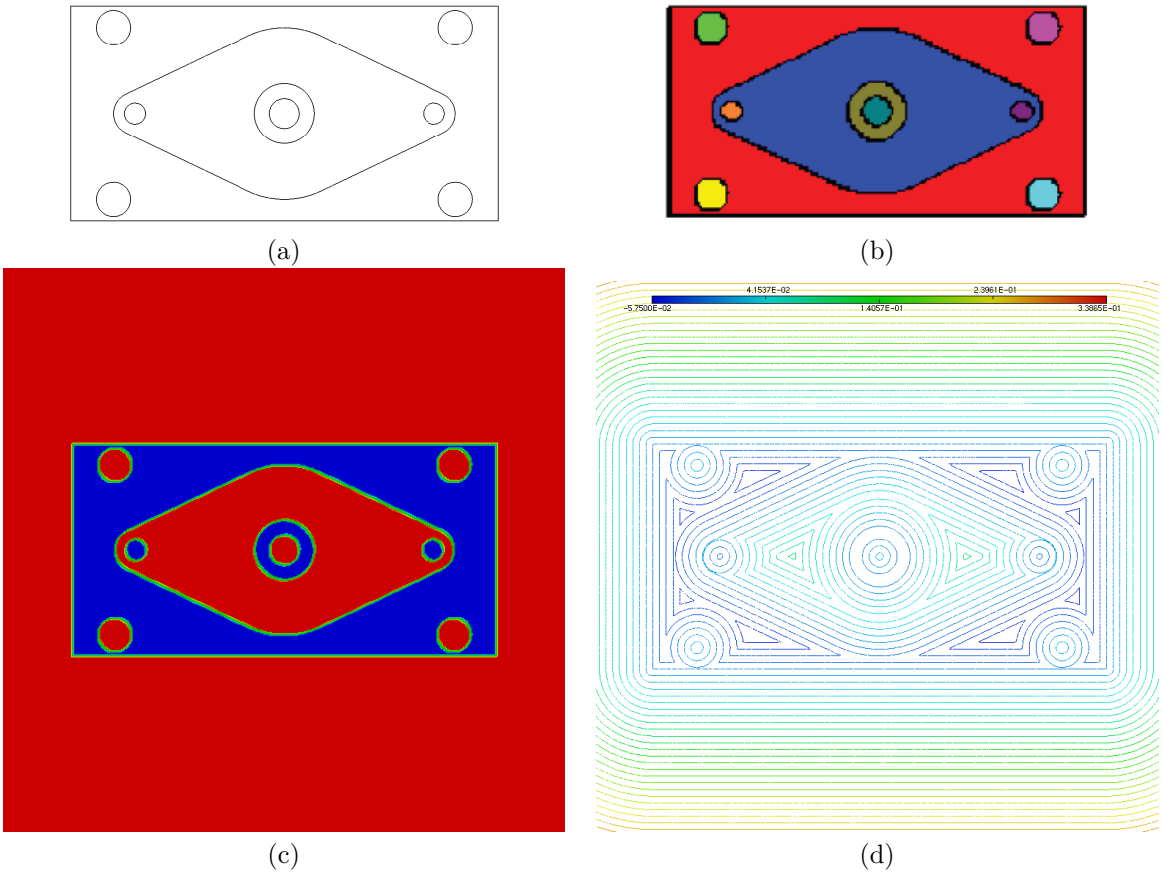
6.8 Numerical Examples

We now present several examples to assess the main theoretical issues presented in the previous sections. All the following computations are held on contours embedded in a bounding box which is a unit square in two dimensions, or a unit cube in three dimensions, and are scaled if need be.

Example 1

We give a two-dimensional example of the computation of the signed distance function to the contour represented in Figure 6.6, and carry out two numerical experiments.

We first hold computations on unstructured, yet non-adapted simplicial computational meshes $\mathcal{T}$ of bounding box D, of increasing sizes, so as to assess both convergence and scaling of the method. For all these examples, a time step $dt = 0.001$ (according to the smallest mesh size among the presented meshes) is used for the steps $1 \leq n \leq 80$, and the computation is finished with a time step $dt = 0.004$. Table 6.8 displays several features of the computation (number of vertices of the mesh, CPU time) as well as errors measured in several norms, and an inferred approximate order for the scheme. To evaluate these errors, the exact signed distance function (or more accurately its $\mathbb{P}^1$-interpolate on the mesh at stake) is computed by a brute force approach - i.e. by computing the minimum distance to a segment of the mesh of the contour at every node of the background mesh $\mathcal{T}$ - and is then compared with the numerical approximation obtained using our scheme. All our computations are held on an OPTERON 6100, 2Ghz. Figure 6.6 displays the result of the computation held on the finest grid - identification of the connected components, initialization of the sign and computation of the signed distance function.



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

Figure 6.6: A two-dimensional example on a regular mesh; *(a)* the considered contour, *(b)* the connected components of the contour, *(c)* signs of these components: red for positive, blue for negative, green for the boundary, *(d)* some isolines of the signed distance function.

Let us make some comments at this point. We solve the *time-dependent* Eikonal equation, so that at first

Number of Vertices	$\|\|e\|\|_{L^\infty}$	Order	$\|\|e\|\|_{L^1}$	Order	$\|\|e\|\|_{L^2}$	Order	CPU Time(s)
2601	0.013723	–	0.000924	–	0.002474	–	0.167
10201	0.008922	0.63	0.000644	0.53	0.001687	0.56	0.462
40401	0.004971	0.74	0.000340	0.73	0.000932	0.71	1.362
74310	0.001731	1.23	0.000204	0.90	0.000476	0.98	2.974
296740	0.001790	0.86	0.000095	0.96	0.000232	0.99	10.293

Table 6.1: Computation of the signed distance function to the contour 6.6 (a) on non-adapted, unstructured meshes

glance, it could be relevant to investigate both time and spatial accuracy of the proposed numerical scheme. Actually, it turns out that the spatial accuracy of the scheme is by far the most critical issue as regards convergence inasmuch as, provided time step dt is small enough (and in all the test-cases we implemented, taking dt of the order of the mesh size proved sufficient), the quality of the final result only depends on the closeness of spatial approximation.

As far as the spatial convergence of this scheme is concerned, one observes that, understandably enough, it happens to be at most first-order. Is seems to behave comparably to the algorithm presented in [236] or to the simplicial version of the Fast Marching algorithm proposed in [194] or of the Fast Sweeping algorithm, described in [262]. Note however that the two latter schemes are probably a bit faster as regards computational time, given they achieve convergence within a fixed (or very limited) number of iterations that are linear in the number of vertices of the mesh. However, we believe that differences in the architectures of the computers used to run the proposed examples are tremendous and do not allow for a meaningful comparison between the execution times of the proposed algorithms. This scheme is also bound to be slower than the ones devised in the case of cartesian computational grids [276, 292, 298, 332], which enjoy immediate standard operation algorithms (e.g. search of the element to which a given point belongs, etc...). What is more, higher-order distancing or redistancing numerical schemes are available in this Cartesian frame [81], whereas, to our knowledge, it is not this case in the unstructured case, which we believe to be of independent relevance.

Now, we adapt the background mesh $\mathcal{T}$ to the computation of the signed distance function to the same contour, relying on the principles enonciated in section 6.5.1, in the vicinity of the boundary $\partial\Omega$ of the considered domain. Therefore, we are only interested in getting a close approximation of this signed distance function near the boundary. We use parameters $\varepsilon = 0.001$ and $h_{min} = 0.001$. This yields the results of figure 6.7, whose features (on one CPU) are reported in table 6.8.

Number of Iterations	Number of Nodes	Number of Elements	CPU Time (Seconds)
236	8,030	16,006	0.42

Table 6.2: Computation of the signed distance function to the contour 6.6 (a) on an adapted mesh.

Example 2

We now turn to the three-dimensional case. Some details about the mesh sizes and CPU times are to be found in table 6.3. We tested our algorithm with the same procedure as before as regards the choice of the time step, and with a parallel implementation on 10 CPU, with a shared memory architecture. Note that only the stage corresponding to the propagation of the distance throughout the domain has been parallelized, and that the initialization step - which could also be easily parallelized - actually takes most of the computation

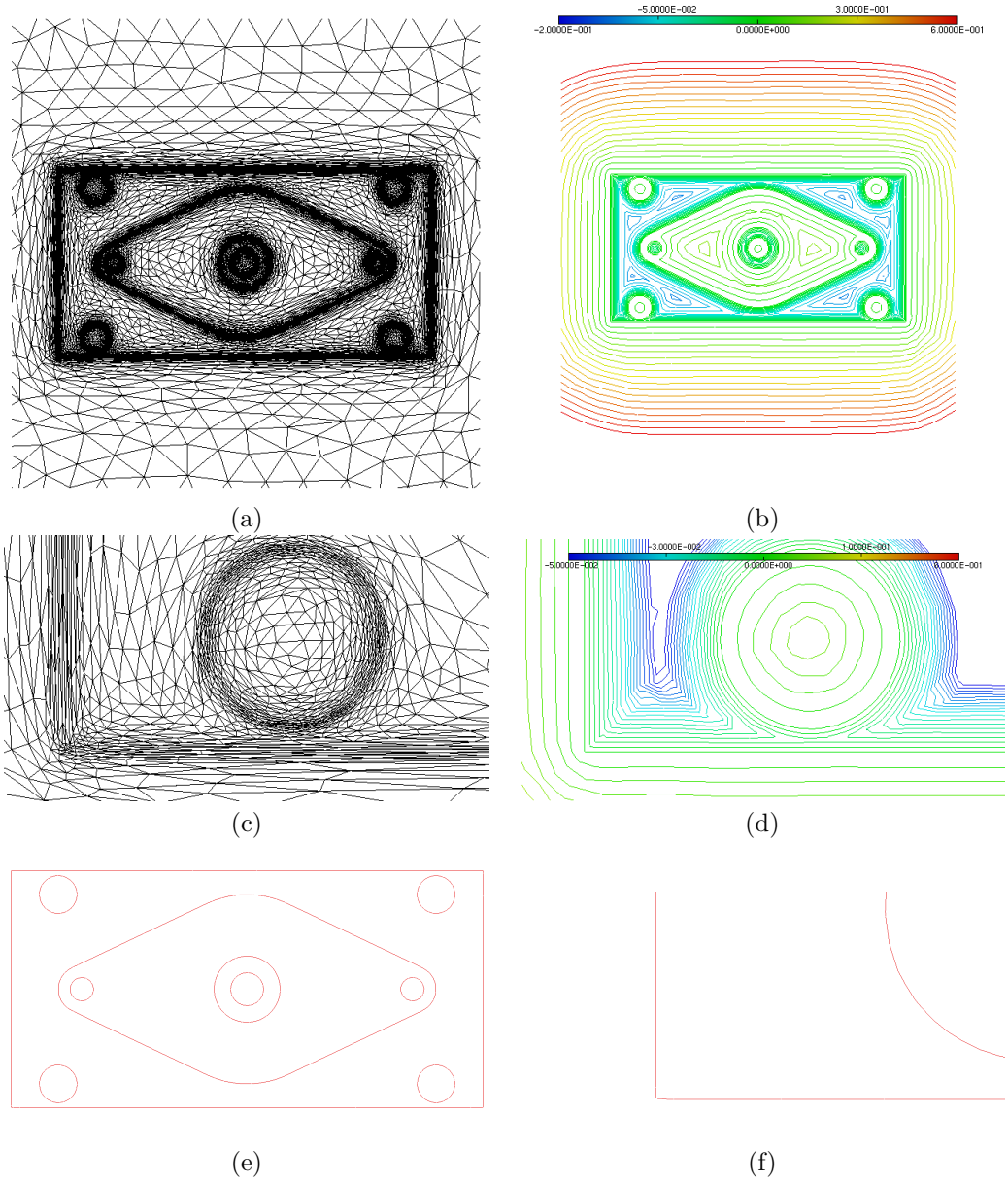


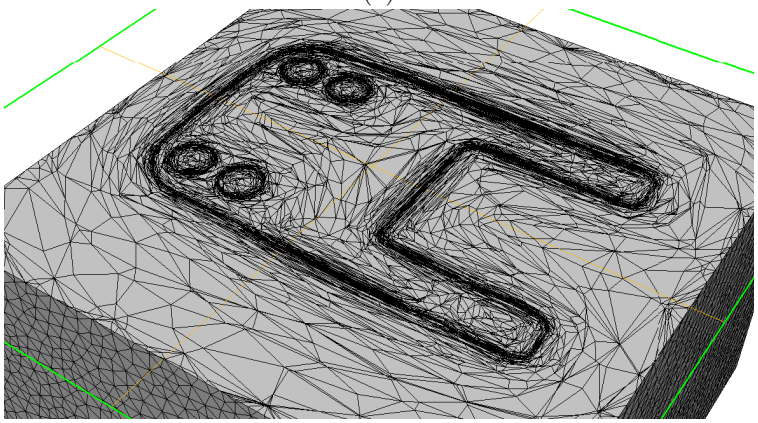
Figure 6.7: A two-dimensional example on an adapted mesh; *(a)* the associated adapted mesh, *(b)* some isolines of the signed distance function on this mesh, *(c)* a detail in the background mesh, *(d)* the corresponding detail among the isolines, *(e)* the piecewise affine reconstructed contour, and *(f)* a zoom on a corner of the reconstructed contour.

time (Table 6.3 provides the number of faces of each initial contour so as to emphasize this feature). However, we thought it better to report in table 6.3 the total computation time. First, we considered a mechanical part called *cimplex*. Figure 6.8 shows the original boundary, its reconstruction as the 0 level set of the computed approximation of the signed distance function to this contour, a level set of the computed function, and some cut in the adapted mesh. Note that the anisotropic feature of the background mesh $\mathcal{T}$ allows for a good approximation of the ridges of the contour, even though we did not apply any special process to achieve so.

Actually, it is worth mentioning that some post-treatment could be implemented so as to recover *exactly* those sharp features in the reconstruction of the interface $\partial\Omega$ [247], but we believe this goes beyond the scope of this paper.



(a)



(b)

Figure 6.8: Computation of the signed distance function to the 'cimplex'; *(a)* left -the initial domain, center - the reconstructed domain, right - isovalue 0.01 of the computed signed distance function, *(b)* a cut in the adapted mesh.

The next example, hereafter named *wheel* (Figure 6.10), emphasizes another difficulty that may arise, especially when it comes to mechanical devices, exhibiting very fine structures (or more generally, very fine details). They are very difficult to track accurately when the background mesh $\mathcal{T}$ is regular, the reason being the suitable size of a regular mesh for this purpose would be tremendous. Independantly, note that the surface mesh of the 0 level set of the computed signed distance function reconstructed by means of intersections with the background mesh $\mathcal{T}$ may be very irregular and contain two much surface elements to allow any further calculation on it. To this end, it is often necessary to proceed to a surface remeshing step in order to generate a suitable computational mesh (see [145]). An example is provided in figure 6.10.

Then, Figure 6.11 gives another example, that of the computation of the signed distance function to the Stanford *Happy Buddha*, and of the approximation of this contour as the 0 level set of the computed function.

Eventually, in several cases, for very detailed contours, we thought necessary to perform an isotropic adaptation of the background mesh, and to make an intermediate computation of the signed distance function on it before indulging in an anisotropic adaptation of the background mesh, so as to make sure to capture any close detail of the contour. See the *Aphrodite*[1] example on Figure 6.12; see also Table 6.3 for a comparison

1. Free model from `http://artist-3d.com/free_3d_models`.

between both meshes.

	Number of faces	Number of iterations	Number of nodes	Number of elements	CPU Time (seconds)
Cimplex	4,160	227	39,593	208,676	4.13
Wheel	19,972	194	367,236	2,065,767	46.641
Happy Buddha	1,085,477	160	549,818	3,218,519	1,294
Aphrodite (isotropic)	63,940	212	2,265,359	13,482,983	225
Aphrodite (anisotropic)	63,940	238	120,404	670,746	605

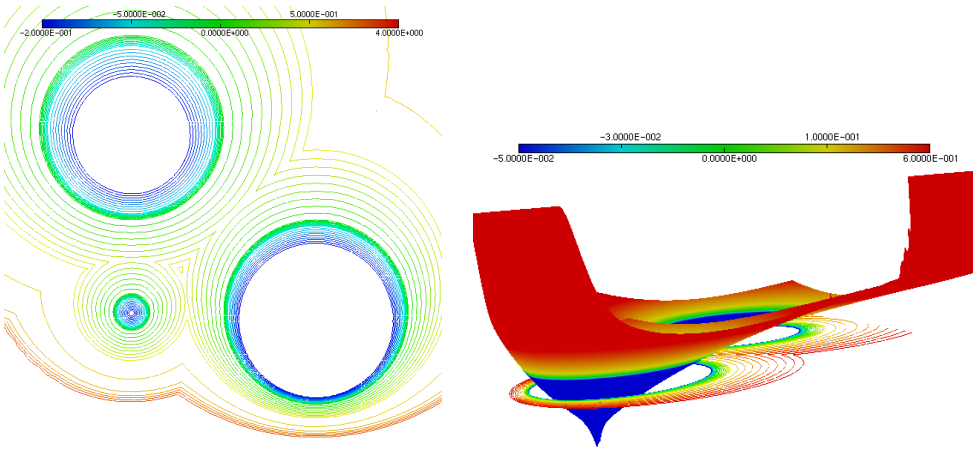
Table 6.3: Computation of the signed distance function to several subdomains of $\mathbb{R}^3$.

Example 3

So as to illustrate the idea hinted at in section 6.7, our last example concerns the computation of the signed distance function to a domain $\Omega \subset \mathbb{R}^2$, $\mathbb{R}^2$ being endowed with the hyperbolic metric of the Poincaré half-plane: let Ω be a union a 3 disks, embedded in the half-plane $\mathbb{H} := \left\{(x,y) \in \mathbb{R}^2,\ y > 0\right\}$ endowed with the so-called *Lobatchevski metric* defined by

$$\forall (x,y) \in \mathbb{H},\ \ M(x,y) := \frac{1}{y^2} I$$

where I stands for the unitary matrix of dimension 2. Figure 6.9 then show some level sets of the signed distance function to Ω with respect to metric M.



(a) (b)

Figure 6.9: *(a)* Level sets of the signed distance function to a domain composed of three disks embedded in the Poincaré half-plane. *(b)* three-dimensional graph of the function.

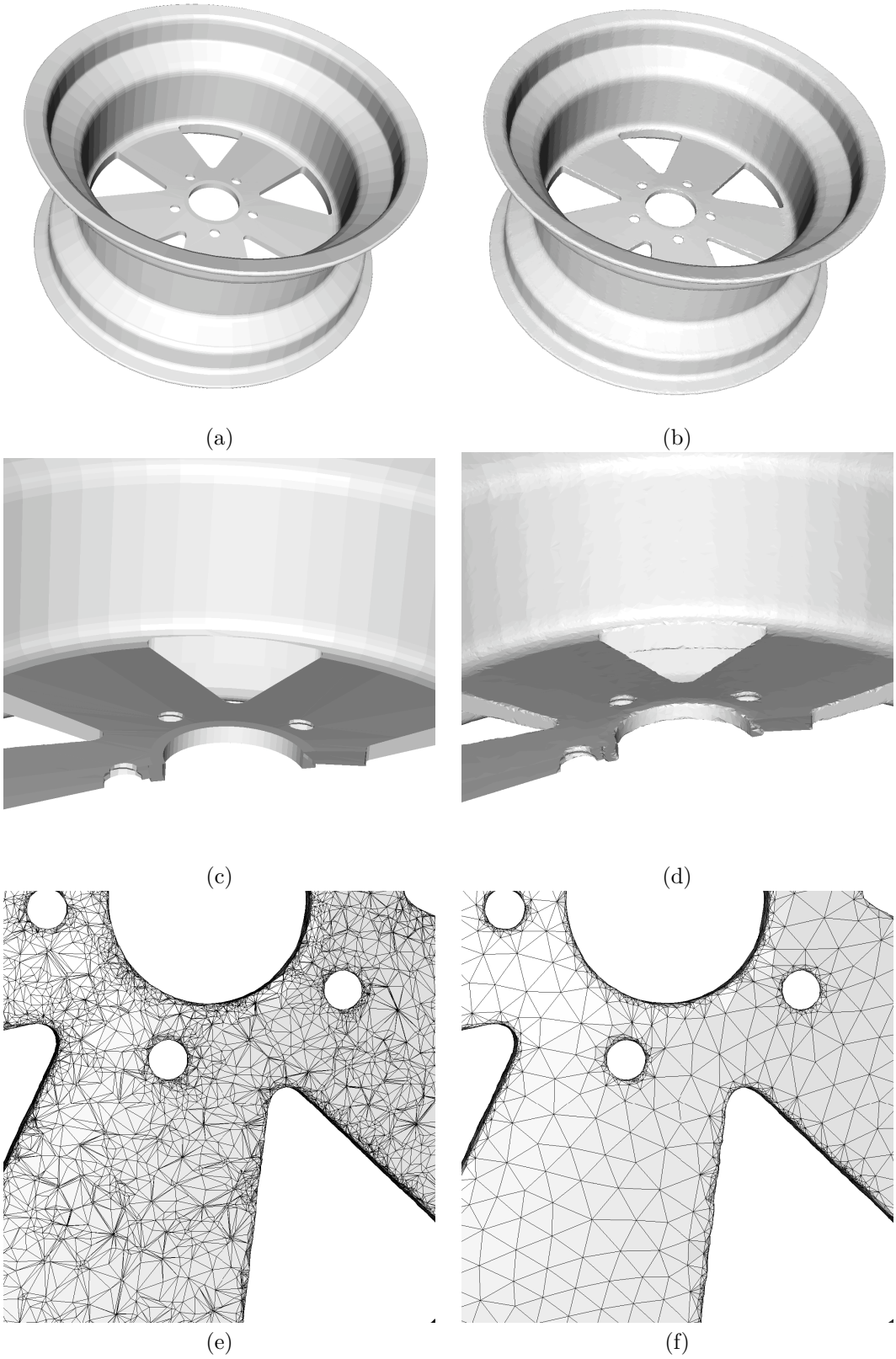


Figure 6.10: Piecewise affine reconstruction of the wheel. *(a)* The original wheel *(b)* its reconstruction *(c)* a cut in the original wheel *(d)* a cut in the reconstruction *(e)* surface triangulation before remeshing *(f)* and after remeshing.

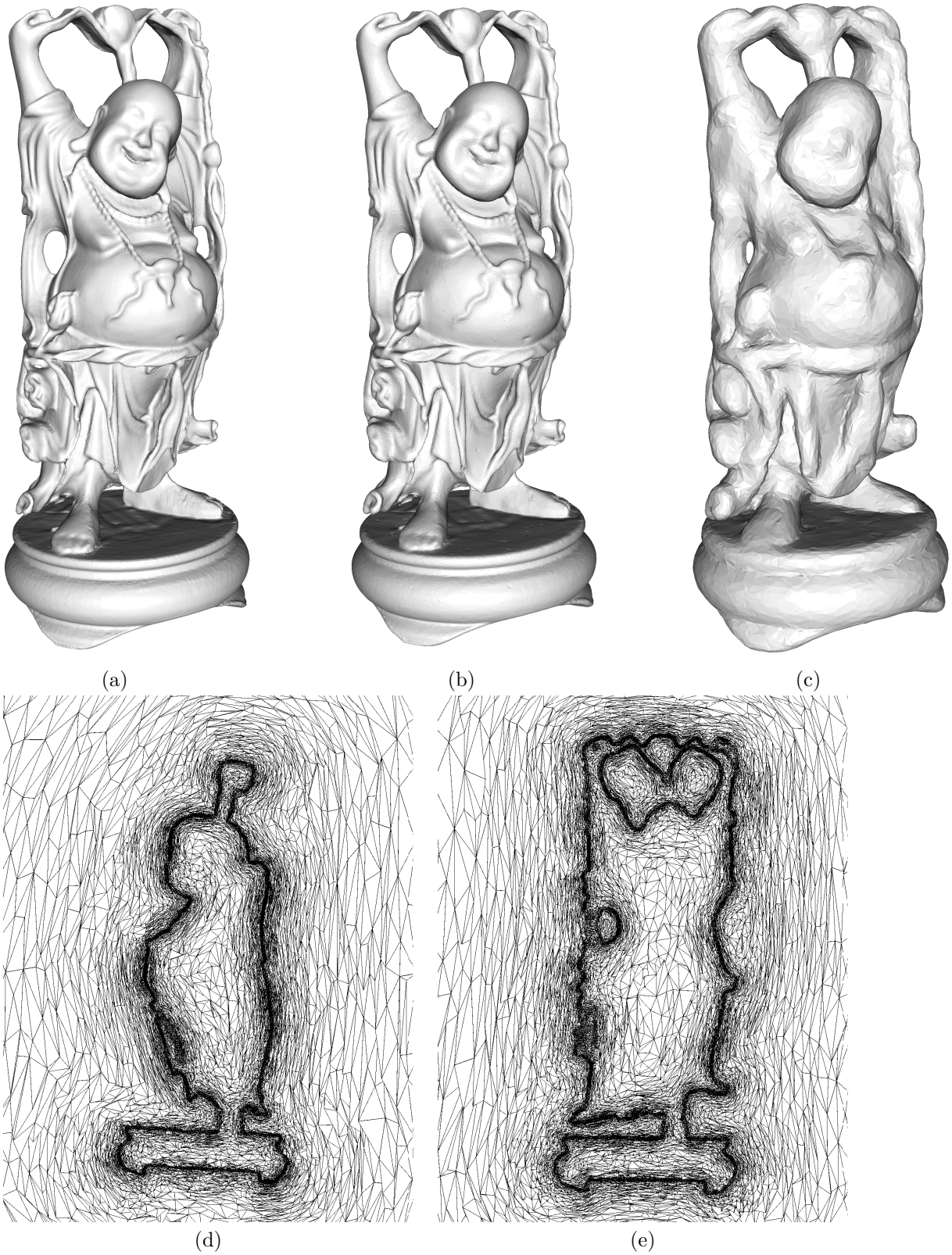


Figure 6.11: Computation of the signed distance function to the Stanford 'Happy Buddha' [102]; *(a)* The initial Buddha *(b)* isovalue 0.001 of the computed signed distance function *(c)* isovalue 0.01 *(d)* -*(e)* Two cuts in the adapted mesh.

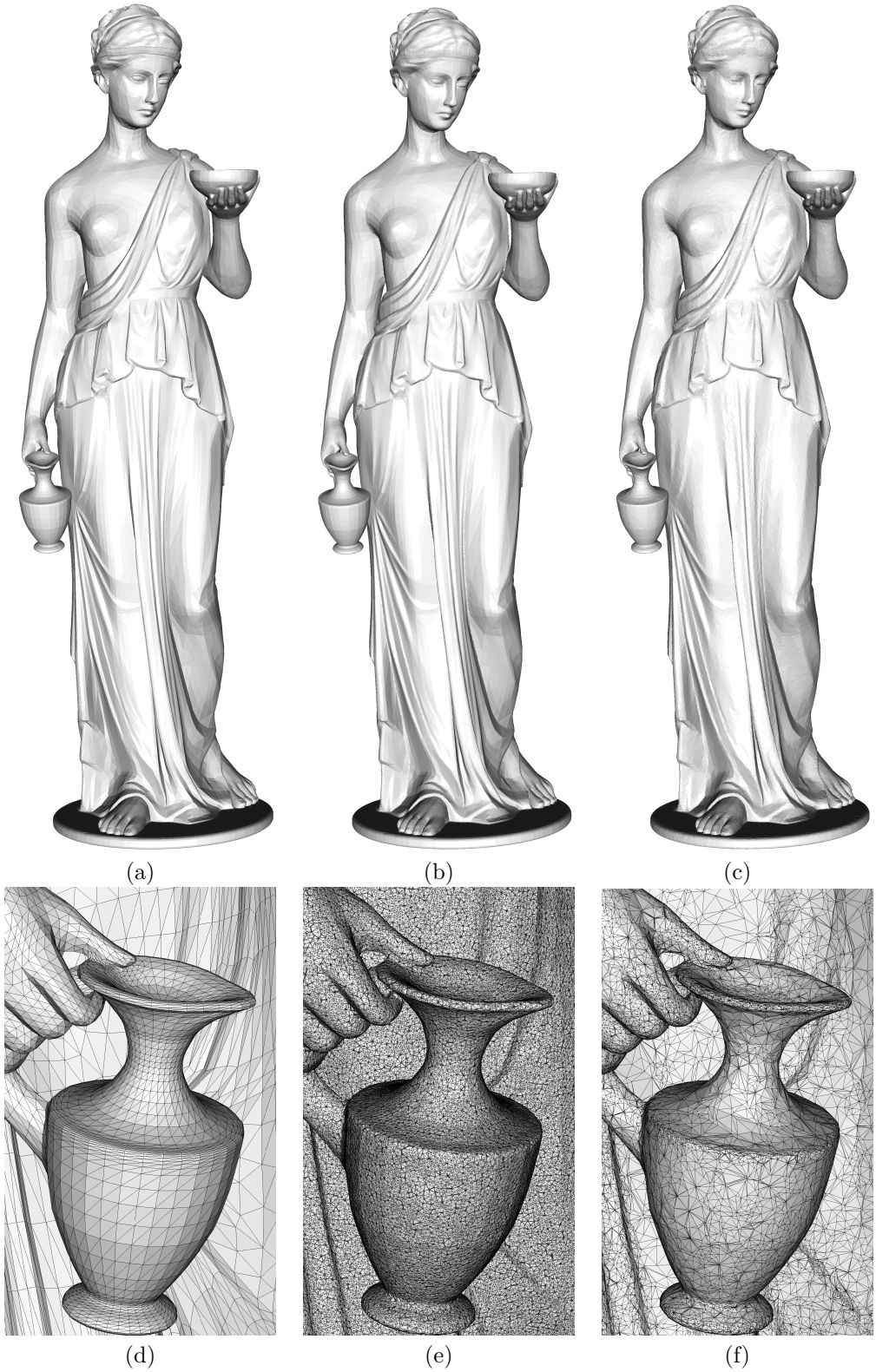


Figure 6.12: Reconstruction of the 'Aphrodite': *(a)* The original Aphrodite, *(b)* Its reconstruction with isotropic mesh adaptation (3093941 surface triangles), *(c)* its reconstruction with anisotropic mesh adaptation (300968 surface triangles), *(d)* zoom on the original Aphrodite, *(e)* zoom on its reconstruction with isotropic mesh adaptation, *(f)* zoom on its reconstruction with anisotropic mesh adaptation.

Chapter 7

An accurate anisotropic adaptation method for solving the level set advection equation

Contents

7.1 Introduction 250
7.2 Some theoretical facts around the advection equation 251
7.3 The proposed numerical method, and its error analysis 252
7.3.1 A numerical method for the advection equation based on the method of characteristics 252
7.3.2 A priori error analysis of the proposed method 253
7.3.3 A priori error estimate in terms of Hausdorff distance in the case of level-set functions 257
7.4 Mesh adaptation for the advection equation 258
7.4.1 Metric-based mesh adaptation 258
7.4.2 The proposed adaptation method 259
7.5 Additional numerical features 262
7.5.1 The need for mesh gradation control 262
7.5.2 Importance of redistancing 263
7.6 Numerical examples 264
7.6.1 Rotation of Zalesak's slotted disk 264
7.6.2 Time-reversed vortex flow 265
7.6.3 Deformation test flow 266
7.6.4 Rotation of Zalesak's sphere 268
7.6.5 Three-dimensional deformation test case 270
Appendix 274

In the present chapter, a mesh adaptation process for solving the advection equation is presented, with a particular emphasis on the case it implicitly describes an evolving surface. The proposed method mainly relies on a numerical scheme based on the method of characteristics. This low-order scheme is thoroughly analyzed on the theoretical side. An anisotropic error estimate is derived and interpreted in terms of the Hausdorff distance between the exact and approximated surfaces. The computational mesh is then adapted according to the metric supplied by this estimate. The whole process enjoys a good accuracy as far as the interface resolution is concerned. Some numerical features are discussed and several classical examples are presented and commented in two and three space dimensions.

This chapter is a joint work with Cuc Bui and Pascal Frey. Its contents have been published under the reference:

C. BUI, C. DAPOGNY AND P. FREY, *An accurate anisotropic adaptation method for solving the level set advection equation*, Int. J. Numer. Meth. Fluids, Volume 70, Issue 7, (2012), pp. 899–922.

7.1 Introduction

Since the seminal work [245], level set methods, which advocate implicit descriptions of surfaces or interfaces between different domains have become increasingly popular. Roughly speaking, the general idea consists in considering a surface in $\mathbb{R}^d$ as the zero level set of a scalar function defined on the whole ambient space. The motion of such a surface driven by a given velocity field turns out to be described by an advection equation for the associated scalar 'level set' function: see [274] or [242] for various topics around the level set method, or applications to physical problems (see also the brief summary in chapter 1). On the other hand, numerous methods are available when it comes to solving the advection equation: see [129] for a review on the topic, or [156] for numerical comparisons between several existing methods for the advection equation in the context of level set methods.

In the present chapter, we intend to solve the standard advection equation associated to a given velocity field V:

$$\frac{\partial \phi}{\partial t} + V.\nabla \phi = 0.$$

in the particular case where the transported scalar quantity $\phi(t,x)$ is a level set function associated to an evolving domain $\Omega(t)$, that is, for all $t > 0$, $x \in \mathbb{R}^d$ ($d = 2, 3$ in our examples):

$$\begin{cases} \phi(t,x) < 0 & \text{if } x \in \Omega(t) \\ \phi(t,x) = 0 & \text{if } x \in \partial\Omega(t) \\ \phi(t,x) > 0 & \text{if } x \in {}^c\overline{\Omega(t)} \end{cases} . \tag{7.1}$$

Celebrated methods are available in the setting where computations are held on a Cartesian grid (see e.g. [124, 230]). On the contrary, the whole work of this chapter unfolds in the context of fully unstructured background mesh, for we believe it is of independent relevance. For instance, it can be used for tracking an interface which evolves in a computational domain with a complex geometry (as is often the case in industrial computations) which is more easily and accurately described by an unstructured mesh. Furthermore, in several applications, the velocity driving the evolution of the interface stems from mechanical computations (flow solvers in computational fluid dynamics, shape-sensitivity analysis in shape optimization,...) which demand a mesh of the implicitly-defined domain Ω at each step of the process. It may thus be desirable that the computational mesh used for advection also encloses a discretization of this domain as a submesh, and such a feature is solely available for fully unstructured meshes. Actually, the process described in this chapter has already been used in such cases in the previous work [63], and will be a component of the mesh evolution strategy set up in chapter 9.

We aim at getting a sharp approximation of the evolving domain $\Omega(t)$ (or equivalently of the evolving boundary $\partial\Omega(t)$, hence the choice of conforming finite elements for the discretization of ϕ - from which a *conforming* piecewise affine reconstruction of the 0 isosurface of ϕ is easily produced - although discontinuous Galerkin methods generally prove very efficient in solving the advection equation (see [216] for instance). Furthermore, the surface under evolution may or may not be related to a physical problem (e.g. in fluid mechanics); for this purpose, we undertook not to address mass conservation enforcement, however crucial this aspect might prove in several applications. Of course, it is possible to couple the proposed method with a mass conservation, or mass restoration scheme.

Our main goal in this chapter is to suggest a mesh adaptation process so as to control the accuracy of the computation. To achieve this, we rely on the classical method of characteristics (see [96] for the seminal

paper around this topic in connection with the advection equation, and [256] for an exhaustive review of its use in many fluid problems) to solve the advection equation. Although it is admittedly low-order as well as very diffusive, it is simple to implement numerically, and we will see it is amenable to an error analysis which yields straightforwardly an anisotropic error estimate for the Hausdorff distance between the evolving interface and its computed approximation. Then, using anisotropic mesh adaptation on the computational mesh according to the obtained estimate brings a close approximation of the advected quantity where it is needed.

The outline of this chapter is as follows: in section 7.2, we briefly recall some basic theoretical facts around the advection equation that will be extensively used. Section 7.3 is devoted to deriving a numerical method for the considered advection equation: this method is presented in subsection 7.3.1, then analyzed in both subsection 7.3.2, where an a priori error estimate is proved, and subsection 7.3.3, where this estimate is specialized to the case of interest, that is when the advected scalar function is a level set function associated to an evolving interface. Section 7.4 tackles the issue of deducing a mesh adaptation process from the previous error estimate. After recalling some useful classical material about metric-based mesh adaptation in subsection 7.4.1, an appropriate adaptation method is detailed in subsection 7.4.2. In section 7.5, we emphasize on two crucial numerical aspects of the proposed technique: the need for a mesh gradation control in the way the computational mesh is adapted, and the role played by redistancing in our computations. Eventually, in section 7.6, several numerical examples in two and three space dimensions are developed to assess the proposed method.

7.2 Some theoretical facts around the advection equation

Given an initial function $\phi^{in} : \mathbb{R}^d \to \mathbb{R}$ and a velocity field $V : \mathbb{R}^d \to \mathbb{R}^d$ defined on the whole space, we consider the Cauchy problem for the *advection equation*: find $\phi \in \mathcal{C}^1([0,T] \times \mathbb{R}^d)$ such that

$$\left\{ \begin{array}{ll} \dfrac{\partial \phi}{\partial t}(t,x) + V(t,x).\nabla\phi(t,x) = 0 & (t,x) \in (0,T) \times \mathbb{R}^d \\ \phi(0,x) = \phi^{in}(x) & x \in \mathbb{R}^d \end{array} \right. . \tag{7.2}$$

Throughout this chapter, unless stated otherwise, we will make (at least) the following assumptions on the velocity field $V : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$:

$$\text{V is of class } \mathcal{C}^1 \text{ on } (0,T) \times \mathbb{R}^d \text{ (i.e. } V \text{ and } \nabla V \text{ are continuous over } [0,T] \times \mathbb{R}^d). \tag{7.3}$$

$$\text{There exists } \kappa > 0 \text{ such that } |V(t,x)| \leq \kappa\,(1 + |x|) \text{ for all } (t,x) \in [0,T] \times \mathbb{R}^d. \tag{7.4}$$

We then recall the following classical result ([159]) related to the advection equation.

Theorem 7.1. *Suppose (7.3) and (7.4) hold, and let $\phi^{in} \in \mathcal{C}^1\left(\mathbb{R}^d\right)$. For all $0 \leq t \leq T$ and $x \in \mathbb{R}^d$, denote $s \mapsto X(s,t,x)$ the characteristic curve emerging from point x at time t, solution to the ODE*

$$\left\{ \begin{array}{rcl} \dfrac{dX}{ds}(s,t,x) & = & V(s, X(s,t,x)) \\ X(t,t,x) & = & x \end{array} \right. , \tag{7.5}$$

then

- *For every point $x \in \mathbb{R}^d$ and every time $t \in [0,T]$, the curve $s \mapsto X(s,t,x)$ is well-defined over $[0,T]$. Furthermore, for every $s,t \in [0,T]$, the application $\mathbb{R}^d \ni x \mapsto X(s,t,x) \in \mathbb{R}^d$ is a $\mathcal{C}^1$, orientation-preserving diffeomorphism.*
- *Cauchy problem (7.2) has a unique solution $\phi \in \mathcal{C}^1([0,T] \times \mathbb{R}^d)$, given by the formula:*

$$\phi(t,x) = \phi^{in}(X(0,t,x)). \tag{7.6}$$

Note that the above result holds for the advection equation over the whole space $\mathbb{R}^d$, and for an evolution driven by a (smooth) velocity field which does not 'blow-up' at infinity. In numerical practice, we restrict ourselves to a (large) bounded computational domain, so that this property will always prove 'numerically true'. However, in this work, we will still solve the advection equation as if it were considered over $\mathbb{R}^d$, so that formula (7.6) stands, the reason being that we are mainly interested in the motion of a surface evolving 'far' from the boundary of the computational domain, so that the values of the corresponding advected level set function which are of interest do not 'see the boundary'. Actually, Cauchy problem (7.2) is ill-posed on a bounded domain, and its analysis is far more difficult: one has to introduce boundary conditions in the so-called *reentrant boundary* where the velocity field V requires information from outside the domain. See [32] for a complete study of this general problem.

In particular, Theorem 7.1 has the following interesting consequence in case the scalar quantity at stake stands for a level-set function associated to a smooth surface:

Corollary 7.1. *Under the hypothesis of theorem 7.1, suppose ϕ^{in} is a smooth level set function associated to a regular domain $\Omega^{in} \subset \mathbb{R}^d$ in the sense that (7.1) hold and such that for all $x \in \partial\Omega^{in}$, $\nabla\phi^{in}(x) \neq 0$. Then for all $t \in [0,T]$, $\Omega(t) := \left\{x \in \mathbb{R}^d,\ \phi(t,x) < 0\right\}$ is a regular domain with regular boundary $\partial\Omega(t) := \left\{x \in \mathbb{R}^d,\ \phi(t,x) = 0\right\}$, and is diffeomorphic to Ω^{in} through the mapping $X(0,t,.)$.*

7.3 The proposed numerical method, and its error analysis

7.3.1 A numerical method for the advection equation based on the method of characteristics

Suppose the whole space $\mathbb{R}^d$ is endowed with a mesh $\mathcal{T}$, and consider an associated Lagrange finite element space $V_{\mathcal{T}}$ (e.g. $\mathbb{P}^1$ or $\mathbb{P}^2$ finite elements). Given a (physical) time step Δt, a time interval $\left[T^n, T^{n+1}\right]$, $T^{n+1} = T^n + \Delta t$ (see remark below for a discussion around time-stepping), and an initial state $\phi(T^n,.)$ which fulfills the hypothesis of Theorem 7.1, we intend to approximate the function $\phi(T^{n+1},.)$, where $\phi(t,.)$ is the unique solution to the Cauchy problem:

$$\left\{\begin{array}{ll} \dfrac{\partial\phi}{\partial t}(t,x) + V(t,x).\nabla\phi(t,x) = 0, & (t,x) \in (T^n, T^{n+1}) \times \mathbb{R}^d \\ \phi(T^n,x) = \phi^{in} & \forall x \in \mathbb{R}^d \end{array}\right. ..$$

To this end, we only have an approximation $\widetilde{V}$ of vector field V, as well as a piecewise affine approximation $\phi^{T^n} \in V_{\mathcal{T}}$ of $\phi(T^n,.) = \phi^{in}$, and we also look for an approximation of $\phi(T^{n+1},.)$ as a function $\phi^{T^{n+1}} \in V_{\mathcal{T}}$, that is to say we only need to compute the desired approximation $\phi^{T^{n+1}}(x)$ for every *degree of freedom* x of the function space $V_{\mathcal{T}}$. In order to mimic the exact formula (7.6), as is well-known with the method of characteristics, we proceed within two steps:

– In a first step of *time discretization*, we compute an approximation $\widetilde{Y}(T^n, T^{n+1}, x)$ of the solution $Y(T^n, T^{n+1}, x)$ to the *approximate* characteristic curve at time T^n:

$$\left\{\begin{array}{lcl} \dfrac{dY}{ds}(s, T^{n+1}, x) & = & \widetilde{V}(s, Y(s, T^{n+1}, x)) \\ Y(T^{n+1}, T^{n+1}, x) & = & x \end{array}\right. ,$$

in which the exact velocity field V has been traded for its numerical approximation $\widetilde{V}$. To achieve this, any classical method for solving ordinary differential equations can be used (Euler's method, or a more accurate Runge-Kutta method, see [111]). For instance, introducing a subintegration time step $\delta t \ll \Delta t$ and subdividing $]T^n, T^{n+1}[= \cup_{l=0}^{L}]t^l, t^{l+1}[$, with $t^l := T^n + l\delta t$, $l = 0, ..., L$, Euler's method writes:

$$\left\{\begin{array}{c} \widetilde{Y}(T^{n+1}, T^{n+1}, x) = x \\ \widetilde{Y}(t^l, T^{n+1}, x) = \widetilde{Y}(t^{l+1}, T^{n+1}, x) - \delta t \widetilde{V}(\widetilde{Y}(t^{l+1}, T^{n+1}, x)) \quad \text{for } l = 0, ..., L-1 \end{array}\right.$$

– Then, a *spatial approximation* step is performed; in other words, if K is a simplex of $\mathcal{T}$ such that $\widetilde{Y}(T^n, T^{n+1}, x) \in K$, we have

$$\begin{aligned} \phi^{in}(X(T^n, T^{n+1}, x)) &\approx \phi^{in}\left(\widetilde{Y}(T^n, T^{n+1}, x)\right) \\ &\approx \phi^{T^n}\left(\widetilde{Y}(T^n, T^{n+1}, x)\right) \end{aligned},$$

the last expression being evaluated on basis of the discrete set of values of ϕ^n at the degrees of freedom belonging to simplex K.

Remarks 7.1.

1. *About time-stepping.* The above method uses two different time steps. The first one, $\Delta t = T^{n+1} - T^n$, is a 'large' time step, mainly related to physics: in most interesting problems, the velocity field V used for advection over $\left[T^n, T^{n+1}\right]$ stems from a mechanical computation at time T^n. Time step Δt is then the period of time for which we assume this velocity physically reliable. The second one, $\delta t \ll \Delta t$ is a sub-integration time step; it is fictitious and merely involved for the integration of the characteristic curves. It is the only one really involved by the method of characteristics. In the sequel, we will often focus on a generic period of time $[0, T]$, standing for any $\left[T^n, T^{n+1}\right]$.
2. *About numerical discretization.* The above method amounts to solving an ordinary differential equation at each degree of freedom of the computational mesh, and involves neither matrix inversion, nor quadrature formulas for approximating integrals. Consequently, it is computationally efficient in practice, and as accurate as can be a spatial first-order scheme. However, as is, it is not readily extended to more general problems, such as convection-diffusion-reaction problems. Moreover, it has been observed (see [257]) that simple first-order characteristic-based numerical scheme for treating a convection term are generally very diffusive, which could be unacceptable for several applications (such as fluid dynamics). For these reasons, characteristic-Galerkin numerical schemes are often used in combination with higher-order finite elements. See [44, 45] for an exhaustive presentation of several aspects of general characteristic-Galerkin finite element methods. In this chapter, we will overcome this drawback of being low-order by resorting to a mesh adaptation procedure.

7.3.2 A priori error analysis of the proposed method

The goal of this section is to develop an a priori error estimate for the numerical scheme presented in section 7.3.1. As already mentioned, neither the idea of using the method of characteristics for advection, nor its error analysis is new: see e.g. [256] [258] for considerations in a far more general context. However, to our knowledge, the following a priori error estimate, on which relies our adaptation scheme, is not so classical under this form (even if it is actually a variation of existing ones).

In the following, we still consider equation (7.2) for a generic period of time $[0, T]$ and over the whole space $\mathbb{R}^d$, which is endowed with two distinct simplicial meshes $\mathcal{T}$ (which carries the approximations of functions $\phi(t, x)$) and $\mathcal{T}_v$, on which we have an approximation $\widetilde{V}$ of a continuous vector field V. Such a situation frequently arises in numerical practice, where the velocity field V stems from mechanical or physical computations held on a support different from the one used to advect the level set function of interest. We denote $V_{\mathcal{T}}$ (resp. $V_{\mathcal{T}_v}$) the space of continuous functions over $\mathbb{R}^d$, whose restriction to every simplex $K \in \mathcal{T}$ (resp. $K \in \mathcal{T}_v$) is a $\mathbb{P}^1$ finite element function. For every continuous function f over $\mathbb{R}^d$, we denote $\pi_{\mathcal{T}}(f)$ (resp. $\pi_{\mathcal{T}_v}(f)$) the $\mathbb{P}^1$-interpolate of f over $\mathcal{T}$ (resp. $\mathcal{T}_v$), i.e. the unique function in $V_{\mathcal{T}}$ (resp. $V_{\mathcal{T}_v}$) which coincides with f at every node of $\mathcal{T}$ (resp. $\mathcal{T}_v$).

Theorem 7.2. *Assume moreover that V is a stationary, uniformly Lipschitz continuous vector field over $\mathbb{R}^d$, with constant k, and ϕ^{in} is a $\mathcal{C}^2$, uniformly Lipschitz continuous function over $\mathbb{R}^d$ with constant k' (so that (7.3) and (7.4) are satisfied). Assume as well the approximation $\widetilde{V}$ of V is such that both*

$\sup_{x\in\mathbb{R}^d}|V(x)-\widetilde{V}(x)|$ *and* $\sup_{K\in\mathcal{T}_v\,,\,x\in K}|\nabla V(x)-\nabla\widetilde{V}|_T|$ *are bounded.*
Let δt *be a time step, and consider the sequence of times* $t^n=n\delta t$, $0=t^0<t^1<...<t^N=T$. *Denote* $\phi^N\in V_{\mathcal{T}}$ *the sought approximation of* $\phi(T,.)$ *defined as*

$$\text{For each node } x \text{ of } \mathcal{T},\ \phi^N(x)=\pi_{\mathcal{T}}(\phi^{in})(\widetilde{Y}(0,T,x)), \tag{7.7}$$

where $\widetilde{Y}(0,T,x)$ *is the approximation of the solution* $Y(0,T,x)$ *to the ODE*

$$\left\{\begin{array}{lcl}\dfrac{dY}{ds}(s,T,x) &=& \widetilde{V}(Y(s,T,x))\\ Y(T,T,x) &=& x\end{array}\right. , \tag{7.8}$$

at time $s=0$ *by means of Euler method with time step* δt. *Then there exists a constant* C *which only depends on* V, *such that*

$$\begin{aligned}||\phi(T,.)-\phi^N||_{L^\infty(\mathbb{R}^d)}\leq\ & ||\phi(T,.)-\pi_{\mathcal{T}}\phi(T,.)||_{L^\infty(\mathbb{R}^d)}+||\phi^{in}-\pi_{\mathcal{T}}\phi^{in}||_{L^\infty(\mathbb{R}^d)}\\ &+\frac{k'}{k}(e^{kT}-1)||V-\widetilde{V}||_{L^\infty(\mathbb{R}^d)}+\frac{k'}{k}CTe^{C\delta t}\delta t\end{aligned} \tag{7.9}$$

Proof. The proof is divided into two steps.

Step 1*: time approximation of the integral curves of vector field* V. Given $x\in\mathbb{R}^d$, the aim is to estimate the approximation of $X(0,T,x)$ by $\widetilde{Y}(0,T,x)$. Here, X stands for the exact integral curve associated to vector field V, solution of (7.5), and Y is the exact integral curve associated to the approximate vector field $\widetilde{V}$ (solution of (7.8)), which is itself to be numerically approximated by $\widetilde{Y}(0,T,x)$, obtained with a first-order Euler scheme. First, the use of Lemma 7.1 below allows to quantify the gap between $Y(0,T,x)$ and $\widetilde{Y}(0,T,x)$:

$$|Y(0,T,x)-\widetilde{Y}(0,T,x)|\leq CTe^{C\delta t}\delta t$$

for a constant C which only depends on $\widetilde{V}$, or alternatively on V, owing to the assumptions made over the approximation of V by $\widetilde{V}$. Then, thanks to a variation of Gronwall's lemma recalled in appendix (see Lemma 7.3) for the approximation of $X(0,T,x)$ by $Y(0,T,x)$, we get

$$\begin{array}{rcl}|X(0,T,x)-\widetilde{Y}(0,T,x)| &\leq& |X(0,T,x)-Y(0,T,x)|+CTe^{C\delta t}\delta t\\ &=& \frac{(e^{kT}-1)}{k}||V-\pi_{\mathcal{T}_v}V||_{L^\infty(\mathbb{R}^d)}+CTe^{C\delta t}\delta t\end{array} . \tag{7.10}$$

Step 2*: spatial approximation on mesh* $\mathcal{T}$. We now turn to the error entailed by the definition of the approximation ϕ^N of function $\phi(T,.)$, for a given vertex x of mesh $\mathcal{T}$:

$$\begin{array}{rcl}|\phi(T,x)-\phi^N(x)| &=& \left|\phi^{in}\left(X(0,T,x)\right)\right)-\pi_{\mathcal{T}}\phi^{in}\left(\widetilde{Y}(0,T,x)\right)\right|\\ &\leq& \left|\phi^{in}\left(X(0,T,x)\right)\right)-\phi^{in}\left(\widetilde{Y}(0,T,x)\right)\right|\\ &&\quad+\left|\phi^{in}\left(\widetilde{Y}(0,T,x)\right)-\pi_{\mathcal{T}}\phi^{in}\left(\widetilde{Y}(0,T,x)\right)\right|\\ &\leq& k'\left|X(0,T,x)-\widetilde{Y}(0,T,x)\right|+|\phi^{in}-\pi_{\mathcal{T}}\phi^{in}||_{L^\infty(\mathbb{R}^d)}\\ &\leq& \frac{k'}{k}(e^{kT}-1)||V-\pi_{\mathcal{T}_v}V||_{L^\infty(\mathbb{R}^d)}+\frac{k'}{k}CTe^{C\delta t}\delta t+||\phi^{in}-\pi_{\mathcal{T}}\phi^{in}||_{L^\infty(\mathbb{R}^d)}.\end{array}$$

Eventually, consider *any* point $x\in\mathbb{R}^d$, and let $K\in\mathcal{T}$ any simplex containing x. Denote $a_0,...,a_d$ its vertices,

and $\lambda_0, ..., \lambda_d$ the associated barycentric coordinate functions. It stems from the definition of ϕ^N that

$$\begin{aligned} \phi(T,x) - \phi^N(x) &= \phi(T,x) - \sum_{i=0}^{d} \lambda_i(x)\phi^N(a_i) \\ &= \phi(T,x) - \pi_{\mathcal{T}}\phi(T,x) + \sum_{i=0}^{d} \lambda_i(x)\left(\phi(T,a_i) - \phi^N(a_i)\right) \end{aligned} .$$

Finally, we get the following estimate

$$\begin{aligned} ||\phi(T,.) - \phi^N||_{L^\infty(\mathbb{R}^d)} \leq ||\phi(T,.) - \pi_{\mathcal{T}}\phi(T,.)||_{L^\infty(\mathbb{R}^d)} + \frac{k'}{k}(e^{kT}-1)||V - \widetilde{V}||_{L^\infty(\mathbb{R}^d)} \\ + \frac{k'}{k} CTe^{C\delta t}\delta t + ||\phi^{in} - \pi_{\mathcal{T}}\phi^{in}||_{L^\infty(\mathbb{R}^d)}. \end{aligned}$$

□

In the first step of the above proof, we made use of the following lemma which is a version of the well-known estimate for Euler's method for an ODE, in case the velocity field is only continuous and piecewise differentiable; its proof is a mere variation of the arguments in [111] for instance.

Lemma 7.1. *Let $x \in \mathbb{R}^d$, and $V \in (\mathcal{V}_{\mathcal{T}_v})^d$ a continuous and piecewise affine vector field over $\mathbb{R}^d$, such that*

$$C_1 := \sup_{K \in \mathcal{T}_v} |\nabla V|_K| < +\infty \quad ; \quad C_2 := \sup_{x \in \mathbb{R}^d} |V(x)| < +\infty. \tag{7.11}$$

Denote by y the exact solution to the ODE:

$$\left\{ \begin{array}{rcl} y'(s) &=& V(y(s)) \\ y(T) &=& x \end{array} \right. , \tag{7.12}$$

and y^0 its approximation at time $t^0 = 0$ obtained by using a simple Euler method over the interval $[0,T]$, with time step δt. Then,

$$|y(0) - y^0| \leq 2C_1C_2\, Te^{C_1\delta t}\delta t \tag{7.13}$$

Proof. As the vector field V is continuous and uniformly Lipschitz over $\mathbb{R}^d$ because of assumption (7.11), the exact solution y to the ODE (7.12) exists over $[0,T]$, is unique, and differentiable on this interval.

Introduce the sequence $y^0, ..., y^N = x$ of approximated values for y at times $t^0, ..., t^N$ obtained by Euler's method with time step δt. For any $n = 0, ..., N-1$, this means

$$y^n = y^{n+1} - \delta t V(y^{n+1}) \tag{7.14}$$

Next, subdivide the time interval $\left[t^n, t^{n+1}\right]$ as $t^n = r^0 < r^1 < ... < r^p = t^{n+1}$ for a certain $p \in \mathbb{N}$ in such a way that for all $j = 0, ..., p-1$, $y\left(\left[r^j, r^{j+1}\right]\right)$ is included in a single simplex K^j of $\mathcal{T}'$. Then, y being differentiable on each subinterval $\left[r^j, r^{j+1}\right]$,

$$\begin{aligned} y(t^n) &= y(t^{n+1}) + \sum_{j=0}^{p-1} \left(y(r^j) - y(r^{j+1})\right) \\ &= y(t^{n+1}) + \sum_{j=0}^{p-1} \left(V(y(r^{j+1}))(r^j - r^{j+1}) + \int_0^1 (1-s) f_j''(s) ds \right) \end{aligned} , \tag{7.15}$$

where we introduced $f_j(s) = y\left(r^{j+1} + s(r^j - r^{j+1})\right)$. Substracting (7.14) to (7.15) yields

$$\begin{aligned} y(t^n) - y^n &= y(t^{n+1}) - y^{n+1} + \sum_{j=0}^{p-1} \left(\left(V(y(r^{j+1})) - V(y^{n+1})\right)(r^j - r^{j+1}) + \int_0^1 (1-s) f_j''(s) ds \right) \\ &= y(t^{n+1}) - y^{n+1} + \sum_{j=0}^{p-1} \left(\left(V(y(r^{j+1})) - V(y(t^{n+1}))\right)(r^j - r^{j+1}) + \int_0^1 (1-s) f_j''(s) ds \right) . \\ &\quad + \sum_{j=0}^{p-1} \left(V(y(t^{n+1})) - V(y^{n+1})\right)(r^j - r^{j+1}) \end{aligned}$$

Subdividing segment $\left[y(t^{n+1}), y^{n+1}\right]$ into smaller subsegments, each one being included into a single element $K \in \mathcal{T}'$, the last sum is such that

$$\left| \sum_{j=0}^{p-1} \left(V(y(t^{n+1})) - V(y^{n+1})\right)(r^j - r^{j+1}) \right| \leq C_1 \delta t |y(t^{n+1}) - y^{n+1}| \tag{7.16}$$

Furthermore, using the same argument of decomposition over smaller subsegments of each interval $\left[r^j, t^{n+1}\right]$, $j = 0, ..., p-1$, we have, thanks to the hypothesis involving constants C_1, C_2,

$$\left| \sum_{j=0}^{p-1} \left(V(y(r^{j+1})) - V(y(t^{n+1}))\right)(r^j - r^{j+1}) \right| \leq C_1 C_2 \delta t^2 \tag{7.17}$$

Eventually, from the definition of f_j, we infer $f_j''(s) = (r^j - r^{j+1})^2 \nabla V|_{T_j} . V\left(y\left(r^{j+1} + s(r^j - r^{j+1})\right)\right)$. Thus, gathering (7.16), (7.17) and the last expression, it comes

$$|y(t^n) - y^n| \leq (1 + C_1 \delta t)|y(t^{n+1}) - y^{n+1}| + 2C_1 C_2 \delta t^2.$$

From the discrete Gronwall lemma (see [111] for instance), we conclude that

$$\begin{aligned} |y(0) - y^0| &\leq e^{C_1 T}|y(T) - y^N| + 2C_1 C_2 \sum_{j=0}^{N-1} e^{C_1 \delta t} \delta t^2 \\ &\leq 2C_1 C_2 T e^{C_1 \delta t} \delta t \end{aligned} , \tag{7.18}$$

which is the expected estimate. □

Remarks 7.2.

- This estimate is low-order. As pointed out by Pironneau [256] [257], the presented method can be greatly improved by the choice of higher order methods in time, or spatial approximation. Thus, approximating the integral curves of vector field V with a 4^{th} order Runge-Kutta scheme, or the functions at stake by means of Lagrange $\mathbb{P}^2$-finite elements tremendously improves the quality of the obtained results. Unfortunately, there is no rigorous proof of these assertions (note that the previous proof cannot be generalized to those cases), however relevant they are in numerical practice.
- Many variations over this result are available: the vector field V could be analytically prescribed (as will be the case in the examples of section 7.6), and the same estimate holds, except that, understandably enough, the term $||V - \widetilde{V}||_{L^\infty(\mathbb{R}^d)}$ vanishes; it could also arise from a finite elements analysis (e.g. solution of Stokes' problem) on mesh $\mathcal{T}'$, in which case $||V - \widetilde{V}||_{L^\infty(\mathbb{R}^d)}$ should be controlled by the interpolation error $||V - \pi_{\mathcal{T}_v} V||_{L^\infty(\mathbb{R}^d)}$ (generally, it is the case in other norms, thanks to Céa's lemma).

– Theorem 7.2 holds for a generic period of time $[0,T]$, subdivided into smaller intervals of length δt. Going back to the general case of section 7.3.1, we have to apply it successively on each interval $\left[T^n, T^{n+1}\right]$; the errors in the right-hand side of (7.9) will then sum up, and it is not difficult to see that the accumulation of spatial errors will become dominant: understandably enough, the intervals on which we backtrack the characteristic curves should be as large as possible, and follow the time step Δt prescribed by the physics of the studied problems.

7.3.3 A priori error estimate in terms of Hausdorff distance in the case of level-set functions

As already pointed out, we are especially interested in the case where the transported scalar function - which we denoted $\phi(t,.)$ in the previous section - is a level set function associated to an evolving (regular) domain $\Omega(t)$. The control conveyed by Theorem 7.2 results in that case - at least formally speaking - in an estimate of the Hausdorff distance (whose definition is recalled below for convenience) between the continuous evolving interface $\partial\Omega(t)$ and its approximation as the 0 level set of the numerically computed level set function.

Definition 7.1. *Let K_1, K_2 be two compact subsets of $\mathbb{R}^d$. For any $x \in \mathbb{R}^d$, denote $d(x,K_1) = \inf_{y\in K_1} d(x,y)$ the Euclidean distance from x to K_1 and:*

$$\rho(K_1,K_2) := \sup_{x\in K_1} d(x,K_2).$$

The Hausdorff distance between K_1 and K_2 , is the quantity $d^H(K_1,K_2)$ defined by:

$$d^H(K_1,K_2) := \max(\rho(K_1,K_2),\rho(K_2,K_1))\,.$$

Using the notations of the previous section, and under the hypothesis of theorem 7.2, suppose moreover that ϕ^{in} is a level set function associated to a regular bounded domain $\Omega^{in} \subset \mathbb{R}^d$ in the sense that (7.1) holds, and that ϕ^{in} does not admit any critical point in a vicinity of $\partial\Omega^{in}$. According to the material presented in section 7.2, it follows that for all $t \in [0,T]$, $\Omega(t) := \left\{x \in \mathbb{R}^d,\ \phi(t,x) < 0\right\}$ is a bounded regular domain, with smooth boundary $\partial\Omega(t) := \left\{x \in \mathbb{R}^d,\ \phi(t,x) = 0\right\}$, and $\phi(t,.)$ does not admit any critical point within a vicinity of $\partial\Omega(t)$. We also consider, for $n = 0,...,N$, Ω^n and $\partial\Omega^n$ the piecewise affine reconstructions of $\Omega(t^n)$ and $\partial\Omega(t^n)$ obtained as

$$\Omega^n := \left\{x \in \mathbb{R}^d,\ \phi^n(x) < 0\right\} \quad ; \quad \partial\Omega^n := \left\{x \in \mathbb{R}^d,\ \phi^n(x) = 0\right\},$$

where ϕ^n is the sequence of $\mathbb{P}^1$ finite element functions produced by our numerical scheme. Recall the previous result from chapter 6:

Lemma 7.2. *Let $\phi \in \mathcal{C}^1(\mathbb{R}^d)$, which does not present any critical point within a certain tubular neighbourhood $W := \left\{x \in \mathbb{R}^d,\ |\phi(x)| < \alpha\right\}$ of $\partial\Omega$ (for some $\alpha > 0$), so that $\partial\Omega$ is a submanifold of $\mathbb{R}^d$, and Ω is a bounded subdomain of $\mathbb{R}^d$ with $\mathcal{C}^1$ boundary, and for any point $x \in W$ we have the estimate:*

$$d(x,\partial\Omega) \leq \frac{\sup_{z\in W} |\nabla\phi(z)|}{\inf_{z\in W} |\nabla\phi(z)|^2} |\phi(x)| \tag{7.19}$$

A *formal* use of this lemma yields:

$$\begin{aligned}\rho\left(\partial\Omega(T),\partial\Omega^N\right) &\leq \sup_{x\in\partial\Omega(T)} \frac{\sup_{K\in\mathcal{T}}|\nabla\phi^N|_K|}{\inf_{K\in\mathcal{T}}|\nabla\phi^N|_K|^2}|\phi^N(x)| \\ &= \sup_{x\in\partial\Omega(T)} \frac{\sup_{K\in\mathcal{T}}|\nabla\phi^N|_K|}{\inf_{K\in\mathcal{T}}|\nabla\phi^N|_K|^2}|\phi^N(x)-\phi(T,x)| \\ &\leq \frac{\sup_{K\in\mathcal{T}}|\nabla\phi^N|_K|}{\inf_{K\in\mathcal{T}}|\nabla\phi^N|_K(z)|^2}||\phi^N-\phi(T,.)||_{L^\infty(\mathbb{R}^d)}\end{aligned}$$

And now, symmetrically:

$$d^H\left(\partial\Omega(T),\partial\Omega^N\right) \leq \sup\left(\frac{\sup_{z\in\mathbb{R}^d}|\nabla\phi(T,z)|}{\inf_{z\in\mathbb{R}^d}|\nabla\phi(T,z)|^2}, \frac{\sup_{K\in\mathcal{T}}|\nabla\phi^N|_K|}{\inf_{K\in\mathcal{T}}|\nabla\phi^N|_K|^2}\right)||\phi^N-\phi(T,.)||_{L^\infty(\mathbb{R}^d)}. \tag{7.20}$$

Therefore, the estimate provided by Theorem 7.2 allows for a control over the discrepancy, measured in terms of Hausdorff distance, between the interface of interest $\partial\Omega(T)$, and its piecewise affine approximation $\partial\Omega^N$, which is the actually handled quantity.

7.4 Mesh adaptation for the advection equation

7.4.1 Metric-based mesh adaptation

As we have already been discussing in chapters 3 and 6, the main goal of mesh adaptation is to alter an initial mesh $\mathcal{T}$ in such a way its elements' size and orientation allow to perform the computation of interest with optimal efficiency -i.e. fewer elements, and an enhanced accuracy. Since [311], the idea of *metric-based* mesh adaptation has been increasingly popular: the local desired size, shape and orientation related information at a node x of mesh $\mathcal{T}$ are stored in a Riemannian metric tensor field $M(x)$, which may arise from various possible preoccupations: a posteriori geometric error estimates, analytic error estimates, etc... (see for instance [6, 25, 180]).

Given a metric tensor field $M(x)$, defined at each point $x\in\mathbb{R}^d$, (recall that in practice, $M(x)$ is defined only at the nodes of $\mathcal{T}$ and then interpolated from these values [145]) we consider respectively the *length* $\ell_M(\gamma)$ of a curve $\gamma:[0,1]\to\mathbb{R}^d$, the *volume* $V_M(K)$ of a simplex K, and the *distance* $d_M(x,y)$ between two points $x,y\in\mathbb{R}^d$ in the Riemannian space $\left(\mathbb{R}^d,M\right)$:

$$\ell_M(\gamma)=\int_0^1\sqrt{\langle M(\gamma(t))\gamma'(t),\gamma'(t)\rangle}dt\ ,\ V_M(K)=\int_K\sqrt{det(M(x))}dx,$$

$$d_M(x,y)=\inf_{\substack{\gamma\in\mathcal{C}([0,1],\mathbb{R}^d)\\ \gamma(0)=x,\gamma(1)=y}}\ell_M(\gamma).$$

We aim at modifying mesh $\mathcal{T}$ so as to make it *quasi-unit* with respect to the metric $M(x)$, that is to say all its simplices K have edges lengths lying in $\left[\frac{1}{\sqrt{2}},\sqrt{2}\right]$. What is more, we expect the *anisotropic quality measure*:

$$\mathcal{Q}_M(K):=\alpha_d\frac{V_M(K)^2}{\left(\sum_{i=1}^{na}\ell_M(e_i)^2\right)^d}$$

of all elements of the mesh (where $na = d(d+1)/2$ is the number of edges of a d-dimensional simplex, e_i are the edges of K and α_d is a normalization factor) to be as close to 1 as possible. The underlying geometrical interpretation is that for any given node x_0 of a quasi-unit mesh $\mathcal{T}$ with respect to $M(x)$, every element $K \in \mathcal{T}$ which shares x_0 as a vertex fits 'at best' in the unit pseudo-ellipsoid

$$\Phi_M(x_0) = \left\{ x \in \mathbb{R}^d,\ d_M(x, x_0) = 1 \right\},$$

which reduces to a true ellipsoid when M is constant over $\mathbb{R}^d$. In this latter case, the eigenvectors $e_1, ..., e_d$ of M give the directions of the principal axis of this ellipsoid, while the associated eigenvalues $\lambda_1, ..., \lambda_d$ are linked to the principal radii (or characteristic lengths) $h_1, ..., h_d$ in the respective directions $e_1, ..., e_d$ by: $h_i = \frac{1}{\sqrt{\lambda_i}}$, $i = 1, ..., d$.

In several applications (see the next sections), it turns out to be desirable to adapt a mesh *at the same time* to several a priori independent information, supplied by two (or more) metric tensor fields M_1, M_2. This is classically achieved resorting to a so-called *metric intersection* procedure: operating on the simultaneous reductions of $M_1(x)$ and $M_2(x)$ at any point $x \in \mathbb{R}^d$,

$$M_1(x) = {}^tP(x) \begin{pmatrix} \lambda_1(x) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_d(x) \end{pmatrix} P(x), M_2 = {}^tP(x) \begin{pmatrix} \mu_1(x) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mu_d(x) \end{pmatrix} P(x),$$

where P is an invertible matrix, $\lambda_1, ..., \lambda_d, \mu_1, ..., \mu_d > 0$, the *intersected metric*

$$M_1 \cap M_2(x) := {}^tP(x) \begin{pmatrix} \sup(\lambda_1(x), \mu_1(x)) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sup(\lambda_d(x), \mu_d(x)) \end{pmatrix} P(x)$$

carries both information in the sense that its unit pseudo-ellipsoid $\Phi_{M_1 \cap M_2}(x)$ is a maximal pseudo-ellipsoid enclosed in both $\Phi_{M_1}(x)$ and $\Phi_{M_2}(x)$, at least if M_1 and M_2 are constant over $\mathbb{R}^d$ (see [145] for details).

Let us eventually recall that several techniques have been thought up for generating anisotropic meshes according to a metric tensor field, which can be roughly classified into two categories. On the one hand, global methods, such as Delaunay-based methods and advancing-front methods (see chapter 3), perform the same kind of operations as in the classical case with adapted notions of length and volume. On the other hand, local mesh modification methods [115] - to which belongs the approach followed in this chapter - start from an existing non-adapted mesh and adapt it so that it fulfills at best the above conditions.

7.4.2 The proposed adaptation method

In this section, we go on in the framework of sections 7.3.2 and 7.3.3 and rely on the previous error estimates (7.9) and (7.20) to infer a mesh adaptation method which allows to obtain a good discrete approximation $\partial\Omega_T$ of $\partial\Omega(T)$. For the sake of clarity we shall now denote ϕ_T the approximation of $\phi(T, .)$ obtained by the algorithm of section 7.3.1 and

$$\Omega_T = \left\{ x \in \mathbb{R}^d,\ \phi_T(x) < 0 \right\} \quad , \quad \partial\Omega_T = \left\{ x \in \mathbb{R}^d, \phi_T(x) = 0 \right\}$$

the associated polyhedral domain and surface. From formula (7.9), the control we get over $||\phi(T, .) - \phi_T||_{L^\infty(\mathbb{R}^d)}$ -or indifferently $d^H(\partial\Omega(T), \partial\Omega_T)$ - consists of three independent contributions:

1. The first one
$$\left(||\phi(T, .) - \pi_{\mathcal{T}}\phi(T, .)||_{L^\infty(\mathbb{R}^d)} + ||\phi^{in} - \pi_{\mathcal{T}}\phi^{in}||_{L^\infty(\mathbb{R}^d)} \right)$$
is only related to the way mesh $\mathcal{T}$ is adapted to the interpolation of functions $\phi(T, .)$ and ϕ^{in}. The proposed adaptation method focuses on dampening this part of the error.

2. The second one,
$$\frac{k'}{k} C T e^{C\delta t} \delta t$$
is solely related to the time discretization of interval $[0, T]$ with the substep δt, and for this contribution to be decreased, substep δt has to be decreased.

3. The last part,
$$\frac{k'}{k}(e^{kT} - 1)||V - \widetilde{V}||_{L^\infty(\mathbb{R}^d)}$$
is connected to the quality of the approximation of velocity field V. If this vector field is to be obtained by means of a finite element computation on a mesh $\mathcal{T}_v$, this has to do with the quality of $\mathcal{T}_v$ as regards this computation, and we do not intend to discuss these aspects. As said previously, in all our numerical examples, we only considered analytically prescribed velocity fields, and this contribution actually does not appear.

This leaves us with the objective of adapting mesh $\mathcal{T}$ in such a way *both* interpolation errors $||\phi(T,.) - \pi_{\mathcal{T}}\phi(T,.)||_{L^\infty(\mathbb{R}^d)}$ and $||\phi^{in} - \pi_{\mathcal{T}}\phi^{in}||_{L^\infty(\mathbb{R}^d)}$ are made small. In other words, in view of the estimates in section 7.3.3, this means that mesh $\mathcal{T}$ allows for a suitable approximation of the 0 level sets of both ϕ^{in} and $\phi(T,.)$ by the 0 level set of their respective $\mathbb{P}^1$-interpolate $\pi_{\mathcal{T}}\phi^{in}$, $\pi_{\mathcal{T}}\phi(T,.)$. This is actually very intuitive: neglecting the errors on the knowledge of the velocity field, the accuracy of the process is controlled by how well mesh $\mathcal{T}$ allows for a good knowledge of the initial surface, and how well it fits a description of the final one.

To achieve such a mesh, we recall the classical L^∞ error estimate for the Lagrange finite element $\mathbb{P}^1$-interpolation error of a function u of class $\mathcal{C}^2$ [103]:

Theorem 7.3. *Let $\mathcal{T}$ a simplicial mesh of $\mathbb{R}^d$ (or a polyhedral subset of it) and ϕ a $\mathcal{C}^2$ function on $\mathbb{R}^d$. Then for every simplex $K \in \mathcal{T}$,*
$$||\phi - \pi_{\mathcal{T}}\phi||_{L^\infty(K)} \leq \frac{1}{2}\left(\frac{d}{d+1}\right)^2 \max_{x\in K}\max_{y,z\in K} \langle |\mathcal{H}(\phi)|(x)yz, yz\rangle$$
where $\mathcal{H}(\phi)$ is the Hessian matrix of u and, for a symmetric matrix $S \in \mathcal{S}_d(\mathbb{R})$ which admits the following diagonal shape in orthonormal basis $S = P\,\mathrm{diag}(\{\lambda_i\}_{1\leq i\leq d})\,P^T$, we denote $|S| := P\,\mathrm{diag}(\{|\lambda_i|\}_{1\leq i\leq d})\,P^T$.

According to [6], this theorem expresses the idea that a mesh $\mathcal{T}$ suitable for the $\mathbb{P}^1$-Lagrange interpolation of a smooth function ϕ -i.e. such that $||\phi - \pi_{\mathcal{T}}\phi||_{L^\infty(K)} \leq \varepsilon$ for a given tolerance $\varepsilon > 0$ and every simplex $K \in \mathcal{T}$ - can be roughly obtained as a quasi-unit mesh for the metric M_ϕ defined at each vertex x of $\mathcal{T}$ by:
$$M_\phi(x) = P(x)\begin{pmatrix} \widetilde{\lambda_1} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \widetilde{\lambda_d} \end{pmatrix} P(x)^T \tag{7.21}$$
where
$$\widetilde{\lambda_i} = \min\left(\max\left(\frac{c|\lambda_i|}{\varepsilon}, \frac{1}{{h_{max}}^2}\right), \frac{1}{{h_{min}}^2}\right), \quad \widetilde{|\mathcal{H}(\phi)|}(x) = P(x)\begin{pmatrix} |\lambda_1| & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & |\lambda_d| \end{pmatrix} P(x)^T$$
being an approximation of the Hessian of ϕ around node x, written here in diagonal form in an orthonormal basis, h_{min} (resp. h_{max}) being the smallest (resp. largest) size allowed for an element in any direction, and c being the above constant.

Remark 7.3. Actually, we do not need to adapt the whole mesh $\mathcal{T}$ with respect to metric M_ϕ: as we are mostly interested in an accurate approximation of the zero level sets of the considered functions by the piecewise affine zero level sets of their $\mathbb{P}^1$-Lagrange interpolates, we only need to reach a quasi-unit mesh $\mathcal{T}$ according to M_ϕ in a vicinity of the 0 level set of ϕ. See section 7.5.1 for a further discussion on this point.

In our applications, we are interested in modifying mesh $\mathcal{T}$ so that it becomes adapted to *both* Lagrange $\mathbb{P}^1$ interpolation of functions ϕ^{in} and $\phi(T,.)$. Such a mesh is built as a quasi-unit mesh according to the intersection $M_{\phi^{in}} \cap M_{\phi(T,.)}$ of metrics $M_{\phi^{in}}$ and $M_{\phi(T,.)}$, respectively adapted to ϕ^{in} and $\phi(T,.)$ (see figure 7.1 for an example).

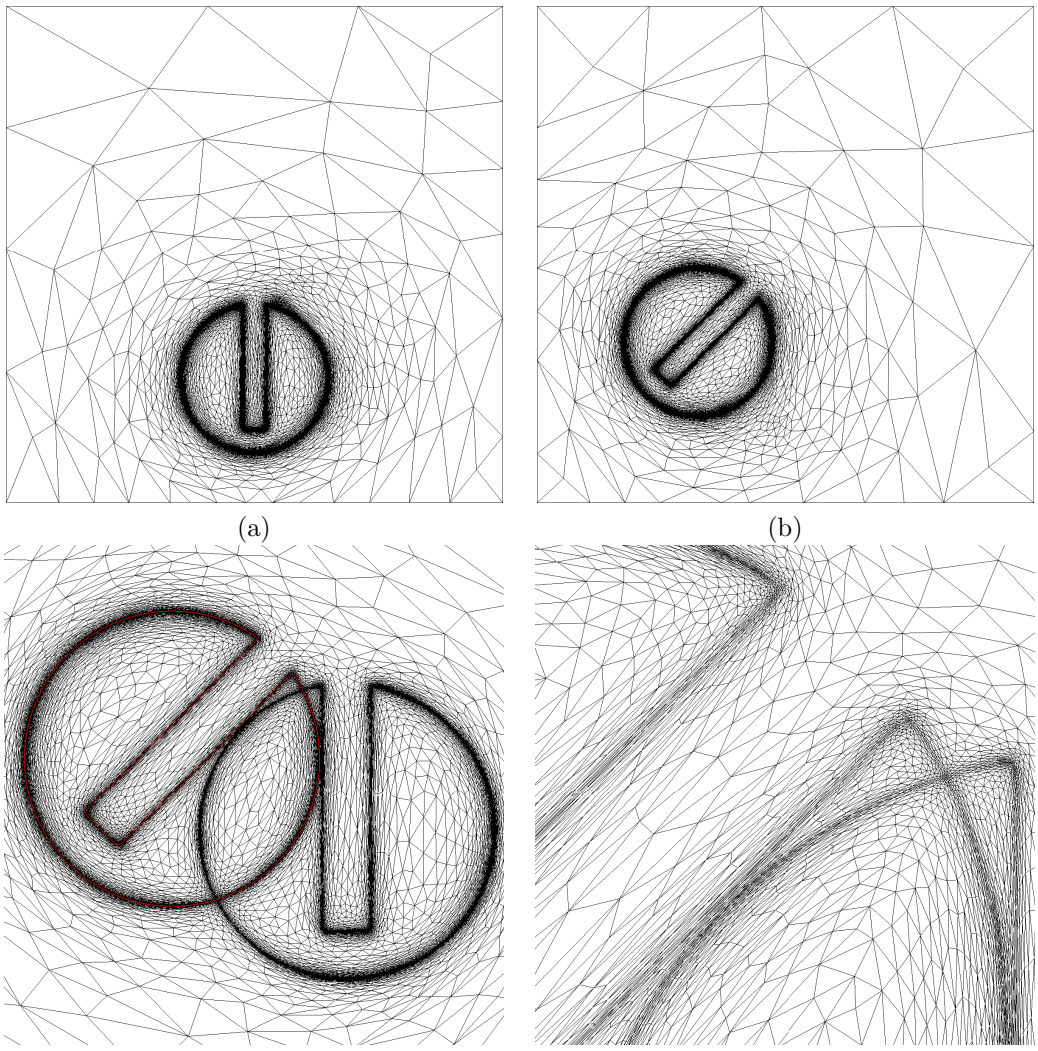


Figure 7.1: Rotation of Zalesak's slotted disk of angle $\frac{\pi}{4}$: *(a)* adapted mesh at time $t = \pi$, *(b)* adapted mesh at time $\frac{5\pi}{8}$, *(c)* mesh adapted to both interfaces $\partial\Omega^{in}$ and $\partial\Omega(T)$ (displayed in red), *(d)* zoom on the mesh.

In practice, our adaptation procedure is *iterative*: it starts with an initial function ϕ^{in}, on an adapted mesh $\mathcal{T}^{in}$, adapted to metric $M_{\phi^{in}}$, and in order to compute an approximation ϕ_T of $\phi(T,.)$, the advection

equation is solved up to m times over the period $[0,T]$: the first $m-1$ times are 'virtual', and aimed at getting an increasingly accurate approximation $\widetilde{\phi_T^k}$ $(k=0,...m-1)$ of $\phi(T,.)$, as well as an increasingly well-adapted mesh $\mathcal{T}^k$ to the intersected metric $M_{\phi^{in}} \cap M_{\phi(T,.)}$. Eventually, the m-th resolution of the advection equation is carried out on a well-adapted mesh $\mathcal{T}^{m-1}$ and yields a close approximation $\phi_T = \widetilde{\phi_T^m}$ of $\phi(T,.)$ on a well-adapted mesh $\mathcal{T}^m$ to $M_{\phi^{in}} \cap M_{\phi(T,.)}$. Such an iterative procedure is absolutely crucial, because the time period $[0,T]$ can be very large. Thus, the sought $\partial\Omega(T)$ is likely to be located very far from $\partial\Omega^{in}$, i.e. in an area where the initial mesh is very coarse.

All things considered, the proposed adaptation method for advection equation is summed up in algorithm (3), stille written for a generic time interval $[0,T]$. Of course, in the general case, when several such time periods $\left[T^n, T^{n+1}\right]$ follow one another, this process has to be applied successively to each one of them.

Algorithm 3 Adaptation method for advection equation over $[0,T]$.

Start with an approximation ϕ_0 (e.g. $\mathbb{P}^1$-interpolate) of function ϕ^{in} on mesh $\mathcal{T}$.
for $k = 0, ..., m-1$ **do**
  if $k = 0$ **then**
    Set $\widetilde{\mathcal{T}^0} = \mathcal{T}$ and $\widetilde{\phi^{in,0}} = \phi_0$.
  end if
  Solve the advection equation with velocity field V and initial state $\widetilde{\phi^{in,k}}$, over $[0,T]$, on $\widetilde{\mathcal{T}^k}$. $\widetilde{\phi_T^k}$
  Compute the intersected metric $M^k := M_{\widetilde{\phi_0^k}} \cap M_{\widetilde{\phi_T^k}}$
  Adapt $\widetilde{\mathcal{T}^k}$ with respect to M^k. $\widetilde{\mathcal{T}^{k+1}}$
  if $k+1 < m$ **then**
    Project ϕ_0 on mesh $\widetilde{\mathcal{T}^{k+1}}$. $\widetilde{\phi^{in,k+1}}$
  else
    Project $\widetilde{\phi_T^k}$ on $\widetilde{\mathcal{T}^{k+1}}$. $(\widetilde{\mathcal{T}^m}, \widetilde{\phi_T^m})$
    Adapt $\widetilde{\mathcal{T}^m}$ with respect to $\widetilde{\phi_T^m}$, and project this function on the final mesh. $(\mathcal{T}_T, \phi_T)$
  end if
end for
return $(\mathcal{T}_T, \phi_T)$

7.5 Additional numerical features

7.5.1 The need for mesh gradation control

Considering a function ϕ, we are mainly interested in its 0 level set. Thus, we only need to adapt our computational mesh $\mathcal{T}$ with respect to the metric M_ϕ defined in the previous section on a neighborhood of the interface of interest $\partial\Omega := \left\{x \in \mathbb{R}^d,\ \phi(x) = 0\right\}$. As is classical in mesh adaptation, we could be tempted to ask $\mathcal{T}$ to show very large, isotropic elements 'far' from this interface, in such a way $\mathcal{T}$ consists of very few elements, with optimal size and orientation. However, doing so, variations in the size and orientation prescriptions are bound to be very sharp, resulting in a shock of features which can yield severe instabilities during the mesh adaptation process (see figure 7.5 for an example). To get past this difficulty, we impose a control over mesh gradation near the interface [53]. One possible way to do so consists in, roughly speaking, bounding the allowed ratio between lengths $\ell_M(e_1)$ and $\ell_M(e_2)$ of any two edges e_1, e_2 belonging to a common simplex K by a constant value r (in the examples of section 7.6, we used $r = 4$), i.e.

$$\frac{1}{r} \leq \frac{\ell_M(e_1)}{\ell_M(e_2)} \leq r.$$

7.5.2 Importance of redistancing

In the context of level set methods, it has been observed that too steep or too loose variations in the level sets of the function $\phi(t,.)$ under evolution may jeopardize the accuracy of the computation. To overcome this feature, since [89], a great attention has been paid to maintaining or restoring $\phi(t,.)$ as the signed distance function to its 0 level set $\partial\Omega(t)$ at least near $\partial\Omega(t)$ (so that $|\nabla\phi(t,.)| = 1$ in a vicinity of $\partial\Omega(t)$), even though, doing so, the handled interface may inevitably end up perturbed: see for instance [27, 298] for mass-preserving approaches, or [37] for a smoothing redistancing process. Here we apply the previous study of chapter 6, merely replacing the computed approximation ϕ_T of $\phi(T,.)$ with the signed distance function d_T to Ω_T. Given a small time step dt, d_T is computed as the steady state of the sequence of $\mathbb{P}^1$-finite element functions d^n defined in algorithm 4, which is based on the properties of the *unsteady Eikonal equation*.

Algorithm 4 Redistancing of the level set function

Initialize function d^0 with:

$$\begin{cases} d^0(x) = \text{approximation of the signed distance function to } \Omega_T \text{ if } x \\ \qquad\qquad \text{belongs to a simplex of } \mathcal{T} \text{ intersecting } \partial\Omega_T \\ d^0(x) = \pm d_{MAX} \text{ otherwise} \end{cases}$$

for $n = 1, \ldots$ until convergence **do**
  $d^n(x) = d^{n-1}(x)$ for each node x of $\mathcal{T}$
  for each node x of $\mathcal{T}$ which does not belong to a simplex intersecting $\partial\Omega_T$ **do**
    if $x \notin \Omega$ **then**

$$d^n(x) = \min\left(d^{n-1}(x), \min_{K\in\mathcal{T}\ s.t.\ x \text{ is a node of } K} d^{n-1}\left(x - \frac{\nabla(d^{n-1}|_K)}{|\nabla(d^{n-1}|_K)|}dt\right) + dt\right),$$

    else

$$d^n(x) = \max\left(d^{n-1}(x), \max_{K\in\mathcal{T}\ s.t.\ x \text{ is a node of } K} d^{n-1}\left(x + \frac{\nabla(d^{n-1}|_K)}{|\nabla(d^{n-1}|_K)|}dt\right) - dt\right).$$

    end if
  end for
end for
return d^n

As discussed in chapter 6, the time step dt must be chosen small enough so that the update of $d^n(x)$ with the formulae of algorithm 4 does not require values of d^{n-1} lying 'too far' on the other side of the boundary. For this reason, dt should be taken of the order of the local mesh size near the interface $\partial\Omega_T$. However, this time step can be steadily increased, as values of the sequence d^n converge near the interface. What is more, we actually need d_T to enjoy the distance property only in a neighborhood of $\partial\Omega_T$. For this reason, in practice, we limit ourselves to performing a fixed number of the iterations of this algorithm.

The study carried out in chapter 6 showed that doing so yields formally an error in terms of Hausdorff distance between $\partial\Omega_T$ and the new interface $\widetilde{\partial\Omega_T}$:

$$d^H\left(\partial\Omega_T, \widetilde{\partial\Omega_T}\right) \leq \sup\left(\frac{\sup_{K\in\mathcal{T}}|\nabla d_T|_K|}{\inf_{K\in\mathcal{T}}|\nabla d_T|_K|^2}, \frac{\sup_{K\in\mathcal{T}}|\nabla\widetilde{d_T}|_K|}{\inf_{K\in\mathcal{T}}|\nabla\widetilde{d_T}|_K|^2}\right) \max_{K\in\mathcal{T}}\max_{y,z\in K}\langle|\mathcal{H}(\widetilde{d_T})|yz, yz\rangle. \tag{7.22}$$

Consequently, given mesh $\mathcal{T}$ is adapted to $|\mathcal{H}(d_T)| \approx |\mathcal{H}(\widetilde{d_T})|$ in the sense of section 7.4.2, this error is actually very small, controlled by the fixed precision parameter ε for the quality of the approximation of $\partial\Omega(T)$ by means of $\partial\Omega_T$.

Note that in the proposed context, the redistancing process is twice as important as in the case of level set methods performed on a fixed mesh Indeed, the narrow band on which we impose our mesh to be adapted with respect to a metric M_ϕ as in section 7.4.2 is numerically identified according to the values of ϕ. Hence, very stretched level sets of ϕ would also cause mesh adaptation to be performed on an ill-identified narrow band around $\partial\Omega$.

7.6 Numerical examples

In this section, we present several numerical examples in two or three dimensions in order to assess the validity of the proposed adaptation method for the level set advection equation.

As presented in the analysis of section 7.3.2, all functions are approximated by $\mathbb{P}^1$ Lagrange finite element functions, and we discretized the ODE (7.5) with a 4-th order Runge-Kutta method. As hinted at previously, the use of higher order spatial approximation (e.g. resorting to $\mathbb{P}^k$ Lagrange finite element functions) would improve the method. It is also worth mentionning that we used the aforementioned redistancing procedure at each step, even for the cases (e.g. rigid-body motions) that do not theoretically bring about any distortion of the level sets of the evolving function.

Each one of the presented examples consists in letting evolve an initial interface $\partial\Omega^{in}$ submitted to a more or less deforming velocity field V from time 0 to a final time T, cooked up in such a way the final (theoretical) surface $\partial\Omega(T)$ coincides with $\partial\Omega^{in}$. Hence, analyzing the gap between the initial interface $\partial\Omega^{in}$ and the one $\partial\Omega_T$ resulting from the process allows to gauge the quality of our numerical method. The accuracy of the computation is evaluated in terms of the Hausdorff distance between the numerical initial interface associated to $\partial\Omega^{in}$ and the computed final interface $\partial\Omega_T$, which is computed with a brute-force approach (in $2d$ only). We believe this is the relevant way to measure the error entailed by the method, inasmuch as it is the quantity we mean to control with our mesh adaptation procedure, through estimate (7.20). We are however well aware this is not so classical an error measure, and propose also more standard error measures. Although no particular attention has been paid to mass conservation during this work, we also display the loss of mass between initial and final step for the sake of completeness.

All our $2d$ examples were run on a MacBook Pro, 2.66 Ghz, (4 Go), and our $3d$ examples on an OPTERON 2.1 Ghz.

7.6.1 Rotation of Zalesak's slotted disk

As initially proposed in [331], consider a unit square as a computational domain, in which lies a disk of radius 0.15 centered at $(0.5, 0.75)$, with a slot of length 0.25, submitted to a uniform rotation of center $(0.5, 0.5)$, corresponding to an evolution under the velocity field:

$$V(t,x,y) = \begin{pmatrix} -(y-0.5) \\ (x-0.5) \end{pmatrix}.$$

between $0 \leq t \leq T = 2\pi$, so that the final interface should theoretically overlay the initial one. This test-case classically allows for an assessment of the well-preservation of the sharp features of an interface through an evolution process. We subdivide the time interval into 8 time periods on which algorithm 3 is successively applied. We perform two computations with different parameters as regards the mesh size prescription so as to test the scaling of the method. Figure 7.2 displays a comparison between both interfaces. In table 7.1, we provide the associated parameters (precision parameter ε, minimum size parameter h_{min}), along with their translation in terms of mesh size: maximum number np of points of a mesh adapted to a single interface (the number of points of a mesh adapted to both interfaces being approximately equal to twice this number), as well as the error estimates of interest: we compute the Hausdorff distance between the initial interface,

as well as two error measures that are more classical in the literature [2, 73, 192, 230]: the measure of the symmetric difference between the initial and final domains Ω^{in} and Ω_T, (or sometimes referred to as L^1-*error measure*)

$$E_{sd}(\Omega^{in}, \Omega_T) := |\,(\Omega_{in} \cup \Omega_T) \setminus (\Omega_{in} \cap \Omega_T)\,|$$

and the L^∞-*error measure* between both numerically obtained corresponding level-set functions u^{in} and u_T:

$$E_\infty(u^{in}, u_T) := \sup_{K \in \mathcal{K}} ||u^{in} - u_T||_{L^\infty(K)},$$

$\mathcal{K} \subset \mathcal{T}$ being the set of simplices crossed by the 0-level set of either u^{in} or u_T. The whole computation of the first test takes roughly 5 minutes, while it takes about 8 minutes for the second.

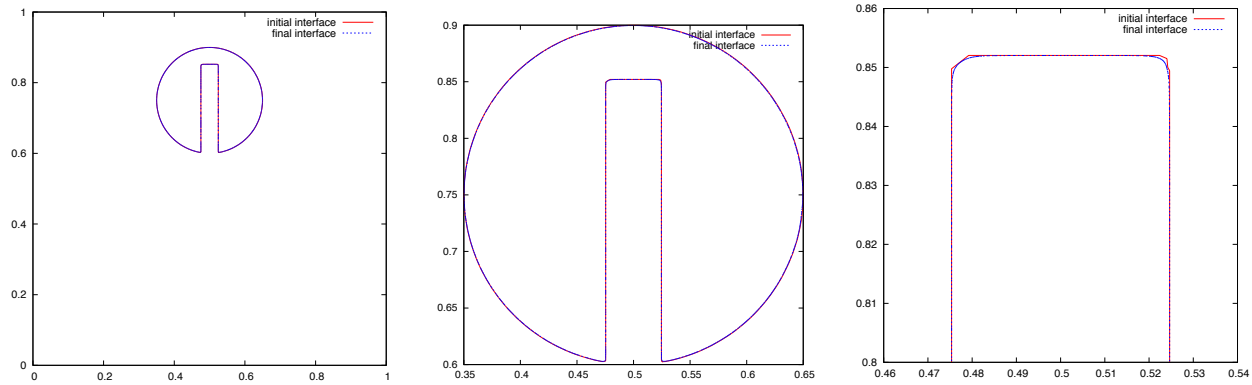


Figure 7.2: *(left)* Superposition of the slotted disk after one complete revolution (blue line) over the initial one (red line), and zoom on the surfaces *(middle and right)*.

	ε	h_{min}	np	Volume loss (% initial vol.)	$d^H(\partial\Omega^{in}, \partial\Omega_T)$	$E_{sd}(\Omega^{in}, \Omega_T)$	$E_\infty(u^{in}, u_T)$
Test (1)	$1e^{-3}$	$1e^{-3}$	5699	-0.889	$2.08e^{-3}$	$3.36e^{-3}$	$2.60e^{-3}$
Test (2)	$1e^{-4}$	$1e^{-4}$	14477	-0.299	$8.61e^{-4}$	$2.18e^{-3}$	$1.34e^{-3}$

Table 7.1: Details on the two-dimensional test of Zalesak's slotted disk.

We notice that a very nice accuracy can be obtained with a very small number of points. The sharp features of the slotted disk are well preserved, and the final interface presents no oscillation whatsoever. However, it seems difficult to say much more than that, especially when it comes to comparison with other methods, or order computation of the process: we use a numerical scheme for the advection equation that is admittedly low-order, and the accuracy of the process stems from the mesh adaptation process. The number of points (or triangles) of the meshes at hand can vary considerably from one iteration to the other depending on the 'wildness' of the interface to be captured; they also depend in an unclear way of the precision parameters ε and h_{min}.

7.6.2 Time-reversed vortex flow

The second test-case, proposed in [199], is a very good opportunity to investigate the behaviour of our method when dealing with velocity fields entailing serious deformations of the initial shape. In a unit square

domain, consider a disk of radius 0.15, centered at $(0.5, 0.75)$, evolving according to the velocity field

$$V(t,x,y) = \begin{pmatrix} -\sin^2(\pi x)\sin(2\pi y)\cos(\frac{\pi t}{T}) \\ \sin^2(\pi y)\sin(2\pi x)\cos(\frac{\pi t}{T}) \end{pmatrix}.$$

for $0 \leq t \leq T = 8$. Because of the vorticity of the velocity field, some parts of the initial disk are compressed, while some others become very stretched. What is more, the shape tends to become a more and more thin filament, and very small (or, in our case, very elongated) elements have to be put so that mesh resolution allows for a description of these parts. Because of the modulation in time, the shape reaches its most distorted state at time $t = T/2$, and has returned to its initial state at time T. We performed 20 intermediate time steps on this test-case, and the whole computation takes about 40 minutes. Figure 7.3 shows four steps of the computation. Comparison between the initial and final state is reported in figure 7.4, and subsequent details and error measures on this test-case can be found in table 7.2.

In order to emphasize the importance of mesh gradation, as discussed in section 7.5, we report in figure 7.5 the comparison between the first step of the presented computation, and the first step of the same computation, performed with the same parameters, but without any mesh gradation. Note that the 0 level set of the evolving function has not been represented in the latter case for it has been utterly 'lost' ! This shows that, even with a correct minimum size allowed, and with relevant precision parameters, the mesh adaptation process needs such a gradation in the mesh, for the sake of robustness: indeed, at the beginning of each iteration, the 0-level set of the evolving function is advected towards an area where the computational mesh is likely to be coarse, then refined with the m internal iterations, as expressed in algorithm 3. If no gradation in the mesh is enforced, this first 'virtual' advection may be too rough for capturing small details, that will be missed by the subsequent iterations.

ε	h_{min}	np	Volume loss (% initial vol.)	$d^H(\partial\Omega^{in}, \partial\Omega_T)$	$E_{sd}(\Omega^{in}, \Omega_T)$	$E_\infty(u^{in}, u_T)$
$1e^{-4}$	$3e^{-4}$	34862	-0.380	$1.81e^{-3}$	$8.56e^{-4}$	$1.83e^{-3}$

Table 7.2: Details on the two-dimensional time-reversed vortex flow test-case.

Remark 7.4. We hinted at the fact that, during a single iteration of the process, the 0 level set of the considered scalar function may be advected from an area where the mesh is suitably refined, towards an area where it is dramatically under-sampled (this is particularly likely to happen if the physical time step Δt is chosen very large). One could wonder whether it could prove beneficial to make a first refinement of the 'landing area', which could be achieved by applying the numerical scheme for advection to the size prescription (which is computationally inexpensive), before turning to the first internal iteration, just so as to get a well-sampled landing area. This merely amounts to roughly adding some degrees of freedom where we know the next 0 level set of interest will be located. Actually, this process is rather easy to implement numerically, at least when it comes to transporting the sole size prescription; things grow more tedious if we are to advect both size and orientation prescriptions. However, this yields disappointing numerical results: almost no improvement on the method has been observed when using to this technique.

7.6.3 Deformation test flow

An even more serious test case when it comes to shape distortion has been proposed by [288]. In a unit square domain, a disk of radius 0.15, centered at $(0.5, 0.75)$ is evolved according to the following velocity field:

$$V(t,x,y) = \begin{pmatrix} \sin(4\pi(x+\frac{1}{2}))\sin(4\pi(y+\frac{1}{2}))\cos(\frac{\pi t}{T}) \\ \cos(4\pi(x+\frac{1}{2}))\cos(4\pi(y+\frac{1}{2}))\cos(\frac{\pi t}{T}) \end{pmatrix}.$$

for $0 \leq t \leq T = 3$. Periodicity of the domain with respect to the top and bottom sides is enforced. Roughly speaking, this velocity field makes the domain composed of 16 vortices, and several parts of the disk are

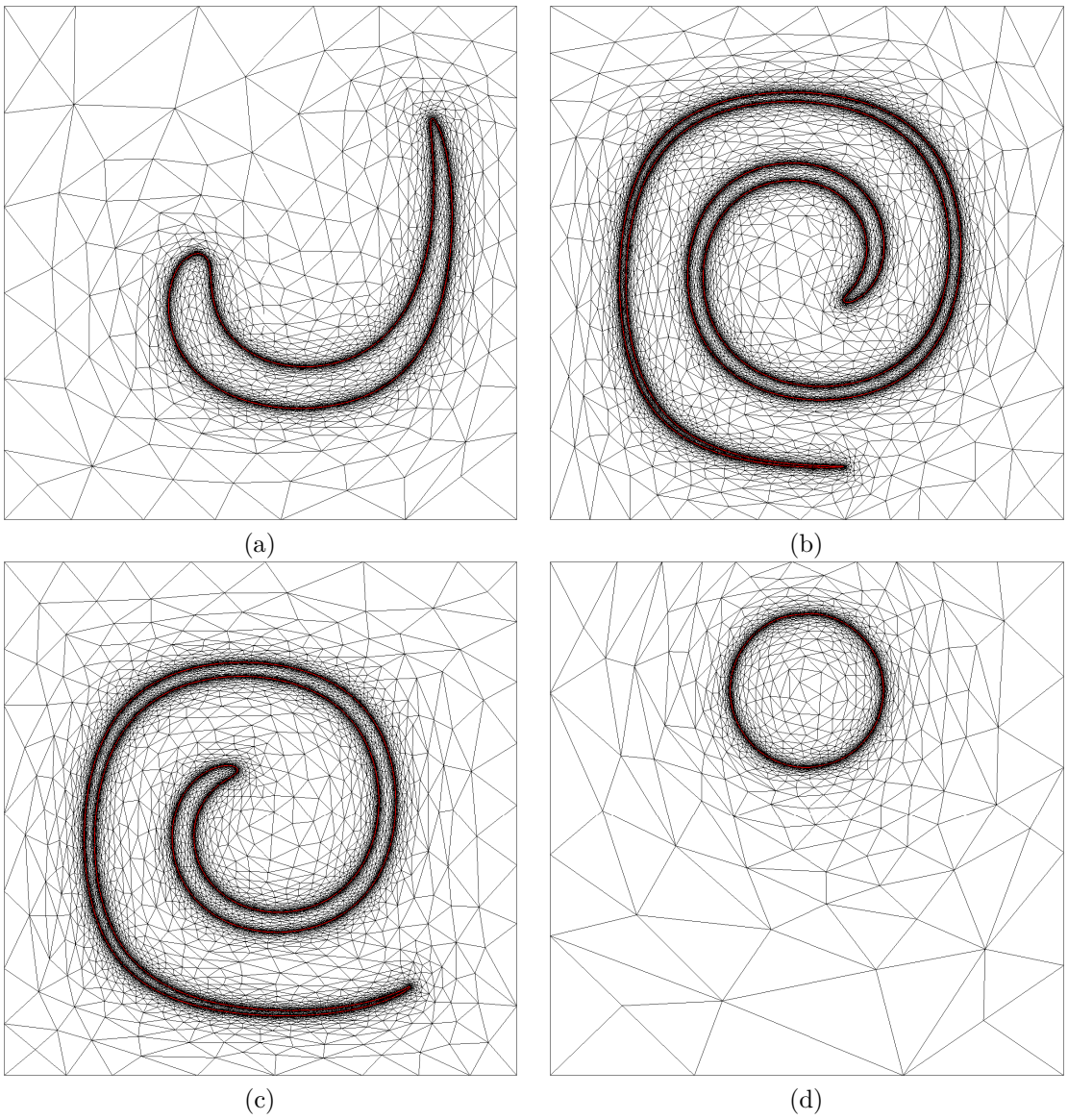


Figure 7.3: Four steps of the computation for the time-reversed vortex flow, *(a)* $t = 0.8$, *(b)* $t = 4$, *(c)* $t = 5.6$ and *(d)* $t = 8$, together with the corresponding 0-isovalues.

dragged by different ones, literally tearing the shape into pieces. See figure 7.6 for some illustrations of this test case, and figure 7.7 for a comparison between the initial and final interfaces. The whole computation takes around 70 minutes, and details as well as error measures regarding this test-case are to be found in table 7.3.

ε	h_{min}	np	Volume loss (% initial vol.)	$d^H(\partial\Omega^{in}, \partial\Omega_T)$	$E_{sd}(\Omega^{in}, \Omega_T)$	$E_\infty(u^{in}, u_T)$
$1e^{-3}$	$5e^{-5}$	56536	-0.097	$4.81e^{-3}$	$6.77e^{-3}$	$4.09e^{-3}$

Table 7.3: Details on the two-dimensional deformation flow test-case.

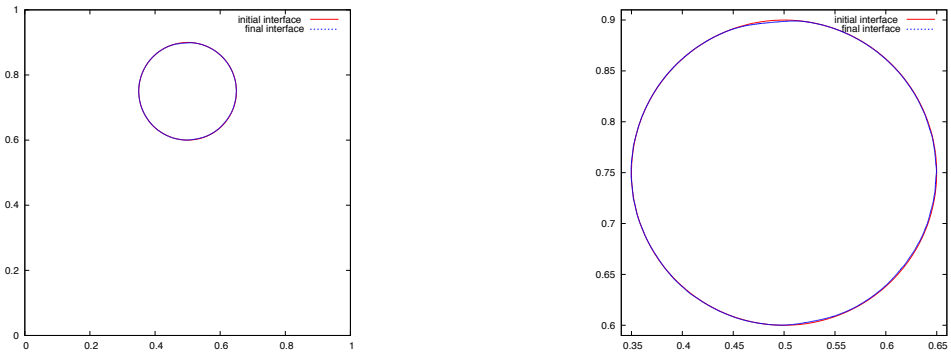


Figure 7.4: *(Left)* Superposition of the final disk (blue line) over the initial one (red line), and *(right)* zoom on the comparison.

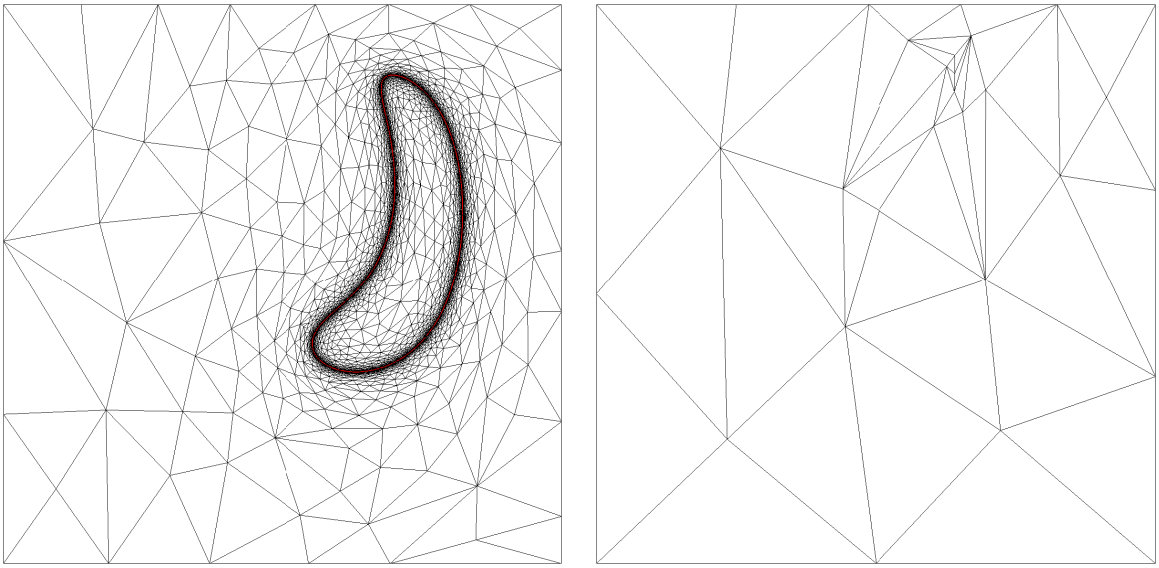


Figure 7.5: Evolving surface at time $t = 0.4$ with *(left)* and without *(right)* mesh gradation control.

7.6.4 Rotation of Zalesak's sphere

Very similarly to the first two-dimensional example comes the rotation of Zalesak's slotted sphere in 3 dimensions. In a unit cube, a sphere of radius 0.15 with a slot of width 0.15 is initially centered at $(0.5, 0.5, 0.25)$, and undergoes a uniform rotation with respect to the x-axis, corresponding to a velocity field:

$$V(t,x,y,z) = \begin{pmatrix} 0 \\ -(z-0.5) \\ (y-0.5) \end{pmatrix}.$$

over $0 \leq t \leq 2\pi$. This sphere shows both ridges and triple points, that are naturally the most difficult features to preserve throughout the advection process.

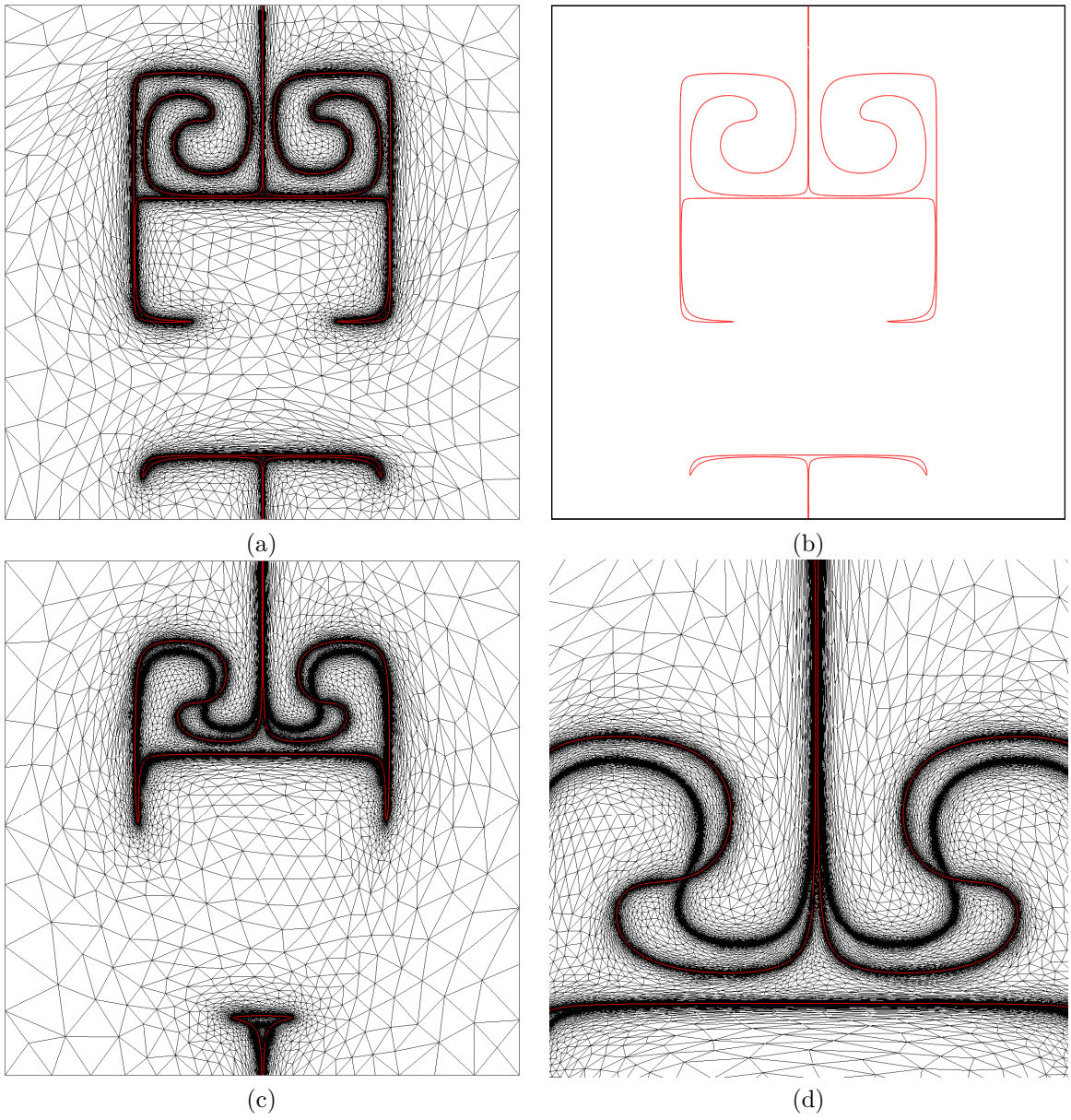


Figure 7.6: *(a)* Maximum elongation step ($t = T/2$), *(b)* corresponding 0-level set, *(c)* intersected mesh, adapted to both steps $t = 0.9$ and $t = 1.2$, *(d)* zoom on the intersected mesh (the surface associated to $t = 1.2$ is displayed in red).

We split the time interval into 8 subperiods, and work with parameters $\varepsilon = 0.005$, $h_{min} = 0.005$, in such a way each computational mesh has about $200,000$ points ($\approx 1,200,000$ tetrahedra).The sequence of obtained surfaces is displayed on figure 7.8, while a zoom on both initial and final states is to be found on figure 7.9, and several cuts into an 'intersected mesh' are to be found on figure 7.10. The whole computation process takes about 100 minutes. Comparison of the initial and final interfaces demonstrate a good accuracy of the method, even though, of course, the ridges and triple points of the surface have been a little smeared.

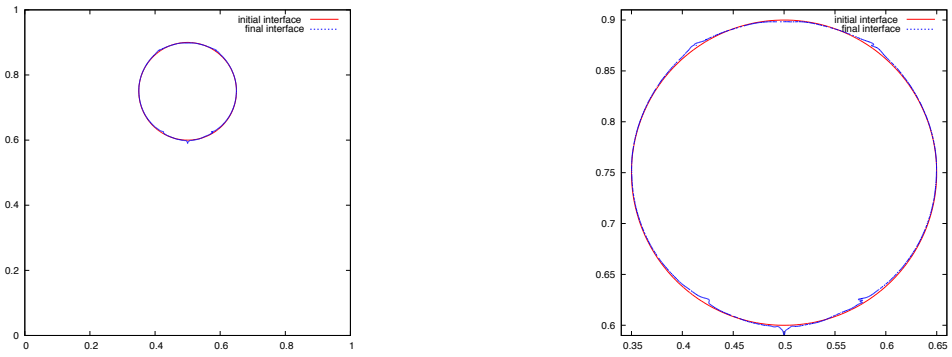


Figure 7.7: *(Left)* Superposition of the final disk (blue line) over the initial one (red line), and *(right)* zoom on the comparison.

7.6.5 Three-dimensional deformation test case

Eventually, we turn to yet another test case proposed in [199]. In a unit cube, a sphere of radius 0.15, centered at $(0.35, 0.35, 0.35)$ is made evolved according to the following velocity field:

$$V(t,x,y,z) = \begin{pmatrix} 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)\cos(\frac{\pi t}{T}) \\ -\sin(2\pi x)\sin^2(\pi y)\sin(2\pi z)\cos(\frac{\pi t}{T}) \\ -\sin(2\pi x)\sin(2\pi y)\sin^2(\pi z)\cos(\frac{\pi t}{T}) \end{pmatrix}.$$

for $0 \leq t \leq T = 3$. We split the time interval into 10 subperiods, and the results are reported in figure 7.11. Figure 7.12 displays two cuts in the most stretched interface of the evolution (the one at time $t = 1.5$), while figure 7.13 displays two cuts in two different adapted meshes (one is anisotropic, the other is isotropic) to the latter interface. The results presented in figure 7.11, which are the best obtained among different tests carried out with different parameters, are those corresponding to a computation held with isotropic adaptation with a minimum size parameter $h_{min} = 0.002$ (which amounts to anisotropic adaptation with very small precision parameters). The largest mesh of the computation is worth $3,763,497$ vertices, and the whole computation took about 21 hours.

Taking a close look at the displayed sequence, one realizes that the final interface is not exactly as smooth as the initial one, notably near its horizontal diameter; actually, this area corresponds to the most stretched zone at the maximum elongation time $t = T/2$, and is the most difficult to track accurately (see the results in e.g. [192] or [27] for similar behavior); of course, this effect vanishes with enhanced resolution.

The proposed method for solving the level set advection equation brings into play various tools, and it is interesting to wonder which parts exactly take most of the computational expense. For each example, we reported in table 7.4, the details of the longest iteration of the process, that is, the total time Δt_{it} of the iteration, and the percentage of it which has been spent in each main step of the process (note that the interpolation steps have been neglected). We notice that, understandably enough, the remeshing procedure is by far the most costly in every case, which is quite understandable since it is the point in where lies the main complexity of the method.



Figure 7.8: *Rotation of Zalesak's sphere: sequence of computed surfaces.*

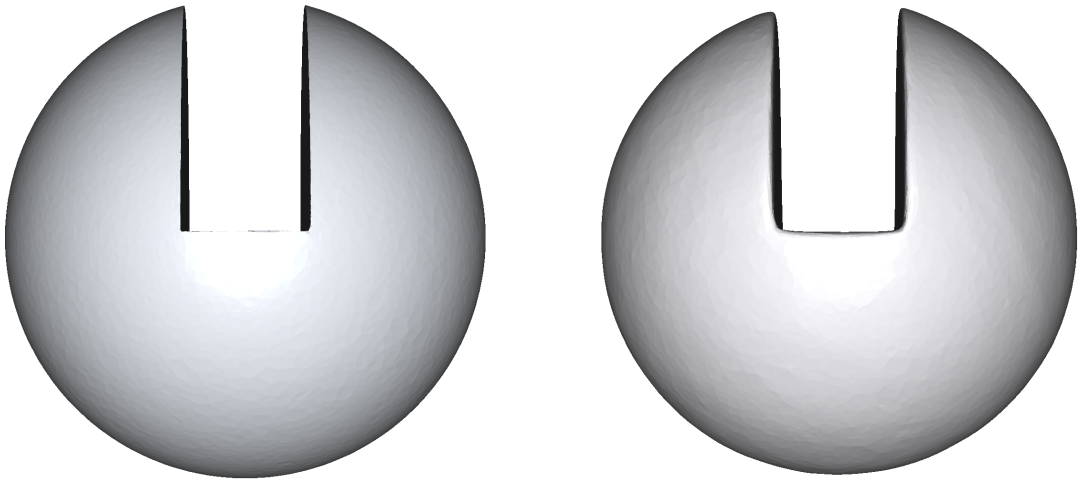


Figure 7.9: Zoom on initial Zalesak's sphere *(left)* and on Zalesak's sphere after a whole rotation *(right)*.

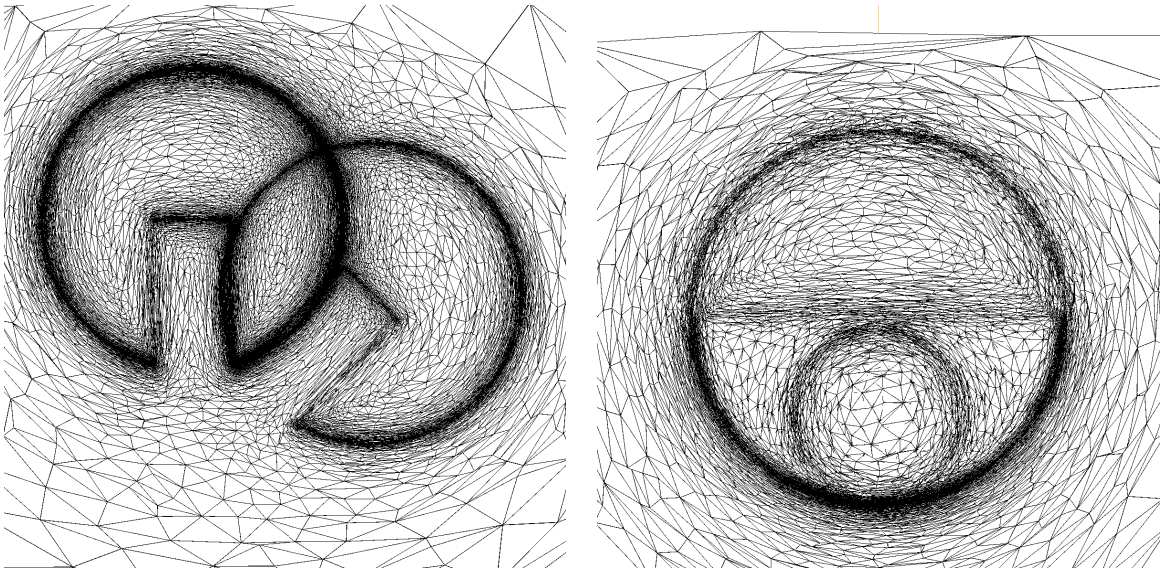


Figure 7.10: Cut on a mesh adapted with respect to both surfaces at $t = T/2$ and $t = 5T/8$, following a plane $x = cste$ *(left)* and following a plane $y = cste$ *(right)*.

	Δt_{it} (s)	cost of advection $(\%\Delta t_{it})$	cost of remeshing $(\%\Delta t_{it})$	cost of redistancing $(\%\Delta t_{it})$	cost of metric computations $(\%\Delta t_{it})$
Zalesak's disk (Test 2)	62.1	4.0	90.6	3.1	2.3
$2d$ time-reversed vortex flow	142.6	5.4	83.0	9.0	2.6
$2d$ deformation flow	253.1	4.7	87.4	7.9	3.0
Zalesak's sphere	1416	9.8	71.2	4.2	14.8
$3d$ deformation	12417	16.9	51.2	26.1	5.8

Table 7.4: Costs of the steps of the proposed algorithm.

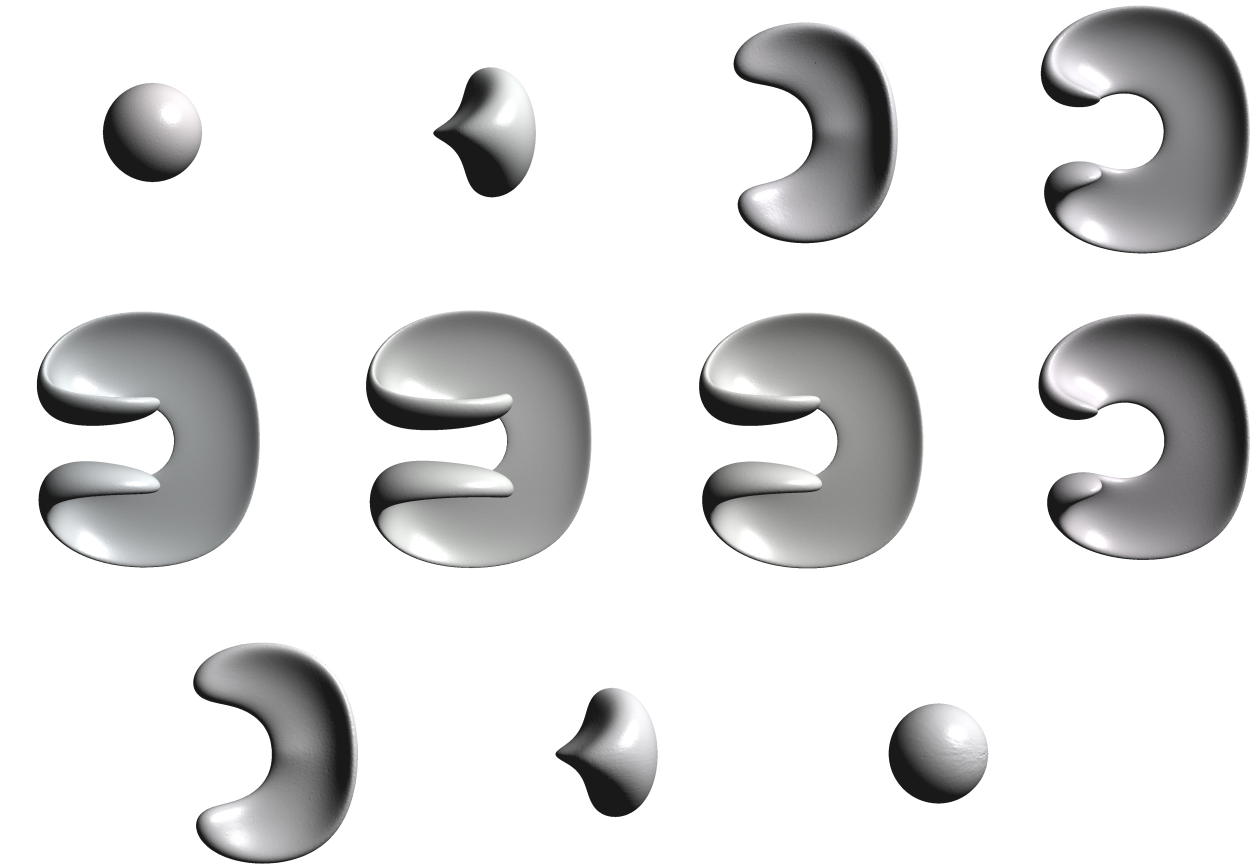


Figure 7.11: 3*d deformation test case: sequence of computed surfaces.*

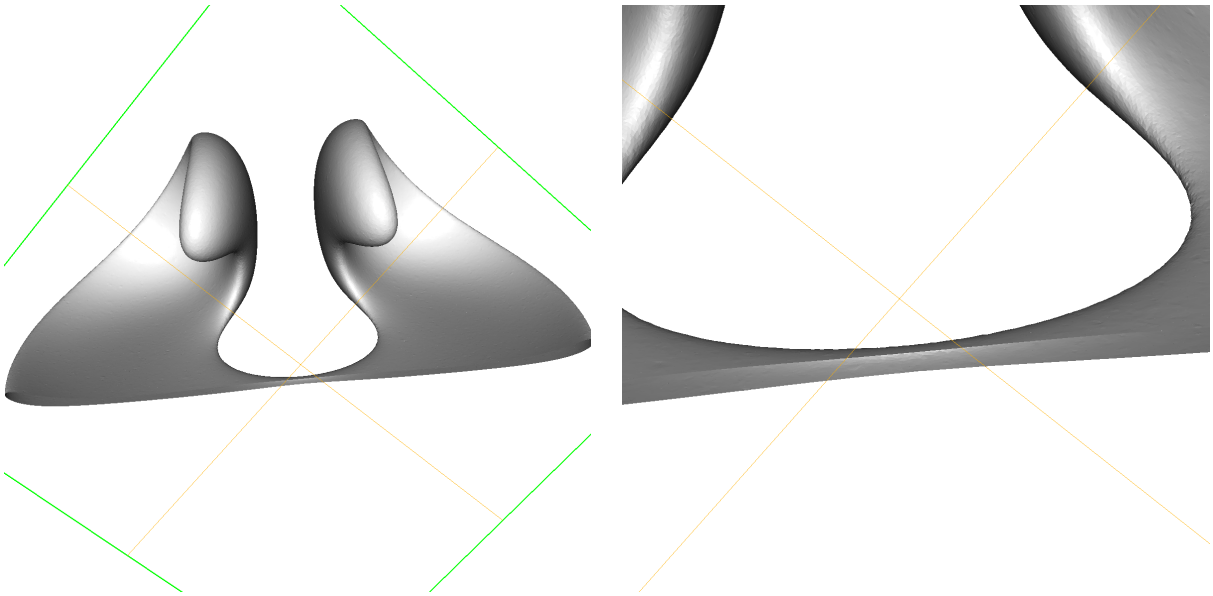


Figure 7.12: A cut in the most stretched interface, at time $t = 1.5$ *(left)*; a zoom on the cut *(right)*.

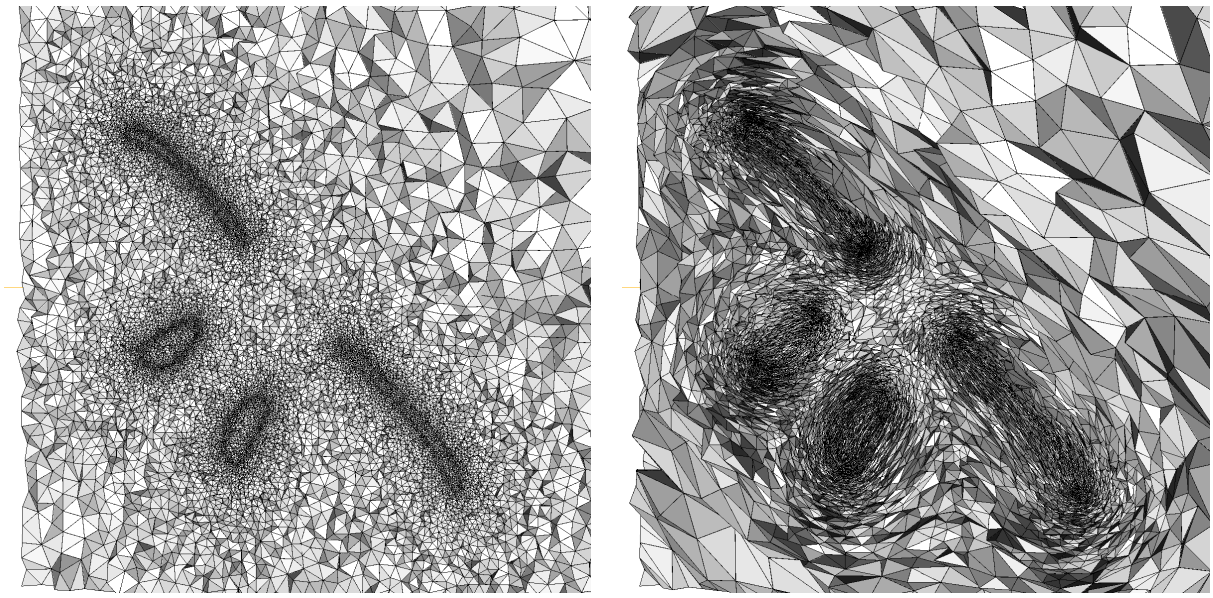


Figure 7.13: *(left)* A cut in an mesh adapted to the evolving interface at time $t = 1.5$ using isotropic mesh adaptation ($\approx 1,500,000$ points) and *(right)* anisotropic mesh adaptation ($\approx 700,000$ points).

Appendix

Here is a variation of classical Gronwall's lemma for the estimation of the discrepancy between the solutions of two ODEs associated to different vector fields (see [313] for proof).

Lemma 7.3. *Let $V_1, V_2 : \mathbb{R}^{d+1} \to \mathbb{R}^d$ two vector fields, V_1 being Lipschitz continuous in the second variable with constant k, and such that there exists $\varepsilon > 0$ with, for all $x \in \mathbb{R}^d$, $|V_1(x) - V_2(x)| \leq \varepsilon$. Let $x_1, x_2 : \mathbb{R}_+ \to \mathbb{R}^d$ some respective solutions to:*

$$\left\{ \begin{array}{rcl} x_1'(t) & = & V_1(t, x_1(t)) \\ x_1(0) & = & u_1 \end{array} \right. , \quad \left\{ \begin{array}{rcl} x_2'(t) & = & V_2(t, x_2(t)) \\ x_2(0) & = & u_2 \end{array} \right.$$

Then the following estimate holds

$$\forall t \in \mathbb{R}_+ , \ |x_1(t) - x_2(t)| \leq |u_1 - u_2| e^{kt} + \frac{\varepsilon}{k}(e^{kt} - 1).$$

Part IV

Three-dimensional surface and domain remeshing

Chapter 8

Discrete three-dimensional surface and domain remeshing

Contents

8.1 Remeshing of surface triangulations 280
8.1.1 Reconstruction of the geometry 281
8.1.2 Local reconstruction of the ideal surface from the discrete geometry 284
8.1.2.1 Choice of the three 'vertex' control points 284
8.1.2.2 Choice of the six 'curve' control points 285
8.1.2.3 Choice of the central coefficient 285
8.1.3 Description of the local remeshing operators 286
8.1.3.1 Edge split 286
8.1.3.2 Edge collapse 287
8.1.3.3 Edge swap 287
8.1.3.4 Vertex relocation 289
8.1.4 Definition of a size map adapted to the geometric approximation of a surface . . . 290
8.1.4.1 Meshes and metric size maps 290
8.1.4.2 A general L^∞ error estimate 291
8.1.4.3 A control over the Hausdorff distance between a smooth surface and an interpolating triangulation 291
8.1.4.4 Deduction of an isotropic size prescription 293
8.1.4.5 Gradation of the size map 294
8.1.5 The complete strategy 296
8.1.6 Numerical examples 297
8.1.6.1 Curvature-dependent surface remeshing of smooth surfaces 297
8.1.6.2 High-quality surface remeshing of mechanical parts 298
8.1.6.3 Remeshing of a surface with respect to an externally prescribed size map 299
8.1.6.4 Some miscellaneous surfaces remeshing 300
8.2 Discrete surface remeshing in the anisotropic context 303
8.2.1 A wee bit of Riemannian geometry 303
8.2.1.1 Notations 303
8.2.1.2 Riemannian distance and volume 303
8.2.1.3 Connections and parallel transport of vector fields on a Riemannian manifold 304
8.2.1.4 Connections and parallel transport of higher-dimensional objects on M . 306

8.2.1.5 Geodesics and the exponential map . . . 307
8.2.1.6 Expressions in local coordinates . . . 309
8.2.2 Numerical approximation of parallel transport on a submanifold of $\mathbb{R}^d$ . . . 310
8.2.2.1 Schild's ladder algorithm for approximating the parallel transport of a tangent vector on a Riemannian manifold . . . 310
8.2.2.2 Parallel transport of a bilinear form over a Riemannian manifold . . . 314
8.2.3 Definition of a suitable Riemannian structure for anisotropic surface remeshing . . 315
8.2.3.1 Definition of a Riemannian structure adapted to the geometric approximation of a surface . . . 315
8.2.3.2 Gradation of an anisotropic metric tensor field . . . 317
8.2.4 Metric tensor fields on triangulated surfaces in numerical practice . . . 318
8.2.4.1 A word about the numerical storage of fields of symmetric bilinear forms 318
8.2.4.2 Intersection of two fields of symmetric bilinear forms over Γ . . . 318
8.2.4.3 Interpolation of two symmetric bilinear forms along a curve . . . 319
8.2.5 Geometric anisotropic surface remeshing of a discrete surface . . . 320
8.2.5.1 Anisotropic mesh quality . . . 320
8.2.5.2 Vertex relocation in the anisotropic remeshing context . . . 321
8.2.5.3 The complete anisotropic remeshing strategy . . . 322
8.2.6 Numerical examples . . . 323
8.2.6.1 Anisotropic remeshing of triangulated surfaces presenting anisotropic features . . . 323
8.2.6.2 Anisotropic adaptation of a surface triangulation to a user-defined metric tensor field . . . 323
8.3 Discrete three-dimensional domain remeshing . . . 327
8.3.1 Description of the local remeshing operators . . . 328
8.3.1.1 Edge split . . . 328
8.3.1.2 Edge collapse . . . 328
8.3.1.3 Edge swap . . . 329
8.3.1.4 Node relocation . . . 331
8.3.2 Construction of a size map adapted to the geometric approximation of the ideal domain . . . 331
8.3.3 The complete remeshing strategy . . . 332
8.3.4 Numerical examples . . . 334
8.3.4.1 High-quality remeshing of smooth domains or mechanical parts . . . 334
8.3.4.2 Remeshing of a domain with respect to a user-defined size map . . . 334
8.4 Implicit domain meshing and applications . . . 339
8.4.1 Explicit discretization of the 0 level set of $\phi_{\mathcal{T}}$ into $\mathcal{T}$ . . . 340
8.4.2 From discrete domain remeshing to discrete domain and subdomains remeshing . . 341
8.4.3 Numerical examples and applications . . . 343
8.4.3.1 Meshing of the negative subdomains associated to some level set functions 343
8.4.3.2 Application to mesh generation from a possibly invalid surface triangulation 344

In this chapter, we investigate two problems of *discrete three-dimensional surface,* and *domain remeshing.*

By discrete surface remeshing, we mean that a discrete triangulation $\mathcal{S}$ is supplied, which accounts for an unknown continuous geometry, and may be arbitrarily under-sampled, over-sampled or of poor mesh quality, in senses that will be made precise later. The aim is then to carry out mesh modifications to transform $\mathcal{S}$ into a new, hopefully well-shaped, triangulation $\widetilde{\mathcal{S}}$ which accounts for the same underlying geometry.

Very similarly, discrete domain remeshing starts with a tetrahedral mesh $\mathcal{T}$, whose topological boundary is a surface triangulation $\mathcal{S}_{\mathcal{T}}$. Both $\mathcal{T}$ and $\mathcal{S}_{\mathcal{T}}$ may be of arbitrarily poor quality, and $\mathcal{S}_{\mathcal{T}}$ may be under-sampled, over-sampled,... The objective is then to transform $\mathcal{T}$ into a 'nice-quality' mesh $\widetilde{\mathcal{T}}$, which is a good approximation of the underlying continuous domain.

Both problems have a very broad range of applications; they are perhaps as ubiquitous as mesh generation techniques themselves in numerical simulation, since the initially supplied discretization of a surface, or a domain of interest is very often a (simplicial) mesh of very poor quality; this feature indeed jeopardizes the accuracy of most numerical techniques (finite element methods, finite volume methods to name a few). For instance, most of the triangulated surfaces arising from Computer Aided-Design (CAD) modeling are supplied under the so-called STL (STereoLithography) format, whose sole purpose is rendering: such triangulated surfaces are very good descriptors of the continuous geometries they are intended for, but are minimal in terms of number of elements, and may contain very stretched elements (see [38, 318] for further discussions on the issue of remeshing such surfaces). On a different note, many surface models of real-life structures (e.g. statues, monuments) are obtained by scanning processes [102, 310], and many biomedical data are produced owing to the Marching Cubes algorithm (see [208], or the short presentation of this method in Chapter 3, §3.2.1.3), which typically produce *valid*, yet over-sampled and very ill-shaped surface triangulations, and must be modified (e.g. decimated, enhanced in terms of quality, etc...) before any numerical treatment is possible. As far as 'volume' meshes are concerned, we have seen in Chapter 3 that most mesh generation algorithms (e.g. Delaunay-based algorithms, or advancing front methods) are mainly focused on constructing a mesh of a domain $\Omega \subset \mathbb{R}^d$ from the datum of a surface triangulation, and do not focus too much on the qualities of its elements. They should therefore by supplemented with a remeshing stage.

Admittedly, although each one of these two problems has its own specificities, they are closely connected, and we aim at dealingl with them in a common framework insofar as possible.

This aim influences many choices as far as the adopted remeshing strategy is concerned: the topic of discrete surface remeshing has been largely tackled (see e.g. the overviews in [18, 145], or that proposed in Chapter 3 §3.3.2) and several fundamentally different approaches are available, which can be roughly divided into two categories: *global remeshing approaches* use the provided triangulation only to infer a global (e.g. parametric) description of the continuous surface it is meant to account for, then generate a whole new mesh of this surface (see for instance [19, 265]). On the contrary, *local remeshing algorithms* only perform local operations on the supplied triangulation to transform it into a 'better' one through an iterative process [140, 294]. Although both approaches prove robust and efficient, they do not equivalently lend themselves to generalization to the context of domain remeshing. In the latter case, if each operation does not simultaneously affect both the surface and volume parts of the mesh, a whole new tetrahedral mesh has to be generated from the modified surface triangulation, and this may prove difficult, as we have discussed in Chapter 3. Our main concern in this chapter is to devise a robust approach. Indeed, we have in mind to rely on this algorithm as a component of a mesh evolution process; in this view, we though it better to rely only on a *local* remeshing approach, which iteratively transforms an always existing input mesh in the best possible way, and never has to generate a three-dimensional mesh of a domain from the sole datum of a surface triangulation.

The remainder of this chapter is organized as follows. Section 8.1 deals with discrete surface remeshing. It is intended to define as rigorously as possible its main purposes, then describes a possible strategy: the effective modifying operators acting on a surface mesh are detailed, as well as the way they are driven, which requires to rely on a *metric size map*, accordingly defined. Several numerical examples, emphasizing the different features of the proposed method are eventually presented. We then extend this strategy to the context of *anisotropic mesh adaptation* in section 8.2. After recalling supplementary material from Riemannian geometry, we describe a slightly different approach from the existing ones, which strengthens even more the paradigm of a Riemannian structure based anisotropic meshing initiated in [311], and has been extensively used since then. Last but not least, we describe how the previous approach for surface remeshing can be generalized to the problems of domain remeshing, and implicit domain meshing. A final appendix gathers some considerations about the well-known validity checks on local remeshing operators, which are most probably far anterior to this manuscript, but which we did not find exactly under this form in the

literature.

This chapter is a joint work with Cécile Dobrzynski and Pascal Frey. Parts of its contents have been submitted to publication as the journal article:

C. DAPOGNY, C. DOBRZYNSKI AND P. FREY, *Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems*, submitted (2013).

8.1 Remeshing of surface triangulations

In this section, we consider a discrete surface Γ embedded in $\mathbb{R}^3$, which is only known via a *triangulation*, i.e. a *mesh* $\mathcal{S} = (T_i)_{i=1,...,N_{\mathcal{S}}}$. In the sequel, we constantly identify $\mathcal{S}$ with the associated piecewise linear surface. We also make the following assumptions:

- the triangulation $\mathcal{S}$ is *conforming*, that is, the intersection between any two triangles T_i, T_j, $i \neq j$ amounts to either the empty set, a common vertex, or a common edge.
- the resulting surface is a compact orientable manifold, with or without boundary.

(8.1)

The discrete surface $\mathcal{S}$ is intended as an *interpolating* - potentially under-sampled, over-sampled, or ill-shaped - piecewise linear approximation of an unknown ideal surface Γ. We aim at performing local modifications on the original defining triangulation $\mathcal{S}$, thus producing a sequence $\mathcal{S}_1, ..., \mathcal{S}_n$ of triangulations (which will all be denoted by $\mathcal{S}$ in the following, to save notations) getting closer and closer to a final approximation $\widetilde{\mathcal{S}}$ of Γ, which is:

- *well-shaped*, that is, composed of *high-quality* elements. As we have seen in Chapter 3, the choice of a quality function for evaluating elements is very important in numerical practice, and the retained criterion should be neither too 'strict', nor too 'lenient' in tagging elements as 'bad' elements. After several trials, we made up our minds to appraise the shape of a surface triangle T in terms of the quantity $Q(T)$ defined by:
$$Q(T) = \frac{4\,\mathrm{Vol}(T)}{\sqrt{e_1^2 + e_2^2 + e_3^2}}, \tag{8.2}$$
where e_1, e_2, e_3 are the edges of T.
- *close to* Γ, in the sense that the Hausdorff distance $d^H(\widetilde{\mathcal{S}}, \Gamma)$ between $\widetilde{\mathcal{S}}$ and Γ is no larger than a user-defined tolerance ε (see Figure 8.1).

As hinted at above, the ideal surface Γ is unknown, and so as to guide the local modifications of the surface mesh, we need to 'guess' it (actually, we should say 'invent it') from the data at hand, i.e. the triangulation $\mathcal{S}$. One could think of mainly two ways for doing so:

- The first one consists in constructing a whole underlying surface Γ to the discrete surface $\mathcal{S}$ as a pre-processing stage for remeshing. This surface Γ is then kept in memory, for instance under the form of a *parametrization* $\sigma : U \to \Gamma$ (this is the point of view retained in [19]). The *parameter space* U can be an open subdomain of $\mathbb{R}^2$ [122, 265], the surface $\mathcal{S}$ itself [294], etc... This parametrization is stored, and at each stage of the remeshing procedure, the current triangulation $\mathcal{S}$ is compared to Γ in order to assess the geometric approximation. Such an approach is well-posed on the theoretical side: all the produced triangulations are compared to one single continuous surface; however the storage and numerous comparisons involved generally prove quite costly. See [136] for a review of surface parametrization techniques.
- On the other side, we could limit ourselves to constructing *local* models for Γ: at any stage of the process, when an operation around a node x of the current triangulation $\mathcal{S}$ is performed, a local parametrization $\sigma_U : U \to V \subset \Gamma$ from an open set $U \subset \mathbb{R}^2$ to a neighborhood V of x in Γ is computed

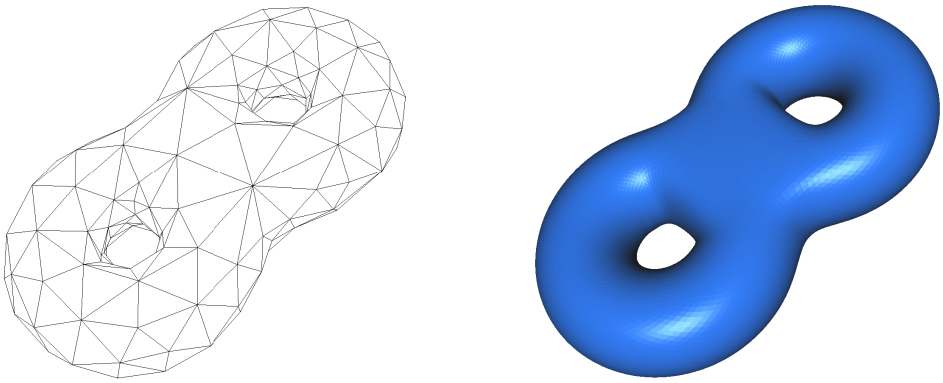


Figure 8.1: Poor geometric approximation *(left)* of a smooth surface *(right)*.

from local features of $\mathcal{S}$ around x. This approach is undoubtedly more efficient from the computational point of view, but it raises a fundamental difficulty: the triangulation $\mathcal{S}$ supporting the information from which these local parameterizations are generated inescapably changes from one stage to another, which challenges the very idea of considering *one* continuous model Γ for $\mathcal{S}$.

In what follows, we rely on the second approach, referring with some abuse in terminology to *the* ideal surface Γ associated to the various surface meshes at hand during the process, neglecting the fundamental dependance of this ideal surface on the triangulation used to compute it. We will however see that some heuristics can guarantee that this generated geometric support is not 'too much' altered along the steps of the process.

We are thus led to set rules to infer a piece of the underlying surface Γ from the datum of a piece of the current discrete geometry $\mathcal{S}$ at the investigated stage. We already mentioned the fact that we want to assume minimal input information around the considered surface, i.e. we only have at our disposal a set of triangles, without further structure. This is the case, for instance, when the input surface mesh stems from a mesh evolution procedure such as the one we will describe in Chapter 9. Hence, as a pre-processing stage, we need to extract additional geometrical features about the surface Γ from $\mathcal{S}$ (normal vectors, ridge edges,...). This is the purpose of the next section.

8.1.1 Reconstruction of the geometry

Let $\mathcal{S} = (T_i)_{i=1,\dots,N_\mathcal{S}}$ be a surface mesh, about which no more information than the sole lists of the constituting triangles and vertices are supplied. The first step in associating to $\mathcal{S}$ a relevant continuous geometry Γ consists in identifying different kinds of points, edges,... corresponding to significant geometrical features of $\mathcal{S}$ (and thus Γ) which will constrain the admissible operations carried out during remeshing. In this view, we operate a distinction between (see Figure 8.2 [1] for an illustration):

1. *required edges*: any user-specified edge which should not be affected by the process.
2. *open edges*: the ideal surface Γ associated to the input triangulation $\mathcal{S}$ may be a compact manifold *with boundary*. Then, the *open edges* of the mesh are the edges of the topological boundary $\partial\mathcal{S}$ of $\mathcal{S}$.

1. On the right: free model from `http://artist-3d.com/free_3d_models`.

3. *geometric edges*, or *ridges*: edges delimiting two portions of surface which intersect with a sharp angle. Ridges can be identified from the input triangulation by relying on a threshold value on the dihedral angle between pairs of adjacent triangles.
4. *reference edges*: different regions, supporting different labels, may exist on the input triangulation, corresponding for instance to different material properties. The *reference edges* of mesh $\mathcal{S}$ are defined as the edges at the interface between two triangles $T_i \neq T_j$ carrying different labels.
5. ordinary edges.

A fairly close classification holds for the vertices of $\mathcal{S}$: one then talks about *ridge points* (points lying on a ridge edge), *open points*, *reference points*,... and *singular points*, which are points that arise as endpoints of at least three special edges, and thus cannot be considered as 'normal vertices', lying on a ridge curve, reference curve, ...

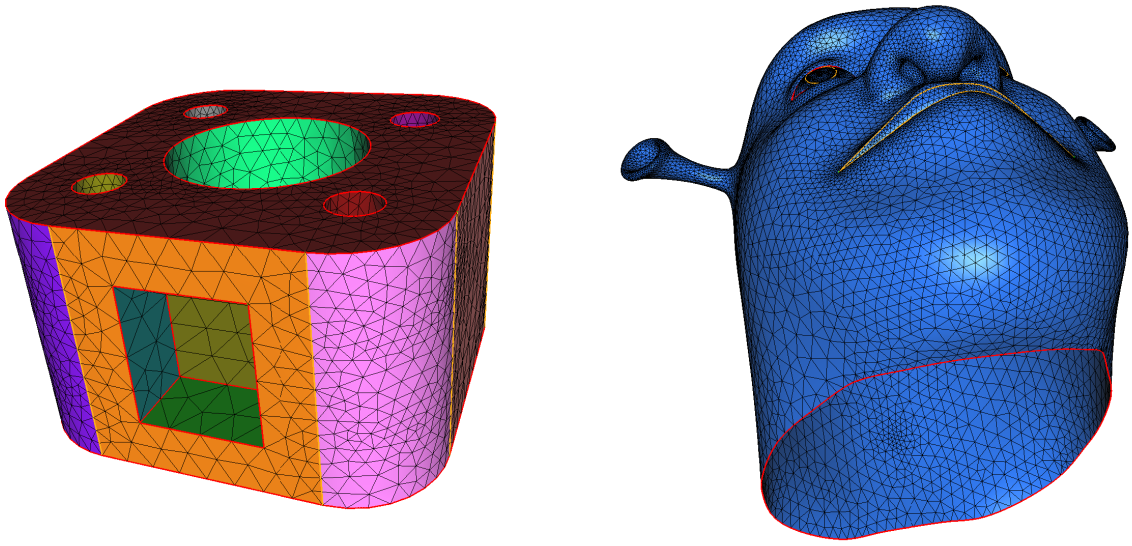


Figure 8.2: *(Left):* Ordinary edges (in black), reference edges (in yellow) and ridges (in red) on a triangulation. Different colors on triangles account for different labels. The points belonging to at least three reference, ridge or required edges are singular points (in green); *(right)*: open curve on a triangulation (in red).

Once such a classification of vertices and edges of $\mathcal{S}$ is operated, $\mathcal{S}$ is endowed with an *orientation* (which is possible because of our initial assumptions (8.1)). Roughly speaking, this consists in *locally* opposing two sides separated by $\mathcal{S}$ in a *globally* consistent fashion. For instance, if $\mathcal{S}$ is the topological boundary of a three-dimensional polyhedral domain Ω, the choice of an orientation for $\mathcal{S}$ boils down to deciding if either Ω or $\mathbb{R}^3 \setminus \overline{\Omega}$ is the interior domain associated to $\mathcal{S}$. In concrete terms, the orientation procedure for a surface triangulation $\mathcal{S}$ consists in reordering the three vertices of each triangle $T \in \mathcal{S}$ so that the *direct* normal vectors to all the triangles of $\mathcal{S}$ consistently point towards one side of $\mathcal{S}$. This is classically achieved by using a hashing procedure [145].

Next, from the features of $\mathcal{S}$, we compute approximations of some geometrical data attached to Γ, at the vertices of $\mathcal{S}$. Depending on the nature of the considered vertex of $\mathcal{S}$, different quantities prove relevant (see for instance [145], §19.2.1); see Figure 8.3:

1. No further information is attached to singular points, or required entities, for they are by essence bound not to be affected by remeshing.

2. In the neighborhood of a regular vertex x, the surface Γ is smooth (at least of class $\mathcal{C}^1$). It will prove useful to compute an approximation of $n(x)$, the unit normal vector to Γ at x. This is achieved from the discrete surface $\mathcal{S}$ by using a weighted sum of the normal vectors to the triangles of $\mathcal{B}_{\mathcal{S}}(x)$ of the form:
$$n(x) \approx \frac{\sum_{T\in\mathcal{B}(x)} \alpha_T \, n_T}{\left|\sum_{T\in\mathcal{B}(x)} \alpha_T \, n_T\right|},$$
where α_T are coefficients in $[0,1]$ such that $\sum_{T\in\mathcal{B}(x)} \alpha_T = 1$, and n_T is the unit normal to a triangle T. Note that such a formula can only make sense if $\mathcal{S}$ has already been oriented. Several choices are possible as for the values of α_T. Some authors are used to taking them all equal to one another, others take each α_T proportional to the area of T, etc... As far as we are concerned, all these reconstruction formulae worked more or less equivalently well, and we retained the former one.

3. According to the above terminology, non singular ridge vertices x of $\mathcal{S}$ are vertices belonging to a *ridge curve* of Γ, that is a curve delimiting two portions of surfaces which intersect at a sharp dihedral angle. Such points enjoy two normal vectors (one for each piece of surface), say $n_1(x), n_2(x)$, which are reconstructed in the discrete context in the same way as for regular vertices, and a tangent vector $\tau(x)$ (the tangent vector to the ridge curve), which is uniquely determined by its belonging to two distinct 'tangent' planes.

4. At a reference vertex x of $\mathcal{S}$, which is neither singular, nor ridge at the same time, two geometrical features are of interest, namely the unit normal vector $n(x)$ to Γ at x (which is approximated as previously), and the unit tangent vector $\tau(x)$ to the reference curve going through x and delimiting two regions carrying different labels. Once again, several formulae exist in the literature for the approximation of $\tau(x)$ [145]. Here, we use:
$$\tau(x) \approx \frac{x_0x_1}{|x_0x_1|},$$
where x_0x and xx_1 are the two reference edges of $\mathcal{S}$ sharing x as an endpoint.

These supplementary pieces of information about the ideal surface Γ, approximated from the discrete geometry, allow us to define a local surface model for Γ; this is the purpose of the next section.

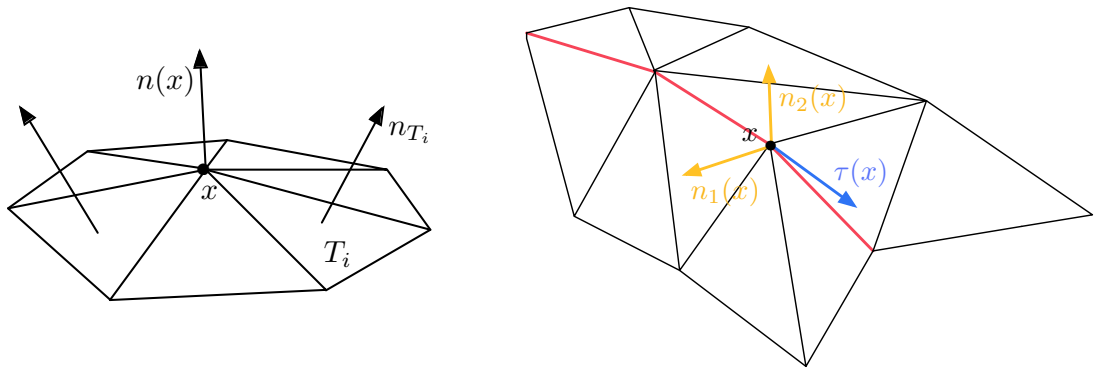


Figure 8.3: Approximation of the normal vector to Γ at x as a (weighted) average of the normals n_{T_i} to the triangles of the ball of x, in the case of a regular vertex x *(left)*; two normal vectors and a tangent vector associated to a ridge vertex x *(right)*.

8.1.2 Local reconstruction of the ideal surface from the discrete geometry

The purpose of this section is to described the rules for inferring the local geometry of the ideal surface Γ around a triangle T of $\mathcal{S}$ from the entities attached to T and its three vertices, introduced in the previous section. This surface model is broadly based on the one introduced in [316].

In the following, we make the assumption that each triangle $T = a_0a_1a_2 \in \mathcal{S}$ accounts for a *smooth* portion of Γ, whose boundaries may still be ridge curves, reference curves, etc... The portion of Γ associated to T is modeled as a cubic piece of surface $\sigma(\widehat{T})$, where

$$\widehat{T} := \big\{(u,v) \in \mathbb{R}^2,\ u \geq 0,\ v \geq 0,\ w := 1-u-v \geq 0\big\}$$

is the reference triangle in the plane, and each component of $\sigma : \widehat{T} \to \mathbb{R}^3$ is a polynomial of total degree 3 in the two variables $u, v \in \widehat{T}$. Equivalently, one could parametrize the same piece of surface directly from the triangle T (i.e. without bringing the reference element in the plane into play), using another mapping $\phi : T \to \mathbb{R}^3$; of course then, $\sigma = \phi \circ A_T$, where $A_T : \widehat{T} \to \mathbb{R}^3$ is the unique affine mapping which transforms $\widehat{T}$ into T. It will turn out convenient to write σ under the form of a Bézier cubic polynomial (see [131] for exhaustive details around the assets of such a representation):

$$\forall (u,v) \in \widehat{T},\ \ \sigma(u,v) = \sum_{\substack{i,j,k \in \{0,1,2,3\} \\ i+j+k=3}} \frac{3!}{i!j!k!} w^i u^j v^k\, b_{i,j,k}, \tag{8.3}$$

where the $b_{i,j,k} \in \mathbb{R}^3$ are *control points*, yet to be specified. See Figure 8.4 for an illustration.

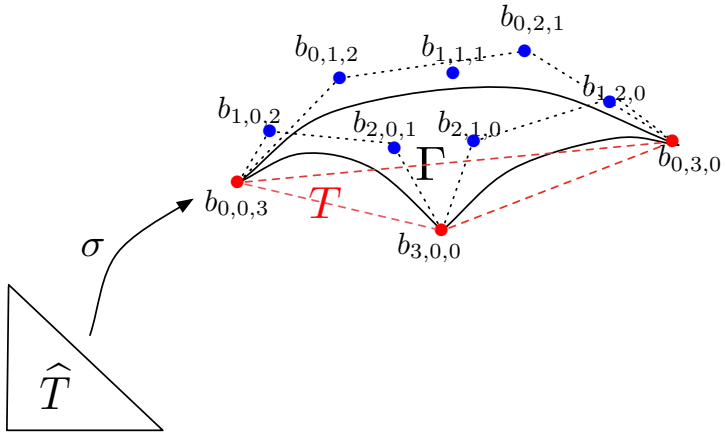


Figure 8.4: A piece of parametric Bézier cubic surface, associated to a triangle $T \in \mathcal{S}$, with control points $b_{i,j,k}$.

The *boundary curves* $\gamma_0, \gamma_1, \gamma_2$ of the portion of surface $\sigma(\widehat{T})$ are respectively:

$$\forall t \in [0,1],\ \gamma_0(t) = \sigma(1-t,t),\ \gamma_1(t) = \sigma(0,t),\ \gamma_2(t) = \sigma(t,0).$$

The choice of the control points $b_{i,j,k}$ is - at least partially - dictated by the geometrical features of Γ we approximated in the first step, or by other requirements we may want our local geometry to meet.

8.1.2.1 Choice of the three 'vertex' control points

We already mentioned that the triangulated surface $\mathcal{S}$ interpolates the ideal surface Γ, i.e. its vertices lie on Γ. This prompts the natural choice of the three vertices of T as the three extremities of $\sigma(\widehat{T})$. As the

vertices a cubic Bézier polynomial (see formula (8.3)) are $b_{3,0,0}, b_{0,3,0}$, and $b_{0,0,3}$, we impose:

$$b_{3,0,0} = a_0, \quad b_{0,3,0} = a_1, \text{ and } b_{0,0,3} = a_2. \tag{8.4}$$

8.1.2.2 Choice of the six 'curve' control points

We imposed that $\sigma(\widehat{T})$ should be a smooth piece of surface. In particular, $\sigma(\widehat{T})$ has a tangent plane $T_{a_i}\Gamma$ at each vertex a_i. Thanks to the reconstructed information of the previous section, it is rather easy to approximate each tangent plane $T_{a_i}\Gamma$ - and equivalently *the* normal vector to Γ at a_i, with respect to the considered piece of surface, say n_i: for instance, if a_i is a regular vertex of $\mathcal{S}$, take the reconstructed normal at a_i; in case it is a ridge vertex, take the 'most consistent' of the two reconstructed normals at a_i, etc...

On the other hand, the whole geometry of Bézier curves and surfaces can be expressed in terms of their control points. More specifically, a mere derivation in (8.3) shows that, for instance, the tangent vector at a_0 to the boundary curve γ_2 is $3(b_{2,1,0} - b_{3,0,0})$, and that to γ_1 is $3(b_{2,0,1} - b_{3,0,0})$.

Hence, the tangent plane to $\sigma(\widehat{T})$ at a_0 is the expected tangent plane $T_{a_0}\Gamma$ provided $b_{2,1,0}$ and $b_{2,0,1}$ are chosen in such a way that $(b_{2,1,0} - a_0)$ and $(b_{2,0,1} - a_0)$ are non colinear, and both orthogonal to n_0. Similar relations hold when it comes to a_1, a_2 and the control points $b_{0,2,1}, b_{1,2,0}, b_{1,0,2}$ and $b_{0,1,2}$.

This still allows some latitude as for the choice of these coefficients. In [316], the authors propose to take, for instance, $b_{2,1,0}$ as the orthogonal projection over $T_{a_0}\Gamma$ of the point $(a_0 + (a_1 - a_0)/3)$. Instead of this, we propose to incorporate the requirement that we want our local surface reconstruction by means of σ to be as independent as possible from the support triangle T used for its computation (even if, of course, complete independence is out of reach, as discussed in the introduction), in order to devise some heuristics as for the choice of these control points.

Loosely speaking, this requirement means that we would like the three boundary curves $\gamma_0, \gamma_1, \gamma_2$ to be independent from the choice of the points on these curves used to generate them. Because on any Riemannian manifold two 'close enough' points are connected by a unique geodesic curve (see section 8.2.1), a way to enforce this independency would be to choose their control points so that $\gamma_0, \gamma_1, \gamma_2$ are geodesics of $\sigma(\widehat{T})$, that is, curves with constant speed. This property can in turn only be enforced in some kind of 'weak sense': for instance, in the case of γ_0 (similar conditions hold for γ_1, γ_2), we imposed that $\gamma_0'(0)$ should be colinear to the orthogonal projection of $(a_2 - a_1)$ over $T_{a_1}\Gamma$, and have a fixed norm $|\gamma_0'(0)| = |a_2 - a_1|/3$, and symmetrically for $\gamma_0'(1)$. Doing so uniquely determines the six coefficients attached to the boundary curves.

8.1.2.3 Choice of the central coefficient

We simply take:

$$b_{1,1,1} = m + \frac{m - v}{2}, \;\; v := \frac{a_0 + a_1 + a_2}{3} \;; \;\; m := \frac{b_{2,1,0} + b_{2,0,1} + b_{1,2,0} + b_{0,2,1} + b_{1,0,2} + b_{0,1,2}}{6},$$

which guarantees that, if there exists a *quadratic* polynomial parametrization $\widetilde{\sigma} : \widehat{T} \to \mathbb{R}^3$ whose boundary curves $t \mapsto \widetilde{\sigma}(1-t, t)$, $\widetilde{\sigma}(t, 0)$ and $\widetilde{\sigma}(0, t)$ coincide with γ_0, γ_1 and γ_2 respectively, then $\sigma = \widetilde{\sigma}$ over $\widehat{T}$ [131]. Note that the choice of the central coefficient $b_{1,1,1}$ does not affect the geometry of $\gamma_0, \gamma_1, \gamma_2$.

Remark 8.1. The above rules for computing the four control points along each boundary curve γ_i only involve geometric data attached to this curve (or to its endpoints). This fact has a very important practical consequence. It implies that our rules for generating a piece of Γ are consistent from one triangle to its neighbor, that is, if $T_i, T_j \in \mathcal{S}$ share a common edge pq, the underlying boundary curve associated to pq via the local parametrization generated from T_i is the same as from T_j. Actually, the proposed rules for generating portions of Γ from triangles of $\mathcal{S}$ are also rules for generating curves drawn on Γ from edges of $\mathcal{S}$.

Putting parametrization σ under the form (8.3) allows for a close and fast evaluation of the Hausdorff distance between the considered triangle $T \in \mathcal{S}$ and the corresponding piece of ideal surface $\sigma(\widehat{T}) \subset \Gamma$.

Indeed, $\sigma(\widehat{T})$ is comprised in the convex hull of the control points $b_{i,j,k}$, because for all $(u,v) \in \widehat{T}$, $\sigma(u,v)$ is a convex combination of the $b_{i,j,k}$. As a consequence, one easily sees that

$$d^H(T, \sigma(\widehat{T})) \leq \max_{\substack{l=0,1,2,\\ i+j+k=3}} d(a_l, b_{i,j,k}), \tag{8.5}$$

where $d^H(.,.)$ stands for the Hausdorff distance between compact subsets of $\mathbb{R}^3$. A similar estimate holds when it comes to controlling the Hausdorff distance between each edge a_1a_2, a_0a_2, or a_0a_1 of T and the corresponding boundary curve γ_0, γ_1, or γ_2 of $\sigma(\widehat{T})$.

At this point, we are able to measure (at least to control) how far each triangle T of the considered triangulation $\mathcal{S}$ lies from the ideal surface Γ. This is a precious guide when it comes to deciding whether an operation performed on $\mathcal{S}$ (see the next section 8.1.3) enhances, or does not degrade too much the geometric approximation of Γ by means of $\mathcal{S}$.

8.1.3 Description of the local remeshing operators

This section is devoted to the description of the operations we intend to carry out on surface triangulations. The four presented operators are *local*, i.e. they affect only a small neighborhood of the considered entities. Note that they are widely used in the literature (see e.g. [56, 145]); the forthcoming descriptions are essentially meant to specify how exactly they fit in our particular setting.

8.1.3.1 Edge split

This is the main tool when it comes to enriching an undersampled triangulation (with respect to whatever criterion). Splitting an edge pq of a triangulation $\mathcal{S}$ consists in introducing a new vertex m in $\mathcal{S}$, then replacing pq by the two edges pm and mq (the local geometry of $\mathcal{S}$ being updated accordingly). In our context, the new point $m \in \Gamma$ is inserted directly on the ideal surface Γ, and an appropriate location is achieved by computing the curve $[0,1] \ni t \mapsto \gamma(t) \in \Gamma$ associated to pq, then taking $m = \gamma(\frac{1}{2})$.

Now, there are several ways to split edges within a triangulation $\mathcal{S}$: the simplest one consists is to travel all the edges in $\mathcal{S}$, and as soon as an edge complying with the splitting criterion is met, perform the splitting operation. However, we noticed that repeatedly splitting elements along only one edge tends to produce very ill-shaped elements in the long run.

For this reason, we favored another approach, which consists in identifying in a first step all the edges of $\mathcal{S}$ that should be split, then to proceed to splitting, resorting to patterns on each triangle T (possibly reiterating the process if some edges need to be split several times); see Figure 8.5 for an example of splitting of a triangle along one or three edges.

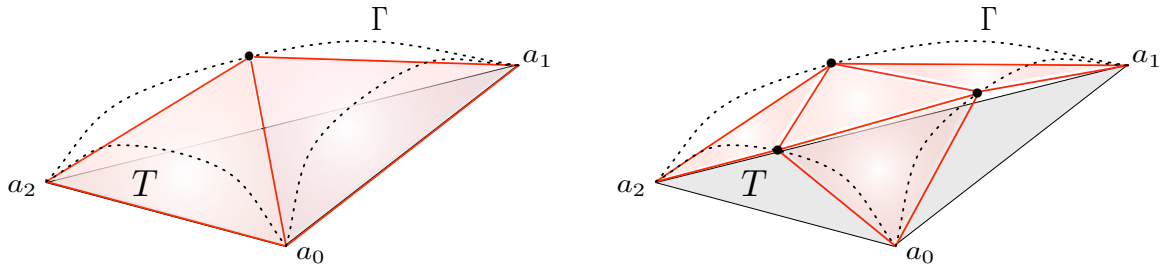


Figure 8.5: Splitting of *(left)* one or *(right)* three edges of triangle T, positioning the three new points on the ideal surface Γ (dotted).

Under the (formal) assumption that the local parameterizations σ described in section 8.1.2 represent pieces of the invariant surface Γ, splitting a triangle $T \in \mathcal{S}$ always enhances the approximation of Γ.

8.1.3.2 Edge collapse

This is the key ingredient in decimating an oversampled triangulation. Collapsing an edge pq of $\mathcal{S}$ consists in merging pq into a single point, say q itself for simplicity. One or two triangles end up deleted during the process, those of the *shell* $\mathcal{S}h(pq)$ of pq, i.e. the triangles sharing pq as an edge, and the other triangles of $\mathcal{S}$ which shared p as a vertex then enjoy q instead (see figure 8.6).

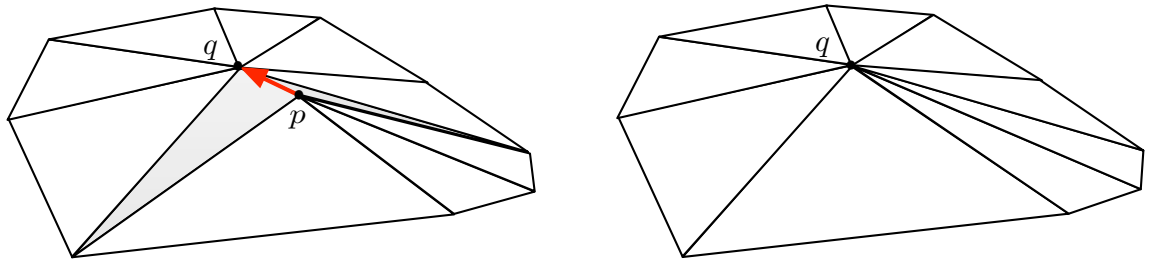


Figure 8.6: Collapse of point p over q. Both triangles in the shell of pq (in grey) disappear.

This operator ought to be driven with much caution:

- Obviously, not all edges of $\mathcal{S}$ are subject to collapse, and some coherence with geometrical data has to be respected. For instance, singular points must not be collapsed, ridge points must not be collapsed onto regular points, etc...
- Collapsing a point $p \in \mathcal{S}$ onto a point q may lead to an invalid configuration, because some triangles in the ball $\mathcal{B}(p)$ may end up inverted (see figure 8.7 for a two-dimensional example). It is therefore mandatory to perform checks on the triangles of $\mathcal{B}(p)$, anticipating the resulting configuration before triggering the effective collapse operation.
- Even if the resulting configuration is valid, it may turn out to be 'folded', with respect to the initial one, i.e. the collapse operation may have entailed too large a deviation between the normal vectors to the updated triangles. Such cases must be prevented, for instance by enforcing a maximal tolerance (with an arbitrary threshold value) as for the authorized deviation between normal vectors to adjacent triangles.
- The resulting triangulation from this operation is in general rougher than the initial one as a geometric approximation of Γ. One must check, thanks to the tests expressed in formula (8.5), that $\mathcal{S}$ stays within the prescribed range of Γ in terms of Hausdorff distance.

These two operators are mainly *sampling* operators in the sense they allow to transform an arbitrary initial triangulation $\mathcal{S}$ of an ideal surface Γ into a new triangulation which enjoys a correct node density with respect to the prescribed criterions (desired edge length, tolerance over the geometric approximation of Γ,...). On the contrary, the next two ones are essentially meant to enhance the *quality* of a surface triangulation (and, to a lesser extant, the accuracy of the approximation of the associated underlying surface Γ).

8.1.3.3 Edge swap

As we shall see in section 8.3, this operator is the only one which is fundamentally different between the cases of surface triangulations and three-dimensional 'volume meshes'. It plays a key role in improving the overall mesh quality, and acts only on the connectivities of a mesh $\mathcal{S}$, leaving its vertices unchanged. Swapping a common edge pq to two adjacent triangles $T_1 = pqa$, $T_2 = pbq \in \mathcal{S}$ consists in replacing pq by

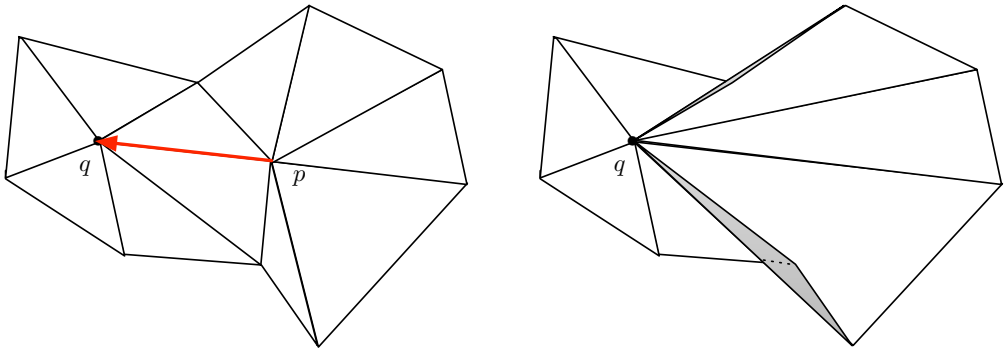


Figure 8.7: Impossible collapse of p onto q in $2d$: *(left)* before and *(right)* after the operation: the two grey elements are inverted.

the alternate edge ab; the two triangles T_1 and T_2 become respectively $\widetilde{T_1} = abq$ and $\widetilde{T_2} = apb$ during the process (see figure 8.8).

This operator as well should be handled with caution:

- So as not to violate the geometry of the ideal surface Γ, only *regular* edges, that is edges that are neither singular, nor ridge, nor reference edges should be held eligible for swap.
- Swapping a common edge pq to two surface triangles $T_1, T_2 \in \mathcal{S}$ may very well invalidate a configuration (for instance, in the case of a two-dimensional volume mesh, this happens when the union of both triangles $T_1 \cup T_2$ is not convex), or result in a 'folded' triangulation. For this reason, checks must be performed over the expected configuration before proceeding indeed to the operation.
- Such an edge swap may also deteriorate (or improve) the geometric approximation of Γ by means of $\mathcal{S}$. Checks on the resulting configuration, applying the control (8.5) to the expected triangles $\widetilde{T_1}, \widetilde{T_2}$ must be performed.

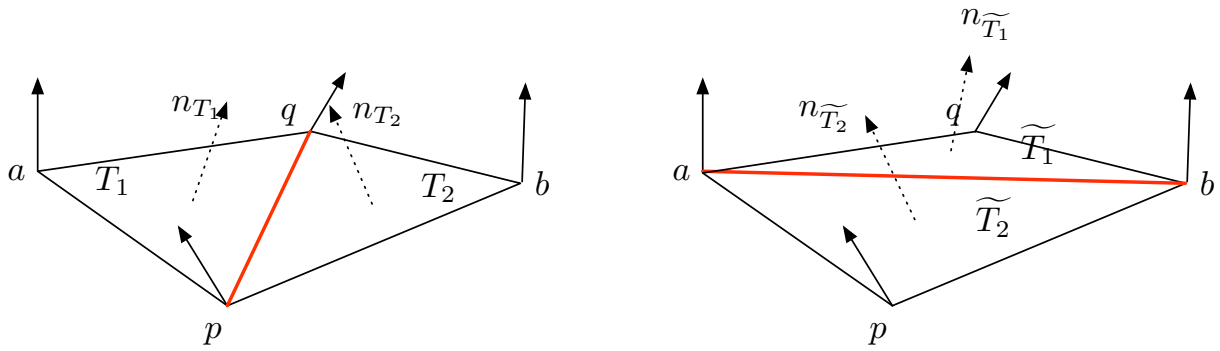


Figure 8.8: Swap of edge pq: triangles T_1, T_2 are updated to $\widetilde{T_1}, \widetilde{T_2}$, a configuration more consistent with the geometric data.

Pre-empting our study of $3d$ domain remeshing (see Section 8.3), it is convenient to notice that swapping a common edge pq to two triangles $T_1 = paq$ and $T_2 = pbq$ is equivalent to splitting pq at its midpoint m, then collapsing either of the newly created edges ma or mb (see Figure 8.9). Actually, one may prove that in the $2d$, or $3d$ 'volume' remeshing context, the swapped configuration is valid if and only if, for any position of m in the segment pq, the collapse of m onto a and onto b yields a valid configuration.

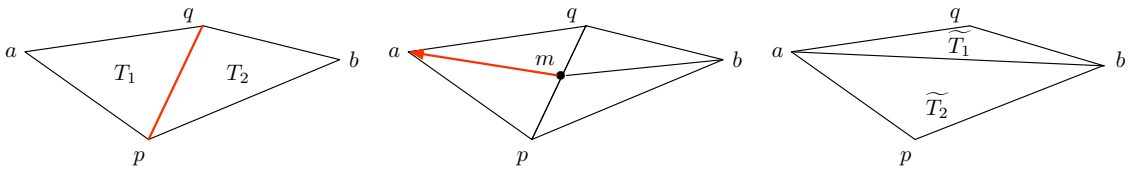


Figure 8.9: *(Left)* swap of edge pq by combining an edge split and an edge collapse: *(middle)* pq is split by introducing its midpoint m, then *(right)* m is collapsed onto a (or b) to deliver the swapped configuration.

8.1.3.4 Vertex relocation

This last operator mainly serves the purpose of improving the overall mesh quality. It consists in moving a vertex p of $\mathcal{S}$, while all the other positions of the vertices of $\mathcal{S}$ are fixed, as well as its connectivities. This is achieved by using a simple procedure (see Figure 8.10): suppose for instance p is a regular vertex (a similar, simpler procedure holds in the other cases), and consider its ball $\mathcal{B}(p) = (T_i)_{1,\dots,N_p}$. Consider the orthogonal projections $\overline{T_i}$ of the T_i onto the tangent plane $T_p\Gamma$ of Γ at p. From then, a new position $\overline{q} \in T_p\Gamma$ is computed for p in the tangent plane $T_p\Gamma$: one may for instance follow the direction of the gradient of the quality function of the triangle with the worst quality among the T_i with respect to the position of $p \in \Gamma$ (this gradient naturally belongs to $T_p\Gamma$, or resort to more heuristic procedures, such as taking p as a weighted sum of the endpoints of triangles $\overline{T_i}$: this is the well-known *Laplacian smoothing procedure* [134]. Note that a wide literature exists as for the choice of this position $\overline{q}$ (see for instance the very interesting procedure proposed by [139], based on non smooth optimization, or the overview in [145]). Actually, we tested several of these procedures and observed they yield similar results in terms of the resulting mesh quality, in the context of our surface remeshing algorithm. We thus retained the simplest - and most heuristic - among them, which consists in taking $\overline{q}$ as the centroid of the union of the triangles $\overline{T_i}$.

Then, $\overline{q}$ belongs to one of the $\overline{T_i}$, say $\overline{T_{i_0}}$, and is projected first to $q \in T_{i_0}$, then onto Γ by taking $\phi(q)$, where $\phi : T_{i_0} \to \Gamma$ is the local parametrization of Γ corresponding to T_{i_0} (see Section 8.1.2).

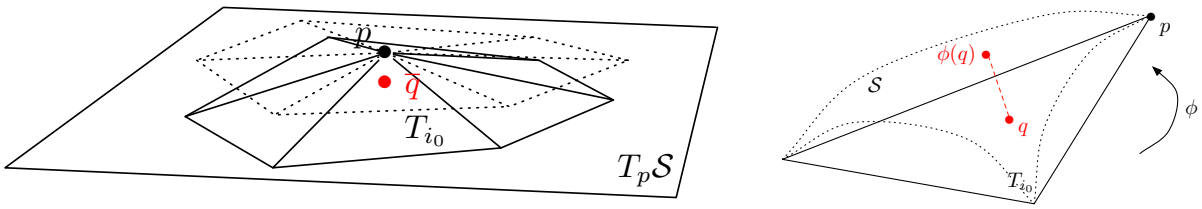


Figure 8.10: Relocation of vertex p: *(left)* $\mathcal{B}(p)$ is projected onto T_pS, and an optimal position $\overline{q}$ is sought on T_pS. *(Right)* the associated point $q \in T_{i_0}$ is projected onto Γ.

As in the case of edge collapse and edge swap, this operation may invalidate $\mathcal{S}$, or degrade the associated geometric approximation of Γ. Hence the need to check the validity of the resulting configuration, its proximity with Γ, and its quality - the chosen criterion fro computing the relocation position for p being heuristic, there is no strong guarantee that the quality of $\mathcal{S}$ will end up increased in the process.

We now have at hand several operators for modifying an initial triangulation $\mathcal{S}$, intended as a computational approximation of an underlying surface Γ into a new triangulation $\widetilde{\mathcal{S}}$, which retains the geometric features of Γ, and stays close from Γ (in terms of Hausdorff distance) within a prescribed range.

Unfortunately, that is not enough to ensure the resulting triangulation will be well-shaped. Indeed, these operators can only analyze very local configurations, then proceed to the respective operation provided it does not degrade the geometric approximation of Γ beyond a certain tolerance. So to speak, the hitherto devised strategy for remeshing lacks a *global vision* of where to add points in priority, where they should be removed in priority, etc... Since [311], a very convenient tool to add such intelligence to the process has been that of *metric tensor field*.

8.1.4 Definition of a size map adapted to the geometric approximation of a surface

We turn to the definition of a *size map* adapted to the control of the geometric approximation of Γ by $\mathcal{S}$ in terms of the Hausdorff distance $d^H(\mathcal{S}, \Gamma)$ between both surfaces.

8.1.4.1 Meshes and metric size maps

In this short and very elementary subsection, we introduce several definitions and notations around the concept of an *(isotropically) adapted surface mesh to a given size specification*, which will come in handy in the following.

Let $\mathcal{S}$ be a triangulation in $\mathbb{R}^3$ agreeing with hypotheses (8.1), and let Γ be the underlying ideal surface. A *size map* h on Γ is nothing more than a *positive* scalar function $h : \Gamma \to \mathbb{R}$, which loosely speaking encodes the 'local feature size' in the neighborhood of each point of Γ, in a way to be specified in Subsection 8.1.4.4.

Definition 8.1. *Let Γ be a smooth surface, $I = [a, b]$ a closed interval of $\mathbb{R}$. Let also $\gamma : I \to \mathbb{R}$ be a piecewise differentiable curve on Γ, i.e. a continuous curve such that there exist $a = t_0 < t_1 < ... < t_k = b$ with each restriction $\gamma|_{[t_i, t_{i+1}]}$ being differentiable. The length $\ell_h(\gamma)$ of γ with respect to h is:*

$$\ell_h(\gamma) = \int_a^b \frac{|\gamma'(t)|}{h(\gamma(t))} dt. \tag{8.6}$$

Hence, a size map $h : \Gamma \to \mathbb{R}$ allows to measure lengths of edges of $\mathcal{S}$: indeed, if pq is such an edge, and γ is the associated curve on the ideal surface Γ (see remark 8.1), with a small abuse in terminology, we will call the quantity $\ell_h(\gamma)$ the *length of pq with respect to h*, and denote it $\ell_h(pq)$.

Remark 8.2. Without explicit reference to any size map on Γ, we denote as $\ell(\gamma)$ (resp. $\ell(pq)$) the *Euclidean length* of a curve γ (resp. of the underlying edge pq), that is the one associated to the constant size map $h \equiv 1$ over Γ.

Once such a size map h is defined on Γ, we aim at modifying $\mathcal{S}$ into a new triangulation $\widetilde{\mathcal{S}}$ of Γ whose edges pq have unit length with respect to h. The motivation for doing so comes from the following fact: imposing that the edges pq of the resulting triangulation $\widetilde{\mathcal{S}}$ of Γ which are associated to a portion of surface $U \subset \Gamma$ have constant (Euclidean) length ℓ_0 is equivalent to imposing that these edges have unit length $\ell_h(pq)$ if the size map h is defined by: $h(x) = \ell_0$, $x \in U$. Hence, defining a (non constant) size map h over Γ is a way to generalize this idea of size feature to *local* prescriptions.

In numerical practice, the size map h cannot be directly defined on Γ, and must be supported by the triangulation $\mathcal{S}$ itself, in a discrete way. In our case, it is stored at the vertices of $\mathcal{S}$ (which belong to Γ), then interpolated whenever it is required elsewhere; for instance, let pq an edge of $\mathcal{S}$, $\gamma : [0, \ell(\gamma)] \to \Gamma$ the associated curve on Γ *parametrized by its arc length* ($\gamma(0) = p$, $\gamma(\ell(\gamma)) = q$). Then $h(\gamma(t))$ is approximated by the formula:

$$\forall t \in [0, \ell(\gamma)], \;\; h(\gamma(t)) \approx (1 - \frac{t}{\ell(\gamma)})h(p) + \frac{t}{\ell(\gamma)} h(q), \tag{8.7}$$

which accounts for a linear interpolation of the size prescriptions with respect to the (intrinsic) parametrization of γ. Of course, as $\mathcal{S}$ is modified during the remeshing process, the numerical approximation of h carried by $\mathcal{S}$ changes accordingly. Nevertheless, the choice of the approximation rules for Γ (see section 8.1.2) should ensure that it does not change 'too much' from its original definition, in the course of the remeshing process.

At this point, defining a *Riemannian structure* over Γ in order to encode the local size prescription may seem a bit artificial. However, we will consider in section 8.2 the more general case where there are *several* size prescriptions (depending on directions) are imposed at a given point $x \in \Gamma$. Then, the real benefit of this formalism will appear more clearly.

8.1.4.2 A general L^∞ error estimate

We now recall a very elementary, yet very powerful L^∞ error estimate for the interpolation error of a smooth enough function over a simplex in $\mathbb{R}^d$, $d = 2, 3$; see [24] for a proof, or [135] for variants around the affine approximation of (smooth) curves, or (smooth) vector-valued functions on a simplex in $\mathbb{R}^d$. Note that we have already encountered this result in Chapters 6 and 7.

Theorem 8.1. *Let $K \subset \mathbb{R}^d$ ($d = 2$ or 3) a (closed) simplex, and f a function of class $\mathcal{C}^2$ on K. Denote as $\pi_K f$ the affine interpolate of f on K, that is,*

$$\forall x \in K,\ \pi_K f(x) = \sum_{i=0}^{d} f(a_i)\lambda_i(x),$$

where a_i are the vertices of K and λ_i the barycentric coordinate functions in K. Then,

$$||f - \pi_K f||_{L^\infty(K)} \leq \frac{1}{2}\left(\frac{d}{d+1}\right)^2 \max_{x\in K}\max_{y,z\in K} |\mathcal{H}(f)(x)|(yz, yz)$$

where, for a symmetric matrix $S \in \mathcal{S}_d(\mathbb{R})$, which admits the following diagonal shape in orthonormal basis $S = P\begin{pmatrix}\lambda_1 & & 0\\ & \ddots & \\ 0 & & \lambda_d\end{pmatrix}P^T$, we denote $|S| = P\begin{pmatrix}|\lambda_1| & & 0\\ & \ddots & \\ 0 & & |\lambda_d|\end{pmatrix}P^T$.

8.1.4.3 A control over the Hausdorff distance between a smooth surface and an interpolating triangulation

Using theorem 8.1 allows to derive a small result around the geometric approximation of a surface by means of an associated surface triangulation, in a particular case. In this section, we shall rely on the material around the *signed distance function* to a domain in $\mathbb{R}^d$ recalled in Chapter 4, §4.2.

The purpose of this section is by no means to prove a powerful result, under optimal assumptions, but only to present an easy and comprehensive result which we shall use as a guideline in defining a size map (and, later, a metric tensor field) associated to the geometric approximation of a surface. From this standpoint, only the result expressed in Theorem 8.2 will be useful in the sequel, and the proofs, which are only given for the sake of completeness, may be skipped.

Let us consider a surface $\Gamma \subset \mathbb{R}^d$ which arises as the boundary of a bounded domain $\Omega \subset \mathbb{R}^d$ (we are only interest in the case $d = 3$ in this chapter, but the forthcoming considerations hold in the general case without additional difficulty). Let also $\mathcal{S}$ be a surface triangulation of Γ. We make the following assumptions:

1. $\Omega \subset \mathbb{R}^d$ is a connected bounded domain of class $\mathcal{C}^2$.
2. The surface mesh $\mathcal{S}$ is a *valid* triangulation (in the sense given in Chapter 3 §3.1.1).
3. $\mathcal{S}$ is an *interpolating* triangulation of $\partial\Omega$, i.e. all the vertices of $\mathcal{S}$ lie on $\partial\Omega$.
4. The surface $\mathcal{S}$ is a compact submanifold of $\mathbb{R}^d$.
5. The whole triangulation $\mathcal{S}$ is comprised in some neighborhood $\partial\Omega_h = \left\{x \in \mathbb{R}^d,\ d(x,\partial\Omega) < h\right\}$, for some $h < \text{reach}(\partial\Omega)$.
6. For any point $x \in \partial\Omega$, the *ray* $\text{ray}_{\partial\Omega}(x)$ of $\partial\Omega$ emerging from x intersects $\mathcal{S}$ at most once. Recall from Chapter 4, §4.2 that $\text{ray}_{\partial\Omega}(x)$ is defined by:

$$\text{ray}_{\partial\Omega}(x) := \{y \in \mathbb{R}^d \text{ such that } d_\Omega \text{ is differentiable at } y \text{ and } p_{\partial\Omega}(y) = x\},$$

where d_Ω is the signed distance function to Ω, and $p_{\partial\Omega}$ is the almost everywhere defined projection application onto $\partial\Omega$.

(8.8)

We then have:

Theorem 8.2. *Let $\Omega \subset \mathbb{R}^d$ a domain, and $\mathcal{S}$ a surface triangulation of $\partial\Omega$, both of them satisfying (8.8). Denoting by d_Ω the signed distance function to Ω, one has:*

$$d^H(\partial\Omega, \mathcal{S}) \leq \frac{1}{2}\left(\frac{d-1}{d}\right)^2 \max_{T\in\mathcal{S}} \max_{x\in T} \max_{y,z\in T} |\mathcal{H}(d_\Omega)(x)|(yz, yz). \tag{8.9}$$

Proof. From the very definition of the Hausdorff distance, we have to show that both $\rho(\partial\Omega, \mathcal{S}) := \sup_{x\in\partial\Omega} d(x, \mathcal{S})$ and $\rho(\mathcal{S}, \partial\Omega) := \sup_{x\in\mathcal{S}} d(x, \partial\Omega)$ are controlled by the right-hand side of (8.9).
First step: control over $\rho(\mathcal{S}, \partial\Omega)$. Let $T \in \mathcal{S}$, and $x \in T$. Then,

$$\begin{aligned} d(x, \partial\Omega) &= |d_\Omega(x)| \\ &= |d_\Omega(x) - \pi_T d_\Omega(x)|, \end{aligned}$$

because the three vertices of T belong to $\partial\Omega$, which implies $T \subset \ker(\pi_T d_\Omega)$. What's more, because $T \subset \partial\Omega_h$ and $h < \text{reach}(\partial\Omega)$, d_Ω is of class $\mathcal{C}^2$ in a neighborhood of T (see Chapter 4 §4.2). Using theorem 8.1 on T, which is a simplex of dimension $d-1$, and taking the supremum over all $x \in T$, and all $T \in \mathcal{S}$ produces:

$$\rho(\mathcal{S}, \partial\Omega) = \sup_{x\in\mathcal{S}} d(x, \partial\Omega) \leq \frac{1}{2}\left(\frac{d-1}{d}\right)^2 \max_{T\in\mathcal{S}} \max_{x\in T} \max_{y,z\in T} |\mathcal{H}(d_\Omega)(x)|(yz, yz).$$

Second step: control over $\rho(\partial\Omega, \mathcal{S})$. We need the following useful lemma, whose proof is postponed to the end of this section:

Lemma 8.1. *Under the hypotheses (8.8), the application $q_\mathcal{S} : \partial\Omega \to \mathcal{S}$, defined as*

$$\forall x \in \partial\Omega,\ q_\mathcal{S}(x) = \text{the unique point } y \in \mathcal{S} \text{ in segment }]x - hn(x), x + hn(x)[$$

is well-defined. It is a homeomorphism, whose inverse mapping is the restriction $p_{\partial\Omega}|_\mathcal{S}$ of the projection application onto $\partial\Omega$ to $\mathcal{S}$.

Now, let $x \in \partial\Omega$; we have

$$d(x, \mathcal{S}) \leq d(x, q_\mathcal{S}(x)) = d(q_\mathcal{S}(x), \partial\Omega) \leq \rho(\mathcal{S}, \partial\Omega),$$

where the equality $d(x, q_\mathcal{S}(x)) = d(q_\mathcal{S}(x), \partial\Omega)$ holds owing to the material recalled at Chapter 4 §4.2. Using the first step ends the proof. □

Remark 8.3.

- Point (5) in conditions (8.8) imposes that $\mathcal{S}$ must remain 'close' to the boundary $\partial\Omega$, i.e. it must not cross its skeleton (see Chapter 4 4.2 for definitions). It is a necessary condition for Theorem 8.2 to apply, as is shown in Figure 8.11, left.
- Point (6) in conditions (8.8) is also a necessary condition for Theorem 8.2, and it is not a consequence of the five other points, as is shown in Figure 8.11, right. Actually, we believe that this point could arise as a consequence to some hypothesis on the approximation of the normal vector field to $\partial\Omega$ by that of $\mathcal{S}$ (which often turns out to be a desirable hypothesis, as far as theoretical studies are concerned [176]) - which is obviously terrible in the example of Figure 8.11, right.

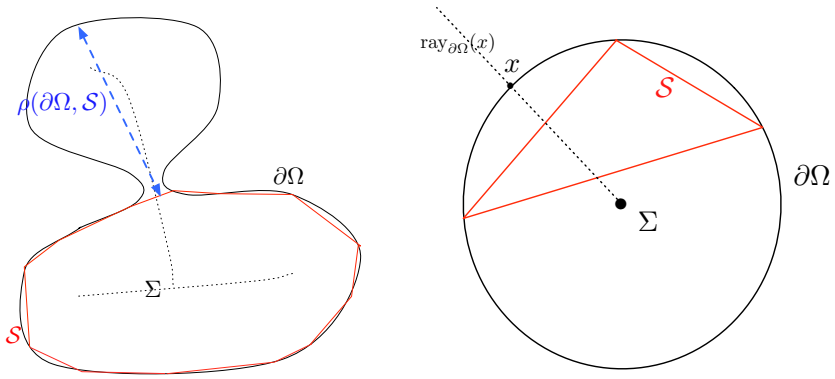


Figure 8.11: In two space dimensions, *(left)* example of a triangulation $\mathcal{S}$ of $\partial\Omega$ violating the condition (5) of hypothesis (8.8): $\rho(\mathcal{S}, \partial\Omega)$ can be very small, with very high $\rho(\partial\Omega, \mathcal{S})$, *(right)* example of a triangulation $\mathcal{S}$ of $\partial\Omega$ which violates condition (6) of hypothesis (8.8).

Eventually, we turn to the proof of lemma 8.1, which owes much to arguments present in [23]:

proof of lemma 8.1. Judging from the properties of the projection application $p_{\partial\Omega}$ (see again Chapter 4 §4.2), the only thing we have to do is to prove that $q_{\mathcal{S}}$ is well-defined and bijective.

First, it is easy to see that $q_{\mathcal{S}}$ is injective, since if there exists $x, y \in \partial\Omega$, and $t_x, t_y \in]-h, h[$ such that $x + t_x\, n(x) = y + t_y\, n(y)$, one has immediately:

$$x = p_{\partial\Omega}(x + t_x\, n(x)) = p_{\partial\Omega}(y + t_y\, n(y)) = y.$$

Second, as $p_{\partial\Omega}$ is continuous from $\mathcal{S} \subset \partial\Omega_h$ into $\partial\Omega$, and because $\mathcal{S}$ is compact, so is the image $p_{\partial\Omega}(\mathcal{S}) \subset \partial\Omega$. Moreover, $p_{\partial\Omega} : \mathcal{S} \to \partial\Omega$ is actually a local homeomorphism; it is indeed easy to check in the vicinity of any point $x \in \mathring{T}$, where T is a triangle of $\mathcal{S}$. In the case where x belongs to at least two triangles of $\mathcal{S}$, it is a consequence of the sixth point of hypotheses (8.8). Consequently, $p_{\partial\Omega} : \mathcal{S} \to \partial\Omega$ is also an open mapping, and $p_{\partial\Omega}(\mathcal{S})$ is a closed and open subset of $\partial\Omega$.

Because $\partial\Omega$ is connected, this ends the proof. □

8.1.4.4 Deduction of an isotropic size prescription

Theorem 8.2 lends itself to a heuristic for the definition of a suitable metric tensor field for the geometric approximation of the considered ideal surface Γ: suppose Γ is a smooth, compact manifold without boundary, which encloses a (smooth) bounded, connected domain Ω, so that $\Gamma = \partial\Omega$. We consider an approximating triangulation $\mathcal{S}$ of Γ that satisfies conditions (8.8), and demand it should be close to Γ up to a user-defined tolerance ε, that is:

$$d^H(\mathcal{S}, \Gamma) \leq \varepsilon.$$

In view of theorem 8.2, it is enough to ask that, for each triangle $T \in \mathcal{S}$, one has:

$$\frac{2}{9} \max_{x \in T} \max_{y,z \in T} |\mathcal{H}(d_\Omega)(x)|(yz, yz) \leq \varepsilon.$$

Now, let $T \in \mathcal{S}$ be a triangle which is 'very close' to a point $x \in \Gamma$, so that we can assume $|\mathcal{H}(d_\Omega)|$ is nearly constant around T. An approximation of the latter sufficient condition is then:

$$\forall y, z \in T, \ \frac{2}{9}|\mathcal{H}(d_\Omega)(x)|(yz, yz) \leq \varepsilon.$$

Remember that, from the material presented in Chapter 4, introducing $\kappa_i(x)$ $(i = 1, 2)$ the two principal curvatures of Γ at x (oriented in the sense that they are nonnegative at a point x around which Ω is locally convex), and $e_i(x)$ the two associated principal directions, the matrix $\mathcal{H}(d_\Omega)(x)$ writes in the orthonormal basis $(e_1(x), e_2(x), n(x))$ of $\mathbb{R}^d$:

$$\mathcal{H}(d_\Omega)(x) = \begin{pmatrix} \kappa_1(x) & 0 & 0 \\ 0 & \kappa_2(x) & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Hence, *formally speaking*, provided any segment yz included in T has length $|yz| \leq \sqrt{\frac{9\varepsilon}{2\max(|\kappa_1(x)|, |\kappa_2(x)|)}}$, i.e. provided the edges of T themselves comply with this bound, the contribution of T to the error in geometric approximation of $\mathcal{S}$ is no more than the desired tolerance ε.

From the heuristic considerations, in our attempt to remesh a triangulation $\mathcal{S}$ of Γ into a new triangulation which is close to Γ up to ε in terms of Hausdorff distance, we are led to remesh $\mathcal{S}$ with respect to the metric size map $h : \Gamma \to \mathbb{R}$ defined as:

$$\forall x \in \Gamma, \ h(x) = \min \left(h_{max} \, , \, \max \left(h_{min} \, , \, \sqrt{\frac{9\varepsilon}{2\max(|\kappa_1(x)|, |\kappa_2(x)|)}} \right) \right), \tag{8.10}$$

where h_{min} and h_{max} are respectively lower and upper bounds on the authorized lengths of edges in $\mathcal{S}$.

Remark 8.4. The size map h defined above only accounts for the local size feature associated to the geometric approximation of surface Γ. Yet, one could also seek to adapt the local size feature to another user-defined size map $m : \Gamma \to \mathbb{R}$, which could arise from an a posteriori error analysis associated to some numerical resolution of a problem (see for instance [4]). In such a case, this additional information is taken into account by trading h for a new size map $\widetilde{h}$ defined as:

$$\forall x \in \Gamma, \ \widetilde{h}(x) = \min(h(x), m(x)).$$

See section 8.1.6.3 for an example.

8.1.4.5 Gradation of the size map

Unfortunately, conforming to the size prescription discussed above is not sufficient in itself to guarantee the resulting mesh will enjoy a nice mesh quality. Indeed, the computed size map h may vary very sharply from one point to one of its neighbors, because the computation of h suffered from noise one the input data $\mathcal{S}$, or because the ideal surface Γ itself shows sharp variations of curvatures. This shock of size prescriptions between close areas may urge the formation of undesired ill-shaped elements during the remeshing process (see figure 8.12).

For this reason, it may be desirable to drive the remeshing operators in such a way that the resulting triangulation $\mathcal{S}$ from the process shows smooth variations in edge lengths. More accurately, we expect the

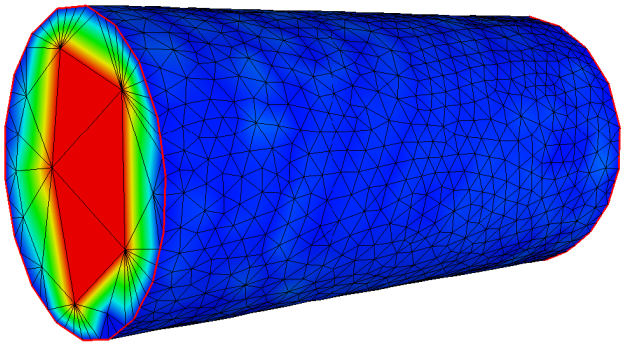


Figure 8.12: Sharp variations of the size map on a cylinder: planar areas may be prescribed an arbitrary large size (displayed in red), while curved areas are more constrained (blue).

resulting mesh to be such that two edges ap and bp sharing a common vertex p have *Euclidean* lengths satisfying:

$$\frac{1}{r} \leq \frac{|b-p|}{|a-p|} \leq r, \tag{8.11}$$

where r is a user-defined bound (typically, we use $r = 1.2$ or 1.3).

To achieve this, the seminal work [53] proposed to enforce a *gradation* on the size map h. More precisely, once the size map h has been computed (for instance - but not necessarily - on account of the size features presented in section 8.1.4), it is truncated in such a way that:

$$\forall x \in \Gamma, \ |\nabla_\Gamma h(x)| \leq h_{grad}, \tag{8.12}$$

where $\nabla_\Gamma h$ stands for the tangential gradient of h on Γ, and h_{grad} is the bound to be enforced on the variations of the size map. Equation (8.12) translates in the discrete framework in the following way:

$$\forall \text{ edge } pq \in \mathcal{S}, \ \frac{|h(p)-h(q)|}{|p-q|} \leq h_{grad}. \tag{8.13}$$

This criterion is imposed on the values of h stored at the vertices of $\mathcal{S}$ by traveling repeatedly the edges $pq \in \mathcal{S}$ and decreasing the largest of the two values $h(p)$ and $h(q)$ so that (8.13) holds.

So as to gain insight as regards the connection between this technical parameter h_{grad}, and the physically relevant one r, appearing in (8.11), let us look at the following idealized and formal situation, as in [201]. Consider a 'long' edge $pq \in \mathcal{S}$ with respect to h, associated to a curve γ on Γ, which is to be split within several sub-edges a_0a_1,..., $a_{n-1}a_n$, with $a_0 = p$ and $a_n = q$, in such a way that $\ell_h(a_ka_{k+1}) = 1$, for $k = 0, ..., n-1$ (see figure 8.13).

Denote as $\ell(\gamma)$ the Euclidean length of curve γ, and take a *normal* parametrization $[0, \ell(\gamma)] \ni s \mapsto \gamma(s)$ of γ, that is $\forall s \in [0, \ell(\gamma)]$, $|\gamma'(s)| = 1$. Denote s_k the value of parameter s at which $\gamma(s_k) = a_k$, $k = 0, ..., n$. By definition of the a_k, and because h varies linearly along γ, we have

$$\forall k = 0, ..., n, \ \ell_h(pa_k) = \int_0^{s_k} \frac{|\gamma'(s)|}{h(\gamma(s))} ds = \int_0^{s_k} \frac{ds}{h(p) + \frac{s}{\ell(pq)}(h(q)-h(p))} = k,$$

which brings, after some easy computation:

$$\forall k = 0, ..., n, \ s_k = \frac{h(p)}{h(q)-h(p)} (e^{k \frac{|h(p)-h(q)|}{\ell(pq)}} - 1).$$

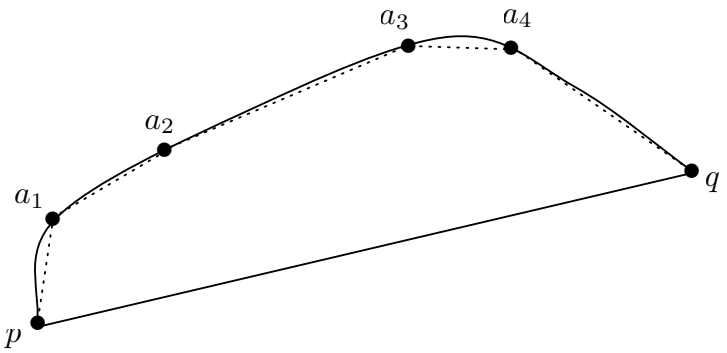


Figure 8.13: Splitting of edge pq within several sub-edges of unit length with respect to h.

Now, the gap between the lengths of two adjacent edges $a_{k-1}a_k$ and a_ka_{k+1} (that is, those of the underlying curves) reads:

$$\forall k = 1, ..., n-1, \ \frac{s_{k+1} - s_k}{s_k - s_{k-1}} = e^{\frac{|h(p)-h(q)|}{\ell(pq)}}.$$

Hence, so that (8.11) holds for some given value of r, the criterion (8.13) should be enforced with the value $h_{grad} = \ln(r)$.

8.1.5 The complete strategy

We have now all the tools we need to devise a strategy for remeshing a given triangulation $\mathcal{S}$ into a new one - that we should call $\widetilde{\mathcal{S}}$ - while retaining a control over the geometric approximation of their ideal surface Γ.

As an entry point, we are supplied with the initial triangulation $\mathcal{S}$, together with four user-defined parameters ε, h_{min}, h_{max}, and h_{grad}, the signification of which follows:

- As hinted at in Sections 8.1.2 and 8.1.4, parameter ε accounts for the accuracy of the geometric approximation of Γ, up to the formal assumptions we have been relying on from the very beginning: the resulting triangulation $\widetilde{\mathcal{S}}$ will stand at a Hausdorff distance $d^H(\widetilde{\mathcal{S}}, \Gamma) \leq \varepsilon$ from Γ,
- h_{min} and h_{max} are respectively the minimum and maximum authorized lengths for an edge of the resulting mesh. Note that imposing a minimum length for edges of $\widetilde{\mathcal{S}}$ may entail a violation of the geometric approximation criterion. In this case, we chose to bypass the imposed minimum length criterion,
- h_{grad} is the mesh gradation control parameter introduced in section 8.1.4.5.

We then divide the remeshing process into 5 steps:

step 1: *Analysis of $\mathcal{S}$.* Additional information about Γ are inferred from $\mathcal{S}$, along the lines of section 8.1.1: special geometric entities of Γ (ridge edges, singular points, etc...) are identified on $\mathcal{S}$, then normal vectors are approximated on $\mathcal{S}$ at regular points, as well as tangent vectors at ridge points,...

step 2: *Rough mesh modifications for a good 'sampling' of the surface.* This first real stage of mesh modifications aims at producing a new triangulation $\widetilde{\mathcal{S}_1}$ of Γ which is a nice geometric approximation, with respect to the prescribed tolerance: $d^H(\widetilde{\mathcal{S}_1}, \Gamma) \leq \varepsilon$. Starting with $\mathcal{S}$,

- edges pq are split if either they are longer (in terms of the Euclidean distance) than h_{max}, or the Hausdorff distance between them and the associated curves on Γ is higher than ε,
- edges pq are collapsed if they are shorter than h_{min}, provided the geometric approximation of Γ within tolerance ε is retained, and the resulting configuration is valid,
- edges are swapped if they end up with a *valid* configuration closer to Γ in terms of Hausdorff distance.

More accurately, we proceed within several iterations of the form:
(1) Identify all the edges that should be split, then split them (relying on patterns).
(2) Travel all the edges of the mesh, and collapse all the ones that should, and can be collapsed.
(3) Travel all the edges of the mesh, and swap all the ones that should, and can be swapped.

step 3: *Construction of the size map.* Although it may still be of poor quality, the new triangulation $\widetilde{\mathcal{S}_1}$ accounts for a suitable approximation of the geometry of Γ. It is then relevant to approximate higher-order geometric entities of Γ (such as curvatures) using $\widetilde{\mathcal{S}_1}$. A metric size map $h : \Gamma \to \mathbb{R}$ for the geometric approximation of Γ within tolerance ε in terms of Hausdorff distance is then computed as in section 8.1.4, and stored on $\widetilde{\mathcal{S}_1}$. This size map is then graded, as detailed in section 8.1.4.5.

step 4: *Rough mesh modifications with respect to the size map.* In this step, we proceed almost exactly as in step 2 so as to get another intermediate triangulation $\widetilde{\mathcal{S}_2}$, except on two points: first, we rely now on lengths measured with respect to h (see formula (8.6)). More specifically, aiming at getting a new triangulation whose edges have length 1 with respect to h (which is, of course, impossible), we choose *rough* bounds, outside which no edge length should lie: typically, we impose the triangulation $\widetilde{\mathcal{S}_2}$ should have no edge with length lying outside $[\ell_{r,min}, \ell_{r,max}]$, with $\ell_{r,min} = 0.3$, $\ell_{r,max} = 2$. Second, we take much more caution about mesh quality as in step 2: edge collapses are presently prevented when they degrade two much the overall quality of the triangulation, and edge swaps are now carried out provided they help improving the quality of the mesh (and possibly rejected if they degrade two much the geometric approximation of surface Γ).

step 5: *Fine mesh modifications with respect to the size map.* Triangulation $\widetilde{\mathcal{S}_2}$ should now be 'pretty good' in terms of geometric approximation of Γ and of mesh quality, and in this last stage, we perform delicately driven operations so as to get the final triangulation $\widetilde{\mathcal{S}}$. Lengths of edges are still evaluated with respect to h, except we now impose $\widetilde{\mathcal{S}}$ have no edge with length lying outside a sharper interval as before, of the form $[\ell_{f,min}, \ell_{f,max}]$, with for instance $\ell_{f,min} = 0.7$, $\ell_{f,max} = 1.3$. We are also much stricter as far as the authorized degradation in mesh quality entailed by our operators is concerned. Another important improvement with respect to steps 2 and 4 is that we now add the vertex relocation operator to our toolbox. The iterations performed during steps 2 and 4 evolve into:
(1) Travel all the edges of the mesh, and split (now in a one-by-one fashion) all the ones that should, and can be split.
(2) Travel all the edges of the mesh, and collapse all the ones that should, and can be collapsed.
(3) Travel all the edges of the mesh, and swap all the ones that should, and can be swapped.
(4) Travel all the vertices of the mesh, and relocate all the ones that should, and can be relocated.

8.1.6 Numerical examples

We end up this section devoted to surface remeshing by showing several numerical applications so as to emphasize the different features of the proposed method. For each of these examples, the following details are reported in Table 8.1: the initial (resp. final) number of points in the mesh np_i (resp. np_f), the total CPU time (in seconds) of the computation, the initial (resp. final) mean quality Q_i (resp. Q_f) of triangles of the mesh, and the worst quality wq_i (resp. wq_f) of an element of the initial (resp. final) triangulated surface.

8.1.6.1 Curvature-dependent surface remeshing of smooth surfaces

Using the proposed strategy, a first idea that comes to mind is to remesh discrete triangulated surfaces which account for a smooth underlying geometry, for instance surfaces arising from data scanning [102, 310].

Figure 8.14 [2] shows an example of the impact of the control parameter ε over the geometric degradation of the ideal surface Γ: an oversampled initial triangulation $\mathcal{S}$ - is remeshed into several new triangulations

2. Data courtesy of `http://cyberware.com/`.

using the same values for parameters h_{min}, h_{max} and h_{grad}, but with different values of ε.

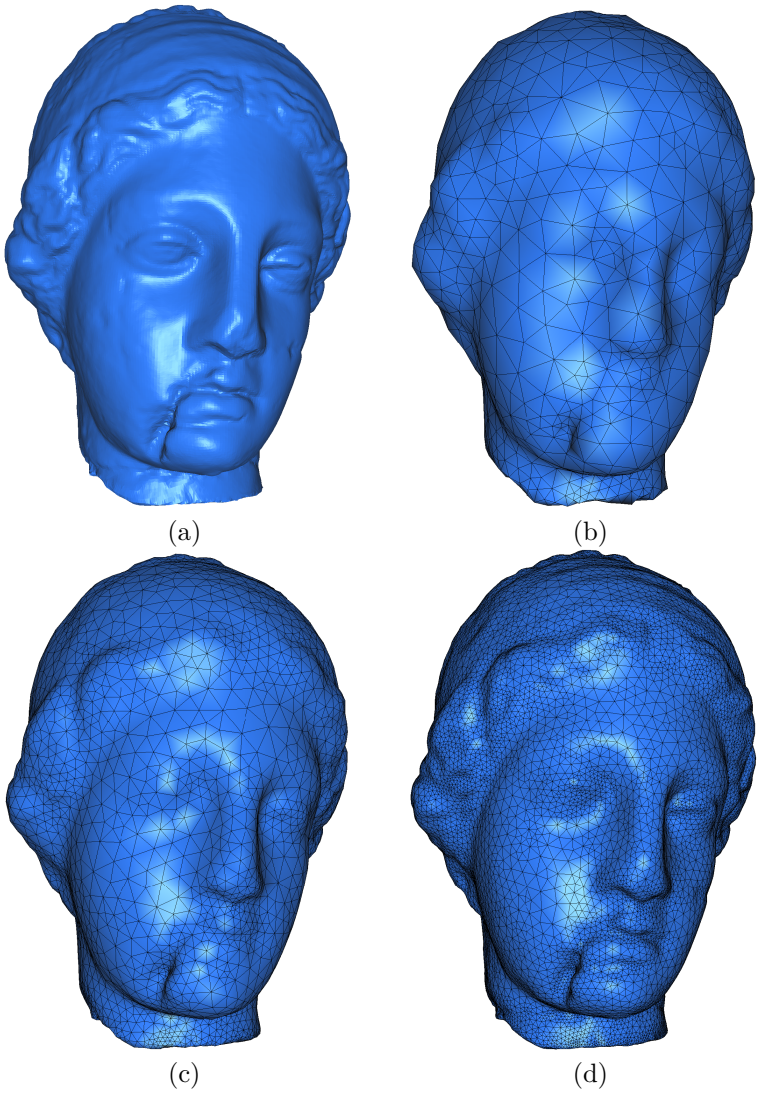


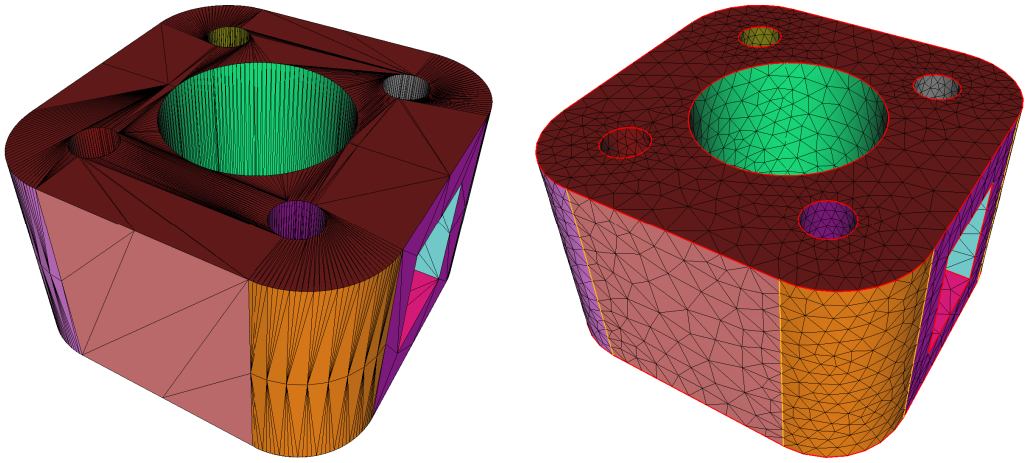
Figure 8.14: Remeshing of the *igea* model, enclosed in a bounding box of dimensions $0.06911 \times 0.09908 \times 0.09934$. Here we take $h_{min} = 0.0001$, $h_{max} = 0.05$, and $h_{grad} = 1.2$; *(a)* initial triangulation; *(b)* remeshed triangulation, using $\varepsilon = 0.005$; *(c)* remeshed triangulation, using $\varepsilon = 0.001$ *(d)* remeshed triangulation, using $\varepsilon = 0.0002$.

8.1.6.2 High-quality surface remeshing of mechanical parts

Let us now turn to a slightly different application of the proposed remeshing procedure, namely the remeshing of mechanical parts. Such triangulated surfaces generally differ from those considered in subsection 8.1.6.2 insofar as they may present sharp features (ridges, corners,...) that must be identified before any treatment is possible. These sharp features account for discontinuities of the geometric behavior of the surface along some curves, and may also separate areas holding different references (corresponding to boundary conditions, material properties, etc...). Last but not least, the meshes associated to such surfaces generally

stem from a CAD modeler (i.e. they have been obtained from so-called *STL* files); as such, they account for a close and minimalistic approximation of the underlying ideal surface, while undergoing a *very* poor quality as far as finite element computations are concerned.

Figure 8.15 shows a remeshing example of such a triangulated surface. Note the strong impact of the gradation imposed on the size map on the quality of the triangulation of the plane parts of the resulting surface.



(a) (b)

Figure 8.15: Remeshing of the *model02* model, enclosed in a bounding box of dimensions $0.2 \times 0.2 \times 0.12$. Here we take $h_{min} = 0.001$, $h_{max} = 1$, $h_{grad} = 1.2$ and $\varepsilon = 0.001$; *(a)* initial triangulation; *(b)* resulting triangulation.

8.1.6.3 Remeshing of a surface with respect to an externally prescribed size map

We then turn to an example where a local size prescription $m : \Gamma \to \mathbb{R}$ is provided by the user. The size map used for remeshing $\mathcal{S}$ is computed as mentioned in remark 8.4. As explained then, the case we have in mind is when m arises from an a posteriori error analysis of a numerical method performed on $\mathcal{S}$. However, for the sake of simplicity, we limit ourselves to the case when m is associated to the interpolation error of a smooth function, which is much simpler as regards the analysis, yet identical as far as the remeshing strategy is concerned.

More precisely, let the ideal surface Γ be the ball of centre 0 and radius 4, and let $f : \Gamma \to \mathbb{R}$ be a function of class $\mathcal{C}^2$, which presents 'sharp variations' on Γ. For instance,

$$\forall p = (x, y, z) \in \Gamma, \quad f(p) = \tanh((x + 1.3)^{20}\,(y - 0.3)^{10}\,z). \tag{8.14}$$

We aim at modifying the initial triangulation $\mathcal{S}$ so that interpolating (linearly) f over the resulting mesh entails an error (in L^∞-norm) controlled by a parameter ε_m. To this end, we proceed as in section 8.1.4, relying on theorem 8.1 to cook an associated size map $m : \Gamma \to \mathbb{R}$ to such a control.

Actually, such an *adaptation procedure* to some externally-given indicator should be performed several times, before all the sharp variations of f could be captured: indeed, if the initial mesh is too coarse, the first remeshing procedure is bound not to detect accurately all the areas concerned with these sharp variations.

Figure 8.16 displays the resulting triangulation after 5 remeshing procedures for adapting a mesh of Γ to the interpolation of f - starting from a rather coarse triangulation $\mathcal{S}$, and using parameter $\varepsilon_m = 0.01$. The parameters chosen for remeshing are (for each one of the 5 steps): $h_{min} = 0.04$, $h_{max} = 0.5$, $h_{grad} = 1.2$, and $\varepsilon = 0.04$. The whole computation took $22.9s$, and the final mesh enjoys 28310 vertices, for an average quality of 0.970.

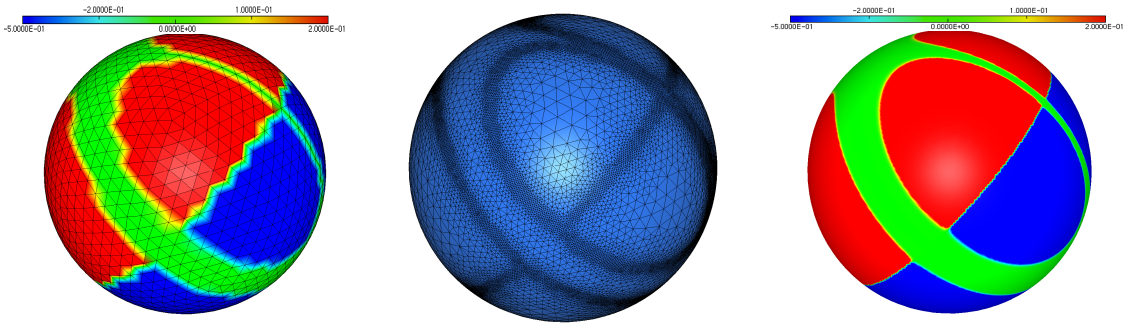


Figure 8.16: *(left)* Initial triangulation $\mathcal{S}$, with the isolines of the linear interpolate of f over $\mathcal{S}$; *(middle)* resulting triangulation after 5 remeshing procedures; *(right)* isolines of the linear interpolate of f over the resulting triangulation.

8.1.6.4 Some miscellaneous surfaces remeshing

Figures 8.17 [3] and 8.18 [4] present other examples of remeshing of surface triangulations.

Test case	np_i	np_f	CPU (s)	Q_i	wq_i	Q_f	wq_f
igea (b)	134345	1319	4.118	0.8261	0.027	0.9438	0.58
igea (c)	134345	4320	4.919	0.8261	0.027	0.9510	0.64
igea (d)	134345	16935	7.434	0.8261	0.027	0.9519	0.41
model02	1204	3244	0.683	0.0768	$8e^{-6}$	0.94228	0.51
buddha	25003	65895	6.786	0.7313	0.033	0.9648	0.16
drwL	1039	18938	2.261	0.52054	0.010	0.92484	0.015
wheel	9966	17989	2.369	0.3328	$2e^{-6}$	0.9285	0.196
Pegasus	125002	21703	7.799	0.758	0.003	0.950	0.194

Table 8.1: Details on the examples of section 8.1.6.

3. Data courtesy of `http://www.archive3d.net` and `http://www-roc.inria.fr/gamma/download/download.php`.
4. Data courtesy of `https://grabcad.com` and `http://www-roc.inria.fr/gamma/download/download.php`.



Figure 8.17: *(a-b)* Remeshing of the *buddha* model, enclosed in a bounding box of dimensions $99.3304 \times 141.9974 \times 93.2093$. Here we take $h_{min} = 0.5$, $h_{max} = 10$, $h_{grad} = 1.2$, and $\varepsilon = 0.2$; *(a)* initial triangulation; *(b)* remeshed triangulation. *(c-d)* Remeshing of the *drwL* model, enclosed in a bounding box of dimensions $141.9963 \times 27.0421 \times 74.2486$, with $h_{min} = 0.1$, $h_{max} = 5$, $h_{grad} = 1.2$, and $\varepsilon = 0.05$; *(c)* initial triangulation; *(d)* remeshed triangulation.

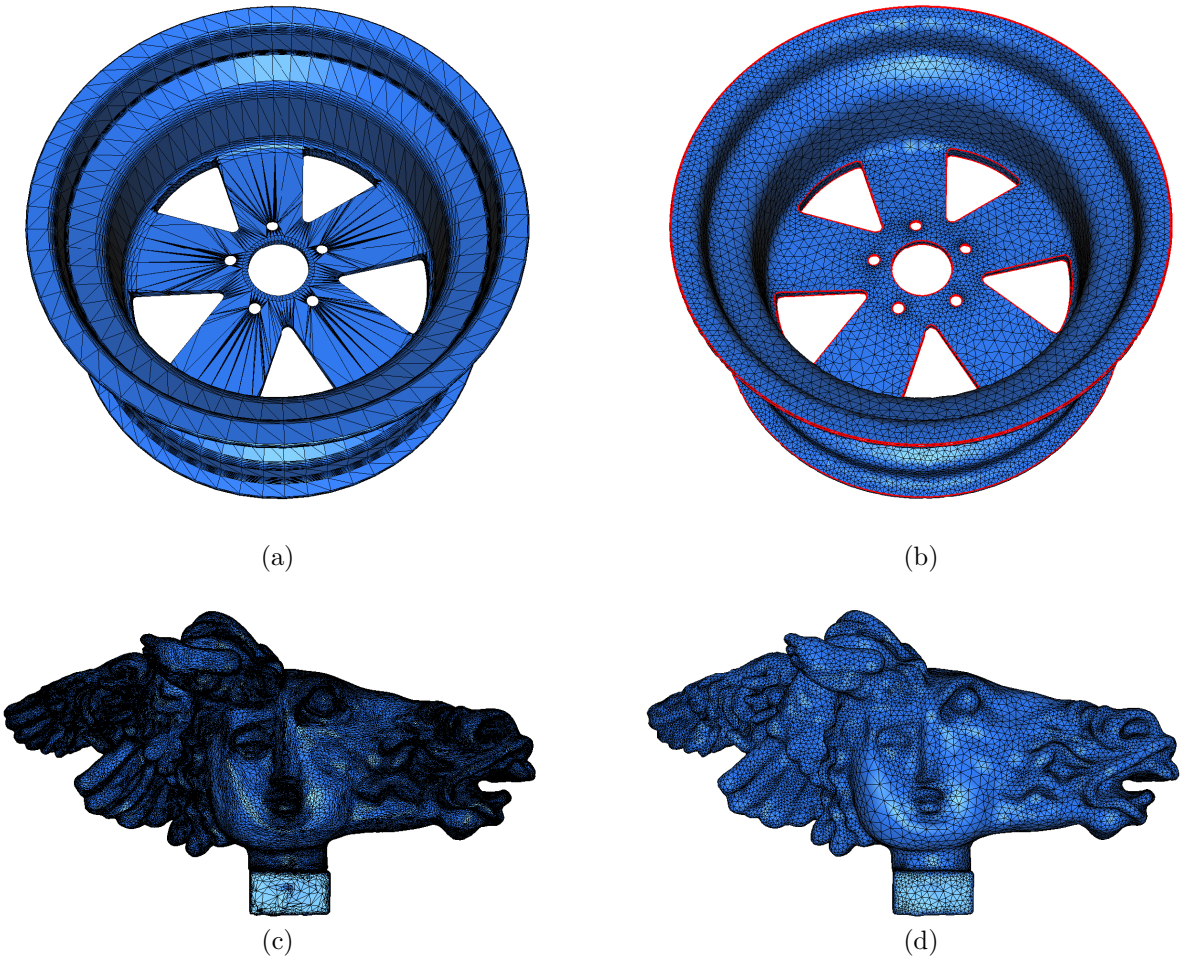


Figure 8.18: *(a-b)* Remeshing of the *wheel* model, enclosed in a bounding box of dimensions $381.882 \times 182.4686 \times 381.882$. Here we take $h_{min} = 0.5$, $h_{max} = 100$, $h_{grad} = 1.2$, and $\varepsilon = 2$; *(a)* initial triangulation; *(b)* remeshed triangulation. *(c-d)* Remeshing of the *Pegasus* model, enclosed in a bounding box of dimensions $875.392 \times 482.3467 \times 372.698$. Here we take $h_{min} = 1$, $h_{max} = 100$, $h_{grad} = 1.2$, and $\varepsilon = 2$; *(c)* initial triangulation; *(d)* remeshed triangulation.

8.2 Discrete surface remeshing in the anisotropic context

Up to this point, the metric tensor field defined at subsection 8.1.4.4 has been merely an artificial tool to drive the local operators described in section 8.1.3. We propose in this section to expand the metaphor of meshing with respect to a metric tensor field to devise a method for anisotropic surface remeshing.

More specifically, instead of generating a scalar local size prescription based on the geometry of the ideal surface Γ, anisotropic remeshing relies on the definition of a bilinear symmetric positive definite form over each tangent plane $T_p\Gamma$, $p \in \Gamma$, which encompasses the various desired size in the *tangent* directions to Γ, which are the only intrinsic directions when looking at this surface (the other ones depending on the choice of an embedding of Γ into some Euclidean space). The problem of anisotropic surface remeshing can thus be cast into the framework of *Riemannian geometry*.

As we shall see soon, the framework of Section 8.1 about isotropic surface remeshing can be used to a large extent in the device of an anisotropic surface remeshing algorithm. Hence, in this section, we mainly put the stress on the descriptions of the specific ingredients to the anisotropic case. This work about anisotropic surface remeshing is organized as follows: after recalling several notions from Riemannian geometry in the next section 8.2.1, we present in Section 8.2.2 a general construction on a Riemannian manifold for approximating parallel transport. Both sections are mostly theoretical, and find an echo in the numerical practice in Sections 8.2.3, 8.2.4, and 8.2.5. Eventually, some examples are presented in Section 8.2.6.

8.2.1 A wee bit of Riemannian geometry

The purpose of this section is to collect some classical definitions and notations about Riemannian manifolds that will play a key role in the sequel. Of course, this short reminder is not at all meant to be exhaustive, nor absolutely rigorous, and full details can be found in [75, 116, 198].

For the sake of simplicity, this presentation is restricted to the smooth context: all the considered manifolds, curves or sections of vector bundles at hand will enjoy $\mathcal{C}^\infty$ regularity, unless otherwise stated.

8.2.1.1 Notations

Let M be a smooth differentiable manifold.

- For any vector bundle E over M, we denote as $\Gamma(E)$ the set of smooth sections of E.
- We denote as TM the *tangent bundle* to M. For any point $x \in M$, we denote as T_xM the fiber of TM at x, in other words the *tangent space to M at x*.
- For $k \in \mathbb{N}$, we denote as $\bigotimes^k(TM^*)$ the vector bundle of k-linear forms over TM, as $\Lambda^k(TM^*)$ the vector bundle of the (alternating) differential k-forms over TM, and as $\mathrm{Sym}^2(TM^*)$ the vector bundle of symmetric bilinear forms over TM.

A *Riemannian metric* (or *Riemannian structure*) on M is the choice of a scalar product $g_x : T_xM \times T_xM \to \mathbb{R}$ over each tangent plane T_xM, $x \in M$, that varies *smoothly* with respect to x (i.e. whose coefficients when pulled-back in a local trivialization of the tangent bundle TM vary smoothly). For any $x \in M$, g induces a natural norm $||.||_x$ on the vector space T_xM.

Throughout this preliminary section 8.2.1, we consider a smooth oriented Riemannian manifold (M, g) of dimension n. As the only Riemannian structure we will be considering over M is the one associated to the metric g, we will not mention explicitly the dependance on g of the forthcoming concepts.

8.2.1.2 Riemannian distance and volume

The following definition provides a natural way of measuring distances on M:

Definition 8.2. – *Let $I = [a, b]$ a closed interval of $\mathbb{R}$, and $I \ni t \mapsto \gamma(t) \in M$ a piecewise differentiable curve on M. The length $\ell(\gamma)$ of γ is:*

$$\ell(\gamma) = \int_a^b \sqrt{g_{\gamma(t)}(\gamma'(t), \gamma'(t))} dt. \tag{8.15}$$

– *Let $x, y \in M$. The distance $d(x, y)$ between x and y is:*

$$d(x, y) = \inf \{\ell(\gamma) |\ \gamma : [a, b] \to M \text{ piecewise differentiable }, \ \gamma(a) = x \ ; \ \gamma(b) = y\}. \tag{8.16}$$

We shall also need a means of measuring volumes on M, which leads us to the notion of *volume form* associated to an oriented Riemannian manifold of dimension n:

Proposition and Definition 8.3. There exists a unique differential n-form $\omega \in \Gamma(\Lambda^n(TM^*))$ over M such that for any point $x \in M$, and any oriented orthonormal basis of tangent vectors $(X_1(x), ..., X_n(x))$ of T_xM, one has:

$$\omega_x(X_1(x), ..., X_n(x)) = 1.$$

This form is called the *volume form* of (M, g).

If $\varphi : U \subset \mathbb{R}^n \to \varphi(U)$ is any local oriented chart around a point $x_0 \in M$, ω reads in this chart:

$$\forall x \in U, \ \ \varphi^*(\omega)_x = \sqrt{\det(g_{i,j}(x))} dx_1 \wedge ... \wedge dx_n, \quad g_{i,j}(x) = g_{\varphi(x)}(d\varphi_x(e_i), d\varphi_x(e_j)), \tag{8.17}$$

where $(e_i)_{i=1,...,n}$ is the canonical basis of $\mathbb{R}^n$ and $dx_1 \wedge ... \wedge dx_n$ is the volume form of $\mathbb{R}^n$ associated to the Euclidean metric.

Now, the *volume* of any compact subset K of M is defined as:

$$\text{vol}(K) = \int_K \omega. \tag{8.18}$$

8.2.1.3 Connections and parallel transport of vector fields on a Riemannian manifold

In the following sections, we will be especially interested in the evolution of quantities (e.g. functions, vector fields) along curves, and in this context, an intrinsic notion of *derivation with respect to a vector field* of such quantities would be appreciated: this is the concept of *connection*, which we now recall.

Let $X \in \Gamma(TM)$ a vector field on M, $f : M \to \mathbb{R}$ a smooth function. The *derivative $\partial_X f(x)$ of f in the direction of X at $x \in M$* is classically defined as:

$$\partial_X f(x) = df_x(X(x)) = \lim_{t \to 0} \frac{f(\gamma(t)) - f(x)}{t}, \tag{8.19}$$

for any differentiable curve $\gamma : [-\varepsilon, \varepsilon] \to M$, with $\gamma(0) = x$, $\gamma'(0) = X(x)$.

Similarly, given $X, Y \in \Gamma(TM)$ two vector fields, one would like to define a notion of *derivative of Y in the direction of X at $x \in M$* mimicking formula (8.19), say:

$$\text{`` } \lim_{t \to 0} \frac{Y(\gamma(t)) - Y(x)}{t}, \tag{8.20}$$

for any differentiable curve $\gamma : [-\varepsilon, \varepsilon] \to M$, with $\gamma(0) = x$, $\gamma'(0) = X(x)$. Unfortunately, this does not make sense, for at least two reasons. First, the subtraction in formula (8.20) has no precise meaning inasmuch as $Y(\gamma(t))$ and $Y(x)$ do not belong to a common vector space. Second, even if they did - for instance, if M is embedded in $\mathbb{R}^d$, and all the tangent planes to M are identified to subspaces of $\mathbb{R}^d$ - there would be no

reason for the limit in (8.20) to belong to T_xM, which is what we would expect of an intrinsic derivation rule over M - that is, a rule which does not depend on any embedding of M into a larger space.

A rigorous definition of such a notion consists in devising a rule for identifying any two tangent planes T_pM, T_qM, $p, q \in M$, then rely on formula (8.20) after identifying $Y(\gamma(t))$ as an element of T_xM. As an analogy with the case when M is a submanifold of $\mathbb{R}^d$, such an identification rule generally takes the name of *parallel transport*.

The classical (and mathematically more convenient) point of view is the exact converse to this intuitive idea: starting from a derivation rule over M which satisfies the expected properties, one then deduces an associated *parallel transport rule* over M.

Definition 8.4. *A connection over M is a mapping $\nabla : \Gamma(TM) \times \Gamma(TM) \to \Gamma(TM)$ such that:*

(i) for all $\lambda, \mu \in \mathbb{R}$, and $X, Y, Z \in \Gamma(TM)$, $\nabla_X(\lambda Y + \mu Z) = \lambda \nabla_X Y + \mu \nabla_X Z$.

(ii) for any smooth function $f : M \to \mathbb{R}$, and $X, Y \in \Gamma(TM)$, $\nabla_X(fY) = (\partial_X f)Y + f\nabla_X Y$.

(iii) for all smooth functions $f, g : M \to \mathbb{R}$, and $X, Y, Z \in \Gamma(TM)$, $\nabla_{(fX+gY)}Z = f\nabla_X Z + g\nabla_Y Z$.

A connection ∇ on M naturally induces a notion of *derivative of a vector field along a curve*.

Proposition and Definition 8.5. Suppose M is equipped with a connection ∇, and let $\gamma : I \to M$ a differentiable curve. There exists a unique operator $\frac{D}{dt}$ which associates to any vector field V defined along γ another vector field $\frac{DV}{dt}$ along γ in such a way as:

1. For any two vector fields V, W along γ, one has $\frac{D}{dt}(V + W) = \frac{DV}{dt} + \frac{DW}{dt}$.
2. For any vector field V along γ, and any differentiable function $f : I \to \mathbb{R}$, one has $\frac{D}{dt}(fV) = \frac{df}{dt}V + f\frac{DV}{dt}$.
3. If a vector field V along γ is the restriction of $X \in \Gamma(TM)$, i.e. $\forall t \in I$, $V(t) = X(\gamma(t))$, then: $\frac{DV}{dt} = \nabla_{\gamma'(t)}X$.

Definition 8.6. *Let M a differentiable manifold, equipped with a connection ∇. A vector field V along a differentiable curve $\gamma : I \to M$ is called parallel if $\frac{DV}{dt} = 0$ on I.*

The link between the notion of connection, and that of parallel transport is achieved thanks to the following definition.

Proposition and Definition 8.7. Suppose M is equipped with a connection ∇, and $\gamma : I \to M$ a differentiable curve in M.

- Let $t_0 \in I$, and $V_0 \in T_{\gamma(t_0)}M$. There exists a unique parallel vector field along γ such that $V(t_0) = V_0$, which is called the *parallel transport* of V_0 along γ.
- Let $t_0, t_1 \in I$, and $V_0 \in T_{\gamma(t_0)}M$. One defines the *parallel transport $T(\gamma)_{t_0}^{t_1}(V_0)$ of V_0 from $T_{\gamma(t_0)}M$ to $T_{\gamma(t_1)}M$ along γ* as $T(\gamma)_{t_0}^{t_1}(V_0) = V(t_1)$, where V is the unique parallel vector field along γ such that $V(t_0) = V_0$. $T(\gamma)_{t_0}^{t_1} : T_{\gamma(t_0)}M \to T_{\gamma(t_1)}M$ defines a linear isomorphism, with inverse $T(\gamma)_{t_1}^{t_0}$.
- Let V a vector field along γ. For any $t_0 \in I$, one has

$$\frac{DV}{dt}(t_0) = \lim_{t \to t_0} \frac{T(\gamma)_t^{t_0}(V(t)) - V(t_0)}{t - t_0}. \tag{8.21}$$

Note that this concept of parallel transport along a differentiable curve $\gamma : I \to M$ is easily generalized to the case of curves that are only piecewise differentiable. This will come in handy in section 8.2.1.5.

Some remarks are in order. First and foremost, let us notice that a lot of connections may exist on a given manifold M, each one accounting for its own rule of parallel transport - that is, its own means of identifying different tangent planes. One would like to select a derivation rule which enjoys intuitive geometric properties in terms of the associated parallel transport. Second, up to this point, we have never used the Riemannian structure of M: a nice way to define an intuitive derivation rule would be, for instance, to make the associated parallel transport rule compatible with lengths measurements. This is the purpose of the following definition.

Definition 8.8.

– *A connection ∇ on M is said to be compatible with metric g if:*

$$\forall X, Y, Z \in \Gamma(TM),\ \partial_X(g(Y,Z)) = g(\nabla_X Y, Z) + g(Y, \nabla_X Z). \tag{8.22}$$

– *A connection ∇ on M is said torsion-free provided:*

$$\forall X, Y \in \Gamma(TM),\ \nabla_X Y - \nabla_Y X = [X, Y], \tag{8.23}$$

where $[X, Y]$ denotes the Lie bracket between vector fields X, Y.

Remark 8.5. The *compatibility* between a connection ∇ and the metric g implies in particular (and actually turns out to be equivalent to) the fact that, for any differentiable curve $\gamma : I \to M$, and any two parallel vector fields V, W along γ, the scalar product $g(V, W)$ is constant as a function on I. Thus, the *metric compatibility condition* of a connection actually means the associated parallel transport rule preserves lengths of tangent vectors, and angles between them.

The torsion-free property is slightly more subtle. Recall that $[X, Y] \in \Gamma(TM)$ is an indicator of the lack of commutation of the flows associated to X and Y (see Figure 8.19): more precisely, let $t > 0$ a small time, and denote as ϕ_t (resp ψ_t, ξ_t) the flow operator associated to X (resp. Y, $[X, Y]$). Then one has:

$$\psi_{-t} \circ \phi_{-t} \circ \psi_t \circ \phi_t = \xi_{t^2} + o(t^2). \tag{8.24}$$

Note that this quantity is utterly intrinsic to the geometry of M. On the other hand, $\nabla_X Y - \nabla_Y X$ is a measure of the lack of commutation of the notion of parallel transport induced by ∇ depending on whether Y is transported in the direction of X or the converse. It is then quite natural to ask this lack of commutation may be caused only by the inherent lack of commutation of X and Y due to the geometry of M, and that ∇ induces no extra 'torsion' in the respective displacement of vector fields.

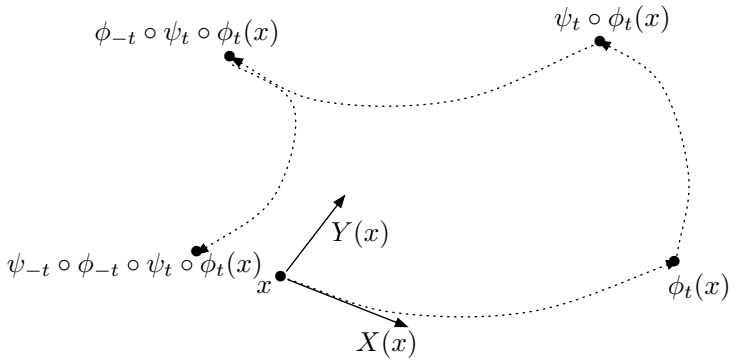


Figure 8.19: Lack of commutation of the flows associated to X and Y, starting from $x \in M$.

Using these two suitable properties for a connection allows to define a notion of 'natural' connection on (M, g):

Theorem 8.3. *There exists a unique connection ∇ on M which is at the same time compatible with g and torsion-free. It is called the Levi-Civita connection on M.*

8.2.1.4 Connections and parallel transport of higher-dimensional objects on M

Hitherto, we have only been investigating parallel transport of tangent vectors along vector fields. However, it may be relevant to compare other kinds of objects defined at different points of M and, to this end, to

transport such quantities along curves. In this section, we focus on describing parallel transport of bilinear forms over the tangent planes to M.

Let ∇ be a connection on M. Recall from proposition 8.5 that ∇ induces a notion of derivative along curves drawn on M.

Definition 8.9. *Let $\gamma : I \to M$ be a parametrized curve, b a field of bilinear forms defined along γ - that is, for all $t \in I$, $b_t : T_{\gamma(t)}M \times T_{\gamma(t)}M \to \mathbb{R}$ is a bilinear form. The derivative $\frac{Db}{dt}$ of b along γ is the field of bilinear forms along γ defined as, for every vector fields V, W along γ,*

$$\forall t \in I,\ \frac{Db_t}{dt}(V(t), W(t)) = \frac{d}{dt}\left(b_t(V(t), W(t))\right) - b_t\left(\frac{DV}{dt}(t), W(t)\right) - b_t\left(V(t), \frac{DW}{dt}(t)\right). \tag{8.25}$$

Besides, b is said parallel along γ if $\frac{Db}{dt} = 0$ on I.

Remarks 8.6.

- Definition 8.9 has been cooked in such a way as, for any two parallel vector fields V, W, any parallel field of bilinear forms b along γ, the quantity $b_t(V(t), W(t))$ is constant on I.
- In view of definition 8.9, if ∇ is compatible with metric g, then g is a parallel along any curve on M.
- This notion of derivative of fields of bilinear forms along curves is actually rigorously expressed introducing a notion of connection over the vector bundle of bilinear forms: this requires the larger framework of *Koszul connections*, in which connections, parallel transport, are defined for sections of an arbitrary given vector bundle. As we will not require so much generality in the sequel, we limit ourselves with this sole definition.
- This development gives rise to a notion of parallel transport for fields of bilinear forms exactly as in the case of vector fields. In the context of definition 8.9, if $t_0 \in I$, and b is a bilinear form over $T_{\gamma(t_0)}M$, one defines the *parallel transport $T(\gamma)^t_{t_0}(b)$ of b along γ at time $t \in I$* as b_t, where b_t stands for the unique parallel field of bilinear forms along γ such that $b_{t_0} = b$.

8.2.1.5 Geodesics and the exponential map

One of the more crucial concepts in Riemannian geometry - as well from the theoretical point of view as for applications (e.g. in geography, general relativity) - is that of *shortest path* between two points $p, q \in M$, i.e. minimizers of the Riemannian distance (8.16). Unfortunately, this notion is very difficult to study for it involves the global geometry of M. A more convenient point of view turns out to be that of *local minimizers* of the Riemannian distance, namely *geodesic curves*.

Roughly speaking, geodesic curves are defined as curves along which no acceleration is felt, i.e. the velocity vector bends as little as possible along the trajectory. This notion is naturally dependent on the choice of a parallel transport rule on M, and in all this section, M will be endowed with its Levi-Civita connection ∇.

Definition 8.10. *A smooth parametrized curve $\gamma : I \mapsto M$ is a geodesic if*

$$\forall t \in I,\ \frac{D}{dt}(\gamma'(t)) = 0. \tag{8.26}$$

If $[a, b] \subset I$ and $\gamma : I \mapsto M$ is a geodesic, the restriction $\gamma|_{[a,b]}$ is called a geodesic segment.

Remarks 8.7.

- The meaning of (8.26) is that γ shows no acceleration along the tangent planes to M. In other words, the motion is completely dictated by the bending of M itself.
- As an immediate consequence of the definition and the compatibility with the metric (8.22) of ∇, a geodesic curve $\gamma : I \to M$ enjoys *constant speed*:

$$\forall t \in I,\ \frac{d}{dt}\left(g_{\gamma(t)}(\gamma'(t), \gamma'(t))\right) = 0. \tag{8.27}$$

The following fundamental result assesses that geodesic curves locally exist, and are actually characterized by their starting point and their initial velocity vector. Furthermore, the dependence of a geodesic curve on these data is uniform.

Theorem 8.4. *Let $p \in M$,*

- *For any $v \in T_pM$, there exists $\delta > 0$, and a geodesic curve $\gamma : (-\delta, \delta) \to M$ such that $\gamma(0) = p$ and $\gamma'(0) = v$. Moreover, such a geodesic is unique in the following sense: if $\gamma_1 : I_1 \to M$ and $\gamma_2 : I_2 \to M$ are two geodesic curves, defined over intervals $I_1, I_2 \subset \mathbb{R}$ containing 0, such that $\gamma_1(0) = \gamma_2(0) = p$, and $\gamma_1'(0) = \gamma_2'(0) = v$, then $\gamma_1(t) = \gamma_2(t)$ for all $t \in I_1 \cap I_2$. This allows to define, for $v \in T_pM$ the maximal geodesic $\gamma(., p, v) : I \to M$ enjoying this property.*
- *There exists a neighborhood V of p in M, and $\varepsilon > 0$, such that the mapping $(-2, 2) \times \mathcal{U} \ni (t, x, v) \mapsto \gamma(t, x, v) \in M$, where $\mathcal{U} = \{(x, v) \in TM;\ x \in V,\ v \in T_xM \text{ s. t. } ||v||_x < \varepsilon\}$ is well-defined and has $\mathcal{C}^\infty$ regularity.*

This theorem in particular allows to define the *exponential map* of a Riemannian manifold, which is grossly speaking a very convenient way to 'unfold' a tangent plane to M into a 'spray of geodesics'.

Proposition and Definition 8.11. Let $p \in M$.

- The *exponential application* $exp_p : B_\varepsilon(0) \subset T_pM \to M$ is defined as, for $v \in T_pM$, $||v||_p < \varepsilon$, $exp_p(v) = \gamma(1, p, v)$, where ε and γ arise as in theorem 8.4.
- Up to restricting ε, there exists a neighborhood V of p in M such that $exp_p : B_\varepsilon(0) \subset T_pM \to M$ is a smooth diffeomorphism. $V = exp_p(B_\varepsilon(0))$ is called the *geodesic ball of center p and radius ε*, and is denoted as $B_\varepsilon(p)$ when no confusion is possible with its Euclidean equivalent. The inverse map $log_p : V \to B_\varepsilon(0)$ is the *logarithm application*.

Relying on these tools, one can define a generalization of the notion of *convexity* to the case of a general Riemannian manifold.

Definition 8.12. *A subset $S \subset M$ is said strongly convex if for any two points $p, q \in \overline{S}$, there exists a unique geodesic $\gamma : I \to \mathbb{R}$ connecting p to q in such a way as $\gamma(I) \setminus \{p, q\}$ is contained in S.*

Theorem 8.5. *For any point $p \in M$, there exists $r > 0$ such that the geodesic ball of center p and radius r is strongly convex.*

It is now high time to relate geodesic curves with shortest paths on M. The first observation is that minimizers of the Riemannian distance actually turn out to be geodesic segments.

Theorem 8.6. *Let $p, q \in M$, and $\gamma : [a, b] \to M$ a piecewise differentiable curve whose parametrization is proportional to arc length. If, for any piecewise differentiable curve $c : [a, b] \to M$ such that $c(a) = \gamma(a)$ and $c(b) = \gamma(b)$ one has $\ell(\gamma) \leq \ell(c)$, then γ is smooth and is a geodesic on M.*

The converse is obviously false. For instance, let $M = \mathbb{S}^2 \subset \mathbb{R}^3$ be the two-dimensional sphere, endowed with the metric induced by the Euclidean metric of $\mathbb{R}^3$, $p = (0, 0, 1) \in \mathbb{S}^2$, and consider the great circle $\gamma : \mathbb{R} \ni t \mapsto (\sin(t), 0, \cos(t)) \in \mathbb{S}^2$. A straightforward computation shows that γ is a geodesic curve of $\mathbb{S}^2$ with $\gamma(0) = p$. However, as soon as $t_0 > \pi$, the geodesic segment $\gamma([0, t_0])$ is no longer a minimizer of the distance between p and $\gamma(t_0)$ (see figure 8.20).

However, as made rigorous by the following result, geodesic curves arise as *local* minimizers of the Riemannian distance.

Theorem 8.7. *Let $p \in M$; there exists $\varepsilon > 0$ such that every geodesic segment γ from $[0, 1]$ into the geodesic ball $B_\varepsilon(p)$ and starting with $\gamma(0) = p$ is a minimizer of the Riemannian distance (8.16) between p and its endpoint $\gamma(1)$, that is: for any piecewise differentiable curve $c : [0, 1] \to B_\varepsilon(p)$ with $c(0) = \gamma(0) = p$ and $c(1) = \gamma(1)$, one has: $\ell(\gamma) \leq \ell(c)$, and equality holds if an only if $\gamma([0, 1]) = c([0, 1])$.*

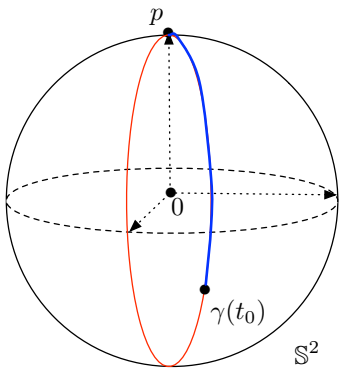


Figure 8.20: The geodesic segment $\gamma : [0, t_0] \to \mathbb{S}^2$ (in red) is a minimizer of the Riemannian distance between p and $\gamma(t_0)$ if and only if $t_0 \leq \pi$ (otherwise, the parametrized curve represented in blue is).

8.2.1.6 Expressions in local coordinates

By definition, any smooth n-dimensional manifold locally 'looks like' an open subset of $\mathbb{R}^n$. This allows to define *local coordinates* on M, in which the concepts of geodesic curves or parallel transport can be translated.

Definition 8.13. *Let $p \in M$, a local set of coordinates around p is a smooth diffeomorphism $(x^1, ..., x^n) : U \to V$ from a open neighborhood U of p in M to a subset $V \subset \mathbb{R}^n$.*

The datum of local coordinates $x = (x^1, ..., x^n)$ on an open subset $U \subset M$ allows to define a natural set of *local trivializing sections* of the dual bundle TM^* as $(dx^1, ..., dx^n)$. In other words, for any $x \in U$, $(dx^1(x), ..., dx^n(x))$ is a basis of the vector space T_xM^*, and the dependence of this basis on x is smooth.

What's more, one can also define a corresponding set of local trivializing sections $(\frac{\partial}{\partial x^1}, ..., \frac{\partial}{\partial x^n})$ of TM: $\frac{\partial}{\partial x^1}, ...,$ and $\frac{\partial}{\partial x^n}$ are vector fields on U such that, for any $x \in U$, $(\frac{\partial}{\partial x^1}(x), ..., \frac{\partial}{\partial x^n}(x))$ is a basis of T_xM, whose dual basis of T_xM^* with respect to metric g_x is $(dx^1(x), ..., dx^n(x))$.

The following proposition expresses some of the above-defined concepts in terms of local coordinates, and will be of great use when it comes to performing practical calculations on manifolds.

Proposition 8.1. *Let M be equipped with a connection ∇, $U \subset M$ an open subset, and $(x^1, ..., x^n)$ local coordinates on U. Define the Christoffel symbols of ∇ in coordinates $(x^1, ..., x^n)$ as:*

$$\forall x \in U,\ \forall i, j, k = 1, ..., n,\ \ \Gamma^i_{j,k}(x) = i\text{-th component of } \nabla_{\frac{\partial}{\partial x^j}} \frac{\partial}{\partial x^k}\ (x). \tag{8.28}$$

Then,

1. *If $Z(t) = \sum_j z^j(t) \frac{\partial}{\partial x_j}(\gamma(t))$ is a parallel vector field along a given curve $\gamma : I \to U$, one has:*

$$\forall t \in I,\ \ \forall i = 1, ..., n,\ \ z^{i\prime}(t) + \sum_{j,k} \Gamma^i_{j,k}(\gamma(t))\gamma^{j\prime}(t)z^k(t) = 0, \tag{8.29}$$

2. *If $\gamma : I \mapsto U$ is a geodesic curve, the geodesic equation (8.26) reads, in coordinates:*

$$\forall t \in I,\ \ \forall i = 1, ..., n,\ \ \gamma^{i\prime\prime}(t) + \sum_{j,k} \Gamma^i_{j,k}(\gamma(t))\gamma^{j\prime}(t)\gamma^{k\prime}(t) = 0, \tag{8.30}$$

8.2.2 Numerical approximation of parallel transport on a submanifold of $\mathbb{R}^d$

8.2.2.1 Schild's ladder algorithm for approximating the parallel transport of a tangent vector on a Riemannian manifold

In this subsection, we present a general construction on a Riemannian manifold - the *Schild's ladder's procedure*[5], originally introduced in [123] - which is especially used in the theory of general relativity for approximating the parallel transport of a tangent vector along a curve. This idea was later applied in various contexts, such as that of computational anatomy. For instance, in [209], the authors register and compare the anatomies of the brains of individual subjects under the form of diffeomorphisms with a fixed 'template' brain; the (vector) quantities attached to the brain of a subject are transported to the reference configuration using Schild's ladder's procedure (see also [271] in the more general context of shape deformation).

Let (M, g) be a Riemannian manifold. For any point $x \in M$, denote as $||.||_x$ the norm induced by g_x on T_xM, and denote as d the Riemannian distance over M. M is endowed with its Levi-Civita connexion ∇ (actually, the only important feature of ∇ in the device of this algorithm is its torsion-free character). Schild's ladder's algorithm consists of two steps.

First step: parallel transport of a tangent vector between two 'close points'. Let $p \in M$, and $v \in T_pM$ a tangent vector at p, whose parallel transport is investigated. There exists a small radius $r > 0$ such that the geodesic ball $B_r(p)$ is strongly convex. Let $q \in B_r(p)$, and take any curve $\gamma : [0, \tau] \to B_r(p)$, parametrized by arc length, connecting p to q, that is $\gamma(0) = p$, $\gamma(\tau) = q$, and denote $u = \gamma'(0) \in T_pM$. The goal of this step is to devise a consistent approximation of the parallel transport of v from T_pM to T_qM along γ. To this end, introducing a small parameter $\sigma > 0$, the following procedure is carried out (see also figure 8.21):

1. Let γ_1 be the unique geodesic on M, starting from p with initial velocity vector v, i.e. $\gamma_1(0) = p$, and $\gamma_1'(0) = v$. A point p_1 is found on M as $p_1 = \gamma_1(\sigma) = \exp_p(\sigma v)$.
2. Let γ_2 be the unique geodesic on M connecting p_1 to q. A new point m is taken as the midpoint of γ_2, that is: $m = \exp_{p_1}\left(\frac{1}{2}\log_{p_1}(q)\right)$.
3. Let γ_3 be the unique geodesic on M connecting p to m. A new point p_2 is obtained by expanding γ_3 past m for the same amount of arc length as that from p to m. In other words: $p_2 = \exp_p\left(2\log_p(m)\right)$.
4. Let γ_4 be the unique geodesic on M from q to p_2, and parametrize γ_4 in such a way as γ_4 has the same (constant) speed as γ_1. Then an approximation $\widetilde{Tv}$ of $T(\gamma)_0^\tau(v)$ is obtained as the initial velocity vector of γ_4 (with the above chosen parametrization), i.e. $\widetilde{Tv} = \frac{1}{\sigma}\log_q(p_2)$.

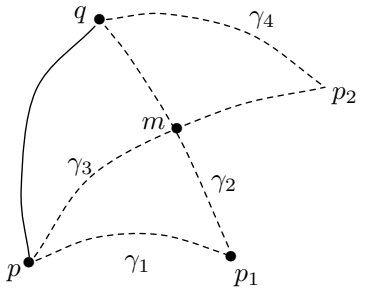


Figure 8.21: Illustration of the Schild's ladder's algorithm for the parallel transport of $v \in T_pM$ from p to q along γ, when p and q are 'close enough'.

This construction is legitimated by the following result:

5. It is amusing that this very elegant idea influenced the science-fiction novelist Greg Egan, who wrote a short story - *Schild's ladder* - based upon this principle.

Proposition 8.2. *Provided σ and τ are chosen small enough, the above construction is valid. Moreover, the error induced by the process can be estimated as:*

$$||T(\gamma)_0^\tau(v) - \widetilde{Tv}||_q = \mathcal{O}\left(\frac{(\tau+\sigma)^3}{\sigma}\right), \tag{8.31}$$

where the $\mathcal{O}\left(\frac{(\tau+\sigma)^3}{\sigma}\right)$ residual only depends on the norm of v and local characteristics of M.

Proof. This proof mostly reproduces the estimates in [191].

First, provided τ and σ are small enough with respect to r, the whole construction is concentrated in $B_r(p)$: points p_1, m, p_2 belong to $B_r(p)$. The existence of the geodesic curves $\gamma_1, \gamma_2, \gamma_3, \gamma_4$ is then guaranteed and the above construction is valid.

Let us now turn to proving (8.31). Take a set of local coordinates $(x^1, ..., x^n)$ on $B_r(p)$ (e.g. but non necessarily *geodesic coordinates*). Denote as $(dx^1, ..., dx^n)$ (resp. $(\frac{\partial}{\partial x_1}, ...\frac{\partial}{\partial x_n})$) the associated trivializing sections of the dual bundle TM^* (resp. of the tangent bundle TM). If $x \in B_r(p)$, we denote with the i exponent the i-th coordinate of x in this set of coordinates, and if $w \in T_xM$, w^i is the i-th coordinate of w in basis $(\frac{\partial}{\partial x_1}(x), ...\frac{\partial}{\partial x_n}(x))$. We eventually denote as $\Gamma^i_{j,k}$ the *Christoffel symbols* of ∇ in this set of coordinates.

Consider any component $i = 1, ..., n$; we search for a first order estimate of $\widetilde{Tv}^i - v^i$ (i.e. with a residual $\mathcal{O}((\tau+\sigma)^2)$). We will repeatedly use the convergence of $\widetilde{Tv}^i$ to v^i, under the form:

$$(\widetilde{Tv}^i - v^i) = \mathcal{O}(\tau+\sigma). \tag{8.32}$$

Such a convergence result is a fairly easy consequence of the exact expression of $\widetilde{Tv}$ provided along steps $(1-4)$ above. We use the decomposition:

$$(p_2^i - q^i) - (p_1^i - p^i) = (p_2^i - m^i) + (m^i - q^i) - (p_1^i - m^i) - (m^i - p^i). \tag{8.33}$$

From the definition of γ_1, γ_4, we have, using a second order Taylor expansion on the coordinate functions:

$$(p_2^i - q^i) = \sigma\widetilde{Tv}^i - \frac{1}{2}\sigma^2 \sum_{j,k} \Gamma^i_{j,k}\widetilde{Tv}^j\widetilde{Tv}^k + \mathcal{O}(\sigma^3), \tag{8.34}$$

$$(p_1^i - p^i) = \sigma v^i - \frac{1}{2}\sigma^2 \sum_{j,k} \Gamma^i_{j,k} v^j v^k + \mathcal{O}(\sigma^3), \tag{8.35}$$

where the $\mathcal{O}(\sigma^3)$ only depends on the third order derivatives of the components of γ_1, γ_4, and thus only on its first order derivatives (which are uniformly bounded) as well as some intrinsic characteristics of M since they are geodesic curves. Notice that the dependance of the Christoffel symbols on the point where it is computed has been dropped for considering its variations only breeds higher-order terms in the calculation: we will therefore proceed in the computation as if they were constant.

Similarly, introducing $\lambda = \frac{1}{2}d(p_1, q)$, $\rho = d(p, m)$, as well as the tangent vectors $z = \gamma_2'(0)$, $w = \gamma_3'(0)$, and their parallel transports $z_{||} = T(\gamma_2)_0^\lambda(z), w_{||} = T(\gamma_3)_0^\rho(w) \in T_mM$, one finds:

$$(p_2^i - m^i) = \rho w^i_{||} - \frac{1}{2}\rho^2 \sum_{j,k} \Gamma^i_{j,k} w^j_{||} w^k_{||} + \mathcal{O}(\rho^3), \tag{8.36}$$

$$(m^i - q^i) = -\lambda z^i_{||} - \frac{1}{2}\lambda^2 \sum_{j,k} \Gamma^i_{j,k} z^j_{||} z^k_{||} + \mathcal{O}(\lambda^3), \tag{8.37}$$

$$(p_1^i - m^i) = -\lambda z^i - \frac{1}{2}\lambda^2 \sum_{j,k} \Gamma^i_{j,k} z^j z^k + \mathcal{O}(\lambda^3), \tag{8.38}$$

$$(m^i - p^i) = \rho w^i - \frac{1}{2}\rho^2 \sum_{j,k} \Gamma^i_{j,k} w^j w^k + \mathcal{O}(\rho^3) \tag{8.39}$$

Putting equations (8.34-8.39) into (8.33) and using the fact that λ and ρ are of order $\mathcal{O}(\tau+\sigma)$ yields:

$$\begin{array}{rl} \sigma(\widetilde{Tv}^i - v^i) - \frac{1}{2}\sigma^2 \sum_{j,k} \Gamma^i_{j,k}(\widetilde{Tv}^j \widetilde{Tv}^k - v^j v^k) = & \rho(w^i_{||} - w^i) - \lambda(z^i_{||} - z^i) - \frac{1}{2}\rho^2 \sum_{j,k} \Gamma^i_{j,k}(w^j_{||} w^k_{||} - w^j w^k) \\ & + \frac{1}{2}\lambda^2 \sum_{j,k} \Gamma^i_{j,k}(z^j_{||} z^k_{||} - z^j z^k) + \mathcal{O}((\tau+\sigma)^3). \end{array} \tag{8.40}$$

On the other hand, we can exploit the parallel transport equation (8.29) to obtain a (rather easy) 0-th order estimate of the discrepancies $(w^i_{||} - w^i)$ and $(z^i_{||} - z^i)$ as:

$$(w^i_{||} - w^i) = \mathcal{O}(\tau+\sigma) \quad ; \quad (z^i_{||} - z^i) = \mathcal{O}(\tau+\sigma), \tag{8.41}$$

then the more interesting first order expansions:

$$(w^i_{||} - w^i) = -\sum_{j,k} \Gamma^i_{j,k} w^j (m^k - p^k) + \mathcal{O}((\tau+\sigma)^2), \tag{8.42}$$

$$(z^i_{||} - z^i) = -\sum_{j,k} \Gamma^i_{j,k} z^j (m^k - p^k_1) + \mathcal{O}((\tau+\sigma)^2). \tag{8.43}$$

Moreover, we also have:

$$\begin{array}{rcl} (m^i - p^i) & = & \frac{1}{2}(p^i_2 - p^i) + \mathcal{O}((\tau+\sigma)^2) \\ & = & \frac{1}{2}(\tau u^i + \sigma v^i) + \mathcal{O}((\tau+\sigma)^2) \end{array}, \tag{8.44}$$

and in the same way

$$(m^i - p^i_1) = \frac{1}{2}(\tau u^i - \sigma v^i) + \mathcal{O}((\tau+\sigma)^2). \tag{8.45}$$

Pulling (8.40 - 8.45) together, we end up with:

$$\begin{array}{rcl} \sigma(\widetilde{Tv}^i - v^i) & = & \frac{\lambda}{2}\sum_{j,k} \Gamma^i_{j,k} z^j (\tau u^k - \sigma v^k) - \frac{\rho}{2}\sum_{j,k} \Gamma^i_{j,k} w^j (\tau u^k + \sigma v^k) + \mathcal{O}((\tau+\sigma)^3) \\ & = & \frac{\tau}{2}\sum_{j,k} \Gamma^i_{j,k}(\lambda z^j - \rho w^j) u^k - \frac{\sigma}{2}\sum_{j,k} \Gamma^i_{j,k}(\lambda z^j + \rho w^j) v^k + \mathcal{O}((\tau+\sigma)^3) \end{array}. \tag{8.46}$$

Now using the expansions:

$$(\lambda z^i + \rho w^i) = \tau u^i + \mathcal{O}((\tau+\sigma)^2), \tag{8.47}$$

$$(\rho w^i - \lambda z^i) = \sigma v^i + \mathcal{O}((\tau+\sigma)^2), \tag{8.48}$$

we eventually get:

$$\begin{array}{rcl} \sigma(\widetilde{Tv}^i - v^i) & = & -\frac{\tau\sigma}{2}\sum_{j,k} \Gamma^i_{j,k}\left(v^j u^k + u^j v^k\right) + \mathcal{O}((\tau+\sigma)^3) \\ & = & -\tau\sigma \sum_{j,k} \Gamma^i_{j,k} u^j v^k + \mathcal{O}((\tau+\sigma)^3) \end{array}, \tag{8.49}$$

where the last equality holds because the Levi-Civita connection is torsion-free (which implies the symmetry of the Christoffel symbols $\Gamma^i_{j,k} = \Gamma^i_{k,j}$). Thus,

$$(\widetilde{Tv}^i - v^i) = -\tau \sum_{j,k} \Gamma^i_{j,k} u^j v^k + \mathcal{O}\left(\frac{(\tau+\sigma)^3}{\sigma}\right). \tag{8.50}$$

Using one last time the parallel transport equation (8.29) to express the very definition of $T(\gamma)^\tau_0(v)$, one obtains the estimate:

$$[T(\gamma)^\tau_0(v)]^i - v^i = -\tau \sum_{j,k} \Gamma^i_{j,k} u^j v^k + \mathcal{O}(\tau^2), \tag{8.51}$$

and comparing this to (8.52) eventually yields:

$$([T(\gamma)_0^\tau(v)]^i - \widetilde{Tv}^i) = \mathcal{O}\left(\frac{(\tau+\sigma)^3}{\sigma}\right), \tag{8.52}$$

which is the desired result. □

Second step: parallel transport of a tangent vector between any two points. Let $p, q \in M$, $v \in T_pM$, and $\gamma : [0, \tau] \to M$ a curve connecting p to q, parametrized by arc length. As q may stand far from p, the above construction no longer holds, for geodesics $\gamma_2, \gamma_3, \gamma_4$ may fail to exist. The solution is then to apply the first step repeatedly on small portions of γ, where it is possible; this leads to a diagram such as that of figure 8.22, after which the algorithm is named.

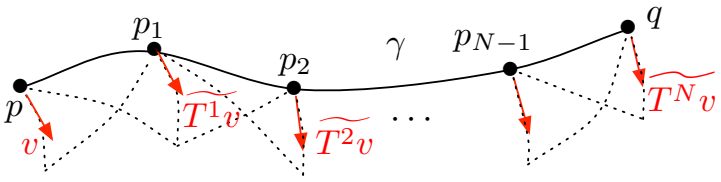


Figure 8.22: Repeated application of the local procedure for an approximation of the parallel transport of v from p to q along γ.

More precisely, since $\gamma([0, \tau])$ is compact, there exists $r > 0$ such that, $\gamma([0, \tau])$ can be covered by finitely many geodesic balls $B_{r_i}(p_i)$, $i = 1, ..., N$, where $p_i = \gamma(t^i)$, for some $t^i \in [0, \tau]$ (by convention, we set $t^0 = 0$, $t^N = \tau$), in which the local procedure described in the first step can be applied. This procedure is then performed between each pair of points p_i, p_{i+1}, and this produces a sequence $\widetilde{T^1 v}, ..., \widetilde{T^N v}$ of approximate parallel transports of v at points $p_1, ..., p_N$, and one can eventually take $\widetilde{T^N v}$ as an approximation of $T(\gamma)_0^\tau(v)$. This construction is legitimated by the following result:

Proposition 8.3. *The convergence error for the Schild's ladder procedure is controlled as:*

$$||T(\gamma)_0^\tau(v) - \widetilde{T^N v}||_q = \mathcal{O}(\sigma), \tag{8.53}$$

where σ is the maximum length of a portion of curve $\gamma([t^i, t^{i+1}])$, and the $\mathcal{O}(\sigma)$ only depends on the norm of v and characteristics of the surface in a neighborhood of γ.

Proof. One has:

$$\begin{aligned} ||T(\gamma)_0^\tau(v) - \widetilde{T^N v}||_q &\leq ||T(\gamma)_0^\tau(v) - T(\gamma)_{t^{N-1}}^{t^N}(\widetilde{T^{N-1}v})||_q + ||T(\gamma)_{t^{N-1}}^{t^N}(\widetilde{T^{N-1}v}) - \widetilde{T^N v}||_q \\ &= ||T(\gamma)_0^{t^{N-1}}(v) - \widetilde{T^{N-1}v}||_{p_{N-1}} + ||T(\gamma)_{t^{N-1}}^{t^N}(\widetilde{T^{N-1}v}) - \widetilde{T^N v}||_q \\ &= ||T(\gamma)_0^{t^{N-1}}(v) - \widetilde{T^{N-1}v}||_{p_{N-1}} + \mathcal{O}(\sigma^2) \end{aligned}, \tag{8.54}$$

where we used the fact that the parallel transport operator $T(\gamma)_{t^{N-1}}^{t^N} : T_{p_{N-1}}M \to T_{p_N}M$ is an isometry, and proposition 8.2 to control the second term. Applying this procedure N times yields the result, because the $\mathcal{O}(\sigma^2)$ which appears depends only on local characteristics of M. □

Remark 8.8. Infringing a little bit on the forthcoming applications, let us note that translating Schild's ladder's algorithm into the numerical contest without introducing any subsequent approximation is not an

easy task, for it involves the computation of geodesic curves on M. Depending on the way M is represented and on the desired accuracy, several ways allow to approximate these geodesic curves. In the context of interest in this chapter, $M = \Gamma$ is a surface embedded in $\mathbb{R}^3$, which is locally parametrized from an associated surface triangulation $\mathcal{S}$: from a given triangle $T \to \mathcal{S}$, a local parameterization $\sigma : T \to \Gamma$ is available. Then, for instance, the geodesic curve emerging from a point $p \in \sigma(T)$, with initial velocity $v \in T_p\Gamma$ is approximated by the curve γ defined by (see Figure 8.23):

$$\gamma : t \mapsto \sigma(t\, d\sigma_p^{-1}(v)).$$

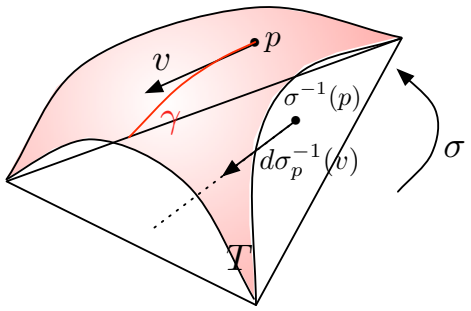


Figure 8.23: Approximation of the geodesic on Γ emerging from p, with velocity $v \in T_p\Gamma$.

8.2.2.2 Parallel transport of a bilinear form over a Riemannian manifold

Let (M, g) be a Riemannian manifold of dimension n, and ∇ the associated Levi-Civita connexion. Given two points $p, q \in M$ connected by a smooth curve γ, and a symmetric bilinear form $b_p : T_pM \times T_pM \to \mathbb{R}$ over T_pM, we shall need to account for the parallel transport of b_p from p to q via γ. A simple way to do this, using the Schild's ladder's algorithm is enabled by the following proposition:

Proposition 8.4. *Let $p, q \in M$, and $\gamma : I \to M$ a differentiable curve, $t_0 \in I$, and denote $p = \gamma(0)$. Let $b \in \mathrm{Sym}^2(TM^*)$ a symmetric bilinear tensor field which is parallel along γ, in the sense of definition 8.9. Denote as $\lambda_1, ..., \lambda_n$ the eigenvalues of b_p, and $(e_1, ..., e_n)$ an associated orthonormal basis of eigenvectors (for the scalar product g_p). Then for any $t \in I$, $b_{\gamma(t)}$ has eigenvalues $\lambda_1, ..., \lambda_n$, and an associated orthonormal basis of eigenvectors of $T_{\gamma(t)}M$ is $(T(\gamma)^t_{t_0}(e_1), ..., T(\gamma)^t_{t_0}(e_n))$.*

Proof. It is enough to prove that, for every $t \in I$,

$$\forall i = 1, ..., n,\ \ \forall v \in T_{\gamma(t)}M,\ \ b_{\gamma(t)}\left(T(\gamma)^t_{t_0}(e_i), v\right) = \lambda_i g_{\gamma(t)}\left(T(\gamma)^t_{t_0}(e_i), v\right). \tag{8.55}$$

Indeed, this will prove that, for any $i = 1, ..., n$, $T(\gamma)^t_{t_0}(e_i)$ is an eigenvector of $b_{\gamma(t)}$ associated to the eigenvalue λ_i. Thus, as $(T(\gamma)^t_{t_0}(e_1), ..., T(\gamma)^t_{t_0}(e_n))$ is an orthonormal basis of $T_{\gamma(t)}M$, owing to the properties of the Levi-Civita connection, the result will follow.

Now, since $T(\gamma)^t_{t_0} : T_{\gamma(t_0)}M \to T_{\gamma(t)}M$ is a linear isomorphism, (8.55) is equivalent to:

$$\forall i = 1, ..., n,\ \ \forall v \in T_{\gamma(t_0)}M,\ \ b_{\gamma(t)}\left(T(\gamma)^t_{t_0}(e_i), T(\gamma)^t_{t_0}(v)\right) = \lambda_i g_{\gamma(t)}\left(T(\gamma)^t_{t_0}(e_i), T(\gamma)^t_{t_0}(v)\right). \tag{8.56}$$

Eventually, because for any $v \in T_{\gamma(t_0)}M$, $T(\gamma)^t_{t_0}(v)$ is a parallel vector field, and b and g are parallel along γ, the quantity

$$b_{\gamma(t)}\left(T(\gamma)^t_{t_0}(e_i), T(\gamma)^t_{t_0}(v)\right) - \lambda_i g_{\gamma(t)}\left(T(\gamma)^t_{t_0}(e_i), T(\gamma)^t_{t_0}(v)\right)$$

is constant on I. As it amounts to 0 at $t = t_0$, this ends the proof. □

This small result turns out to be very convenient when it comes to describing the parallel transport of a symmetric bilinear form $b : T_pM \times T_pM \to \mathbb{R}$ from $p \in M$ to $q \in M$ along a segment of curve γ. Indeed, it basically states that, to achieve this, one only need transport n vectors, the vectors of an orthonormal basis of T_pM composed of eigenvectors of b (and actually it is enough to transport $n-1$ of these vectors, for the n-th being orthogonal to the others).

8.2.3 Definition of a suitable Riemannian structure for anisotropic surface remeshing

From this section on, we will be considering the issue of anisotropic remeshing of a triangulation $\mathcal{S}$ of a surface Γ embedded in $\mathbb{R}^3$. Such a surface is endowed with the Riemannian structure inherited from the canonical scalar product of $\mathbb{R}^3$, which we shall denote as g, and refer to as the *natural Riemannian structure* of Γ. However, we will also be investigating other structures over Γ, induced by different metric tensor fields $b \in \mathrm{Sym}^2(T\Gamma^*)$, which account for the desired size prescriptions for remeshing. In the sequel, we will denote with a ^{b} superscript all the features (length, volume form, Levi-Civita connection, etc...) attached to the Riemannian structure of Γ induced by b, and without any superscript the corresponding features attached to its natural structure.

As in the beginning of section 8.1, we consider a smooth oriented surface $\Gamma \subset \mathbb{R}^3$, together with an interpolating triangulation $\mathcal{S}$ satisfying hypotheses (8.1) of which is given. We aim at modifying $\mathcal{S}$ into a nicely-shaped - potentially anisotropic - triangulation $\widetilde{\mathcal{S}}$ which is a close geometric approximation of Γ in the sense that $d^H(\widetilde{\mathcal{S}}, \Gamma) \leq \varepsilon$, for a prescribed tolerance $\varepsilon > 0$.

By analogy with the isotropic context developed in section 8.1.4, the information about the expected behavior of the sought triangulation $\widetilde{\mathcal{S}}$ is encoded in a Riemannian structure $b \in \mathrm{Sym}^2(T\Gamma^*)$ over the ideal surface Γ. The problem of modifying $\mathcal{S}$ with respect to the prescribed anisotropic information thus boils down to producing a mesh $\widetilde{\mathcal{S}}$ whose edges pq have unit length $\ell^b(pq)$, that is, such that the underlying curves γ on Γ have unit length $\ell^b(\gamma)$.

8.2.3.1 Definition of a Riemannian structure adapted to the geometric approximation of a surface

The sought anisotropic behavior of the resulting mesh may stem from external concerns (e.g. error estimates associated to mechanical analyses held on $\mathcal{S}$), or from the intrinsic anisotropic geometry of Γ itself. This section is intended as the anisotropic twin of section 8.1.4, and explains how a metric tensor field adapted to the geometric approximation of Γ may be constructed.

To this end, we rely once again on a heuristic based upon theorem 8.2. Suppose Γ is a smooth, compact manifold without boundary, delimiting a bounded and connected domain $\Omega \subset \mathbb{R}^3$: $\Gamma = \partial\Omega$. Let $\mathcal{S}$ an approximating triangulation of Γ matching conditions (8.8). We demand that $\mathcal{S}$ satisfy:

$$d^H(\mathcal{S}, \Gamma) \leq \varepsilon. \tag{8.57}$$

For this to hold, a sufficient condition is expressed in terms of the signed distance function d_Ω to Ω; for all triangles $T \in \mathcal{S}$ one should have:

$$\frac{2}{9} \max_{x \in T} \max_{y,z \in T} |\mathcal{H}(d_\Omega)(x)|(yz, yz) \leq \varepsilon. \tag{8.58}$$

Now, if $T \in \mathcal{S}$ is 'very close' to some point $x \in \Gamma$, one may 'assume' that $|\mathcal{H}(d_\Omega)|$ stays constant over T, with value $|\mathcal{H}(d_\Omega)(x)|$. The sufficient condition (8.58) for triangle T then becomes:

$$\forall y, z \in T, \ \frac{2}{9}|\mathcal{H}(d_\Omega)(x)|(yz, yz) \leq \varepsilon. \tag{8.59}$$

Let $n(x)$ the unit normal to Γ at x, pointing outside Ω, $\kappa_i(x)$ $(i = 1, 2)$ the two principal curvatures of Γ at x, and $e_i(x)$ the associated principal directions, so that in basis $(e_1(x), e_2(x), n(x))$, $\mathcal{H}(d_\Omega)(x)$ reads:

$$\mathcal{H}(d_\Omega)(x) = \begin{pmatrix} \kappa_1(x) & 0 & 0 \\ 0 & \kappa_2(x) & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Suppose now that the normal vector field to $\mathcal{S}$ (which is defined almost everywhere) is 'close' to that of Γ. This means that, roughly speaking, if $T \in \mathcal{S}$ is very close from $x \in \Gamma$, T should be enclosed in the tangent plane $T_x\Gamma$.

Define $b \in \mathrm{Sym}^2(T\Gamma^*)$ as follows: for any $x \in \Gamma$, b_x is the bilinear form on $T_x\Gamma$ whose matrix in basis $(e_1(x), e_2(x))$ is:

$$\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}, \quad \lambda_i = \max\left(\frac{1}{h_{max}^2}, \min\left(\frac{1}{h_{min}^2}, \frac{2|\kappa_i(x)|}{9\varepsilon}\right)\right), \tag{8.60}$$

where h_{min} and h_{max} are the same parameters as those introduced in section 8.1.4, representing respectively lower and upper bounds on the authorized (Euclidean) size of the edges of the sought mesh.

From equation (8.59), a sufficient condition for (8.57) to hold is that, for any $T \in \mathcal{S}$ 'close' to a point $x \in \Gamma$, each one of the three edges of T has length less or equal to 1 when measured with respect to b_x.

These informal calculations lead us to look for a triangulation $\widetilde{\mathcal{S}}$ whose edges pq have length $\ell^b(pq)$ as close to 1 as possible, where $b \in \mathrm{Sym}^2(T\Gamma^*)$ is the metric tensor field defined by formula (8.60).

This formal analysis tacitly implies that the considered ideal surface Γ is smooth. However, in many cases, reality is bleaker than that: as mentioned in section 8.1.1, we are often interested in surfaces which do not enjoy so much regularity: for instance, mechanical parts often present ridge angles, or singularities (see for instance figures 8.15, and 8.17 (c-d)). In this context, we should rely on different descriptions of the local desired size features, depending on the regularity of the considered region:

- At a *singular point* $x \in \Gamma$, several different pieces of smooth surface meet. An isotropic local size feature is associated to each one of them by means of the analysis held in section 8.1.4. The metric at x is then defined by means of a single, isotropic, local size feature (i.e. the minimum size prescription among those dictated by the pieces of surface meeting at x).
- At a *ridge point* $x \in \Gamma$ which is not singular, two different pieces of smooth surface intersect, say Γ_1, Γ_2. The local geometry is then defined by means of the two associated normal vectors $n_1(x)$ and $n_2(x)$ at x, and the tangent vector $\tau(x)$ to the intersecting ridge curve. At such a point, we keep in mind three local size features: one in the direction of $\tau(x)$, two in the directions of $n_1(x) \wedge \tau(x) \in T_x\Gamma_1$, $n_2(x) \wedge \tau(x) \in T_x\Gamma_2$. Each time point x is considered, only the relevant information is kept, depending on whether x is considered as belonging to Γ_1 or Γ_2.
- At an *open point* x, the description is identical, except that only one portion of surface lands at x.
- At a *reference point* $x \in \Gamma$ which is not a ridge point, recall that x belongs to a smooth part of Γ, and at the same time to a smooth curve γ drawn on Γ. The local size prescription is then described in the standard way - that is by means of a bilinear form over $T_x\Gamma$, except that the geometric error based metric tensor field (8.60) has to be *intersected* with another very similar tensor field associated to the geometric error of the approximation of γ.
- Eventually, at a *regular point* $x \in \Gamma$ (which is the case of 'most of the points' of Γ), the bilinear form over $T_x\Gamma$ given by (8.60) is used for describing the local size features.

Remark 8.9. In numerical practice, the accuracy of the proposed method in capturing the anisotropic behavior of the considered ideal surface Γ generally depends a lot on the shape of the input triangulation $\mathcal{S}$, mainly because of the strong dependence of the algorithms to approximate the curvature tensor of the ideal surface associated to a discrete triangulation on the shape of the triangulation.

8.2.3.2 Gradation of an anisotropic metric tensor field

Throughout this section, g stands for the Riemannian structure on Γ inherited from the Euclidean scalar product of $\mathbb{R}^3$, and all the geometric features on Γ - Levi-Civita connection, geodesic curves, etc... - are assumed to be related to this natural Riemannian structure. Let also $b \in \mathrm{Sym}^2(T\Gamma^*)$ the metric tensor field associated to the geometric approximation of Γ, as discussed in the previous section.

As in the case of isotropic surface remeshing, relying on this raw metric tensor field to drive our operators may cause the production of poor quality elements. However, this problem is more difficult to formulate and assess in the present anisotropic context.

In the isotropic context of section 8.1.4.5, a gradation was imposed on the size map $h : \Gamma \to \mathbb{R}$ so that a unit triangulation with respect to h would consist of well-shaped triangles, or more accurately of triangles whose edges have 'similar' lengths. Such a prescription would not make any sense in the anisotropic context, since anisotropic elements are by essence distorted. Likewise, one could think of demanding that grading of the metric tensor field b would help in getting a mesh, whose elements have high qualities, but this would be a bit awkward since, as we shall see in section 8.2.5.1, the very concept of mesh quality relies (and should rely) on b itself. Hence, we found no more satisfactory way to carry out the gradation of b than to follow our intuition of what should be a 'good anisotropic mesh', that is, a mesh whose triangles may be distorted, but whose general orientation and sizes in the characteristic directions of this orientation should show smooth variations.

Let us now describe a heuristic way of translating this idea into our mathematical framework which turned out to meet our expectations. Let $p \in \Gamma$ any point, $u \in T_p\Gamma$ a unit tangent vector (with respect to g_p). From our definition of b, and notably relation (8.60) between b and the desired size prescriptions, the *characteristic size at p in direction u*, $h_p(u)$, is defined as:

$$h_p(u) = \frac{1}{\sqrt{b_p(u,u)}}.$$

As we have seen in Section 8.1.4.5, the gradation procedure considers an edge $pq \in \mathcal{S}$ of the surface triangulation, and updates the size prescriptions at either p or q so that a certain condition - yet to be defined in the anisotropic setting - is fulfilled. Let $\gamma : [0, \ell(pq)] \to \Gamma$ the corresponding curve on Γ, parametrized by arc length. If $u \in T_p\Gamma$ is a unit tangent vector at p, the *characteristic size in direction u along γ* is defined as $h_{\gamma(t)}\left(T(\gamma)_0^t u\right)$. Following the heuristic considerations of section 8.1.4.5, our aim then becomes to enforce that:

$$\forall u \in T_p\Gamma \text{ s.t } ||u||_p = 1,\ \forall t \in (0, \ell(pq)),\ \left| \frac{d}{dt}\left(h_{\gamma(t)}\left(T(\gamma)_0^t u\right)\right)\right| \leq h_{grad},$$

where h_{grad} denotes the desired bound on the size prescriptions. Note that, $T(\gamma)_0^{\ell(pq)} : T_p\Gamma \to T_q\Gamma$ being an isometric isomorphism, this condition is actually symmetric in p and q. In numerical practice, this is discretized as:

$$\forall u \in T_p\Gamma \text{ s.t } ||u||_p = 1,\ \left| \frac{h_q\left(T(\gamma)_0^{\ell(pq)}(u)\right) - h_p(u)}{\ell(pq)} \right| \leq h_{grad}. \tag{8.61}$$

At this point, several possible strategies emerge as for the anisotropic gradation procedure: imposing that it should hold for any tangent direction $u \in T_p\Gamma$ amounts to controlling the variations of b itself - as a symmetric bilinear form - along γ. This can be rigorously expressed in terms of the Levi-Civita connection on Γ, but then becomes tedious to implement numerically. We chose instead to stay as simple (and heuristic) as possible, and to enforce (8.61) in only one direction, namely $u = \gamma'(0)$, the direction of the associated curve to the processed edge pq. Similarly, many operations are possible when it comes to modifying b_q (or b_p) in enforcing (8.61): one could think of altering the principal directions of these bilinear forms, or finding the optimal way to modify the two corresponding eigenvalues. For the sake of simplicity, we thought it better to perform only an homothetic transformation on b_q or b_p, so that the shape of the corresponding unit ellipsoid in the tangent plane is retained. All in all, the proposed procedure reads as follows: let $pq \in \mathcal{S}$ and edge,

1. compute the unit vector $u := \gamma'(0)$ to the normal curve γ associated to pq on Γ.
2. Compute the length $\ell(pq)$ of γ, as well as an approximation to the parallel transported vector $Tu := T(\gamma)_0^t(u)$ (see section 8.2.2.1).
3. If $h_q(Tu) > h_p(u) + h_{grad}\,\ell(pq)$, replace b_q by αb_q, where $\alpha \in (0,1)$ is the unique real value which satisfies:
$$\frac{1}{\sqrt{a}} h_q(Tu) - h_p(u) = h_{grad}\,\ell(pq).$$
Else, go back to the first step, looking at pq with the reversed orientation (i.e. as qp): in this case, b_p will be updated.

In addition to this procedure, whenever a gradation on the Riemannian structure b on Γ is performed, we thought it better to impose beforehand that the metric b_p at ridge points $p \in \Gamma$ should be converted into an isotropic one. The reason is that, generally speaking, ridge curves stand for boundaries between pieces of surface enjoying utterly independent features. In this view, specific orientation data attached to different pieces of surface should not be allowed to interfere with one another. See the examples of section 8.2.6 for an illustration of this point.

8.2.4 Metric tensor fields on triangulated surfaces in numerical practice

The purpose of this section is to detail several issues related to the numerical treatment of symmetric bilinear forms defined over a surface in $\mathbb{R}^3$.

Let $b \in \mathrm{Sym}^2(T\Gamma^*)$ to be stored in a discrete fashion over $\mathcal{S}$. We intend to proceed exactly as in the scalar case described in section 8.1.4, keeping the information related to b at each vertex p of $\mathcal{S}$, so that the bilinear form b_x is interpolated from the known data when the considered point $x \in \Gamma$ is not a vertex of $\mathcal{S}$.

8.2.4.1 A word about the numerical storage of fields of symmetric bilinear forms

Let $b \in \mathrm{Sym}^2(T\Gamma^*)$, i.e. for each point $p \in \Gamma$, b_p is a symmetric bilinear form over the two-dimensional space $T_p\Gamma$: b_p inherently corresponds to a symmetric 2×2 matrix, *once a basis of T_pS has been set*. Unfortunately, it is well-known that the tangent bundle $T\Gamma$ is generally not trivial, i.e. there does not exist *globally defined*, smooth vector fields $X, Y \in \Gamma(T\Gamma)$ such that, for any point $p \in \Gamma$, $(X(p), Y(p))$ is a basis of $T_p\Gamma$. Consequently, if we are to store b_p as a 2×2 matrix, we have to store as well the corresponding basis of $T_p\Gamma$ in which b_p is expressed.

We chose a rather different point of view: taking advantage of the natural embedding $T_p\Gamma \subset \mathbb{R}^3$, b_p can be naturally extended to a symmetric bilinear form $\mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$, still denoted as b_p, such that, for all $v \in \mathbb{R}^3$, $b_p(v, n(p)) = 0$, where $n(p)$ is the unit normal to Γ at p (with respect to the specified orientation). Doing so, if $(e_1(p), e_2(p))$ is any orthonormal basis of $T_p\Gamma$, in which b_p has matrix $M \in \mathcal{M}_2(\mathbb{R})$, the matrix of the (extended) form b_p in the direct orthonormal basis $(e_1(p), e_2(p), n(p))$ of $\mathbb{R}^3$ reads:

$$\begin{pmatrix} & & 0 \\ & M & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{8.62}$$

Hence, we will store b_p as a 3×3 symmetric matrix expressed in the canonical basis of $\mathbb{R}^3$, whose kernel contains $n(p)$. From this datum, the matrix of b_p in any orthonormal basis of $T_p\Gamma$ can be recovered using a mere change of basis.

8.2.4.2 Intersection of two fields of symmetric bilinear forms over Γ

We have hitherto only been assuming that the triangulation $\mathcal{S}$ of Γ should be adapted to a Riemannian metric b for the geometric approximation of Γ, computed on account of the analysis of Section 8.2.3. Yet, it

is also interesting to consider adapting $\mathcal{S}$ with respect to any user-defined Riemannian metric m on Γ, which could, once again, arise from an a posteriori error analysis of a numerical simulation performed on $\mathcal{S}$.

In this case, so as to take into account both size prescriptions, an *intersection* of the metrics b and m is performed (see e.g. [145]). Let us briefly describe this operation.

Let $p \in \Gamma$ be a vertex of $\mathcal{S}$, and $B_p \in \mathcal{S}_2(\mathbb{R})$ (resp. $M_p \in \mathcal{S}_2(\mathbb{R})$) the symmetric positive definite matrix associated to b_p (resp. m_p) in a given basis of $T_p\Gamma$. An efficient way to perform the intersection of b_p and m_p involves their *simultaneous reduction* [6]: there exists a matrix $P \in GL_d(\mathbb{R})$ such that both bilinear forms are diagonal in the basis of $\mathbb{R}^d$ accounted for by P, i.e.

$$B_p = P\,\mathrm{diag}(\lambda_i)\,P^T \;\;;\;\; M_p = P\,\mathrm{diag}(\mu_i)P^T. \tag{8.63}$$

As hinted at in section 8.2.3, the bilinear forms we are interested in account for local size prescriptions in that, for any point $x \in \Gamma$, and any direction $v \in T_x\Gamma$ with $g_x(v,v) = 1$, one has $b_x(v,v) = \frac{1}{h_x^2(v)}$, where $h_x(v)$ is the prescribed local size by b at x in direction v (and similarly for m). Consequently, the *intersected* bilinear form $\widetilde{b_p}$ over $T_p\Gamma$, whose matrix $\widetilde{B_p}$ in the considered basis of $T_p\Gamma$ is defined by:

$$\widetilde{B_p} = P\,\mathrm{diag}(\max(\lambda_i,\mu_i))\,P^T$$

retains only the smallest local size features of b_p and m_p (see [6] for a geometric interpretation in terms of unit ellipsoids).

8.2.4.3 Interpolation of two symmetric bilinear forms along a curve

Now, let $x \in \Gamma$ a point which is not a vertex of $\mathcal{S}$, at which $b_x : T_x\Gamma \times T_x\Gamma \to \mathbb{R}$ is to be approximated from the values of b at the vertices of $\mathcal{S}$. In the following, we will only require the case where x is located on a given curve γ connecting two vertices p and q of $\mathcal{S}$, at which b_p and b_q are known. Hence, we focus on this particular situation, which is readily extended to more general ones.

First, consider the simpler problem when all the data belong to a *common* vector space $\mathbb{R}^d$ ($d = 2, 3$), that is, p, q are two points in $\mathbb{R}^d$, and b is a field of symmetric bilinear forms over $\mathbb{R}^d$, to be interpolated along the segment $[0,1] \ni t \mapsto \gamma(t) = (1-t)p + tq$ from its values b_p, b_q at p, and q respectively. Denote as M_p (resp. M_q) the $d \times d$ symmetric positive definite matrix associated to b_p (resp. b_q) in the canonical basis of $\mathbb{R}^d$. Performing the interpolation of M_p and M_q along γ involves the simultaneous reduction of M_p and M_q, as in Section 8.2.4.2: there exists a matrix $P \in GL_d(\mathbb{R})$ such that both bilinear forms b_p and b_q are diagonal in the basis of $\mathbb{R}^d$ accounted for by P, i.e.

$$M_p = P\,\mathrm{diag}(\lambda_i)\,P^T \;\;;\;\; M_q = P\,\mathrm{diag}(\mu_i)\,P^T. \tag{8.64}$$

From the interpretation of the λ_i, μ_i in terms of local size specifications, the simplest interpolation scheme of b_p and b_q along γ, accounting for a linear interpolation of the local size features in the directions of the columns of P reads in terms of the associated matrices:

$$\forall t \in [0,1],\;\; M_{\gamma(t)} = P\,\mathrm{diag}(\delta_i(t))^t P,\;\; \delta_i(t) = \frac{\lambda_i\mu_i}{\left((1-t)\sqrt{\mu_i} + t\sqrt{\lambda_i}\right)^2}. \tag{8.65}$$

In the case we are interested in, b_p and b_q are defined over $T_p\Gamma$ and $T_q\Gamma$ respectively, and are to be interpolated at a point $\gamma(t)$ of a segment of curve $\gamma : [0,1] \to \Gamma$ connecting p to q (γ is parametrized by $[0,1]$ for the sake of convenience). The easiest extension of the above procedure consists in identifying b_p and b_q as symmetric, positive definite bilinear forms over the same tangent plane $T_{\gamma(t)}\Gamma$, then to resort to the above interpolation procedure. The most natural way to identify b_p (resp. b_q) as a bilinear form over $T_{\gamma(t)}\Gamma$ is to consider its parallel transport from p (resp. q) to $\gamma(t)$ along γ.

In view of Proposition 8.4, this leads to the following procedure (see figure 8.24 for an illustration):

step 1: Diagonalize b_p in a frame $(e_1(p), e_2(p))$ which is orthonormal for the natural Riemannian structure g_p of $T_p\Gamma$. Let λ_1, λ_2 the associated eigenvalues.

step 2: Approximate the parallel transport $T(\gamma)_0^t(e_1(p))$ of $e_1(p)$ from p to $\gamma(t)$ along γ using Schild's ladder's procedure (see Remark 8.8). A vector $\widetilde{e_1(p)}$ is then obtained.

step 3: Let $\widetilde{e_2(p)} \in T_{\gamma(t)}\Gamma$ such that $(\widetilde{e_1(p)}, \widetilde{e_2(p)}, n(\gamma(t))$ is a direct orthonormal frame of $\mathbb{R}^3$, and define $\widetilde{b_p}$ as the symmetric positive definite bilinear form over $T_{\gamma(t)}\Gamma$ whose eigenvalues are λ_1, λ_2, with associated eigenvectors $\widetilde{e_1(p)}, \widetilde{e_2(p)}$.

step 4: Apply the same procedure to get an approximation $\widetilde{b_q}$ of the parallel transport $T(\gamma)_1^t(b_q)$.

step 5: Interpolate $\widetilde{b_p}$ and $\widetilde{b_q}$ in the tangent plane $T_{\gamma(t)}\Gamma$ using the aforementioned procedure in the Euclidean case.

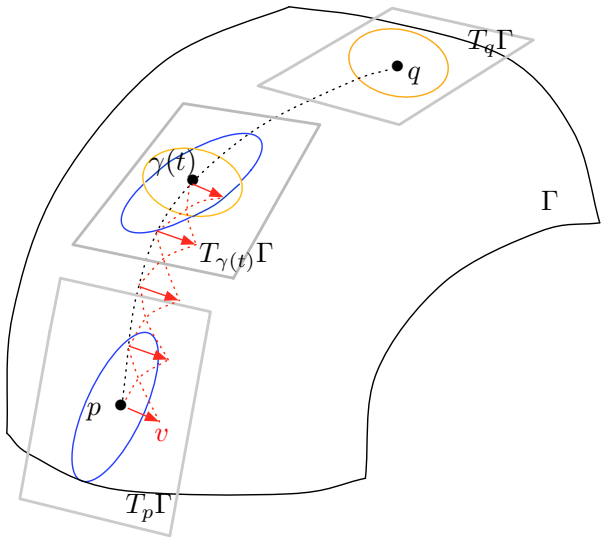


Figure 8.24: Interpolation of two symmetric bilinear forms b_p, b_q (represented by their unit ellipsoids, in blue and yellow) defined over two different tangent planes $T_p\Gamma$ and $T_q\Gamma$. An eigenvector v of b_p is represented (in red), as well as its parallel transport thanks to Schild's ladder's procedure.

8.2.5 Geometric anisotropic surface remeshing of a discrete surface

Thanks to these new ingredients, we are now in position to roll out the proposed anisotropic remeshing procedure. Obviously, this strategy is very similar to the isotropic one, detailed in section 8.1.5, except perhaps on two points, which we now describe.

8.2.5.1 Anisotropic mesh quality

As explained in section 8.1.5, the suggested remeshing procedure relies on a quality function such as (8.2) for evaluating the shape of the triangles of an approximating mesh $\mathcal{S}$ to a surface Γ: among other things, it is a very convenient way to discriminate which operation should or should not be carried out on a mesh, depending on whether it degrades or enhances the shapes of its elements (e.g. whether it produces or eliminates very stretched triangles).

However, in our attempt to obtain an anisotropic mesh, adapted to a Riemannian metric $b \in \mathrm{Sym}^2(T\Gamma^*)$, we have to modify the evaluation of the quality of triangles to take into account the anisotropy in the size prescriptions: a triangle which is very stretched, but whose stretching is aligned with the anisotropy in the local size features must be considered well-shaped, whereas an equilateral triangle for the Euclidean metric, located in an area where the size prescription is highly anisotropic must not be judged satisfactory (see [50] for a discussion over this topic).

Unfortunately, to the best of our knowledge, there is no general theory around mesh quality evaluation in the anisotropic context. In our remeshing process, we used the following quality function Q^b, which is the most natural extension of (8.2) to the anisotropic context: for any triangle $T \in \mathcal{S}$, with edges e_1, e_2, e_3,

$$Q^b(T) = \frac{\mathrm{Vol}^b(T)}{\sqrt{\left(\ell^b(e_1)\right)^2 + \left(\ell^b(e_2)\right)^2 + \left(\ell^b(e_3)\right)^2}}. \tag{8.66}$$

Contrary to the isotropic case, there is no longer any guarantee that the equilateral triangles in metric b should reach the maximum value of (8.66) (hence, there is no point in normalizing Q^b). One can only notice that such a function Q^b tends to qualify as badly-shaped those elements which are flat, or stretched *with respect to the anisotropy in the local values of b.*

8.2.5.2 Vertex relocation in the anisotropic remeshing context

In section 8.1.3.4, the vertex relocation operator was presented as an operator aimed at improving the overall quality of a mesh and was based on the following heuristic: in order to make the triangles of the ball $\mathcal{B}(p)$ of a vertex $p \in \mathcal{S}$ equilateral, one can try and move it in the direction of the centroid of $\mathcal{B}(p)$. Recall that, so as to ensure that the new position of p stays on Γ, the new position was computed on the tangent plane $T_p\Gamma$, then sent back to Γ using its local parametrization (see section 8.1.2).

Here, the same heuristic is translated in the anisotropic context but proves slightly more tedious, since the very notion of 'centre of mass' should be cast into the Riemannian context. This is the purpose of the following result, excerpted from [187].

Theorem and Definition 8.14. Let (M, g) be a Riemannian manifold; denote as $d(.,.)$ the Riemannian distance, and as ω the volume form on M. Let $p \in M$, and $r > 0$ small enough so that, in particular, the geodesic ball of center p and radius r is strongly convex (see [187] for an estimate of the admissible values of r, depending on the sectional curvatures of M near p). Let also $U \subset B_r(p)$ an open set. Define a functional J on $B_r(p)$ as:

$$\forall x \in B_r(p),\ \ J(x) = \frac{1}{2}\int_U d(x,y)^2 \omega(y). \tag{8.67}$$

Then J has a unique minimum point over $B_r(p)$, called the *Riemannian center of mass of* U.

Moreover, J is differentiable over $B_r(p)$, and its differential reads:

$$\forall x \in B_r(p),\ \ \forall v \in T_xM,\ \ dJ_x(v) = -\int_U g_x(\log_x(y), v)\omega(y). \tag{8.68}$$

Back to our setting, suppose a Riemannian metric $b \in \mathrm{Sym}^2(T\Gamma^*)$ has been defined over Γ, which accounts for instance (but not necessarily) to a local size prescription over Γ. Denote with a ^{b} superscript all the geometric features of Γ (Riemannian distance, exponential map) associated to this particular Riemannian structure.

Let $p \in \mathcal{S}$ a vertex to be relocated, $\mathcal{B}(p)$ its ball of surface triangles. Let $\pi : \mathbb{R}^3 \to T_p\Gamma$ the orthogonal projection onto $T_p\Gamma$, and $\phi : \pi\left(\mathcal{B}(p)\right) \to \Gamma$ the local parametrization of Γ from $T_p\Gamma$ (see again section 8.1.2). Contrary to what happened in the isotropic context of section 8.1.3.4, the relocation position for p can no longer be computed explicitly. Thus, we shall steadily move p in the direction of the gradient of function J defined by (8.67), for an adequate choice of set U. Let us indeed consider the Riemannian centre of mass

of the open set $U := \exp_p^b(\pi(\mathcal{B}(p))) \subset \Gamma$, assuming it is small enough to be included in a geodesic ball $B_r^b(p)$ filling the hypotheses of theorem 8.14. Of course, many other subsets $U \subset \Gamma$ could be used instead of this particular one. However, as we shall see below, this one brings convenient formulae for numerical approximation.

Using theorem 8.14, we search for a new position for p that decreases the value of the objective function J^b defined over $B_r^b(p)$ as:

$$\forall x \in B_r^b(p),\ \ J^b(x) = \frac{1}{2}\int_U d^b(x,y)^2 \omega^b(y), \tag{8.69}$$

and to this end, we can rely on the fact that:

$$\forall v \in T_p\Gamma,\ \ dJ_p^b(v) = -b_p(V,v),\ \ \text{where } V = \int_U \log_p^b(y)\,\omega^b(y) \in T_p\Gamma. \tag{8.70}$$

Hence, relocating p at position $\phi(0+\varepsilon V)$ (for a small parameter $\varepsilon > 0$) should decrease the value of our objective function (8.69). This leaves us with putting V under a form suitable for numerical computations. To this end, introducing a basis (e_1, e_2) of $T_p\Gamma$, we pull the integral expression of (8.70) back to $T_p\Gamma$:

$$V = \int_{\pi(\mathcal{B}(p))} x \sqrt{\det(b_{i,j}(x))}dx,\ \ \text{with } \forall\, i,j = 1,2,\ b_{i,j}(x) = b_{\exp_p^b(x)}\left(d(exp_p^b)_x(e_i), d(exp_p^b)_x(e_j)\right). \tag{8.71}$$

The latter formula is then decomposed over the (projected) triangles of $\pi(\mathcal{B}(p))$, and evaluated using quadrature formulae. Note that, once again, for the sake of simplicity, we chose to approximate the exponential map $\exp_p^b$ by the parametrization ϕ of Γ itself, whose analytical expression is known and can be differentiated at will.

8.2.5.3 The complete anisotropic remeshing strategy

Let us sum up the suggested strategy for anisotropic surface remeshing. As it is almost identical with the proposed strategy for isotropic surface remeshing exposed in section 8.1.5, we mostly limit ourselves to pinpoint the differences between both cases when they arise. Starting from an input triangulation $\mathcal{S}$ of an ideal surface Γ, and given the four parameters ε, h_{min}, h_{max} and h_{grad} whose meanings are the same as in section 8.1.5, we still proceed within five main steps:

step 1: *Analysis of* $\mathcal{S}$. Exactly as in the isotropic case, additional geometric information about Γ are recovered from the datum of $\mathcal{S}$, and special geometric entities (ridge edges, etc...) are identified.

step 2: *Rough mesh modifications for a good 'sampling' of the surface.* This stage as well unfolds exactly as in the isotropic framework: a new triangulation $\widetilde{\mathcal{S}_1}$ of Γ is produced, which satisfies the geometric control criterion $d^H(\widetilde{\mathcal{S}_1}, \Gamma) \leq \varepsilon$, without any other size prescription as the fact that the edges of $\widetilde{\mathcal{S}_1}$ should be neither longer than h_{max}, nor shorter than h_{min}.

Note that an initial, user-defined size prescription $m \in \mathrm{Sym}^2(T\Gamma^*)$ may be defined over Γ and stored at the vertices of the initial triangulation $\mathcal{S}$ (as discussed in section 8.2.4.1), with respect to which $\mathcal{S}$ should be eventually adapted. As in the isotropic case, this second step does not rely on the information encoded in m. However, each time a new vertex is introduced (during the edge split operation), a consistent value for m at the new vertex has to be inferred. This is achieved with the interpolation procedure described in section 8.2.4.3.

step 3: *Construction of the metric tensor field.* A bilinear symmetric map $b \in \mathrm{Sym}^2(T\Gamma^*)$ for the geometric approximation of Γ in terms of Hausdorff distance within tolerance ε is then computed as in section 8.2.3, and stored on $\widetilde{\mathcal{S}_1}$. If another Riemannian metric m was supplied by the user, intersect m with b following Section 8.2.4.2. This metric tensor field is then graded, as detailed in Section 8.2.3.2.

step 4: *Rough mesh modifications with respect to the metric tensor field.* This step is once again very similar to its isotropic counterpart. So as to get a next triangulation $\widetilde{\mathcal{S}_2}$ of Γ, we proceed as in step 2, except that, henceforth lengths are measured by means of function ℓ^b (see formula (8.15)). Aiming at getting a new triangulation whose edges have length 1, we choose *rough* bounds, outside which no edge length should lie. Moreover, we take much more caution about mesh anisotropic quality (see formula (8.66)): edge collapses are presently prevented when they degrade two much the overall quality of the triangulation, and edge swaps are carried out provided they help improving the anisotropic quality of the mesh (and possibly rejected if they degrade too much the geometric approximation of Γ).

step 5: *Fine mesh modifications with respect to the metric tensor field.* From the 'pretty good' triangulation $\widetilde{\mathcal{S}_2}$, we perform delicately driven operations so as to get the final triangulation $\widetilde{\mathcal{S}}$. Lengths of edges are still evaluated by means of ℓ^b, except we now impose $\widetilde{\mathcal{S}}$ should have no edge with length lying outside a sharper interval as before. We are also much stricter as far as the authorized degradation in mesh anisotropic quality entailed by our operators is concerned. We eventually use the anisotropic vertex relocation operator described in section 8.2.5.2 to help improving the overall (anisotropic) quality of the mesh.

8.2.6 Numerical examples

As a conclusion to this discussion over anisotropic remeshing of discrete surfaces, let us give several illustrations of the topics described above. For each example, details such as the total number np_i (resp np_f) of points, the average (anisotropic) quality Q_i (resp. Q_f) and the worst (anisotropic) quality wq_i (resp. wq_f) of an element of the input (resp. output) mesh, as well as the total CPU time of the computation are to be found in table 8.2.

8.2.6.1 Anisotropic remeshing of triangulated surfaces presenting anisotropic features

We start by giving several examples of the anisotropic remeshing of a discrete surface triangulation $\mathcal{S}$ whose geometry presents anisotropic features (e.g. cylindrical parts, etc...) into a new triangulation $\widetilde{\mathcal{S}}$ which is a close geometric approximation of the underlying ideal surface Γ, and is adapted to the geometry-based Riemannian structure on Γ defined at Section 8.2.3. The first example concerns the same surface as in Section 8.1.6, namely, the *model02* surface. Figure 8.25 shows several illustrations of anisotropic remeshing of this triangulated surface, where different values of the parameters h_{min}, h_{max}, h_{grad}, ε have been used. Note that, in this figure, the result of (c) has been obtained using the same parameters as the one of (b), except that the gradation procedure has been activated in the latter case. This shows the great impact of this procedure, which admittedly curbs the anisotropic behavior of the elements in the mesh, for the sake of 'smoothness' in the progression of the size prescriptions.

Figure 8.26 shows two other examples of anisotropic remeshing of discrete surfaces [6]. Note that we carried out three consecutive anisotropic remeshing procedures to obtain the second result, for the initial triangulation makes it very difficult to capture at once the anisotropic behavior of the associated surface (actually, the problem mainly comes from the 'long' edges connecting two different ridge curves on the boundary, which make it difficult to properly evaluate the curvature of the ideal surface).

8.2.6.2 Anisotropic adaptation of a surface triangulation to a user-defined metric tensor field

We eventually turn to an example where the proposed strategy for anisotropic surface remeshing is used to adapt an initial triangulation to a user-defined metric tensor field. Let $\mathcal{S}$ a surface triangulation of the sphere $\Gamma \subset \mathbb{R}^3$ of center 0 and radius 4, and consider the three-dimensional metric tensor field $m : \mathbb{R}^3 \to \mathcal{S}_3(\mathbb{R})$

6. Data courtesy of `http://www.scorec.rpi.edu/` and `http://www-roc.inria.fr/gamma/download/download.php`.

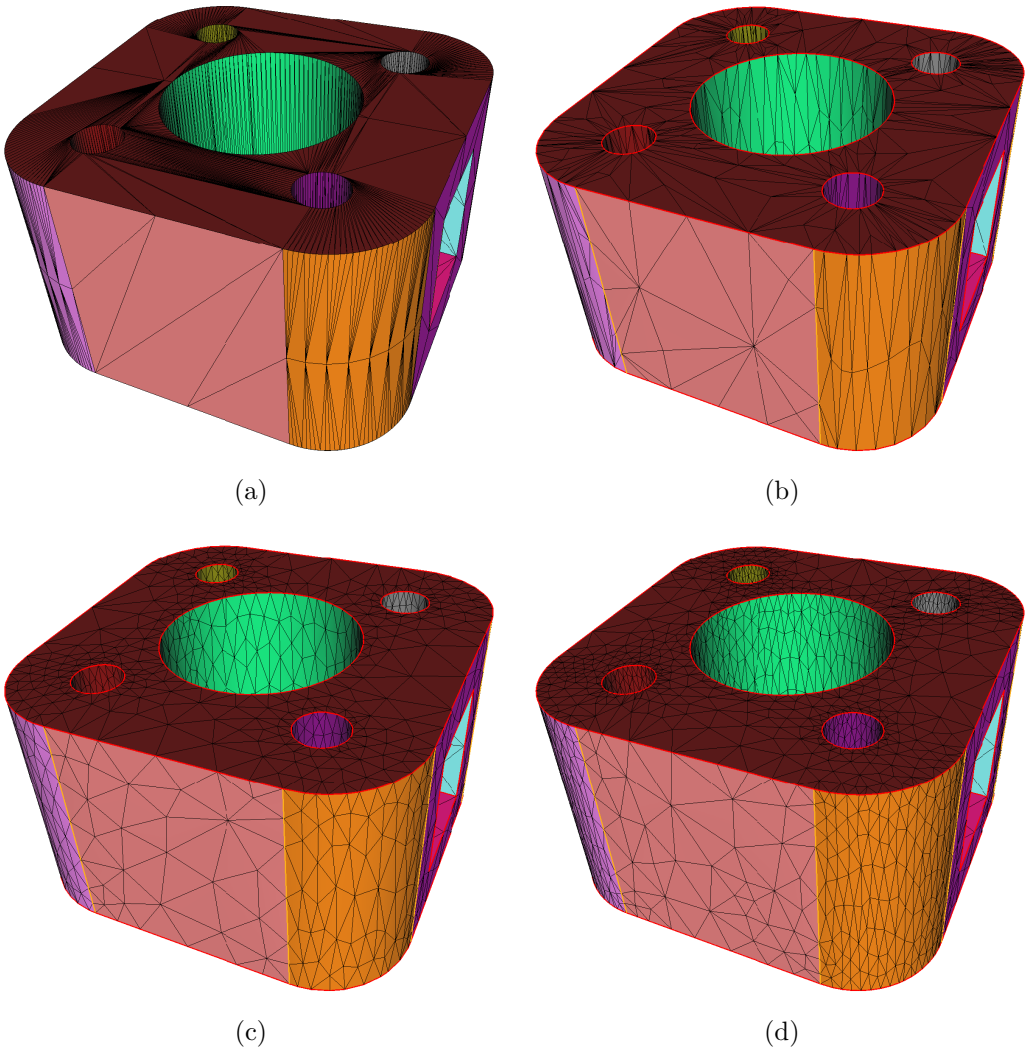


Figure 8.25: Remeshing of the *model02* model, displayed in *(a)*, enclosed in a bounding box of dimensions $0.199 \times 0.199 \times 0.12$. *(b)* Anisotropic remeshing using parameters $h_{min} = 1e^{-4}$, $h_{max} = 0.1$, $\varepsilon = 5e^{-4}$ without gradation; *(c)* anisotropic remeshing using parameters $h_{min} = 1e^{-4}$, $h_{max} = 0.1$, $\varepsilon = 5e^{-4}$ and $h_{grad} = 1.6$; *(d)* anisotropic remeshing using parameters $h_{min} = 1e^{-4}$, $h_{max} = 0.1$, $\varepsilon = 2e^{-4}$ and $h_{grad} = 1.6$.

Test case	np_i	np_f	CPU (s)	Q_i	wq_i	Q_f	wq_f
model02 (b)	1204	997	0.991	0.0768	$8e^{-6}$	0.61	0.02
model02 (c)	1204	2161	2.008	0.0768	$8e^{-6}$	0.87	0.31
model02 (d)	1204	3937	3.136	0.0768	$8e^{-6}$	0.84	0.23
pda06	6768	4652	3.969	0.78	0.019	0.88	0.033
gehaeuse	876	1814	5.04	0.41	$5.6e^{-3}$	0.85	0.11

Table 8.2: Details of the examples of section 8.2.6.

defined as:

$$\forall x \in \mathbb{R}^3,\ \ m(x) = \begin{pmatrix} \frac{\sqrt{3}}{2}\left(h + h_1\cos^2(\theta) + h_2\sin^2(\theta)\right) & \frac{\sqrt{3}}{2}\left(h_1 - h_2\right)\cos(\theta)\sin(\theta) & 0 \\ \frac{\sqrt{3}}{2}\left(h_1 - h_2\right)\cos(\theta)\sin(\theta) & \frac{\sqrt{3}}{2}\left(h + h_1\sin^2(\theta) + h_2\cos^2(\theta)\right) & 0 \\ 0 & 0 & 4 \end{pmatrix},$$

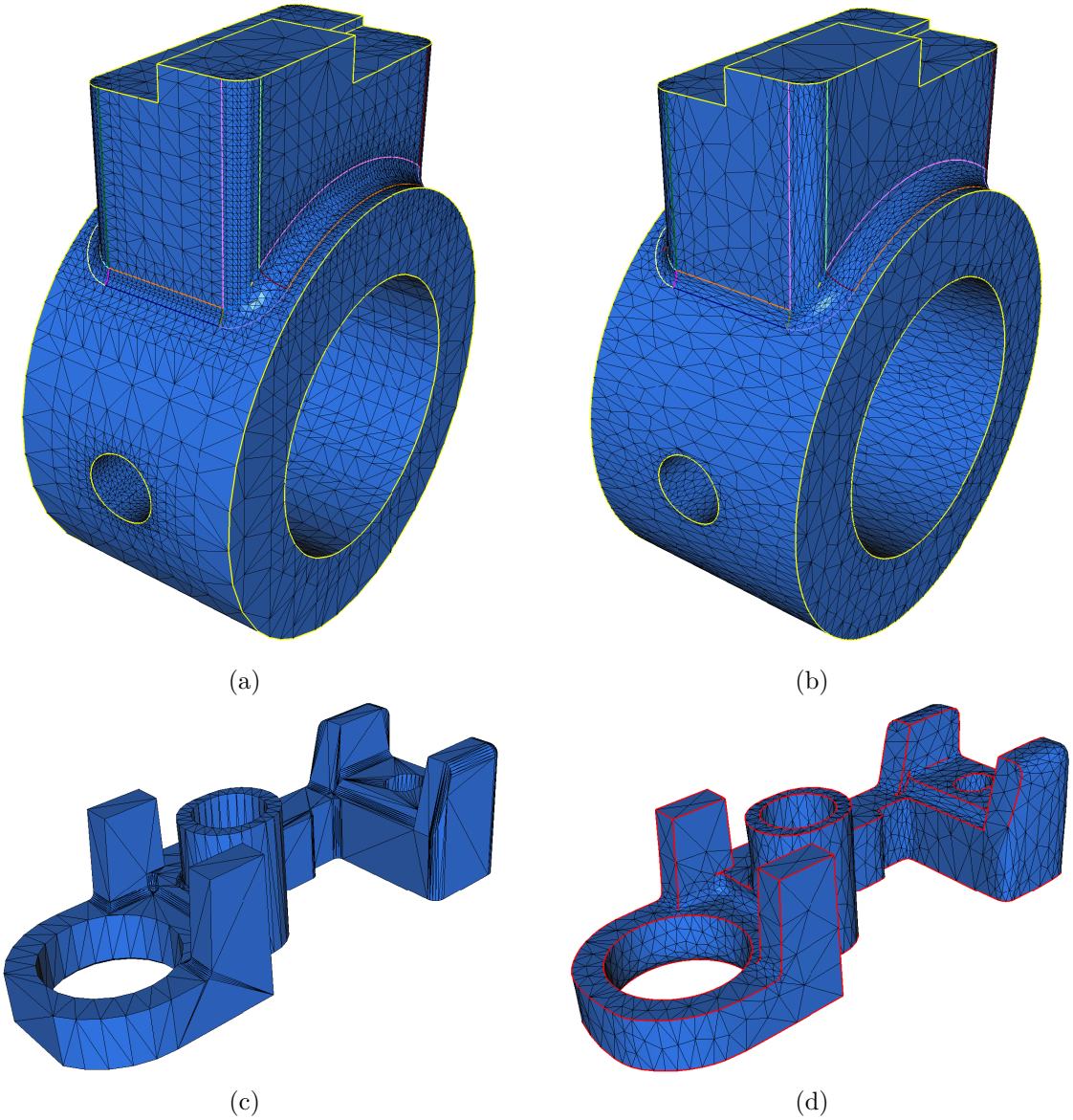


Figure 8.26: *(a-b)* Anisotropic remeshing of the *pda06* model, enclosed in a bounding box of dimensions $1.65 \times 0.86 \times 2.15$. Here we take $h_{min} = 3e^{-4}$, $h_{max} = 1$, $h_{grad} = 1.6$, and $\varepsilon = 3e^{-4}$. *(c-d)* Anisotropic remeshing of the *gehaeuse* model, enclosed in a bounding box of dimensions $0.1 \times 0.037 \times 0.029$, with $h_{min} = 1e^{-4}$, $h_{max} = 1$, $h_{grad} = 1.6$, and $\varepsilon = 2e^{-4}$.

where

$$\mu = \begin{cases} \frac{x^3 - y^2 + 2}{3x} & \text{if } x \neq 0 \\ 1 & \text{otherwise} \end{cases}, \; \theta = \operatorname{atan}\left(\frac{-2y}{3(x^2 - \mu)}\right), \; h = \left((x-1)^2 + y^2 + 0.01\right)^{-2},$$

$$\begin{cases} h_1 = \left(\min(0.2\mu^3 + 0.005, 1)\right)^{-2}, \; h_2 = \left(\min(0.2\mu^3 + 0.2, 1)\right)^{-2} & \text{if } \mu \geq 1 \\ h_1 = \left(\min(0.2\mu^2 + 0.01, 1)\right)^{-2}, \; h_2 = \left(\min(0.2\mu^2 + 0.2, 1)\right)^{-2} & \text{otherwise} \end{cases}.$$

Of course, the concrete -and interesting !- applications bring into play a metric tensor field which is not at all analyical, but for instance arises from an error analysis performed on $\mathcal{S}$. Yet, as far as the remeshing procedure only is concerned, things are just completely equivalent (the only important point being that a metric tensor field is supplied at the vertices of $\mathcal{S}$), which is why we dwell on that case. As suggested in section 8.1.6.3, the procedure is iterated five times, with the parameters: $h_{min} = 0.03$, $h_{max} = 1$, $\varepsilon = 0.05$, $h_{grad} = 1.1$ for the first two iterations, then $h_{grad} = 1.2$. The whole computation takes $74s$, for a final mesh consisting of 7556 vertices, with an average anisotropic quality of 0.89. (see Figure 8.27).

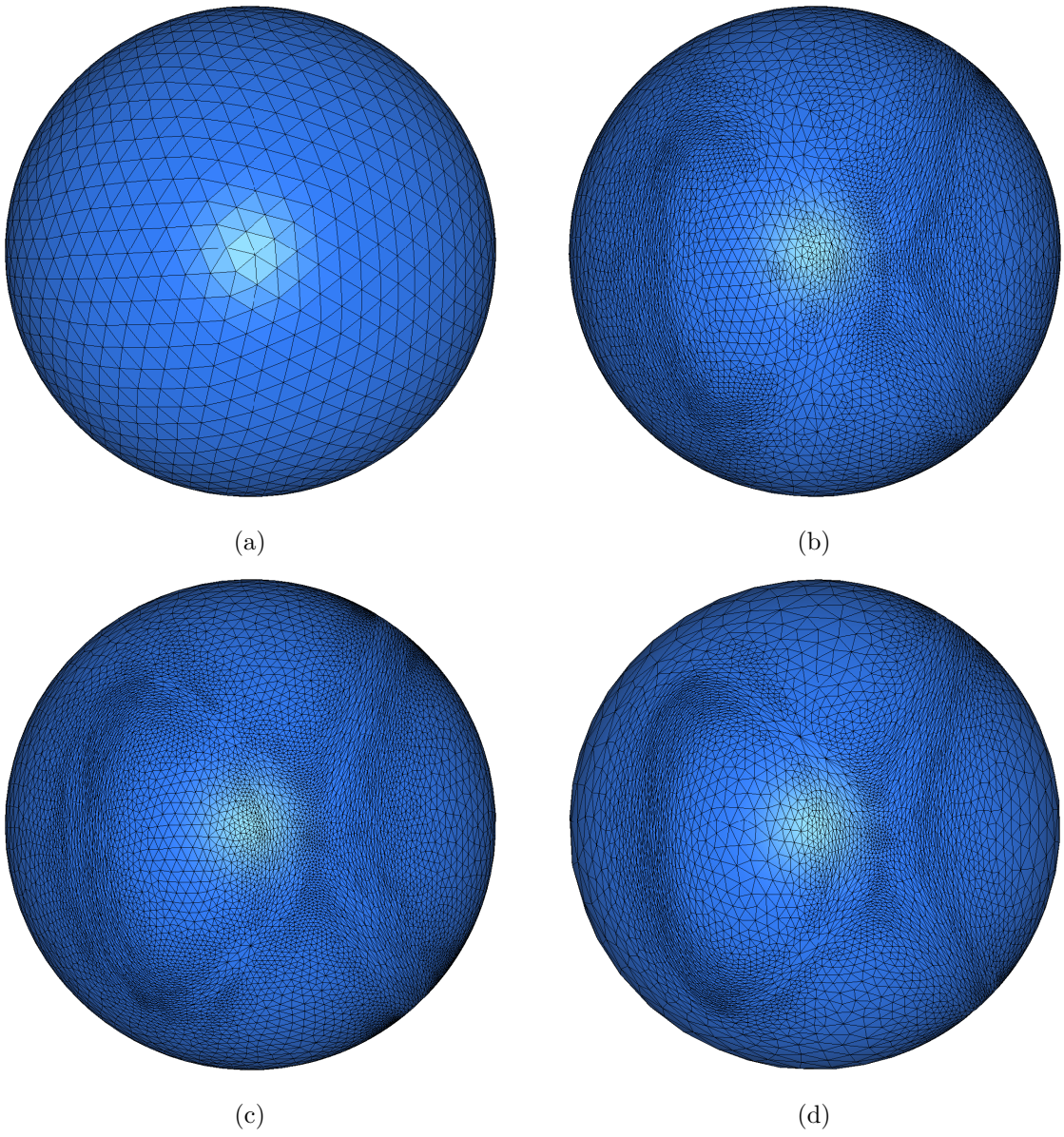


Figure 8.27: Anisotropic remeshing of a surface triangulation of a sphere with respect to a user-specified metric tensor field; *(a)* initial triangulation, *(b)* resulting triangulation after 1 iteration, *(c)* resulting triangulation after 3 iterations, and *(d)* final triangulation.

8.3 Discrete three-dimensional domain remeshing

Elaborating on the strategy for discrete surface remeshing described throughout Section 8.1, we now address the problem of isotropic three-dimensional domain remeshing. Consider a three-dimensional simplicial mesh $\mathcal{T}$, intended as an approximation of an unknown bounded *ideal domain* $\Omega \subset \mathbb{R}^3$, in the sense that:

- the associated boundary triangulation $\mathcal{S}_\mathcal{T}$ to $\mathcal{T}$ is an interpolating surface mesh of $\partial\Omega$,
- the tetrahedral volume mesh $\mathcal{T}$ is a conforming simplicial mesh.

$\mathcal{T}$ may be inappropriate as an approximation of Ω for at least two reasons:

- The surface triangulation $\mathcal{S}_\mathcal{T}$ may be a bad approximation of $\partial\Omega$ in the sense considered in Section 8.1: it may be bad as a geometric approximation of $\partial\Omega$, or ill-shaped.
- $\mathcal{T}$ may be under-sampled or over-sampled with respect to a desired size feature (dictated for instance by an error estimate associated to a mechanical computation), or of poor mesh quality. As for evaluating the quality of a tetrahedron $K \subset \mathbb{R}^3$, we retained the following function among all the possible ones:

$$Q(K) = \alpha \frac{\mathrm{Vol}(K)}{\left(\sum_{i=1}^{6} \ell(e_i)^2\right)^{\frac{3}{2}}}, \quad \alpha = 144\sqrt{3},$$

where e_i, $i = 1, ..., 6$ are the edges of K.

Note that, understandably enough, the geometric approximation of Ω by means of $\mathcal{T}$ merely amounts to the geometric approximation of $\partial\Omega$ by means of the associated surface mesh $\mathcal{S}_\mathcal{T}$. The only constraint on the volume part of $\mathcal{T}$ concerns its quality, and possibly its matching a prescribed size feature (see Figure 8.28).

As we shall see, there is actually no further theory from surface to domain remeshing, and almost all this section is devoted to describing either the three-dimensional 'volume' equivalent local mesh operators to those presented in Section 8.1.3, or the way these surface remeshing operators can be applied to situations involving the surface mesh $\mathcal{S}_\mathcal{T}$, where some tetrahedra of $\mathcal{T}$ are 'attached' to the surface configuration.

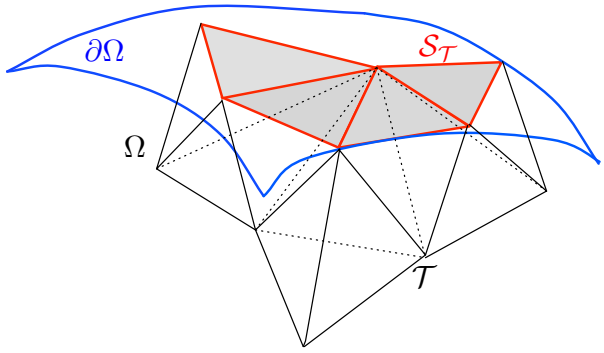


Figure 8.28: Mesh $\mathcal{T}$ (in black) associated to a domain $\Omega \subset \mathbb{R}^3$, together with the corresponding surface mesh $\mathcal{S}_\mathcal{T}$ (red lines), as an approximation of $\partial\Omega$ (in blue).

Our aim is then to devise a strategy for remeshing $\mathcal{T}$ into a mesh $\widetilde{\mathcal{T}}$ which is a nice approximation of Ω in the sense defined above. In practical terms, we will guide local remeshing operators into producing a sequence of meshes $\mathcal{T}_1, ...\mathcal{T}_n$ which are increasingly closer to a convenient approximation of Ω.

As in the case of surface remeshing, the ideal domain Ω is unknown, and must be approximated from the data at hand. More precisely, we only need to describe the boundary $\partial\Omega$, from the knowledge of the surface mesh $\mathcal{S}_\mathcal{T}$ associated to $\mathcal{T}$. To achieve this, we retain the same approach - and the same biases -

as for surface remeshing. A preprocessing stage is aimed at identifying additional geometric features of $\partial\Omega$ from the rough datum of $\mathcal{T}$: singular points, ridge edges, etc... as well as geometric information associated to $\partial\Omega$: normal vectors at regular vertices $x \in \mathcal{S}_\mathcal{T}$ are approximated, tangent vectors at ridge vertices, etc... This step unfolds literally as described in section 8.1.1.

Then, as described in section 8.1.2, we suppose that each surface triangle $T \in \mathcal{S}_\mathcal{T}$ stands for a smooth portion of $\partial\Omega$: from the datum of T and the attached geometric entities, rules are set to generate a local parametrization $\sigma : T \to \partial\Omega$, under the form of a Bezier patch. This enables in particular to decide whether a triangle of $\mathcal{S}_\mathcal{T}$ lies 'too far' from its corresponding portion of $\partial\Omega$ (and then must be split), or 'close enough' (and thus may be subject to collapse), in a nutshell to drive all the local remeshing operations.

Remark 8.10. Remember that this local generation of pieces of $\partial\Omega$ from triangles of $\mathcal{S}_\mathcal{T}$ inherently relies on the abuse that we neglect the dependence of the generated local parameterizations of $\partial\Omega$ on the support triangulation - that is, the fact that during the remeshing process, meshes $\mathcal{T}_1, ..., \mathcal{T}_n$ are produced, and the inferred descriptions of $\partial\Omega$ from $\mathcal{S}_{\mathcal{T}_1}, ..., \mathcal{S}_{\mathcal{T}_n}$ may differ. Such an abuse makes it possible to talk about *the* ideal domain Ω associated to *all* the produced meshes $\mathcal{T}_1, ..., \mathcal{T}_n$.

8.3.1 Description of the local remeshing operators

In this section, we describe the local operators used in the remeshing process of $\mathcal{T}$. All these - once again rather classical [145] - operators enjoy two forms, depending on whether they are applied to a surface configuration (i.e. to a configuration 'close' to surface triangles of $\mathcal{S}_\mathcal{T}$), or to a purely internal one. The former case is very similar to the one described in section 8.1.3, and one must only ensure that the performed operations are also compatible with the volume part of $\mathcal{T}$, in the sense that no tetrahedron of $\mathcal{T}$ results invalidated from the process. The latter case proves in general easier to check (see the discussion in the Appendix).

8.3.1.1 Edge split

As explained in section 8.1.3.1, splitting an edge $pq \in \mathcal{T}$ consists in introducing a new point m in the mesh, then replacing pq by the two edges pm and mq, and updating the connectivities of $\mathcal{T}$ accordingly. In the context of domain remeshing, two cases arise:

- if the processed edge pq is a surface edge, i.e. $pq \in \mathcal{S}_\mathcal{T}$, let $\gamma : [0, 1] \to \partial\Omega$ the associated curve to pq (see section 8.1.2). The new point m is then taken as $m = \gamma(\frac{1}{2})$.
- if pq is an internal edge, m is simply inserted as the midpoint of p and q.

As in the case of surface remeshing, we proceed to edge split relying on patterns: first, all the edges that should be split are identified, and all the points to be inserted are created. Then, patterns are used on each tetrahedron of $\mathcal{T}$ so as to split all such edges of $\mathcal{T}$ at the same time, inserting the created points.

This step slightly differs from its counterpart in section 8.1.3.1 insofar as now, in very degenerate configurations, inserting a new point m while splitting a boundary edge pq of $\mathcal{T}$ may invalidate some of the elements $K \in \mathcal{T}$ of the shell of pq (see for instance the very exaggerated situation of figure 8.29)). Hence, all the elements created during this step must be checked before being created, and some edge splits may possibly be prevented.

8.3.1.2 Edge collapse

Remember that collapsing an edge $pq \in \mathcal{T}$ consists in merging its two endpoints into a single one: say p is collapsed onto q for simplicity. The elements of the shell $\mathcal{S}h(pq)$ disappear in the process, and all the other tetrahedra $K \in \mathcal{T}$ which had p as a vertex (i.e. $K \in \mathcal{B}(p)$), now have q instead (see Figure 8.30).

As evoked in section 8.1.3.2, the edge collapse operator ought to be driven carefully, and several checks are in order, depending on the nature of the processed edge:

- Some collapses are strictly forbidden, e.g. a point p should not be collapsed onto another point q (along edge pq) if p belongs to the associated surface mesh to $\mathcal{T}$ (i.e. $p \in \mathcal{S}_\mathcal{T}$) and q does not, or if q does and

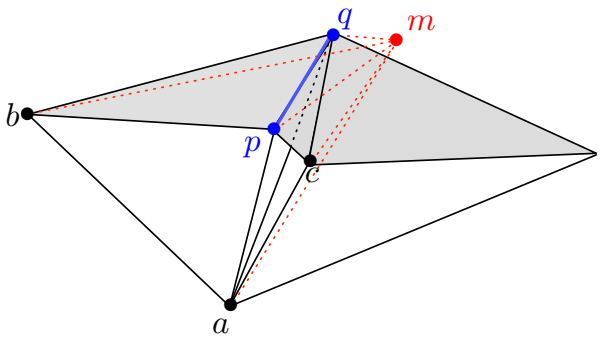


Figure 8.29: Split of boundary edge pq: point m is inserted. Tetrahedron $abpq$ is split into the two valid tetrahedra $abpm$ and $abqm$, and $acqp$ is split into the two *invalid* (flipped) tetrahedra $acmp$ and $acqm$.

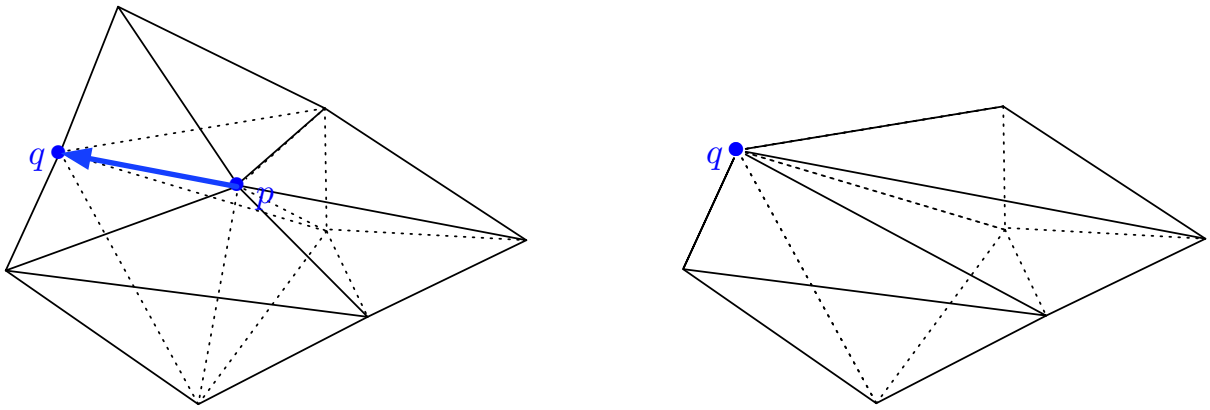


Figure 8.30: Collapse of an internal edge $pq \in \mathcal{T}$: elements of the shell $\mathcal{S}h(pq)$ disappear, while vertex p is replaced by q in the other tetrahedra $K \in \mathcal{B}(p)$ (not all the elements of $\mathcal{B}(p)$ have been represented); *(left)* configuration before collapse, *(right)* configuration after collapse.

the edge pq is not a surface edge (i.e. $pq \notin \mathcal{S}_\mathcal{T}$). See also the discussion in section 8.1.3.2 as to how the collapse operator must behave with respect to geometric entities of $\mathcal{S}_\mathcal{T}$.

- If $pq \in \mathcal{S}_\mathcal{T}$, the same checks as in the case of surface meshes have to be performed, to avoid invalidating or 'folding' $\mathcal{S}_\mathcal{T}$ in the process. Besides, additional checks have to be performed on the support tetrahedra $K \in \mathcal{T}$ to the concerned surface triangles $T \in \mathcal{S}_\mathcal{T}$, so that they do not result invalidated in the process.
- If pq is an internal edge, one only needs to check that the modified elements $K \in \mathcal{T}$ during the operation (which are those of $\mathcal{B}(p) \setminus \mathcal{S}h(pq)$) are not invalidated (i.e. inverted) in the process.

8.3.1.3 Edge swap

Like the edge split and edge collapse operators, edge swap should behave rather differently depending on whether it is applied to a surface edge or an internal one.

- If the processed edge pq lies on $\mathcal{S}_\mathcal{T}$, introducing $T_1 = pqa$, $T_2 = pqb \subset \mathcal{S}_\mathcal{T}$ the two surface triangles sharing pq, the situation is very similar to the one described in section 8.1.3.3: there is only one possibility for swapping pq, namely replacing it by ab and updating the connectivities of $\mathcal{T}$ accordingly

(see again figure 8.8). The same checks as in the surface case must be performed, so that the operation is not inconsistent with the geometry of $\partial\Omega$, and does not yield a 'folded' configuration. What's more, the validity of the affected tetrahedra of $\mathcal{T}$ has to be checked.

– Swapping an internal edge pq is more combinatorial [114, 149, 150]. In this case, the enumeration of the vertices of the elements $K \in \mathcal{S}h(pq)$ which are neither p, nor q defines an oriented pseudo-polygon $a_1...a_n$ (see figure 8.31).

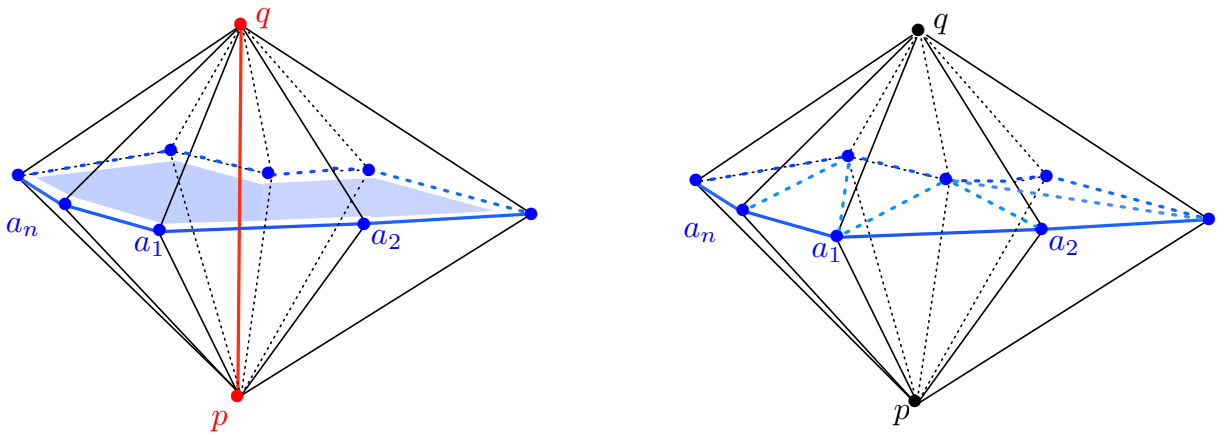


Figure 8.31: Swap of an internal edge pq. *(left)* The pseudo-polygon associated to the shell $\mathcal{S}h(pq)$ is enumerated (in light blue), and must be triangulated before being reconnected to p and q *right)*: one of the many possibilities.

This pseudo-polygon is then triangulated, and each resulting triangulated face is connected to p and q to provide a new tetrahedralization of the area once occupied by $\mathcal{S}h(pq)$. Of course, the resulting configuration must be checked (i.e. simulated) before being created, so that no element is made invalid during the process. Unfortunately, there are many ways for triangulating $a_1...a_n$, and the number of configurations grows dramatically with the number n of vertices of the pseudo-polygon. More accurately, one can show the number of such triangulations is the *Catalan number* of order n, C_n, defined as:

$$C_n = \binom{2n}{n} - \binom{2n}{n+1}.$$

Of course, all these triangulations may not correspond to a valid remeshing of $\mathcal{S}h(pq)$, but in view of this strategy, swapping pq would require to investigate all these configurations (and in particular to implement them all), until a valid one is reached, which is extremely tedious in terms of programming effort (see [114]).

For this reason, we adopted a somewhat different approach, less general yet much easier to implement. The swap of edge pq is achieved within two steps (see figure 8.32):

step 1: pq is split at its midpoint m, using the split operator of section 8.3.1.1. All the connections ma_i, $i = 1, ..., n$ are created in the process.

step 2: The introduced point m is collapsed onto one of the a_i with the operator described in section 8.3.1.2: each one of the collapses of edges $ma_1, ..., ma_n$ is tested in turn, and the first valid operation is carried out.

The latter procedure is less general than the former insofar as only n configurations associated to different triangulations of the pseudo-polygon can be reached; yet it is by far easier to implement, and we found it sufficient to meet our needs.

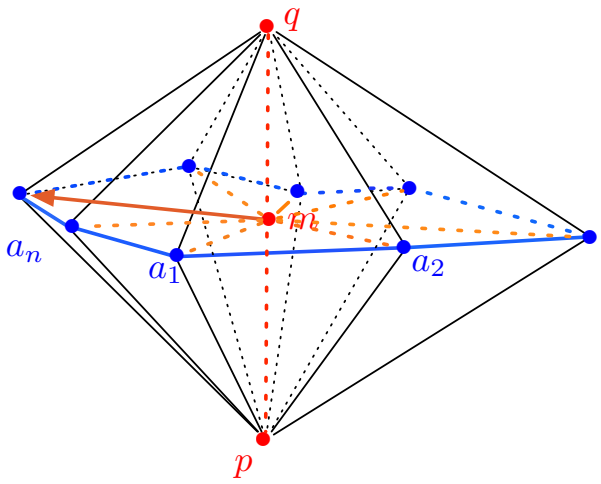


Figure 8.32: Swap of pq, introducing its midpoint m in the mesh, then collapsing it on one of the vertices of the pseudo-polygon associated to $\mathcal{S}h(pq)$.

8.3.1.4 Node relocation

This last operator is the straightforward generalization of the one presented in section 8.1.3.4 in the case of surface triangulations: a given vertex $p \in \mathcal{T}$ is moved to a new position $\widetilde{p}$ in order to increase the quality of the local configuration. Computing the position of $\widetilde{p}$ follows a different heuristic depending on whether p is a surface vertex, or an internal one:

- If $p \in \mathcal{S}_{\mathcal{T}}$, $\widetilde{p}$ is computed as in section 8.1.3.4: the surface ball $\mathcal{B}_{\mathcal{S}}(p)$ of p is enumerated and projected to the tangent plane $T_p\partial\Omega$. The center of mass of this projected ball is computed, and $\widetilde{p}$ is eventually taken as the corresponding point on $\partial\Omega$.
- If p is not a surface point, the ball $\mathcal{B}(p)$ of p is enumerated, and $\widetilde{p}$ is taken as its center of mass.

In both cases, the resulting configuration of the vertex relocation procedure has to be checked, so that no element ends up invalidated in the process, and that the quality of the mesh is indeed enhanced (we already alluded to the fact that more involved strategies could be thought of as for the choice of a relocation position $\widetilde{p}$ for p, but this simple one proves sufficient in our strategy).

8.3.2 Construction of a size map adapted to the geometric approximation of the ideal domain

We have now at our disposal all the tools we need to modify an input simplicial mesh $\mathcal{T}$ into a new mesh $\widetilde{\mathcal{T}}$, as well as the conditions under which each operation ought to be either carried out or prevented. Yet, we still lack a global indicator on where we should split edges, collapse edges,... in a nutshell, on the local density $\widetilde{\mathcal{T}}$ should enjoy so that

$$d^H(\mathcal{S}_{\widetilde{\mathcal{T}}}, \partial\Omega) \leq \varepsilon,$$

holds for a given tolerance parameter ε.

As explained in section 8.1.4, such an intelligence is encoded in a *size map* associated to the geometric approximation of $\partial\Omega$. More precisely, a scalar size function $h : \Omega \to \mathbb{R}$ is defined, with the meaning that, for any point $x \in \Omega$, $h(x)$ is intended as the desired size of the elements of $\mathcal{T}$ 'close' to x. This function is cooked using four parameters, namely the tolerance parameter ε over the geometric approximation of $\partial\Omega$, the minimum (resp. maximum) authorized size h_{min} (resp. h_{max}), and the gradation parameter h_{grad}.

In numerical practice, the information about h is defined and stored at the vertices of $\mathcal{T}$, then interpolated from these data whenever needed elsewhere. It is defined as follows:

- at a surface vertex $x \in \mathcal{S}_{\mathcal{T}}$, the local size feature is completely dictated by the geometric approximation of $\partial\Omega$. The same approach as in section 8.1.4 is used, and $h(x)$ is defined as:

$$h(x) = \min\left(h_{max}\ ,\ \max\left(h_{min}\ ,\ \sqrt{\frac{9\varepsilon}{2\max(|\kappa_1(x)|, |\kappa_2(x)|)}} \right) \right),$$

  where $\kappa_1(x)$ and $\kappa_2(x)$ are (approximations of) the two principal curvatures of $\partial\Omega$ at x.
- at an internal point x, no particular size feature seems legitimate out of the maximal authorized size $h(x) = h_{max}$ for an edge of the resulting mesh $\widetilde{\mathcal{T}}$, for there is no constraint from the geometry of Ω on such an area.

Remark 8.11. As in the case of discrete surface remeshing, we may very well devise a size map $\widetilde{h}$ on Ω which at the same time takes into account the size prescription stemming from the geometric approximation of Ω (encoded in the size map $h : \Omega \to \mathbb{R}$ defined above) and a user-defined size prescription (e.g. arising from an a posteriori error analysis associated to a numerical method performed on the initial mesh $\mathcal{T}$), encoded as another size map $m : \Omega \to \mathbb{R}$. To this end, it is enough to define:

$$\forall x \in \Omega,\ \widetilde{h}(x) = \min(h(x), m(x)).$$

Eventually, a *gradation* of the obtained size map h has to be performed, so that the corresponding local size feature varies 'smoothly' from one vertex of the mesh to the surrounding ones. Recall from section 8.1.4.5 that this step is crucial for ensuring that a mesh agreeing with size map h in the sense that $\ell_h(pq) \approx 1$ for each edge $pq \in \mathcal{T}$ will be well-shaped.

From the datum of $\mathcal{T}$, and the size map $h : \Omega \to \mathbb{R}$, stored at the vertices of $\mathcal{T}$, the gradation of h is carried out exactly as in section 8.1.4.5: starting from its *lowest values*, h is modified in such a way as, eventually, it should observe:

$$\forall \text{ edge } pq \in \mathcal{T},\ \frac{|h(q) - h(p)|}{|q - p|} \leq h_{grad}.$$

8.3.3 The complete remeshing strategy

Now that we have described how the operations and controls introduced in the case of discrete surfaces are extended to the case of discrete domains, we are in position to detail how the general remeshing strategy we used then is extended to the present case. Before doing so, here are several general comments or observations based on what we experienced trying to devise this algorithm:

- First and foremost, it turned out that tetrahedra are generally much more prone to degenerating as (surface) triangles, meaning that within very few operations, a well-shaped tetrahedron may end up nearly flat, unless the quality degeneracy of elements is explicitly controlled and prevented in the course of each operation. For this reason, the operators presented in the previous section are more severely constrained in terms of the degradation in quality they should be authorized to provoke as their surface counterparts of section 8.1.3.
- As a consequence, the whole process is far more constrained in the case of domain remeshing as in the case of surface remeshing, meaning that operations are more often prevented. Actually, talking about the hereafter presented domain remeshing strategy as a mere extension of the surface remeshing one is a bit unfair: the first strategy we devised in the case of surface remeshing was very different from the one presented in section 8.1.5, but its extension to domain remeshing brought terrible results in terms of elements' qualities. Hence, it was when we succeeded in finding a convenient strategy in terms of combination of operators, which seemed to deliver good enough quality meshes, that we modified the surface remeshing strategy so that both of them are as similar as possible.

- It appeared that the edge swap operation has a significantly different impact between the surface and domain cases: when dealing with surface remeshing, it helped noticeably to increase the overall quality of the elements of the mesh, but well-shaped meshes can still be obtained without using edge swap; on the contrary, in the case of domain remeshing, we never obtained any good-quality mesh without the (massive) use of edge swap (for surface as for internal edges); see the example in Section 8.3.4.
- On a different note, we thought it better to proceed both the surface and internal parts of $\mathcal{T}$ at the same time. This was motivated by the observation that a dramatically ill-shaped mesh $\mathcal{T}$ associated to an ideal domain $\partial\Omega$ may present a very well-shaped surface part $\mathcal{S}_{\mathcal{T}}$.

Now, starting from an initial simplicial mesh $\mathcal{T}$, with associated surface triangulation $\mathcal{S}_{\mathcal{T}}$, associated to an ideal domain Ω, and given the four parameters ε, h_{min}, h_{max} and h_{grad}, with the same meaning as in section 8.3.2, the proposed remeshing algorithm reads as follows:

step 1: *Analysis of* $\mathcal{T}$. Exactly as in section 8.1.5, additional geometric information about $\partial\Omega$ are extracted from $\mathcal{S}_{\mathcal{T}}$, and special geometric entities (ridge edges, etc...) are identified on $\mathcal{S}_{\mathcal{T}}$.

step 2: *Rough mesh modifications for a good 'sampling' of the surface.* This stage as well unfolds as in the one described in section 8.1.5: a new mesh $\widetilde{\mathcal{T}_1}$ of Ω is produced, which satisfies the sought control $d^H(\mathcal{S}_{\widetilde{\mathcal{T}_1}}, \partial\Omega) \leq \varepsilon$, without any other size prescription as the fact that the edges of $\widetilde{\mathcal{T}_1}$ should be neither longer than h_{max}, nor shorter than h_{min}. More specifically, from $\mathcal{T}$, we proceed within several iterations of the form:

- Identify all the edges which belong to the surface part of the mesh, and that should be split, then split them (relying on patterns).
- Identify all the internal edges of the mesh that should be split, then split them (still relying on patterns).
- Travel all the edges of the mesh, and collapse all the ones that should, and can be collapsed (remember that the checks in order are not the same depending on whether the processed edge is a surface edge or not).
- Travel all the edges of the mesh, and swap all the ones that should, and can be swapped (same remark as above).

Note that the splitting operation has been divided into two steps: the first one processes surface edges, while the second one only concerns internal ones. This is actually nothing but a technicality aimed at limiting the number of splitting patterns (thus the programming effort), and it does not challenge the commitment to treat the surface and internal parts of the mesh at the same time.

step 3: *Construction of the metric size map.* As in section 8.1.5, a size map $h : \Omega \to \mathbb{R}$ dedicated to the geometric approximation of Ω within tolerance ε in terms of Hausdorff distance is computed, along the lines of section 8.3.2.

step 4: *Rough mesh modifications with respect to the metric size map.* This step is once again very similar to its counterpart in section 8.1.5. So as to get a next triangulation $\widetilde{\mathcal{T}_2}$ of $\mathcal{S}$, we proceed as in step 2, except that, henceforth, we rely on length measurements with respect to the size map h. Aiming at getting a new triangulation whose edges have length 1, we impose that all the edges of the mesh should lie between *rough* bounds around the target size 1. As explained above, we pay close attention to the mesh quality: edge collapses are presently prevented when they degrade two much the overall quality of the triangulation, and edge swaps are now carried out provided they help improving the anisotropic quality of the mesh.

step 5: *Fine mesh modifications with respect to the metric size map.* From the 'pretty good' triangulation $\widetilde{\mathcal{T}_2}$, we perform delicately driven operations so as to get the final triangulation $\widetilde{\mathcal{T}}$. Lengths of edges are still evaluated with respect to h, except we now impose $\widetilde{\mathcal{T}}$ should have no edge with length lying outside a sharper interval as before around 1. We are also even stricter as far as the authorized degradation in mesh quality entailed by our operators is concerned. We eventually use the vertex relocation operator described in section 8.1.3.4 to help improving the quality of the mesh.

8.3.4 Numerical examples

As a support to the previous developments, we end this section showing some numerical illustrations of discrete domain remeshing. Details about the proposed examples - that is, the number np_i (resp np_f) of points, the average quality Q_i (resp. Q_f) and the worst quality wq_i (resp. wq_f) of an element of the input (resp. output) mesh, and the total CPU time of the computation -are reported in table 8.3.

8.3.4.1 High-quality remeshing of smooth domains or mechanical parts

Figures 8.33, 8.34 [7] and 8.35 [8], give a first view of the behavior of the proposed strategy for discrete domain remeshing. In each of these examples, a very ill-shaped three-dimensional mesh is obtained from an initial surface triangulation, using the Delaunay meshing algorithm Ghs3d [154] in such a way as the resulting meshes have almost no interior point. Each on of the obtained meshes is then remeshed into a well-shaped mesh resorting to the proposed approach.

In order to emphasize the huge impact of the edge swap operator in our remeshing process, we performed exactly the same test as that depicted in Figure 8.33, without using the edge swap operator. In this case, the proposed algorithm turns out unable to produce a well-shaped mesh (mainly because the collapse operator is too much constrained by very ill-shaped configurations): after a computation which lasts 100.65s, the final mesh has $46,222$ vertices. Its average quality is only 0.07, and the worst quality of an element of the mesh is $2.e^{-6}$

8.3.4.2 Remeshing of a domain with respect to a user-defined size map

We present here an example which is the three-dimensional counterpart of the one presented in section 8.1.6.3, and investigate the datum of a user-specified size map $m : \Omega \to \mathbb{R}$ on the considered domain Ω. More specifically, once again, we dwell on the case where m is associated to the interpolation error of a smooth function.

Let $f : \Omega \to \mathbb{R}$ a function of class $\mathcal{C}^2$, which presents 'sharp variations' on Ω. For instance,

$$\forall p = (x, y, z) \in \Omega, \;\; f(p) = \tanh(y^3)\ \tanh((z-2)^3). \tag{8.72}$$

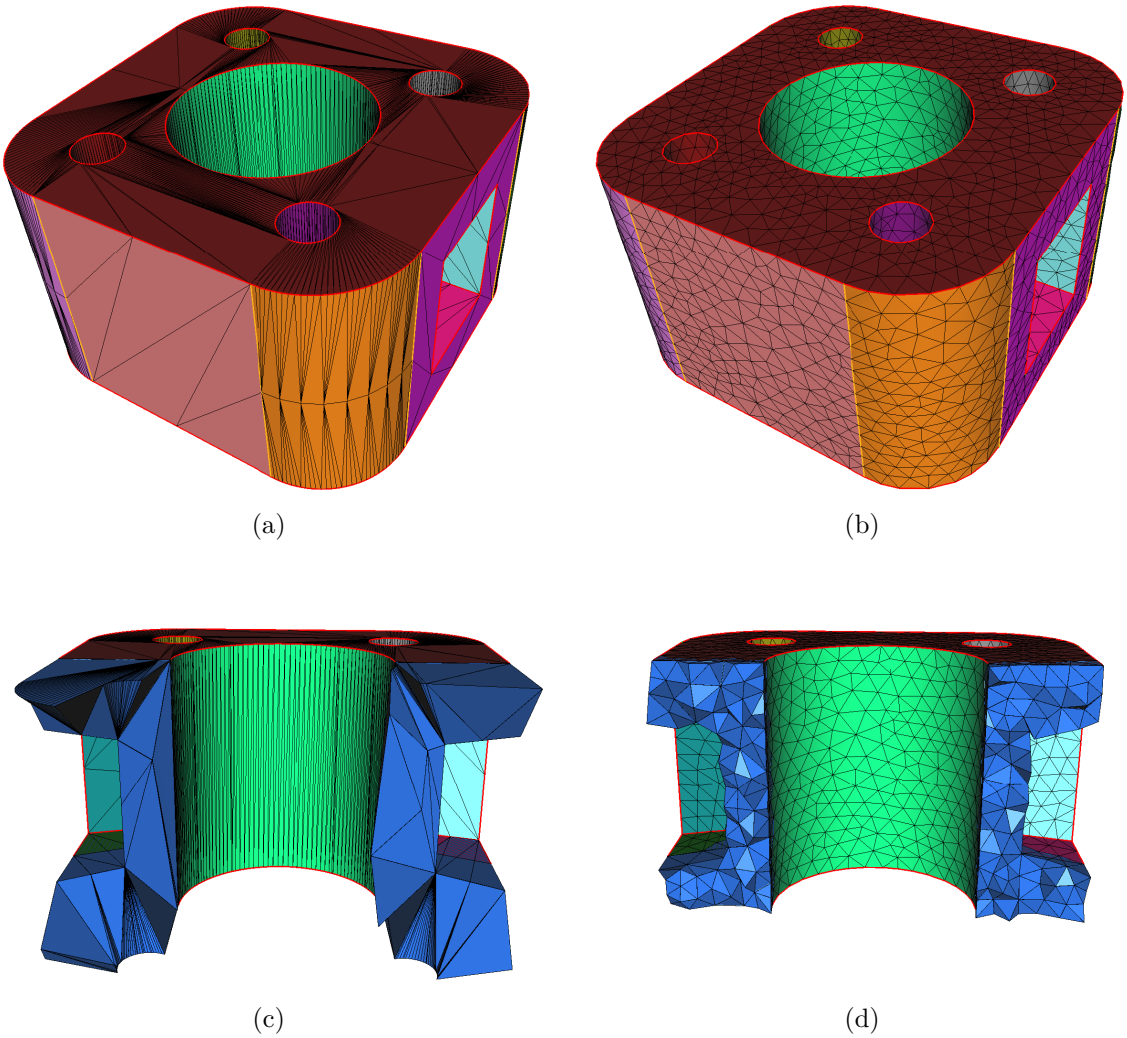
We aim at modifying the initial mesh $\mathcal{T}$ so that interpolating (linearly) f over the resulting mesh $\widetilde{\mathcal{T}}$ entails an L^∞-error controlled by a parameter ε_m, and rely once again on theorem 8.1 to cook an associated size map $m : \Omega \to \mathbb{R}$ to such a control.

Figure 8.36 displays the resulting mesh after 4 remeshing procedures for adaptation to the interpolation of f - starting from a rather coarse triangulation $\mathcal{T}$ of the domain Ω depicted on the figure, with parameter $\varepsilon_m = 0.01$. The parameters chosen for remeshing are (for each one of the 4 steps): $h_{min} = 0.2$, $h_{max} = 3$, $h_{grad} = 1.4$, and $\varepsilon = 0.3$. The whole computation took 2 minutes and 46s, and the final mesh enjoys 73701 vertices, for an average quality of 0.77.

Test case	np_i	np_f	CPU (s)	Q_i	wq_i	Q_f	wq_f
model02	1254	8424	9.403	0.0479	$1e^{-6}$	0.7737	0.19
cog	3426	11665	14.716	0.228	$1.7e^{-5}$	0.7388	0.01
pbscu	25393	64905	154	0.1154	$1.5e^{-5}$	0.776	0.02

Table 8.3: Details on the examples of section 8.3.4.

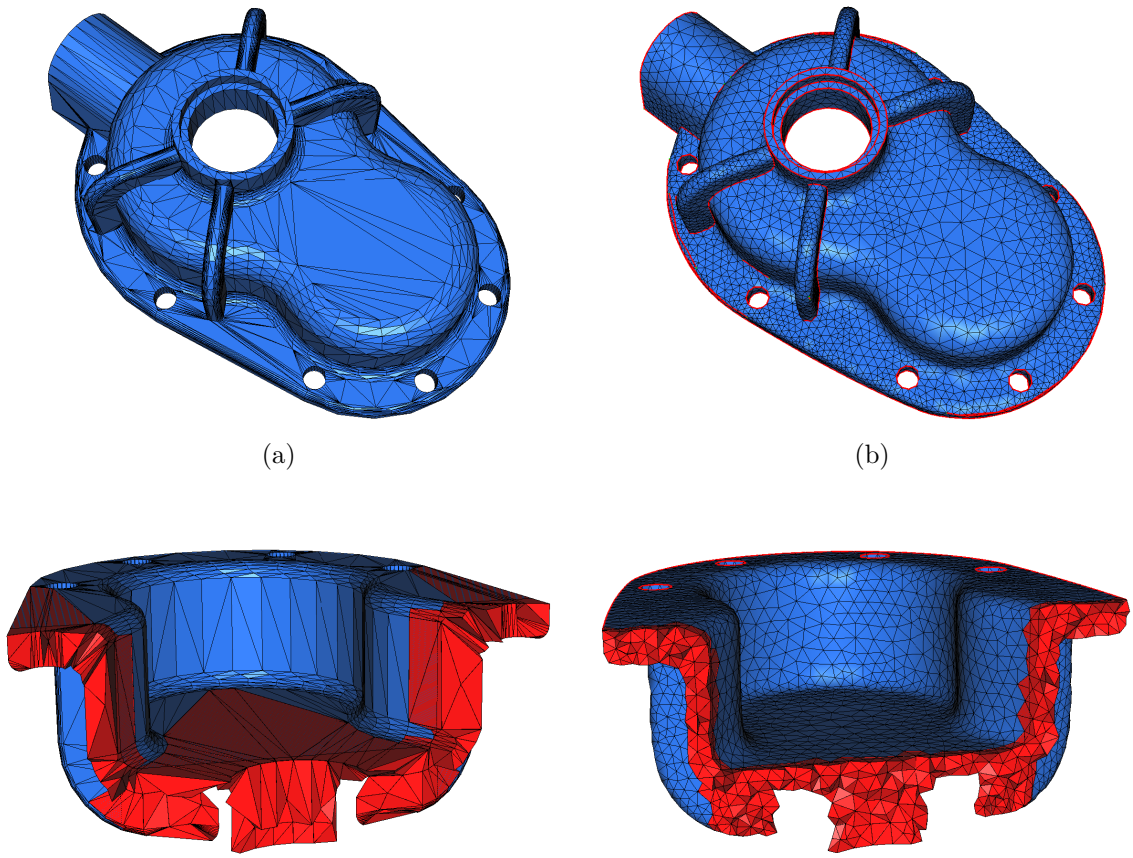
7. Data courtesy of the INRIA mesh repository `http://www-roc.inria.fr/gamma/download/download.php`.
8. Data courtesy of `http://www.archive3d.net`.



(a) (b)

(c) (d)

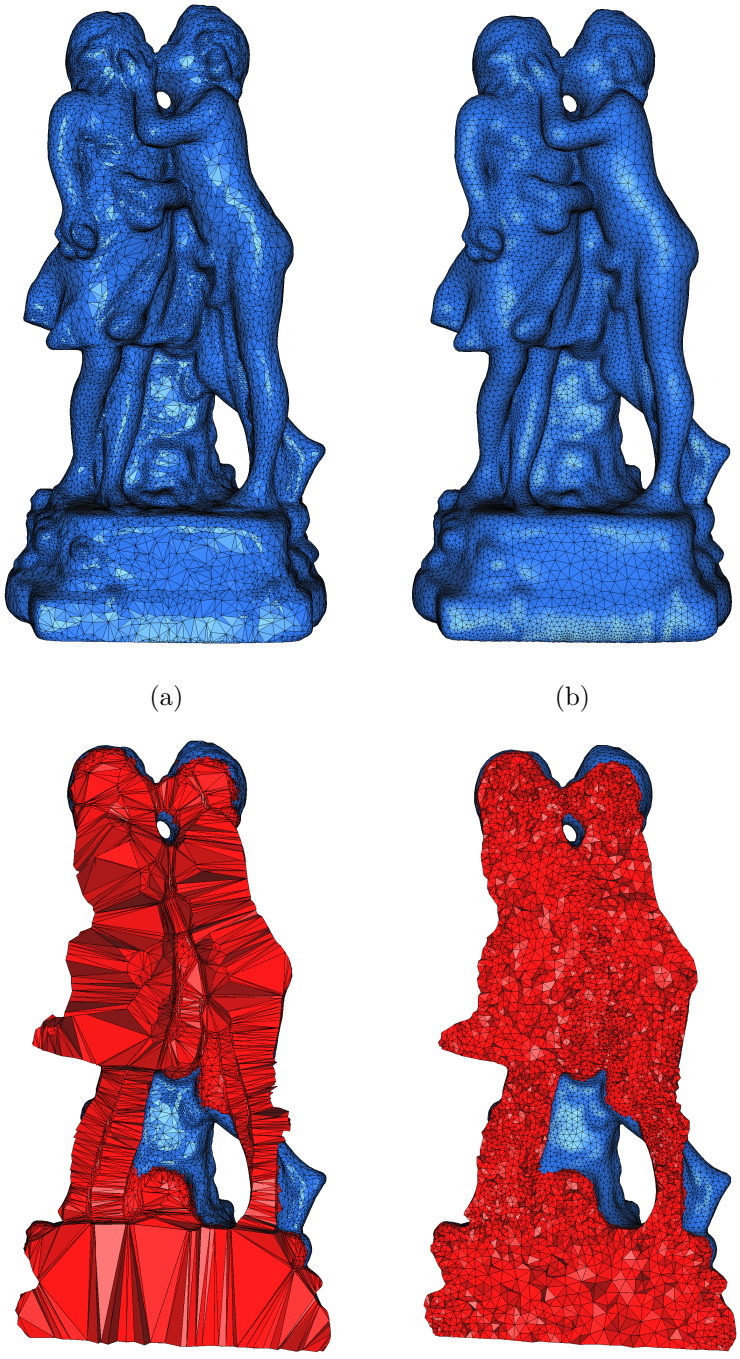
Figure 8.33: Remeshing of the *model02* model, enclosed in a box of dimensions $0.199 \times 0.199 \times 0.12$. The parameters used for the computation are: $h_{min} = 0.001$, $h_{max} = 0.1$, $h_{grad} = 1.2$ and $\varepsilon = 0.001$. *(a)-(c)* Initial mesh $\mathcal{T}$; *(b)-(d)* final result $\widetilde{\mathcal{T}}$.



(a) (b)

(c) (d)

Figure 8.34: Remeshing of the *cog* model, enclosed in a box of dimensions $0.1 \times 0.064 \times 0.028$. The parameters used for the computation are: $h_{min} = 0.0003$, $h_{max} = 0.1$, $h_{grad} = 1.2$ and $\varepsilon = 0.0003$. *(a)-(c)* Initial mesh $\mathcal{T}$; *(b)-(d)* final result $\widetilde{\mathcal{T}}$.



(a) (b)

(c) (d)

Figure 8.35: Remeshing of the *pbscu* model, enclosed in a box of dimensions $304 \times 574 \times 190$. The parameters used for the computation are: $h_{min} = 1$, $h_{max} = 50$, $h_{grad} = 1.2$ and $\varepsilon = 1$. *(a)-(c)* Initial mesh $\mathcal{T}$; *(b)-(d)* final result $\widetilde{\mathcal{T}}$.

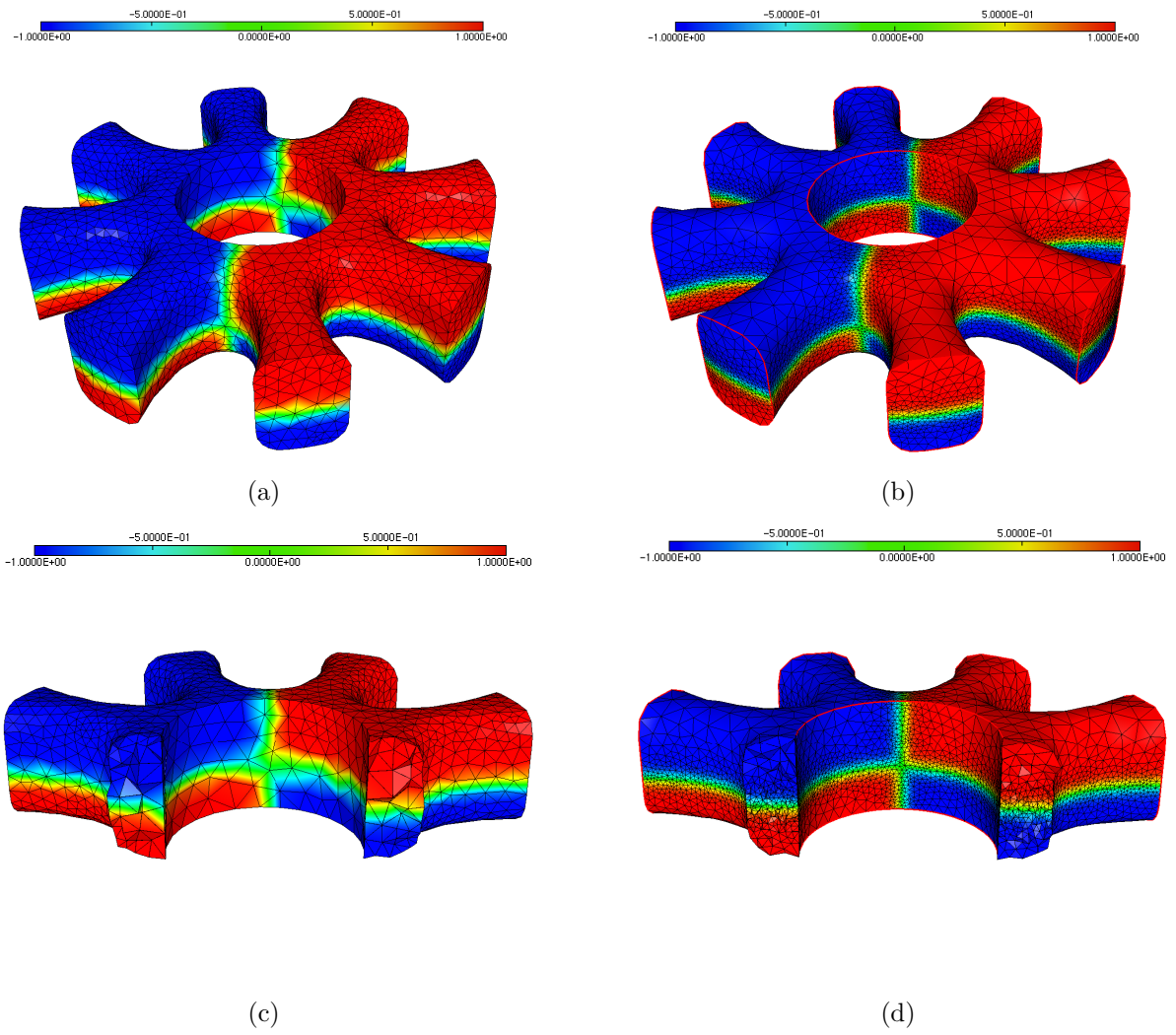


Figure 8.36: Remeshing with respect to a size map devised for the control of the interpolation error of function f defined by (8.72). *(a)* The initial mesh $\mathcal{T}$, together with the isolines of f, *(c)* a cut in the mesh; *(b)* the resulting mesh $\widetilde{\mathcal{T}}$, with the isolines of f, *(d)* a cut in $\widetilde{\mathcal{T}}$.

8.4 Implicit domain meshing and applications

In this section, we go on with our investigation on discrete domain remeshing, from a somewhat different perspective. In section 8.3, we focused on remeshing domains $\Omega \subset \mathbb{R}^3$ supplied by means of an initial simplicial mesh. Here we look into the case when an ideal domain is described as the negative subdomain of a scalar function defined over the whole space.

To set ideas, consider a bounded 'ideal' domain $\Omega \subset \mathbb{R}^3$, supposed to be *at least of class* $\mathcal{C}^2$. Suppose Ω is known via an associated implicit function $\phi : \mathbb{R}^3 \to \mathbb{R}$, i.e. the following relations hold (see Figure 8.37 for an illustration):

$$\forall x \in \mathbb{R}^3, \quad \begin{cases} \phi(x) < 0 & \text{if} \quad x \in \Omega \\ \phi(x) = 0 & \text{if} \quad x \in \partial\Omega \\ \phi(x) > 0 & \text{if} \quad x \in^c \overline{\Omega} \end{cases} .$$

Furthermore, let us assume that ϕ is at least of class $\mathcal{C}^2$, with the property that:

$$\forall x \in \partial\Omega, \ \ \nabla\phi(x) \neq 0.$$

In numerical practice, we are given a simplicial mesh $\mathcal{T}$ of a 'large' computational domain $D \subset \mathbb{R}^3$ (e.g. a box), which is the support of a numerical approximation $\phi_{\mathcal{T}}$ of ϕ. For the sake of simplicity, in this note we will only consider the case when $\phi_{\mathcal{T}}$ is a $\mathbb{P}^1$ Lagrange finite element function over $\mathcal{T}$, i.e. for any $K \in \mathcal{T}$, the restriction $\phi_{\mathcal{T}}|_K$ is affine. Of course, many other choices are relevant as regards the approximation space for functions, without any dramatic change in the forthcoming developments.

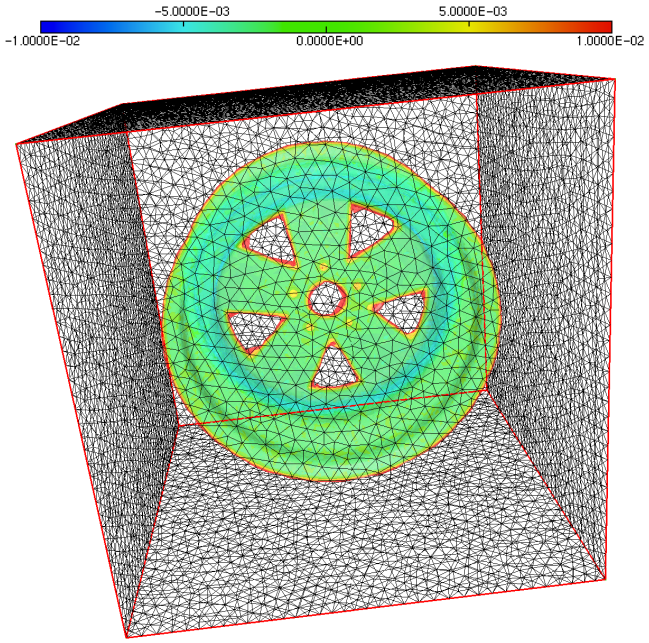


Figure 8.37: Isosurfaces of a function $\phi_{\mathcal{T}}$ defined as a piecewise affine function on a mesh $\mathcal{T}$ of a box D.

Our problem is now to obtain from $\mathcal{T}$ and $\phi_{\mathcal{T}}$ a well-shaped mesh of Ω (actually of $D \cap \Omega$; we will come back to this point later on), which is a good geometric approximation of Ω (of $D \cap \Omega$). The method proposed

to achieve this goal is actually a rather straightforward extension of the remeshing algorithm described in section 8.3, up to the addition of one - possibly two - new ingredients. Indeed, it proceeds within two main steps (see the two-dimensional figure 8.38 for an example):

1. The 0 level set of $\phi_{\mathcal{T}}$ - say $\mathcal{S}_{\phi_{\mathcal{T}}} := \{x \in D,\ \phi_{\mathcal{T}}(x) = 0\}$ - is explicitly discretized in $\mathcal{T}$. A new mesh $\widetilde{\mathcal{T}}_1$ of D is obtained, which contains a mesh $\widetilde{\mathcal{T}}_1'$ of $D \cap \Omega$ as a submesh. As we shall see, doing so, a new type of entity in the classification of section 8.1.1 may appear.
2. The resulting mesh $\widetilde{\mathcal{T}}_1$ is modified, using the algorithm proposed in section 8.3 so that a new, closely approximating, well-shaped mesh $\widetilde{\mathcal{T}}$ of D is obtained, which contains a closely approximating, well-shaped mesh $\widetilde{\mathcal{T}}'$ of $D \cap \Omega$ as a submesh.

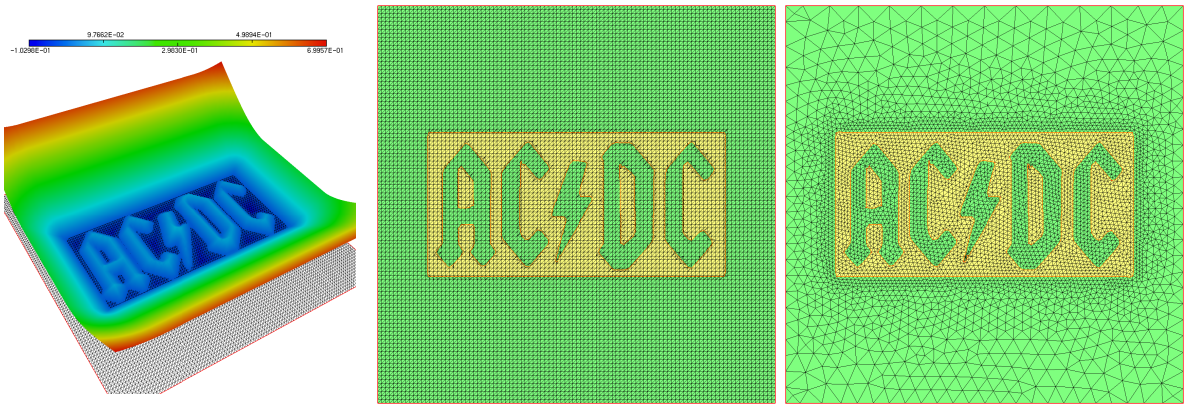


Figure 8.38: *(Left)* Level sets of a $\mathbb{P}^1$ function $\phi_{\mathcal{T}}$ on a two-dimension mesh $\mathcal{T}$ of a box D; *(middle)* the ill-shaped mesh $\widetilde{\mathcal{T}}_1$, obtained by the explicit discretization of the 0 level set of $\phi_{\mathcal{T}}$ into $\mathcal{T}$; *(right)* the final, well-shaped mesh $\widetilde{\mathcal{T}}$, containing a mesh of Ω (in yellow) as a submesh.

Hence, this method produces a bit more than a sole mesh of $D \cap \Omega$, namely a new mesh $\widetilde{T}$ of the whole computational domain D, a submesh $\widetilde{\mathcal{T}}'$ of which is a mesh of $D \cap \Omega$.

8.4.1 Explicit discretization of the 0 level set of $\phi_{\mathcal{T}}$ into $\mathcal{T}$

As explained above, the first step in obtaining a suitable mesh of $D \cap \Omega$ boils down to enforcing an explicit discretization of $D \cap \Omega$ into the mesh $\mathcal{T}$ of D. To achieve this, we rely on the approximation of $D \cap \Omega$ as the negative subdomain of $\phi_{\mathcal{T}}$, and use the following marching tetrahedra procedure [117, 141] (see also the brief description in Chapter 3):

1. Identify the set $\mathcal{K}$ of elements $K \in \mathcal{T}$ intersecting $\mathcal{S}_{\phi_{\mathcal{T}}}$: a tetrahedron $K = a_0a_1a_2a_3$ belongs to $\mathcal{K}$ if and only if there exists $i \neq j$ in $\{0, 1, 2, 3\}$ with $\phi_{\mathcal{T}}(a_i) \geq 0$, and $\phi_{\mathcal{T}}(a_j) \leq 0$.
2. For an element $K = a_0a_1a_2a_3 \in \mathcal{K}$, the intersection of $\mathcal{S}_{\phi_{\mathcal{T}}}$ with K is a plane portion of surface (see figure 8.37). Identify the edges a_ia_j of K which intersect $\mathcal{S}_{\phi_{\mathcal{T}}}$ (i.e. such that $\phi_{\mathcal{T}}(a_i)$ and $\phi_{\mathcal{T}}(a_j)$ have different signs), and compute the coordinates of the associated intersection points m_{a_i,a_j}: as $\phi_{\mathcal{T}}|_K$ is affine, one simply has:

$$m_{a_i,a_j} = (1 - \lambda)a_i + \lambda a_j, \quad \lambda = \frac{\phi_{\mathcal{T}}(a_i)}{\phi_{\mathcal{T}}(a_i) - \phi_{\mathcal{T}}(a_j)}.$$

3. Travel all the elements $K \in \mathcal{K}$, and split them, introducing the pre-computed points m_{a_i,a_j}, then using patterns (see figure 8.39). Up to permutations, there are four possible configurations, depending on the relative signs of the $\phi_{\mathcal{T}}(a_i)$.

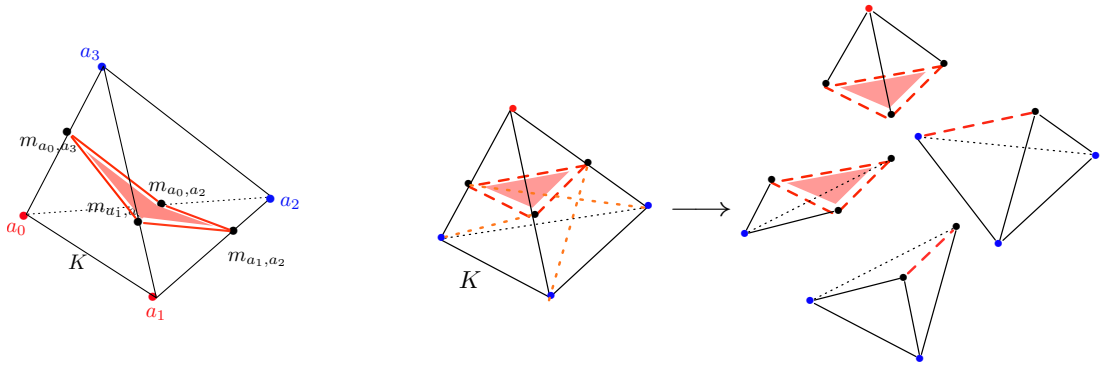


Figure 8.39: *(Left)* One of the possible situations when $\mathcal{S}_{\phi_\mathcal{T}}$ (in light red) crosses an element $K \in \mathcal{T}$; *(right)* splitting pattern for a tetrahedron $K \in \mathcal{T}$ which is crossed by $\mathcal{S}_{\phi_\mathcal{T}}$ in such a way as three of its vertices share the same sign (the blue ones).

This procedure yields a new mesh $\widetilde{\mathcal{T}}_1$ of D, which is most likely very ill-shaped, for the intersections of the elements of $\mathcal{T}$ with $\mathcal{S}_{\phi_\mathcal{T}}$ are utterly arbitrary. However, $\widetilde{\mathcal{T}}_1$ enjoys the two most important characteristics for our purposes:

- It is conforming; this fact is not absolutely evident since it was produced using a pattern-based splitting procedure, but it naturally stems from the fact the operations performed on adjacent elements of $\mathcal{T}$ are consistent with one another (the trace of $\mathcal{S}_{\phi_\mathcal{T}}$ on a face shared by two tetrahedra is the same when computed from either element).
- It encloses a (ill-shaped) mesh $\widetilde{\mathcal{T}}_1{}'$ of $D \cap \Omega$ as a submesh.

As a consequence, so as to obtain a new, well-shaped mesh of $D \cap \Omega$, it is enough to retain only the part $\widetilde{\mathcal{T}}_1{}'$ of $\widetilde{\mathcal{T}}_1$, and use it as an input for the algorithm described in section 8.3.

However, for several applications, it may be relevant to keep trace of an approximation of $D \setminus \Omega$, that is, to remesh the whole mesh $\widetilde{\mathcal{T}}_1$ into a new, well-shaped, mesh of D, which contains a mesh of $D \cap \Omega$ as a submesh (see Chapter 9). Bringing this problem back to the framework of section 8.3 actually demands only marginal adaptations, which we describe now.

8.4.2 From discrete domain remeshing to discrete domain and subdomains remeshing

We now have at our disposal a mesh $\widetilde{\mathcal{T}}_1$ of D, a submesh $\widetilde{\mathcal{T}}_1{}'$ of which accounts for $D \cap \Omega$. Let us denote as $\mathcal{S}_{\widetilde{\mathcal{T}}_1}$ the associated *total* surface mesh, that is,

$$\mathcal{S}_{\widetilde{\mathcal{T}}_1} = \mathcal{E}_{\widetilde{\mathcal{T}}_1} \cup \mathcal{I}_{\widetilde{\mathcal{T}}_1},$$

where $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$ stands for the collection of all the (triangular) external faces to the elements of $\widetilde{\mathcal{T}}_1$, and $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$ is the set of the external faces of the elements of $\widetilde{\mathcal{T}}_1{}'$ *which do not already belong to* $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$. In other terms, $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$ is a surface mesh associated to ∂D, and $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$ is a surface mesh associated to $\partial(D \cap \Omega) \setminus \partial D$. The latter set only contains triangular faces which belong to two distinct elements of $\widetilde{\mathcal{T}}_1$ (see figure 8.40).

Our purpose is to remesh $\widetilde{\mathcal{T}}_1$ into a new, well-shaped mesh $\widetilde{\mathcal{T}}$ of D, a submesh $\widetilde{\mathcal{T}}'$ of which accounts for $D \cap \Omega$, and such that the associated total surface mesh $\mathcal{S}_{\widetilde{\mathcal{T}}}$ is a close approximation of $\partial D \cup \partial(D \cap \Omega)$, within a prescribed range ε in terms of Hausdorff distance.

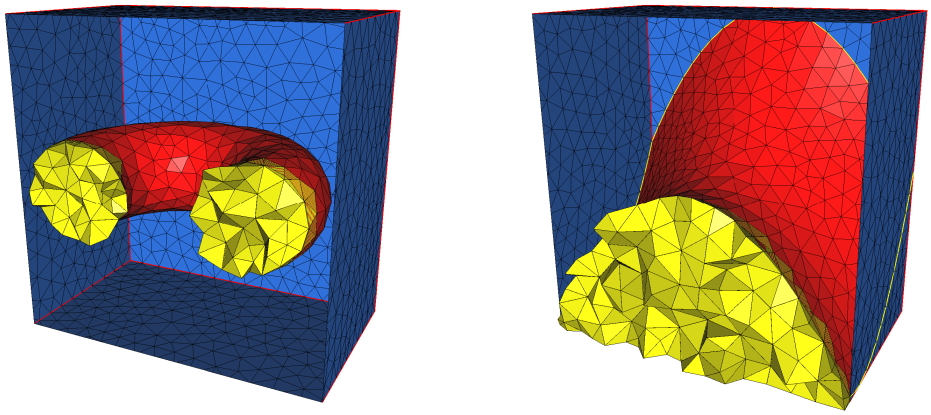


Figure 8.40: Two examples of (well-shaped) meshes of a box D, enclosing a mesh of a subdomain $D \cap \Omega$ as a submesh. In both case, the triangular faces of $\mathcal{E}_{\widetilde{\mathcal{T}}}$ appear in blue, and those of $\mathcal{I}_{\widetilde{\mathcal{T}}}$ in red. Only the tetrahedra of $D \cap \Omega$ have been displayed (in yellow). On the left-side example, D and $D \cap \Omega$ are disjoint, whereas they are not on the right-side one (the yellow edges are those edges which belong to both $\mathcal{E}_{\widetilde{\mathcal{T}}}$ and $\mathcal{I}_{\widetilde{\mathcal{T}}}$).

Actually, this problem happens to be very close to the one presented in section 8.3. Suppose for now that the two surfaces ∂D and $\partial\Omega$ at stake are disjoint - or equivalently that for all $x \in \partial D$, $\phi(x) \neq 0$ (case on Figure 8.40, left). Then, the procedure described in section 8.3 extends straightforwardly to remesh $\widetilde{\mathcal{T}}_1$ as suits our purpose. Indeed, the faces of $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$ and $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$ can be processed *independently*, and exactly in the same way as described previously. Admittedly, the entities belonging to $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$ are *a priori* more severely constrained compared to those of $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$, since they are connected to more elements of $\widetilde{\mathcal{T}}_1$, and the several checks to be performed before applying our local remeshing operators to ensure the validity of the resulting mesh are more likely to fail. Yet, those checks read just the same in both cases (and are physically the same in the written code).

The only real change arises when ∂D and $\partial\Omega$ do intersect one another, which in numerical practice translates as: $\mathcal{E}_{\widetilde{\mathcal{T}}_1} \cap \mathcal{I}_{\widetilde{\mathcal{T}}_1} \neq \emptyset$. In this case, the intersection between these two surface meshes is a collection of curves (see the right-hand side on Figure 8.40).

In this situation, both surface meshes $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$, $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$ considered separately are orientable, since they arise as (parts of) the boundary to a compact domain. Yet, their reunion is not, because of those curves making up $\mathcal{E}_{\widetilde{\mathcal{T}}_1} \cap \mathcal{I}_{\widetilde{\mathcal{T}}_1}$. To deal with this particular configuration, a new category of edges and vertices has to be added to the classification introduced in section 8.1.1, namely that of *non manifold edges*, or *vertices*: a non manifold edge of $\widetilde{\mathcal{T}}_1$ is an edge which belongs to at least one triangle of $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$, and to another triangle of $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$. The description adopted for non manifold edges / vertices is very similar to that of ridge edges / vertices: a non singular, non manifold point x (i.e. a non manifold point which belongs to a non manifold curve) is only equipped with a tangent vector $\tau(x)$ to this curve, and a different normal vector is used, depending on whether the point is processed as belonging to $\mathcal{E}_{\widetilde{\mathcal{T}}_1}$ or to $\mathcal{I}_{\widetilde{\mathcal{T}}_1}$.

Apart from this minor refinement, such configurations raise no further difficulty, and can be tackled in the very same way as in the case when $\partial D \cap \partial(D \cap \Omega) = \emptyset$.

8.4.3 Numerical examples and applications

8.4.3.1 Meshing of the negative subdomains associated to some level set functions

The above method for implicit surface (or domain) meshing is appraised on two examples.

First of all, let D a unit cube, equipped with a refined mesh $\mathcal{T}$ of 648944 vertices. A numerical approximation ϕ of the signed distance function to some mechanical device is generated as a $\mathbb{P}^1$ Lagrange finite element function on $\mathcal{T}$, using the algorithm of Chapter 6. A new mesh $\widetilde{\mathcal{T}}$ of D, enclosing a mesh $\mathcal{K}$ of the negative subdomain of u as a submesh is then obtained, using parameters $h_{min} = 0.001$, $h_{max} = 0.1$, $h_{grad} = 1.2$, and $\varepsilon = 0.005$. The whole computation takes 3 min 22 s, for a resulting mesh $\widetilde{\mathcal{T}}$ enjoying 62219 vertices, and an average quality of 0.78 (see figure 8.41).

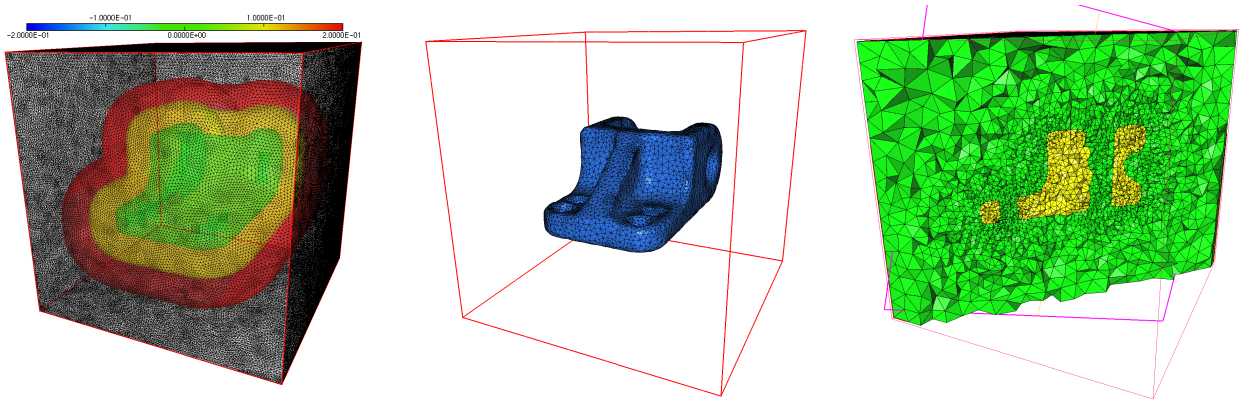


Figure 8.41: *(Left):* Isosurfaces of the implicit function u on the initial mesh $\mathcal{T}$, *(middle)* result of the computation (only the surface mesh $\mathcal{I}_{\widetilde{\mathcal{T}}}$ is represented), *(right)* a cut in the final mesh $\widetilde{\mathcal{T}}$.

In a second time, let D a box of dimensions $2.4 \times 5 \times 3$, equipped with a mesh $\mathcal{T}$ composed of 18024 vertices. A $\mathbb{P}^1$ Lagrange finite element function ϕ is considered on $\mathcal{T}$, which arises as an intermediate step of a structural optimization process (see the next chapter 9 for details). The negative subdomain of ϕ is meshed, using parameters $h_{min} = 0.05$, $h_{max} = 0.3$, $h_{grad} = 1.2$, and $\varepsilon = 0.03$. The total procedure takes 14.7 s, and the final mesh $\widetilde{\mathcal{T}}$ enjoys 13719 vertices, for an average quality of 0.77 (see figure 8.42).

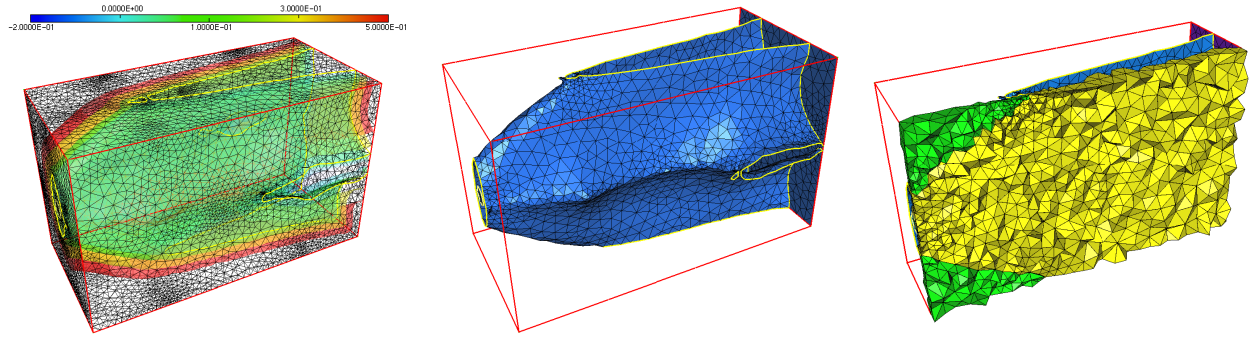


Figure 8.42: *(Left):* Isosurfaces of the implicit function u on the initial mesh $\mathcal{T}$, *(middle)* result of the computation (only the surface mesh $\mathcal{I}_{\widetilde{\mathcal{T}}}$ is represented), *(right)* a cut in the final mesh $\widetilde{\mathcal{T}}$.

8.4.3.2 Application to mesh generation from a possibly invalid surface triangulation

In this paragraph, we present a promising application of the previous implicit domain meshing algorithm in combination to three-dimensional mesh generation.

Let $\mathcal{S} = (T_i)_{i=1,\dots,N_S}$ a surface triangulation of the boundary $\partial\Omega$ of an open bounded domain $\Omega \subset \mathbb{R}^3$. The classical problem of three-dimensional mesh generation consists in constructing a tetrahedral mesh of Ω, whose associated surface mesh is exactly $\mathcal{S}$. This problem is very hard in general, and very few algorithms exist that are sufficiently polyvalent and robust to deal with general enough triangulations $\mathcal{S}$; one such algorithm is INRIA's software Ghs3d [154], which is based on the constrained Delaunay algorithm (see [145], or Chapter 3 for details).

Here, we propose a different approach, which inherently requires to drop the constraint that the initial boundary triangulation $\mathcal{S}$ should be retained through the process.

As a first stage, the initial surface triangulation $\mathcal{S}$ is embedded in a big computational domain D, equipped with a simplicial mesh $\mathcal{K}$. Choosing an arbitrary tolerance parameter $\varepsilon > 0$, the method described in Chapter 6 (which makes use of the $3d$ anisotropic volume remeshing algorithm `mmg3d`, version 4 [115]) for computing the signed distance function to a triangulated contour and adapting the computational mesh to this function can be used to produce simultaneously:

- a new (anisotropic) mesh $\widetilde{\mathcal{K}_1}$ of D,
- an approximation of the signed distance function d_Ω to Ω as a $\mathbb{P}^1$ Lagrange finite element function ϕ on $\widetilde{\mathcal{K}_1}$, which enjoys the following property: the Hausdorff distance $d^H(\mathcal{S}_\phi, \partial\Omega)$ between $\partial\Omega$ and the piecewise affine reconstructed surface $\mathcal{S}_\phi := \{x \in D,\ \phi(x) = 0\}$ is no higher than ε.

Note that the choice of an anisotropic mesh $\widetilde{\mathcal{K}_1}$ as a support of an approximation of d_Ω stems from the concern to guarantee an accurate representation of $\partial\Omega$, using a mesh whose size is moderate.

In a second stage, the negative subdomain of ϕ is meshed, using the method presented in this section, with the mesh $\widetilde{\mathcal{K}_1}$ of D: a new mesh $\widetilde{\mathcal{K}}$ of D is produced, which encloses a mesh $\mathcal{T}$ of Ω as a submesh.

This procedure is applied to the famous *Aphrodite*[9] model displayed on figure 8.43. The total remeshing time, from the datum of the mesh $\widetilde{\mathcal{K}_1}$ of a unit box D and the approximation ϕ to d_Ω is 10 min and 9 s, and the parameters of the computation are: $h_{min} = 0.001$, $h_{max} = 0.1$, $h_{grad} = 1.2$, $\varepsilon = 0.001$, for a final mesh $\widetilde{\mathcal{K}}$ enjoying an average quality of 0.77.

Remark 8.12. Looking carefully at the result displayed on figure 8.43, the obtained mesh $\widetilde{\mathcal{K}'}$ appears far from being completely satisfactory in terms of the accuracy of the approximation of $\partial\Omega$ by means of the associated surface mesh $\mathcal{S}_{\widetilde{\mathcal{K}'}}$. Actually, such inaccuracies are concentrated in those regions of $\partial\Omega$ where ridge edges, or singular points are present; the main reason is that, in those areas, the signed distance function u_Ω is inaccurately computed, whatever the size of the computational mesh $\widetilde{\mathcal{K}_1}$, because the corresponding ridge edges (or singular points) do not explicitly appear in $\widetilde{\mathcal{K}_1}$. A way to deal with this problem would be to discretize those edges, or points into $\widetilde{\mathcal{K}_1}$, before computing an approximation ϕ to d_Ω on it. Thus, u would amount to 0 on those edges (or points), and they would appear (up to a geometric equivalent) into $\widetilde{\mathcal{K}'}$. This latter part is an ongoing work.

9. Free model from `http://artist-3d.com/free_3d_models`.

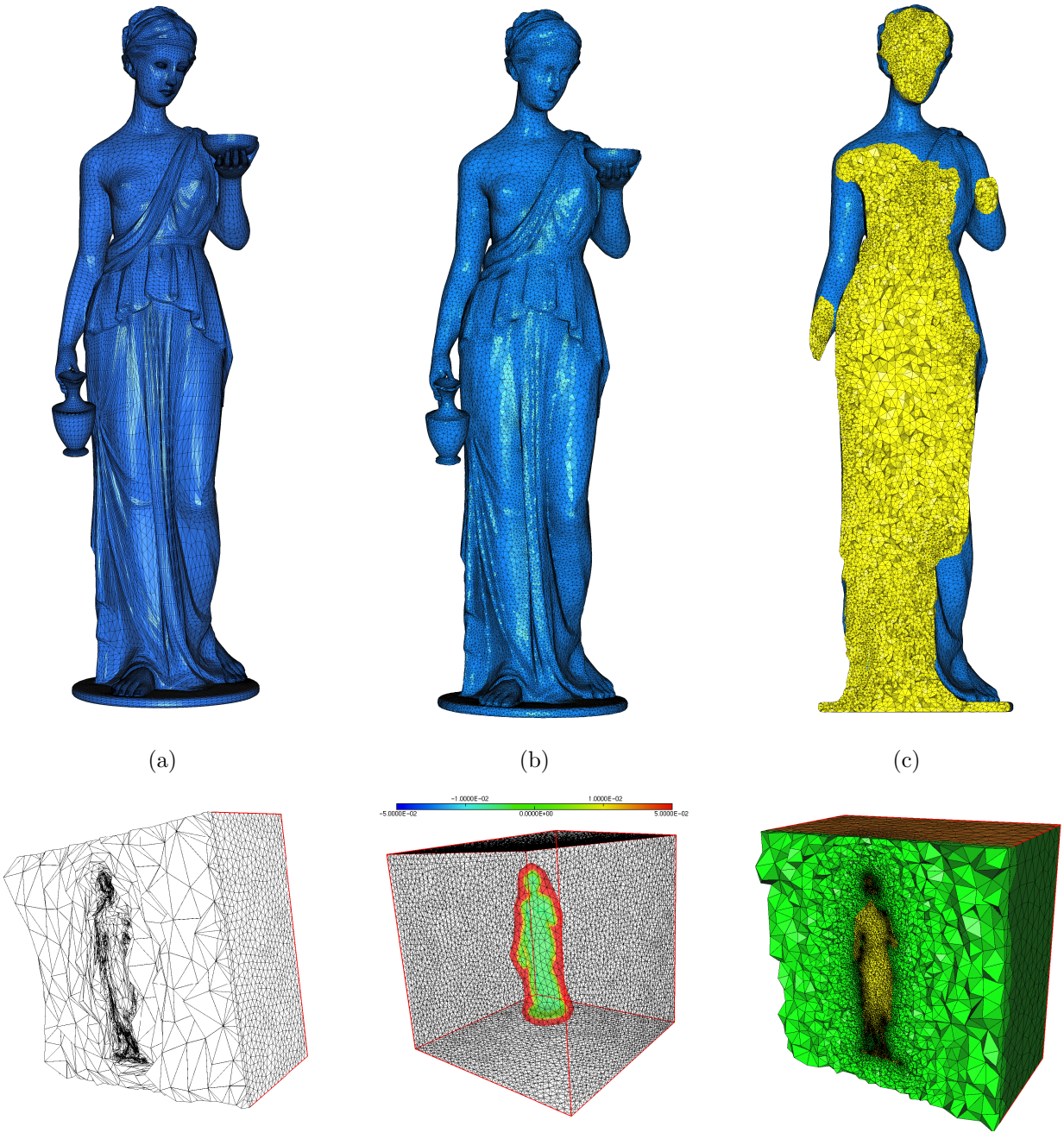


Figure 8.43: Meshing of the *Venus* model: *(a)* the initial surface triangulation $\mathcal{S}$, *(b-c)* the final three-dimensional mesh $\mathcal{T}$, *(d)* the support mesh $\widetilde{\mathcal{K}_1}$ of the approximation ϕ of d_Ω (around 410000 vertices), *(e)* some isosurfaces of ϕ, *(f)* a cut in the final mesh $\widetilde{\mathcal{K}}$ of D, which encloses $\mathcal{T}$ as a submesh, in yellow (around 375000 vertices).

Appendix: validity of remeshing operations

As we have seen in the course of this chapter, most of the local operators available for locally modifying meshes may invalidate them if no particular attention is paid, meaning that one of the axioms of the notion of mesh in effect in this manuscript (see Definition 3.1 in Chapter 3) may end up violated. The purpose of this section is to provide a convenient framework for controlling these operators in some particular cases. Indeed, we are about to see that the situations are very different between the operations performed on surface mesh configurations (i.e. in the case of a surface triangulation or, in the context of domain remeshing, in the case of a surface configuration), and those performed on a completely 'internal' configuration (in the context of domain remeshing).

Let us start by considering the case of fully volumetric configurations. We shall rely on the following definition and proposition.

Definition 8.15. *Let $\Omega \subset \mathbb{R}^d$ a Lipschitz (open) bounded domain. The set Ω^* of points $x \in \Omega$ such that Ω is star-shaped with respect to x, that is, equivalently,*
- *for all $y \in \Omega$, the segment $\{(1-t)x + ty, t \in [0,1]\}$ lies in Ω,*
- *for all $y \in \partial\Omega$, the half-open segment $\{(1-t)x + ty, t \in [0,1)\}$ lies in Ω,*

is called the visibility kernel of Ω.

Proposition 8.5.

1. *Let $\Omega \subset \mathbb{R}^d$ a bounded domain of class $\mathcal{C}^1$, n the (continuous) outer unit normal vector field on $\partial\Omega$. Then,*

$$\{x \in \Omega,\ \forall y \in \partial\Omega,\ (x-y)\cdot n(y) < 0\} \subset \Omega^* \subset \{x \in \Omega,\ \forall y \in \partial\Omega,\ (x-y)\cdot n(y) \leq 0\}.$$

2. *Let $\Omega \subset \mathbb{R}^d$ a polyhedral bounded domain, i.e. a Lipschitz domain whose boundary is a finite union of flat polygons: $\partial\Omega = \bigcup_{i=1}^N P_i$. For each $i = 1, ..., N$, denote as n_{P_i} the unit normal vector to P_i pointing outwards Ω. Then,*

$$\{x \in \Omega,\ \forall i = 1, ..., N, \forall y \in P_i,\ (x-y)\cdot n_{P_i} < 0\} \subset \Omega^* \subset \{x \in \Omega,\ \forall i = 1, ..., N, \forall y \in P_i,\ (x-y)\cdot n_{P_i} \leq 0\}.$$

Proof. We limit ourselves with the proof of the first point, that of the second one being identical. Denote as $K_1 := \{x \in \Omega,\ \forall y \in \partial\Omega,\ (x-y)\cdot n(y) < 0\}$, and $K_2 := \{x \in \Omega,\ \forall y \in \partial\Omega,\ (x-y)\cdot n(y) \leq 0\}$. Using a standard argument of partition of unity, it is easy to see that the bounded domain Ω can be described implicitly, i.e. that there exists a $\mathcal{C}^1$ function $\phi : \mathbb{R}^d \to R$ such that:

$$\forall x \in \mathbb{R}^d,\ \begin{cases} \phi(x) < 0 & \text{if} \quad x \in \Omega \\ \phi(x) = 0 & \text{if} \quad x \in \partial\Omega \\ \phi(x) > 0 & \text{if} \quad x \in^c \overline{\Omega} \end{cases}.$$

What's more, we have already seen that the normal vector n can be expressed in terms of ϕ as:

$$\forall y \in \partial\Omega,\ \ n(y) = \frac{\nabla\phi(y)}{|\nabla\phi(y)|}. \tag{8.73}$$

Now, let $x \in \Omega^*$. By definition, for all $y \in \partial\Omega$, and any $t \in [0,1]$, one has: $\phi(y + t(x-y)) \leq 0$. This implies

$$\forall t \in (0,1],\ \ \frac{\phi(y + t(x-y)) - \phi(y)}{t} \leq 0.$$

Letting t go to 0 and using (8.73) yields the inclusion $\Omega^* \subset K_2$.

Conversely, if $x \in K_1$, suppose there exists $y \in \partial\Omega$ such that not all the set $\{(1-t)x+ty, t \in [0,1)\}$ is comprised in Ω, i.e. $\phi((1-t)x+ty) > 0$ for a certain $t \in (0,1)$. By continuity of ϕ, there exists $t_0 \in (0,1]$, and a sequence (r_n) such that $\phi(x+t_0(y-x)) = 0$, and r_n is an increasing sequence towards t_0, which never takes value t_0, such that $\forall n \in \mathbb{N}$, $\phi((1-r_n)x+r_ny) \geq 0$. This implies

$$\forall n \in \mathbb{N}, \ \frac{\phi((1-r_n)x+r_ny) - \phi((1-t_0)x+t_0y)}{t_0 - r_n} \geq 0.$$

Letting $n \to \infty$ yields the sought contradiction. Consequently, $K_1 \subset \Omega^*$. □

Let us now put in the context of application of the remeshing operators presented in the course of this chapter, in a purely 'internal' configuration.

For the sake of simplicity, consider the model situation of a two-dimensional mesh $\mathcal{T} \subset \mathbb{R}^2$, and let p be an *internal* vertex of $\mathcal{T}$. Suppose p is to be relocated to a new position $\widetilde{p} \in \mathbb{R}^2$, while the connectivities of $\mathcal{T}$ are preserved; obviously, if no specific assumption is made on $\widetilde{p}$, the mesh $\widetilde{\mathcal{T}}$ resulting from the process may end up violating either the non empty interior condition (2) for an element of $\widetilde{\mathcal{T}}$ or the non overlapping condition (3) for elements in $\widetilde{\mathcal{T}}$ in definition 3.1 of Chapter 3 (which are by essence the only possible infringements on the definition of mesh which may be entailed by this operation).

Let $\Omega := \overset{\circ}{\mathcal{B}}(p)$ the *open ball* of p and denote as $e_i = a_{i-1}a_i$, $i = 1, ..., n$ the edges of this polygonal domain, where a_i, $i = 0, ..., n-1$ are its vertices, and we use the convention $a_n = a_0$. Obviously, $\widetilde{\mathcal{T}}$ stays valid if and only if $\widetilde{p} \in \Omega^*$ (see figure 8.44 (left)). From proposition 8.5, a necessary condition for this to hold is:

$$\forall i = 1, ..., n, \ \det(a_{i-1}a_i, a_{i-1}\widetilde{p}) \det(a_{i-1}a_i, a_{i-1}p) \geq 0.$$

As $\det(a_{i-1}a_i, a_{i-1}p) \neq 0$, the only way for $\widetilde{\mathcal{T}}$ to satisfy the non overlapping condition and to violate the non empty interior condition is that

$$\det(a_{i-1}a_i, a_{i-1}\widetilde{p}) = 0$$

should hold for a certain i. Thus, $\widetilde{\mathcal{T}}$ remains valid in the terms of definition 3.1 in Chapter 3 if and only if:

$$\forall i = 1, ..., n, \ \det(a_{i-1}a_i, a_{i-1}\widetilde{p}) \det(a_{i-1}a_i, a_{i-1}p) > 0,$$

a mere orientation predicate over the triangles of $\mathcal{B}(p)$, which is very easy to check numerically. This short analysis straightforwardly extends to the three-dimensional case, and also legitimates the fact that a simple check on the preservation of orientations of the triangles of the balls of the affected points allow for a rigorous control of operations such as internal edge collapses, internal edge swaps, etc...

Unfortunately, things are downright different when considering vertices or edges on the boundary of $\mathcal{T}$; those criteria relying on orientation predicates happen not to be sufficient to guarantee the resulting mesh validity, and one has to resort to some heuristics. For instance, there is no simple criterion such as the fact that $\widetilde{p}$ should belong to the visibility kernel of its associated open ball to ensure $\widetilde{\mathcal{T}}$ stays valid after relocating a boundary point p to $\widetilde{p}$; see figure 8.44 (right).

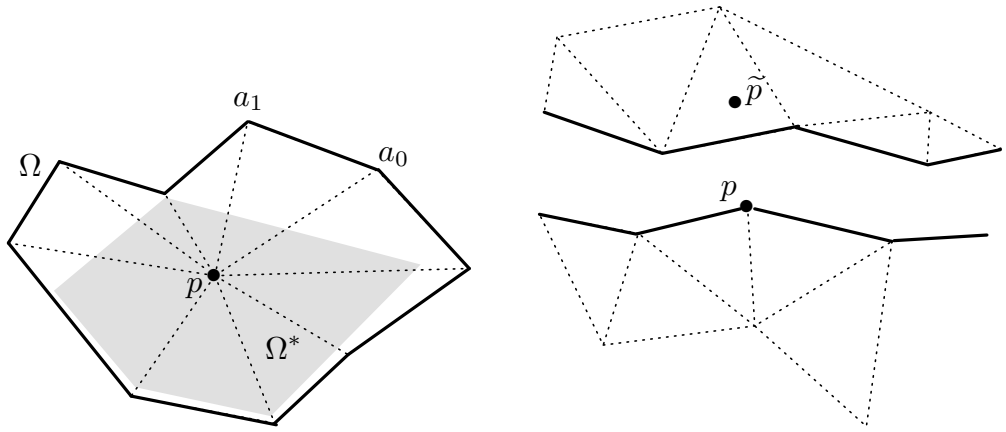


Figure 8.44: *(Left):* Visibility kernel associated to the open ball of point p (greyed area), *(right)* invalid relocation for a point p lying on the boundary of $\mathcal{T}$ (displayed as bold lines).

To put things in a nutshell, operations performed on a configuration which is completely 'surrounded by volume' are easy to control rigorously, whereas such a control in the context of a boundary configuration (or in the case of a surface mesh in $\mathbb{R}^3$) can be at most ensured by heuristic checks.

Part V

A level-set based method for mesh evolution for shape optimization

Chapter 9

Shape optimization with a level set based mesh evolution method

Contents

9.1 Introduction . . . 352
9.2 A model problem in shape optimization of elastic structures . . . 354
9.3 Two complementary ways for representing shapes . . . 356
9.3.1 Generating the signed distance function to a discrete domain . . . 356
9.3.2 Meshing the negative subdomain of a scalar function . . . 357
9.3.2.1 Step 1: discretization of the 0 level set of a scalar function into a simplicial mesh . . . 358
9.3.2.2 Step 2: local modifications of a simplicial mesh for quality and geometric approximation improvements . . . 358
9.4 Accounting for shape evolution . . . 360
9.4.1 A brief reminder of the level set method . . . 360
9.4.2 Resolution of the level set advection equation on an unstructured mesh . . . 361
9.4.3 Computation of a descent direction . . . 361
9.5 The global algorithm . . . 362
9.6 Numerical examples . . . 363
9.6.1 Minimization of the compliance . . . 363
9.6.1.1 Two-dimensional examples . . . 363
9.6.1.2 A two-dimensional example using the topological derivative . . . 366
9.6.1.3 3*d* examples . . . 368
9.6.2 Multi-loads compliance minimization . . . 371
9.6.3 Chaining topological and geometric optimization . . . 373
9.6.4 Multi-materials compliance minimization . . . 376
9.6.5 Minimization of least-square criteria . . . 379
9.6.5.1 A gripping mechanism . . . 379
9.6.5.2 A three-dimensional example of worst-case design in shape optimization . 381
9.6.6 Stress criterion minimization . . . 383
9.7 Conclusions and perspectives . . . 384

In this chapter, we discuss an approach for structural optimization which benefits from an accurate description of shapes at each stage of the iterative process - by means of a mesh amenable for mechanical

analyses - while retaining the whole versatility of the level set method when it comes to accounting for their evolution. The key ingredient of this method is a set of operators for switching from a meshed representation of a domain to an implicit one, and conversely; this notably brings into play an algorithm for generating the signed distance function to an arbitrary discrete domain, and a mesh generation algorithm for implicitly-defined geometries.

This chapter is a joint work with Grégoire Allaire and Pascal Frey; parts of its contents have been published in the following two brief notes:

G. ALLAIRE, C. DAPOGNY AND P. FREY, *Topology and Geometry Optimization of Elastic Structures by Exact Deformation of Simplicial Mesh*, C. R. Acad. Sci. Paris, Ser. I, vol. 349, no. 17, pp. 999-1003 (2011),

G. ALLAIRE, C. DAPOGNY AND P. FREY, *A mesh evolution algorithm based on the level set method for geometry and topology optimization*, accepted in Struct. Multidisc. Optim. (2013), DOI 10.1007/s00158-013-0929-2,

and in the following two conference proceedings:

G. ALLAIRE, C. DAPOGNY AND P. FREY, *Shape optimization of elastic structures using a level-set based mesh evolution method*, Fifth International Conference on Advanced COmputational Methods in ENgineering (ACOMEN), Liège, Belgium, 2011,

G. ALLAIRE, C. DAPOGNY AND P. FREY, *A mesh evolution algorithm based on the level set method for geometry and topology optimization*, 10^{th} World Congress on Structural and Multidisciplinary Optimization (2013), Orlando, Florida, USA.

9.1 Introduction

In the simulation of a free or moving boundary problem driven by a physical motion, one usually has to strike a balance between numerical accuracy and robustness: the more faithful the representation of the tracked boundary, the more accurate the computation of the characteristics of the motion, and unfortunately, the more tedious the numerical implementation. This issue is especially critical in shape optimization which features problems where the changes in geometry and topology of shapes in the course of the evolution often turn out to be dramatic.

In a very summary way, to focus only on the field of shape and topology optimization, three main classes of techniques can be distinguished, depending on the representation of shapes they involve:

- *Density-based methods*, such as the SIMP method [40], or the homogenization method [10, 39], transform the problem of finding the optimal shape $\Omega \subset \mathbb{R}^d$ with respect to a mechanical criterion $J(\Omega)$ into that of finding the optimal density function $\rho : D \to [0,1]$ of a mixture of material and void inside a fixed working domain D. The shape optimization problem has to be converted into this new framework, which sometimes proves difficult.
- *Eulerian methods*, such as the phase field method [59], or the level set method [14, 243, 278, 319] account for shapes in an implicit way; for instance, in the latter case, a large, fixed working domain D, meshed once and for all is introduced, and a shape $\Omega \subset D$ is described in terms of a scalar function $\phi : D \to \mathbb{R}$ whose negative subdomain matches with Ω. Finite element analyses cannot be performed directly on Ω since it is not meshed exactly, and approximations have to be made to trade mechanical problems posed on Ω for problems posed on D. The most notorious of them is the Ersatz material approach, which consists in filling the 'void' $D \setminus \overline{\Omega}$ with a very soft material to avoid degeneracy in the stiffness matrix (however, alternatives exist, which are based on e.g. the immersed boundary method [278], or the XFEM method [120, 202]).

– *Lagrangian methods* are perhaps the most natural ones and date back to the early hours of computational structural optimization [338]; shapes are represented by means of a computational mesh (or a CAD model [60]), which enables accurate mechanical analyses, but this mesh has to be updated in the course of the optimization process, which is notoriously difficult, especially in $3d$. Note that there is still ongoing research in this direction [17, 231].

Of course, this rough classification ignores the numerous particular instances of each category of methods and combinations between them (see the recent review papers [104], or [112] for a stronger emphasis on level-set based structural optimization).

In this chapter, we ambition to devise a shape optimization method which benefits from the flexibility of level-set based shape optimization methods for tracking evolution of shapes, while enjoying an exact, meshed description of shapes.

Admittedly, this idea of combining an implicit domain evolution method with an explicit type of shape representation is not new: in the two-dimensional work [326], the evolution of shapes is tracked on a triangular mesh $\mathcal{T}$ of a working domain D owing to the level set method, and at each iteration of the process, an exact mesh of the current shape Ω is obtained by relocating vertices of $\mathcal{T}$ onto $\partial\Omega$. In [325], a similar strategy is applied for dealing with the motion of shapes; a computational mesh for any shape Ω arising during the process is moreover constructed by first identifying the intersection points of the implicitly-defined boundary $\partial\Omega$ with the edges of the mesh $\mathcal{T}$ of D, then using them as an input for a Delaunay-based mesh generation algorithm. Last but not least, let us mention the work in [254] (chap. 5), taken over in [167], in which the evolution of shapes occurs on a finite difference grid of the working domain D, and an original meshing algorithm for implicit geometries is used to get an exact representation of shapes. The precisely calculated shape gradient must then be projected back to the Cartesian grid of D to close the loop.

The method detailed in this chapter has something in common with this last one. A computational domain D is defined, and is equipped with an unstructured mesh which is modified from one iteration of the algorithm to the next, in such a way that any shape Ω arising in the course of the process is *explicitly discretized* in this mesh - i.e. the vertices, edges, faces (and tetrahedra in $3d$) of a mesh of Ω also exist as elements of the mesh of D. In such a configuration, we shall also say that (a mesh of) Ω exists as a *submesh* of that of D. This kind of representation allows for accurate finite element analyses, held on a well-defined, high-quality mesh of Ω (which is possibly adapted to an error estimate for the mechanical problem at stake), and lends itself to the use of the level set method in an unstructured mesh framework, to account for large shape deformations (including potential topological changes). It relies crucially on efficient algorithms for moving back and forth, from a situation where a shape Ω is known as a submesh of the computational mesh of D to a level set description on a mesh of D.

This strategy presents several attractive assets; first, no projection between different meshes is needed between the computation of a descent direction for the considered objective function of the domain (which occurs when the shape is in a meshed description), and the further deformation of the shape (which is carried out using the level set method). Most importantly, the proposed method does not pose any theoretical obstruction to the extension from the two-dimensional case to the three-dimensional one (even if it is then considerably more tedious to implement). This is an important and non trivial feature insofar as meshing algorithms are involved, and meshing issues (e.g. Delaunay-based mesh generation) are well-known to be by far more difficult to deal with in $3d$ than in $2d$. The only *mesh generation* operation involved in our strategy is a purely logical (thus very robust) one, and most of the difficulty of the problem is transferred to a *remeshing*[1] problem, which always starts from an existing - possibly very ill-shaped mesh - and proceeds 'in the best possible way'.

In this chapter, we are mostly interested in the three-dimensional setting; consequently, most of the descriptions will be held with this case in mind (especially as far as meshing issues are concerned), except when the $2d$ and $3d$ settings are completely equivalent.

1. Depending on the authors, this term may either refer to the operation of *creating* a whole new mesh, or to that of *modifying* an existing one by means of local mesh operations. In this chapter, as in the previous one, the latter meaning is retained.

This chapter is organized as follows. The next section presents the model linear elasticity problem, and the basic material from shape-sensitivity analysis using Hadamard's method involved in the forthcoming study; section 9.3 then describes the two different representations of shapes used in our method - namely the level set representation, and the meshed representation - as well as the algorithms for switching from one of these representations to the other. Then, section 9.4 describes how the velocity field driving the motion of shapes is computed, and how the level set method is used to account for this motion. The global mesh evolution strategy is summed up in section 9.5, and several numerical examples are discussed in section 9.6. Eventually, we draw several conclusions around the present study in section 9.7, and outline some natural directions for future work.

9.2 A model problem in shape optimization of elastic structures

In this chapter, we consider *shapes*, that is, bounded open sets $\Omega \subset \mathbb{R}^d$ ($d = 2, 3$ in our applications) with at least Lipschitz regularity, filled with a linear isotropic elastic material with Hooke's law A:

$$\forall \xi \in \mathcal{S}(\mathbb{R}^d), \;\; A\xi = 2\mu\xi + \lambda \mathrm{tr}(\xi),$$

where $\mathcal{S}(\mathbb{R}^d)$ is the set of real $d \times d$ symmetric matrices, and λ and μ are the Lamé coefficients of the material. These shapes are clamped on a part $\Gamma_D \subset \partial\Omega$ of their boundaries, and they are submitted to body forces f, as well as to traction loads g, applied on a part $\Gamma_N \subset \partial\Omega$ disjoint from Γ_D. The remaining, traction-free region $\Gamma := \partial\Omega \setminus (\Gamma_D \cup \Gamma_N)$ is called the *free boundary* of Ω.

Provided $f \in L^2(\Omega)^d$, $g \in H^1(\Omega)^d$, and that $\Gamma_D \neq \emptyset$ (unless an equilibrium condition between f and g is fulfilled), the displacement of a shape Ω is the unique solution $u_\Omega \in H^1(\Omega)^d$ to the linear elasticity system:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(u)) & = & f & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ Ae(u)n & = & g & \text{on } \Gamma_N \\ Ae(u)n & = & 0 & \text{on } \Gamma \end{array} \right. , \tag{9.1}$$

where $e(u) := \left(\nabla u^T + \nabla u\right)/2$ is the linearized strain tensor, and n is the unit normal vector to $\partial\Omega$ (pointing outward Ω).

Our purpose is to minimize a given functional $J(\Omega)$ of the domain. This classically demands some knowledge about the derivatives of J, hence the need to account for variations of shapes.

In this perspective, we follow the lead of [234] (see also [172, 290], and Chapter 2), and rely on Hadamard's method: variations of a smooth shape $\Omega \subset \mathbb{R}^d$ of the form $\Omega_\theta := (I + \theta)(\Omega)$ are considered, for $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, $||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1$. It is indeed well-known [9] that under such conditions, $(I + \theta)$ is a Lipschitz diffeomorphism of $\mathbb{R}^d$. The induced notion of shape differentiation is then the following:

Definition 9.1. *A real-valued function $J(\Omega)$ of the domain is shape differentiable at a shape Ω if the underlying function $\theta \mapsto J((I + \theta)(\Omega))$ from $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ into $\mathbb{R}$ is Fréchet differentiable at $\theta = 0$. The associated derivative $J'(\Omega)$ is called the shape derivative of J at Ω, and the following asymptotic expansion holds in the neighborhood of $0 \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$:*

$$J(\Omega_\theta) = J(\Omega) + J'(\Omega)(\theta) + o(\theta), \quad \textit{where } \frac{o(\theta)}{||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)}} \overset{\theta \to 0}{\longrightarrow} 0. \tag{9.2}$$

Let us now precise the setting of the forthcoming study. The parts Γ_D and Γ_N of the boundaries of shapes where they are respectively clamped and submitted to traction loads are given *a priori*, and are not subject to optimization. The minimization of $J(\Omega)$ is thus investigated over the set $\mathcal{U}_{ad}$ of *admissible shapes* defined as:

$$\mathcal{U}_{ad} = \left\{ \Omega \subset \mathbb{R}^d \text{ is an open, Lipschitz bounded set, } \Gamma_D \cup \Gamma_N \subset \partial\Omega \right\}.$$

The corresponding set for *admissible variations* of shapes is:

$$\Theta_{ad} = \left\{\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d),\ \theta = 0 \text{ on } \Gamma_D \cup \Gamma_N\right\}.$$

Throughout this chapter, we shall consider integral functionals $J(\Omega)$, which bring into play the solution u_Ω to the linear elasticity system (9.1). A rigorous study of such functions is not an easy task; fortunately, Céa's method [72] (see also the presentation in chapter 2) allows for an easy, albeit formal, computation of their shape derivatives.

To set ideas, let us recall a classical result, devoted to functionals of the form (see e.g. [14] for details):

$$\forall \Omega \in \mathcal{U}_{ad},\ \ J(\Omega) = \int_\Omega j(x, u_\Omega(x))\, dx + \int_{\Gamma \cup \Gamma_N} k(x, u_\Omega(x))\, ds, \tag{9.3}$$

where $j, k : \mathbb{R}^d_x \times \mathbb{R}^d_u \to \mathbb{R}$ are two smooth functions satisfying adequate growth conditions (we shall meet several different instances of objective functions in section 9.6).

Theorem 9.1. *Provided f and g are smooth enough, the function $J(\Omega)$ defined by (9.3) is shape differentiable at any $\Omega \in \mathcal{U}_{ad}$, and its shape derivative reads:*

$$\forall \theta \in \Theta_{ad},\ \ J'(\Omega)(\theta) = \int_\Gamma \left(j(x, u_\Omega) + Ae(u_\Omega) : e(p_\Omega) - f \cdot p_\Omega + \frac{\partial}{\partial n}(k(x, u_\Omega)) + \kappa k(x, u_\Omega) \right) \theta \cdot n\, ds, \tag{9.4}$$

where κ is the mean curvature of $\partial\Omega$ (oriented in the sense that $\kappa(x)$ is positive when Ω is locally convex near x), and $p_\Omega \in H^1(\Omega)^d$ is the adjoint state, unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}(Ae(p)) & = & -j'(x, u_\Omega) & \text{in } \Omega \\ p & = & 0 & \text{on } \Gamma_D \\ Ae(p)n & = & -k'(x, u_\Omega) & \text{on } \Gamma \cup \Gamma_N \end{array} \right. . \tag{9.5}$$

More specifically, in good agreement with the structure theorem (see [105], Th 9.3.6), the shape derivatives of all the handled functionals $J(\Omega)$ in this chapter will turn out to be of the form:

$$\forall \theta \in \Theta_{ad},\ \ J'(\Omega)(\theta) = \int_\Gamma v_\Omega\, (\theta \cdot n)\, ds, \tag{9.6}$$

for a certain scalar field v_Ω on Γ. A descent direction θ for J is then easily revealed as $-v_\Omega n$, since letting

$$\theta = -t\, v_\Omega n \tag{9.7}$$

in (9.2) yields, for small $t > 0$:

$$J(\Omega_{t\theta}) = J(\Omega) - t \int_\Gamma v_\Omega^2\, ds + o(t) < J(\Omega).$$

Actually, for both theoretical and numerical reasons, one cannot take directly (9.7) as a descent direction, but we shall come back to this issue in section 9.4.3.

Remark 9.1. We have hitherto been discussing the unconstrained minimization of a function $J(\Omega)$ over $\mathcal{U}_{ad}$. For the sake of simplicity, in this chapter, we shall only impose a volume constraint on shapes, to be enforced by trading $J(\Omega)$ for a weighted sum $\mathcal{L}(\Omega)$ of $J(\Omega)$ and the volume of shapes $\mathrm{Vol}(\Omega)$, so that the problem boils down to the following constraint-free problem:

$$\min_{\Omega \in \mathcal{U}_{ad}} \mathcal{L}(\Omega),\ \ \mathcal{L}(\Omega) := J(\Omega) + \ell\, \mathrm{Vol}(\Omega), \tag{9.8}$$

where ℓ is a fixed Lagrange multiplier.

Note that this very rough understanding of constraints already contains some degree of generality, since many efficient optimization algorithms (e.g. the augmented Lagrangian method) impose constraints by using formulations of the form (9.8) in combination with an update strategy for the Lagrange multiplier ℓ.

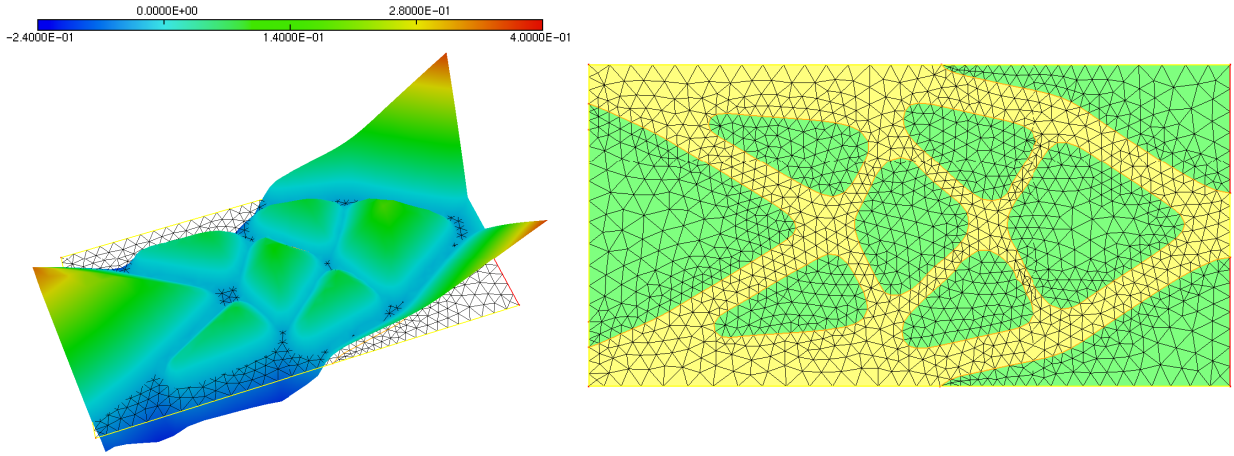


Figure 9.1: *(Left)* graph of a level set function ϕ associated to a shape Ω; *(right)* a corresponding meshed description: the whole computational box D is equipped with a mesh $\mathcal{T}_\Omega$ (composed of the yellow and green elements), and the submesh $\mathcal{T}'_\Omega$ is composed of the yellow triangles.

9.3 Two complementary ways for representing shapes

From on now, let D be a fixed, large computational domain that encloses all the considered shapes. The central point of the proposed method consists in juggling with two different ways for describing a shape $\Omega \subset D$ during the optimization process (see Figure 9.1), using alternatively one or the other depending on the nature of the ongoing operation:

– *The level set description:* Ω is implicitly defined by the datum of a scalar function $\phi : D \to \mathbb{R}$, in the sense that the following holds:

$$\forall x \in D, \begin{cases} \phi(x) < 0 & \text{if } x \in \Omega \\ \phi(x) = 0 & \text{if } x \in \partial\Omega \\ \phi(x) > 0 & \text{if } x \in {}^c\overline{\Omega} \end{cases} . \tag{9.9}$$

From the numerical standpoint, ϕ is discretized as a $\mathbb{P}^1$ Lagrange finite element function on a (simplicial) mesh of D. As we shall recall in section 9.4, this way of representating shapes is particularly well-suited when its comes to tracking their evolutions.

– *The meshed description:* the whole computational domain D is equipped with a simplicial (conforming) mesh, denoted $\mathcal{T}_\Omega$, which encloses a mesh $\mathcal{T}'_\Omega$ of Ω as a *submesh*, i.e. the elements (points, edges, faces, and tetrahedra in $3d$) of $\mathcal{T}'_\Omega$ also exist as elements of the larger mesh $\mathcal{T}_\Omega$.
This description of Ω is convenient when it comes to performing mechanical computations on it (e.g. using the finite element method): the 'exterior' part $\mathcal{T}_\Omega \setminus \mathcal{T}'_\Omega$ of $\mathcal{T}_\Omega$, i.e. that corresponding to $D \setminus \Omega$, is 'forgotten', and only the computational mesh $\mathcal{T}'_\Omega$ of Ω is retained.
At this point, one may question over the decision to systematically mesh a shape Ω together with its complementary part $D \setminus \Omega$, but the need to do so will become apparent in the next sections.

Let us now describe the two operators for switching from one representation to the other as we see fit.

9.3.1 Generating the signed distance function to a discrete domain

The first operation under scrutiny consists in generating a level set function for a domain $\Omega \subset \mathbb{R}^d$, at the vertices of a mesh $\mathcal{T}_\Omega$ of D in which Ω is explicitly discretized.

To achieve this, we compute the *signed distance function* d_Ω to Ω, defined by:

$$\forall x \in \mathbb{R}^d,\ d_\Omega(x) = \begin{cases} -d(x,\partial\Omega) & \text{if } x \in \Omega \\ 0 & \text{if } x \in \partial\Omega \\ d(x,\partial\Omega) & \text{if } x \in {}^c\overline{\Omega} \end{cases},$$

where $d(.,\partial\Omega)$ is the Euclidean distance function to $\partial\Omega$. Indeed, since [89], the signed distance function has been known to enjoy several important properties - the most crucial of them being that its gradient is of unit norm wherever it is defined, i.e. $|\nabla d_\Omega| = 1$ a.e. on $\mathbb{R}^d$ - which tremendously increase the numerical accuracy and stability of computations performed in a process making use of the level set method.

We use the numerical scheme of chapter 6, which relies on the property that d_Ω is the stationary state of the *unsteady Eikonal equation*:

$$\begin{cases} \frac{\partial\phi}{\partial t}(t,x) + \mathrm{sgn}(\phi_0)\,(|\nabla\phi| - 1) = 0 & \text{for } (t,x) \in (0,\infty) \times \mathbb{R}^d \\ \phi(t=0,x) = \phi_0(x) & \text{for } x \in \mathbb{R}^d \end{cases}, \tag{9.10}$$

where ϕ_0 is *any* level set function associated to Ω. Note that such a function is easily generated in practice, e.g. by defining $\phi_0(x)$ with the exact signed distance function to Ω at 'close' vertices $x \in \mathcal{T}_\Omega$ to $\partial\Omega$ (which is then inexpensive), and with an arbitrarily great value at the remaining points of $\mathcal{T}_\Omega$.

Taking advantage of the fact that 'the' solution to (9.10) is known through an explicit formula [26], an iterative scheme is obtained which 'straightens up' an initial level set function ϕ_0 for Ω into a new one enjoying a unit gradient, starting from areas near $\partial\Omega$ to farther ones.

Remark 9.2. The above hypothesis that Ω should be explicitly discretized in the computational mesh of D is not mandatory for this operation: Ω could well be supplied via *any* mesh, independently from that of D; see chapter 6 for details.

9.3.2 Meshing the negative subdomain of a scalar function

The second operation of interest considers a simplicial mesh $\mathcal{T}$ of D, and a level set function ϕ, discretized at the vertices of $\mathcal{T}$, associated to a domain $\Omega \subset D$. The aim is to modify $\mathcal{T}$ into a new mesh $\mathcal{T}_\Omega$ of D in which Ω is explicitly discretized.

We shall also impose two additional features as regards $\mathcal{T}_\Omega$:

- The mesh $\mathcal{T}_\Omega$ should be *well-shaped*, in terms of the qualities of its elements, i.e. the simplices $K \in \mathcal{T}_\Omega$ should have equal length edges, insofar as possible. This is a natural requirement since we plan on performing mechanical analyses on $\mathcal{T}_\Omega$, in particular using the finite element method, whose accuracy is well-known to be directly impacted by the quality of the computational support.
- The submesh $\mathcal{T}'_\Omega$ of $\mathcal{T}_\Omega$ should be *adapted* to the geometrical features of Ω, in particular show smaller elements around the regions of $\partial\Omega$ where curvature is high. This requirement might seem a bit loose at first glance, since in our framework, ϕ and Ω are only known in discrete way - and are respectively a piecewise linear function and a polyhedral domain (thus, strictly speaking, there is no such thing as curvature as far as Ω is concerned). Actually, for a number of reasons, it proves convenient to create a continuous geometric model for Ω, from the datum of a mesh of Ω and additional reconstructed information (e.g. the normal vectors at vertices of this mesh). In more practical terms, rules are established to infer a local portion of continuous surface Γ from any given discrete surface triangle T of $\partial\Omega$. In our setting, as suggested in [316], this piece of surface is parametrized as a cubic Bézier patch $\sigma : T \to \Gamma$, which interpolates the three vertices and three normal vectors of T. This local model serves then as a guide when it comes to introducing new points on $\partial\Omega$, and results in simple predicates over the vertices and normal vectors of the surface mesh when it comes to measuring whether such or such operation degrades too much the geometrical features of Ω. See chapter 8 for further details.

Modifying $\mathcal{T}$ into such a mesh $\mathcal{T}_\Omega$ is achieved within two steps, which we now outline: at first, a mesh $\mathcal{T}_{temp}$ of D is obtained, in which Ω is explicitly discretized, but which may be very ill-shaped, or may be a poor representative of the geometry of Ω. In a second step, this intermediate mesh $\mathcal{T}_{temp}$ is remeshed into a high-quality mesh $\mathcal{T}_\Omega$, which is a fine representative of Ω.

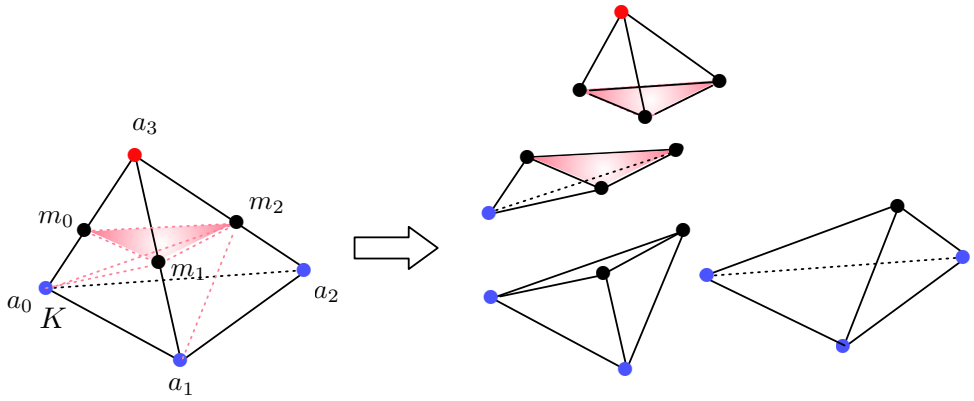


Figure 9.2: A pattern for splitting a simplex K so that the 0 level set (red face) of a linear function ϕ in K explicitly appears in the resulting decomposition. The function ϕ takes a positive value at the red vertex of K, and negative ones at the blue vertices.

9.3.2.1 Step 1: discretization of the 0 level set of a scalar function into a simplicial mesh

If no particular attention is paid to the qualities of its elements, obtaining a mesh $\mathcal{T}_{temp}$ of D in which Ω is explicitly discretized is a fairly easy matter: we use a *marching tetrahedra* approach [141] - a variant of the well-known marching cubes algorithm in the context of a Cartesian computational support [208].

The simplices of $\mathcal{T}$ intersecting $\partial\Omega = \{x \in D,\ \phi(x) = 0\}$ are exactly those bearing at least two vertices where ϕ takes different signs. For any such simplex $K \in \mathcal{T}$, as $\phi|_K$ is linear, $\partial\Omega \cap K$ is a portion of plane passing through the points m_i of the edges of K where ϕ vanishes. Once the positions of these points have been computed, depending on the relative signs of ϕ at the vertices of K, a pattern is chosen for splitting K into several simplices in such a way that a triangulation of $\partial\Omega \cap K$ explicitly appears (see Figure 9.2).

This step is unfortunately very likely to produce a severely ill-shaped mesh $\mathcal{T}_{temp}$, with very small or nearly degenerate elements, since the intersections of the simplices $K \in \mathcal{T}$ with $\partial\Omega$ are arbitrary (for instance, in the configuration of Figure 9.2, if the portion of plane $\partial\Omega \cap K$ lies very close to the vertex a_3, the new tetrahedron $a_3 m_0 m_1 m_2$ will be far too small and $a_0 m_0 m_1 m_2$ will be almost flat).

Nevertheless, let us note that this purely logical step is the only mesh *generation* operation involved in the mesh evolution method at stake in this chapter, which is very robust in delivering *valid* simplicial meshes. The remaining meshing ingredients, whose descriptions follow, only consider, modify and deliver valid meshes, doing their utmost in increasing their qualities. This is the main reason for the robustness of the proposed approach.

9.3.2.2 Step 2: local modifications of a simplicial mesh for quality and geometric approximation improvements

We are now left with the problem of remeshing a possibly ill-shaped mesh $\mathcal{T}_{temp}$ of D, enclosing an explicit discretization of Ω (which may be poor as a geometric approximation of the continuous underlying model).

To achieve this, we rely on the four usual local mesh modification operators (see [145], or chapter 8 for more details around the actual implementation), which are briefly described hereafter. Note that, in our application, each one of them exists under two different forms depending on whether it affects the surface triangulation of $\partial\Omega$, or it is applied to a completely internal configuration.

– *Edge split:* an edge pq of $\mathcal{T}_{temp}$ which is 'too long' is split by introducing a new vertex m in the mesh, and replacing pq by pm and qm, updating the connectivities of the mesh accordingly. An edge may be

deemed 'too long' if it is too long with respect to a user-defined size prescription, or if it entails too large a gap between $\partial\Omega$ and the underlying continuous surface Γ. The new vertex m is inserted either on Γ if pq is an edge of $\partial\Omega$, or as the midpoint of pq if it is an internal one.

- *Edge collapse:* the two endpoints of a 'too short' edge pq of $\mathcal{T}_{temp}$ are merged. This operator should be cautiously monitored: not only is it likely to degrade the quality of the representation of the geometrical features of Ω, but it may also invalidate elements of the mesh (i.e. cause overlappings), or provoke topologically invalid (e.g. non manifold) surface configurations.
- *Edge swap:* An edge is removed in $\mathcal{T}_{temp}$, and the connectivities of the mesh are updated accordingly. This operator is easy to apprehend in $2d$, or as its action on a surface configuration: in this case, a configuration of two triangles $T_1 = apq$ and $T_2 = bpq$ sharing common edge pq is simply replaced by the alternate configuration of triangles $\widetilde{T_1} = pab$ and $\widetilde{T_2} = qab$, sharing edge ab (see Figure 9.3, (c)); however, it becomes much more combinatorial and tedious in $3d$, as regards the necessary reconnections in $\mathcal{T}_{temp}$ (see [114, 150], or chapter 8 for further details). In both cases, this operator too may invalidate $\mathcal{T}_{temp}$, or degrade the geometric features of $\partial\Omega$, and should be carefully controlled.
- *Vertex relocation:* A vertex $p \in \mathcal{T}_{temp}$ is relocated to an *improving position* $\widetilde{p}$, keeping the connectivities of $\mathcal{T}$ unchanged. The choice as for the *improving position* $\widetilde{p}$ depends on whether p belongs to $\partial\Omega$ or it is an internal vertex. In the former case, $\widetilde{p}$ should lie on the continuous model associated to $\partial\Omega$, whereas in the latter case, it is simply chosen as the centroid of the simplices sharing p as a vertex (see however [145, 139] for other possibilities as for the relocation position).

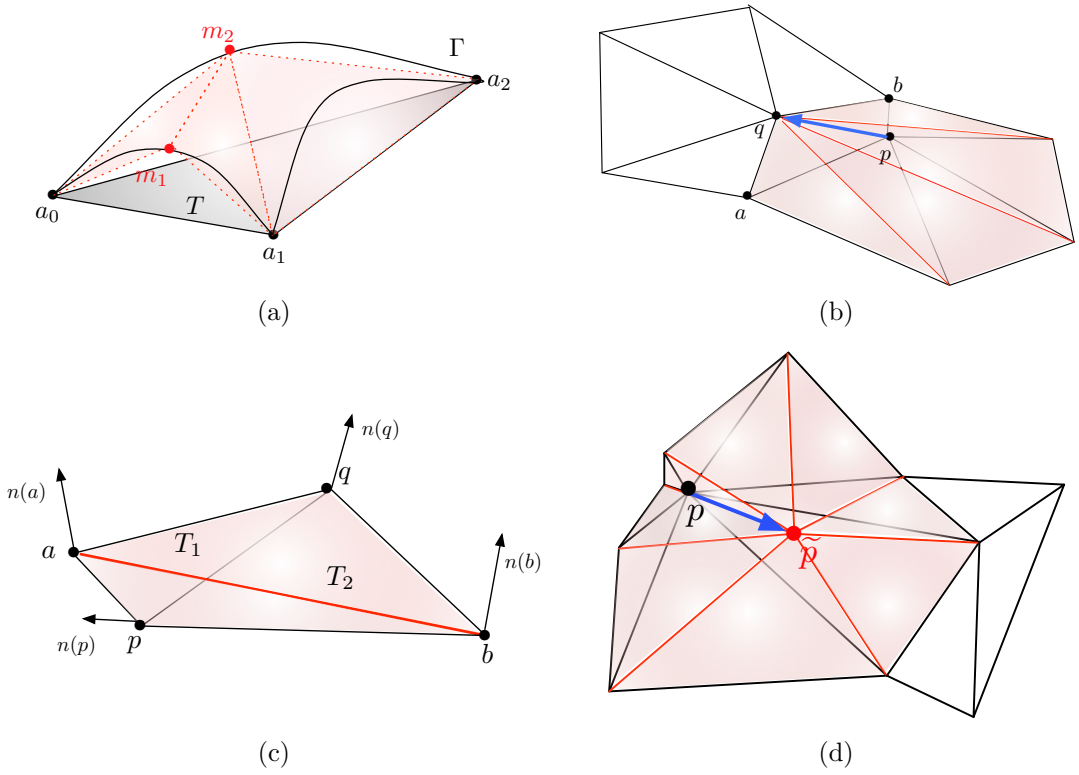


Figure 9.3: The four local remeshing operators, applied on a boundary configuration. In all four pictures, the previous configuration is shown in black, and the resulting one in red: *(a)* split of two edges a_0a_1, a_0a_2 of a surface triangle T, *(b)* collapse along a boundary edge pq; *(c)* swap of a boundary edge pq; *(d)* relocation of a vertex p to an improving position $\widetilde{p}$.

These operators serve different purposes: while the first two are mainly 'sampling operators', insofar as they make it possible to reach a desired element density in terms of a user-defined size prescription, or of geometric approximation concerns, the last two are essentially quality improvement operators.

The way to combine them - however completely heuristic an issue - turns out to be at least as important as their individual performances in a remeshing algorithm. Without delving into details, here is the outline of the global strategy which showed the best efficiency in our study:

1. In a first stage, operations are focused on modifying $\mathcal{T}_{temp}$ into a 'geometric mesh' $\widetilde{\mathcal{T}_{temp}}$ of D, with respect to Ω: $\widetilde{\mathcal{T}_{temp}}$ may still be very ill-shaped, but it encloses a discretization of a close approximation to the continuous geometric model for Ω. This stage mostly involves edge split and collapse operations.
2. A *size map* $h : D \to \mathbb{R}$ is computed (it is actually stored at the vertices of $\widetilde{\mathcal{T}_{temp}}$), which describes the local desired size features for remeshing, based on curvature estimates at the vertices of $\partial\Omega$, and taking into account user-defined bounds for the minimal and maximal authorized edge lengths.
3. The intermediate mesh $\widetilde{\mathcal{T}_{temp}}$ is modified into the high-quality mesh $\mathcal{T}_\Omega$: edge splits and collapses are performed to reach the size feature expressed by the size map h; at first, only the configurations which are 'very much' deviant from the size prescription are considered, then the criteria become increasingly strict. These operations are intertwined with massive uses of edge swaps and vertex relocations, whenever they help in improving the overall quality of the mesh.

Remark 9.3. This remeshing algorithm can serve two additional purposes, illustrations of which are provided in section 9.6:
- It makes it possible to obtain a mesh of the optimal shape resulting from a shape and topology optimization procedure performed using the 'classical' level set method of [14, 319] on a fixed mesh of D; one could also imagine to carry on the shape optimization procedure from this point using the method presented in this chapter for a computation which hopefully enjoys an enhanced accuracy.
- It could also be used to produce a high-quality mesh of the final shape of the presented shape optimization process, as a first step towards its post-processing (e.g. in reverse engineering).

9.4 Accounting for shape evolution

9.4.1 A brief reminder of the level set method

The evolution of shapes is numerically tracked while they are under implicit representation, by using the level set method [245], originally introduced in the context of shape optimization in [14, 319].

Grossly speaking (see also [274], or to a lesser extent, chapter 1 for more details), let $\Omega(t) \subset \mathbb{R}^d$ be a domain, whose motion over a time period $[0,T]$ is driven by a velocity field $V : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$. At any time $t \in [0,T]$, let also $\phi(t,.)$ be a level set function associated to $\Omega(t)$. Using a *formal* argument, (implying notably that V and ϕ are smooth over $(0,T) \times \mathbb{R}^d$), the motion of $\Omega(t)$ is translated in terms of ϕ into the following level set advection equation (see Chapter 1):

$$\frac{\partial \phi}{\partial t}(t,x) + V(t,x) \cdot \nabla \phi(t,x) = 0 \text{ on } (0,T) \times \mathbb{R}^d. \tag{9.11}$$

If in addition V is consistently oriented along the normal to $\Omega(t)$, say $V(t,x) = v(t,x)\, n_{\Omega(t)}(x)$, for a certain scalar field v, where $n_{\Omega(t)} = \frac{\nabla\phi(t,.)}{|\nabla\phi(t,.)|}$ denotes (an extension of) the outer unit normal vector to $\partial\Omega(t)$, (9.11) is best rewritten as a Hamilton-Jacobi equation:

$$\frac{\partial \phi}{\partial t}(t,x) + v(t,x)|\nabla \phi(t,x)| = 0 \text{ on } (0,T) \times \mathbb{R}^d. \tag{9.12}$$

Unfortunately, in the present context (9.11) or (9.12) are impossible to solve as such, since at every time t, $V(t,.)$ (or equivalently $v(t,.)$) is a descent direction for a given functional of the domain J, and depends on global features of the evolving domain $\Omega(t)$ (hence of the level set function ϕ) in a highly non trivial way.

However, like in most applications, the time interval $(0,T)$ stands for a 'small' generic time period between two stages of an iterative process. Freezing V, i.e. assuming that $V(t,x) \approx V(0,x) =: V_0(x)$ over $[0,T]$ allows to transform (9.11) into a passive transport equation:

$$\frac{\partial \phi}{\partial t}(t,x) + V_0(x) \cdot \nabla \phi(t,x) = 0 \text{ on } (0,T) \times \mathbb{R}^d.$$

When $V(t,x) = v(t,x)\, n_{\Omega(t)}(x)$ is oriented along the normal to $\Omega(t)$, another possibility consists in freezing only the scalar field v over $[0,T]$, i.e. assuming $v(t,x) \approx v(0,x) =: v_0(x)$. Thus (9.12) becomes a passive (still non linear) Hamilton-Jacobi:

$$\frac{\partial \phi}{\partial t}(t,x) + v_0(x)|\nabla \phi(t,x)| = 0 \text{ on } (0,T) \times \mathbb{R}^d,$$

a structure which preserves the information that the velocity has a normal direction to $\partial\Omega(t)$ at any time.

9.4.2 Resolution of the level set advection equation on an unstructured mesh

Let us now consider the following passive transport equation of a scalar quantity ϕ:

$$\left\{ \begin{array}{ll} \frac{\partial \phi}{\partial t}(t,x) + V(x) \cdot \nabla \phi(t,x) = 0 & \text{for } (t,x) \in (0,T) \times \mathbb{R}^d \\ \phi(t=0,x) = \phi_0(x) & \text{for } x \in \mathbb{R}^d \end{array} \right. , \tag{9.13}$$

over a generic period of time $(0,T)$, where $V : \mathbb{R}^d \to \mathbb{R}^d$ is the (autonomous) velocity field, and $\phi_0 : \mathbb{R}^d \to \mathbb{R}$ is the initial state for ϕ.

As in [292] and chapter 7, we propose to rely on the *method of characteristics* to solve (9.13) (see [256] for additional details): under adequate regularity and growth hypotheses on V (and ϕ_0), the unique $\mathcal{C}^1$ solution ϕ to (9.13) can be proved to be:

$$\phi(t,x) = \phi_0\left(X(0,t,x)\right), \tag{9.14}$$

where $s \mapsto X(s,t,x)$ is the *characteristic curve* of V passing at x at time t, defined as the solution to the ODE:

$$\left\{ \begin{array}{l} \mathring{X}(s,t,x) = V(X(s,t,x)) \text{ for } s \in \mathbb{R} \\ X(t,t,x) = x \end{array} \right. , \tag{9.15}$$

which describes the trajectory of a particle driven be the velocity field V standing at x at time t.

In the numerical setting of this chapter, V and ϕ_0 are discretized as $\mathbb{P}^1$ finite element functions on a (simplicial) mesh $\mathcal{T}$ of the computational domain D, and an approximation ϕ_T of the solution ϕ to (9.13) at time $t = T$ is sought under the same form. To this end, we simply mimic formula (9.14): ϕ_T is computed at the vertices of $\mathcal{T}$, using the following formula:

$$\forall \text{ vertex } x \in \mathcal{T},\ \ \phi_T(x) = \phi_0(\widetilde{X}(0,t,x)),$$

where $\widetilde{X}(0,t,x)$ is a numerical approximation to $X(0,t,x)$, provided by a numerical integration of the ODE (9.15), e.g. using a first-order Euler's method, or a more accurate Runge-Kutta procedure.

9.4.3 Computation of a descent direction

From a given shape $\Omega \in \mathcal{U}_{ad}$, a descent direction $V \in \Theta_{ad}$ for the considered objective function $J(\Omega)$ is computed on a whole mesh $\mathcal{T}_\Omega$ of D which encloses an explicit discretization of Ω.

The generic expression (9.6) for the shape derivative of J suggests the immediate choice:

$$\forall x \in \partial\Omega,\ \ V(x) = -v_\Omega(x)\, n(x). \tag{9.16}$$

As we have seen, v_Ω depends on the solution to one or several systems of the form (9.1) posed on Ω, which can be accurately solved on the submesh $\mathcal{T}'_\Omega$ of D, using the finite element method. Unfortunately, the choice (9.6) for a descent direction turns out to be hazardous for two independent reasons:

– Formula (9.6) makes sense only on $\partial\Omega$, whereas we are in search for a velocity field V which is defined at least in a vicinity of $\partial\Omega$, as well to comply with the requirement $V \in \Theta_{ad}$ as to use V in the context of the level set method.
– Anyway, as exemplified in Theorem 9.1, the scalar field v_Ω generally depends on (traces of) derivatives of the solution u_Ω to (9.1) (and possibly on those of the adjoint state p_Ω), which may be quite irregular - in the theoretical framework as well as when it comes to their numerical approximation. This may endanger the numerical stability of the process.

As advocated by [64, 162] (see also Chapter 2, §2.2.3.3), an efficient way to address both problems at the same time consists in using as a descent direction the gradient of J associated to a different scalar product from the canonical one of $L^2(\Gamma)$.

More accurately, $\alpha > 0$ being a small 'extension - regularization' parameter, let us consider the functional space

$$H^1_{\Gamma_D \cup \Gamma_N}(D) = \left\{ w \in H^1(D),\ \ w = 0 \text{ on } \Gamma_D \cup \Gamma_N \right\},$$

and let $\widetilde{v} \in H^1_{\Gamma_D \cup \Gamma_N}(D)$ be the unique solution to the variational problem (see [64] for alternative choices):

$$\forall w \in H^1_{\Gamma_D \cup \Gamma_N}(D),\ \ \int_D (\widetilde{v} w + \alpha \nabla \widetilde{v} \cdot \nabla w)\ dx = \int_\Gamma v_\Omega w\ ds = J'(\Omega)(wn). \tag{9.17}$$

Consider now the choice:

$$\forall x \in D,\ \widetilde{V}(x) = \widetilde{v}(x)\, n(x),$$

where n is an extension to D of the normal vector field to $\partial\Omega$. Combining (9.17) with the asymptotic expansion (9.2) shows that $\widetilde{V}$ is still a descent direction for J. However, $\widetilde{V}$ intrinsically enjoys more regularity than v_Ω owing to the classical regularity theory for elliptic equations, and is inherently defined on the whole domain D.

In the numerical setting, $\widetilde{v}$ is easily computed by solving (9.17) with the classical finite element method carried out on mesh $\mathcal{T}_\Omega$, after computing v_Ω (the discretization of the right-hand side being straightforward since the computational mesh $\mathcal{T}_\Omega$ encloses an explicit discretization of $\partial\Omega$). The (vector) velocity field $\widetilde{V}$ is eventually derived once Ω has been given an associated level set function ϕ, by using the usual extension to the normal vector field $n = \frac{\nabla\phi}{|\nabla\phi|}$.

9.5 The global algorithm

Gathering the material introduced in the previous sections, we are now in position to outline the proposed general strategy for handling mesh evolution in the context of shape optimization (see Figure 9.4 for an illustration).

Start with an initial shape Ω^0, and a simplicial mesh $\mathcal{T}_{\Omega^0}$ of D in which Ω^0 is explicitly discretized.

For $n = 0, \ldots$ **till convergence**, the current shape Ω^n is known via a mesh $\mathcal{T}_{\Omega^n}$ of D, a submesh $\mathcal{T}'_{\Omega^n}$ of which is a mesh of Ω^n.

1. Compute the value of the scalar field v_{Ω^n} appearing in the shape derivative of the considered functional (9.6). This may involve one, or several finite element analyses for solving the state (9.1) and (possibly) adjoint systems, to be held on the part $\mathcal{T}'_{\Omega^n}$ of the mesh $\mathcal{T}_{\Omega^n}$ corresponding to Ω^n. The quantity v_{Ω^n} is defined only on $\partial\Omega^n$, i.e. in the numerically setting, on the discretization of $\partial\Omega^n$ which explicitly appears in both $\mathcal{T}_{\Omega^n}$ and $\mathcal{T}'_{\Omega^n}$.
2. Generate the signed distance function d_{Ω^n} to Ω^n on the whole mesh $\mathcal{T}_{\Omega^n}$ of D.
3. Extend v_{Ω^n} to a vector field V^n defined on the whole mesh $\mathcal{T}_{\Omega^n}$ of D, along the lines of Section 9.4.3.

4. Choose a descent step $\tau^n > 0$, and solve the following level set advection equation on $\mathcal{T}_{\Omega^n}$:

$$\begin{cases} \frac{\partial \phi}{\partial t}(t,x) + V^n(x) \cdot \nabla \phi(t,x) & \text{for } (t,x) \in (0,\tau^n) \times D \\ \phi(t=0,x) = d_{\Omega^n}(x) & \text{for } x \in D \end{cases}.$$

   This produces a new level set function $\phi^{n+1} := \phi(\tau^n, .)$ associated to the new shape Ω^{n+1}.
5. Obtain the meshed representation of Ω^{n+1} by using the algorithm of section 9.3.2 from the set of data $(\mathcal{T}_{\Omega^n}, \phi^{n+1})$. A new mesh $\mathcal{T}_{\Omega^{n+1}}$ of D is produced, which encloses a mesh $\mathcal{T}'_{\Omega^{n+1}}$ of Ω^{n+1}.
6. Evaluate $J(\Omega^{n+1})$. If $J(\Omega^{n+1}) < J(\Omega^n)$, Ω^{n+1} is retained as the new shape; else $\Omega^{n+1} = \Omega^n$. Then, go back to stage (5), decreasing the chosen value as for the time step.

Remarks 9.4.
- Of course, the previous description is merely a synthetic, computationally non efficient sketch of the proposed method; depending on the forms of the considered functional J and its derivative, several quantities (such as the solution u_{Ω^n} to the state equation (9.1)) may be computed only when evaluating $J(\Omega^n)$ at step (6), then stored for further use in the computation of the velocity field V^n during the subsequent step (1).
- This strategy could be (and already has been [304]) applied to different physical models involving free or moving boundaries.

9.6 Numerical examples

In this section, we present and discuss several numerical models, in two and three space dimensions, to assess the interest of the proposed mesh evolution method for shape optimization and illustrate some of its features. All the discussed computations were performed on a laptop computer (MacBook Pro, 2.66 GHz), and, unless stated otherwise, the coefficients for the elastic material are set to $E = 1$, $\nu = 0.3$.

9.6.1 Minimization of the compliance

For the sake of simplicity, in all this section, we assume that no body forces are applied, i.e. $f = 0$.

9.6.1.1 Two-dimensional examples

Unsurprisingly enough, our first examples are concerned with the design of elastic structures with maximal rigidity. The objective function $J(\Omega)$ under consideration is the *compliance*:

$$J(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\, dx = \int_{\Gamma_N} g \cdot u_\Omega\, ds, \tag{9.18}$$

where u_Ω is the solution to (9.1). This objective function is of the general form (9.3) with $j = 0$, $k(x,u) = g \cdot u$ on Γ_N, and $k(x,u) = 0$ on Γ. It is well-known that, in this case, the problem of minimizing J is self-adjoint, i.e. the function p_Ω involved in the expression (9.4) for $J'(\Omega)$, solution to (9.5), boils down to $p_\Omega = -u_\Omega$. So that the problem is not trivial, a volume constraint is incorporated under the form of a penalization by a fixed Lagrange multiplier ℓ, as explained in remark 9.1.

We start with the benchmark *Cantilever* test case: in a working domain D of dimensions 2×1, a beam is clamped near its top and bottom left corners, and surface loads $g = (0, -1)$ are applied on a small area located at the centre of its right-hand side (see the details on Figure 9.5). The Lagrange multiplier associated to the volume constraint is set to $\ell = 3$, and 200 iterations of the algorithm of Section 9.5 are performed. Each mesh $\mathcal{T}_{\Omega^n}$ of D arising in the course of the process has approximately 1500 vertices (and twice as many triangles), and the whole computation takes about 3 minutes. Several intermediate shapes are displayed on

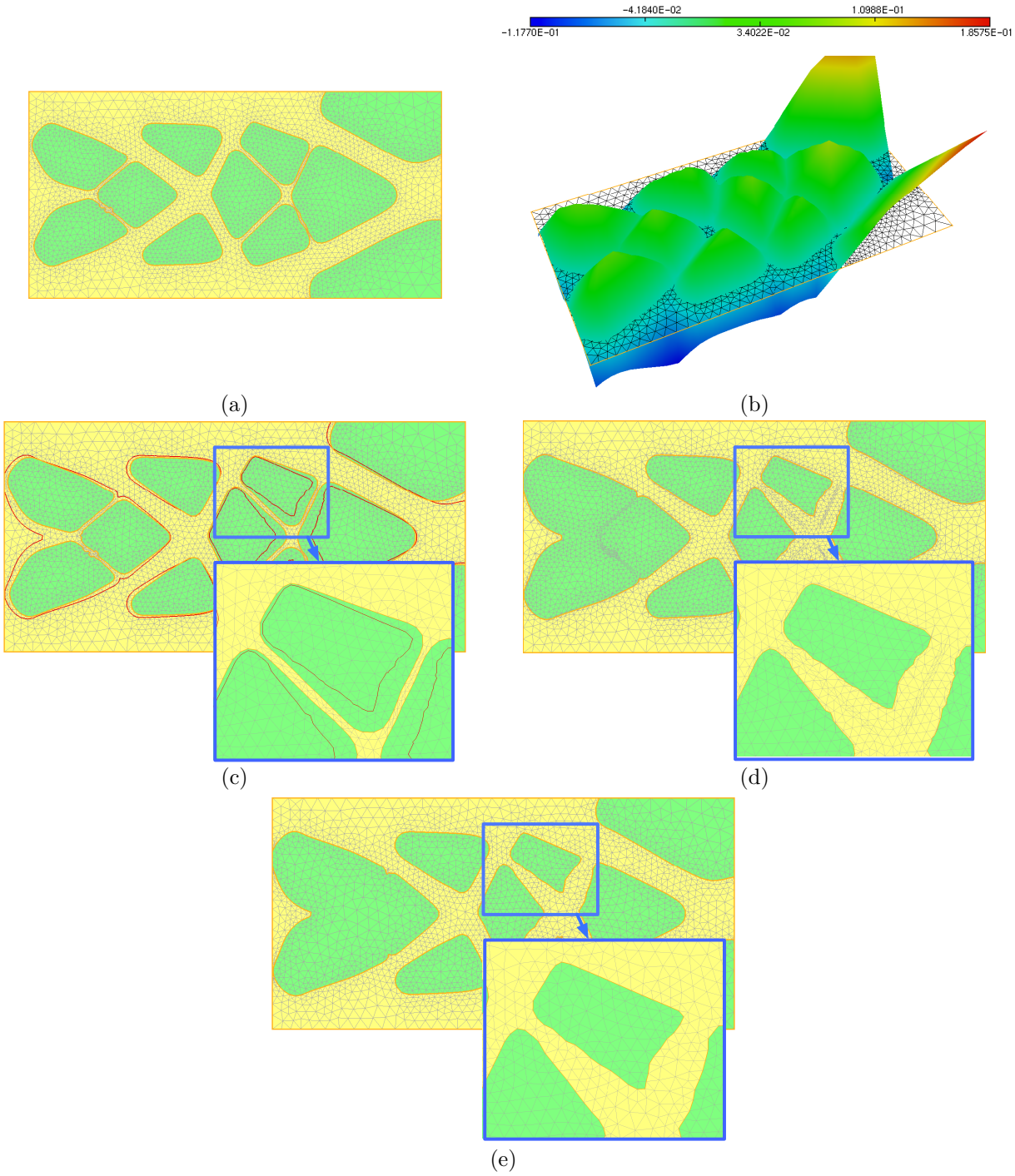


Figure 9.4: *(a)* The mesh $\mathcal{T}_{\Omega^n}$ of D accounting for a shape Ω^n; *(b)* the graph of the corresponding level set function ϕ^n; *(c)* advection of ϕ^n according to the velocity field V^n on $\mathcal{T}_{\Omega^n}$; the 0 isoline of the level set function ϕ^{n+1} for the new shape Ω^{n+1} is shown in red and is not yet discretized in the computational mesh; *(d)* explicit discretization of this 0 level set; the obtained mesh $\mathcal{T}_{temp}$ is very ill-shaped; *(e)* high-quality mesh $\mathcal{T}_{\Omega^{n+1}}$ in which the new shape Ω^{n+1} is discretized.

Figure 9.5, and the convergence history for the aggregated objective functional $\mathcal{L}(\Omega) := J(\Omega) + \ell \operatorname{Vol}(\Omega)$ is reported on Figure 9.7, left.

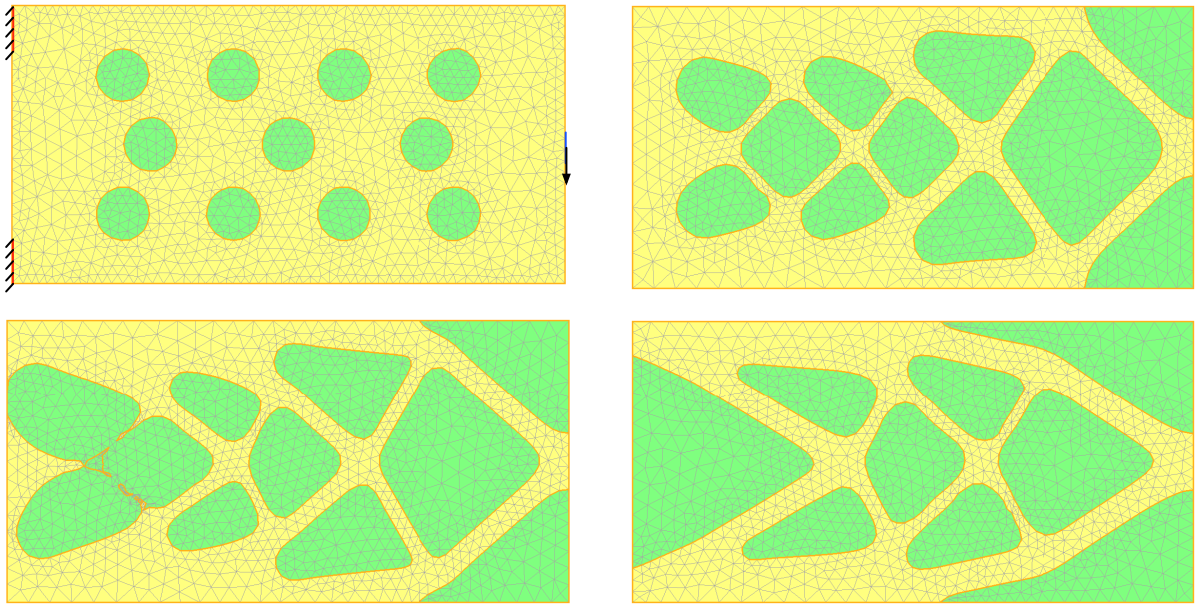


Figure 9.5: *(From top to bottom)* Initial (with boundary conditions), 30^{th}, 72^{th} and final iterations of the $2d$ cantilever test case. The 'inner' domains Ω^n are displayed in yellow, and the 'outer' ones $D \setminus \Omega^n$ in green. Note the ongoing topological change at the 72^{th} iteration.

We observe that the shape has been able to change topology without much trouble, while it is exactly meshed at each iteration of the process.

The very same strategy is applied to another benchmark example in structural optimization, namely that of the *optimal mast*: in a T-shaped working domain D, of height 120, width 80 at the top and 40 at the bottom, a mast is clamped around its bottom-left and bottom-right corners, and submitted to surface loads $g = (0, -1)$ around the corners on its arms (see Figure 9.6). Here, the Lagrange multiplier associated to the volume constraint is set to $\ell = 1$, and 100 iterations of the proposed algorithm are performed. Each intermediate mesh has about 8000 vertices, and the whole computation takes about 5 minutes. Results are shown on Figure 9.6, and the convergence history for the weighted sum of the compliance and the volume of shapes is reported on Figure 9.7, right.

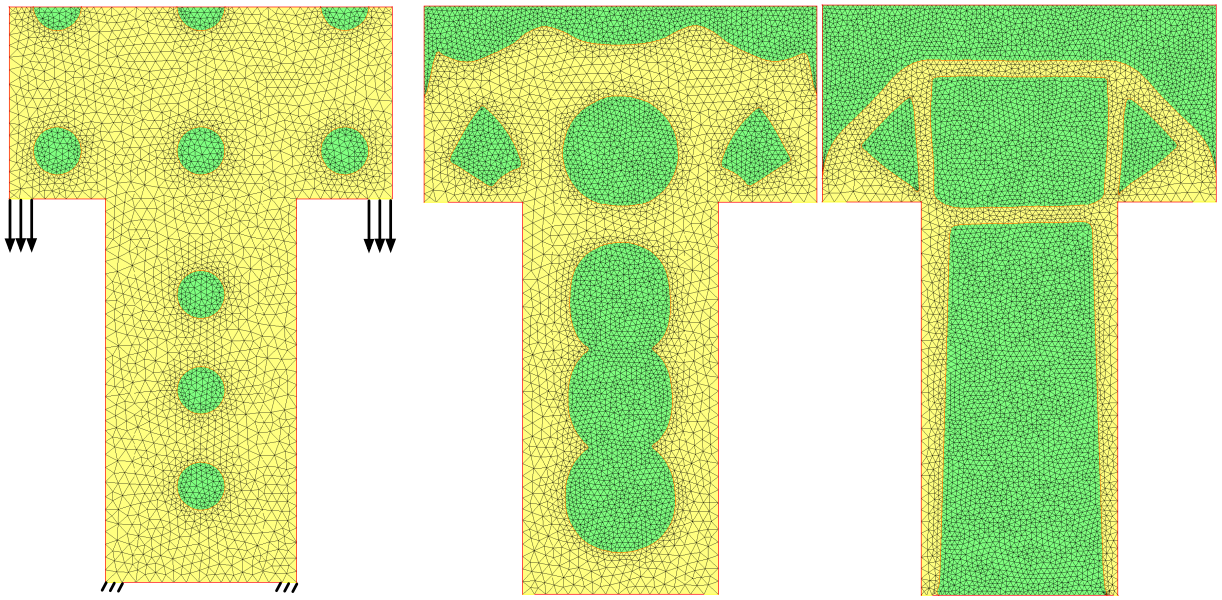


Figure 9.6: *(From left to right)* Initial (together with boundary conditions), 30^{th} and final iterations of the optimal mast test case.

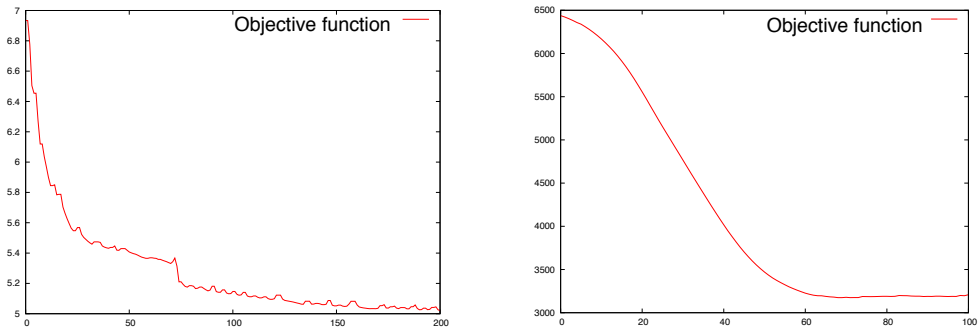


Figure 9.7: *(Left)* convergence history for the $2d$ cantilever test case, *(right)* convergence history for the $2d$ mast test case.

9.6.1.2 A two-dimensional example using the topological derivative

Hadamard's boundary variation method allows to describe the evolution of shapes via deformations of their boundaries. Theoretically speaking, the various shapes obtained in the course of such an evolution process are all diffeomorphic to one another; in particular, they share the same topology. In numerical practice, a small abuse in this setting allows holes to merge (in $2d$), or walls to collide into handles (in $3d$), but holes can never be nucleated in the bulk parts of shapes; this results in a strong dependency of the final design on the topology of the initial one, especially in $2d$. To circumvent this difficulty, the works [16, 65], based on results of [125, 147, 289], proposed to incorporate an altogether different information to the process, that of the sensitivity of a shape with respect to the nucleation of a small hole, which we briefly outline now.

Definition 9.2. *Let Ω be a shape, $x \in \Omega$ a fixed point. For $\rho > 0$ small enough, denote $\Omega_\rho := \Omega \setminus (x + \rho\omega)$, where ω stands for the unit ball in $\mathbb{R}^d$. A real-valued function J of the domain admits a topological derivative*

$D_T J(\Omega)(x)$ at x if there exists a continuous function $f : \mathbb{R} \to \mathbb{R}$, with $f(0) = 0$ such that the following expansion holds in the neighborhood of $\rho = 0$:

$$J(\Omega_\rho) = J(\Omega) + D_T J(\Omega)(x)\, f(\rho) + \mathrm{o}\left(f(\rho)\right), \text{ with } \frac{\mathrm{o}(f(\rho))}{f(p)} \overset{\rho \to 0}{\longrightarrow} 0.$$

The following result, proved in [147], gives the topological derivative of the compliance:

Theorem 9.2. *For any shape Ω, the compliance functional $J(\Omega)$ defined by (9.18) admits a topological derivative at any point $x \in \Omega$, given by the following formula:*

$$D_T J(\Omega)(x) = \frac{\pi(\lambda + 2\mu)}{2\mu(\lambda + \mu)} \left(4\mu A e(u_\Omega) : e(u_\Omega) + (\lambda - \mu)\mathrm{tr}\left(A e(u_\Omega)\right) \mathrm{tr}\left(e(u_\Omega)\right)\right)(x)$$

By definition, nucleating an infinitesimally small hole in a shape Ω at a point x where $D_T J(\Omega)(x)$ is negative decreases the value of J. As initially proposed in [16], we then consider coupling the shape optimization method of section 9.5 with a periodic use of this topological sensitivity information: every n_{top} iterations, the topological derivative $D_T J(\Omega^n)$ of the actual shape Ω^n is evaluated, and a small percentage (typically, we took the value of 2%) of the elements where it is most negative are removed from Ω^n.

As an example, consider the *optimal bridge* test case of Figure 9.8: a bridge, enclosed in a rectangle D of dimensions 2×1.2 is clamped near its bottom-left and bottom-right corners, and is submitted to surface loads $g = (0, -1)$, applied on a small region around the middle of its bottom side. The Lagrange multiplier for the volume constraint is set to $\ell = 0.1$, and 500 iterations of the aforementioned coupling strategy are performed, with a stage of topological sensitivity analysis replacing the sensitivity analysis using Hadamard's method every $n_{top} = 10$ iterations. Each mesh of D has about 2500 vertices, and the computation takes less than 10 minutes. Results are displayed on Figure 9.8, and the convergence history is that of Figure 9.9.

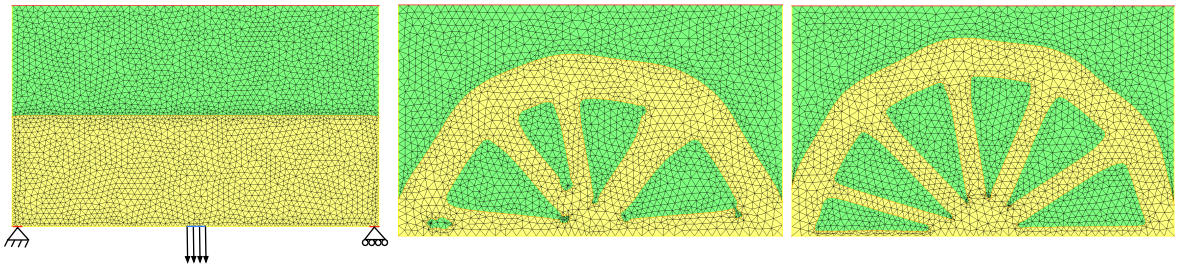


Figure 9.8: *(From left to right)* Initial (together with boundary conditions), 60^{th} and final iterations of the $2d$ optimal bridge test case, using information from the topological derivative.

Conspicuously, several holes have been nucleated in the course of evolution. Note also that the initial symmetry in the shape has been lost. We shall repeatedly witness this phenomenon in the following (in a less spectacular way however).

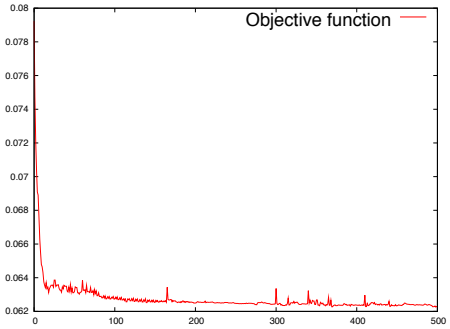


Figure 9.9: Convergence history for the $2d$ optimal bridge test case, using the coupling strategy between shape and topological sensitivity analyses.

9.6.1.3 $3d$ examples

We now turn to three-dimensional test cases, still in the context of compliance minimization under a volume constraint.

Our first example is a cantilever; the computational domain D is a rectangle of dimensions $2.4 \times 5 \times 3$, and the considered shapes are clamped at their right-hand side, while being subject to surface loads, applied on a small area near the centre of their left-hand side (see Figure 9.10). The Lagrange multiplier for the volume constraint is $\ell = 0.05$ and 80 iterations of the strategy presented in Section 9.5 are performed. Each mesh has about 16000 vertices (thus approximately six times more tetrahedra), and the whole computation takes about an hour. Results are displayed on Figure 9.10, and the associated convergence history is represented at Figure 9.13, left. Note that some intermediate shapes may show dramatic stretching, and that the resulting shape is nevertheless very regular.

Let us now consider the bridge model, depicted in Figure 9.11: in a working domain D of dimensions $40 \times 200 \times 50$, the considered shapes are clamped on two symmetric parts of their bottom side, and surface loads are applied all over their superior part; the Lagrange multiplier for the volume constraint is $\ell = 100$ and 70 iterations of our algorithm are performed. The average number of vertices for shapes is 9000, an the computation takes about 45 minutes. See Figure 9.13, right for the convergence history.

Figure 9.12 exemplifies Remark 9.3, that the remeshing algorithm outlined in Section 9.3.2.2 can be used to produce a high-resolution mesh of the optimal shape: in this particular case, the final shape (or more accurately, the last mesh of D) is enriched into a now one enjoying about 70000 vertices.

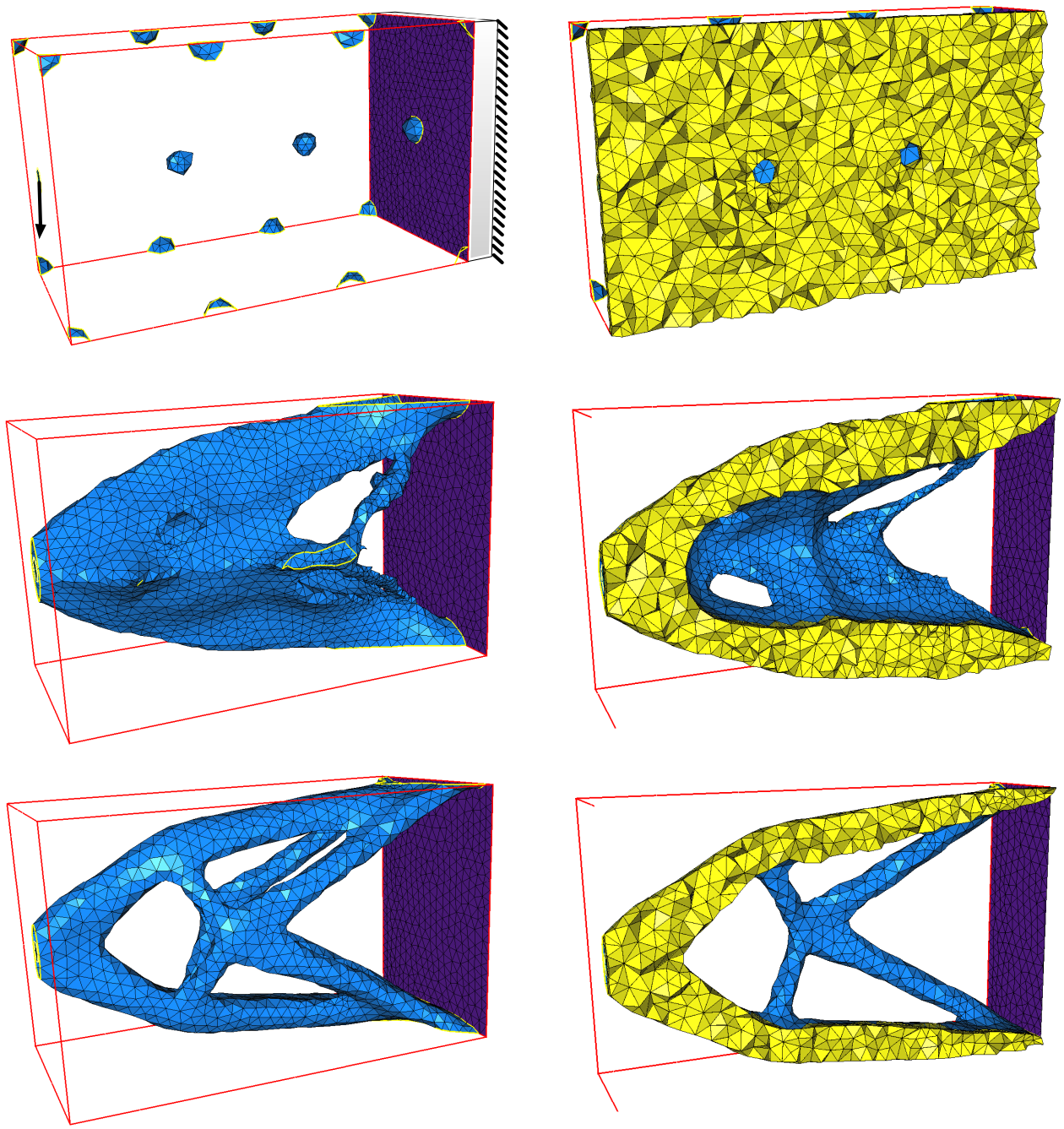


Figure 9.10: *(From top to bottom)* Initial (with boundary conditions), 30^{th} and final (80^{th}) iterations of the $3d$ Cantilever test case. Only the boundary $\partial\Omega^n$ of each shape Ω^n is displayed in the left column, and only the interior part of each mesh $\mathcal{T}_{\Omega^n}$ is displayed in the right column.

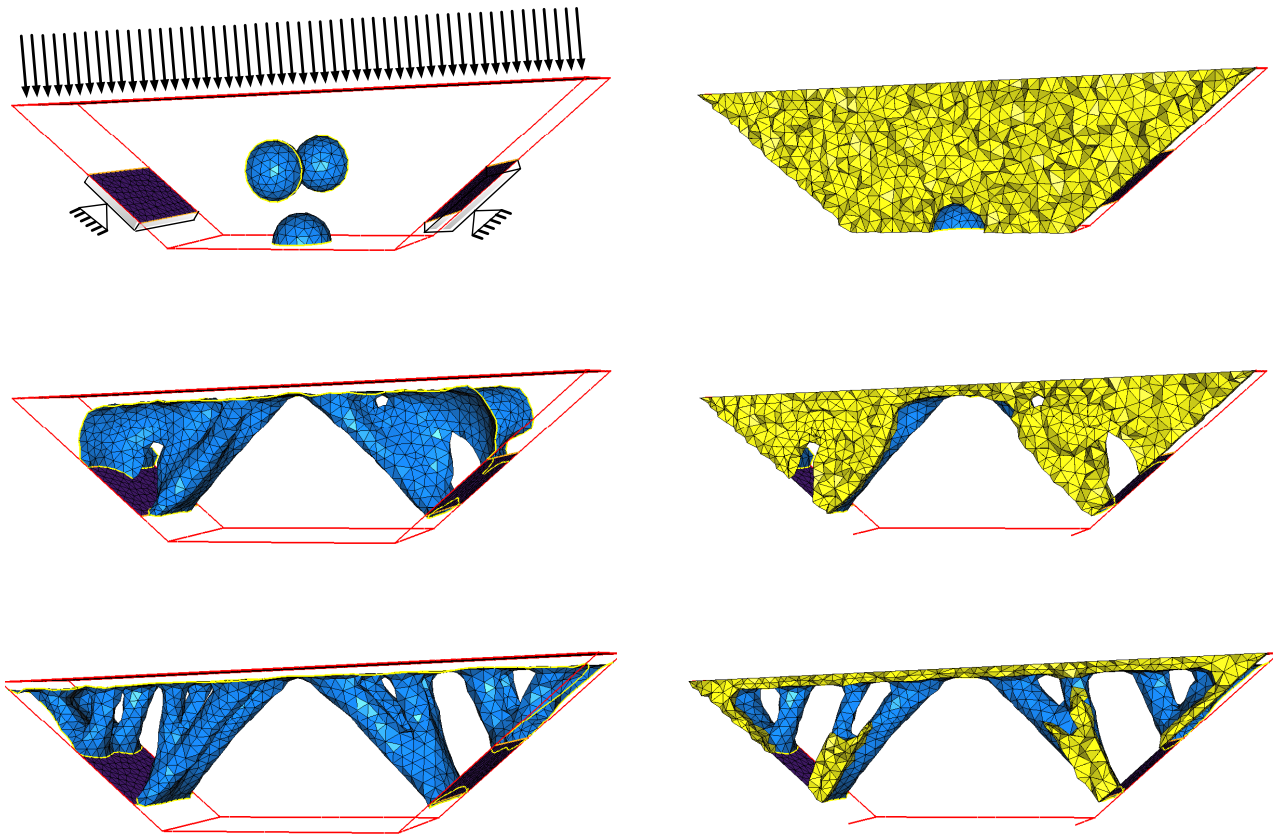


Figure 9.11: *(From top to bottom)* Initial (with boundary conditions), 30^{th} and final (70^{th}) iterations of the $3d$ optimal bridge test case.

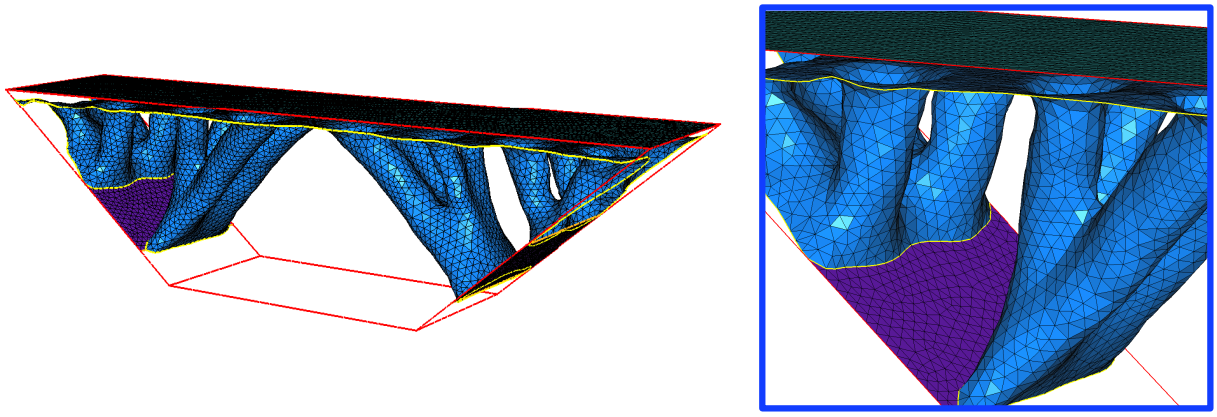


Figure 9.12: *(Left)* High-resolution mesh of the final shape obtained in the $3d$ optimal bridge example, *(right)* zoom on the surface mesh.

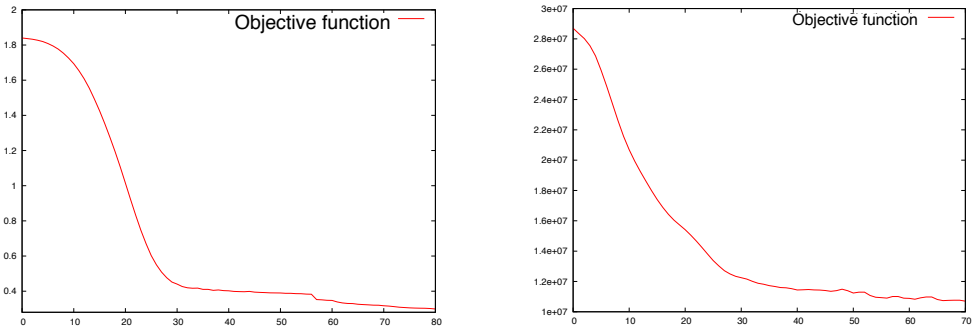


Figure 9.13: *(Left)* convergence history for the $3d$ Cantilever test-case, *(right)* convergence history for the $3d$ Optimal Bridge test-case.

9.6.2 Multi-loads compliance minimization

This section stays framed within the context of compliance minimization, except that we are now leaving room for the possibility that several independent load cases may exert on the considered shapes.

More specifically, to put things into the general context of section 9.2, let $f_i \in L^2(\mathbb{R}^d)^d$, $i = 1, ..., N$ be N body forces, and $g_i \in H^1(\mathbb{R}^d)^d$ be N surface loads, all of them being applied on the same non-optimizable subset Γ_N of the boundary of shapes in $\mathcal{U}_{ad}$ (of course, each g_i may vanish on a different subset of Γ_N). For any $\Omega \in \mathcal{U}_{ad}$, denote by $u_{\Omega,i} \in H^1(\Omega)^d$ the unique solution to:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(u)) & = & f_i & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ Ae(u)n & = & g_i & \text{on } \Gamma_N \\ Ae(u)n & = & 0 & \text{on } \Gamma \end{array}\right. .$$

As in [12], the problem of finding the most rigid shape with respect to the N load cases is expressed as that of minimizing the sum of the individual compliances associated to each load case; the considered objective function is thus:

$$J(\Omega) = \sum_{i=1}^{N} \int_\Omega Ae(u_{\Omega,i}) : e(u_{\Omega,i})\, dx = \sum_{i=1}^{N} \left(\int_\Omega f_i \cdot u_{\Omega,i}\, dx \int_{\Gamma_N} g_i \cdot u_{\Omega,i}\, ds \right). \tag{9.19}$$

As an example, we consider the *optimal chair* test case, as represented in Figure 9.14: shapes are embedded in a box of dimensions $0.7 \times 0.5 \times 1$, and submitted to two independent load case: the first one $g_1 = (0, -1)$ is applied on the seat of the chair, and the second one $g_2 = (-1, 0)$ is applied on the back (in both cases, no body forces are applied). Function (9.19) is minimized after a volume constraint has been incorporated under the form of a fixed Lagrange multiplier $\ell = 200$: 100 iterations of our algorithm are performed, for a computational time of approximately 90 minutes (each mesh enjoying about 11000 vertices). Results are displayed on Figure 9.14 (see also Figure 9.15 for the convergence history).

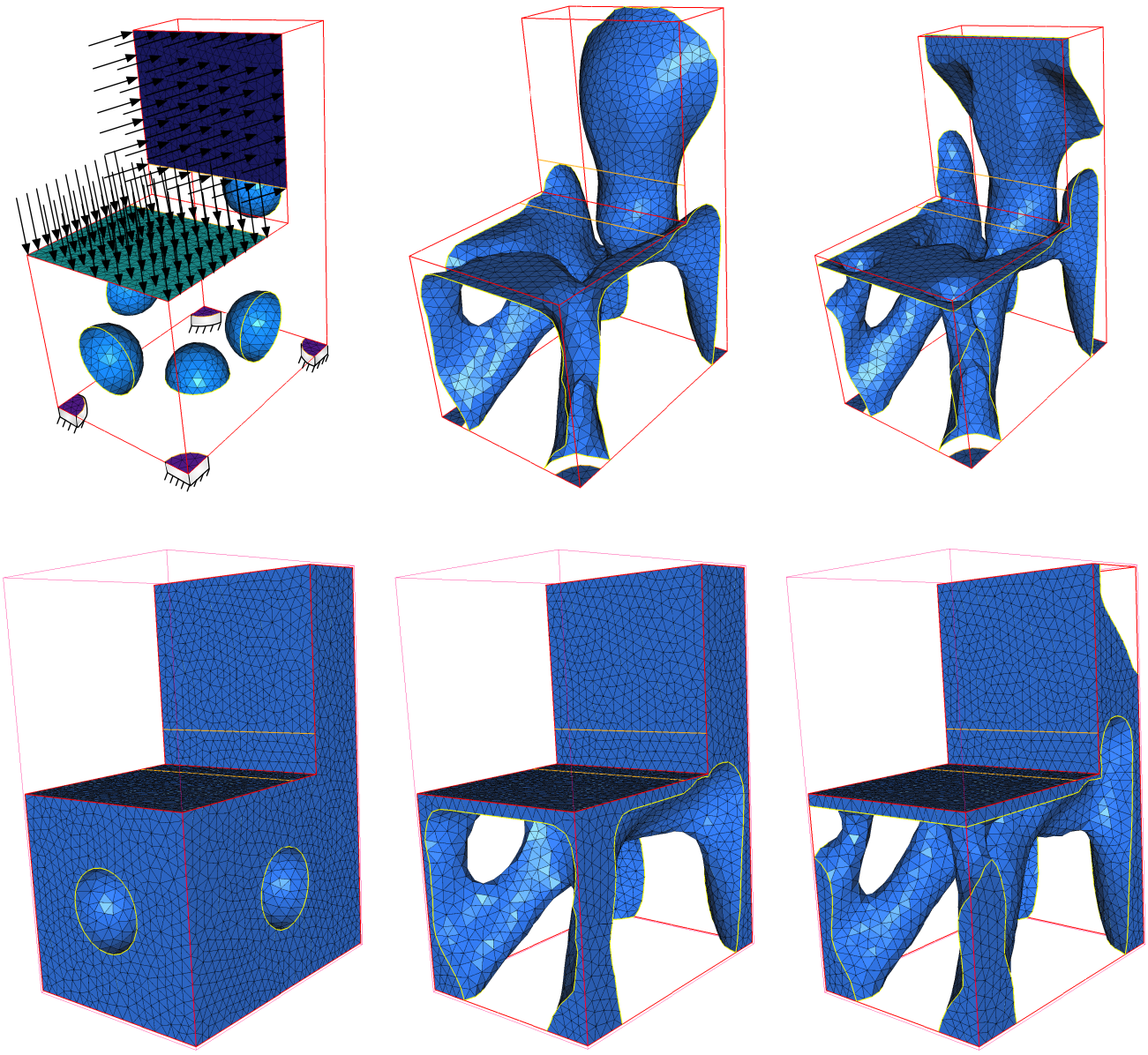


Figure 9.14: *(From left to right)* Initial (with boundary conditions), 50^{th} and final (100^{th}) iterations of the $3d$ optimal chair test case. To help visualization, the whole boundaries of shapes (and not only that corresponding to the 0 level set of the evolving implicit function) are displayed on the lower row.

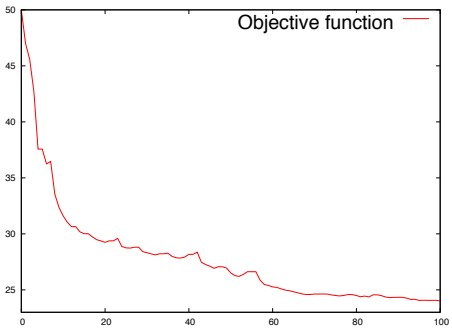


Figure 9.15: *Convergence history for the optimal chair test case.*

9.6.3 Chaining topological and geometric optimization

In this section, we elaborate on the first point of Remark 9.3 about the possibility and benefits of combining the mesh evolution method for shape optimization of this chapter with the 'classical' level-set based structural optimization method on a fixed mesh, as in [14, 319] (see also the description in Chapter 1, §2.3).

More precisely, we examine the following two-stage strategy for minimizing a function $J(\Omega)$:

1. *Optimization of the shape using the 'classical' level set method:* the working domain D is endowed with a *fixed* mesh $\mathcal{T}$ (which may be simplicial, Cartesian, etc...), and shapes $\Omega \subset D$ are consistently described by a corresponding level set function ϕ (discretized on $\mathcal{T}$). The main source of approximation of this class of shape gradient algorithms (which is also its fundamental difference with the method of this chapter) is that no mesh of a shape $\Omega \subset D$ is available when it comes to performing the necessary computation of the solution u_Ω to (9.1) (or the adjoint state p_Ω) for the derivation of a descent direction for J. Hence, the *Ersatz material approach* is used, whereby the void part $D \setminus \Omega$ is filled with a 'very soft' material of Hooke's law εA, $\varepsilon \ll 1$, so that the problem (9.1) is approximated by the following one, posed on D:
$$\left\{ \begin{array}{rcll} -\mathrm{div}\,(A_\Omega e(u)) & = & f & \text{in D} \\ u & = & 0 & \text{on } \Gamma_D \\ A_\Omega e(u)n & = & g & \text{on } \Gamma_N \end{array} \right. ,$$
where the total Hooke's tensor A_Ω is defined over D as:
$$\forall x \in D,\ \ A_\Omega(x) = \left\{ \begin{array}{ll} A & \text{if } x \in \Omega \\ \varepsilon A & \text{if } x \in D \setminus \Omega \end{array} \right. .$$
This step ends with an 'optimal' shape $\widetilde{\Omega}$, known as a level set function $\widetilde{\phi}$, defined on mesh $\mathcal{T}$.
2. *Optimization of the shape using the mesh evolution method:* The resulting shape $\widetilde{\Omega}$ from the first step is explicitly discretized in the computational mesh $\mathcal{T}$, which produces a new mesh $\mathcal{T}_{\widetilde{\Omega}}$ of D in which $\widetilde{\Omega}$ is enclosed as a submesh. From this point, the algorithm of section 9.5 is applied, retaining exactly the same parameters (loads, Lagrange multipliers, etc...) as in the first stage (except of course for the coefficient ε which no longer serves any purpose), and produces a new 'optimal' shape, say Ω^*.

To appraise this procedure, we limit ourselves to the case of the aggregated sum $\mathcal{L}(\Omega)$ of the compliance (9.18) and the volume $\mathrm{Vol}(\Omega)$ as an objective function of the domain, and first consider the two-dimensional cantilever example of section 9.6.1.1, using the same parameters as those introduced then.

The first stage is performed on a fixed triangular mesh $\mathcal{T}$ of the working domain D containing 6518 vertices. The coefficient for the weak material is $\varepsilon = 1.e^{-3}$, and 200 iterations of the `FreeFem++` implementation of the fixed mesh level set method described in Chapter 5, §5.5.2.1 are performed to produce the intermediate 'optimal' shape $\widetilde{\Omega}$.

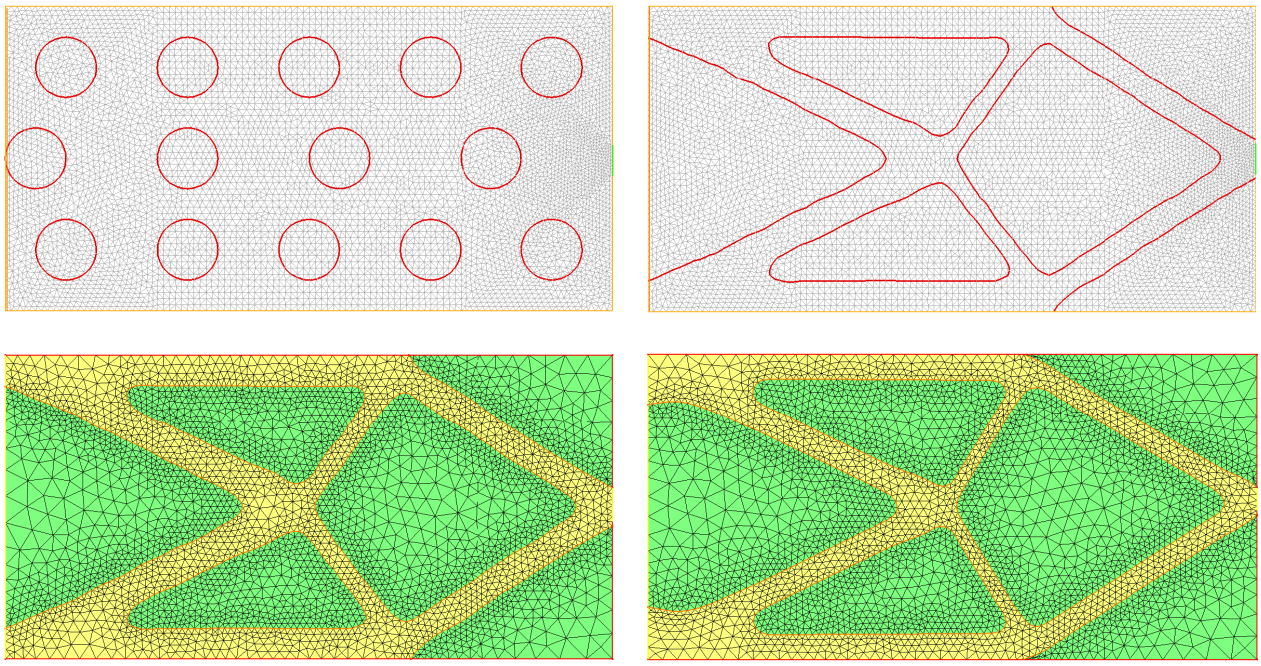


Figure 9.16: *(Top)* Initial and final iterations of the $2d$ Cantilever test case, using the level set method for shape optimization on a fixed unstructured mesh. The 0 level set accounting for the shape of interest is displayed in red. *(Bottom-left)* The 0 level set of obtained during the first stage is discretized into the computational mesh; *(Bottom-right)* final result of the combination of both methods.

	$\mathcal{L}(\widetilde{\Omega})$ with the Ersatz material approximation	$\mathcal{L}(\widetilde{\Omega})$ without the Ersatz material approximation	$\mathcal{L}(\Omega^*)$
$2d$ Cantilever	1.41182	1.632306	1.090604
$3d$ Cantilever, $\ell = 100$	188.5576	191.876896	162.661392
$3d$ Cantilever, $\ell = 200$	259.2133	258.723087	221.106033

Table 9.1: Values of the objective functions at different stages of the chaining strategy of the 'classical' level set method and the mesh evolution method for structural optimization.

As for stage (2), we apply the algorithm of section 9.5 for another 200 iterations, using the exact same parameters as in the first stage. All the meshes of this second sequence have more or less 3500 vertices. Results are displayed on Figure 9.16 (see Figure 9.18 for the convergence histories). Note that the respective shapes $\widetilde{\Omega}$ and Ω^* obtained at the end of stage (1) and (2) are qualitatively different, and that the final Ω^* is a noticeable improvement of the first 'optimal shape' $\widetilde{\Omega}$. Note also the non negligible gap between the values of $\mathcal{L}(\widetilde{\Omega})$ depending on whether it is computed by using the Ersatz material approximation or not (see Table 9.1).

The same strategy is applied to the $3d$ Cantilever test case. Here, the setting of the problem is slightly different from that of Section 9.6.1.3: the working domain D is now a $2 \times 1 \times 1$ box; shapes are clamped at their left-hand side and a point load is applied at the centre of the right-hand side.

During stage (1), D is equipped with a Cartesian mesh of size $40 \times 20 \times 20$ (18081 vertices), and the resulting optimal shapes are courtesy of G. Michailidis [228]. Two examples are presented, associated to different Lagrange multipliers $\ell = 100$ and $\ell = 200$ for the volume constraint. Results are displayed on

Figure 9.17. In both cases, stage (2) converges within only ten iterations (hence, only the final values of the objective functions are recorded in Table 9.1), and the intermediate 'optimal' shapes $\widetilde{\Omega}$ are significantly improved (even by topological changes !) by the respective final 'optimal' shapes Ω^*. In the case $\ell = 100$ (resp. $\ell = 200$), the average number of vertices of the meshes arising during stage (2) is 15000 (resp.12000).

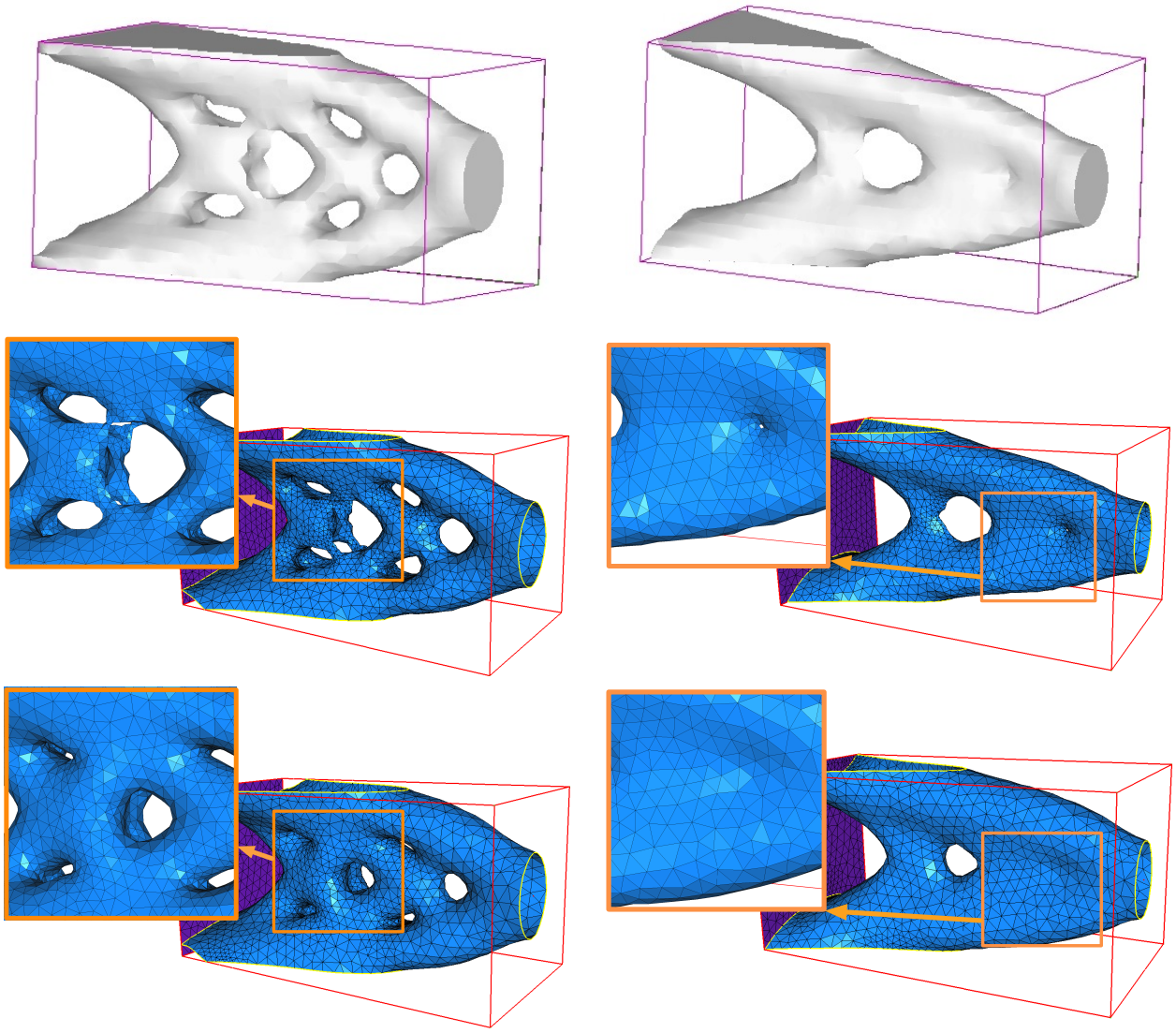


Figure 9.17: *(From top to bottom):* 0 level set of the implicit function for $\widetilde{\Omega}$, associated discretization in the mesh of D, and final shape Ω^*, using a Lagrange multiplier *(left column)* $\ell = 100$, *(right column)* $\ell = 200$.

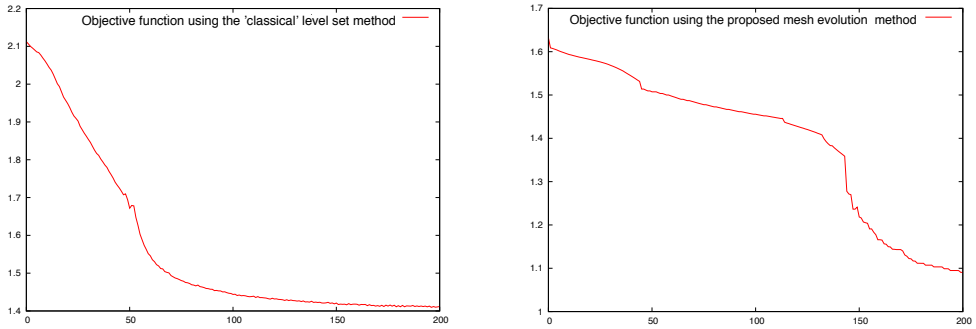


Figure 9.18: Convergence histories for the shape optimization procedure performed on *(left)* a fixed computational mesh, and *(right)* using the proposed mesh evolution procedure

9.6.4 Multi-materials compliance minimization

In this section, we discuss a model which does not exactly fit into the general framework of section 9.2, namely that of *multi-phase* shape optimization.

Let D be a fixed working domain composed of two materials, referred to as 0 and 1, with different properties reflected by their respective Hooke's tensor A_0 and A_1. Ω^0 and Ω^1 occupy the respective (smooth) subsets $\Omega^0 \subset D$ and $\Omega^1 = D \setminus \overline{\Omega^0}$; for the sake of simplicity we assume that Ω^0 lies 'far' from ∂D, i.e. $\partial\Omega^0 \cap \partial D = \emptyset$.

D is clamped on a region Γ_D of its boundary ∂D, and surface loads $g \in H^1(\mathbb{R}^d)^d$ are applied on another subset $\Gamma_N \subset \partial D$, disjoint from Γ_D. Body forces $f \in L^2(\mathbb{R}^d)^d$ are also exerted, and the displacement $u_{\Omega^0} \in H^1(D)^d$ of D induced by those stresses is the unique solution to:

$$\left\{ \begin{array}{rcll} -\mathrm{div}\,(A_{\Omega^0} e(u)) & = & f & \text{in D} \\ u & = & 0 & \text{on } \Gamma_D \\ A_{\Omega^0} e(u) n & = & g & \text{on } \Gamma_N \end{array} \right. ,$$

where the total Hooke's tensor A_{Ω^0} is defined over D as:

$$\forall x \in D,\ A_{\Omega^0}(x) = \left\{ \begin{array}{ll} A_0 & \text{if } x \in \Omega^0 \\ A_1 & \text{if } x \in \Omega^1 \end{array} \right. . \tag{9.20}$$

We consider the total compliance $J(\Omega^0)$ of the structure as an objective function of the subdomain Ω^0:

$$J(\Omega^0) = \int_D A_{\Omega^0} e(u_{\Omega^0}) : e(u_{\Omega^0})\, dx = \int_D f \cdot u_{\Omega^0}\, dx + \int_{\Gamma_N} g \cdot u_{\Omega^0}\, ds. \tag{9.21}$$

The following result, proved in [15], accounts for the shape derivative of J:

Theorem 9.3. *The shape derivative of the cost function J defined by (9.21) reads*

$$J'(\Omega^0)(\theta) = \int_\Gamma \mathcal{D}(u_{\Omega^0}, u_{\Omega^0})\, \theta \cdot n\, ds,$$

$$\mathcal{D}(u,u) = -\sigma(u)_{nn} : [e(u)_{nn}] - 2\sigma(u)_{n\tau} : [e(u)_{n\tau}] + [\sigma(u)_{\tau\tau}] : e(u)_{\tau\tau}. \tag{9.22}$$

where $\mathcal{M}_{nn}$, $\mathcal{M}_{n\tau}$ and $\mathcal{M}_{\tau\tau}$ are the minors of a tensor field $\mathcal{M} = \begin{pmatrix} \mathcal{M}_{\tau\tau} & \mathcal{M}_{\tau n} \\ \mathcal{M}_{n\tau} & \mathcal{M}_{nn} \end{pmatrix}$ expressed in an orthonormal basis of $\mathbb{R}^d$ obtained by assembling an orthonormal basis of tangent vectors τ to $\partial\Omega^0$ with its normal vector $n = n_{\Omega^0}$, $[\cdot] = \cdot^1 - \cdot^0$ denotes the jump through $\partial\Omega^0$, and $\sigma(u) = A_{\Omega^0}\, e(u)$.

As explained in Chapter 4, §4.3.2, this problem is very difficult to handle in a fixed mesh framework (at least without any change in the formulation), mainly because formula (9.22) brings into play the transmission conditions at the interface $\partial\Omega^0$, which cannot be accurately approximated using Lagrange finite element methods for solving (9.6.4) unless a mesh of D is used which features an explicit discretization of $\partial\Omega^0$. This is consequently a good opportunity to test the method at stake in this chapter.

Consider a box of dimensions $40 \times 200 \times 60$ as D, and assume it is clamped near its four bottom corners, and submitted to surface loads on a region near the center of its upper side. The working domain D is filled with two materials: material 0 is the one we have been using hitherto, i.e. its Young modulus and Poisson ratio are respectively $E_0 = 1$, and $\nu_0 = 0.3$; material 1 is weaker, and has a Young modulus $E_1 = 0.3$, and Poisson ratio: $\nu_1 = \nu_0 = 0.3$. The compliance (9.21) of the total structure D is minimized, and a constraint over the volume $\mathrm{Vol}(\Omega^0)$ of the stronger phase is imposed by means of a fixed Lagrange multiplier $\ell = 0.02$; 100 iterations of our algorithm are performed, and the results are displayed on Figure 9.20 (see also Figure 9.19 for the convergence history). As expected, the stronger material connects the regions where the shape is clamped to the one where loads are applied. A portion of the stronger material at the bottom of the beam allows it to better withstand bending.

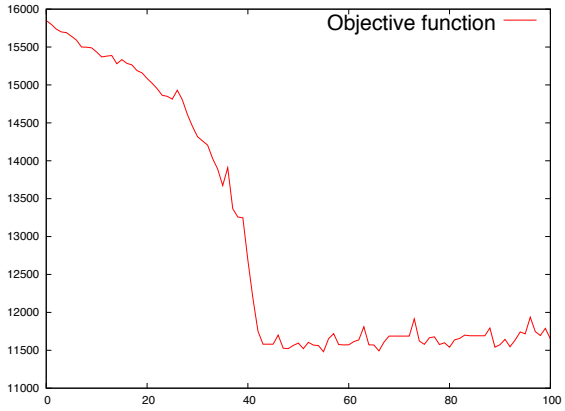


Figure 9.19: Convergence history for the multi-material $3d$ Beam test case.

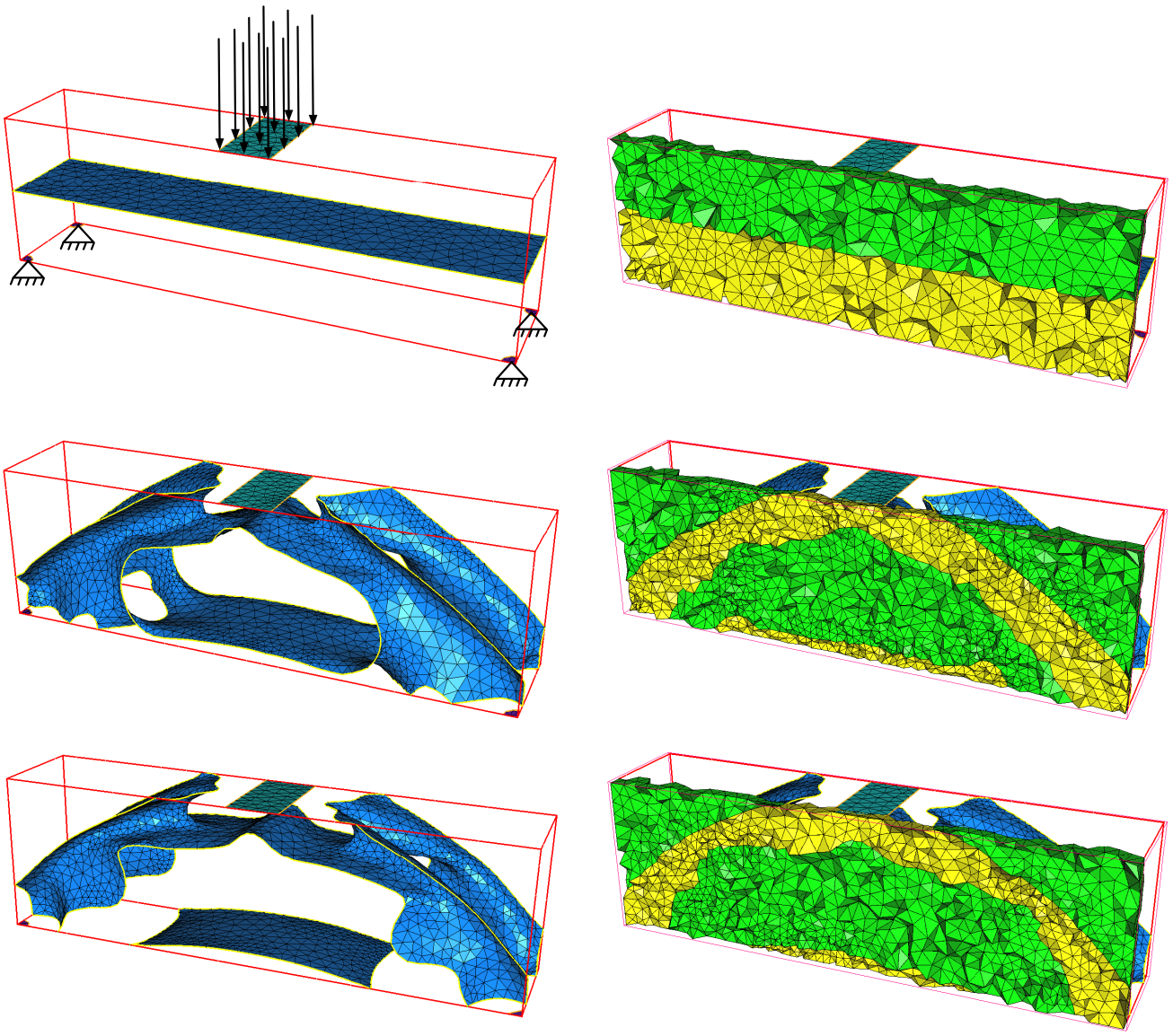


Figure 9.20: *(From top to bottom)* Initial (with boundary conditions), 50^{th} and final (100^{th}) iterations of the multiphase beam test case. The stronger material is the one displayed in yellow in the right-hand column.

9.6.5 Minimization of least-square criteria

9.6.5.1 A gripping mechanism

We now touch on the optimal design of a *compliant mechanism*, that is, a mechanism which is intentionally flexible, so that its displacement under a given external stress complies with a prescribed behavior.

This problem falls into the general framework of Section 9.2, and is modeled as that of minimizing the following least-square criterion over the displacement u_Ω of shapes (see e.g. [40]):

$$J(\Omega) = \int_\Omega k(x)|u_\Omega - u_0|^2 \, dx, \tag{9.23}$$

where $k \in L^\infty(\Omega)$ is a (non negative) localization factor, and $u_0 \in H^1(\mathbb{R}^d)^d$ is a target displacement.

This objective function is of the general form (9.3), with $j(x,u) = k(x)|u - u_0|$, and $k(x,u) = 0$. As far as the mathematical setting is concerned, the only difference between this case and those in the previous sections lies in that the optimization problem is no longer self-adjoint, i.e. formula (9.4) for the shape derivative $J'(\Omega)$ brings in an adjoint state p_Ω different from $\pm u_\Omega$.

We apply this example to the optimal design of a gripping mechanism, as described on Figure 9.21: the working domain D has dimensions 5×4, and the considered shapes are clamped on two small rivets, while being submitted to surface loads $g = \pm(0, 0.02)$ on small regions near the upper-right and bottom-right corners of D (no body force is applied). The target displacement u_0 for shapes is set to ± 0.5 so that their jaws will hopefully close, and the localization factor k is set to 1 in a small area near the jaws, and 0 elsewhere. As usual, a very small volume constraint is incorporated into the objective function under the form of a fixed Lagrange multiplier $\ell = 0.001$, and 100 iterations or our algorithm are performed; each mesh has about 5000 vertices, and the whole computation takes about 6 minutes. Results are displayed on Figure 9.21.

Admittedly, this test case proves very difficult; as evidenced by the convergence history in Figure 9.22, the algorithm finds itself stuck at several points, those at which the chosen descent step τ^n is chosen 'too large' in the sense that the shape ends disconnected around the two very thin areas. This is a real problem in the context of optimal design of compliant mechanisms, since those thin areas are precisely those which endow the sought flexibility to shapes, and will thus inevitably appear during the optimization process. Actually, the optimal pattern for this kind of behavior would be to create pointwise junctions between members of the shape, which is of course impossible in our context where shapes are 'massive'. Note that the 'classical' level set method only suffers from this drawback to a lesser extent: indeed, in this case, if the shape ends disconnected at some point, the presence of soft material between members still allows for the desired flexibility; it is our experience that, in this setting, shapes tend to consistently disconnect, then reconnect, which is impossible in our context. This explains why our algorithm seems to capture the correct trend in shapes (as the resulting displacement confirms), but produce a somewhat 'unconverged' shape, inasmuch as it seems still too 'massive'.

We believe that the problem could be better formulated by imposing a minimum thickness to the desired shapes, so that this trend to disconnect would be prevented (see [228] for extensive explanations about this issue). For the moment, we limit ourselves to the observation that imposing those regions where the thickness of shapes is already very small are no longer subject to optimization, and carrying on the optimization procedure in this way (which is by no means as efficient as performing the same optimization problem under a constraint over the minimum thickness of shapes) gives promising results in this direction (see Figure 9.23).

A similar test case is investigated in $3d$, whose details are reported on Figure 9.24. Here, the working domain is a box of dimensions $1 \times 1 \times 2$; the considered shapes are clamped on small regions of the upper and lower sides, and a force $g = (0.4, 0)$ is applied on a region around the centre of the right hand side of their boundaries. The localization function k equals 1 in two small areas near the jaws, 0 elsewhere, and the target displacement is $u_0 = \pm(0, 0, 0.1)$. A small Lagrange multiplier $\ell = 0.02$ is added for the volume

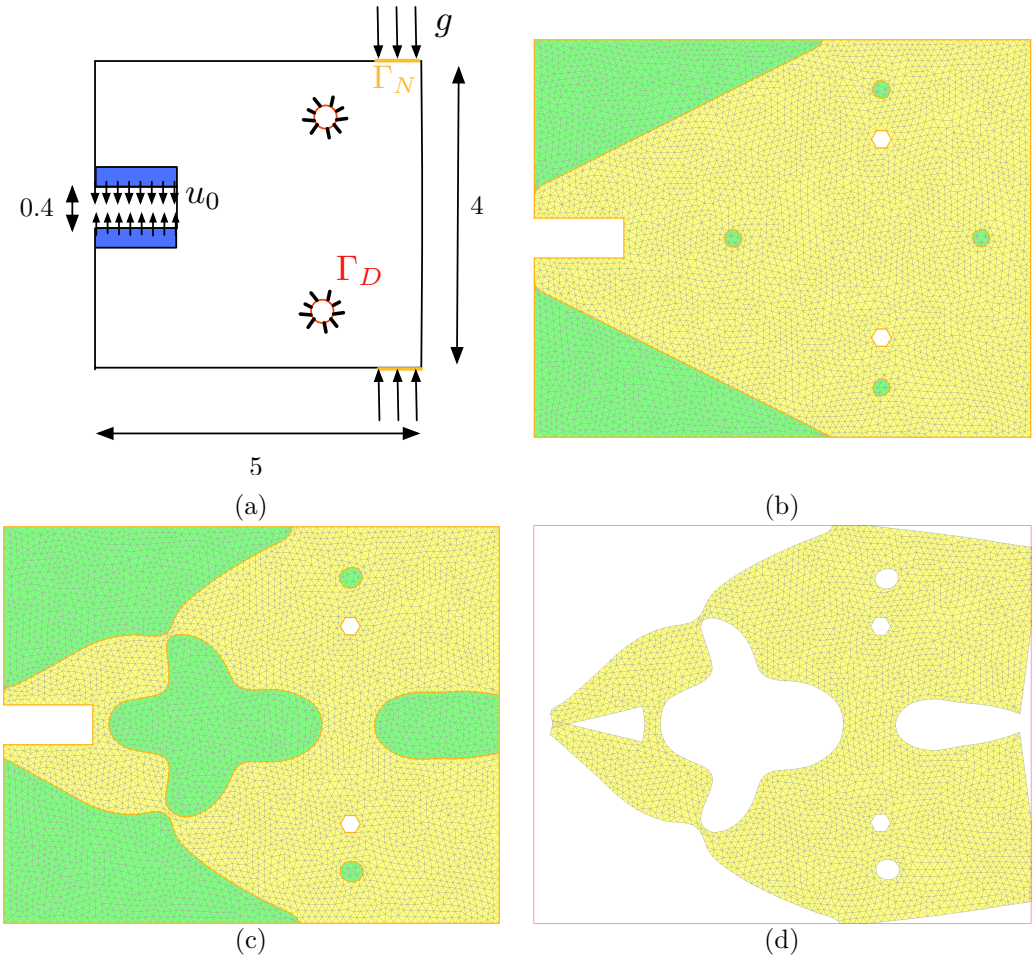


Figure 9.21: *(a)*: details of the two-dimensional gripping mechanism test case, *(b)*: initial shape, *(c)* final shape, and *(d)* deformed configuration.

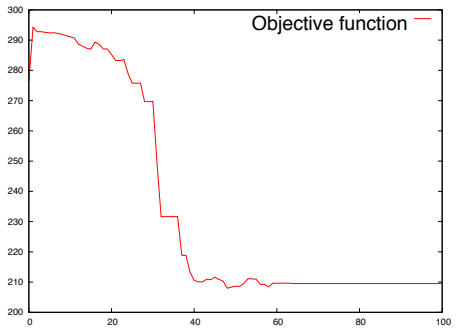


Figure 9.22: *Convergence history for the two-dimensional gripping mechanism example.*

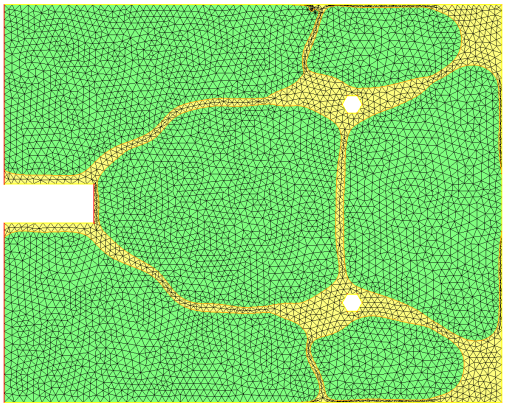


Figure 9.23: *Final shape of the two-dimensional gripping mechanism test case obtained after* 400 *additional iterations of our algorithm, while imposing that very thin parts are no longer subject to optimization.*

constraint, and 100 iterations of our algorithm are performed. Each mesh is worth about 12000 vertices, and the whole computation takes about 90 minutes. Results are displayed on Figures 9.24 and 9.25, and the prevailing observation is the same as in the $2d$ case, namely that the process could have gone further.

9.6.5.2 A three-dimensional example of worst-case design in shape optimization

The problem at stake in this section stems from a very different concern from that of the previous subsection; yet, their mathematical formulations and treatments happen to be quite similar. Our purpose is to apply the framework introduced in Chapter 5 for dealing with optimization of the worst-case scenario under 'small' perturbations to a simple problem of compliance minimization in three space dimensions.

Still in the context of section 9.2, we now foresee that unknown perturbations of 'small' amplitude m may alter the body force term f, which then ends up of the form $(f + \chi\xi e)$, where:

- $\chi \in L^\infty(\mathbb{R}^d)$ is the (known) characteristic function of the area where perturbations are expected,
- $\xi \in L^2(\mathbb{R}^d)$ is the amplitude of perturbations, whose sole known feature is an upper bound m: $||\xi||_{L^2(\mathbb{R}^d)} \leq m$.
- $\widehat{e} \in \mathbb{R}^d$ is a unit vector (fixed for simplicity), indicating the direction of the expected perturbations.

For any perturbation term $\xi \in L^2(\mathbb{R}^d)$, $||\xi||_{L^2(\mathbb{R}^d)} \leq m$, denote as $u_{\Omega,f+\chi\xi\widehat{e}}$ the solution to the perturbed linear elasticity system:

$$\left\{\begin{array}{rcll} -\mathrm{div}(Ae(u)) & = & f + \chi\xi\widehat{e} & \text{in } \Omega \\ u & = & 0 & \text{on } \Gamma_D \\ Ae(u)n & = & g & \text{on } \Gamma_N \\ Ae(u)n & = & 0 & \text{on } \Gamma \end{array}\right. .$$

In our search of the shape whose worst behavior when such perturbations are expected show maximal rigidity, the 'worst-case' functional $\mathcal{J}(\Omega)$ of interest is:

$$\mathcal{J}(\Omega) = \sup_{\substack{\xi \in L^2(\mathbb{R}^d) \\ ||\xi||_{L^2(\mathbb{R}^d)} \leq m}} \left(\int_\Omega (f + \chi\xi\widehat{e}) \cdot u_{\Omega,f+\chi\xi\widehat{e}} dx + \int_{\Gamma_N} g \cdot u_{\Omega,f+\chi\xi\widehat{e}} ds \right). \tag{9.24}$$

To get a fair approximation of functional $\mathcal{J}(\Omega)$, we have shown in Chapter 5 that linearizing the compliance term $\xi \mapsto \left(\int_\Omega (f + \chi\xi\widehat{e}) \cdot u_{\Omega,f+\chi\xi\widehat{e}} dx + \int_{\Gamma_N} g \cdot u_{\Omega,f+\chi\xi\widehat{e}} ds \right)$, then taking the supremum of the linearized quantity over $||\xi||_{L^2(\mathbb{R}^d)} \leq m$ produces the following *approximate worst-case functional*, hereafter denotes as

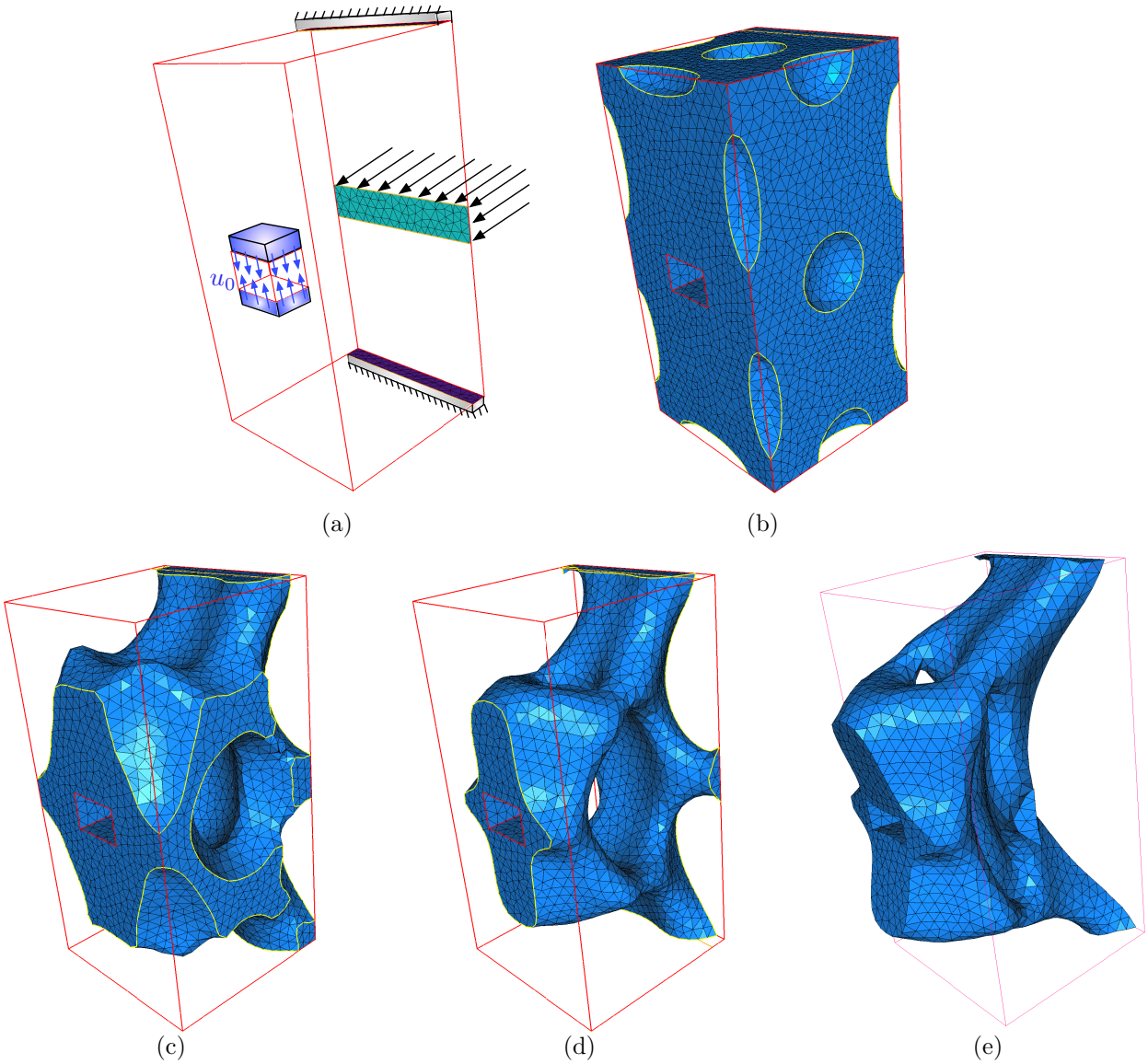


Figure 9.24: *(a)* details of the three-dimensional gripping mechanism test case, *(b)* initial shape, *(c)* 20^{th} iteration, *(d)* final shape (100^{th} iteration) and *(e)* deformed configuration of the final shape. The whole boundary of shapes is displayed.

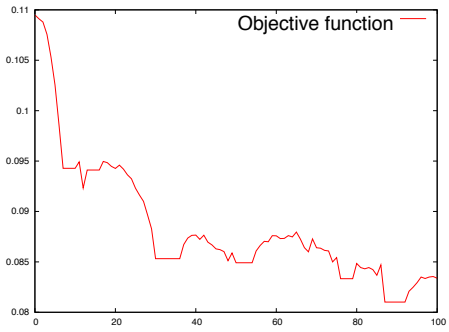


Figure 9.25: *Convergence history for the three-dimensional gripping mechanism example.*

$J(\Omega)$, defined by:

$$J(\Omega) = \int_\Omega f \cdot u_\Omega \, dx + \int_{\Gamma_N} g \cdot u_\Omega \, ds + 2m||\chi u_\Omega \cdot \widehat{e}||_{L^2(\mathbb{R}^d)},$$

where $u_\Omega := u_{\Omega,f}$ stands for the solution to the unperturbed system. This formula is very reminiscent of the objective function (9.23) studied in the previous section, and is easily put under the general form (9.3).

Note that, as discussed in Chapter 5, this problem has already been adressed in [84, 163] by a completely different method, which allows to work directly at the level of the exact worst-case function $\mathcal{J}(\Omega)$ given by (9.24), (and is thus certainly more accurate than the presented approximation), but is restricted to the case of the compliance as an objective function.

As an illustration, consider the model represented in Figure 9.26, which is a variation of the well-known *optimal mast* test case: a mast, enclosed in a T-shaped box of dimensions $40 \times 80 \times 126$ is clamped on its bottom side, and submitted to surface loads $g = (0, 0, -1)$ on two areas at the extremities of its arms. Body forces developed in the unperturbed state are set to $f = 0$, and vertical perturbations (i.e. $\widehat{e} = (0, 0, -1)$) are expected on the two yellow regions on the arms.

We perform 100 iterations of the proposed algorithm for three different values of m, namely $m = 0, 5, 10$, using, for the sake of simplicity, the same Lagrange multiplier $\ell = 5$ in the three cases, which is then associated to different volume constraints. Each mesh produced in the course of the process is approximately worth 12000 vertices, and the total computational time is about 90 minutes for $m = 5, 10$, and less than an hour for $m = 0$ (since no adjoint state is involved then). Results are reported in Figure 9.27, and convergence histories lie in Figure 9.28.

9.6.6 Stress criterion minimization

Our last example concerns the design of structures withstanding stress; the objective function at stake is then:

$$J(\Omega) = \int_\Omega k(x)||\sigma(u_\Omega)||^2 \, dx,$$

where $k \in L^\infty(\mathbb{R}^d)$ is a localization factor, $\sigma(u) := Ae(u)$ is the stress tensor associated to a displacement u, and $||.||$ stands for the Frobenius matrix norm. Strictly speaking, this objective function is not of the form (9.3), since it depends on the displacement u_Ω of shapes via their gradient. However, its shape derivative has been computed in [13]: it notably features an adjoint state p_Ω, which differs from that involved in the test cases of section 9.6.5.

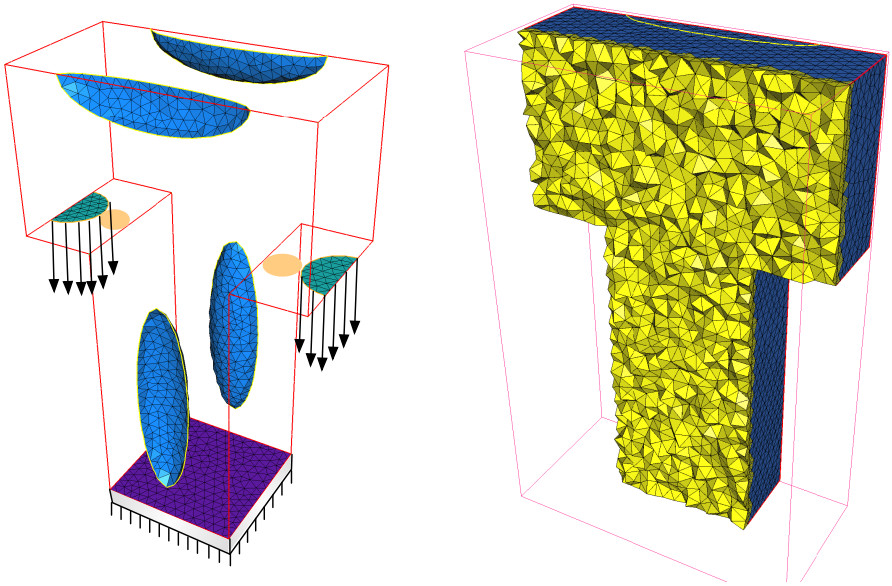


Figure 9.26: *(Left)* initial shape in the perturbed optimal mast test-case, together with boundary conditions for the test-case; *(right)* a cut in the corresponding $3d$ mesh.

As for an example, let us consider the *L-Beam* test case, as depicted on Figure 9.29: shapes enclosed in a L-shaped box of dimensions $2 \times 1 \times 2$ are clamped on their upper side, and submitted to a pointwise unit load, applied at the centre of their front side (once again, no body forces are applied). A volume constraint is enforced by means of a fixed Lagrange multiplier $\ell = 200$, and 100 iterations of our algorithm are performed. Each mesh arising in the optimization sequence has about 17000 vertices, and the whole computation takes about 100 minutes. Results are depicted in Figure 9.29 (see Figure 9.30 for the convergence history).

9.7 Conclusions and perspectives

This chapter has shown some applications of the proposed method for dealing with mesh evolution in shape optimization, which leads to the following general comments:

- The proposed method is able to handle dramatic changes in shapes (even topological changes), while keeping an explicit mesh of them at each iteration of the evolution process. Some intermediate shapes may show very stretched features (especially when a topological change occurs), which ineluctably are meshed with stretched elements. This could cause difficulties in the case of a mechanical equation which suffers greater numerical sensibility than the linearized elasticity system.
- The proposed method in this chapter seems to show greater numerical sensitivity than the 'classical' level set method, mainly owing to the fact that the mesh of the working domain is utterly changed from one iteration to the next; for instance, at some point, it may be difficult to evaluate to what extent any measured improvement in the values of the objective function results from the change in computational meshes.
- It also features a greater accuracy; this is especially true in the case of models where an explicit discretization of the boundaries of shapes is needed (e.g. when it comes to computing the stress developed in shapes; see also the example of section 9.6.4 where we hinted at the fact that a discretization of the interface is mandatory to obtain an acceptable approximation of the transmission conditions between two phases), but even in when the 'simple' cantilever example is considered, Section 9.6.3 demonstrated that the mesh evolution method manages to hone results produced by the fixed mesh level set method.

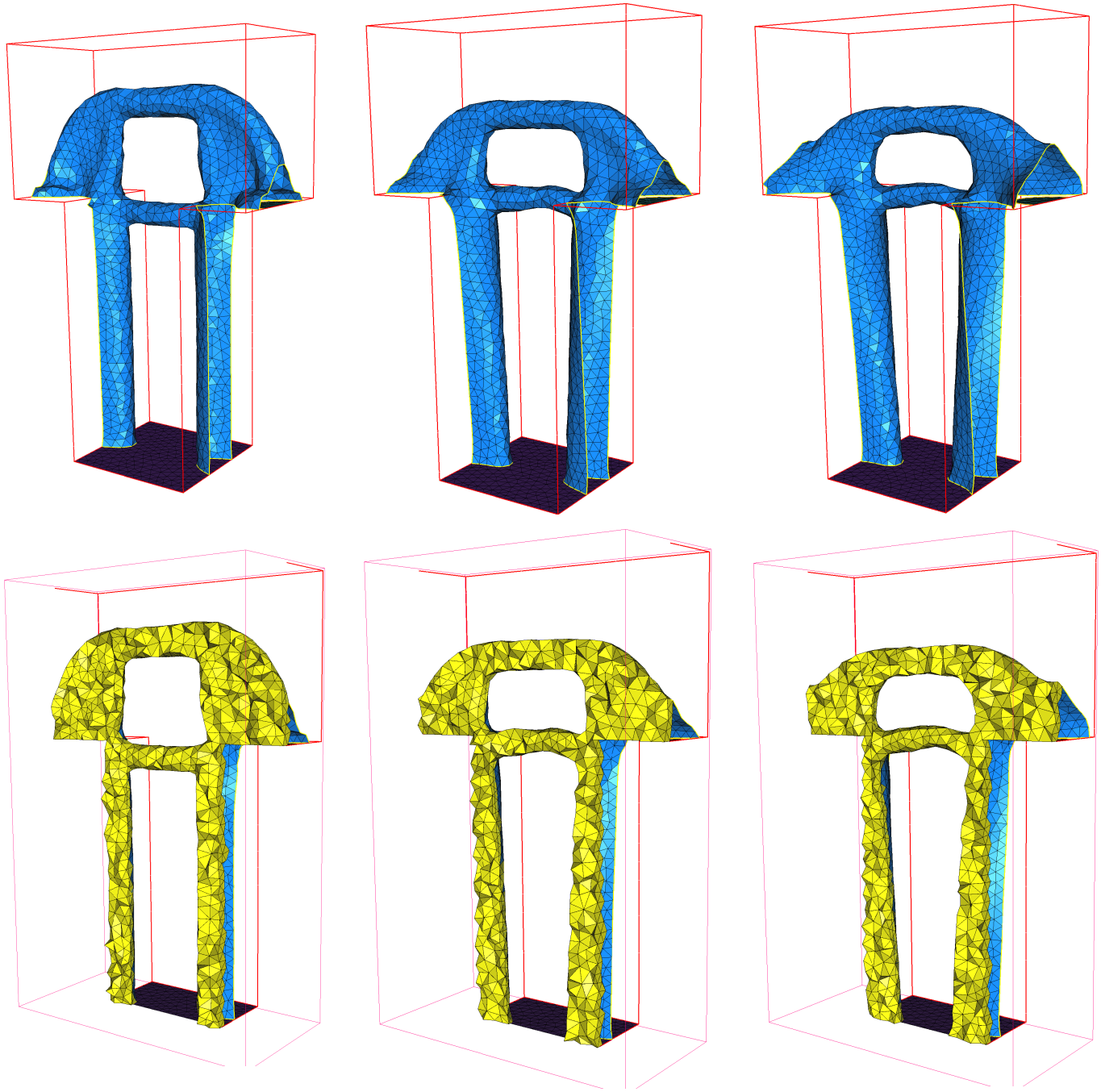


Figure 9.27: *(Upper range)* optimal shapes in the perturbed optimal mast test case for perturbations of amplitude $m = 0, 5, 10$; *(lower range)* cuts in the corresponding $3d$ meshes.

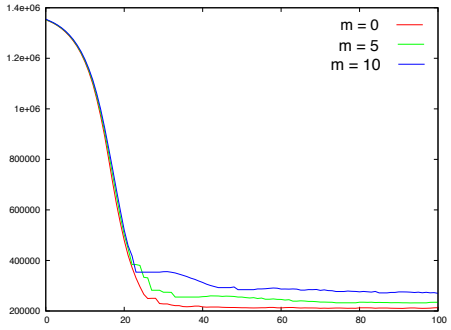


Figure 9.28: *Convergence histories for the robust mast test case.*

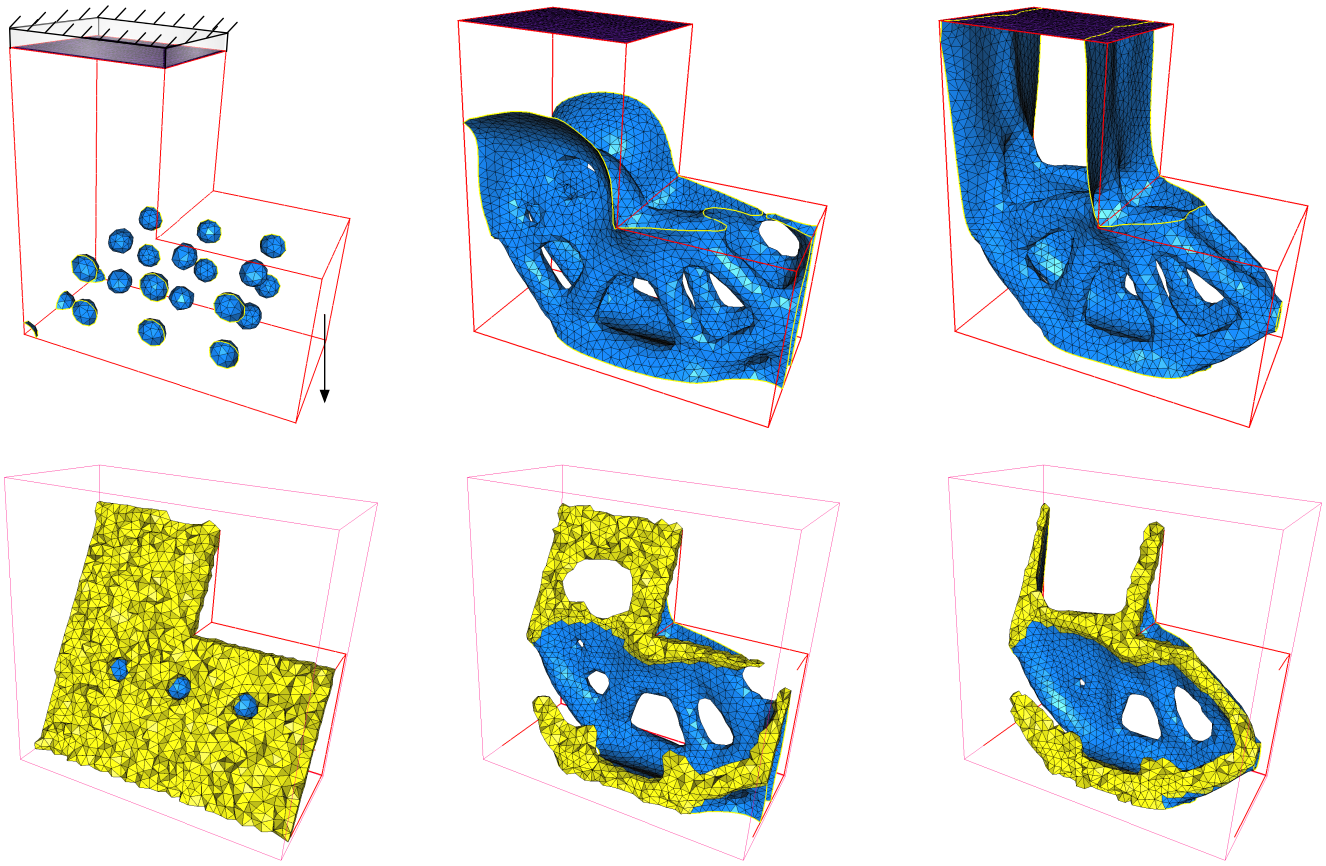


Figure 9.29: *(From top to bottom)* Initial (with boundary conditions), 50^{th} and final iterations of the $3d$ L-Beam test case.

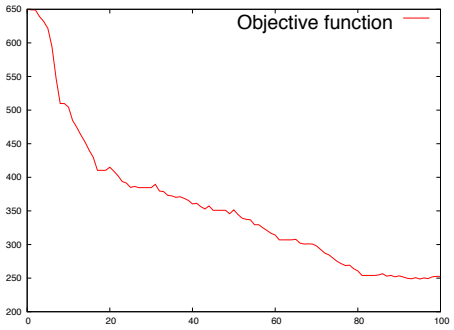


Figure 9.30: *Convergence history for the 3d L-Beam test-case.*

– On the other hand, the presence of an Ersatz material instead of void in the exterior of shapes allows for a flexibility and 'tolerance' which allow to deal with models where shapes tend to exhibit very thin parts, (and in the numerical process, to consistently disconnect, then reconnect) - see the discussion in Section 9.6.5.1.
– The proposed method does not allow to retain the symmetry of shapes, as can be observed on several of the above examples (however, it sometimes happens that the evolution is non symmetric, and the final result miraculously is !). It seems very difficult to enforce any symmetry in shapes with our fully unstructured method unless all the computations are held only on a representative domain, which is replicated by symmetry to get the corresponding shape.

From now on, we believe that at least the following topics would be worth considering as for future work:
– As we have hinted at above, shapes can become quite noisy at some iterations of the process, and a process for 'smoothing out' shapes could prove to be of interest. As far as this issue is concerned, at first sight, it seems easier to carry out denoising directly at the level of the level set function (i.e. before performing the meshing step for the associated negative subdomain).
– It is very tempting to use the proposed method in combination with a mesh adaptation process. Actually, two different types of mesh adaptation could be considered; at first, a 'geometrical' mesh adaptation method could be devised so that, during the step of evolution of the level set function, an increased resolution of the capture of the new shape is possible (see the work of Chapters 6 and 7 in this direction). A different mesh adaptation method, based e.g. on a posteriori error estimates for finite element methods [4], could be aimed at getting a sharper resolution of the linearized elasticity system involved in the computation of a descent direction for the given objective functional.
– It is also very natural to apply this method to other models which could take advantage of an explicit discretization of the boundary of shapes (geometrical constraints - e.g. minimum and maximum thickness constraints - are of this nature). Many other mechanical models would also be worth considering with this method, e.g. in fluid mechanics.
– Eventually, it could be worth considering a 'true', less costly, mesh deformation method in the spirit of [29, 115, 231] for the last iterations of the optimization procedure, where shapes evolve very little (in particular, their topology is fixed).

Bibliography

[1] R. Abgrall, *Numerical Discretization of the First-Order Hamilton-Jacobi Equation on Triangular Meshes*, Com. Pure Applied Math. XLIX, (1996) pp. 1339–1373.

[2] H.T. Ahn and M. Shashkov, *Adaptive Moment-of-Fluid Method*, Submitted to Journal of Scientific Computing,(2010).

[3] M. Aiffa and J. E. Flaherty, *A geometrical approach to mesh smoothing*, Comput. Methods Appl. Mech. Engrg., 192, (2003) pp. 4497–4514.

[4] M. Ainsworth and J. T. Oden, *A posteriori error estimation in finite element analysis*, Comput. Meths. Appl. Mech. Engrg., 142, (1997), pp. 1–88.

[5] F. Alauzet and P. Frey, *Anisotropic Mesh Adaptation for CFD Computations*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 5068–5082.

[6] F. Alauzet and P. Frey, *Estimateur d' erreur géométrique et métriques anisotropes pour l' adaptation de maillage. Partie I: aspects théoriques*, INRIA: Technical Report, 4759 (2003).

[7] F. Alauzet and A. Loseille, *Continuous mesh framework. Part I: well-posed continuous interpolation error*, SIAM J. Numer. Anal., 49, 1 (2011), pp 38–60.

[8] G. Allaire, *Shape optimization by the homogenization method*, Springer Verlag, New York (2002).

[9] G. Allaire, *Conception optimale de structures, Mathematiques et Applications 58, Springer, Heidelberg* (2006).

[10] G. Allaire, E. Bonnetier, G. Francfort and F. Jouve, *Shape optimization by the homogenization method*, Numerische Mathematik, 76, (1997), pp. 27–68.

[11] G. Allaire, C. Castro, *A new approach for the optimal distribution of assemblies in a nuclear reactor,* Numerische Mathematik, 89, pp.1-29 (2001).

[12] G. Allaire and F. Jouve, *A level-set method for vibration and multiple loads structural optimization*, Comput. Meths. Appl. Mech. Engrg., 194, (2005), pp. 3269–3290.

[13] G. Allaire and F. Jouve, *Minimum stress optimal design with the level set method*, Engineering Analysis with Boundary Elements, 32 (2008) pp. 909–918.

[14] G. Allaire and F. Jouve and A.M. Toader, *Structural optimization using shape sensitivity analysis and a level-set method*, J. Comput. Phys., 194 (2004) pp. 363–393.

[15] G. Allaire, F. Jouve and N. Van Goethem, *Damage evolution in brittle materials by shape and topological sensitivity analysis,* J. Comput. Phys., 230 (2011) pp. 5010–5044.

[16] G. Allaire, F. de Gournay, F. Jouve and A.-M. Toader, *Structural optimization using topological and shape sensitivity via a level set method*, Control and Cybernetics 34 (2005) pp. 59–80.

[17] G. Allaire and O. Pantz, *Structural Optimization with FreeFem++*, Struct. Multidiscip. Optim., 32, (2006), pp. 173–181.

[18] P. Alliez, G. Ucelli, C. Gotsman and M. Attene, *Recent advances in remeshing of surfaces*, In Leila de Floriani and Michela Spagnuolo, editors, Shape Analysis and Structuring, Mathematics and Visualization. Springer, Berlin, (2007).

[19] P. Alliez, É. Colin de Verdière, O. Devillers and M. Isenburg, *Isotropic Surface Remeshing*, Shape Modeling International, (2003), pp 49–58.

[20] F. Alouges, A. DeSimone and L. Heltai, *Numerical Strategies for Stroke Optimization of Axisymmetric Microswimmers*, Math. Models Methods Appl. Sci. 21, n. 2 (2011) pp. 361–387.

[21] L. Ambrosio and G. Buttazzo, *An optimal design problem with perimeter penalization*, Calc.Var. 1, (1993) pp. 55–69.

[22] N. Amenta, M. Bern and D. Eppstein, *Optimal Point Placement for Mesh Smoothing*, Journal of Algorithms, 30, 2 (1999), pp 302–322.

[23] N. Amenta, S. Choi, T.K. Dey and N. Leekha, *A Simple Algorithm for Homeomorphic Surface Reconstruction*, International Journal of Computational Geometry and Applications, (2000), pp 213–222.

[24] M. V. Anglada, N.P. Garcia and P. B. Crosa, *Directional Adaptive Surface Triangulation*, Computer Aided Geometric Design, 16 (1999), pp. 107–126.

[25] T. Apel, *Anisotropic Finite Elements: Local Estimates and Applications*, B.G. Teubner Stuttgar, Leipzig, Series of Advances in Numerical Mathematics, (1999).

[26] J.F. Aujol and G. Auber, *Signed distance functions and viscosity solutions of discontinuous Hamilton-Jacobi Equations*, INRIA Technical Report, 4507 (2002).

[27] R. Ausas, E. Dari and G. Buscaglia, *A geometric mass-preserving redistancing scheme for the level set function.* , Int. J. Numer. Methods in Fluids, 65 (2011), pp. 989–1010.

[28] J. Baez and J.P. Muniain, *Gauge Fields, Knots and Gravity*, Series on Knots and Everything, Vol. 4, World Scientific (1994).

[29] T. J. Baker, *Mesh Movement and Metamorphosis*, Eng. Comput. 18, 1, (2002), pp. 188–198.

[30] S. Barb , *Topics in geometric analysis with applications to partial differential equations*, Ph.D., University of Missouri-Columbia (2009).

[31] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, 2nd printing, Modern Birkhäuser Classics, (2008).

[32] C. Bardos, *Problèmes aux limites pour les équations aux dérivées partielles du premier ordre à coefficients réels ; théorèmes d'approximation ; application à l'équation de transport*, Annales scientifiques de l'E.N.S., 4ème série, tome 3, (1970), pp. 185-233.

[33] A. W. Bargteil, T. G. Goktekin, J. F. O' Brien and J. A. Strain, *A semi-Lagrangian contouring method for fluid simulation*, ACM Trans. Graph. 25, 1, (2006), pp. 19–38.

[34] G. Barles, *Remarks on a flame propagation model*, INRIA: Technical Report, 451 (1985).

[35] G. Barles, *Solutions de viscosité des équations de Hamilton-Jacobi*, Springer-Verlag, SMAI (1994).

[36] G. Barles, H. M. Soner and P.E. Souganidis, *Front propagation and phase field theory*, SIAM J. Control Optim., 31, 2, (1992), pp. 439–469.

[37] L. Battaglia and M.A. Storti and J. D'Elia, *Bounded renormalization with continuous penalization for level set interface-capturing methods*, International Journal for Numerical Methods in Engineering, vol 84, *nb.*7 (2010), pp. 830-848.

[38] É. Béchet, J.-C. Cuillere and F. Trochu, *Generation of a finite element MESH from stereolithography (STL) files*, Computer Aided Design, 34, 1, (2002), pp. 1-17.

[39] M.P. Bendsøe and N. Kikuchi, *Generating Optimal Topologies in Structural Design using a Homogenization method*, Comp. Meth. Appl. Mech. Eng., 71, (1988), pp. 197–224.

[40] M.P. Bendsøe and O. Sigmund, *Topology Optimization, Theory, Methods and Applications, 2nd Edition* Springer Verlag, Berlin Heidelberg (2003).

[41] A. BENSOUSSAN, J.-L. LIONS AND G. PAPANICOLAU, *Asymptotic Analysis for Periodic Structures* North-Holland (1978).

[42] A. BEN-TAL AND A. NEMIROVSKI , *Robust Solutions of Linear Programming problems contaminated with uncertain data*, Math. Program. Ser., 88 (2000), pp. 411–424.

[43] M. BERGER AND B. GOSTIAUX, *Differential Geometry: Manifolds, Curves and Surfaces*, Graduate Texts in Mathematics, Springer-Verlag (1987).

[44] A. BERMÚDEZ AND M.R. NOGUEIRAS AND C. VÁSQUEZ, *Numerical analysis of convection-diffusion-reaction problems with higher order characteristics/finite elements. Part I: time discretization* , SIAM J. NUMER. ANAL, Vol. 44, No 5 (2006), pp. 1854-1876.

[45] A. BERMÚDEZ AND M.R. NOGUEIRAS AND C. VÁSQUEZ, *Numerical analysis of convection-diffusion-reaction problems with higher order characteristics/finite elements. Part II: fully discretized scheme and quadrature formulas* , SIAM J. NUMER. ANAL, Vol. 44, No 5 (2006), pp. 1829-1853.

[46] CH. BERNARDI, O. PIRONNEAU, *Sensitivity of Darcy's law to discontinuities*, Chinese Ann. Math. Ser. B **24**, no. 2, pp.205-214 (2003).

[47] C. J. BISHOP, *Nonobtuse triangulations of PSLGs*, preprint available at `http://www.math.sunysb.edu/ bishop/papers/ nonobtuse.pdf` (2011).

[48] G. E. BLELLOCH, G. L. MILLER, AND D. TALMOR, *Developing a Practical Projection-Based Parallel Delaunay Algorithm*, Proceedings of the 12th Annual Symposium on Computation Geometry (1996), pp 61–83.

[49] J. BLOOMENTHAL, *Polygonization of implicit surfaces*, Computer Aided Geometric Design, 5, (1988), pp. 341-355.

[50] H. BOROUCHAKI AND P.J. FREY, *Surface mesh quality evaluation*, Int. j. numer. methods engng., 45 (1999), pp 101–118.

[51] H. BOROUCHAKI AND P.J. FREY, *Surface meshing using a geometric error estimate*, Int. j. numer. methods engng., 58 (2003), pp 227–245.

[52] H. BOROUCHAKI AND P. FREY, *Simplification of surface mesh using Hausdorff enveloppe*, Comput. Methods Appli. Mech. Engrg., 194 (2005), pp. 4864–4884.

[53] H. BOROUCHAKI, P.FREY AND F.HECHT, *Mesh gradation control*, Int. J. Numer. Methods in Engineering, 43 (1998), pp. 1143–1165.

[54] H. BOROUCHAKI, P.-L. GEORGE, F. HECHT AND É. SALTEL, *Delaunay mesh generation governed by metric specifications. Part 1: Algorithms*, Finite Elements in Analysis and Design, 25, (1997), pp 186–195.

[55] H. BOROUCHAKI, P. LAUG AND P.-L. GEORGE, *Parametric surface meshing using a combined advancing-front generalized Delaunay approach*, Int. J. Numer. Methods in Engineering, 49 (2000), pp. 233–259.

[56] M. BOTSCH, *High Quality Surface Generation and Efficient Multires-olution Editing Based on Triangle Meshes*, Aachen: Shaker Verlag, (2005).

[57] H. BREZIS, *Functional Analysis, Sobolev Spaces and Partial Differential Equations*, Springer (2000).

[58] M. BOTSCH, L. KOBBELT, M. PAULY, P. ALLIEZ AND BRUNO LÉVY, *Polygon Mesh Processing*, A K Peters, Ltd. Natick, Massachusetts, (2010).

[59] B. BOURDIN AND A. CHAMBOLLE, *Design-dependent loads in topology optimization*, ESAIM Contr. Optim. Calc. Var., 9, (2003), pp. 19–48.

[60] V. BRAIBANT AND C. FIEURY, *Shape optimal design using B-splines*, Comput. Meths. Appl. Mech. Engrg., 44, (3), (1984), pp. 247–267.

[61] T. BROCHU AND R. BRIDSON, *Robust topological operations for dynamic explicit surfaces*, SIAM J. Sci. Comp, 31, 4, (2009), pp 2472–2493.

[62] D. Bucur and G. Buttazzo, *Variational Methods in Shape Optimization Problems*, Birkäuser (2005).

[63] T.T.C. Bui, P. Frey and B. Maury, *A coupling strategy for solving two-fluid flows*, Int. J. Numer. Methods in Fluids, to appear (2010).

[64] M. Burger, *A framework for the construction of level-set methods for shape optimization and reconstruction*, Interfaces and Free Boundaries, 5 (2003) pp. 301–329.

[65] M. Burger, B. Hackl and W. Ring, *Incorporating topological derivatives into level set methods*, J. Comp. Phys., 194, 1 (2004) pp. 344–362.

[66] H.-J. Butt, K. Graf, M. Kappl, *Physics and Chemistry of Interfaces*, Wiley (2003).

[67] E. Cances, R. Keriven, F. Lodier and A. Savin, *How electrons guard the space: shape optimization with probability distribution criteria*, Theor. Chem. Acc., 111, (2004) pp. 373–380.

[68] A. Canelas, A. A. Novotny and J.-R. Roche, *A new method for inverse electromagnetic casting problems based on the topological derivative*, J. Comput. Phys., 230 (2011) pp. 3570–3588.

[69] P. Cannarsa and P. Cardaliaguet, *Representation of equilibrium solutions to the table problem for growing sandpiles*, J. Eur. Math. Soc. 6 (2004), pp. 1–30.

[70] V. Caselles, F. Catté, B. Coll and F. Dibos, *A geometric model for edge detection*, Num. Mathematik, 66, (1993), pp. 1–31.

[71] V. Caselles, R. Kimmel and G. Sapiro, *Geodesic Active Contours*, International Journal of Computer Vision, 22, 1 (1997) pp. 61–79.

[72] J. Céa , *Conception optimale ou identification de formes, calcul rapide de la dérivée directionnelle de la fonction coût*, Math. Model. Num. 20, 3 (1986), pp. 371–420.

[73] A. Cervone and S. Manservisi and R. Scardovelli and S. Zaleski, *A geometrical predictor - corrector advection scheme and its application to the volume fraction function*, Journal of Scientific Computing, vol 228,(2009), pp. 406-419.

[74] A. Chambolle , *A density result in two-dimensional linearized elasticity and applications*, Arch. Ration. Mech. Anal., 167, (2003), pp. 211–233.

[75] I. Chavel, *Riemannian Geometry, a modern introduction*, 2nd Edition, Cambridge University Press (2006).

[76] B. Chazelle, *Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm*, SIAM J. Comput, 13, 3 (1984), pp 488–507.

[77] S. Chen and W. Chen , *A new level-set based approach to shape and topology optimization under geometric uncertainty*, Struct Multidisc Optim, 44, 1 (2011), pp. 1–18.

[78] S. Chen, B. Merriman, S. J. Osher, and P. Smereka, *A Simple Level Set Method for Solving Stefan Problems*, J. Comput. Phys., (1997), 135, pp. 8–29.

[79] D. Chenais , *On the existence of a solution in a domain identification problem*, J. Math. Anal. Appl., 52, (1975), pp. 189–289.

[80] S.-W. Cheng, T.K. Dey and J. R. Shewchuk, *Delaunay Mesh Generation*, CRC Press, Boca Raton, Florida, (2012).

[81] L.-T. Cheng and Y.-T. Tsai, *Redistancing by flow of time dependent eikonal equation*, Journal of Computational Physics, 227 (2008), pp. 4002–4017.

[82] A. Cherkaev, *Variational Methods for Structural Optimization*, Springer, New York (2000).

[83] A. Cherkaev and E. Cherkaeva, *Optimal design for uncertain loading conditions*, in V. Berdichevsky et al., editor, Homogenization, volume 50 of Series on Advances in Mathematics for Applied Sciences (1999), pp.193–213.

[84] A. Cherkaev and E. Cherkaeva, *Principal Compliance and Robust Optimal Design*, Journal of Elasticity, 72 (2003), pp.71–98.

[85] L. P. CHEW, *Constrained Delaunay Triangulations*, Algorithmica, 4, 1, (1989), pp. 97–108.

[86] L. P. CHEW, *Guaranteed-Quality Mesh Generation for Curved Surfaces*, Proceedings of the Ninth Annual Symposium on Computation Geometry, (1993), pp. 274–280.

[87] L. P. CHEW, *Guaranteed-quality triangular meshes*, Tech. Report TR 89-983, Cornell University, (1989).

[88] S.-K. CHOI, R. GRANDHI, AND R. A. CANFIELD , *Reliability-based Structural Design*, Springer, (2007).

[89] D. CHOPP, *Computing minimal surfaces via level-set curvature flow*, J. Comput. Phys., 106, pp. 77-91 (1993).

[90] D. L. CHOPP AND J. A. SETHIAN, *Flow under curvature: singularity formation, minimal surfaces, and geodesics*, Experiment. Math. 2, 4 (1993), pp. 235–255.

[91] P.G. CIARLET, *The Finite Element Method for Elliptic Problems*, North Holland Publishing Company, (1978).

[92] P.G. CIARLET, *Mathematical Elasticity, vol I: Three Dimensional Elasticity*, North Holland Publishing Company (1988).

[93] A. CLAISSE AND P. FREY, *Level Set Driven Smooth Curve Approximation From Unorganized or Noisy Point Set*, ESAIM: Proceedings,(2008).

[94] F. H. CLARKE AND R.J. STERN AND P.R. WOLENSKI, *Proximal smoothness and the lower C^2 property*, Journal of Convex Analysis, Vol 2 (1995) no. 1/2, pp. 117–144.

[95] S. CONTI, H. HELD, M. PACH, M. RUMPF, AND R. SCHULTZ , *Shape optimization under uncertainty - a stochastic programming approach*, SIAM J. Optim, 19, 4 (2009), pp. 1610–1632.

[96] R. COURANT, E. ISAACSON AND M. REES, *On the solution of nonlinear hyperbolic differential equations by finite differences*, Comm. Pure Appl. Math. 5, (1952), pp. 243–255.

[97] S.J. COX, *The Shape of the Ideal Column*, The Mathematical Intelligencer vol. 14, no. 1, (1992) pp. 16–24.

[98] M. G. CRANDALL AND P.-L. LIONS, *Two Approximations of Solutions of Hamilton-Jacobi Equations*, Math. Comput., 167, Vol. 43, (1984), pp. 1–19.

[99] M. G. CRANDALL, H. ISHII AND P.-L. LIONS, *UserÕs guide to viscosity solutions of second order partial differential equations*, Bull. Amer. Math. Soc. (N.S.), 27 (1992), pp. 1–67

[100] M. G. CRANDALL AND P.-L. LIONS, *Viscosity solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc. 277 (1983), pp. 1–42.

[101] E. CRISTIANI AND M. FALCONE, *Fast semi-Lagrangian schemes for the Eikonal equation and applications*, SIAM J. Numer. Anal. Vol. 45, No. 5, (2007) pp. 1979–2011.

[102] B. CURLESS AND M. LEVOY, *A volumetric method for building complex models from range images*, ACM Transactions on Graphics, Special issue on Proceedings of SIGGRAPH 1996 (1996), pp. 303–312.

[103] E.F. D'AZEVEDO AND R.B. SIMPSON, *On Optimal Triangular Meshes for Minimizing the Gradient Error*, Numerische Mathematik, 59 (1991), pp. 321–348.

[104] J. D. DEATON AND R. V. GRANDHI, *A survey of structural and multidisciplinary continuum topology optimization: post 2000*, Struct. Multidisc. Optim., (2013) DOI 10.1007/s00158-013-0956-z.

[105] M.C. DELFOUR AND J.-P. ZOLESIO, *Shapes and Geometries: Metrics, Analysis, Differential Calculus, and Optimization*, SIAM, Philadelphia 2nd ed. (2011).

[106] M.C. DELFOUR AND J.-P. ZOLESIO, *Velocity method and Lagrangian formulation for the computation of the shape Hessian*, SIAM J. Control Optim. 29, No 6 (1991) pp. 1414–1442.

[107] M. C. DELFOUR AND J.-P. ZOLESIO, *Oriented distance function and its evolution equation for initial sets with thin boundary*, SIAM J. Control Optim. 42, No 6 (2004) pp. 2286–2304.

[108] M. C. DELFOUR AND J.-P. ZOLESIO, *Shape identification via metrics constructed from the oriented distance function*, Control and Cybernetics 34, No 1 (2005) pp. 137-164.

[109] M.C. DELFOUR AND J.-P. ZOLESIO, *Anatomy of the Shape Hessian*, Annali di Matematica Pura ed Applicata, 159, 1, (1991), pp. 315–339.

[110] G. DELGADO, *Ph.D. thesis*, Ecole Polytechnique. In preparation.

[111] J.P. DEMAILLY, *Analyse numérique et équations différentielles.*, EDP Sciences (2006).

[112] N. P. VAN DIJK, K. MAUTE, M. LANGELAAR AND F. VAN KEULEN, *Level-set methods for structural topology optimization: a review*, Struct. Multidisc. Optim., (2013) DOI 10.1007/s00158-013-0912-y.

[113] M. DO CARMO, *Riemannian Geometry*, Mathematics : Theory & Applications, 2*nd* Edition, Birkhäuser, (1993).

[114] C. DOBRZYNSKI, *Adaptation de maillage anisotrope* 3*d et application à l'aéro-thermique des bâtiments*, Thèse de l'Université Paris VI, (2005).

[115] C. DOBRZYNSKI AND P. FREY, *Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations*, Proc. 17th Int. Meshing Roundtable, Pittsburgh, (2008).

[116] M. DO CARMO, *Riemannian Geometry*, Mathematics: Theory & Applications, 2*nd* Edition, Birkhäuser, (1993).

[117] A. DOI AND A. KOIDE, *An Efficient Method of Triangulating Equivalued Surfaces by Using Tetrahedral Cells*, IEICE Transactions on Communications and Electronics Information Systems E74, 1, (1991), pp 214–224.

[118] J. DONEA, A. HUERTA, J.-P. PONTHOT AND A. RODRÍGUEZ-FERRAN, *Arbitrary Lagrangian-Eulerian Methods*, chap. 14 in Encyclopedia of Computational Mechanics, (2004).

[119] V. DUCROT AND P. FREY, *Contrôle de l'approximation géometrique d une interface par une métrique anisotrope*, Comptes Rendus de l'Académie des Sciences, 345 (2007), pp. 537–542.

[120] P. DUYSINX, L. VAN MIEGROET, T. JACOBS AND C. FLEURY, *Generalized shape optimization using X-FEM and level set methods*, IUTAM Symposium on Topological Design Optimization of Structures, Machines and Materials Solid Mechanics and Its Applications, 137, (2006), pp. 23–32.

[121] M. ECK, T. DEROSE, T. DUCHAMP, H. HOPPE, M. LOUNSBERY AND W. STUETZLE, *Mesh optimization*, ACM Transactions on Graphics, (SIGGRAPH Õ95 Proceedings), (1993), pp. 173–182.

[122] M. ECK, T. DEROSE, T. DUCHAMP, H. HOPPE, M. LOUNSBERY AND W. STUETZLE, *Multiresolution analysis of arbitrary meshes*,SIGGRAPH '95 Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, (1995), pp 173–182.

[123] J. EHLERS, F.A.E. PIRANI, AND A. SCHILD, *The geometry of free fall and light propagation*, General Relativity, Papers in Honor of J. L. Synge. Oxford University Press, (1972).

[124] D. ENRIGHT AND R. FEDKIW AND J. FERZIGER AND I. MITCHELL, *A Hybrid Particle Level Set Method for Improved Interface Capturing*, Journal of Computational Physics, 183 (2002), pp. 83–116.

[125] H. ESCHENAUER, V. KOBELEV, A. SCHUMACHER, *Bubble method for topology and shape optimization of structures,* Structural Optimization, 8, (1994), pp. 42–51.

[126] L.C. EVANS, *Partial Differential Equations*, American mathematical society,(1997).

[127] L. C. EVANS AND R. F. GARIEPY, *Measure theory and fine properties of functions*, CRC Press (1992).

[128] L.C. EVANS AND J. SPRUCK, *Motion of level sets by mean curvature. I*, J. Differential Geom. Volume 33, Number 3 (1991), pp 635–681.

[129] R.E. EWING AND H. WANG, *A Summary of Numerical Methods for Time-Dependent Advection-Dominated Partial Differential Equations*, Num. Anal., *V I I* (2000), pp. 423-445.

[130] M. FALCONE AND R. FERRETTI, *Semi-Lagrangian Schemes for HamiltonÐJacobi Equations, Discrete Representation Formulae and Godunov Methods*, J. Comput. Phys., 175, (2002), pp. 559-575.

[131] G. FARIN, *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*, Academic Press Inc, 4th Edition, (1997).

[132] O. FAUGERAS AND R. KERIVEN, *Variational principles, surface evolution, pdes, level set methods and the stereo problem*, IEEE Transactions on Image Processing, 7, 3 (1998), pp. 336–344.

[133] H. FEDERER, *Curvature Measures*, Transactions of the American Mathematical Society Vol. 93, No. 3 (1959), pp. 418-491.

[134] D.A. FIELD, *Laplacian smoothing and Delaunay triangulations*, Communications in Applied Numerical Methods, 4, (1988), pp 709–712.

[135] D. FILIP, R. MAGEDSON AND R. MARKOT, *Surface algorithms using bounds on derivatives*, Computer Aided Geometric Design, 3 (1986), pp. 295–311.

[136] M.S. FLOATER AND K. HORMANN, *Surface parameterization: a tutorial and survey*, In Advances in Multiresolution for Geometric Modelling, (2005), pp 157–186.

[137] D. M. FRANGOPOL AND K. MAUTE, *Life-cycle reliability-based optimization of civil and aerospace structures*, Computers & Structures, 81, (2003), pp.397–410.

[138] C. O. FREDERICK, Y.C. WONG AND F.W. EDGE, *Two-Dimensional Automatic Mesh Generation for Structural Analysis*, Int. J. Numer. Methods in Engineering, 2 (1970), pp. 133–144.

[139] L. FREITAG AND C. OLLIVER-GOOCH, *Tetrahedral Mesh Improvement Using Face Swapping and Smoothing*, Int. J. Numer. Methods in Engineering, 40 (1997), pp. 3979–4002.

[140] P. FREY, *About Surface Remeshing*, Proc. of the 9th IMR, (2000), pp. 123–136.

[141] P. FREY AND H. BOROUCHAKI, *Texel : triangulation de surfaces implicites. Partie I : aspects théoriques.*, INRIA : Technical Report, 3066 (1996).

[142] P. FREY AND H. BOROUCHAKI, *Geometric surface mesh optimization*, Comput Visual Sci, 1 (1998), pp. 113–121.

[143] P. FREY AND H. BOROUCHAKI, *Surface meshing using a geometric error estimate*, Int. J. Numer. Methods in Engineering, 58 (2003), pp. 227–245.

[144] P. FREY, H. BOROUCHAKI AN P.-L. GEORGE, *Delaunay tetrahedralization using an advancing front approach*, Proc. of the 5th IMR, (1996), pp. 31–46.

[145] P.J. FREY AND P.L. GEORGE, *Mesh Generation: Application to Finite Elements*, Wiley, 2nd Edition, (2008).

[146] A. FUHRMANN, G. SOBOTTKA AND C. GROSS, *Abstract Distance Fields for Rapid Collision Detection in Physically Based Modeling*, International Conference Graphicon: Proceedings,(2003).

[147] S. GARREAU AND PH. GUILLAUME AND M. MASMOUDI, *The topological asymptotic for PDE systems : the elasticity case*, SIAM J. Control. Optim., 39 (2001) pp. 1756–1778.

[148] J. A. GEORGE, *Computer Implementation of the Finite Element Method*, Ph.D Thesis of Stanford University, (1971).

[149] P.-L. GEORGE, *Improvements on Delaunay-based three-dimensional automatic mesh generator*, Finite Elements in Analysis and Design 25 (1997) pp. 297–317.

[150] P.-L. GEORGE, H. BOROUCHAKI, *Back to edge flips in* 3 *dimensions*, Proc. 12th Int. Meshing Roundtable, Santa Fe, New Mexico, U.S.A., (2003).

[151] P.-L. GEORGE AND H. BOROUCHAKI, *Delaunay triangulation and meshing : Application to Finite Elements*, Hermes, (1998).

[152] P.-L. GEORGE AND H. BOROUCHAKI, *Premières expériences de maillage automatique par une méthode de Delaunay anisotrope en trois dimensions*, INRIA : Technical Report, 0272 (2002).

[153] P.-L. GEORGE, H. BOROUCHAKI AND É. SALTEL, *'UltimateÕ robustness in meshing an arbitrary polyhedron*, Int. J. Numer. Methods in Engineering, 58 (2003), pp. 1061–1083.

[154] P.-L. GEORGE, F. HECHT AND É. SALTEL, *Automatic mesh generator with specified boundary*, Comp. Meth. Appl. Mech. Eng., 92, (1991), pp. 269–288.

[155] P.-L. George and E. Seveno, *The advancing-front mesh generation method revisited*, Int. J. Numer. Methods in Engineering, 37-21 (1994), pp. 3605–3619.

[156] S. Ghosh Moulic and A. Salih, *Some numerical studies of interface advection properties of level set method*, Sādhanā, 34 (2009), pp. 271–298.

[157] Y. Giga, *Surface Evolution Equations, a Level Set Approach*, Monographs in Mathematics, 99. Birkhäuser, Basel-Boston-Berlin, (2006).

[158] J. Glimm, J. W. Grove, X. L. Li, K.-M. Shyue, Y. Zeng, Q. Zhang, *Three Dimensional Front Tracking*, SIAM J. Sci. Comp, 19, (1995), pp 703–727.

[159] F. Golse, *Distributions, analyse de Fourier, équations aux dérivées partielles*, Cours de l'Ecole Polytechnique (2010).

[160] J. Gomes and O. Faugeras, *Reconciling distance functions and level sets*, Scale-Space Theories in Computer Vision, pp. 70-81, Springer (1999).

[161] A.J.P Gomes, I. Voiculescu, J. Jorge, B. Wyvill and C. Gallsbraith, *Implicit Curves and Surfaces: Mathematics, Data Structures and Algorithms*, Springer,(2009).

[162] F. de Gournay, *Velocity extension for the level-set method and multiple eigenvalues in shape optimization.* SIAM J. on Control and Optim., 45, no. 1, 343–367 (2006).

[163] F. de Gournay, G. Allaire and F. Jouve , *Shape and topology optimization of the robust compliance via the level set method*, ESAIM: Control, Optimization and Calculus of Variations, 14, (2008), pp. 43–70.

[164] P. Guigue and O. Devillers, *Fast and Robust Triangle-Triangle Overlap Test using Orientation Predicates*, Journal of Graphics, Gpu, and Game Tools, 8 (2003), pp. 25–42.

[165] X. Guo, W. Zhang and L. Zhang , *Robust structural topology optimization considering boundary uncertainties*, Comput. Methods Appl. Mech. Engrg., 253, (2013), pp. 356–368.

[166] X. Guo, W. Bai, W. Zhang and X. Gao , *Confidence structural robust design and optimization under stiffness and load uncertainties*, Comput. Methods Appl. Mech. Engrg., 198, (2009), pp. 3378–3399.

[167] S.-H. Ha, S. Cho, *Level set based topological shape optimization of geometrically nonlinear structures using unstructured mesh*, Computers and Structures, 86, (2008), pp. 844Ð-868

[168] J. Hadamard, *Mémoire sur le problème d'analyse relatif à l'équilibre des plaque élastiques encastrées*, Bull. Soc. Math. France, (1907).

[169] R. T. Haftka and R. V. Grandhi, *Structural shape optimization - a survey*, Comput. Methods Appl. Mech. Engrg., 57, (1986), pp. 91–106.

[170] E. Harten and S. J. Osher, *Uniformly High-Order Accurate Nonoscillatory Schemes. I*, SIAM J. Numer. Anal., 24, 2, (1987), pp. 279–309.

[171] J. Haslinger, J. Dvorak, *Optimum composite material design,* RAIRO M2AN, 29, pp.657-686 (1995).

[172] A. Henrot and M. Pierre, *Variation et optimisation de formes, une analyse géométrique*, Springer (2005).

[173] A. Henrot and J. Sokolowski, *Mathematical challenges in shape optimization*, Control and Cybernetics 34,1 (2005), pp. 37–57

[174] R. Herzog and F. Schmidt , *Weak lower semi-continuity of the optimal value function and applications to worst-case robust optimal control problems*, Optimization, 61, 6, (2012), pp. 685–697.

[175] F. Hettlich, W. Rundell, *The determination of a discontinuity in a conductivity from a single boundary measurement*, Inverse Problems, **14**, no. 1, pp.67-82 (1998).

[176] K. Hildebrandt, K. Polthier and M. Wardet, *On the Convergence of Metric and Geometric Properties of Polyhedral Surfaces*, Geometriae Dedicata, 123, (2006), pp 89–112.

[177] M. HINTERMÜLLER AND W. RING, *A Second Order Shape Optimization Approach for Image Segmentation*, SIAM J. Appl. Math. 64, No 2 (2004) pp. 442–467.

[178] I. HLAVÁČEK, A.A. NOVOTNY, J. SOKOLOWSKI AND A. ŻOCHOWSKI , *On topological derivatives for elastic solids with uncertain input data*, J. Optim. Theory Appl., 141 (2009), pp. 569–595.

[179] H. HOPPE, T. DEROSE, T. DUCHAMP, J. MCDONALD AND W. STUETZLE, *Mesh optimization*, ACM Transactions on Graphics, (SIGGRAPH Õ93 Proceedings), (1993), pp. 19–26.

[180] W. HUANG, *Metric Tensors for Anisotropic Mesh Generation*, Journal of Computational Physics, 204 (2005), pp. 633–665.

[181] W. HUANG AND R. D. RUSSEL, *Adaptive Moving Mesh Methods*, Applied Mathematical Sciences, Vol. 174, (2011).

[182] H. ISHII, *Existence and uniqueness of solutions of Hamilton-Jacobi equations*, Funkcialaj Ekvacioj, 29 (1986), pp. 167–188.

[183] X. JIAO, *Face offsetting: A unified approach for explicit moving interfaces*, J. Comput. Phys., 220 (2007) pp. 612–625.

[184] B. JOE, *Construction of three-dimensional Delaunay triangulations using local transformations*, Computer Aided Geometric Design, 8, (1991), pp. 123-142.

[185] M.W. JONES, *3D distance from a Point to a Triangle*, Department of Computer Sciences, University of Wales Swansea: technical report, (1995).

[186] M. KALSI, K. HACKER AND K. LEWIS, *A Comprehensive Robust Design Approach for Decision Trade-Offs in Complex Systems Design*, in Complex Systems Design,Ó ASME Journal of Mechanical Design, (2001), pp. 1–10.

[187] H. KARCHER, *Riemannian Center of Mass and Mollifier Smoothing*, Communications on Pure and Applied Mathematics, vol. 30, no. 5, (1977), pp 509–541.

[188] A.L. KARCHEVSKY, *Reconstruction of pressure velocities and boundaries of thin layers in thinly-stratified layers*, J. Inverse Ill-Posed Probl. 18 (2010), no. 4, 371-388.

[189] M. KASS, A. WITKIN AND D. TERZOPOULOS, *Snakes: Active contour models*, International Journal of Computer Vision, 1 (1988) pp. 321–331.

[190] G. KHARMANDA, N. OLHOFF, A. MOHAMED AND M. LEMAIRE, *Reliability-based topology optimization*, Struct. Multidisc. Optim., 26, (2004), pp. 295–307.

[191] A. KHEYFETS, W. A. MILLER AND G. A. NEWTON, *Riemannian Schild's Ladder Parallel Transport Procedure for an Arbitrary Connection*, International Journal of Theoretical Physics, Vol. 39, No. 12, (2000), pp 2891–2898.

[192] H. KIM AND M.S. LIOU, *Accurate adaptive level set method and sharpening technique for three dimensional deforming interfaces*, Computer and Fluids, vol 44,(2011), pp. 111-129.

[193] R. KIMMEL AND J.A. SETHIAN, *Fast Voronoi Diagrams and Offsets on Triangulated Surfaces*, Proc. of AFA Conf. on Curves and Surfaces, (1999), pp. 193–202.

[194] R. KIMMEL AND J.A. SETHIAN, *Computing geodesic paths on manifolds*, Proc. Natl. Acad. Sci. USA, vol.95 (1998), pp. 8431–8435.

[195] B. M. KLINGNER AND J. R. SHEWCHUK, *Aggressive Tetrahedral Mesh Improvement*, Proc. 16th Int. Meshing Roundtable, (2008), pp. 3–23.

[196] R. V. KOHN AND G. STRANG, *Optimal design and relaxation of variational problems*, Comm. Pure Appl. Math., 39 (1986) pp. 1-25 (Part I), 139–182 (Part II), and 353–377 (Part lII).

[197] F. LABELLE AND J. R. SHEWCHUK, *Isosurface Stuffing: Fast Tetrahedral Meshes with Good Dihedral Angles*, ACM Transactions on Graphics, Special issue on Proceedings of SIGGRAPH 2007, 26 (3), (2007), pp. 28–37.

[198] S. LANG, *Fundamentals of differential geometry*, Graduate Texts in Mathematics, 191, Springer-Verlag, New York, (1999).

[199] R.J. LEVEQUE, *High-Resolution Conservative Algorithms for Advection in Incompressible Flow.*, SIAM J. Numer. Anal., 33 (1996), pp. 627–665.

[200] B. LEVY, S. PETITJEAN, N. RAY AND J. MAILLOT, *Least squares conformal maps for automatic texture atlas generation*, ACM Transactions on Graphics, (SIGGRAPH Õ02 Proceedings), (2002), pp. 362–371.

[201] X. LI, J.-F. REMACLE, N. CHEVAUGEON AND M.S. SHEPARD, *Anisotropic Mesh Gradation Control*, Proceedings, 13th International Meshing Roundtable, Sandia National Laboratories, (2004), pp 401–412.

[202] L. LI, M. Y. WANG AND P. WEI *XFEM schemes for level set based structural optimization*, Front. Mech. Eng. 7, 4, (2012), pp. 335–356.

[203] R. LIPTON, *Design of functionally graded composite structures in the presence of stress constraints*, Internat. J. Solids Structures 39 (2002), no. 9, 2575-2586.

[204] A. LIU AND B. JOE, *Relationship between tetrahedron shape measures*, BIT, 34 (3), (1994), pp. 268–287.

[205] A. LIU AND B. JOE, *On the shape of tetrahedra from bisection*, Math. Comp. 63 (1994), pp. 141–154

[206] R. LÖHNER, *Regridding surface triangulations*, J. Comput. Phys., 126, (1996), pp. 1-10.

[207] R. LÖHNER AND A. PARIKH, *Generation of three-dimensional unstructured grids by the advancing-front method*, Int. J. Numer. Meth. Fluids, 8 (1988), pp. 1135–1149.

[208] W. E. LORENSEN AND H. E. CLINE, *Marching cubes: a high resolution 3d surface construction algorithm*, COMPUTER GRAPHICS, 21, 4 (1987), pp. 163–169.

[209] M. LORENZI, N. AYACHE AND X. PENNEC, *Schild's ladder for the parallel transport of deformations in time series of images*, in Proceedings of Information Processing in Medical Imaging (IPMIÕ11), G. Szekely and H. Hahn, eds., vol. 6801 of LNCS, (2011), pp. 463–474.

[210] A. LOSEILLE AND F. ALAUZET, *Continuous mesh framework. Part I: well-posed continuous interpolation error*, SIAM J. Numer. Anal.,Vol. 49, Issue 1,(2011), pp 38–60.

[211] A. LOSEILLE AND F. ALAUZET, *Continuous mesh framework. Part II: validations and applications*, SIAM J. Numer. Anal.,Vol. 49, Issue 1,(2011), pp 61–86.

[212] K.A. LURIE, A.V. FEDOROV AND A.V. CHERKAEV, *Regularization of optimal design problems for bars and plates*, J. Optim. Theory Appl., 37 (1982) pp. 499-521 (Part I), and 523-543 (Part II).

[213] M.MALLADI, J.A. SETHIAN AND B.C. VEMURI, *A Fast Level Set based Algorithm for Topology-Independent Shape Modeling*, J. Math. Imaging and Vision, 6,2 (1996), pp. 269–290.

[214] C. MANTEGAZZA AND A.C. MENNUCCI, *Hamilton-Jacobi Equations and Distance Functions on Riemannian Manifolds*, Appl. Math. Opt., 47 (2003), pp. 1–25.

[215] E. MARCHANDISE, C. CARTON DE WIART, W. G. VOS, C. GEUZAINE AND J-F REMACLE, *High Quality Surface Remeshing Using Harmonic Maps. Part II: Surfaces with High Genus and of Large Aspect Ratio.*, Int. J. Numer. Methods in Engineering, 86 (2011), pp. 1303–1321.

[216] E. MARCHANDISE AND J.-F. REMACLE, *A stabilized finite element method using a discontinuous level set approach for solving two phase incompressible flows*, J. Comput. Phys., 219 (2006), pp. 780–800.

[217] E. MARCHANDISE AND J.-F. REMACLE AND N. CHEVAUGEON, *A Quadrature free discontinuous Galerkin method for the level set equation.*, Journal of Computational Physics, 212 (2005), pp. 338–357.

[218] M. MASMOUDI, J. POMMIER, AND B. SAMET, *The topological asymptotic expansion for the Maxwell equations and some applications*, Inverse Problems, 21, (2), (2005), pp. 547–564.

[219] S. V. MATVEYEV, *Approximation of Isosurface in the Marching Cube: Ambiguity Problem*, In: Proceedings of visualization Õ94, Washington, (1994), pp. 288-292.

[220] D. J. MAVRIPILIS, *An advancing front Delaunay triangulation algorithm designed for robustness*, ICASE report, 92-49 (1992).

[221] T. MCINERNEY AND D.TERZOPOULOS, *Topology Adaptive Deformable Surfaces for Medical Image Volume Segmentation*, IEEE transactions on medical imaging, 18, 10 (1999), pp. 840–850.

[222] W. MC LEAN, *Strongly Elliptic Systems and Boundary Integral Equations*, Cambridge University Press, Cambridge (2000).

[223] Y. MEI AND X. WANG, *A level set method for structural topology optimization with multi-constraints and multi-materials,* Acta Mechanica Sinica, Vol.20, No.5, (2004).

[224] Y. MEI AND X. WANG, *A level set method for structural topology optimization and its applications,* Advances in Engineering software, Vol.35, 415-441 (2004).

[225] E. A. MELISSERATOS, *L^p optimal d dimensional triangulation for piecewise linear interpolation: a new result on data dependent triangulations*, Technical Report RUU-CS-93-13, Department of Computer Science, Utrecht University, (1993).

[226] F. MEMOLI AND G. SAPIRO, *Fast Computation of Weighted Distance Functions and Geodesics on Implicit Hyper-Surfaces*, Journal of Computational Physics, 173 (2001), pp. 730–764.

[227] M. MEYER, M. DESBRUN, P. SCHRÖDER AND A. BARR, *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*, Visualization and Mathematics III (2003), pp. 35–57.

[228] G. MICHAILIDIS, *Thèse de l'École Polytechnique* (in preparation)

[229] G. MILTON, *The theory of composites,* Cambridge University Press (2001).

[230] C. MIN AND F. GIBOU, *A Second Order Accurate Level Set Method on Non-graded Adaptive Cartesian Grids.*, Journal of Computational Physics, 225 (2007), pp. 300–321.

[231] M. K. MISZTAL AND J. A. BAERENTZEN, *Topology Adaptive Interface Tracking Using the Deformable Simplicial Complex*, ACM Trans. Graph. 31, 3, (2012), pp. 1–24.

[232] B. MOHAMMADI AND O. PIRONNEAU, *Applied Shape Optimization for Fluids* Numerical Mathematics and Scientific Computation, Oxford University Press, 2nd edition (2010).

[233] W. MULDER, S. OSHER AND J.A. SETHIAN, *Computing interface Motion in Compressible Gas Dynamics*, J. Comput. Phys., 100, (1992), pp. 209-228.

[234] F. MURAT AND J. SIMON, *Sur le contrôle par un domaine géométrique*, Technical Report RR-76015, Laboratoire d'Analyse Numérique (1976).

[235] M. MURPHY, D. MOUNT, AND C. W. GABLE, *A Point-Placement Strategy for Conforming Delaunay Tetrahedralization*, Proceedings of the 11th Annual Symposium on Discrete Algorithms, (2000) pp. 67–74.

[236] F. MUT, G.C. BUSCAGLIA AND E. A. DARI, *A New Mass-Conserving Algorithm for Level-Set Redistancing on Unstructured Meshes* , Mecanica Computational, 23 (2004), pp. 1659–1678.

[237] J. NOCEDAL AND S.J. WRIGHT , *Numerical Optimization* Springer-Verlag (2000).

[238] C. NOUR, R.J. STERN AND J. TAKCHE, *The θ-exterior sphere condition, φ-convexity, and local semiconcavity*, Nonlinear Analysis, 73 (2010), pp 573–589.

[239] ZH.O. ORALBEKOVA, K.T. ISKAKOV, A.L. KARCHEVSKY, *Existence of the residual functional derivative with respect to a coordinate of gap point of medium*, to appear in Appl. Comput. Math.

[240] J.M. ORTEGA AND W.C. RHEINBOLDT, *On discretization and differentiation of operators with application to Newton's method*, SIAM journal on numerical analysis, 3,1, 143-156 (1966).

[241] S. OSHER, L.-T. CHENG, M. KANG, H. SHIM AND Y.-H. TSAI, *Geometric Optics in a Phase-Space-Based Level Set and Eulerian Framework*, J. Comput. Phys., 179, 2 (2002), pp. 622-648.

[242] S. J. OSHER AND R. FEDKIW, *Level Set Methods and Dynamic Implicit Surfaces*, Springer, (2002).

[243] S. J. OSHER AND F. SANTOSA, *Level Set Methods for Optimization Problems Involving Geometry and Constraints I. Frequencies of a Two-Density Inhomogeneous Drum*, J. Comput. Phys., 171 (2001) pp. 272–288.

[244] S. J. OSHER, P. SMEREKA AND M. SUSSMAN, *A Level Set Approach for Computing Solutions to Incompressible Two-phase Flow*, Journal of Computational Physics, 114 (1994), pp. 146–159.

[245] S.J. OSHER AND J.A. SETHIAN, *Fronts propagating with curvature-dependent speed : Algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.

[246] S.J. OSHER AND C.-W. SHU, *High order essentially non-oscillatory schemes for Hamilton-Jacobi equations*, SIAM J. Numer. Anal., 28, 4, (1991), pp. 907–922.

[247] D.L. PAGE, Y. SUN, A.F. KOSCHAN, J. PAIK AND M.A. ABIDI, *Normal Vector Voting: Crease Detection and Curvature Estimation on Large, Noisy Meshes* , Graphical Models, 64 (2004), pp. 199–229.

[248] O. PANTZ, *Sensibilité de l'équation de la chaleur aux sauts de conductivité*, C. R. Acad. Sci. Paris, Ser. I 341 pp.333-337 (2005).

[249] O. PANTZ AND K. TRABELSI, *Simultaneous shape, topology, and homogenized properties optimization*, Struct. Multidiscip. Optim., 32(3), (2007), pp. 361–365.

[250] P. PEBAY, *Construction dÕune triangulation surfacique Delaunay-admissible*, INRIA : Technical Report, 3369 (1998).

[251] M. A. PELETIER, *Newton's Problem of the Body of Minimal Resistance*, url: `http://www.win.tue.nl/~mpeletie/ Research/PubsNewton.shtml`.

[252] J. PERAIRE, M. VADHATI, K. MORGAN AND O.C. ZIENKEWICZ, *Adaptive remeshing for compressible flow simulations*, J. Comput. Phys., 72 (1987) pp. 449–466.

[253] P.-O. PERSSON AND G. STRANG, *A Simple Mesh Generator in MATLAB*, SIAM Review, Vol. 46 (2), (2004) pp. 329–345.

[254] P.-O. PERSSON, *Mesh Generation for Implicit Geometries*, Ph.D. thesis, Department of Mathematics, MIT, (2004).

[255] O. PIRONNEAU, *Optimal Shape Design for Elliptic Systems*, Springer, (1984).

[256] O. PIRONNEAU, *The finite element methods for fluids.*, Wiley (1989).

[257] O. PIRONNEAU, *Finite Element Characteristic Methods Requiring no Quadrature*, Journal of Scientific Computing, vol 43, *nb*.3 (2010), pp. 402-415.

[258] O. PIRONNEAU, *On the Transport-Diffusion Algorithm and its Applications to the Navier-Stockes Equations*, Numerische Mathematik, vol 38 (1982), pp. 309-332.

[259] O. PIRONNEAU, F. HECHT, A. LE HYARIC, *FreeFem++ version 2.15-1*, `http://www.freefem.org/ff++/`.

[260] J.-P. PONS AND J.-D. BOISSONAT, *Delaunay Deformable Models: Topology-Adaptive Meshes Based on the Restricted Delaunay Triangulation*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (1997).

[261] E. PRADOS, O. FAUGERAS AND E. ROUY, *Shape-from-Shading and Viscosity Solutions*, INRIA: Technical Report, 4638 (2002).

[262] J. QIAN, Y.T. ZHANG AND H. ZHAO, *Fast Sweeping Methods for Eikonal Equations on Triangular Meshes*, SIAM J. Numer. Anal., 45 (2007), pp. 83–107.

[263] T. LASSILA, A. MANZONI, A. QUARTERONI AND G. ROZZA , *Boundary control and shape optimization for the robust design of bypass anastomoses under uncertainty*, ESAIM: Mathematical Modelling and Numerical Analysis, (2013), 24 p.

[264] A. RASSINEUX, *Generation and optimization of tetrahedral meshes by advancing front technique*, Int. J. Numer. Methods in Engineering, 41 (1998), pp. 651–674.

[265] J.-F. REMACLE, C. GEUZAINE, G. COMPÈRE AND E. MARCHANDISE, *High Quality Surface Remeshing Using Harmonic Maps*, Int. J. Numer. Methods in Engineering, 83 (2009), pp. 403–425.

[266] S. RIPPA, *Minimal roughness property of the Delaunay triangulation*, Computer Aided Geometric Design, 7, (1990), pp. 489-497.

[267] S. RIPPA, *Long and thin triangles can be good for linear interpolation*, SIAM J. Numer. Anal., 29, 1 (1992), pp 257–270.

[268] M.-C. RIVARA, *Mesh refinement processes based on the generalized bisection of simplices*, SIAM J. Numer. Anal., 21 (1984), pp. 604–613.

[269] A. ROUY AND E. TOURIN, *A viscosity solutions approach to shape-from-shading*, SIAM J. Numer. Anal., 29, 3, (1992), pp. 867–884.

[270] L.I. RUDIN AND S.J. OSHER, *Total Variation Based Image Restoration with Free Local Constraints*, ICIP 1, IEEE (1994), pp. 31–35.

[271] M. RUMPF AND B. WIRTH, *Discrete geodesic calculus in the space of viscous fluidic objects*, submitted, (2012).

[272] J. RUPPERT AND R. SEIDEL, *On the Difficulty of Triangulating Three-Dimensional Nonconvex Polyhedra.*, Discrete & Computational Geometry 7(3), (1992), pp 227–253.

[273] J. RUPPERT, *A Delaunay Refinement Algorithm for Quality 2-Dimensional Mesh Generation*, Journal of Algorithms 18(3), (1995), pp 548–585.

[274] J.A. SETHIAN, *Level Set Methods and Fast Marching Methods : Evolving Interfaces in Computational Geometry,Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University Press, (1999).

[275] J.A. SETHIAN, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. USA Vol. 93, (1996), pp. 1591–1595.

[276] J.A. SETHIAN, *Fast Marching Methods*, SIAM Review, 41 (1999), pp. 199–235.

[277] J.A. SETHIAN AND P. SMEREKA, *Level Set Methods for Fluid Interfaces*, Annual Review of Fluid Mechanics, 35 (2003), pp. 341–372.

[278] J. A. SETHIAN AND A. WIEGMANN, *Structural boundary design via level set and immersed interface methods*, J. Comput. Phys., 163, 2 (2000), pp. 489-528.

[279] M. S. SHEPHARD AND M. K. GEORGES, *Automatic three-dimensional mesh generation by the finite octree technique*, Int. J. Numer. Methods in Engineering, 32, 4 (1991), pp. 709–749.

[280] J. R. SHEWCHUK, *Lecture Notes on Delaunay Mesh Generation*, Lecture notes, available at `http://www.cs.berkeley. edu/~jrs/meshpapers/delnotes.pdf` (2012).

[281] J. R. SHEWCHUK, *Unstructured Mesh Generation*, in Combinatorial Scientific Computing (Uwe Naumann and Olaf Schenk, editors), CRC Press, (2012), pp. 257–297.

[282] J. R. SHEWCHUK, *What is a Good Linear Element? - Interpolation, Conditioning, and Quality Measures* , Proc. 11th Int. Meshing Roundtable, (2002), pp. 115–126.

[283] R. SHU, C. ZHOU, AND M. S. KANKANHALLI, *Adaptive marching cubes*, in The Visual Computer, 11, (1995), pp. 202–217.

[284] H. SI AND K. GÄRTNER, *3D boundary recovery by constrained Delaunay tetrahedralization*, Int. J. Numer. Meth. Engrg., 85, (2011), pp. 1341–1364.

[285] O. SIGMUND, *Design of multiphysics actuators using topology optimization-part ii: Two-material structures*, Computer methods in applied mechanics and engineering, 190(49):6605-6627 (2001).

[286] O. SIGMUND, *Manufacturing tolerant topology optimization*, Acta Mech. Sin., 25, (2009), pp. 227–239.

[287] O. SIGMUND AND S. TORQUATO, *Design of materials with extreme thermal expansion using a three-phase topology optimization method*, Journal of the Mechanics and Physics of Solids, 45(6):1037-1067 (1997).

[288] P.K. SMOLARKIEWICZ, *The Multi-Dimensional Crowley Advection Scheme*, Monthly Weather Review, vol 110 (1982), pp. 1968-1983.

[289] J. SOKOŁOWSKI, A. ŻOCHOWSKI, *Topological derivatives of shape functionals for elasticity systems.* Mech. Structures Mach., 29, no. 3, 331–349 (2001).

[290] J. SOKOLOWSKI AND J.-P. ZOLESIO, *Introduction to Shape Optimization; Shape Sensitivity Analysis*, Springer Series in Computational Mathematics, 16, Springer, Heidelberg (1992).

[291] P. E. SOUGANIDIS, *Approximation Schemes for Viscosity Solutions of Hamilton-Jacobi Equations*, J. Differential Equations, 59, no. 1 (1985) pp. 1–43.

[292] J. STRAIN, *Semi-Lagrangian Methods for Level Set Equations*, J. Comput. Phys., 151 (1999) pp. 498–533.

[293] N. SUKUMAR, D.L. CHOPP, N. MOES AND T. BELYTSCHKO, *Modeling holes and inclusions by level sets in the extended finite element method ,* Comput. Methods Appl. Mech. Engrg., 190, (2001), 6183-6200.

[294] V. SURAZHSKY, P. ALLIEZ AND C. GOTSMAN, *Isotropic Remeshing of Surfaces: a Local Parameterization Approach*, RR-4967 INRIA (2003).

[295] V. SURAZHSKY AND C. GOTSMAN, *Explicit surface remeshing*, in proceedings of Eurographics Symposium on Geometry Processing (2003), pp. 17–28.

[296] S. SURESH, A. MORTENSEN, *Fundamentals of functionally graded materials*, London: Institute of Materials (1998).

[297] M. SUSSMAN, P. SMEREKA AND S. OSHER, *A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow*, J. Comput. Phys., 114, 1 (1994), pp. 146-159.

[298] M. SUSSMAN AND E. FATEMI, *An Efficient, Interface-Preserving Level Set Redistancing algorithm and its Applications to Interfacial Incompressible Fluid Flow*, SIAM J. Sc. Comp., 20 (1999), pp. 1165–1191.

[299] M. SUSSMAN, E. FATEMI AND P. SMEREKA AND S. OSHER, *An Improved Level-Set Method for Incompressible Two-Phase Flows*, Computers and Fluids, 27 (1997), pp. 663–680.

[300] K. SVANBERG, *The method of moving asymptotes - a new method for structural optimization*, Int. J. Numer. Meth. Engng., 24, pp.359-373 (1987).

[301] K. SVANBERG, *The method of moving asymptotes - a new method for structural optimization*, Int. J. Numer. Meth. Engng., 24, pp.359-373 (1987).

[302] V. ŠVERAK, *On optimal shape design*, J. Math. Pures Appl., 72, 6, (1993), pp.537-551.

[303] C.C. SWAN, I. KOSAKA, *Voigt-Reuss topology optimization for structures with linear elastic material behaviors*, Int. J. Numer. Methods Engrg. 40 (1997).

[304] T.T.M. TA, *Thèse de l'Université Pierre et Marie Curie* (in preparation)

[305] L. TARTAR, *The general theory of homogenization. A personalized introduction*, Lecture Notes of the Unione Matematica Italiana, 7. Springer-Verlag, Berlin; UMI, Bologna (2009).

[306] L. TARTAR, *An introduction to homogenization method in optimal design*, Lectures notes in mathematics. Springer (2000).

[307] G. TAUBIN, *Estimating The Tensor Of Curvature Of A Surface From A Polyhedral Approximation*, Proc. ICCVÕ95, (1995), pp 902–907.

[308] R. TILLEY, *Understanding Solids: The Science of Materials*, Wiley (2004).

[309] G. TRYGGVASON, B. BUNNER, A. ESMAEELI, D. JURIC, N. AL-RAWAHI, W. TAUBER, J. HAN, S. NAS, AND Y.-J. JAN, *A Front-Tracking Method for the Computations of Multiphase Flow*, J. Comput. Phys., 169, pp. 708-759 (2001).

[310] G. TURK AND M. LEVOY, *Zippered Polygon Meshes from Range Images*, ACM Transactions on Graphics, Special issue on Proceedings of SIGGRAPH 1994 (1994), pp. 311–318.

[311] M.G. VALLET, F. HECHT AND B. MANTEL, *Anisotropic Control of Mesh Generation Based upon a Voronoi Type Method*, Numerical Grid Generation in Computational Fluid Dynamics and Related Fields, (1991).

[312] G.N. VANDERPLAATS AND F. MOSES, *Structural optimization by methods of feasible directions*, Computers & Structures, 3, pp.739-755 (1973).

[313] C. VITERBO, *Systèmes dynamiques et équations différentielles*, Cours de l'Ecole Polytechnique (2009).

[314] N. VERMAAK, G. MICHAILIDIS, Y. BRECHET, G. ALLAIRE, G. PARRY AND R. ESTEVEZ, *Material Interface Effects on the Topology Optimization of Multi-Phase Thermoelastic Structures Using A Level Set Method*, In preparation.

[315] L.A. VESE AND T.F. CHAN, *A multiphase level set framework for image segmentation using the Mumford and Shah model*, Int. J. Comput. Vision 50 (3) (2002) 271-293.

[316] A. VLACHOS, J. PETERS, C. BOYD AND J.L. MITCHELL, *Curved PN Triangles*, Symposium on Interactive 3*d* Graphics, (2001), pp 159–166.

[317] J. VOLLMER, R. MENCL, AND H. MÜLLER, *Improved Laplacian Smoothing of Noisy Surface Meshes*, Computer Graphics Forum, (1999), pp. 131–138.

[318] D. WANG, O. HASSAN, K. MORGAN AND N. WEATHERILL, *Enhanced remeshing from STL files with applications to surface grid generation*, Communications in Numerical Methods in Engineering, 23, (2006); pp. 227–239.

[319] M.Y. WANG, X. WANG, D. GUO, *A level set method for structural topology optimization,* Comput. Methods Appl. Mech. Engrg., 192, 227–246 (2003).

[320] M. WANG, S. CHEN, X. WANG AND Y. MEI, *Design of Multimaterial Compliant Mechanisms Using Level-Set Methods*, J. Mech. Des. 127(5), 941-956 (2005).

[321] M. WANG AND X. WANG, *Color level sets: a multi-phase method for structural topology optimization with multiple materials*, Comput. Methods Appl. Mech. Engrg. 193, 469-496 (2004).

[322] M. WANG AND X. WANG, *A level-set based variational method for design and optimization of heterogeneous objects*, Computer-Aided Design 37, 321-337 (2005).

[323] M. WICKE, D. RITCHIE, B. M. KLINGNER, S. BURKE, J. R. SHEWCHUK, AND J. F. O'BRIEN, *Dynamic Local Remeshing for Elastoplastic Simulation*, ACM Trans. Graph. 29, 4 (2010), pp. 1–11.

[324] C. WOJTAN, N. THÜREY, M. GROSS AND G. TURK, *Deforming Meshes that Split and Merge*, ACM Trans. Graph. 28, 3 (2009), art. 76.

[325] Q. XIA, T. SHI, S. LIU, M. Y. WANG, *A level set solution to the stress-based structural shape and topology optimization*, Computers and Structures, 90-91, pp. 55-64 (2012).

[326] S. YAMASAKI, T. NOMURA, A. KAWAMOTO, S. NISHIWAKI, *A level set-based topology optimization method targeting metallic waveguide design problems*, Int. J. Numer. Meth. Engng , 87, pp.844-868 (2011).

[327] D.-M. YAN, B. LÉVY, Y. LIU, F. SUN AND W. WANG, *Isotropic Remeshing with Fast and Exact Computation of Restricted Voronoi Diagram*, Computer Graphics Forums (Proc. of SGP), 28(5), (2009), pp. 1445–1454.

[328] L. YIN AND G.K. ANANTHASURESH, *Topology optimization of compliant mechanisms with multiple materials using a peak function material interpolation scheme*, Struct. Multidiscip. Optim. 23, 49-62 (2001).

[329] L. YOUNES, *Shapes and Diffeomorphisms*, Springer, Applied Mathematical Sciences, vol. 171 (2010).

[330] A. ZAHARESCU, E. BOYER AND R.P. HORAUD, *Topology-Adaptive Mesh Deformation for Surface Evolution, Morphing, and Multi-View Reconstruction*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 33, 4, (2011), pp. 823–837.

[331] S.T. ZALESAK, *Fully Multidimensional Flux-Corrected Transport Algorithms for Fluids*, Journal of Computational Physics, 31 (1979), pp. 335–362.

[332] H. ZHAO, *A fast sweeping method for Eikonal equations*, Mathematics of Computation, Vol. 74, (2010), pp. 603–627.

[333] H. ZHAO, S.J. OSHER AND R. FEDKIW, *Fast Surface Reconstruction Using the Level Set Method*, Proceedings of IEEE Workshop on Variational and Level Set Methods in Computer Vision, (2001).

[334] K. ZHOU, J. HUANG, J. SNYDER, X. LIU, H. BAO, B. GUO AND H.-Y. SHUM, *Large Mesh Deformation Using the Volumetric Graph Laplacian*, ACM Trans. Graph. 24, 3 (2005), pp. 496-503.

[335] S. ZHOU AND Q. LI, *Computational design of multi-phase microstructural materials for extremal conductivity*, Computational Materials Science, 43:549-564 (2008).

[336] S. ZHOU AND M.Y. WANG, *Multimaterial structural optimization with a generalized Cahn-Hilliard model of multiphase transition*, Struct. Multidisc. Optim. 33:89-111 (2007).

[337] J. ZHU AND J.A. SETHIAN, *Projection Methods Coupled to Level Set Interface Techniques*, J. Comput. Phys., 102, (1992), pp. 128–138.

[338] O.C. ZIENKIEWICZ AND J.S. CAMPBELL, *Shape optimization and sequential linear programming*, in: R.H. Gallagher and O.C. Zienkiewicz, eds., Optimum Structural Design, Wiley, New York, (1973), pp. 109–126.