UCLouvain
Institute of Mechanics,
Materials and Civil Engineering

Metric-based curvilinear mesh generation and adaptation

Ruili Zhang

Thesis submitted in partial fulfillment of the requirements for the Ph.D
Degree in Engineering Sciences

Thesis Committee

Advisors	Jean-François Remacle	*UCLouvain*
Jury	Vincent Legat	*UCLouvain*
	Jonathan Lambrechts	*UCLouvain*
	Thierry Coupez	*MINES ParisTech*
	Thomas Toulorge	*Cenaero*
Chairperson	Renaud Ronsse	*UCLouvain*

Louvain-la-Neuve, February 2022

Abstract

This thesis aims at curvilinear mesh generation and adaptation, we start from summary ideally the research project as a simple yet fundamental question: assume a unit square

$$\Omega = \{(x^1, x^2) \in [0,1] \times [0,1]\}$$

and a smooth function $f(x^1, x^2)$ defined on the square, and consider a mesh $\mathcal{T}$ made of P^2 triangles that exactly covers the square, how can we compute the mesh $\mathcal{T}$ that minimizes the interpolation error $\|f - \Pi f\|_\Omega$. Here, Π is the nodal interpolation of f on the mesh Ern and Guermond (2013).

We state the problem as to build a unit curvilinear mesh, i.e. build a mesh with unit edge lengths that are possibly curvilinear. We solve the problem in three stages of increasing complexity: 1) input a unit square and a metric field $g(x, y)$, to build a unit curvilinear mesh that is possibly anisotropic; 2) input a function $f(x, y)$, to build an anisotropic curvilinear mesh that minimizes the approximation error; 3) input a high order finite element solution, to build an anisotropic curvilinear adapted mesh.

We propose *a new framework* of curvilinear mesh generation and adaptation: metric field construction, generation of points (point sampling on the boundary and point sampling in the domain), straight-sided mesh generation and adaptation (triangulation and straight-sided edges swap), curvilinear mesh generation and adaptation (straight-sided edges curving, curvilinear edges swap, and Curvilinear Small Polygon Reconnection). A unit curvilinear mesh containing only valid "Geodesic Delaunay triangles" is obtained this way. In this approach, the curvature is not only used to match curved boundaries but also to capture features of the interpolated solutions, and it results in meshes that would not have been achievable by simply curving *a posteriori* a straight-sided mesh. A number of application examples are presented in order to demonstrate the capabilities of the mesh adaptation procedure.

Thanks

In the end of my Ph.D. research journey, I would like to express my gratitude to all the people who accompanied me all along this long journey and given me support and assistance.

First, I would like to express my sincere gratitude to my supervisor, Prof. Jean-François Remacle, who always have good ideas. His motivation and enthusiasm on science and life make me have a very wonderful Ph.D. journey. I appreciate the trust and continuous support he showed from the very beginning.

Sincere gratitude to Dr. Amaury Johnen, who always tried to do his best to help me, give me advises, say what to do, explain what I did not understand, show how things work, show how to use tools I never used, etc.

Sincere gratitude to the members of my supervisory committee Prof. Vincent Legat and Dr. Jonathan Lambrechts, and the members of my doctoral jury Prof. Renaud Ronsse, Prof. Thierry Coupez, Dr. Thomas Toulorge for their attention on reading my thesis and the valuable insights and comments during my defense.

Give sincere gratitude to our Hextreme team members: Alexandre, Jeanne, Kilian, Jovana, Célestin, Ruiyang, Pierre-Alexandre, Christos, Maxence, Arthur, Alexis, Mattéo, Thomas and Antoine. As friends and colleague, I would like to thank all you for sharing your knowledge and experience, and all the interesting discussions we had not only on science but also on life.

Finally, give sincere gratitude to all the beloved ones who have been with me during the period. In particular, thanks to my family and friends for all the constant support they gave me in my life.

Ruili Zhang

Contents

Chapter 1

Introduction

The story of finite element mesh generation started about fifty-year ago, about at the same time finite elements started to become popular in the engineering community. To our best knowledge, two research teams, one in US and one in France both started to work on an algorithm to automatically generate triangular meshes on 2D domains. After a couple of years, both research teams came with solutions and their respective national research agencies pushed them to extend their results in 3D. They rapidly realized how huge was the gap between 2D and 3D. Fifty years after, 3D mesh generation is still a very active research subject and dealing with meshes is still the nightmare of finite element practitioners.

This thesis addresses a brain new topic in mesh generation: adaptive curvilinear meshing. It is indeed quite nice that our research question can be explained within a very few words. Consider a smooth function $f(x, y)$ defined say on the unit square $(x, y) \in [0, 1] \times [0, 1]$. Consider a geometrically quadratic mesh $\mathcal{M}$ on the square that is potentially *anisotropic and curvilinear.* Using an isoparametric finite element interpolation of f on that mesh leads to define a piecewise quadratic interpolation of $f(x,y)$ that we call $f_h(x,y)$. *How can we build* $\mathcal{M}$ *in such a way that the interpolation error* $\|f - f_h\|$ *(using some norm) is minimal?* We are pretty much the first ones that have entered that topic so the state of the art on the subject is almost empty. Pr. Coupez Coupez (2017) has been an inspiration. Yet, it is still important to draw a line between the early developments in mesh generation and our research subject in order to "put ourselves on the map". We will thus start the introduction by giving three central questions that motivate this research work, and then browsing different aspects of finite element mesh generation and relate them to our research question when it is relevant.

1 | Three central questions motivate this research work

After reviewing the state of mesh generation, we see the advantages of metric-based mesh generation - say, the adaptation mesh is obtained directly at the mesh generation step rather than doing a modification of an existing mesh by enrichment or moving points; and the advantages of boundary curvilinear mesh generation - say, accurately capturing the geometry of the surfaces using high-order meshes; and we also see the lack to coupled these two advantages and the domain curvilinear mesh generation. In this thesis, based on and taking advantage of well-defined Riemannian differential geometry, we extend the state of mesh generation to a new branch - metric-based curvilinear mesh generation and adaptation in computation domain, considering both the curved geometry and curved solution features. The main idea is to generate a mesh whose edges are the unit geodesics in a prescribed Riemannian metric space, and then to build a unit quadratic mesh, i.e. a mesh that has quasi-unit curvilinear edges and quasi-unit curvilinear triangles. The geodesic between two points as well as the unit geodesic starting at a given point with a given direction are the two main tools that allow us to address our issue. Our mesh generation procedure is done in two steps. At first, points are distributed in a frontal fashion, ensuring that two points are never too close to each other in the geodesic senses, and a simple isotropic Delaunay triangulation of those points is created. Then, straight edge swaps, straight edge curving, curvilinear edge swaps and Curvilinear Small Polygon Reconnection (CSPR) are performed in order to build the unit mesh. Notions of curvilinear mesh quality is defined as well that allow to drive the edge swapping and Curvilinear Small Polygon Reconnection (CSPR) procedure.

In this thesis, we focus on the method to generate curvilinear adapted meshes. Metric fields based on Hessians of function are still the right tools for driving mesh adaptation at higher orders. Thus in the examples of Chapter 2, metric fields that have been used are based on the Hessians of function. But we should remark that to construct $\boldsymbol{g}(x^1, x^2)$ based on Hessians of a function is not correct anymore for higher orders of approximation. We extend our research to build metric fields that are suited for driving high orders mesh adaptation in Chapter 3. In our research, we propose a way to build the metric field at nodes rather than just averaging the element based metrics. For gradient interpolation and Hessian computation, we take the similar method as shown in Rusinkiewicz (2004). It is easy generalizability to derivatives of any order higher-order differential properties on triangular meshes by this way.

To assess the metric-based curvilinear mesh generation and adaptation process along with its efficiency and accuracy, we apply it to analytic metric fields (Chapter 2), analytic functions (Chapter 3), and numerical solutions of PDEs (Chapter 4).

1.1 Given a unit square and a metric field $M(x, y)$, how to build a unit curvilinear mesh that is possibly anisotropic

In Chapter 2, we aim at addressing the following issue. Assume a unit square: $\Omega = \{(x^1, x^2) \in [0,1]\times[0,1]\}$ and a Riemannian metric $g_{ij}(x^1, x^2)$ defined on U. Assume a mesh $\mathcal{T}$ of U that consists in non overlapping valid quadratic triangles that are potentially curved. Is it possible to build a unit quadratic mesh of U i.e. a mesh that has quasi-unit curvilinear edges and quasi-unit curvilinear triangles ? This Chapter 2 aims at providing an embryo of solution to the problem of curvilinear mesh adaptation. The method that is proposed is based on standard differential geometry concepts. At first, the concept of geodesics in Riemannian spaces is quickly presented: the geodesic between two points as well as the unit geodesic starting at a given point with a given direction are the two main tools that allow us to address our issue. Our mesh generation procedure is done in two steps. At first, points are distributed in the unit square U in a frontal fashion, ensuring that two points are never too close to each other in the geodesic sense. Then, a simple isotropic Delaunay triangulation of those points is created. Curvilinear edge swaps as then performed in order to build the unit mesh. Notions of curvilinear mesh quality is defined as well that allow to drive the edge swapping procedure. Examples of curvilinear unit meshes are finally presented.

1.2 Given a function $f(x, y)$, how to build an anisotropic curvilinear mesh that minimizes the approximation error

In Chapter 3, we propose a new framework for the generation and adaptation of unit curvilinear P^2 meshes in dimension 2. In this approach, curvature is not only used to match curved boundaries but also to capture features of the interpolated solutions, and it results in meshes that would not have been achievable by simply curving *a posteriori* a straight-sided mesh. We proceed as follows. Starting with a smooth function $f(x, y)$, a metric field, based on f and its derivatives up to order 3, is constructed. A unit P^2 mesh is then generated, with edges within an adimensional length range of $[0.7, 1.4]$ with respect to this metric field. Points are then spawned in such a way that their geodesic distance corresponds to edges of unit size, and these points are then connected in a standard *isotropic* fashion. A curvilinear mesh quality criterion is then proposed to drive the mesh optimization process. The triangulation is subsequently modified using straight-sided edge swap, straight-sided edge curving, curvilinear edge swap and Curvilinear Small Polygon Reconnection (CSPR) to form the desired unit mesh. A unit curvilinear mesh containing only valid "Geodesic Delaunay triangles" is obtained this way. A number of application examples are presented in order to demonstrate the capabilities of the mesh adaptation procedure. The resulting adapted meshes allow, most of the times, a significant reduction of the approximation error compared with straight-sided P^2 meshes of the same density.

1.3 Given a high order finite element solution, how to build an anisotropic curvilinear adapted mesh

In Chapter 4, we move to replace the analytical functions with high-order finite element solutions. We go forward and use computational fluid dynamics (CFD) results to create adapted meshes. The CFD code that is used here is a high order finite element code. The finite element approximation that is used is continuous. Standard Lagrange shape functions on (possibly) curvilinear triangles are used to approximate velocity and pressure field.

Until now, there is very little literature on this subject - curvilinear mesh generation and adaptation - and our work is pioneer.

2 | Finite element mesh generation

Finite elements have become the most common numerical analysis tool in engineering. Finite elements are used in automotive, aerospace, ship building, biomedical and many more industries. The main reasons of that success are i) the ability of finite element to solve most of the relevant non-linear problems in engineering, ii) the nice structure of proof that is endowed with finite element formulations and iii) the ability of finite elements to handle complex geometries.

The finite element analysis procedure is often broken up into the following five principal steps

1. Mesh Generation: subdividing the geometrical domain into *finite elements*;
2. Local formulation: computation of local elementary matrices;
3. Assembly: obtaining the equations of the entire system from the local matrices;
4. Solving the system of equations (linear or not);

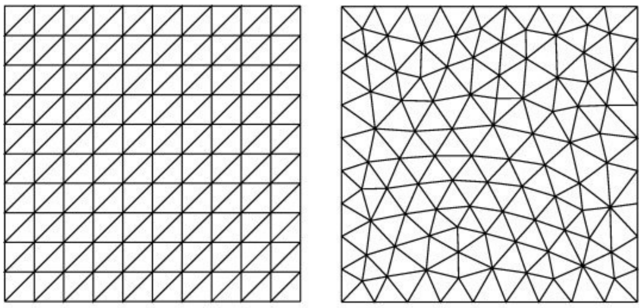


Figure 1.1: Structured mesh (left) and Unstructured mesh (right) Hiester et al. (2014).

5. Postprocessing: determining quantities of interest, such as stresses and strains, and obtaining some visualization of the solution.

Mesh generation is one of the 5 mandatory steps of finite element analysis. The ability to generate tetrahedral meshes in general $3D$ domains has allowed finite elements to handle the geometrical complexity of engineering parts. Yet, this has a price: mesh generation still considered as the main bottleneck of the finite element pipeline. Mesh generation algorithms are very complex pieces of codes and only a few research teams are producing finite element mesh generators that are of industrial robustness.

Not only it is difficult to generate a mesh, but a mesh must be a quality mesh to enable quality results. Indeed, it is well known in the community that mesh quality greatly impacts the efficiency, the stability and the accuracy of finite elements Freitag and Ollivier-Gooch (2000).

Definition: A mesh $\mathcal{M}$ is a geometrical discretization of a domain Ω that consists of

- A collection of mesh entities $\mathcal{M}_i^d$ of controlled size and distribution;
- Topological relationships or adjacencies forming the graph of the mesh.

In a finite conforming element mesh, elements can only intersect at vertices, edges or faces. This excludes the existence of overlapping elements or T-junctions.

There are various ways of *classifying meshes*:

- **2D vs. 3D.** In engineering analysis, meshes can be either 2-dimensional or 3-dimensional. As said before, the 3D case is clearly not an extension of the 2D case. In this thesis, we stay in the plane and thus consider 2D meshes exclusively.

- **Structured vs. Unstructured.** People usually distinguish structured meshes and unstructured meshes (see Figure 1.1). Structured mesh topological relationships are implicit. For example, in a 2D structured mesh, vertex M_i^1 knows implicitly all its adjacent edges, vertices and faces. Unstructured mesh topologies are explicit: the adjacencies of every mesh entity must be explicitly stored. In this work, we are on the unstructured side.

- **Tet/Tri vs. Hex/Quad.** Mesh entities can be of various types: triangles and quads in $2D$ and tetrahedra, hexahedra, prisms and pyramids in 3D (see Figure 1.2). A mesh is said to be hybrid if it contains at least two different types of mesh entities. In this thesis, we consider triangular meshes only.

- **Linear vs. High-Order.** Now comes the classification of the geometry of mesh entities. Meshes can be straight-sided/linear or curvilinear/high-order (see Figure 1.3). A linear mesh has all its edges straight sided while high-order meshes can have edges that are curved. In this thesis, we are dealing with curvilinear elements.

- **Uniform vs. Adaptive.** A mesh can be adaptive or uniform. An adaptive mesh has mesh sizes that may vary over the domain. In section 4, we will present the classical theory of H-adaptivity.

- **Isotropic vs. Anisotropic.** A mesh can be isotropic or anisotropic (see Figure 1.4). Anisotropic meshes are now quite common in the mesh generation community. We will discuss that topic in the section 4 as well.

This classification is obviously not exhaustive. Yet, it allows us to position this research work in the broad domain of finite element mesh generation. This thesis presents some seminal results on the generation of *high-order anisotropic adaptive triangular meshes*. Thus, we are working in $2D$, we use unstructured adaptive triangular meshes that are both curvilinear and anisotropic. In the following subsections of this introduction, we will give an overview of the different aspects of mesh generation that are actually used in our work.

2.1 Mesh generation – the simplical case

Even though efficient 2D mesh generation techniques were already available in the early 1970's Lawson (1972), the first automatic unstructured mesh generation system for general 3D domains was proposed in the early 1990's with Paul-Louis George's seminal work on 3D constrained Delaunay triangulation George et al. (1990). It is interesting to note that today's most widely used 3D mesh generation algorithm is still the one developed at that time by those 3 authors. Three-dimensional mesh generation is a problem that is extraordinary complicated. Only half a dozen research teams Si (2015); Boissonnat et al. (2002); Lévy in the world have the technology to build

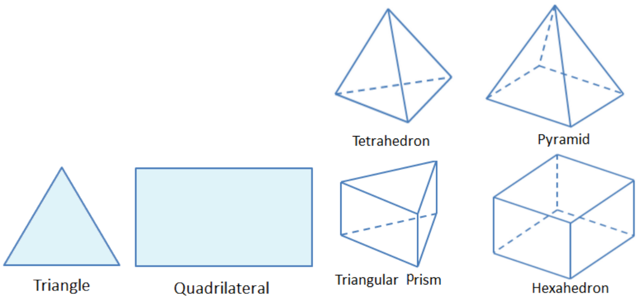


Figure 1.2: The different kind of elements used in a finite element mesh, both in 2D (left) and in 3D (right) Wikipedia contributors (2022).

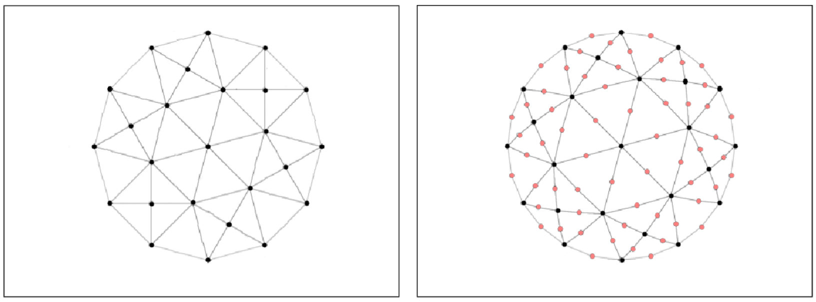


Figure 1.3: Linear mesh (left) and High-order quadratic mesh (right) Ims et al. (2015).

tetrahedral meshes for general 3D domains in an automatic manner. My advisor, Pr. Jean-François Remacle belongs to this short list with Gmsh Geuzaine and Remacle (2009) , the only open source complete mesh generator available today. Célestin Marot, one of the members of our research team, has developed the most advanced $3D$ mesh generator (to parallelize and optimize all stages of tetrahedral mesh generation) available today Marot et al. (2019, 2020); Marot and Remacle (2020). It is now fully integrated in Gmsh. In this work we will actually be lucky to be able to borrow some of Gmsh's algorithm to generate planar straight-sided triangular meshes, both isotropic and anisotropic.

The classical pipeline for generating a $2D$ mesh is the following. Consider the simple model of Figure 1.5. Meshing takes as input a domain $G \subset \mathbb{R}^2$ that has to be triangulated. The most common way to describe G is to use a boundary-based scheme where the geometric domain is represented as a set of topological entities together with adjacencies. Model vertices and model edges of G from a boundary representation of G. Each model edge is topologically oriented: it has a starting and

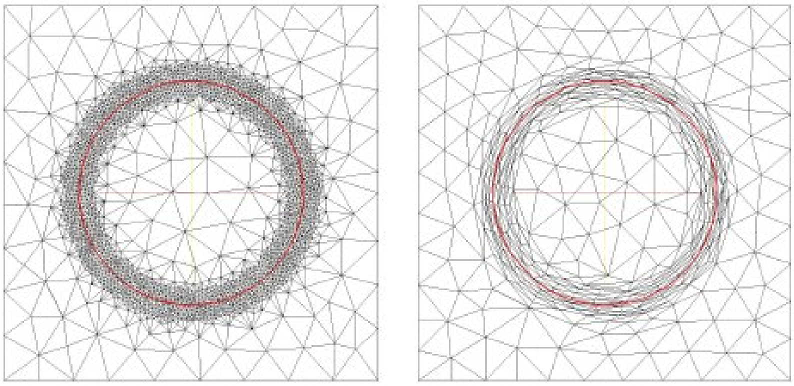


Figure 1.4: Isotropic mesh (left) and Anisotropic mesh (right) Coupez et al. (2010).

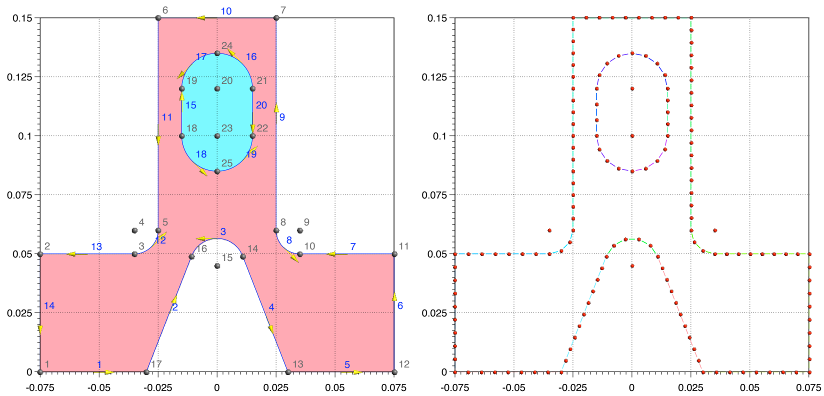


Figure 1.5: A simple $2D$ the model (left) and the $1D$ mesh (right)

a ending model vertex. Model faces of G are bounded by oriented model edges. As an example, Figure 1.5 presents a simple model that is composed of two model faces, 20 model edges and 25 model vertices. The first step for doing the mesh is to discretize (or mesh) the edges of the model. The result is presented in Figure 1.5 for a uniform mesh size. Then, each of the two model faces is triangulated.

The standard way of triangulating a model face starts by creating an *empty mesh* i.e. a mesh that contains all vertices of the $1D$ mesh and that contains all mesh edges of the $1D$ boundary. The empty mesh corresponding to the simple model of

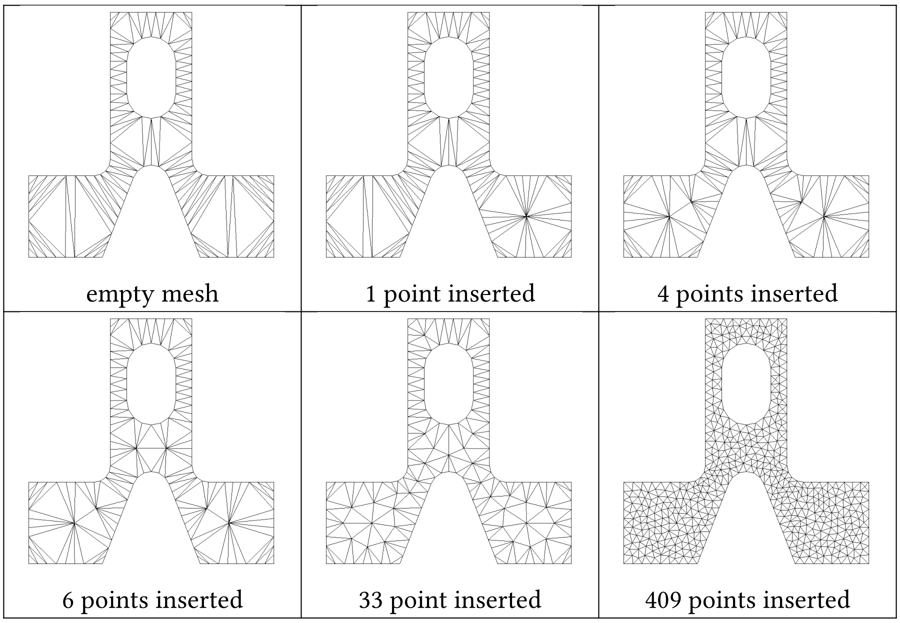


Figure 1.6: Illustration of the Bowyer-Watson algorithm with a point insertion scheme based on the circumcenter of the largest triangle

Figure 1.5 is presented on Figure 1.6. Then points are added inside the domain. Each time a new point is added, the triangulation is updated using the Bowyer-Watson algorithm. Figure 1.6 gives an example of the Bowyer and Watson algorithm with a point insertion based on the circumcenter of the largest triangle. Points on the final mesh are nicely distributed so that only a very light optimization (Laplace smoothing) is necessary for obtaining the final mesh.

In this thesis, our meshing approach is not the usual one for which points and triangles are generated at the same time. In our approach, points are generated first, and then connected in a second step. Points are generated using a frontal algorithm that is very similar to the one described in Baudouin et al. (2014); Georgiadis et al. (2017). We will come back in details on this frontal algorithm in further chapters. Nevertheless, with the aim of teasing the reader, Figure 1.7 shows a small illustration of the frontal procedure applied to the simple geometry of Figure 1.5.

2.2 Mesh generation – quad- and hex-meshing

Quadrilateral meshes in $2D$ and hexahedral meshes in 3D and are usually considered to be superior to triangular/tetrahedral meshes. In a computational perspective, a hexahedral mesh contains about 7 times fewer elements that a tetrahedral mesh

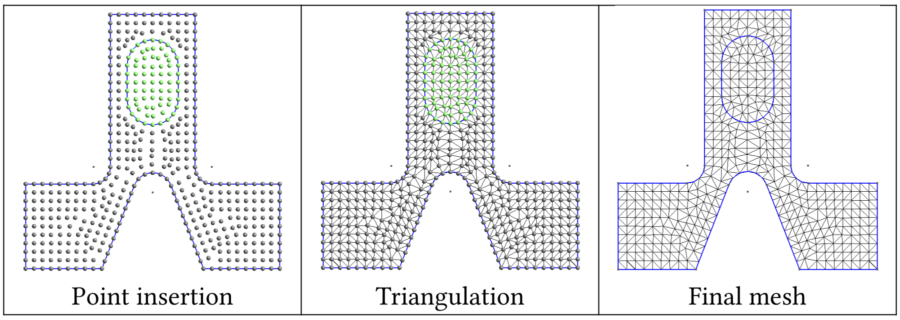


Figure 1.7: Illustration of the frontal procedure that is used in this thesis.

with the same number of nodes. This means less data storage and a faster assembly procedure. The availability of tensor-product basis functions allow to dramatically reduce the number of floating point operations for computing finite element operators Fischer (1997). The local Cartesian structure of the mesh allows the creation of natural overlaps between elements and the building of efficient local preconditioners, even for GPUs Remacle et al. (2016). There are also numerous modeling reasons to prefer hexes: boundary layers in CFD Puso and Solberg (2006), the inaccuracy or locking problems in solid mechanics Benzley et al. (1995).

Generating quad meshes in a reliable is still an open problem in mesh generation. Yet, some solutions exist. Block structured quad meshing was among the first techniques that allowed to generate finite element meshes. Those early methods were essentially manual Thompson et al. (1998). Research on automatic blocking is very active today Bommes et al. (2013). People from our group are also actively working on that subject Jezdimirovic et al. (2021). Research on generating unstructured quad meshing has also led to a variety of methods: paving Blacker and Stephenson (1991), indirect meshing Remacle et al. (2012, 2013a), quadtree methods Pascal and Marechal (1998)...

Hex meshing is way more complex. There exist manual or semi-automatic ways of generating block structured meshes that are available in industry. Yet, generating structured or unstructured hex-meshes on general 3D domains is still an open problem.

In our work, we do not consider quad- or hex-meshes, so we will not do a complete state of the art on hex meshing.

3 | Curvilinear mesh generation

3.1 High-order mesh generation

Scientific computing is now an old science. Solving partial differential equations on a computer is a very common task for aerospace/chemical/ mechanical/electrical engineers. Still, numerical methods for PDEs that have reached a production level such as finite elements are, for most of them, based on numerical schemes that are of the second order of accuracy. Some applications in fluid mechanics or in electromagnetic nonetheless require numerical schemes that are of higher order of accuracy (those schemes are sometimes called high fidelity schemes). It has been proved in many contributions that high-order finite element schemes require high-order meshes, i.e., meshes that capture the curvilinear features of the geometry with a high fidelity as well Bernard et al. (2009). In the last decade, a significant part of the research in mesh generation has thus been devoted to the generation of body fitted curvilinear meshes. The main issue of generating curved meshes is that there exists for now no algorithm that actually generates a P^2 mesh in a direct fashion. State-of-the-art methods generate a straight-sided mesh and place high-order points on the CAD geometry. Then, invalid elements are untangled using various approaches Fortunato and Persson (2016); Hartmann and Leicht (2016); Moxey et al. (2016); Karman et al. (2016); Ruiz-Gironés et al. (2016); Toulorge et al. (2013); Remacle et al. (2013b). Nowadays, body fitted curvilinear meshes start to be used in an industrial context Kroll (2006); Kroll et al. (2015).

High-order meshes have exclusively been used for increasing geometrical accuracy, i.e., to make the mesh represents the geometry of curved parts with high fidelity. The natural extension of the use of curvilinear meshes is *high-order/curvilinear mesh adaptation*. In the linear case, extensive work has been done in anisotropic mesh adaptationAlmeida et al. (2000); Buscaglia and Dari (1997); Castro-Díaz et al. (1997); Formaggia et al. (2004); Dompierre et al. (1997); Huang (2005); Frey and Alauzet (2005); Gruau and Coupez (2005); Li et al. (2005); Tam et al. (2000); Pain et al. (2001). The concept of metric tensor is always central in anisotropic adaptation: it allows to define mesh sizes and directions that allow to minimize the interpolation error Schall et al. (2004); Courty et al. (2006); Chen et al. (2007); Alauzet et al. (2006); Loseille and Alauzet (2011a,b). Yet, all those methods end up with a straight-sided mesh.

3.2 Body fitted meshes – curving and untangling

Let's start with a question. Why should we want high-order meshes while generating straight-sided ones is already so complex? This question is 100% valid and deserves an answer.

There is a growing consensus in the computational mechanics community that state of the art solver technology requires, and will continue to require too extensive computational resources to provide the necessary resolution for a broad range of demanding applications, even at the rate that computational power increases. The requirement for high resolution naturally leads us to consider methods which have a higher order of grid convergence than the classical (formal) 2 order provided by most industrial grade codes. This indicates that higher-order discretization methods will replace at some point the current finite volume and finite element solvers, at least for part of their applications.

The development of high-order numerical technologies for engineering analysis has been underway for many years now. For example, Discontinuous Galerkin methods (DGM) have been largely studied in the literature, initially in a theoretical context Cockburn et al. (2000), and now from the application point of view Kroll et al. (2010, 2015). In many contributions, it is shown that the accuracy of the method strongly depends on the accuracy of the geometrical discretization Bassi and Rebay (1997); Bernard et al. (2009); Toulorge and Desmet (2010).

Since 2012 and the first high order workshop for CFD Wang et al. (2013), researchers in CFD are actively working on high order meshing. The UCLouvain team Quan et al. (2014) has produced theoretical results and algorithms to generate valid *body-fitted* linear meshes for capturing geometries defined implicitely by high-order discretizations. All the work that was done was aiming at accurately capturing the geometry of the surfaces and ensuring that the resulting mesh was valid.

Modern mesh generation procedures take as input CAD[1] models composed of *model entities*: vertices G_i^0, edges G_i^1, faces G_i^2 or regions G_i^3. Each model entity G_i^d has a geometry (or shape) Geuzaine and Remacle (2009) for which solid modelers usually provide a parametrization, that is, a mapping $\boldsymbol{\xi} \in R^d \mapsto \boldsymbol{x} \in R^3$. There are also four kind of mesh entities M_i^d that are said to be classified on model entities[2]. Each mesh entity is classified on the model entity of the smallest dimension that contains it. The way of building a high order mesh is to first generate a straight sided mesh. Then, mesh entities (edges, faces and regions) are curved according to the geometry of the CAD entity it is classified on.

In the p-version of finite elements, high order nodes are added to edges, faces and regions of the element with the aim of creating curvilinear elements with their shape based on high order (Lagrangian or not) polynomial bases (see Figure 1.8) :

Other authors Dey et al. (1997); Dey (1997); Sevilla et al. (2008a,b) would rather use the exact mappings of the geometry and build a so-called isogeometric mesh.

The naive curving procedures described above does not ensure that all the elements of the final curved mesh are valid. Figure 1.9 gives an illustration of this important issue: some of the curved triangles are tangled after having been curved. It is important to note that this problem is not related to the accuracy of the geometrical discretization: in Figure 1.9, the mesh would not be valid in the iso-geometric case i.e. if the curved edge was assigned the exact geometry (blue curve).

[1]Computer Aided Design.

[2]We use the symbol $\sqsubset$ for indicating that a mesh entity is classified on a model entity.

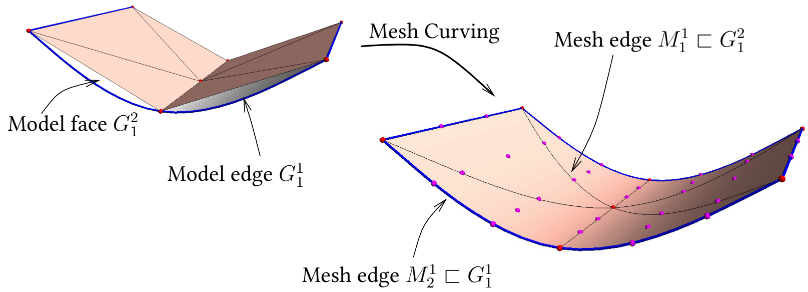


Figure 1.8: Straight sided mesh (left) and curvilinear (cubic) mesh (right).

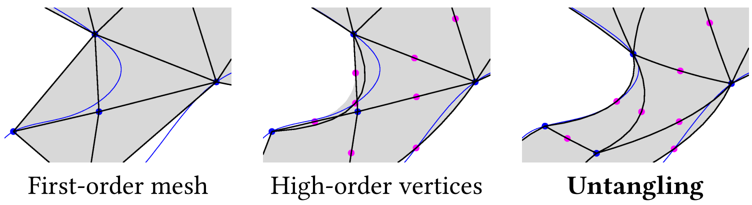


Figure 1.9: Straight sided mesh (left) basic curvilinear (quadratic) mesh with tangled elements (center) and untangled mesh (right).

In this work, the point of view is quite different. The curvature of the mesh will be used to *adapt to the curvilinear features of the solution* and not of the geometry. This is indeed quite different because a solution is not defined in term of this geometry (even though...).

3.3 Validity Bounds for Second Order Planar Triangles

In Figure 1.9, the basic curvilinear mesh is *visually* incorrect: one curvilinear mesh edge of one of the triangles intersect with the two other edges. In this work, we will use P^2 triangles exclusively. At that point it is interesting to recall a fundamental result on high order mesh validity Johnen et al. (2013).

The geometry of the six-node quadratic triangle is shown in Figure 1.10. Inspection reveals two types of nodes: corners (1, 2 and 3) and midside nodes (4, 5 and 6).

The determinant $J(\boldsymbol{\xi}) = J(\xi, \eta) = \det \left(\frac{d\mathbf{X}}{d\boldsymbol{\xi}} \right)$ for a planar quadratic triangle is a polynomial in ξ and η of order 2 as well.

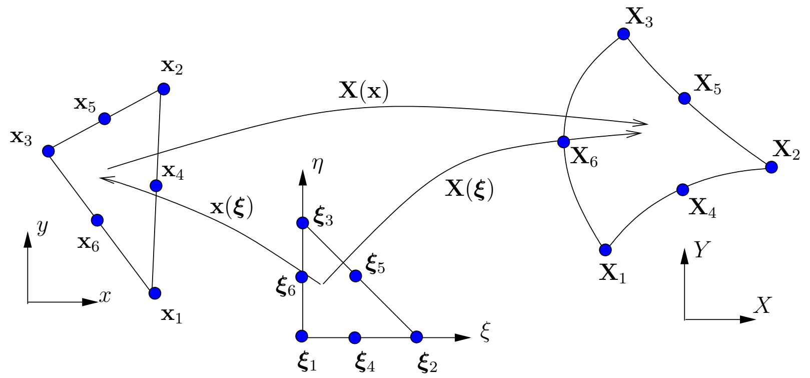


Figure 1.10: Reference unit triangle in local coordinates $\boldsymbol{\xi} = (\xi, \eta)$ and the mappings $\mathbf{x}(\boldsymbol{\xi})$, $\mathbf{X}(\boldsymbol{\xi})$ and $\mathbf{X}(\mathbf{x})$.

If J_i is defined as $J(\xi, \eta)$ evaluated at node i, it is possible to write the Jacobian determinant exactly as a finite element expansion whose coefficients are the Jacobian determinants at the nodes:

$$\begin{aligned} J(\xi,\eta) \quad = \quad & J_1 \underbrace{(1-\xi-\eta)(1-2\xi-2\eta)}_{\mathcal{L}_1^{(2)}(\xi,\eta)} + J_2 \underbrace{\xi(2\xi-1)}_{\mathcal{L}_2^{(2)}(\xi,\eta)} + J_3 \underbrace{\eta(2\eta-1)}_{\mathcal{L}_3^{(2)}(\xi,\eta)} + \\ & J_4 \underbrace{4(1-\xi-\eta)\xi}_{\mathcal{L}_4^{(2)}(\xi,\eta)} + J_5 \underbrace{4\xi\eta}_{\mathcal{L}_5^{(2)}(\xi,\eta)} + J_6 \underbrace{4(1-\xi-\eta)\eta}_{\mathcal{L}_6^{(2)}(\xi,\eta)} . \end{aligned} \tag{1.1}$$

In equation (1.1), the functions $\mathcal{L}_i^{(2)}(\xi, \eta)$ are the equidistant quadratic Lagrange shape functions that are commonly used in the finite element community.

It is obvious that a necessary condition for having $J(\xi, \eta) > 0$ everywhere is that $J_i > 0$, $i = 1, \ldots, 6$. Yet, this condition is not sufficient. The expression (1.1) does not give more information because the quadratic Lagrange shape functions $\mathcal{L}_i^{(2)}(\xi, \eta)$ change sign on the reference triangle. The main idea of Johnen et al. (2013) is to use the quadratic triangular Bézier functions $\mathcal{B}_2^{(2)}(\xi,\eta)$ Piegl and Tiller (1997):

$$\begin{aligned} J(\xi,\eta) &= J_1 \underbrace{(1-\xi-\eta)^2}_{\mathcal{B}_1^{(2)}(\xi,\eta)} + J_2 \underbrace{\xi^2}_{\mathcal{B}_2^{(2)}(\xi,\eta)} + J_3 \underbrace{\eta^2}_{\mathcal{B}_3^{(2)}(\xi,\eta)} \\ &+ \left(2J_4 - \frac{1}{2}(J_2+J_1)\right) \underbrace{2\,\xi\,(1-\xi-\eta)}_{\mathcal{B}_4^{(2)}(\xi,\eta)} \\ &+ \left(2J_5 - \frac{1}{2}(J_3+J_2)\right) \underbrace{2\,\xi\eta}_{\mathcal{B}_5^{(2)}(\xi,\eta)} \\ &+ \left(2J_6 - \frac{1}{2}(J_1+J_3)\right) \underbrace{2\,\eta\,(1-\xi-\eta)}_{\mathcal{B}_6^{(2)}(\xi,\eta)} . \end{aligned} \tag{1.2}$$

Since $\sum_{i=1}^{6} \mathcal{B}_i^{(2)}(\xi,\eta) = 1$ and $\mathcal{B}_i^{(2)}(\xi,\eta) \geq 0$, we obtain the following estimate

$$\begin{aligned} J_{\min} &\geq \min\left\{J_1,\ J_2,\ J_3,\ 2J_4 - \frac{J_1+J_2}{2},\ 2J_5 - \frac{J_2+J_3}{2},\ 2J_6 - \frac{J_3+J_1}{2}\right\} \\ &\leq \min\left\{J_1,\ J_2,\ J_3\right\}. \end{aligned} \tag{1.3}$$

This estimate provides two conditions on the geometrical validity of the triangle: a *sufficient* condition (if $\min\{J_1, J_2, J_3, 2J_4 - \frac{J_1+J_2}{2}, 2J_5 - \frac{J_2+J_3}{2}, 2J_6 - \frac{J_3+J_1}{2}\} > 0$, the element is valid) and a *necessary* condition (if $\min\{J_1, J_2, J_3\} < 0$, the element is invalid). However, these two conditions are sometimes insufficient to determine the validity of the element, as the bound (1.3) is often not sharp enough (having $\min\{2J_4 - \frac{J_1+J_2}{2}, 2J_5 - \frac{J_2+J_3}{2}, 2J_6 - \frac{J_3+J_1}{2}\} < 0$ does not imply that the element is invalid).

A sharp necessary and sufficient condition on the geometrical validity of an element can be achieved in a general way by refining the Bézier estimate adaptively so as to achieve any prescribed tolerance—and thus provide bounds as sharp as necessary for a given application. In our work, we will use the sufficient condition (1.3) to assess the validity of our adaptive curvilinear triangles.

4 | Classical theory of H-adaptivity

A mesh should be adapted. Adaptivity consists in finding the right mesh size and order at the right place. In this section, we cover the classical theory of H-adaptivity. H-adaptivity consists in changing the mesh size while leaving the approximation order constant. There are two steps in H-adaptivity: i) approximating the error and ii) computing the optimal mesh. We also distinguish isotropic and anisotropic H-adaptivity. This section covers both cases.

4.1 Mesh adaptation – the isotropic case

A mesh $\mathcal{M}$ is a geometrical discretization of a domain Ω that consists of a collection of mesh entities $\Omega_i, \quad i = 1, \dots, N$ (triangles, quadrangles, tetrahedra, hexahedra, etc.) of controlled shape and size.

Let us consider a field $f(x, y)$ defined in domain Ω and a finite element approximation f_h of f defined on $\mathcal{M}$.

The quality of a finite element solution f_h depends strongly on its underlying mesh. A mesh is optimum when it covers the domain strategically: it is dense where the solution exhibits strong variations and coarse in places where the discretization error is low. h-adaptivity consists in controlling the mesh size in order to control the discretization error.

Let us define the elementary discretization error as some norm $\|.\|$ of the difference between the finite element solution f_h and the exact solution f

$$e_i^2 = \int_{\Omega_i} \|f - f_h\|^2 \, dv \tag{1.4}$$

A posteriori error estimation techniques aim at producing estimates of e_i Ainsworth and Oden (2011). The local error converges to zero at a certain convergence rate k

$$e_i = C h_i^k \tag{1.5}$$

where C is a positive constant depends on f but is independent of h_i – the *local mesh size*. The size h_i of a triangle or of a tetrahedron Ω_i is usually chosen as its circumradius. Let us call $\mathcal{M}^*$ the optimal mesh and h_i^* the optimal mesh size in the area defined by element Ω_i in the original mesh, $\mathcal{M}$. This defines a size field on Ω: for each element Ω_i of the mesh $\mathcal{M}$ (that actually covers Ω), an optimal size h_i^* is defined. In general, the mesh adaptation procedure requires as an input a size field $h(\boldsymbol{x})$ that returns the optimal size of the mesh at any point $\boldsymbol{x}$. A background mesh is a special case of a size field.

The total error contained in the optimal mesh over the area defined by Ω_i is

$$e_i^{*2} = e_i^2 \left(\frac{h_i^*}{h_i}\right)^{2k}. \tag{1.6}$$

The total error contained in the optimal mesh is therefore

$$e^{*2} = \sum_{i=1}^{N} e_i^{*2} = \sum_i e_i^2 \left(\frac{h_i^*}{h_i}\right)^{2k} = \sum_i e_i^2 r_i^{-2k} \tag{1.7}$$

where r_i is the size reduction factor of element Ω_i. The total number of elements in the optimal mesh can be written as

$$N^* = \sum_{i=1}^{N} \left(\frac{h_i}{h_i^*}\right)^d = \sum_{i=1}^{N} r_i^d. \tag{1.8}$$

where d is the dimension of the problem.

With the above statements, it is possible to give two definitions of what could be called *an optimal mesh*. An optimal mesh could be defined as a mesh that results in a specified discretization error $e^* = \bar{e}$ while minimizing the number of elements N^*. The solution of this problem can be written in closed form Ladevèze et al. (1991), posing $\alpha = 2k/d$, as

$$r_i = \left(\frac{e_i^{\frac{2\alpha}{(1+\alpha)}} \left(\sum_{j=1}^{N} e_j^{\frac{2}{1+\alpha}} \right)}{\bar{e}^2} \right)^{\frac{1}{d\alpha}} = K e_i^{\frac{2}{2k+d}} \tag{1.9}$$

with K independent of i. The error in one element of the optimal mesh is

$$\frac{e^{*2}_i}{r_i^d} = e_i^2 r_i^{-2k-d} = e_i^2 e_i^{2(-2k-d)/(2k+d)} K^{-2k-d} = K^{-2k-d}. \tag{1.10}$$

The mesh optimization process aims therefore at building a mesh with errors that are uniformly distributed.

Another definition of mesh optimality would be to impose the number of elements in the optimal mesh $N^* = \bar{N}$ while minimizing the discretization error e^{*2}. The optimal formula

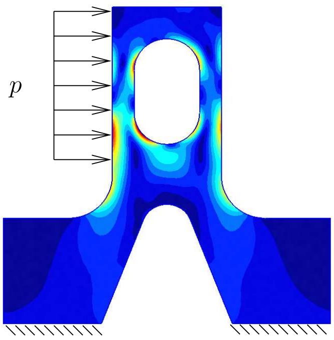
$$r_i = \left(\frac{e_i^{\frac{2}{(1+\alpha)}}}{\left(\sum_{j=1}^{N} e_j^{\frac{2}{1+\alpha}} \right)} \bar{N} \right)^{\frac{1}{d}} \tag{1.11}$$

is very similar to (1.9) and also leads to uniform error repartition.

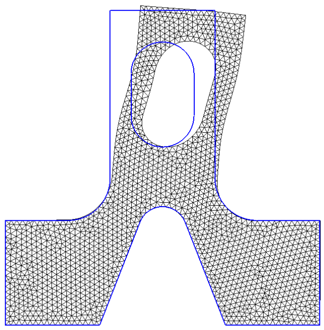
Note that factor k may be variable: this is true in the case of p-adaptivity where p is the polynomial degree of the element or when singularities are present in the solution. When this is the case, no closed form formula like (1.9) is available and numerical optimization has to be used to find the optima.

As an example, let us consider the following problem briefly depicted in Figure 1.11-(a). Here, the equation of plane strain elasticity has been solved using linear finite elements. The error norm that has been chosen is the energy norm. The theoretical convergence rate of the error in energy is $k = 1$. The simple error procedure of Zienkiewicz and Zhu (1992a,b) has been used for estimating the error.

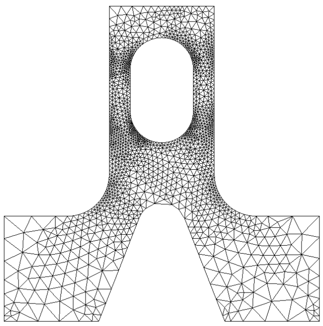
Starting with a mesh of $3,301$ triangles, formula (1.11) was used with $\bar{N} = 3,301$ i.e. we aim at producing a mesh with the same number of elements that produces a minimum amount of discretization error. It took two mesh adaptation iterations to produce a mesh with $3,230$ elements and with a discretization error of $e^2 =$



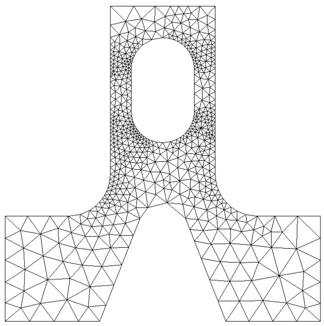
(a) Von-Mises stresses, loads and fixations



(b) Initial mesh ($3,301$ triangles) deformed structure



(c) Optimal mesh using of formula (1.11)
$\bar{N} = 3,301$ triangles, $N = 3,230$ triangles



(d) Optimal mesh using of formula (1.9)
$\bar{e}^2 = 5.66\ 10^7$, $N = 1,090$ triangles

Figure 1.11: An example of the use of Equations (1.9) and (1.11).

$2.37\ 10^7$ (see Figure 1.11). The initial discretization error with 3301 elements was $e^2 = 5.66\ 10^7$.

Starting with the same mesh of $3,301$ triangles, formula (1.9) was used with $\bar{e}^2 = 5.66\ 10^7$ i.e. we aim now at producing a mesh with about the same discretization error but with a minimum number of elements. Again, it took two adaptation iterations to produce a mesh with $1,090$ elements and with a discretization error of $e^{*2} = 5.90\ 10^7$ (see Figure 1.11).

4.2 Error estimation – the isotropic case

In (1.4), we assume that we have access to the exact solution f of a problem. This is obviously not the case. Even though elementary errors e_i are not computable, it is indeed possible to *estimate* e_i. There are various ways of estimating ***a posteriori*** (i.e. after having computed f_h) the error of finite element solutions Ainsworth and Oden (2011). In this thesis, we only focus on approximation error i.e. the difference between f and its Lagrange interpolation $\Pi_h^k f$ Ainsworth and Oden (2011). In short, $\Pi_h^k f$ belongs to the same finite element space as f_h. It consists in choosing as nodal values the exact values of the exact function f. The Lagrange interpolation is of outmost importance in error estimation Kobayashi and Tsuchiya (2014) because it is often true that

$$\|f - f_h\| \leq C\|f - \Pi_h^k f\|.$$

where C is a positive constant. Thus, even though $\|f - f_h\|$ cannot be computed, standard approximation theory based on Taylor series allows us to estimate the interpolation error $\|f - \Pi_h^k f\|$, leading to the possibility to replace the true error $\|f - f_h\|$ by an error estimator $\|f - \Pi_h^k f\|$.

Let $W^{l,p}(\Omega)$, $l \in N$, $p \in [1, \infty]$, be the Sobolev spaces with the seminorms

$$|f|^p_{W^{l,p}(\Omega)} := \sum_{|\alpha|=l} \int_\Omega |D^\alpha f|^p.$$

The case $p = 2$ refers to standard Sobolev spaces: $H^m(\Omega) = W^{m,2}(\Omega)$ and $H^0(\Omega) = L^2(\Omega)$. Consider a triangle T. Classical estimate of the interpolation error reads

$$|f - \Pi_h^k f|_{H^m(T)} \leq C(\text{diam}(T))^{l-m}|f|_{H^l(T)},$$

$\text{diam}(T)$ being the longest edge of T. The admissible range of the parameters l and m depend on the space dimension d and on the polynomial degree k.

In this work, $d = 2$ and we essentially work with polynomial orders $k = 2$. Thus, the L^2 ($m = 0$) and H^1 ($m = 1$) norms of the error can be estimated by approximating the third order derivatives of f inside T:

$$\|f - \Pi_h^1 f\|_{L^2(T)} \leq C(\text{diam}(T))^3|f|_{H^3(T)},$$

$$|f - \Pi_h^1 f|_{H^1(T)} \leq C(\text{diam}(T))^2|f|_{H^3(T)}.$$

In this work, we start by creating optimal meshes based on a known analytical function $f(x, y)$. In this case, we have at hand the exact derivatives of f at any order. We will thus compute the error estimates based on the exact derivatives.

In Chapter 4, we use finite element solutions and do a posteriori error estimation. We thus have to approximate third order derivatives of f based on f_h. This is indeed quite classical: Zienkiewicz and Zhu have based their very first error estimator based on a reconstruction of the mechanical stresses σ Zienkiewicz and Zhu (1992a,b).

There are indeed many ways of approximating $D^3 f$ based on a given P^2 finite element solution f_h: one can compute continuous nodal gradients using least squares like Zienkiewicz and Zhu (1992a,b), then compute nodal Hessians based on the continuous gradients using the same procedure, and so forth until the required high order derivatives are computed.

4.3 Mesh adaptation – the anisotropic case

For the problems with anisotropic features, anisotropic mesh adaptation is a tool that allows to build optimal meshes by adapting not only the mesh size but also its orientation and its anisotropy. The classical anisotropic adaptation setting produces straight sided anisotropic meshes. Even though we are willing to produce anisotropic meshes that are curvilinear, it is worth to define the key ingredients of classical anisotropic mesh adaptation because we will use them as well in the curvilinear case.

4.3.1 Metric field

We consider the 2D space $\mathbb{R}^2$ and a *smooth metric field* $\boldsymbol{g}(\mathbf{x})$ that associates to every point $\mathbf{x}$ of the plane a 2×2 symmetric positive-definite matrix. If $\boldsymbol{g}(\mathbf{x})$ is differentiable, it is called a Riemannian metric. Metric field $\boldsymbol{g}$ allows to define an inner product that modifies the standard definition of length, angles and areas. The main idea of using metric fields for anisotropic mesh adaptation comes from Hecht et al. (1997). We are going to define $\boldsymbol{g}$ in such a way that an optimal mesh with respect to the new inner product will be a unit mesh i.e. a mesh with all its edges of length one (length is measured with respect to $\boldsymbol{g}$).

Consider two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. The inner product between $\boldsymbol{u}$ and $\boldsymbol{v}$ associated with $\boldsymbol{g}$ at a point $\mathbf{x}$ is defined as:

$$\boldsymbol{g}_{\mathbf{x}}(\boldsymbol{u},\boldsymbol{v}) = \langle \boldsymbol{u},\boldsymbol{v}\rangle_{\boldsymbol{g}(\mathbf{x})} = \langle \boldsymbol{u}, \boldsymbol{g}(\mathbf{x})\boldsymbol{v}\rangle = \boldsymbol{u}^T \boldsymbol{g}(\mathbf{x})\, \boldsymbol{v}.$$

If $\boldsymbol{g}$ is symmetric definite positive, then this inner product has the right properties of symmetry and positive:

1. $\forall\, (\boldsymbol{u},\boldsymbol{v}) \in V \times V, \boldsymbol{g}_{\mathbf{x}}(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{g}_{\mathbf{x}}(\boldsymbol{v},\boldsymbol{u})$
2. $\forall \boldsymbol{u} \in V, \boldsymbol{g}_{\mathbf{x}}(\boldsymbol{u},\boldsymbol{u}) \geq 0$
3. $\forall \boldsymbol{u} \in V, \boldsymbol{g}_{\mathbf{x}}(\boldsymbol{u},\boldsymbol{u}) = 0 \Rightarrow \boldsymbol{u} = 0$

Note that isotropic mesh generation algorithms actually work in the canonical Euclidean space which can be seen as the particular case of the Riemannian space for which the metric tensor field is the identity matrix scaled by a constant factor.

4.3.2 Metric-based mesh adaptation

In the standard Euclidean inner product space, the scalar product of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is defined as:

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{v}, \boldsymbol{u} \rangle = \boldsymbol{u}^T \boldsymbol{v}$$

and the length of a vector $\boldsymbol{u}$ is defined as:

$$\|\boldsymbol{u}\| = \sqrt{\langle \boldsymbol{u}, \boldsymbol{u} \rangle} = \sqrt{\boldsymbol{u}^T \boldsymbol{u}}$$

and the surface of the parallelogram defined by $\boldsymbol{u}$ and $\boldsymbol{v}$ is computed as:

$$\frac{1}{2}\|\boldsymbol{u} \times \boldsymbol{v}\|.$$

The metric field provides a different way to measure the length of a vector or the angle between two vectors. In a Riemannian metric space $(\mathbb{R}^2, \boldsymbol{g}(\mathrm{x}))$, the scalar product of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ with respect to the metric field $\boldsymbol{g}(\mathrm{x})$ is defined as:

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\boldsymbol{g}(\mathrm{x})} = \langle \boldsymbol{u}, \boldsymbol{g}(\mathrm{x})\boldsymbol{v} \rangle = \boldsymbol{u}^T \boldsymbol{g}(\mathrm{x})\boldsymbol{v}.$$

The length of a vector $\boldsymbol{u}$ with respect to the constant metric field $\boldsymbol{g}(\mathrm{x})$ is defined as:

$$\mathcal{L}_{\boldsymbol{g}(\mathrm{x})}(\boldsymbol{u}) = \|\boldsymbol{u}\|_{\boldsymbol{g}(\mathrm{x})} = \sqrt{\langle \boldsymbol{u}, \boldsymbol{u} \rangle_{\boldsymbol{g}(\mathrm{x})}} = \sqrt{\boldsymbol{u}^T \boldsymbol{g}(\mathrm{x})\boldsymbol{u}}$$

Now consider a curve C and its smooth parametrization $\mathrm{x}(t),\ t \in [0,1]$. Then the length with respect to the metric field of C is given by:

$$\mathcal{L}_{\boldsymbol{g}(\mathrm{x})}(C) = \int_0^1 \sqrt{\dot{\mathrm{x}}^T \boldsymbol{g}(\mathrm{x})\dot{\mathrm{x}}}\ dt$$

where $\dot{\mathrm{x}} = \frac{\mathrm{dx}}{\mathrm{d}t}$. Similarly, consider a surface S with its parametrization $\mathrm{x}(\xi, \eta)$, $(\xi, \eta) \in [0,1] \times [0,1]$. Then, the area of S with respect to the metric can be computed as

$$\mathcal{A}_{\boldsymbol{g}(\mathrm{x})}(S) = \int_0^1 \int_0^1 \det \left(J^T \boldsymbol{g}(\mathrm{x}) J\right)\ d\xi d\eta$$

where J is the Jacobian of the parametrization i.e. a 2×2 matrix which two columns are $\dfrac{\partial \mathbf{x}}{\partial \xi}$ and $\dfrac{\partial \mathbf{x}}{\partial \eta}$.

It is then possible to extend the notion of angle within this context. The angle θ between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ in metric $\boldsymbol{g}(\mathrm{x})$ is given by

$$\theta_{\boldsymbol{g}(\mathrm{x})} = \arccos\left(\frac{\boldsymbol{u}^t\ \boldsymbol{g}(\mathrm{x})\ \boldsymbol{v}}{\|\boldsymbol{u}\|_{\boldsymbol{g}(\mathrm{x})}\|\boldsymbol{v}\|_{\boldsymbol{g}(\mathrm{x})}}\right).$$

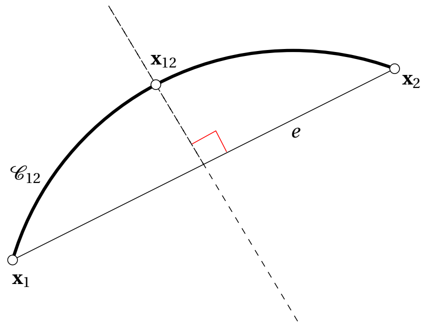


Figure 1.12: A smooth curve that goes from $\mathbf{x}_1$ to $\mathbf{x}_2$

The definition of $\boldsymbol{g}(\mathbf{x})$ allows to define the notion of geodesic which is the shortest path between two points Gu and Yau (2008). The distance between two points $\mathbf{x}_1$ and $\mathbf{x}_2$ with respect to the metric field $\boldsymbol{g}(\mathbf{x})$ is defined as

$$\mathcal{D}_v(\mathbf{x}_1, \mathbf{x}_2) = \min_C \mathcal{L}_{\boldsymbol{g}(\mathbf{x})}(C)$$

where C is any smooth curve that goes from $\mathbf{x}_1$ to $\mathbf{x}_2$ (see Figure 1.12).

In this thesis, we consider meshes and paths between vertices that are mesh edges. As said before, we restrict our work to quadratic meshes so mesh edges are at most parabolas. Thus, for a quadratic mesh, the geodesic between two points $\mathbf{x}_1$ and $\mathbf{x}_2$ it is restricted to the shortest parabola, i.e. the parabola that has the smallest length with respect to the metric field $\boldsymbol{g}(\mathbf{x})$. We will see in further chapters that only a small subset of all possible parabolas will be considered in order to be able to compute geodesic parabolas in reasonable time.

Metric based mesh adaptation consists in computing a metric field $\boldsymbol{g}(\mathbf{x})$ that makes lengths $\mathcal{L}_{\boldsymbol{g}(\mathbf{x})}$ adimensional. The main idea is quite simple: eigenvectors of $\boldsymbol{g}(\mathbf{x})$ are two orthogonal unit vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ and eigenvalues λ_1 and λ_2 of $\boldsymbol{g}(\mathbf{x})$ with respect to the two eigenvectors actually depend on the mesh size h_1 and h_2 on the two directions:

$$\lambda_1 = \frac{1}{h_1^2} \quad , \quad \lambda_2 = \frac{1}{h_2^2}.$$

A vector $\boldsymbol{u}$ with an adimensional length of

$$\mathcal{L}^2_{\boldsymbol{g}(\mathbf{x})}(\boldsymbol{u}) = \boldsymbol{u}^T \begin{pmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1^T \\ \boldsymbol{v}_2^T \end{pmatrix} \boldsymbol{u} = \lambda_1 \left(\boldsymbol{u}^T \boldsymbol{v}_1\right)^2 + \lambda_2 \left(\boldsymbol{u}^T \boldsymbol{v}_2\right)^2.$$

Thus, a vector $\boldsymbol{u} = h_1 \boldsymbol{v}_1$ has an adimensional length of

$$\mathcal{L}^2_{\boldsymbol{g}(\mathbf{x})}(h_1 \boldsymbol{v}_1) = 1.$$

In this thesis, we aim at generating a curved mesh whose edges are of adimensional length 1 in a given metric field. Such a mesh is usually called a unit mesh. Of course, a mesh being a discrete object, we will relax the constraints on this very strong assessment. A tolerance will be applied on edge lengths: an edge is acceptable in a unit mesh if its adimensional length is in an interval

$$\mathcal{L}_{\min} \leq \mathcal{L}_{\boldsymbol{g}(\mathbf{x})}(\boldsymbol{u}) \leq \mathcal{L}_{\max}$$

where $\mathcal{L}_{\max}/2 \geq \mathcal{L}_{\min}$ and $\mathcal{L}_{\min} \leq 1 \leq \mathcal{L}_{\max}$. We usually choose

$$0.7 \leq \mathcal{L}_{\boldsymbol{g}(\mathbf{x})}(\boldsymbol{u}) \leq 1.4.$$

We finally define the unit circle in a Riemannian metric space as the location of every point that is at distance 1 from a center $\mathbf{x}_0$:

$$U_{\boldsymbol{g}(\mathbf{x})}(\mathbf{x}_0) = \left\{ \mathbf{x} \in \mathbb{R}^2 \,|\, \mathcal{D}_{\boldsymbol{g}(\mathbf{x})}(\mathbf{x}_0, \mathbf{x}) = 1 \right\}.$$

For a constant metric field, the unit circles are ellipses. On the contrary, for general metric field, the unit circles can take many different shapes and may not be *visually* convex. In reality, the unit circle may remain convex in the sense that every geodesic between two points that line inside the circle remains inside the circle. Yet, there is no strict guarantee of convexity for general Riemannian metrics. Figure 1.13 shows such unit circles for a non-constant toy metric field Equation 1.12.

$$\boldsymbol{g}(x^1, x^2) = \begin{pmatrix} g_{11} & g_{12} \\ g_{12} & g_{22} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \frac{1}{l_{\min}^2} & 0 \\ 0 & \frac{1}{l_{\max}^2} \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

with

$$\mathbf{x} = \{x^1, x^2\}, \quad r = \|\mathbf{x}\|, \quad \theta = \arctan(x^2/x^1),$$
$$l_{\min} = \epsilon + l_{\max}(1 - \exp(-((r - r_0)/h)^2).$$

Let $P_0 = (x_0, y_0, z_0)$ be a point on a surface S, and let C be any curve passing through P_0 and lying entirely in S. At P_0, the tangent plane (Figure 1.14) to S is the plane that the tangent lines to all curves C at P_0 lie in. The tangent plane assumption that is made in all publications up to now is used as an linear approximation to the function $z = f(x, y)$ with continuous partial derivatives that exist at point (x_0, y_0):

$$f(x, y) = f(x_0, y_0) + f_x(x_0, y_0)(x - x_0) + f_y(x_0, y_0)(y - y_0).$$

This leads to unit circles that are ellipsis and where geodesic remain straight lines Borouchaki et al. (1997a); Hecht et al. (1997); Bottasso (2004); Huang (2005); Gruau and Coupez (2005); Alauzet et al. (2006); Huang et al. (2010); Coupez (2011); Huang et al. (2013); Marcum and Alauzet (2013); Coulaud and Loseille (2016). On Figure 1.15,

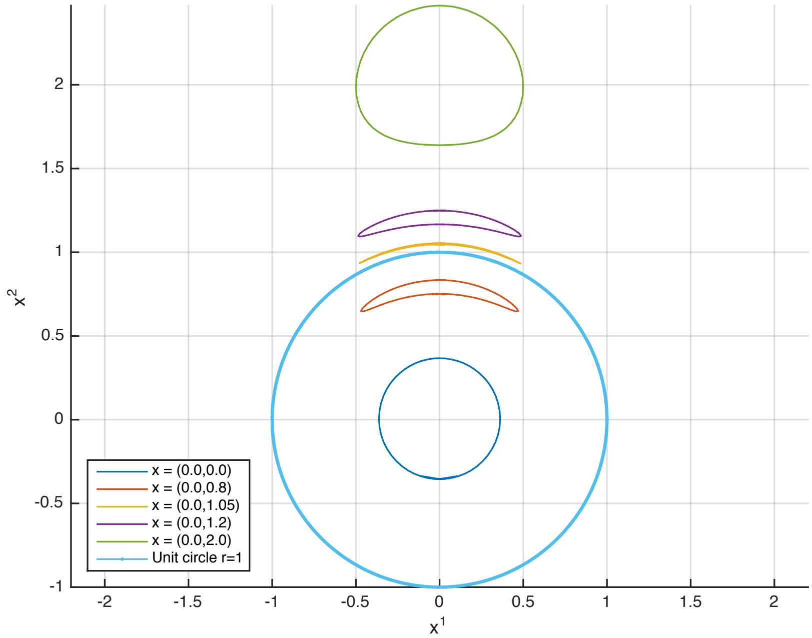


Figure 1.13: Unit circles at different centers for the toy metric - a given metric field (Equation 1.12) that allows to compute unit distances.

unit circles corresponding to the principal directions of the metric at point $(0, 1.2)$ are drawn, both for true geodesics and in the case of the tangent plane approximation.

The main advantage of this approach is that the adaptation mesh is obtained directly at the mesh generation step rather than doing a modification of an existing mesh by enrichment or moving points. This statement does not refer specifically to the metric-based approach. The approach consists in three main steps, namely:

1. the definition of a suitable anisotropic interpolation error estimate;
2. the construction of an optimal metric field;
3. the mesh generation and adaptation with respect to the optimal metric field.

4.4 Error estimation – the anisotropic case

Unstructured mesh adaptation has largely proved its efficiency for capturing the behavior of physical phenomena (even if it is not known a priori), improving the accuracy of the numerical solution and reducing the number of degrees of freedom.

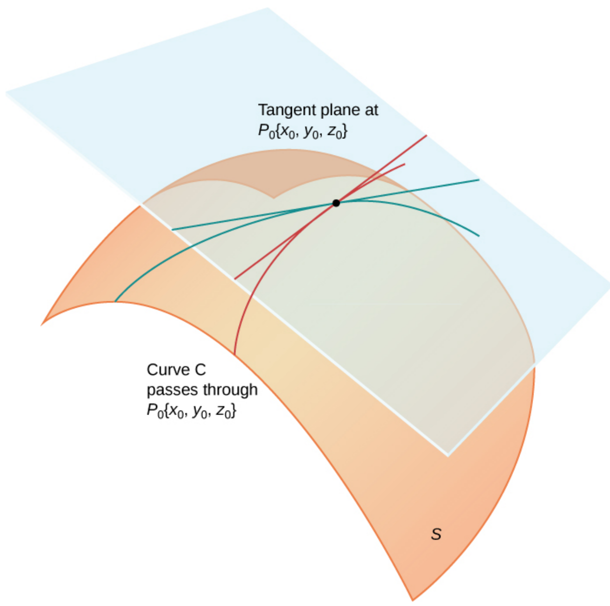


Figure 1.14: The definition of tangent plane Gilbert Strang (Mar 30, 2016)

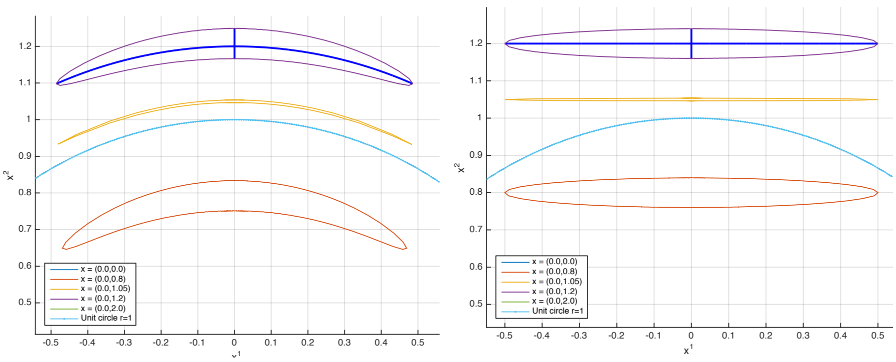


Figure 1.15: Unit circles at different centers for the toy metric - a given metric field (Equation 1.12) that allows to compute unit distances. Left Figure shows circles computed using the exact geodesics while right Figure assumes a constant metric (tangent plane approximation)

The aim of an anisotropic mesh adaptation is still to equidistribute the error over the mesh, and there is an a posteriori correlation between the local mesh element and the error on the numerical solution. The definition of a general purpose error estimate allows to construct an anisotropic metric field that can be used to create an anisotropic mesh.

4.4.1 Error estimations

Let $W^{l,p}(\Omega)$, $l, m \in N$, $p, q \in [1, \infty]$. $h_{1,T}$ and $h_{2,T}$ is the element length in the x and y direction, respectively. $\Pi_h^1 f$ is the Lagrangian linear interpolation operator. The anisotropic error estimate is Apel (1999a)

$$|f - \Pi_h^1 f|_{W^{m,p}(T)} \leq C \sum_{\alpha_1,\alpha_2 \geq 0}^{\alpha_1+\alpha_2=l-m} h_{1,T}^{\alpha_1} h_{2,T}^{\alpha_2} |\frac{\partial^{l-m} f}{\partial x^{\alpha_1} \partial y^{\alpha_1}}|_{W^{m,q}(T)}. \tag{1.12}$$

We define a invertible affine map from the reference triangle $\hat{T}$ to the element T as:

$$\mathbf{x} = \mathcal{T}_T(\hat{\mathbf{x}}) = A\hat{\mathbf{x}} + \boldsymbol{b},$$

where A is a matrix and $\boldsymbol{b}$ is a vector.

Using the polar decomposition and the singular value decomposition, we have:

$$A = (R^T \Lambda R)(R^T P),$$

with $R = [\boldsymbol{r}_{1,T}\ \boldsymbol{r}_{2,T}]^T$ is the orthonormal matrix and its rows are the eigenvectors of $R^T \Lambda R)$, and $\Lambda = diag(\lambda_{1,T}, \lambda_{2,T})$ is the diagonal matrix of the corresponding eigenvalues with $\lambda_{1,T} \geq \lambda_{2,T}$.

The anisotropic nature of Equation 1.12 is represented by having preserved separately the two directions $\boldsymbol{r}_{1,T}$ and $\boldsymbol{r}_{2,T}$ and the corresponding length-scales $\lambda_{1,T}$ and $\lambda_{2,T}$. We can state the results Formaggia and Perotto (2001) as:

For any $f \in H^2(T)$, let $\hat{f} \in H^2(\hat{T})$ be the corresponding function defined on the reference element $\hat{T}$, the following estimate holds:

$$\begin{aligned} |\hat{f}|_{H^2(\hat{T})} \leq \Big[& \frac{\lambda_{1,T}^3}{\lambda_{2,T}} (|\boldsymbol{r}_{1,T}^T|Q|\boldsymbol{r}_{1,T}|)^2 + \frac{\lambda_{2,T}^3}{\lambda_{1,T}} (|\boldsymbol{r}_{2,T}^T|Q|\boldsymbol{r}_{2,T}|)^2 \\ & + 2\lambda_{1,T}\lambda_{2,T} (|\boldsymbol{r}_{1,T}^T|Q|\boldsymbol{r}_{2,T}|)^2 \Big]^{\frac{1}{2}}, \end{aligned}$$

with Q being the symmetric matrix as:

$$Q = \begin{pmatrix} \| f_{xx} \|_{L^2(T)} & \| f_{xy} \|_{L^2(T)} \\ \| f_{xy} \|_{L^2(T)} & \| f_{yy} \|_{L^2(T)} \end{pmatrix}.$$

Let $f \in H^1(T)$ and $\hat{f} \in H^1(\hat{T})$ be the corresponding function defined on the reference element $\hat{T}$, the following inequalities holds:

$$\left(\frac{\lambda_{2,T}}{\lambda_{1,T}}\right)^{\frac{1}{2}} |\hat{f}|_{H^1(\hat{T})} \leq |f|_{H^1(T)} \leq \left(\frac{\lambda_{1,T}}{\lambda_{2,T}}\right)^{\frac{1}{2}} |\hat{f}|_{H^1(\hat{T})}.$$

There exists a constant $C = C(\hat{T}, \Pi_T^1)$ such that the following estimate holds:

$$\begin{aligned}\|f - \Pi_T^1 f\|_{L^2(T)} \leq C\big[&\lambda_{1,T}^4(|\boldsymbol{r}_{1,T}^T|Q|\boldsymbol{r}_{1,T}|)^2 + \lambda_{2,T}^4(|\boldsymbol{r}_{2,T}^T|Q|\boldsymbol{r}_{2,T}|)^2 \\ &+ 2\lambda_{1,T}^2\lambda_{2,T}^2(|\boldsymbol{r}_{1,T}^T|Q|\boldsymbol{r}_{2,T}|)^2\big]^{\frac{1}{2}},\end{aligned}$$

$$\begin{aligned}|f - \Pi_T^1 f|_{H^1(T)} \leq C\left(\frac{\lambda_{1,T}}{\lambda_{2,T}}\right)^{\frac{1}{2}} \big[&\frac{\lambda_{1,T}^3}{\lambda_{2,T}}(|\boldsymbol{r}_{1,T}^T|Q|\boldsymbol{r}_{1,T}|)^2 + \frac{\lambda_{2,T}^3}{\lambda_{1,T}}(|\boldsymbol{r}_{2,T}^T|Q|\boldsymbol{r}_{2,T}|)^2 \\ &+ 2\lambda_{1,T}\lambda_{2,T}(|\boldsymbol{r}_{1,T}^T|Q|\boldsymbol{r}_{2,T}|)^2\big]^{\frac{1}{2}}.\end{aligned}$$

Let $\mathcal{Q}_T^1$ be the Clement interpolant Clément (1975) or the Scott-Zhang operator Apel (1999b); Scott and Zhang (1990), the following estimates holds:

$$\|f - \mathcal{Q}_T^1 f\|_{L^2(T)} \leq C\left[\lambda_{1,T}^2 \boldsymbol{r}_{1,T}^T G_{\boldsymbol{r}_{1,T}} + \lambda_{2,T}^2 \boldsymbol{r}_{2,T}^T G_{\boldsymbol{r}_{2,T}}\right]^{\frac{1}{2}},$$

with G being the symmetric non-negative matrix as:

$$G = \sum_{T \in \mathcal{P}_T} \begin{pmatrix} \| f_x \|_{L^2(T)} & \int_T f_x f_y \, d\mathbf{x} \\ \int_T f_x f_y \, d\mathbf{x} & \| f_y \|_{L^2(T)} \end{pmatrix}.$$

4.4.2 Optimal metric field

This problem of mesh adaptation is, for a given function, to seek for the optimal continuous mesh minimizing the interpolation error or meet a given interpolation error. Thus, in a general framework of unstructured anisotropic mesh adaptation, one of the most important problem is to reduce the number of freedom degrees while preserving the desired level of accuracy for the numerical solution.

The aims is to create the optimal mesh $\mathcal{M}^*$, i.e., the optimal continuous metric $\boldsymbol{g}^*(\mathbf{x})$, to minimize the interpolation error e in L^p norm. We formulate the problem in two related forms :

1. Given a total number $\bar{N}$, to minimize the interpolation error e in L^p norm.
2. Given a interpolation error $\bar{e} > 0$ in L^p norm, to minimize the number of elements N.

Based on and taking advantage of well-defined Riemannian differential geometry, metric-based mesh generation and adaptation is to generate a quasi-uniform mesh - specifying the shape, size, orientation of elements with respect to a metric filed and all geometric operations are performed in the Riemannian metric space. Thus it is crucial to construct an appropriate metric field in the Riemannian metric space. Considering the optimal metric is to minimizes the interpolation error e, we need to draw out the relationship between the optimal metric and the interpolation error. We formulate the problem in two related forms :

1. Given a total number $\bar{N}$, to process to the metric analysis: 1) derive optimal stretching directions; 2) derive optimal sizes.
2. Given a interpolation error $\bar{e} > 0$, to process to the metric analysis: 1) derive optimal stretching directions; 2) derive optimal sizes.

Mathematically, we need to solve the following minimization problem:

$$\text{find } \boldsymbol{g}^*(\mathbf{x}) \qquad \text{such that } min \int_\Omega \| e^p_{\boldsymbol{g}^*(\mathbf{x})}(\mathbf{x}) \| d\mathbf{x}.$$

Let $\mathbf{x}_0$ be a point of domain Ω and $\mathbf{B}_{\boldsymbol{g}(\mathbf{x})}(\mathbf{x}_0) = \{\mathbf{x} \in \Omega \mid d_{\boldsymbol{g}(\mathbf{x})}(\mathbf{x}, \mathbf{x}_0) \leq 1\}$ be its unit ball. Geometrically Mbinky et al. (2012), the local optimal metric $\boldsymbol{g}^*(\mathbf{x})$ is to find the maximal unit ball in the isoline 1 of $|e(\mathbf{x})|$ (see Figure 1.16), i.e. in the vicinity of $\mathbf{x}_0$, the local error $e_{\boldsymbol{g}(\mathbf{x})}$ is:

$$e_{\boldsymbol{g}(\mathbf{x})}(\mathbf{x}_0) = max_{\mathbf{x} \in \mathbf{B}_{\boldsymbol{g}(\mathbf{x})}(\mathbf{x}_0)} \mid f(\mathbf{x}) - \Pi_h f(\mathbf{x}) \mid .$$

Concerning linear $k = 1$ interpolation, a lot of works Nadler (1985); D'Azevedo and Simpson (1991); Rippa (1992); Simpson (1994); Cao (2005); Alauzet et al. (2006) have been done and the conclusion is clear now - say, at a point $\mathbf{x}$, stretching direction is aligned with the eigenvectors of the Hessian $\bigtriangledown^2 f(\mathbf{x})$ and the stretching length is the square root of the eigenvalues of the Hessian $\bigtriangledown^2 f(\mathbf{x})$, the error e (in various norms) for the linear interpolation of a function f is nearly the minimum; the interpolation error is expressed in terms of the Hessian of the solution - a natural connection between the metric-based generation algorithm and the error estimate; the globally optimal or nearly optimal mesh can be further characterized by the equidistribution of the interpolation error over each element Huang and Sun (2003); Chen and Xu (2004) Alauzet et al. (2006).

Concerning high order $k \geq 2$ interpolation in 2-dimension, we can state the significant results Mbinky et al. (2012). Based on the Sylvester's theorem Comon and Mourrain (1996), rewrite $e(\mathbf{x})$ as:

$$e(\mathbf{x}) = \sum_{i=1}^{r(\mathbf{x})} \lambda_i(\mathbf{x})(\alpha_i(\mathbf{x})x + \beta_i(\mathbf{x})y)^k,$$

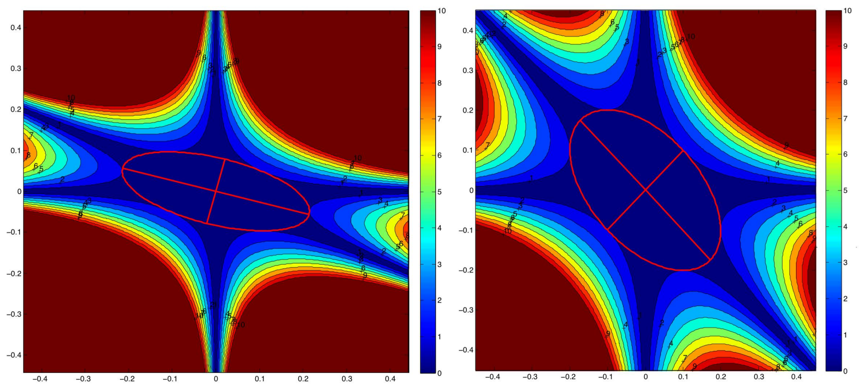


Figure 1.16: Examples of error models: $P_e = 50x^3 - 120x^2y - 1000xy^2 - y^3$ (left), $P_e = x^3 - 500x^2y - 500xy^2 - y^3$ (right). Representation of their iso-values and their corresponding optimal ellipse included in the isoline 1(in red) Mbinky et al. (2012).

where $r(\mathbf{x})$ is the decomposition rank.

For real case,

$$\boldsymbol{g}^*(\mathbf{x}) = Q^{*T}(\mathbf{x}) \begin{pmatrix} \frac{1}{h_1^{*2}(\mathbf{x})} & 0 \\ 0 & \frac{1}{h_2^{*2}(\mathbf{x})} \end{pmatrix} Q^*(\mathbf{x}),$$

and for complex case,

$$\boldsymbol{g}^*(\mathbf{x}) = 2^{-\frac{1}{3}} \bar{Q}^{*T}(\mathbf{x}) \begin{pmatrix} \frac{1}{h_1^{*2}(\mathbf{x})} & 0 \\ 0 & \frac{1}{h_2^{*2}(\mathbf{x})} \end{pmatrix} Q^*(\mathbf{x}),$$

with $Q^{*T}(\mathbf{x})$ is the real transpose and $\bar{Q}^{*T}(\mathbf{x})$ is the conjugate complex transpose of $Q^*(\mathbf{x})$.

The *optimal directions*

$$Q^*(\mathbf{x}) = \begin{pmatrix} \alpha_1(\mathbf{x}) & \beta_1(\mathbf{x}) \\ \alpha_2(\mathbf{x}) & \beta_2(\mathbf{x}) \end{pmatrix}.$$

The *optimal sizes*

$$h_i^*(\mathbf{x}) = \frac{1}{|\lambda_i(\mathbf{x})|^{\frac{1}{k}}}.$$

Concerning high order $k \geq 2$ interpolation in $n \geq 2$-dimension, only a few works Apel (1999a); Coulaud and Loseille (2016) have been made and the conclusion is far from clear until now, and the natural link between the error estimate and the metric-based mesh generation procedure does not exist anymore. The main issue lies in converting the interpretation of the $k+1 (k \geq 1)$ differential of the solution into a metric field. It is generally impossible to factor a homogeneous polynomial in three or higher dimensions into the product of linear and non-negative quadratic functions. But we know, algebraically, the optimal metric field relies on a constrained minimization, i.e. the smallest bound for the $W_{m,p}$-error of kth-order interpolations. A dimensional reduction method can be used to find an approximate $Q^*(\mathbf{x})$ to measure the anisotropic behavior of $\nabla_{k+1} f(\mathbf{x})$ Cao (2007). A log-simplex algorithm can be used to derive the local error model of a given initial k-order homogeneous polynomial Coulaud and Loseille (2016). This provides the optimal metric $\boldsymbol{g}^*(\mathbf{x})$.

Define the mesh vertices density $d = \prod_{i=1}^{n} \frac{1}{h_i}(\mathbf{x})$, the number of elements of the mesh is:

$$N(\boldsymbol{g}(\mathbf{x})) = c \int_{\mathbf{x}\in\Omega} d(\mathbf{x}) d\mathbf{x} = c \int_{\mathbf{x}\in\Omega} \prod_{i=1}^{n} \frac{1}{h_i}(\mathbf{x}) d\mathbf{x},$$

where c is a constant.

Consider error $e(\mathbf{x})$ as:

$$\mid e(\mathbf{x}) \mid \leq \left(\mathbf{x}^T \boldsymbol{g}^*(\mathbf{x})\mathbf{x}\right)^{\frac{k}{2}} \qquad \text{for} \qquad \mathbf{x} = (x, y) \in \mathbb{R}^2.$$

The global optimal continuous metric is to solve a variational calculus problem:

$$\boldsymbol{g}^{*\bar{N}}_{L_p}(\mathbf{x}) = min \left(\int_\Omega |e_{\boldsymbol{g}}(\mathbf{x})|^p d\mathbf{x} \right)^{\frac{1}{p}} = min \left(\int_\Omega |\mathbf{x}^T \boldsymbol{g}(\mathbf{x})\mathbf{x}|^{\frac{kp}{2}} d\mathbf{x} \right)^{\frac{1}{p}}$$

under the constraint $N(\boldsymbol{g}^*(\mathbf{x})) = c \int_\Omega (h_1(\mathbf{x})h_2(\mathbf{x}))^{-1} = \bar{N}$.

Use the Euler-Lagrange necessary condition

$$\forall \delta M, \delta e(\boldsymbol{g}(\mathbf{x}), \delta \boldsymbol{g}(\mathbf{x})) = \alpha \delta N(\boldsymbol{g}(\mathbf{x}), \delta \boldsymbol{g}(\mathbf{x})),$$

here α is a constant. We write:

$$\begin{aligned} &\boldsymbol{g}^*_{L_p}(\mathbf{x}) \\ &= c_1 N \left(\int_\Omega (h_1(\mathbf{x})h_2(\mathbf{x}))^{-\frac{k^2 p}{2(kp+2)}} \right)^{-1} (h_1(\mathbf{x})h_2(\mathbf{x}))^{\frac{k}{kp+2}} \begin{pmatrix} h_1^{-k}(\mathbf{x}) & 0 \\ 0 & h_2^{-k}(\mathbf{x}) \end{pmatrix}, \end{aligned}$$

where c_1 is a constant.

Given a total number $\bar{N} > 0$, the optimal value of the error is:

$$e^*(\mathbf{x}) = c_2 2^{\frac{k}{2}} \bar{N}^{-\frac{k}{2}} \left(\int_\Omega (h_1(\mathbf{x})h_2(\mathbf{x}))^{-\frac{k^2 p}{2(kp+2)}} \right)^{\frac{k}{2}} (h_1(\mathbf{x})h_2(\mathbf{x}))^{-\frac{k^2 p}{2(kp+2)}},$$

here c_2 is a constant.

Given a interpolation error $\bar{e} > 0$, the optimal value of the number is:

$$N^* = 2c_3\bar{e}^{-\frac{2}{k}} \left(\int_\Omega (h_1(\mathbf{x})h_2(\mathbf{x}))^{-\frac{k^2 p}{2(kp+2)}} \right) (h_1(\mathbf{x})h_2(\mathbf{x}))^{-\frac{kp}{kp+2}},$$

where c_3 is a constant.

For a sequence of continuous meshes, a global kth-order of spatial mesh convergence is given:

$$\| f(\mathbf{x}) - \Pi^k_{\boldsymbol{g}^*_{L_p}} f(\mathbf{x}) \|_{L^p(\Omega)} \leq \frac{C_{st}}{N^{\frac{k}{2}}}.$$

5 | Interpolation error with curvilinear meshes

Error estimates in the context of non-affine finite elements has been proved since the early efforts: given a finite element set K and $\| \cdot \|_{H^m(K)}$ denotes the Sobolev norms, a general theory Ciarlet and Raviart (1972) have been developed for obtaining asymptotic estimates of the form $\| u - \Pi^k_h u \|_{H^m(K)} = O(h^{k+1-m})$ for simplical and quadrilateral curved finite elements of isoparametric type, where h is asymptotically equal to the diameter of K. The paper Botti (2012) compared physical and reference frame discontinuous Galerkin (dG) discretizations with emphasis on the influence of reference-to-physical frame mappings on the discrete space properties, assessed the excellence of physical frame discrete spaces in terms of approximation capabilities as well as the increased flexibility compared to reference frame discretizations - say, whenever curved elements are considered, non-affine reference-to-physical frame mappings are able to spoil the convergence properties of reference frame discrete spaces, but this poorly documented drawback does not affect basis functions defined directly in the physical frame. In this thesis Chapter 3, we shows that, even if $f(\mathbf{x})$ is only quadratic in $\mathbf{x}$, i.e., $C_{ijk} = 0$, the finite element interpolation error of $f(\mathbf{x}(t))$ with quadratic isoparametric elements does not vanish, due to the curvature of the curve $\mathcal{C}$.

But we can see by a wise choice of triangle geometry, we still be able to have a good convergence properties. Let g be the element order and k be the interpolation order, we have:

$g1, kn$ **finite triangles** For such triangles, we have:

$$||f - \Pi f|| \leq C_1 h|f'| + C_2 h^2 |f''| + \dots$$

Where h is the triangle size, and the constants $C_i = 0$ for $i \leq n$ which implies that the term in h^{n+1} dominates for small h. This works because we have an affine transformation for the element which implies that Πf is a polynomial of order n. If we take non-affine elements, this is no more the case. But the interpolate is polynomial in the reference space so we can look at the error in that space.

$gn, k1$ **finite triangles** We now look at $gn, p1$ finite triangles.

In infinity-norm, we have:

$$||f - \Pi f||_{\Omega}^{\infty} = \max_{e} ||\hat{f}_e - \Pi \hat{f}_e||_{\Omega_{\text{ref}}}^{\infty}$$

where $\hat{f}_e$ is the function mapped into the reference triangle and $\Pi \hat{f}_e$ is the interpolate in the reference element which is polynomial. We can bound the error in the reference triangle the same way we do it for classical FE:

$$||\hat{f}_e - \Pi \hat{f}_e||_{\Omega_{\text{ref}}}^{\infty} \leq C_{\infty} |\hat{f}_e''|$$

here C_{∞} is a constant.

Now, the second derivative of $\hat{f}_e$ can be bounded from f'' [3]:

$$|\hat{f}_e''| \leq \alpha_e h_e^2 |f''|_{\Omega_e}$$

where h_e is the triangle size, and α_e are constants that depend on the geometry of the triangles. If the triangle is linear, α_e is equal to 1. α_e can be bigger than 1 which can explain the conclusion of Botti—that curved meshes can deteriorate the convergence—since he did not take curved meshes that are optimal for α_e. What is interesting is that if we choose wisely the mapping of the triangle, then α_e can be smaller than 1 and possibly very close to zero which would implies an order-3 error. If this is true, then for a given fixed point set and a given triangulation (whose topology is fixed), it is always possible to find a geometry of the triangles such that $\alpha_e \leq 1$. Indeed, in the worst case, the optimal geometry is for linear triangles for which $\alpha_e = 1$.

In 1-norm, we have:

$$||f - \Pi f||_{\Omega}^{1} = \sum_{e} ||(\hat{f}_e - \Pi \hat{f}_e) J_e||_{\Omega_{\text{ref}}}^{1}$$

where J is the Jacobian determinant. Moreover, we have:

$$||(\hat{f}_e - \Pi \hat{f}_e) J_e||_{\Omega_{\text{ref}}}^{1} \leq C_1 |\hat{f}_e''| \max_{\Omega_{\text{ref}}} J_e$$

here C_1 is a constant.

In 2-norm, we have:

$$||f - \Pi f||_{\Omega}^{2} = \sqrt{\sum_{e} ||(\hat{f}_e - \Pi \hat{f}_e) J_e||_{\Omega_{\text{ref}}}^{2}}$$

[3] We also have: $|\hat{f}_e| = |f|$ and $|\hat{f}_e'| \leq c_e h_e |f'|$ where $c_e = 1$ for a linear triangle.

and:

$$||(\hat{f}_e - \Pi\hat{f}_e)J_e||^2_{\Omega_{\text{ref}}} \leq C_2|\hat{f}''_e| \max_{\Omega_{\text{ref}}} J_e$$

here C_2 is a constant, and $|\hat{f}''_e|$ can be bounded as before and

$$\max_{\Omega_{\text{ref}}} J_e \leq \beta_e h_e^2$$

where β_e also are constants that depend on the geometry of the triangles and is bounded by 1 for linear triangles. For a valid curved triangle, we can prove that β_e is bounded by n^2 if n is its order.

In all three cases, we obtain this bound:

$$||f - \Pi f||_{\Omega} \leq C \max_e(\gamma_e h_e^2 |f''|_{\Omega_e})$$

here C is a constant.

We see that we can do the same developments for high-order interpolation and obtain the same conclusion, with coefficients γ_e that can possibly be reduced by a wise choice of triangle geometry.

CHAPTER 2

Curvilinear mesh adaptation with a given metric

This chapter is a reproduction of the following paper

Ruili Zhang, Amaury Johnen, Jean-François Remacle, *Curvilinear mesh adaptation*, In 28th International Meshing Roundtable, Paper, October 14 - 17, 2019.

Abstract

This paper aims at addressing the following issue. Assume a unit square: $\Omega = \{(x^1, x^2) \in [0,1] \times [0,1]\}$ and a Riemannian metric $g_{ij}(x^1, x^2)$ defined on U. Assume a mesh $\mathcal{T}$ of U that consist in non overlapping valid quadratic triangles that are potentially curved. Is it possible to build a unit quadratic mesh of U i.e. a mesh that has quasi-unit curvilinear edges and quasi-unit curvilinear triangles? This paper aims at providing an embryo of solution to the problem of curvilinear mesh adaptation. The method that is proposed is based on standard differential geometry concepts. At first, the concept of geodesics in Riemannian spaces is quickly presented: the geodesic between two points as well as the unit geodesic starting at a given point with a given direction are the two main tools that allow us to address our issue. Our mesh generation procedure is done in two steps. At first, points are distributed in the unit square U in a frontal fashion, ensuring that two points are never too close to each other in the geodesic sense. Then, a simple isotropic Delaunay triangulation of those points is created. Curvilinear edge swaps as then performed in order to build the unit mesh. Notions of curvilinear mesh quality is defined as well that allow to drive the edge swapping procedure. Examples of curvilinear unit meshes are finally presented.

Keywords: curvilinear mesh generation, mesh adaptation, Riemannian metric field, geodesic

1 Introduction

There is a growing consensus that state of the art Finite Volume and Finite Element technologies require, and will continue to require too extensive computational resources to provide the necessary resolution, even at the rate with which computational power increases. The requirement for high resolution naturally leads us to consider methods with higher order of grid convergence than the classical (formal) 2nd order provided by most industrial grade codes. This indicates that higher-order discretization methods will replace at some point the finite volume/element solvers of today, at least for part of their applications. The development of high-order numerical technologies for CFD is underway for many years now. For example, Discontinuous Galerkin methods (DGM) have been largely studied in the literature, initially in a quite theoretical context Cockburn and Shu (1989), and now in the application point of view Kroll (2010). In many contributions, it is shown that the accuracy of the method strongly depends of the accuracy of the geometrical discretization Bernard et al. (2009). In other words, the following question is raised: yes, we have the high order methods, but how do we get the meshes?

Several research teams are now actively working in the domain of curvilinear meshing. This new subject is considered as crucial for the future of CFD Slotnick et al. (2014) and large fundings have been given to some brilliant researchers to allow innovation in the domain (our colleague Xevi Roca has recently obtained an ERC starting grant on the subject).

A good research project should ideally be summarized as a simple yet fundamental question. It is very much the case here. Assume a unit square

$$\Omega = \{(x^1, x^2) \in [0,1] \times [0,1]\}$$

and a smooth function $f(x^1, x^2)$ defined on the square. Consider a mesh $\mathcal{T}$ made of P^2 triangles that exactly covers the square. How can we compute the mesh $\mathcal{T}$ that minimizes the interpolation error $\|\Pi f - f\|_\Omega$. Here, Π is the so-called Clément interpolation of f on the mesh Ern and Guermond (2013). This problem is the problem of curvilinear mesh adaptation. The solution of that problem requires to address three main open questions:

1. What is the geometrical structure of the interpolation error in the P^2 case?
2. How can we relate this structure with the geometry/shape of a P^2 triangle?
3. How can we build a mesh made of optimal P^2 triangles?

The first question is related to error estimation and we will not deal with it in this paper.

In this first attempt, we will start with a simpler statement. A Riemannian metric field $g_{ij}(x^1, x^2)$ is defined on the unit square. This metric field is supposed to be the result of the error estimation. Our aim is thus to build a unit P^2 mesh with respect to that metric. A discrete mesh $\mathcal{T}$ of a domain Ω is a unit mesh with respect to Riemannian metric space $\mathbf{g}(x^1, x^2)$ if all its elements are quasi-unit. More specifically, a curvilinear triangle t defined by its list of edges $e_i,\ i = 1, 2, 3$ is said to be quasi-unit if all its adimensional edges lengths $\mathcal{L}_{e_i} \in [0.7, 1.4]$[1]. Generating unit straight-sided meshes is a problem that has been largely studied, both in the theoretical point of view and on the application point of view Frey and Alauzet (2005). Here, our aim is to allow edges to become curved, leading to unit meshes that would potentially contain way less triangles.

The paper is structured as follows. Our mesh generation technique essentially relies on the computation of the shortest parabola between two points and on a unit-size parabola starting in a given direction. In Section 2, standard notions of geodesics in Riemann spaces are briefly exposed. Algorithms that compute geodesic parabolas are explained as well.

[1]This range is not arbitrary. When a long edge of size 1.4 is split, it should not become a short edge. Other authors choose $[\sqrt{2}/2, \sqrt{2}]$

The mesh generation approach that we advocate is in two steps. We first generate the points in a frontal fashion Baudouin et al. (2014). In that process, we ensure that (i) two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are never too close to each other and (ii) that there exist four points $\mathbf{x}_{ij},\ j = 1,\dots,4$ in the vicinity of each point $\mathbf{x}_i$ that are not too far to $\mathbf{x}_i$ i.e. that can form edges in the prescribed range $[0.7, 1.4]$.

Then, points are connected in a very standard "isotropic" fashion. The mesh is subsequently modified using curvilinear edge swaps in order to form the desired unit mesh. A curvilinear mesh quality criterion is proposed that allow to drive the edge swapping process.

In §5, some unit meshes are presented that adapt to analytical metric fields.

In what follows, we illustrate concepts of unit circle and geodesics using the following *toy metric tensor*:

$$\mathbf{g}(x^1,x^2) = \begin{pmatrix} g_{11} & g_{12} \\ g_{12} & g_{22} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \frac{1}{l_{\min}^2} & 0 \\ 0 & \frac{1}{l_{\max}^2} \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{2.1}$$

with

$$\mathbf{x} = \{x^1, x^2\},\ \ r = \|\mathbf{x}\|,\ \ \theta = \arctan(x^2/x^1),$$

$$l_{\min} = \epsilon + l_{\max}(1 - \exp(-((r - r_0)/h)^2).$$

2 | Geodesics

In a Riemannian space, the length of curve $\mathcal{C}$ is computed as

$$\mathcal{L}_{\mathcal{C}} = \int_{\mathcal{C}} \sqrt{g_{ij}dx^i dx^j}$$

The geodesic between two points $\mathbf{x}_1$ and $\mathbf{x}_s$ is the shortest path $\mathcal{C}$ between those two points. It is possible to compute geodesics by solving a set of coupled ordinary differential equation (ODE). Defining the so-called Christoffel symbols

$$\Gamma^i{}_{kl} = \tfrac{1}{2} g^{-1}_{im} \left(\frac{\partial g_{mk}}{\partial x^l} + \frac{\partial g_{ml}}{\partial x^k} - \frac{\partial g_{kl}}{\partial x^m} \right) = \tfrac{1}{2} g^{-1}_{im} (g_{mk,l} + g_{ml,k} - g_{kl,m}),$$

the ODE's of geodesics are written:

$$\frac{d^2 x^i}{dt^2} + \Gamma^i_{jk} \frac{dx^j}{dt}\frac{dx^k}{dt} = 0. \tag{2.2}$$

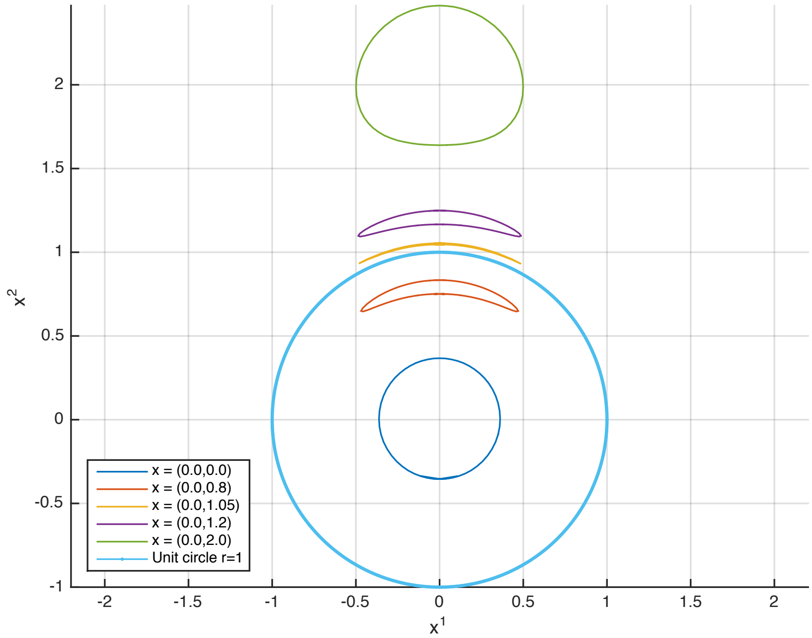


Figure 2.1: Unit circles at different centers for the toy metric (2.1)

2.1 Geodesics and unit circle

Assume a point $\mathbf{x} = \{x^1, x^2\}$ and an initial velocity $\dot{\mathbf{x}} = \{\cos(\alpha), \sin(\alpha)\}$. Equation (2.2) allows to compute geodesic $\mathcal{C}(\alpha)$ which is the geodesic passing by $\mathbf{x}$ and which tangent vector at $\mathbf{x}$ is $\dot{\mathbf{x}}$. In this work, a simple RK2 scheme is used to integrate Equation (2.2) explicitly.

The unit circle centered at $\mathbf{x}$ is the set of end-points of all geodesics $\mathcal{C}(\alpha)$ with $\mathcal{L}_{\mathcal{C}(\alpha)} = 1$ starting at point $\mathbf{x}$. Figure 2.1 shows unit circles with different centers for the toy metric (2.1).

The tangent plane assumption that is usually made in anisotropic meshing theory Frey and Alauzet (2005) leads to unit circles that are ellipsis and where geodesic remain straight lines. Here, Unit circles have a *banana shape* that differes very much with an ellipsis. On Figure 2.2, unit circles corresponding to the principal directions of the metric at point $\{x^1, x^2\} = \{0, 1.2\}$ are drawn, both for true geodesics (left) and in the case of the tangent plane approximation (right).

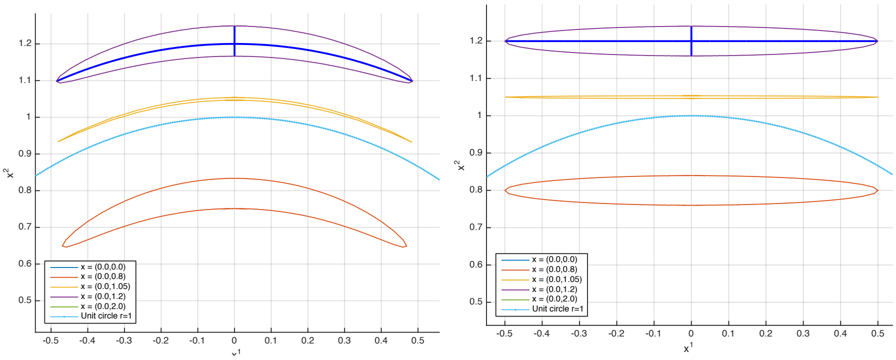


Figure 2.2: Unit circles at different centers for the toy metric (2.1). Left Figure shows circles computed using the exact geodesics while right Figure assumes a constant metric (tangent plane approximation)

2.2 Geodesic curve between two points

Shooting a geodesic from a point $\mathbf{x}$ with velocity $\dot{\mathbf{x}}$ can be solved by integrating the geodesic ODE (2.2) explicitly in t. Now, consider two points $\mathbf{x}_1$ and $\mathbf{x}_2$. If our aim is to find a geodesic between those points, we need to integrate the geodesic ODE (2.2) implicitly. In this work, we choose to simplify that procedure. Quadratic meshes are considered in this paper, which means that "mesh geodesics" are parabola. In order to simplify our formulation even more, we assume that the mid point $\mathbf{x}_{12}$ on the geodesic parabola $\mathcal{C}_{12}$ between $\mathbf{x}_1$ and $\mathbf{x}_2$ is located on the orthogonal bisector of segment $\mathbf{x}_1\mathbf{x}_2$ as:

$$\mathbf{x}_{12} = \frac{1}{2}(\mathbf{x}_1 + \mathbf{x}_2) + \alpha(\mathbf{x}_2 - \mathbf{x}_1) \times \mathbf{e}_3 \;\; , \;\; \alpha \in R.$$

Parametric equation of this geodesic parabola is given by:

$$\begin{aligned} \mathcal{C}_{12} \equiv \mathbf{x}(t,\alpha) &= (1-t)(1-2t)\mathbf{x}_1 + t(2t-1)\mathbf{x}_2 + 4t(1-t)\mathbf{x}_3(\alpha) \\ &= \mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1) + 4t(1-t)\alpha(\mathbf{x}_2 - \mathbf{x}_1) \times \mathbf{e}_3. \end{aligned}$$

Tangent vector at t is computed as,

$$\dot{\mathbf{x}}(t,\alpha) = (\mathbf{x}_2 - \mathbf{x}_1) + (4-8t)\alpha(\mathbf{x}_2 - \mathbf{x}_1) \times \mathbf{e}_3.$$

So, point $\mathbf{x}_{12}$ is computed by minimizing the length of that parabola

$$\mathbf{x}_{12} = arg \min_{\alpha} \mathcal{L}_{\mathcal{C}_{12}} = \int_0^1 \sqrt{\dot{x}^i \dot{x}^j \; g_{ij}(x^i, x^j)} \; dt \tag{2.3}$$

using a golden section algorithm.

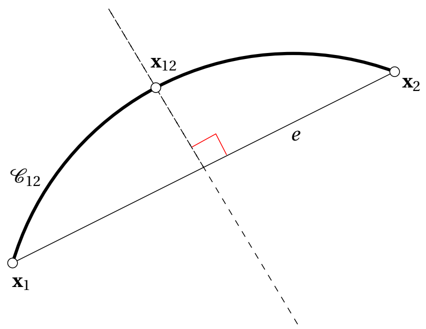


Figure 2.3: Midpoint $\mathbf{x}_{12}$ of a parabola situated on the orthogonal bissector of the straight line $\mathbf{x}_1\mathbf{x}_2$

3 | Generation of points

Assume a $1D$ mesh of the unit square that is compatible with the metric field $g_{ij}(\mathbf{x})$ i.e. where every boundary mesh edges is quasi-unit. The main idea here is to proceed as we did for generating hex dominant meshes Baudouin et al. (2014). The point sampling algorithm is presented in Algorithm 1.

Algorithm 1 Point sampling for the generation of a unit curvilinear mesh

Input: A LIFO queue Q is initialized containing all mesh vertices of the 1D mesh and a metric field $g_{ij}(\mathbf{x})$.
Output: A list L of accepted vertices
while Q is not empty **do**
    $\mathbf{x} \leftarrow Q$: pop vertex $\mathbf{x}$ at the begin of the queue
    Compute $\mathbf{g}(\mathbf{x})$ as well as its eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$ at point $\mathbf{x}$
    Four tentative points $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, $\mathbf{x}_4$ are computed at a geodesic distance equal to 1 in the four directions $\mathbf{v}_1$, $-\mathbf{v}_1$, $\mathbf{v}_2$, $-\mathbf{v}_2$ solving Equation (2.2).
    for $i = 1, \ldots, 4$ **do**
        if $\mathbf{x}_i$ is not too close to any accepted point in L **then**
            Add $\mathbf{x}_i$ at the end of the queue Q
        end if
    end for
    $L \leftarrow L + \mathbf{x}$: add $\mathbf{x}$ in the list L of accepted vertices
end while

Algorithm 1 ensures that there exists no point in the mesh that are too close to another while, on the other hand, ensuring that there exist 4 points that are suffi-

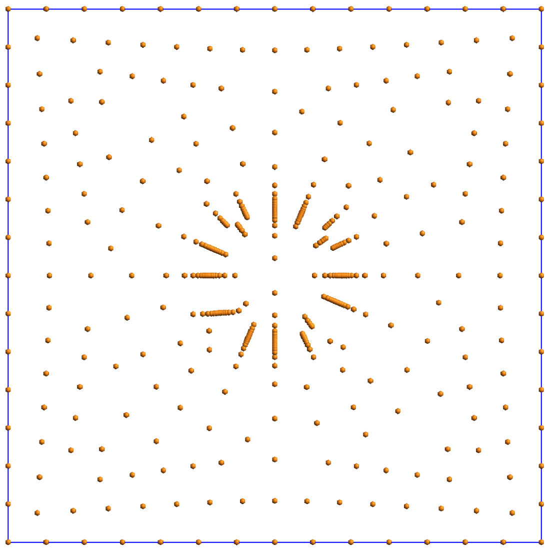


Figure 2.4: Sampling of points using toy metric (2.1) with parameters $\epsilon = 0.01$, $h = 1/\sqrt{10}$, $r_0 = 0.5$ and $l_{\max} = 0.3$. The square is of size 4×4 and is centered at $(x^1, x^2) = (0, 0)$.

ciently close to any point of the mesh. Principal directions of the metric field $\mathbf{v}_1$ and $\mathbf{v}_2$ are used as a "direction field". This is an arbitrary choice. Yet, it has the advantage in most cases to generate meshes that are more structured.

Ensuring that two points are not too close is done using a RTree Beckmann et al. (1990) spatial search structure. The distance between two points is computed as the shortest parabola in the given metric (see Equation (2.3)). Our sampling algorithm applied to the toy metric (2.1) provides the set of points of Figure 2.4.

4 | Generation of triangles

The set of points optimally sampled are then triangulated using an off the shelf constrained Delaunay triangulator such as Gmsh Geuzaine and Remacle (2009) or Triangle Shewchuk (1996). We see on Figure 2.5 that isotropic straight sided elements are not suited for the proposed metric. Here, local mesh modifications Li et al. (2005) will be used to align the mesh with the desired metric. We do not move the points that are optimally sampled. Only edge swaps will be performed, yet in a non usual fashion.

High order points are initially placed on every edge of the straight sided mesh using Equation (2.3). Assume two triangles $t_1(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4)$ and $t_2(\mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_3)$ that

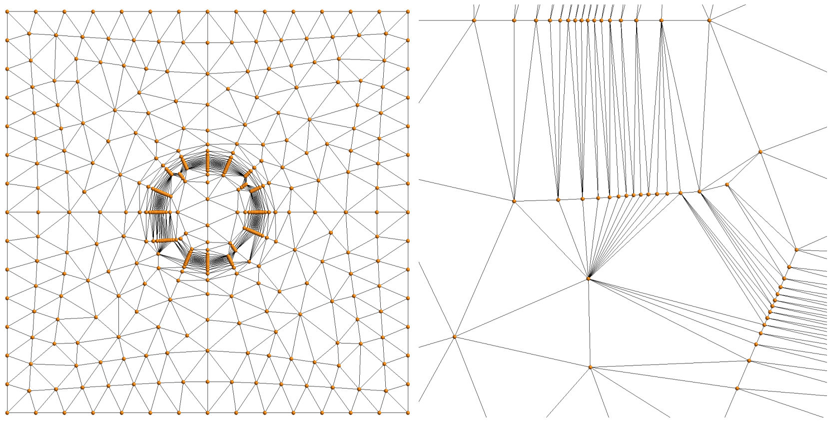


Figure 2.5: Constrained Delaunay mesh constructed using sampled points of Figure 2.4. The triangulation is straight sided. It has been done using no specific metric and is thus clearly not adapted.

share an edge e (see Figure 2.6). Triangles t_1 and t_2 are possibly curvilinear (as in the Figure) and we aim at evaluating the opportunity of replacing edge e by edge e' (edge e' is the geodesic between $\mathbf{x}_3$ and $\mathbf{x}_4$). Two indicators will help us to decide whether an edge swap should be performed:

- The new curvilinear triangles $t'_1(\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2)$ and $t'_2(\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_1)$ have to be both valid. The validity criterion that is used is based on robust estimates that have been developed in Johnen et al. (2013). In short, for t'_1, determinants of Jacobians J_4 ,J_3,J_2,J_{43},J_{32},J_{24} are computed at its 6 nodes. A sufficient condition for triangle t'_1 to be valid is

  $$J_4 > 0\,,\ J_3 > 0\,,\ J_2 > 0\,,\ 4J_{43} > J_3 + J_4\,,\ 4J_{32} > J_3 + J_2\,,\ 4J_{24} > J_2 + J_4.$$

- The quality of the mesh has to be improved by the swap:

  $$\min(q_{\mathbf{g}}(t_1), q_{\mathbf{g}}(t_2)) < \min(q_{\mathbf{g}}(t'_1), q_{\mathbf{g}}(t'_2))$$

  where $q_{\mathbf{g}}(t)$ is a curvilinear quality measure of triangle t with respect to metric field $\mathbf{g}$.

The quality measure that is used here is a direct extension to standard quality measures defined in Shewchuk (2002). We define

$$q_{\mathbf{g}}(t) = \frac{12}{\sqrt{3}} \frac{\int_t \sqrt{\det \mathbf{g}}\, d\mathbf{x}}{\mathcal{L}_{e_1}^2 + \mathcal{L}_{e_2}^2 + \mathcal{L}_{e_3}^2} \tag{2.4}$$

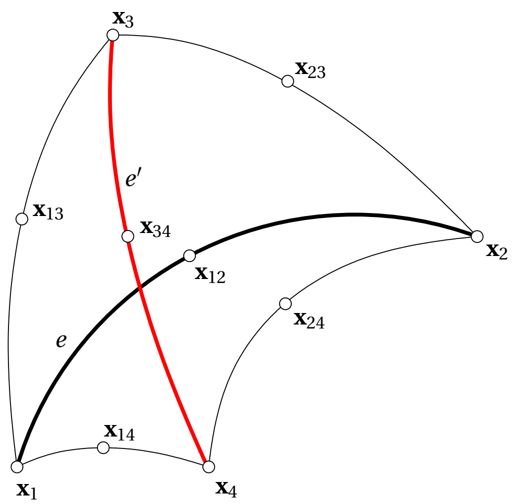


Figure 2.6: Curvilinear edge swap.

where e_1, e_2 and e_3 are the three edges of t, $\mathcal{L}_e$ is the length of e with respect to the metric. Note that triangle inequality is not necessary verified in Riemannian metrics i.e. $\mathcal{L}_{e_1} \leq \mathcal{L}_{e_2} + \mathcal{L}_{e_3}$ is not necessary true. In consequence, quality measure $q_{\mathbf{g}}(t)$ may be larger than one. Edges are swapped until a stable configuration is found.

5 | Examples

5.1 Unit mesh for the toy metric

Figure 2.7 present meshes for the toy metric (2.1). All triangles are valid by construction.

Note here that the corresponding P^1 mesh of our P^2 mesh is totally invalid. It is indeed not possible to generate a P^1 mesh and curving it afterwards without doing curvilinear local mesh modifications (see Figure 2.8).

In the sampling process, points are placed along true geodesics while edges of the mesh are parabola. Parabola that have the same endpoints as true unit geodesics could potentially be longer than 1. Even though the number of long edges that are the consequence of this approximation is quite small, this discrepancy could potentially become annoying. We have addressed that issue by reducing the size of geodesics with the aim at producing parabolas that are of the right unit size. With this fix, edge lengths are in the range $[0.701, 1.66]$ which is very close to the optimal range (see

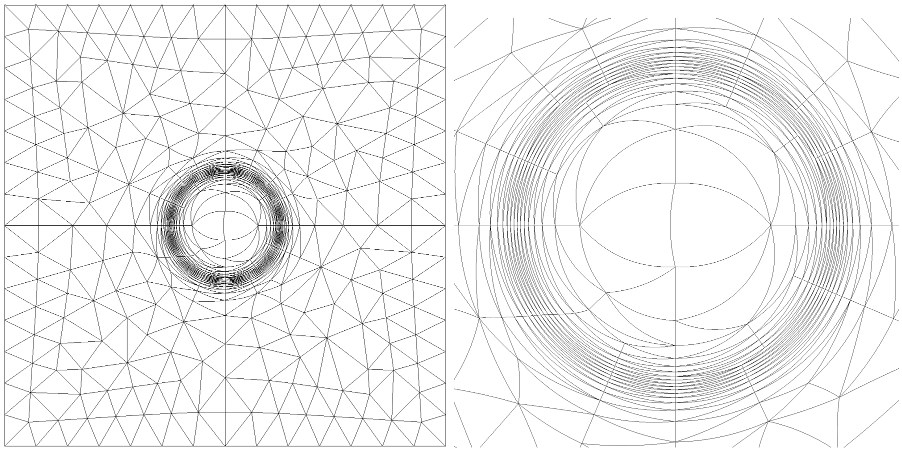


Figure 2.7: Curvilinear mesh of the unit square using the toy metric.

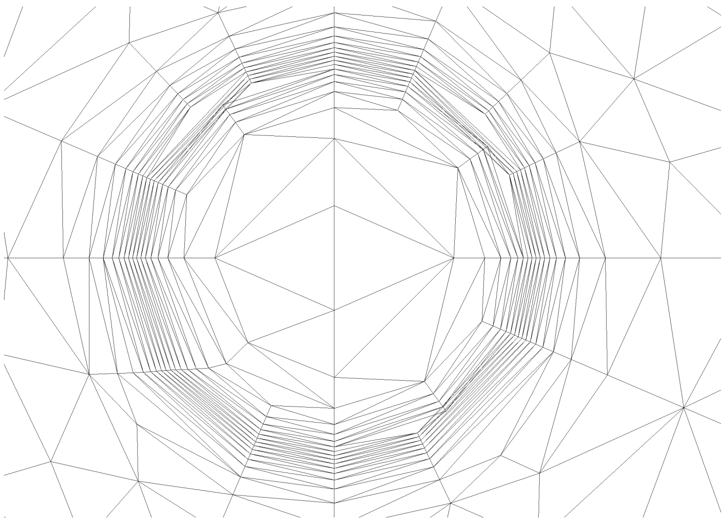


Figure 2.8: This Figure depicts the corresponding P^1 straight sided version of the curvilinear mesh of Figure 2.7. A large amount of the P^1 triangles are invalid while every single P^2 triangle of Figure 2.7 is valid.

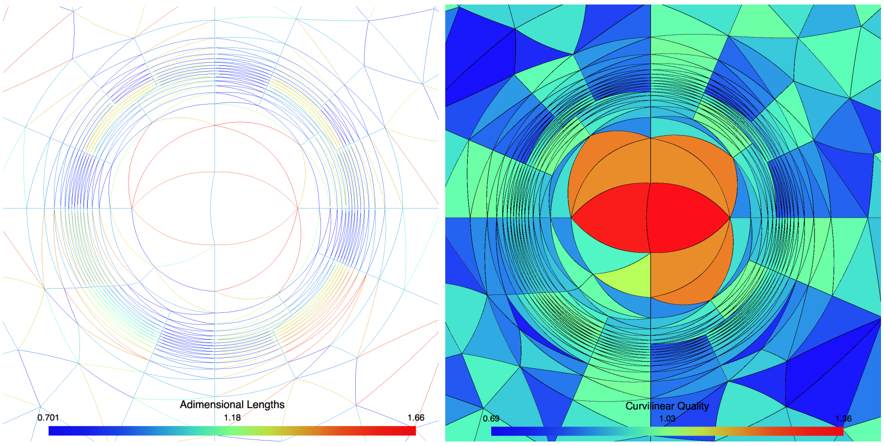


Figure 2.9: Left Figure shows adimensional lengths of edges of the mesh for the toy metric. Right Figure presents P^2 triangle quality measures (2.4).

Figure 2.9). Note that no short edges can exist in the mesh by construction. Long edges are due to the inability of the swapping process to connect points that are close enough without generating invalid P^2 triangles. In further work, other mesh optimizations will be put into place that could enhance even further the quality of the P^2meshes. Quality measures (2.4) are also depicted in Figure 2.9.

5.2 Intersection of three toy metrics

This example consists in placing three toy metrics g_1, g_2 and g_3 in the 4×4 square, centered at different locations with different mesh sizes and intersecting them Frey and Alauzet (2005):

$$g = g_1 \cap g_2 \cap g_3.$$

Meshes are presented in Figure 2.10. A total of 1270 mesh vertices were inserted in the unit square. Then, 840 curvilinear swaps were performed to produce the final mesh. Edges of the mesh have sizes that are in the range $[0.7, 1.8]$.

To obtain this intersection metric, let g_1 and g_2 denote these two metrics:

$$g_1(x^1, x^2) = \begin{pmatrix} g_{111} & g_{112} \\ g_{121} & g_{122} \end{pmatrix} = \begin{pmatrix} v_{11x} & v_{12x} \\ v_{11y} & v_{12y} \end{pmatrix} \begin{pmatrix} \mid \lambda_{11} \mid & 0 \\ 0 & \mid \lambda_{12} \mid \end{pmatrix} \begin{pmatrix} v_{11x} & v_{11y} \\ v_{12x} & v_{12y} \end{pmatrix}$$

$$g_2(x^1, x^2) = \begin{pmatrix} g_{211} & g_{212} \\ g_{221} & g_{222} \end{pmatrix} = \begin{pmatrix} v_{21x} & v_{22x} \\ v_{21y} & v_{22y} \end{pmatrix} \begin{pmatrix} \mid \lambda_{21} \mid & 0 \\ 0 & \mid \lambda_{22} \mid \end{pmatrix} \begin{pmatrix} v_{21x} & v_{21y} \\ v_{22x} & v_{22y} \end{pmatrix}$$

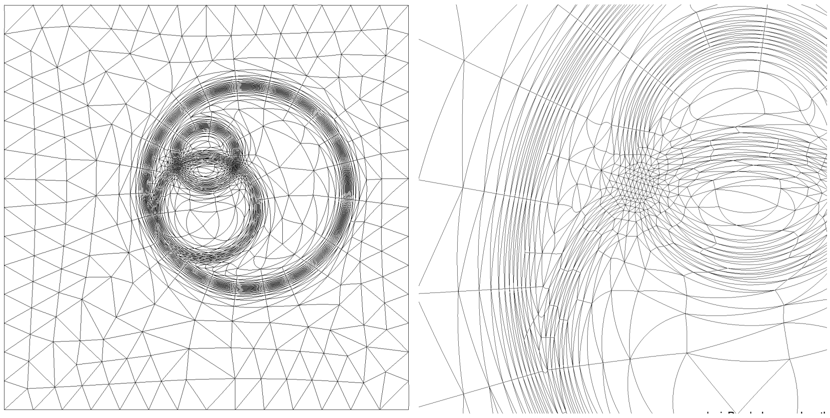


Figure 2.10: Curvilinear mesh of the unit square using the intersection of three toy metrics.

The intersection metric $g_1 \cap g_2$ is then defined by

$$g(x^1, x^2) = \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} = \begin{pmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{pmatrix} \begin{pmatrix} |\lambda_1| & 0 \\ 0 & |\lambda_2| \end{pmatrix} \begin{pmatrix} v_{1x} & v_{1y} \\ v_{2x} & v_{2y} \end{pmatrix}$$

with $v_{1x}, v_{1y}, v_{2x}, v_{2y}$ and λ_1, λ_2 computed in the Algorithm 2.

Then, we can compute the intersection of more metrics:

$$g_1 \cap \cdots \cap g_n = (((g_1 \cap g_2) \cap g_3) \cap \cdots) \cap g_n.$$

And, we can use a weight $w_i, i = 1, 2, \cdots, n$ for each metric:

$$w_1 g_1 \cap \cdots \cap w_n g_n = (((w_1 g_1 \cap w_2 g_2) \cap w_3 g_3) \cap \cdots) \cap w_n g_n.$$

5.3 Other analytical metrics

We have used our technique to adapt to iso-zero of two functions (Figure 2.11 and Figure 2.12). Our procedure seems to remain stable and robust for thicker and thinner adaptations.

6 | Conclusions

In this paper, a new methodology for generating unit curvilinear meshes has been proposed. The method guarantees two important properties in the final mesh:

Algorithm 2 Metric Intersection

input : two given metrics g_1 and g_2
output: a intersection metric $g = g_1 \cap g_2$

if $\lambda_{11}/\lambda_{12} >= \lambda_{21}/\lambda_{22}$ **then**

$$\lambda_{21} = v11x * v11x * g211 + 2 * v11x * v11y * g212 + v11y * v11y * g222;$$
$$\lambda_{22} = v12x * v12x * g211 + 2 * v12x * v12y * g221 + v12y * v12y * g222;$$
$$C = v11x;$$
$$S = v11y;$$

else

$$\lambda_{11} = v21x * v21x * g111 + 2 * v21x * v21y * g112 + v21y * v21y * g122;$$
$$\lambda_{12} = v22x * v22x * g111 + 2 * v22x * v22y * g121 + v22y * v22y * g122;$$
$$C = v21x;$$
$$S = v21y;$$

end

end

if $\lambda_{11} > \lambda_{21}$ **then**
$\lambda_1 = \lambda_{11}$
else
$\lambda_1 = \lambda_{21}$
end
end

if $\lambda_{12} > \lambda_{22}$ **then**
$\lambda_2 = \lambda_{12}$
else
$\lambda_2 = \lambda_{22}$
end
end

$$g_{11} = C * C * \lambda_1 + S * S * \lambda_2;$$
$$g_{12} = C * S * (\lambda_1 - \lambda_2)$$
$$g_{21} = C * S * (\lambda_1 - \lambda_2)$$
$$g_{22} = C * C * \lambda_2 + S * S * \lambda_1;$$
$$v_{1x} = C, v_{1y} = S, v_{2x} = -S, v_{2y} = C$$

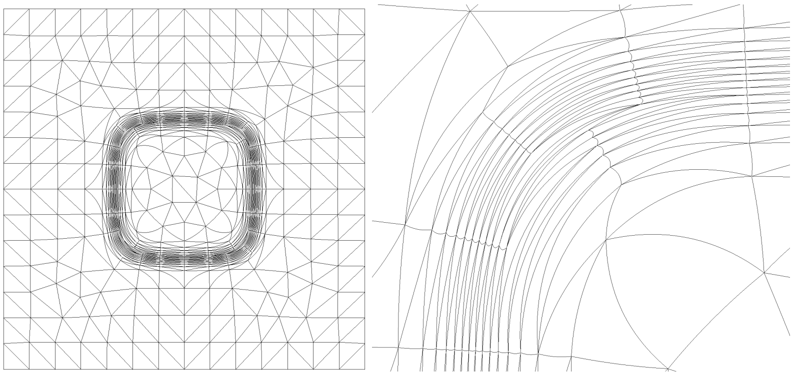


Figure 2.11: Curvilinear mesh adapted to capture $(x^1)^4 + (x^2)^4 = R^4$.

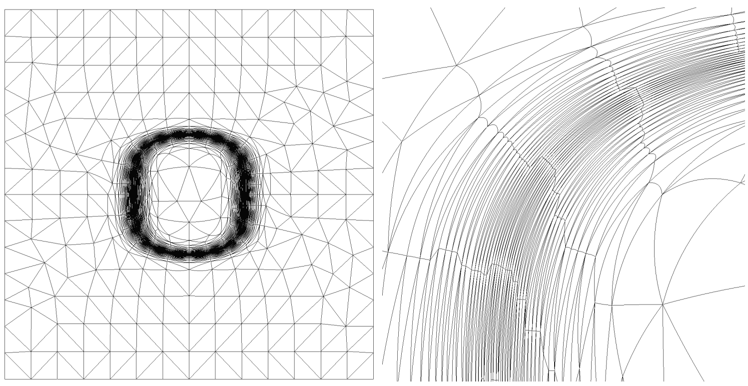


Figure 2.12: Curvilinear mesh adapted to capture $(x^1)^2 + 2(x^2)^4 = R^4$.

1. Generated meshes are valid. The validity of the initial curved mesh (before edge swapping) guaranteed whatever the smoothness of the metric $\mathbf{g}$, given that mesh edges are only parabola and mid-points are located according to parabola geodesic. This important property is due to the fact that P2 meshes are valid at any point of the algorithm. The initial mesh is curved along geodesic. A backtracking (constrained optimization: minimize the length of the parabola while keeping both neighboring triangles valid) step is applied to ensure that every triangle of the mesh is valid. Then, edge swaps are only applied if elements are valid. Note that the validity criterion that is used is robust.

2. No short edges will be exist in the mesh. A spatial search procedure is used for ensuring that any point that is inserted is not too close in the sense of geodesics

than any other point.

This work is now being extended to true adaptation i.e. adapting a mesh to a given function $f(x^1, x^2)$. Even though metrics are still the right tool for driving mesh adaptation at higher orders, basing $\mathbf{g}$ on Hessians of f is not correct anymore for higher orders of approximation. Our future work will be to build metric fields that are suited for high order.

CHAPTER 3

Curvilinear mesh adaptation with an analytic function

This chapter is a reproduction of the following paper

Ruili Zhang, Amaury Johnen, Jean-François Remacle, François Henrotte, Arthur Bawin, *The generation of unit P^2 meshes: Error estimation and mesh adaptation*, Paper, In 29th International Meshing Roundtable (IMR) Information, June 21 - 25, 2021.

Abstract

We propose a new framework for the generation and adaptation of unit curvilinear P^2 meshes in dimension 2. In this approach, curvature is not only used to match curved boundaries but also to capture features of the interpolated solutions, and it results in meshes that would not have been achievable by simply curving *a posteriori* a straight-sided mesh. We proceed as follows. Starting with a smooth function $f(x, y)$, a metric field, based on f and its derivatives up to order 3, is constructed. A unit P^2 mesh is then generated, with edges within an adimensional length range of $[0.7, 1.4]$ with respect to this metric field. Points are then spawned in such a way that their geodesic distance corresponds to edges of unit size, and these points are then connected in a standard *isotropic* fashion. A curvilinear mesh quality criterion is then proposed to drive the mesh optimization process. The triangulation is subsequently modified using straight-sided edge swap, straight-sided edge curving, curvilinear edge swap and Curvilinear Small Polygon Reconnection (CSPR) to form the desired unit mesh. A unit curvilinear mesh containing only valid "Geodesic Delaunay triangles" is obtained this way. A number of application examples are presented in order to demonstrate the capabilities of the mesh adaptation procedure. The resulting adapted meshes allow, most of the times, a significant reduction of the interpolation error compared with straight-sided P^2 meshes of the same density.

Keywords: curvilinear mesh generation, mesh adaptation, Riemannian metric field, geodesic, analytic functions, high-order error, finite element method

1 | Introduction

Scientific computing is now an old science. Solving partial differential equations on a computer is a very common task for aerospace/chemical/ mechanical/electrical engineers. Still, numerical methods for PDEs that have reached a production level such as finite elements are, for most of them, based on numerical schemes that are of the second order of accuracy. Some applications in fluid mechanics or in electromagnetic nonetheless require numerical schemes that are of higher order of accuracy (those schemes are sometimes called high fidelity schemes). It has been proved in many contributions that high-order finite element schemes require high-order meshes, i.e., meshes that capture the curvilinear features of the geometry with a high fidelity as well Bernard et al. (2009). In the last decade, a significant part of the research in mesh generation has thus been devoted to the generation of body fitted curvilinear meshes. The main issue of generating curved meshes is that there exists for now

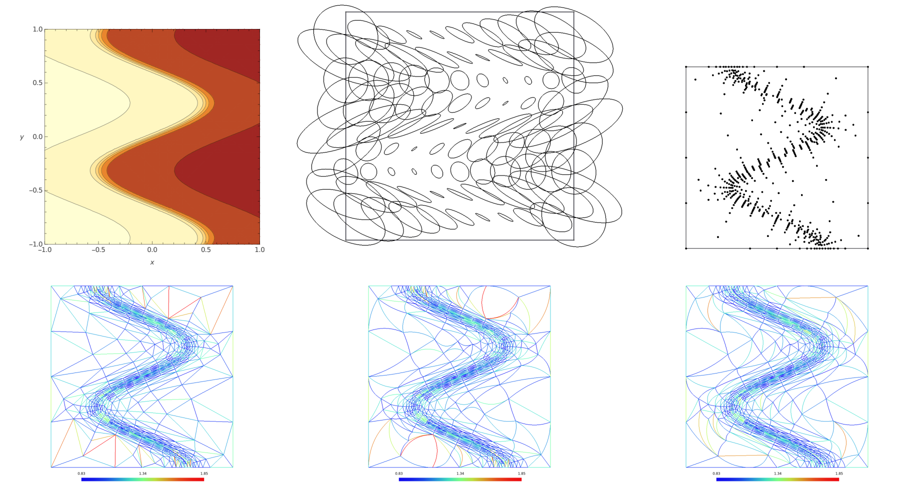


Figure 3.1: Illustration of the whole process of curvilinear mesh generation and adaptation based on a given analytic function. The top left image represents the function(3.19) on the unit square. The top center image depicts the metric field $\mathcal{M}(\mathbf{x})$ computed as explained in section 3. The top right image shows the point spawned according to this metric field. The bottom left image is the straight-sided anisotropic mesh based on the spawned points. The bottom center image shows the initial curvilinear mesh, with the edges colored according to their adimensional lengths. Finally, the bottom right image is the adapted anisotropic mesh including the new Curvilinear Small Polygon Reconnection (CSPR) procedure.

no algorithm that actually generates a P^2 mesh in a direct fashion. State-of-the-art methods generate a straight-sided mesh and place high-order points on the CAD geometry. Then, invalid elements are untangled using various approaches Fortunato and Persson (2016); Hartmann and Leicht (2016); Moxey et al. (2016); Karman et al. (2016); Ruiz-Gironés et al. (2016); Toulorge et al. (2013); Remacle et al. (2013b). Nowadays, body fitted curvilinear meshes start to be used in an industrial context Kroll (2006); Kroll et al. (2015).

High-order meshes have exclusively been used for increasing geometrical accuracy, i.e., to make the mesh represents the geometry of curved parts with high fidelity. The natural extension of the use of curvilinear meshes is *high-order/curvilinear mesh adaptation*. In the linear case, extensive work has been done in anisotropic mesh adaptationAlmeida et al. (2000); Buscaglia and Dari (1997); Castro-Díaz et al. (1997); Formaggia et al. (2004); Dompierre et al. (1997); Huang (2005); Frey and Alauzet (2005); Gruau and Coupez (2005); Li et al. (2005); Tam et al. (2000); Pain et al. (2001). The concept of metric tensor is always central in anisotropic adaptation: it allows to define mesh sizes and directions that allow to minimize the interpolation error Schall et al. (2004); Courty et al. (2006); Chen et al. (2007); Alauzet et al. (2006); Loseille and Alauzet (2011a,b). Yet, all those methods end up with a straight-sided mesh.

This paper is in line with recent paper Zhang et al. (2018) which have provided an embryo of solution to the problem of curvilinear mesh adaptation. In Zhang et al. (2018), an analytical metric field was assumed and unit P^2 meshes were generated based on that metric. In the present paper, we essentially tackle two additional problems that allow to move forward to "true anisotropic curvilinear mesh adaptation" (i.e., optimizing a mesh based on a high-order finite element solution):

1. P^2 error estimates that are currently used in the literature are based on the implicit hypothesis that underlying meshes are straight-sided. Assuming a function $f(x, y) \in C^3$, we construct a metric field that does not assume mesh edges to be straight-sided.

2. The mesh generation procedure that we use for generating the curvilinear meshes is based on the one of Zhang et al. (2018). Yet, we show that using edge swaps only does not always allow to reach a unit mesh. We propose a more general operator - curvilinear small polygon reconnection - that allows to reconnect points in a wider range and generate P^2 unit meshes very robustly.

The paper is structured as follows: in Section 2, a brief review of the interpolation error and algorithm that compute geodesic parabolas is presented; Section 3 describes a new idea of the definition of the metric field that takes into account the curvilinear nature of the mesh; in Section 4, we give a simple illustrative example; in Section 5, we describe and detail the mesh generation approach. The whole process of curvilinear mesh adaptation will be explained based on a running example. All the stages of that process are illustrated in Figure 3.1. Interpolation error is analyzed at the end of the paper.

2 | Interpolation error

2.1 A point of departure

This section starts with a small reflection about a very interesting paper published in 2011 by Lorenzo Botti Botti (2012). In his paper, Botti shows that, in a standard finite element context, curving a mesh may have a dramatic cost in terms of the quality of the finite element interpolation. When we first read this paper, we were already working in curvilinear meshing: we were thus quite puzzled by Botti's conclusions. Clearly, using P^2 meshes and P^2 finite elements (isoparametric finite elements) causes the interpolation to barely pass the patch test. Such an interpolation

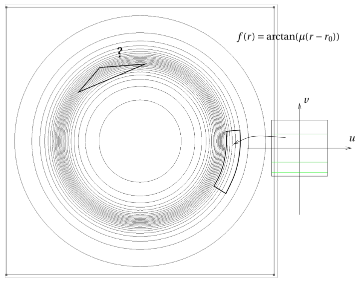


Figure 3.2: ideal elements

is not able to exactly represent a quadratic function in the $\mathbf{x}$ plane which causes damages to the approximation properties of the element.

Now imagine a function $f(r, \theta)$ in polar coordinates that is, say, parabolic in r and θ. On the one hand, f is not polynomial in Euclidean coordinates. On the other hand, with a mesh whose edges are aligned with $\boldsymbol{e}_r$ and $\boldsymbol{e}_\theta$ there would be no interpolation error at all with P^2 finite elements. In Fig. 3.2, the straight-sided anisotropic triangle does its best to align with the isolines of function f whereas the curvilinear quadrangle is able to align with the solution. With a same mesh size, a much lower interpolation error is expected with this quadrangle than with the triangle. This is our starting point: optimal curvilinear edges should be adapted to match the local parametrization of f.

Let thus $f(\mathbf{x}) \in C^3$ be a three time differentiable function, with $\mathbf{x} = (x^1, x^2)$. Its derivatives up to order 3 are respectively its gradient

$$G_i = \frac{\partial f}{\partial x^i},$$

its hessian

$$H_{ij} = \frac{\partial G_i}{\partial x^j} = \frac{\partial^2 f}{\partial x^i \partial x^j},$$

and its third order derivative tensor

$$C_{ijk} = \frac{\partial H_{ij}}{\partial x^k} = \frac{\partial^3 f}{\partial x^i \partial x^j \partial x^k},$$

with $i, j, k = 1, 2$.

Based on these derivatives of $f(\mathbf{x})$, we present an approach to compute a metric field $\mathcal{M}(\mathbf{x})$ that takes into account the curvilinear nature of the P^2 mesh of $f(\mathbf{x})$.

2.2 Parabolic edges

This paper having the aim of building P^2 meshes, the mesh edges are going to be represented by parabolas.

In our approach, the set of parabolas that connect the points $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ is restricted to those with the midpoint $\boldsymbol{X}_{12}$

$$\boldsymbol{X}_{12} = \frac{\boldsymbol{X}_1 + \boldsymbol{X}_2}{2} + \alpha\,(\boldsymbol{X}_2 - \boldsymbol{X}_1) \times \boldsymbol{e}_3\,, \quad \alpha \in \mathbb{R}$$

located on the orthogonal bisector of the segment $\boldsymbol{X}_1\boldsymbol{X}_2$, as shown in Fig. 3.3.

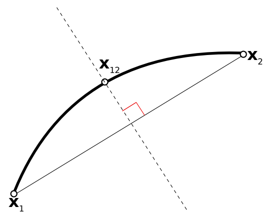


Figure 3.3: Midpoint $\boldsymbol{X}_{12}$ of a parabola situated on the orthogonal bisector of the straight line $\boldsymbol{X}_1\boldsymbol{X}_2$.

The parametric equation of the parabola is then given by

$$\begin{aligned}\mathcal{C} &\equiv \mathbf{x}(t)\\ &= (1-t)(1-2t)\,\boldsymbol{X}_1 + t\,(2t-1)\,\boldsymbol{X}_2 + 4t\,(1-t)\,\mathbf{x}_{12}(\alpha)\\ &= \boldsymbol{X}_1 + t\,(\boldsymbol{X}_2 - \boldsymbol{X}_1) + 4t\,(1-t)\,\alpha\,(\boldsymbol{X}_2 - \boldsymbol{X}_1) \times \boldsymbol{e}_3\\ &= \boldsymbol{X}_1 + th\,\boldsymbol{u} + 4th\,(1-t)\,\alpha\,\boldsymbol{b}, \end{aligned} \tag{3.1}$$

where $\boldsymbol{u} = (u^1, u^2)$ is the unit vector parallel to $\boldsymbol{X}_2 - \boldsymbol{X}_1$, $h = \|\boldsymbol{X}_2 - \boldsymbol{X}_1\|$ and $\boldsymbol{b} = \boldsymbol{u} \times \boldsymbol{e}_3$. Here, α is the only unknown coefficient that allows to move midpoint $\boldsymbol{X}_{12}$ along $\boldsymbol{b}$ and it is computed in such a way that the length of that parabola is minimized. This minimization is performed using a golden section algorithm.

2.3 Interpolation error and mesh size

Assume a curve $\mathbf{x}(t)$, $t \in [0,1]$, a function $f(\mathbf{x}(t))$, and its quadratic Lagrange interpolate $\pi^2 f(\mathbf{x}(t))$. On basis of a Taylor expansion, one knows that the interpolation error can be bounded as follows:

$$\max_{t\in[0,1]} |f(\mathbf{x}(t)) - \pi^2 f(\mathbf{x}(t))| \leq \frac{1}{6} \sup_{t\in[0,1]} \left| \frac{d^3 f(\mathbf{x}(t))}{dt} \right|. \tag{3.2}$$

When $\mathcal{C}$ is a straight edge (i.e., $\alpha = 0$ in (3.1)), one has

$$\dot{\mathbf{x}}(t) = h\boldsymbol{u} \tag{3.3}$$

so that

$$\frac{d^3 f(\mathbf{x}(t))}{dt^3} = C_{ijk}\, \dot{x}^i \dot{x}^j \dot{x}^k = h^3 C_{ijk}\, u^i u^j u^k,$$

and (3.2) becomes

$$\max_{\mathbf{x}\in\mathcal{C}} |f(\mathbf{x}) - \pi^2 f(\mathbf{x})| \leq \frac{h^3}{6} \sup_{\mathbf{x}\in\mathcal{C}} \left| C_{ijk}(\mathbf{x}) u^i u^j u^k \right|, \tag{3.4}$$

where $C_{ijk} = C_{ijk}(\mathbf{x}(t))$, and repeated indices are implicitly summed over (Einstein summation).

Now, if $\mathcal{C}$ is the parabola (3.1), we have

$$\dot{\mathbf{x}}(t) = h\boldsymbol{u} + \alpha h\,(4 - 8t)\,\boldsymbol{b} \quad , \quad \ddot{\mathbf{x}}(t) = -8\alpha h \boldsymbol{b},$$

and it is easy to show that

$$\frac{d^3 f(\mathbf{x}(t))}{dt^3} = C_{ijk}\dot{x}^i \dot{x}^j \dot{x}^k + 3H_{ij}\dot{x}^i \ddot{x}^j \tag{3.5}$$

and

$$\max_{\mathbf{x}\in\mathcal{C}} |f(\mathbf{x}) - \pi^2 f(\mathbf{x})| \leq \frac{h^3}{6} \sup_{\mathbf{x}\in\mathcal{C}} \left| C_{ijk}\dot{x}^i \dot{x}^j \dot{x}^k + 3H_{ij}\dot{x}^i \ddot{x}^j \right|.$$

Equation (3.5) shows that, even if $f(\mathbf{x})$ is only quadratic in $\mathbf{x}$, i.e., $C_{ijk} = 0$, the finite element interpolation error of $f(\mathbf{x}(t))$ with quadratic isoparametric elements does not vanish, due to the curvature of the curve $\mathcal{C}$. This was the observation of Botti in Botti (2012) - say, in a standard finite element context, curving a mesh may have a dramatic cost in terms of the quality of the finite element interpolation.

A relationship similar to (3.3) is needed for the parabolic case, and it is convenient to derive it to reparametrize the curve $\mathbf{x}(t)$ by arc length, i.e., with the arc length

$$s(t) = \int_0^t |\dot{\mathbf{x}}(u)|\, \mathrm{d}u \tag{3.6}$$

as parameter. One has then

$$\partial_s \mathbf{x}(s) = \boldsymbol{g} + s\kappa\, \boldsymbol{e}_3 \times \boldsymbol{g},$$

where $\boldsymbol{g}$ is the unit vector tangent to the curve, and κ the geodesic curvature, using the basis vectors of a Darboux frame Radzevich (2013). One has now the relationship equivalent to (3.3) for the parabola,

$$\begin{aligned} \dot{\mathbf{x}}(t=0) &= \partial_s x(s=0)\, \partial_t s(t=0) \\ &= \boldsymbol{g}\, |\dot{\mathbf{x}}(0)| = \mathcal{L}\boldsymbol{g} \end{aligned}$$

with
$$\mathcal{L} = h\sqrt{1+16\alpha^2}.$$

Moreover, one can show that

$$(\mathbb{I} - \boldsymbol{g}\boldsymbol{g}^T)\ddot{\mathbf{x}}(t=0) = \kappa(\boldsymbol{e}_3 \times \boldsymbol{g})\mathcal{L}^2.$$

Assuming that $(\mathbb{I}-\boldsymbol{g}\boldsymbol{g}^T)\ddot{\mathbf{x}}(t=0)$ is not too different from $\ddot{\mathbf{x}}(t=0)$, and that $C_{ijk}(\mathbf{x})$ and $H_{ij}(\mathbf{x})$ do not vary too much over $\mathcal{C}$, one can write

$$\begin{aligned}&\max_{\mathbf{x}\in\mathcal{C}} |f(\mathbf{x}) - \pi^2 f(\mathbf{x})| \leq \\ &\frac{\mathcal{L}^3}{6}\left|C_{ijk}(\mathbf{x})g^i g^j g^k\right| + \frac{\mathcal{L}^3}{2}\left|H_{ij}(\mathbf{x})g^i\kappa(\boldsymbol{e}_3 \times \boldsymbol{g})^j\right|\end{aligned}$$

although doing so may be too conservative in the sense that this estimate assumes that errors due to polynomial approximation and geometry add to each other *while they may actually balance each other*. We thus stick to

$$\begin{aligned}&\max_{\mathbf{x}\in\mathcal{C}} |f(\mathbf{x}) - \pi^2 f(\mathbf{x})| \\ &\leq \frac{\mathcal{L}^3}{6}\sup_{t\in[0,1]}\left|C_{ijk}(\mathbf{x}(t))\,g^i g^j g^k + 3H_{ij}(\mathbf{x}(t))\,g^i\kappa(\boldsymbol{e}_3 \times \boldsymbol{g})^j\right|.\end{aligned} \tag{3.7}$$

If we pose
$$E = \left|C_{ijk}(\mathbf{x})\,g^i g^j g^k + 3H_{ij}(\mathbf{x})\,g^i\kappa(\boldsymbol{e}_3 \times \boldsymbol{g})^j\right|, \tag{3.8}$$

the interpolation error is thus of the form $\epsilon \simeq \mathcal{L}^3/6\,E$, and since the goal is to adapt the meshsize in order to have an error equidistribution ϵ among edges, we choose

$$\mathcal{L} = (6\epsilon/E)^{1/3}. \tag{3.9}$$

3 | Construction of the metric field

A classical technique in anisotropic mesh generation consists in defining an auxiliary metric field $\mathcal{M}(\mathbf{x})$ under which the sought anisotropic mesh is a unit mesh, i.e., a mesh with edges of approximately unit length. This metric field is defined by two orthogonal unit vectors $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ and two mesh sizes h_1 and h_2 at every point $\boldsymbol{p}$ of the domain. It is customary to choose the vectors $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ as the eigenvectors of the Hessian H_{ij}, so that one has

$$\mathcal{M}(\mathbf{x}) = \begin{pmatrix}\boldsymbol{g}_1 & \boldsymbol{g}_2\end{pmatrix}\begin{pmatrix}\frac{1}{h_1^2} & 0 \\ 0 & \frac{1}{h_2^2}\end{pmatrix}\begin{pmatrix}\boldsymbol{g}_1 & \boldsymbol{g}_2\end{pmatrix}^T. \tag{3.10}$$

This arbitrary choice has two principal virtues:

1. It allows a maximum amount of anisotropy in the P^1 case.
2. If the function f is C^2, $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ are continuous and mesh orientation varies smoothly.

The meshsizes h_1 and h_2 are calculated so as to maintain the interpolation error below a prescribed target error ϵ. We shall try to keep up with these properties in the context of curvilinear meshes.

The main idea is to assume that the edges of the anisotropic mesh at point $\boldsymbol{p}$ are oriented along either the iso-contour of f or the direction of the gradient ∇f, i.e., along the curves defined as follows:

1. Curve $\mathcal{C}_1 \equiv \mathbf{x}_1(t)$ is the quadratic approximation of the iso-contour $f(\mathbf{x}_1(t)) = f(\boldsymbol{p})$.
2. Curve $\mathcal{C}_2 \equiv \mathbf{x}_2(t)$ is the quadratic approximation of the local downhill gradient going through $\boldsymbol{p}$.

Here again, it is convenient to work with curves parametrized by arc-length. Let

$$\mathcal{C}_1 \equiv \mathbf{x}_1(s) = \boldsymbol{p} + \mathbf{g}_1 s + \kappa_1 \mathbf{g}_2 \frac{s^2}{2} \tag{3.11}$$

and

$$\mathcal{C}_2 \equiv \mathbf{x}_2(s) = \boldsymbol{p} + \boldsymbol{g}_2 s + \kappa_2 \boldsymbol{g}_1 \frac{s^2}{2}. \tag{3.12}$$

with, as explained above, the unit tangent vectors

$$\boldsymbol{g}_1 = \frac{\nabla^\perp f|_{\boldsymbol{p}}}{\|\nabla^\perp f|_{\boldsymbol{p}}\|} = \frac{1}{(f_{x^1}^2 + f_{x^2}^2)^{1/2}} \begin{pmatrix} -f_{x^2} \\ f_{x^1} \end{pmatrix},$$

$$\boldsymbol{g}_2 = \frac{\nabla f|_{\boldsymbol{p}}}{\|\nabla f|_{\boldsymbol{p}}\|} = \frac{1}{(f_{x^1}^2 + f_{x^2}^2)^{1/2}} \begin{pmatrix} f_{x^1} \\ f_{x^2} \end{pmatrix}$$

where the shorthand notations

$$f_{x^1} = \frac{\partial f}{\partial x^1}(\boldsymbol{p}) \quad , \quad f_{x^2} = \frac{\partial f}{\partial x^2}(\boldsymbol{p})$$

have been used.

Taylor expansion limited to order 2 of $f(\mathbf{x})$ around $\boldsymbol{p}$ writes

$$\begin{aligned} f(\mathbf{x}) \simeq f(\boldsymbol{p}) &+ \nabla f|_{\boldsymbol{p}} (\mathbf{x} - \boldsymbol{p}) \\ &+ \frac{1}{2} (\mathbf{x} - \boldsymbol{p})^T \, H|_{\boldsymbol{p}} (\mathbf{x} - \boldsymbol{p}). \end{aligned} \tag{3.13}$$

If the curve $\mathbf{x}_1(s)$ runs along the isovalue of f, one can write $f(\mathbf{x}_1(s)) = f(\boldsymbol{p})$ and, using (3.11), one can write

$$\mathbf{x} - \boldsymbol{p} \equiv \mathbf{x}_1(s) - \boldsymbol{p} = \mathbf{g}_1 s + \kappa_1 \mathbf{g}_2 \frac{s^2}{2}$$

to be inserted into (3.13) to give

$$\underbrace{\nabla f|_{\boldsymbol{p}} \cdot \boldsymbol{g}_1}_{=0} s + \left[\kappa_1 \nabla f|_{\boldsymbol{p}} \cdot \boldsymbol{g}_2 + \boldsymbol{g}_1^T H|_{\boldsymbol{p}} \boldsymbol{g}_1\right] \frac{s^2}{2} + \mathcal{O}(s^3) = 0$$

Since the equation is true in a neighborhood of $s = 0$, the quantity between bracket must vanish, and the identity

$$\kappa_1 \nabla f|_{\boldsymbol{p}} \cdot \boldsymbol{g}_2 + \boldsymbol{g}_1^T H|_{\boldsymbol{p}} \boldsymbol{g}_1 = 0$$

gives the expression of the curvature κ_1 needed to finalize the identification of the curve (3.11)

$$\begin{aligned}\kappa_1 &= -\frac{\boldsymbol{g}_1^T H|_{\boldsymbol{p}} \boldsymbol{g}_1}{\nabla f|_{\boldsymbol{p}} \cdot \boldsymbol{g}_2} \\ &= \frac{-f_{x^2}^2 f_{x^1x^1} + 2 f_{x^1} f_{x^2} f_{x^1x^2} - f_{x^1}^2 f_{x^2x^2}}{(f_{x^1}^2 + f_{x^2}^2)^{3/2}}.\end{aligned}$$

Next, we turn to the second curve (3.12), which is to be aligned with $\nabla f(\boldsymbol{p})$. A Taylor expansion around $\boldsymbol{p}$ of the gradient this time gives

$$\nabla f(\mathbf{x}) \simeq \nabla f|_{\boldsymbol{p}} + H|_{\boldsymbol{p}}(\mathbf{x} - \boldsymbol{p}) \tag{3.14}$$

with, using (3.12),

$$\mathbf{x} - \boldsymbol{p} \equiv \mathbf{x}_2(s) - \boldsymbol{p} = \mathbf{g}_2 + \kappa_2 \mathbf{g}_1 \frac{s^2}{2}. \tag{3.15}$$

On the other hand, one has

$$\frac{\partial \mathbf{x}_2(s)}{\partial s} = \boldsymbol{g}_2 + \kappa_2 \boldsymbol{g}_1 \tag{3.16}$$

and the alignment of the curve with the gradient reads

$$\nabla f(\mathbf{x}_2(s)) \times \frac{\partial \mathbf{x}_2(s)}{\partial s} = 0, \tag{3.17}$$

where $\times$ is the vector product. Substituting (3.14), (3.15) and (3.16) into (3.17) yields

$$\underbrace{\nabla f|_{\boldsymbol{p}} \times \boldsymbol{g}_2}_{=0} + \left[\kappa_2 \nabla f|_{\boldsymbol{p}} \times \boldsymbol{g}_1 + (H|_{\boldsymbol{p}} \boldsymbol{g}_2) \times \boldsymbol{g}_2\right] s + \mathcal{O}(s^2) = 0.$$

Again, as this equation is true in a neighborhood of $s = 0$, the quantity between bracket must vanish. Noting that, for any vector $\boldsymbol{V}$, one has $\boldsymbol{V} \times \boldsymbol{g}_2 = \boldsymbol{V} \cdot \boldsymbol{g}_1$ and $\boldsymbol{V} \times \boldsymbol{g}_1 = -\boldsymbol{V} \cdot \boldsymbol{g}_2$, one obtains for the curvature κ_2 needed to finalize the identification of the curve (3.12) the expression

$$\begin{aligned}\kappa_2 &= \frac{\boldsymbol{g}_1^T H|_{\boldsymbol{p}} \boldsymbol{g}_2}{(f_{x^1}^2 + f_{x^2}^2)^{1/2} \boldsymbol{g}_2 \cdot \boldsymbol{g}_2} \\ &= \frac{f_{x^1} f_{x^2} (f_{x^2x^2} - f_{x^1x^1}) + (f_{x^1}^2 - f_{x^2}^2) f_{x^1x^2}}{(f_{x^1}^2 + f_{x^2}^2)^{3/2}}.\end{aligned}$$

Using (3.8), we can thus finally write

$$E_{1,2} = \left| C_{ijk}(\boldsymbol{p}) g_{1,2}^i g_{1,2}^j g_{1,2}^k + 3\kappa_{1,2} \, H_{ij}(\boldsymbol{p}) g_{1,2}^i g_{2,1}^j \right|.$$

Using (3.9), the mesh sizes are finally defined by

$$h_{1,2} = \mathcal{L}_{1,2} = (6\epsilon / E_{1,2})^{1/3}. \tag{3.18}$$

Two remarks need to be made regarding the error estimate that has just been proposed. At first, some modification is clearly needed whenever $f_{x^1} = 0$ or $f_{x^2} = 0$, i.e., when the function is locally constant. In order to define $\mathcal{M}$ everywhere in the domain, orthogonal directions $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ are computed everywhere where ∇f does not vanish. Then, well-defined directions are extended by a smoother borrowed from our cross field solver Beaufort et al. (2017). When the function f is constant, mesh sizes are limited to a user defined maximal size h_{max}.

Finally, we can note that the orthogonal directions $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ that has been chosen here are somewhat arbitrary. There might be other choices that have some advantages over the eigenvectors of H. Yet, the analytical example described in §4 shows that our choice makes sense.

4 | A simple illustrative example

Assume function $f(x^1, x^2) = (x^1)^2 + (x^2)^2$ and P^1 interpolation on triangles, on the standard context of straight-sided mesh adaptation, we clearly see f as an isotropic function, and its hessian being

$$H = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}.$$

In the context of standard P^1 adaptation, the optimal mesh is isotropic and its size for having an interpolation error of ϵ^2 is $\mathcal{L} = \epsilon$.

Now, the same function, which is isotropic in Euclidean coordinates (x^1, x^2), is actually anisotropic in polar coordinates. There, $f(r, \theta) = r^2$ is independent of θ. If the mesh "mimics" polar coordinates, we could potentially have a significant gain by going anisotropic.

Now, let us allow the use of quadratic triangles and consider point $(x^1, x^2) = (a, 0)$, we have

$$\kappa_1 = -\frac{1}{\sqrt{(x^1)^2 + (x^2)^2}} \quad \text{and} \quad \kappa_2 = 0$$

and then

$$\mathbf{x}_1(t) = \begin{pmatrix} a - \frac{t^2}{2a} \\ t \end{pmatrix} \quad \text{and} \quad \mathbf{x}_2(t) = \begin{pmatrix} a + t \\ 0 \end{pmatrix}.$$

Along $\mathbf{x}_1(t)$, the function

$$f(\mathbf{x}_1(t)) = t^2 + \left(a - \frac{t^2}{2a}\right)^2 = a^2 + \frac{t^4}{4a^2},$$

is a constant up to order $\mathcal{O}(t^4)$. Assuming a linear interpolation of f, i.e., $\pi^1 f(\mathbf{x}_1(t)) = f(0) + f'(0)t = a^2$, the exact error is bounded by

$$\left|f(\mathbf{x}_1(t)) - \pi^1 f(\mathbf{x}_1(t))\right| = \left|\frac{t^4}{4a^2}\right| < \epsilon^2$$

so that $|t| < \sqrt{2\epsilon a} = \mathcal{L}_1$.

Along the perpendicular curve $\mathbf{x}_2(t)$, we have $f(\mathbf{x}_2(t)) = (a + t)^2$ and

$$\begin{aligned} \left|f(\mathbf{x}_2(t)) - \pi^1 f(\mathbf{x}_2(t))\right| &= \left|(a + t)^2 - a^2 - 2at\right| \\ &= |t^2| < \epsilon^2 \end{aligned}$$

which is independent of a. Thus, the optimal mesh has a constant mesh size $\mathcal{L}_2 = \epsilon$. The optimal mesh is thus anisotropic with an anisotropic scale factor equal to

$$\frac{\mathcal{L}_1}{\mathcal{L}_2} = \sqrt{\frac{2a}{\epsilon}}.$$

So, far away from the origin ($a \gg 0$), the optimal mesh is highly anisotropic.

Now, whenever the exact solution is unknown, but we can still estimate the derivatives of the numerical approximation, an estimator similar to the one explained in the previous section can be used. We have

$$\frac{d^2 f(\mathbf{x}_2(t))}{dt^2} = H_{ij}\dot{x_2}^i \dot{x_2}^j + 2G_i \ddot{x_2}^i = 2.$$

Our proposed estimate leads to

$$|f(\mathbf{x}_2(t)) - \pi^1 f(\mathbf{x}_2(t))| \simeq \frac{\mathcal{L}_2^2}{2} \left| \frac{d^2 f(\mathbf{x}_2(0))}{dt} \right| = \mathcal{L}_2^2 < \epsilon^2$$

which yields $\mathcal{L}_2 < \epsilon$.

The second order derivative along $\mathbf{x}_1(t)$, on the other hand, is computed as

$$\frac{d^2 f(\mathbf{x}_1(t))}{dt^2} = \underbrace{H_{ij}\dot{x_1}^i \dot{x_1}^j}_{2\left(\frac{t}{a}\right)^2+2} + \underbrace{2G_i \ddot{x_1}^i}_{-2+\left(\frac{t}{a}\right)^2} = 3\left(\frac{t}{a}\right)^2$$

and vanishes for $t = 0$. This indicates that our estimate allows choosing $\mathcal{L}_1$ arbitrarily large, which should come as no surprise, since in this case, the “true” anisotropic ratio is large, and probably beyond the capabilities of the mesh generator in terms of anisotropic elements.

5 Mesh adaptation

This section describes our mesh generation and adaptation approach. For illustrating the different steps of the procedure, the following analytic function

$$f(x) = \arctan(10(\sin(3\pi y/2) - 2x))) \tag{3.19}$$

will serve as running example. Figure 3.1 shows the general behavior of $f(x)$ on the unit square as well as the different steps of the procedure.

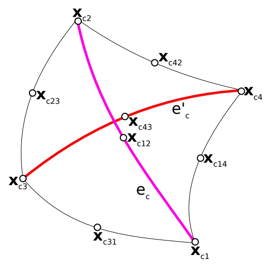


Figure 3.4: Curvilinear edge swap.

5.1 Generation of corner points

Our meshing approach is not the usual one, for which points and triangles are generated at the same time. In our approach, points are generated first, and then connected

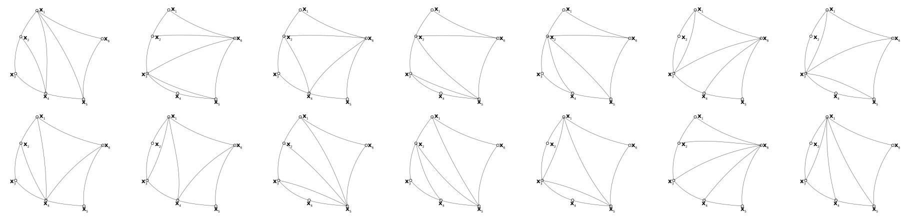


Figure 3.5: All curvilinear triangulations of a 4-cavity. There are 20 distinct triangles $\{0,1,2\}$, $\{0,1,3\}$, $\{0,2,3\}$, $\{1,2,3\}$, $\{0,1,4\}$, $\{0,2,4\}$, $\{1,2,4\}$, $\{0,3,4\}$, $\{1,3,4\}$, $\{2,3,4\}$, $\{0,1,5\}$, $\{0,2,5\}$, $\{1,2,5\}$, $\{0,3,5\}$, $\{1,3,5\}$, $\{2,3,5\}$, $\{0,4,5\}$, $\{1,4,5\}$, $\{2,4,5\}$, $\{3,4,5\}$. in the $T_4 = 14$ triangulations shown in the Figure.

in a second step. Points are generated using the frontal algorithm described in Baudouin et al. (2014); Gu and Yau (2008). In short, one proceeds as follows. The points of the domain boundary are inserted in a queue. Then, the point at the end of the queue is popped out, and 4 neighbor points are created, located at unit distance from it along the parabolas $\pm\mathbf{x}_1$ and $\pm\mathbf{x}_2$, provided they are not too close to already existing points. This algorithm ensures thus that (i) two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are never too close to each other and (ii) that there exist four points $\mathbf{x}_{ij} (j = 1, 2, 3, 4)$ at unit distance from each point, i.e., that can form with it edges whose length is in the range $[0.7, 1.4]$. Figure 3.1 shows the points generated with a metric field based on a target error $\epsilon = 0.02$.

5.2 Generation of a straight-sided anisotropic mesh

The generated points are then connected together using a standard anisotropic mesh generator based on the metric filed $\mathcal{M}(\mathbf{x})$ Dobrzynski (2012). Figure 3.1 shows the resulting straight-sided mesh. The connectivity of this mesh is however not optimal. Yet, it constitutes a good starting point for constructing the unit curvilinear mesh that will further reduce the interpolation error.

5.3 Curving the straight sided mesh

The straight edges of the mesh are now to be transformed into parabolas. This operation is however endowed with the risk of creating invalid P^2 triangles. The following backtracking procedure allows however to provide a provably valid P^2 mesh. At first, invalid triangles are identified using the simple and robust validity criterion described in Johnen et al. (2013). Since straight-sided triangles are always valid, the 3 mid-edge points of a given invalid triangle T are moved simultaneously backwards towards the mid points of the straight edges, until the triangle becomes valid again.

All triangles sharing an edge with T are then checked, and if needed added to the list of invalid triangles. This algorithm always terminates, and as for limit case the recovery of a straight-sided mesh. Yet, in general, mild modifications of the initial curvilinear mesh are sufficient to restore a valid mesh.

It is very important at this point to ensure that one has a valid curvilinear mesh, since all subsequent optimization operations will improve the mesh quality, and thus per definition always preserve the validity of the mesh.

The first mesh improvement method is a basic curvilinear edge swap (see Figure 3.4). Assume two curvilinear triangles $T_{c1}(\mathbf{x}_{c1}, \mathbf{x}_{c2}, \mathbf{x}_{c3})$ and $T_{c2}(\mathbf{x}_{c1}, \mathbf{x}_{c4}, \mathbf{x}_{c2})$ sharing a common edge e_c. Let e'_c be the geodesic between $\mathbf{x}_{c3}$ and $\mathbf{x}_{c4}$. The curvilinear edge swap operator evaluates the opportunity of replacing edge e_c by edge e'_c. Two indicators decide whether the edge swap should be performed:

1. The new curvilinear triangles $T'_{c1}(\mathbf{x}_{c1}, \mathbf{x}_{c4}, \mathbf{x}_{c3})$ and $T'_{c2}(\mathbf{x}_{c4}, \mathbf{x}_{c2}, \mathbf{x}_{c3})$ have to be both valid, according to the criterion based on robust estimations developed in Johnen et al. (2013).

2. The quality of the mesh has to be improved by the edge swap:

$$\begin{aligned} &\min(q_{ct'_{c1}}(\mathcal{M}(\mathbf{x})), q_{ct'_{c2}}(\mathcal{M}(\mathbf{x}))) \\ > &\min(q_{ct_{c1}}(\mathcal{M}(\mathbf{x})), q_{ct_{c2}}(\mathcal{M}(\mathbf{x}))) \end{aligned}$$

   where $q_{ct}(\mathcal{M}(\mathbf{x}))$ is the curvilinear quality measure of triangle T_c with respect to metric field $\mathcal{M}(\mathbf{x})$.

The quality measure used here is a straigthforward extension of the standard quality measure defined in Shewchuk (2002). We define:

$$q_{ct}(\mathcal{M}(\mathbf{x})) = \frac{4\sqrt{3}}{\mathcal{L}(\mathcal{M}(\mathbf{x}))} \int_{t_c} \sqrt{|\mathcal{M}(\mathbf{x})|}\, \mathrm{d}\mathbf{x} \tag{3.20}$$

with

$$\mathcal{L}(\mathcal{M}(\mathbf{x})) = \mathcal{L}^2_{e_{c1}}(\mathcal{M}(\mathbf{x})) + \mathcal{L}^2_{e_{c2}}(\mathcal{M}(\mathbf{x})) + \mathcal{L}^2_{e_{c3}}(\mathcal{M}(\mathbf{x})),$$

where e_{c1}, e_{c2} and e_{c3} are the curvilinear edges of T_c, and $\mathcal{L}_{e_{c1}}$, $\mathcal{L}_{e_{c2}}$ and $\mathcal{L}_{e_{c3}}$ are their geodesic lengths according to the metric field $\mathcal{M}(\mathbf{x})$).

Note that triangle inequality is not necessary verified in Riemannian metrics, i.e., the inequality $\mathcal{L}_{e_{c1}} < \mathcal{L}_{e_{c2}} + \mathcal{L}_{e_{c3}}$ does not always hold. In consequence, the quality measure $q_{ct}(\mathcal{M}(\mathbf{x}))$ may be larger than one. Edges are swapped until a stable configuration is found. Here, we generalize Delaunay triangulation to Geodesic Delaunay triangulation Gu and Yau (2008).

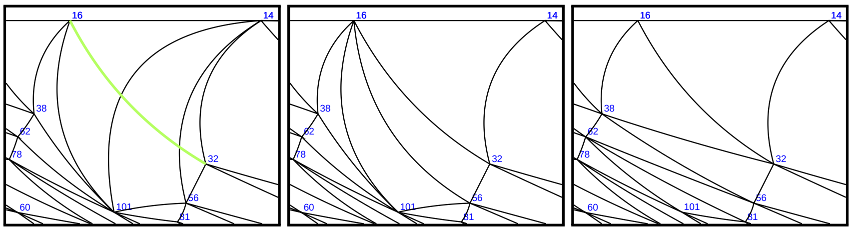


Figure 3.6: Mesh quality improved by CSPR(Curvilinear Small Polygon Reconnection).

5.4 Curvilinear Small Polygon Reconnection

The mesh curving procedure explained in the previous section 5.3 is very similar to the one proposed in Zhang et al. (2018). In this paper, we pointed out that this operation never produces short edges by construction. Figure 3.1 shows the curvilinear mesh generated through basic curved swaps, and one can indeed check that no short edges has been generated. The shortest edge has an adimensional length of $0.837 > 0.7$. Yet long edges exist, with a maximum edge length of $1.82 > 1.4$. Long edges may remain in the mesh, due to the inability of the basic edges swap process to properly connect points that are close with respect to the metric $\mathcal{M}$ without generating invalid P^2 triangles.

This issue is fixed in this paper by introducing a new local mesh optimization operator, called Curvilinear Small Polygon Reconnection (CSPR), that allows overriding local quality maxima to further enhance the overall quality of the P^2 mesh.

The CSPR is the curvilinear version of the small polygon reconnection (SPR) technique, a local mesh modification operator initially proposed by Liu Jianfei (2006). Considering a n-cavity, i.e., a set of n contiguous triangles with no internal vertex and with $n+2$ boundary vertices, the SPR algorithm finds the best triangulation of the n-cavity among all possible triangulations. Catalan numbers $T_n = \dfrac{1}{n+1}\dbinom{2n}{n}$ give the number of possible triangulations of a n-cavity, which can be all found using a branch and bound algorithm Marot et al. (2020). In our work, all triangulations of n-cavities with $n < 10$ have been tabulated, so as to avoid any on-the-fly combinatorial computations of triangulations. Figure 3.5 shows all triangulations of a 4-cavity and the corresponding curved mesh with respect to the running test case metric.

The CSPR requires cavities, which are obtained in two ways:

1. Along unit geodesics: consider all unit geodesic connecting the points of the mesh. Whenever those geodesics are not in the mesh, form cavity with all the triangles that intersect that geodesic.

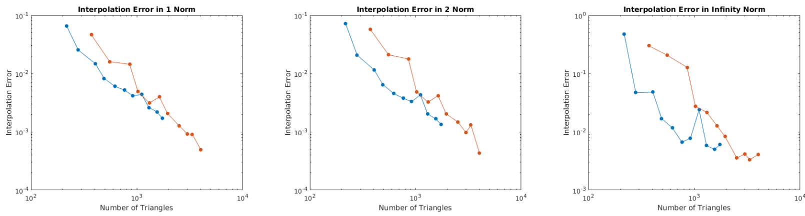


Figure 3.7: 1-norm, 2-norm, ∞-norm interpolation error of analytic function $f(x)$ (3.19), the points 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 11 respectively correspond to mesh size uniform scaling factors $a = 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.2, 1.3, 1.4, 1.5$. The mesh size $\mathcal{L}_a$ with a is $\mathcal{L}_a = \mathcal{L}/a^2$ and $\mathcal{L}$ be the mesh size computed by (3.9).

2. Choose a long edge ($\mathcal{L} > 1.4$) and form cavity with its two adjacent curvilinear triangles. If there are other long edges in those triangles, repeat the process.

In the process, if an internal edge has the same two points as before, we will keep the midpoint as before, otherwise, we compute a new midpoint; for all boundary edges, we always keep the same midpoint as before. For each possible swapped configuration, if the worst quality of all the elements is improved, the configuration is kept and will be in the new mesh unless another swapped configuration provides a better quality improvement. It works as:

i. The new curvilinear triangles $t'_{c1}(\mathbf{x}_{c_1}, \mathbf{x}_{c_2}, \mathbf{x}_{c_3})$, $t'_{c_2}(\mathbf{x}_{c_2}, \mathbf{x}_{c_3}, \mathbf{x}_{c_4})$, ..., $t'_{cn}(\mathbf{x}_{c_n}, \mathbf{x}_{c_{n+1}}, \mathbf{x}_{c_{n+2}})$ have to be all valid. The validity criterion that is used is the same one for curvilinear edges swap in Section 5.3.

ii. The quality of the mesh has to be improved by the reconnection:

$$\begin{aligned} &\min(q_{ct'_{c_1}}(\mathcal{M}(\mathbf{x})), q_{ct'_{c_2}}(\mathcal{M}(\mathbf{x})), \ldots, q_{ct'_{c_n}}(\mathcal{M}(\mathbf{x}))) \\ &> \min(q_{ct_{c_1}}(\mathcal{M}(\mathbf{x})), q_{ct_{c_2}}(\mathcal{M}(\mathbf{x})), \ldots, q_{ct'_{c_n}}(\mathcal{M}(\mathbf{x}))) \end{aligned}$$

where $q_{ct}(\mathcal{M}(\mathbf{x}))$ is the curvilinear quality measure of triangle t_c with respect to metric field $\mathcal{M}(\mathbf{x})$ - the same one for curvilinear edges swap in Section 5.3.

Figure 3.6 illustrates the CSPR applied to the running example. A first 3-cavity is constructed with all elements crossing the green geodesic between points 16 and 32. This cavity is remeshed using CSPR, producing a mesh that is better, but still not optimal. Indeed, a shorter curvilinear edge still exists between points 38 and 32 that is not in the mesh. A 5-cavity is then constructed around this edge, and remeshed to eventually produce an optimal mesh.

6 | Interpolation error

We complete our analysis by running our algorithm using the same function (3.19), but now defined on the geometrically non-trivial domain depicted in Figure 3.8, and by analyzing whether or not the curvilinear mesh adaptation reduces the interpolation error.

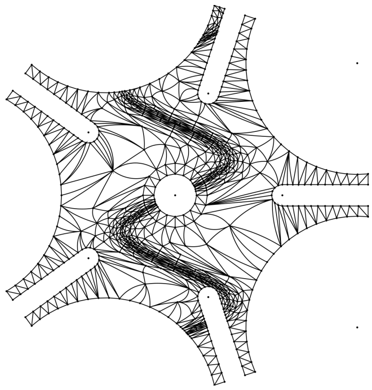


Figure 3.8: Curvilinear mesh adaptation based on function (3.19) on a mechanical part.

In order to compare the level of accuracy reached by meshes generated with our method or by meshes generated with a conventional straight-sided anisotropic mesh adaptation, the ∞-norm, the 1-norm and the 2-norm of the quadratic interpolation error are computed with both meshes, and plotted against the number of triangles of the mesh. Eleven different meshes were generated, by varying a global mesh size uniform scaling parameter a between 0.4 and 1.5. The convergence curves are shown in Figure 3.7. The numerical results show that our adapted curvilinear meshes allow a significant error reduction of about 50% with respect to straight-sided P^2 meshes of the same size. This mainly attributes to the numerical solutions of middle points of curvilinear meshes is more exact than numerical solutions of middle points of straight-sided P^2 meshes. Figure 3.9 shows that the case $a = 1.2$ (8th point) is an outlier. As shown in Figure 3.9, this is due to the persistence of long edges that the CSPR has not been able to remove.

7 | Conclusions

Building on the paper Zhang et al. (2018), which provided an embryo of solution to the problem of curvilinear mesh adaptation, we have extended our work to the true P^2 mesh adaptation, i.e., the adaptation of a P^2 mesh to a given function $f(x, y)$. A

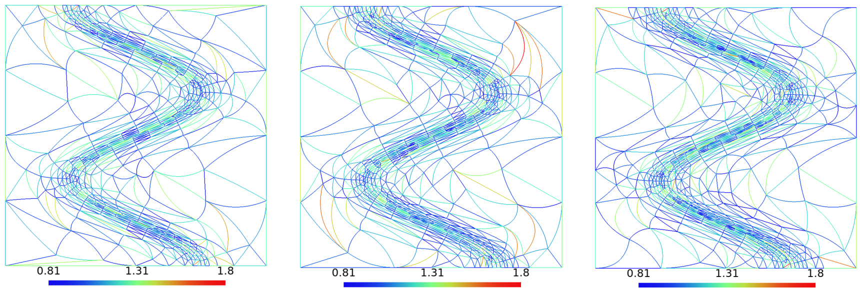


Figure 3.9: Adapted curvilinear meshes for computing interpolation error 1-norm, 2-norm, ∞-norm at points 7-left, 8-middle, and 9-right(the same points 7, 8, and 9 in Figure 3.7 of curvilinear mesh).

methodology for building a specific metric field suited for this P^2 mesh adaptation has been explained, and a new local mesh modification operator called Curvilinear Small Polygon Reconnection (CSPR) has been developed to build the optimal curvilinear triangulation of a given curvilinear polygon/cavity. This paper focuses, for didactically reasons, on a single running example function (3.19), but the methodology has been applied successfully to other functions, with similar conclusions. The method is however not yet able to build a unit mesh in all cases. Figure 3.9 shows that, long edges may sometime fail to be eliminated by the CSPR, leaving an eventually interpolation error that is not better than a P^1 meshes. Yet, in most of cases, a clear reduction of the interpolation error is observed when curving the elements.

For the failure of the method to build a unit mesh in all cases, it maybe attribute to: the adimensional length used to insert points is 1.0 and the adimensional length used to check if two points are two close is 0.7, this may result the geodesic between two points is about 1.7, and this maybe be fixed by change the the adimensional length used to insert points to between 1.0 and 0.7.

Our next move will be to replace the analytical functions with high-order finite element solutions. We foresee new issues there, like the accurate computation of the third order derivatives of finite element solutions. For 3D, the approach is straightforward by constructing a 3D metric field and computing 3D geodesics, it maybe time cost but still in control.

CHAPTER 4

Curvilinear mesh adaptation for a numerical solution

Up to now, we only have considered analytical functions $f(x, y)$ for adapting the meshes. This essentially means that we can compute all derivatives of f without making any error. In this Chapter, we go forward and use computational fluid dynamics (CFD) results to create adapted meshes. The CFD code that is used is the research code of Arthur Bawin and we thank him warmly for his help.

The CFD code that is used here is a high order finite element code. The finite element approximation that is used is continuous. Standard Lagrange shape functions on (possibly) curvilinear triangles are used to approximate velocity and pressure field.

1 | Computing derivatives

Assume a high order mesh with n nodes and a finite element solution of the form

$$u_h(x,y) = \sum_{i=1}^{n} \phi_i(x,y) u_i$$

where ϕ_i are high order nodal shape functions and u_i nodal values. We call S_h the high order finite element space $u_h \in S_h$. Zhang and Naga introduced in Zhang and Naga (2005) a new gradient recovery operator $G_h : S_h \to S_h \times S_h$ that computes a high order version of the two components of $g_x(x,y) = \partial_x u_h$ and $g_y(x,y) = \partial_y u_h$ of the gradient of u_h in such a way that $g_x \in S_h$ and $g_y \in S_h$.

Second order derivatives are computed using G_h applied to g_x and g_y. Higher order derivatives are then computed by applying G_h to higher and higher derivatives. We have to confess that the quality of the recovery process degrades when G_h is applied several times. Yet, the third order derivatives that are required here seem to be sufficiently smooth to compute metrics that are not too oscillatory.

2 | Vortex in a box

The vortex in a box is a standard case that tests the ability of the scheme to accurately resolve thin filaments on the scale of the mesh which can occur in stretching and tearing flows. Assume a stream function of the form:

$$\Psi(x,y) = \frac{1}{\pi} \sin^2(\pi x) \sin^2(\pi y).$$

Stream function Ψ allows to define a divergence free velocity field $\mathbf{u} = (u_x, u_y)$ that is computed as:

$$u_x = \frac{\partial \Psi(x,y)}{\partial y} = \sin(2\pi y) \sin^2(\pi x)$$

$$u_y = -\frac{\partial \Psi(x,y)}{\partial x} = -\sin(2\pi x) \sin^2(\pi y)$$

The box we are dealing with is a square of size $[0,1] \times [0,1]$. The evolution equation that is computed here is pure advection. Assume a function $\phi(\mathbf{x},t)$ with $\phi(\mathbf{x},0)$ that is the squared distance function to a disk of radius 0.15 placed at $(0.75, 0.75)$:

$$\phi(\mathbf{x},0) = (x-0.75)^2 + (y-0.75)^2 - 0.15^2.$$

We solve

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0.$$

The flow satisfies $u_x = u_y$ on the boundaries of the unit square, so ϕ remains constant and equal to its initial value on the boundary.

We have used Arthur Bawin's high order code (quadratic finite elements) and have computed high order curvilinear meshes at different timesteps. Figure 4.1 shows that the resulting function ϕ stretches out the circle into a very long, thin filaments.

We see in Figure 4.1 that our high order mesh procedure has the ability of capturing the highly anisotropic and curvilinear features of the solution in a quite impressive manner. To our best knowledge, this is the first time one shows an adapted curvilinear high order mesh applied to a finite element solution. We see that our point insertion procedure was able to produce meshes that are structured in good portions of the domain. The story is not over: our methodology has its drawbacks. Sole elements are unnecessarily curved, especially when the error is very low (see for example the middle of the square in Figure 4.1). This is essentially due to the fact that there is not much of an error difference between possible paths that define a curvilinear edge when the error is very low. This could be prevented by giving preference to straight sided edges when error are small.

3 | Lid driven cavity

The lid-driven cavity is a standard test case that serves as a benchmark for testing numerical methods. A comprehensive review is provided in Bruneau and Saad (2006) Kuhlmann and Romanò (2019). We are dealing with a square cavity consisting of three rigid walls with no-slip conditions and a upper lid moving with a tangential unit velocity. The lower left corner has a reference static pressure of 0. We are interested in the velocity and pressure distribution for a Reynolds number of 400. We of course use here a polynomial order of $k = 2$.

At such a low Reynolds number of 400, Navier-Stokes's equations provide a steady state solution. We have thus computed the solution on a first mesh and have done the curvilinear adaptation in a second time. Figure 4.2 shows the two components of the velocity as well as some of the computed derivatives of the velocity. We see that high order derivatives are noisier and noisier when the order of the derivative increases. The mesh was initially refined anisotropically using method advocated in Hecht and Kuate (2014). Derivatives were computed on that initially adapted mesh and a curvilinear mesh was subsequently produced (see Figure 4.3). Here, the mesh is less impressive than the one of the vortex in the box. Curvilinear features are not so present in that flow. Yet, we see that the mesh was able to smoothly follows the shape of the main vortex and that it was able to capture the upper boundary layer in a pretty structured fashion. We believe that, even though those results are very preliminary, that our methodology for generating high order curvilinear meshes is promising and will definitively be able to tackle more complex problems in a near future.

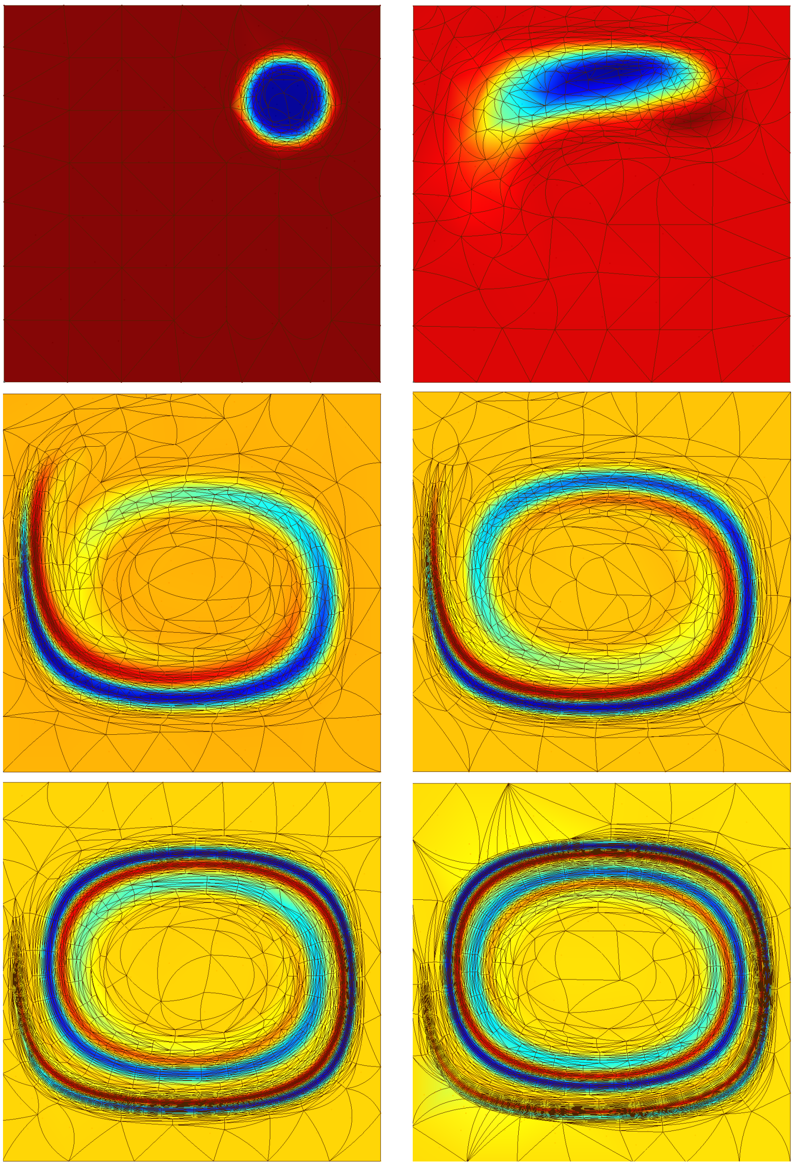


Figure 4.1: Function ϕ at increasing time steps together with the underlying curvilinear mesh. All meshes were computed for a target error of $\epsilon = 0.01$.

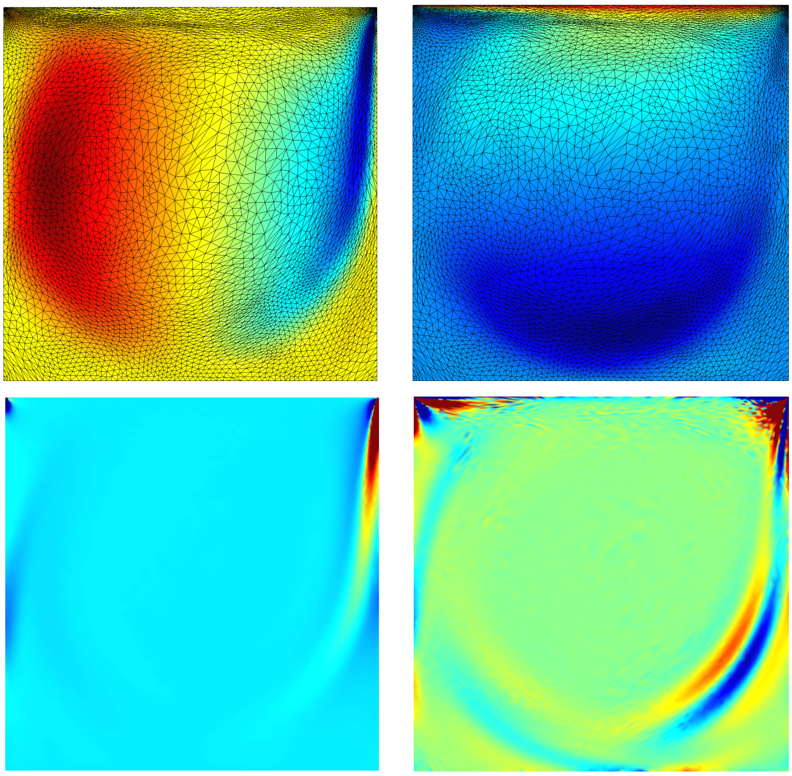


Figure 4.2: The two components of the velocity for the lid driven cavity (top figures) and high order derivatives $\frac{\partial^2 u_y}{\partial x^2}$ and $\frac{\partial^3 u_y}{\partial x \partial y^2}$.

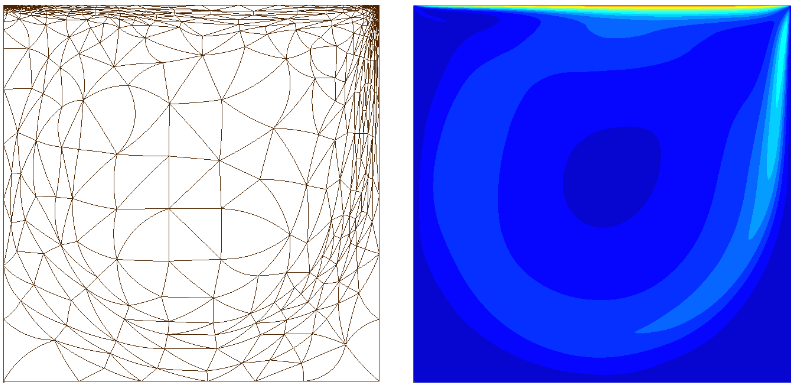


Figure 4.3: Adapted mesh for the lid driven cavity as well as the norm of the velocity $\sqrt{u_x^2 + u_y^2}$.

CHAPTER 5

CONCLUSION

A research journey As a pioneer, in the research journey of curvilinear mesh generation and adaptation, we start from summary ideally the research project as a simple yet fundamental question: assume a unit square

$$\Omega = \{(x^1, x^2) \in [0,1] \times [0,1]\}$$

and a smooth function $f(x^1, x^2)$ defined on the square, and consider a mesh $\mathcal{T}$ made of P^2 triangles that exactly covers the square, how can we compute the mesh $\mathcal{T}$ that minimizes the interpolation error $\|\Pi f - f\|_\Omega$. Here, Π is the nodal interpolation of f on the mesh Ern and Guermond (2013). This problem is the problem of curvilinear mesh adaptation.

We state the problem as to build a unit curvilinear mesh, i.e. build a mesh with unit edge lengths that are possibly curvilinear.

Our goal is:

$\boxed{\text{FINITE ELEMENT SOLUTION} \rightarrow \text{UNIT CURVILINEAR MESH}}$

We solve the problem in three stages of increasing complexity:

1. Stage 1 - Chapter 2

   Input metric field g(x,y) $\rightarrow$ UCM

2. Stage 2 - Chapter 3

   Input analytical function f(x,y) $\rightarrow$ g(x,y) $\rightarrow$ UCM

3. Stage 3 - Chapter 4

   Input FEM solution $\rightarrow$ g(x,y) $\rightarrow$ UCM

In this first stage, a Riemannian metric field $g_{ij}(x^1, x^2)$ is defined on the unit square. This metric field is supposed to be the result of the error estimation. Our aim is thus to build a unit P^2 mesh with respect to that metric. A discrete mesh $\mathcal{T}$ of a domain Ω is a unit mesh with respect to Riemannian metric space $\boldsymbol{g}(x^1, x^2)$ if all its elements are quasi-unit. More specifically, a curvilinear triangle t defined by its list of edges $e_i,\ i = 1, 2, 3$ is said to be quasi-unit if all its adimensional edges lengths $\mathcal{L}_{e_i} \in [0.7, 1.4]$[1]. Generating unit straight-sided meshes is a problem that has been largely studied, both in the theoretical point of view and on the application point of view Frey and Alauzet (2005). Here, our aim is to allow edges to become curved, leading to unit meshes that would potentially contain way less triangles. Our mesh generation technique essentially relies on the computation of the shortest parabola between two points and on a unit-size parabola starting in a given direction. In Section 2, standard notions of geodesics in Riemann spaces are briefly exposed. Algorithms that compute geodesic parabolas are explained as well. The mesh generation approach that we advocate is in two steps. We first generate the points in a frontal fashion Baudouin et al. (2014). In that process, we ensure that (i) two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are never too close to each other and (ii) that there exist four points $\mathbf{x}_{ij},\ j = 1, \ldots, 4$ in the vicinity of each point $\mathbf{x}_i$ that are not too far to $\mathbf{x}_i$ i.e. that can form edges in the prescribed range $[0.7, 1.4]$. Then, points are connected in a very standard "isotropic" fashion. The mesh is subsequently modified using curvilinear edge swaps in order to form the desired unit mesh. A curvilinear mesh quality criterion is proposed that allow to drive the edge swapping process.

The method guarantees two important properties in the final mesh:

[1]This range is not arbitrary. When a long edge of size 1.4 is split, it should not become a short edge. Other authors choose $[\sqrt{2}/2, \sqrt{2}]$

1. Generated meshes are valid. This important property is due to the fact that P2 meshes are valid at any point of the algorithm. The initial mesh is curved along geodesic. A backtracking step is applied to ensure that every triangle of the mesh is valid. Then, edge swaps are only applied if elements are valid. Note that the validity criterion that is used is robust.

2. No short edges will be exist in the mesh. A spatial search procedure is used for ensuring that any point that is inserted is not to close in the sense of geodesics than any other point.

In the second stage, our work is being extended to true adaptation i.e. adapting a mesh to a given function $f(x^1, x^2)$. Even though metrics are still the right tool for driving mesh adaptation at higher orders, basing $\boldsymbol{g}$ on Hessians of f is not correct anymore for higher orders of approximation. We build metric fields that are suited for high order. In this approach, curvature is not only used to match curved boundaries but also to capture features of the interpolated solutions, and it results in meshes that would not have been achievable by simply curving *a posteriori* a straight-sided mesh. We proceed as follows. Starting with a smooth function $f(x, y)$, a metric field, based on f and its derivatives up to order 3, is constructed. A unit P^2 mesh is then generated, with edges within an adimensional length range of $[0.7, 1.4]$ with respect to this metric field. Points are then spawned in such a way that their geodesic distance corresponds to edges of unit size, and these points are then connected in a standard *isotropic* fashion. A curvilinear mesh quality criterion is then proposed to drive the mesh optimization process. The triangulation is subsequently modified using straight-sided edge swap, straight-sided edge curving, curvilinear edge swap and Curvilinear Small Polygon Reconnection (CSPR) to form the desired unit mesh. A unit curvilinear mesh containing only valid "Geodesic Delaunay triangles" is obtained this way. A number of application examples are presented in order to demonstrate the capabilities of the mesh adaptation procedure. The resulting adapted meshes allow, most of the times, a significant reduction of the approximation error compared with straight-sided P^2 meshes of the same density.

We essentially tackle two additional problems that allow to move forward to "true anisotropic curvilinear mesh adaptation" (i.e., optimizing a mesh based on a high-order finite element solution):

1. P^2 error estimates that are currently used in the literature are based on the implicit hypothesis that underlying meshes are straight-sided. Assuming a function $f(x, y) \in C^3$, we construct a metric field that does not assume mesh edges to be straight-sided.

2. The mesh generation procedure that we use for generating the curvilinear meshes is based on the one of Zhang et al. (2018). Yet, we show that using edge swaps only does not always allow to reach a unit mesh. We propose a

more general operator - curvilinear small polygon reconnection - that allows to reconnect points in a wider range and generate P^2 unit meshes very robustly.

In the third stage, we go forward and use computational fluid dynamics (CFD) results to create adapted meshes. The CFD code that is used here is a high order finite element code. The finite element approximation that is used is continuous. Standard Lagrange shape functions on (possibly) curvilinear triangles are used to approximate velocity and pressure field.

General conclusions Many physical problems shows a anisotropic feature -their solutions change more significantly in one direction than the others. For these problems with anisotropic solution features, a properly anisotropic mesh will be able to improvement significantly in accuracy and efficiency - to efficiently improve the ratio between solution accuracy and the number of degrees of freedom. Based on and taking advantage of well-defined Riemannian differential geometry, metric-based mesh generation and adaptation is to generate a quasi-uniform mesh - specifying the shape, size, orientation of elements with respect to a metric filed and all geometric operations are performed in the Riemannian metric space. Thus it is crucial to construct an appropriate metric field in the Riemannian metric space and an adapted anisotropic mesh in a Riemannian metric space is simply same than a uniform mesh in the Euclidean space. The main advantage of this approach is that the adaptation mesh is obtained directly at the mesh generation step rather than doing a modification of an existing mesh by enrichment or moving points.

In this thesis, we propose a new framework for the generation and adaptation of unit curvilinear P^2 meshes in dimension 2. In this approach, curvature is not only used to match curved boundaries but also to capture features of the interpolated solutions, and it results in meshes that would not have been achievable by simply curving *a posteriori* a straight-sided mesh.

We essentially tackle *three fundamental problems*:

1. What is the geometrical structure of the interpolation error in the P^2 case?
2. How can we relate this structure to the geometry/shape of a P^2 triangle?
3. How can we build a mesh made of optimal P^2 triangles?

To solve all these problems, we propose *a new framework* of curvilinear mesh adaptation in this thesis:

1. Metric field construction

2. Generation of points
   Point sampling on the boundary
   Point sampling in the domain
3. Straight-sided mesh generation and adaptation
   Triangulation
   Straight-sided edges swap
4. Curvilinear mesh generation and adaptation
   Straight-sided edges curving
   Curvilinear edges swap
   Curvilinear Small Polygon Reconnection

In the new framework, we develop seven important *algorithms*:

1. *Metric field construction*
   - *Toy metric field*
   - *Analytic function metric field*
   - *Numerical solution metric field*
2. *Aligned physical-space-based geodesic points*
3. *Aligned metric-based geodesic points*
4. *Generation of a straight-sided anisotropic mesh*
5. *Ensure-valid curving*
6. *Ensure-valid curvilinear edge swap*
7. *Curvilinear Small Polygon Reconnection (CSPR)*

We develop two important *mathematical tools*:

1. *Geodesic of given length*
2. *Geodesic between two given points*

The overall methodology is implemented in a $2D$ process that is fully extendable to $3D$.

A new framework of curvilinear mesh adaptation in $3D$:

1. Metric field construction in $3D$

$$
\begin{aligned}
&g(x^1, x^2, x^3)\\
&= \begin{pmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{pmatrix}\\
&= \begin{pmatrix} v_{1x} & v_{2x} & v_{3x} \\ v_{1y} & v_{2y} & v_{3y} \\ v_{1z} & v_{2z} & v_{3z} \end{pmatrix} \begin{pmatrix} h_1^2(\mathbf{x}) & 0 & 0 \\ 0 & h_2^2(\mathbf{x}) & 0 \\ 0 & 0 & h_3^2(\mathbf{x}) \end{pmatrix} \begin{pmatrix} v_{1x} & v_{1y} & v_{1z} \\ v_{2x} & v_{2y} & v_{2z} \\ v_{3x} & v_{3y} & v_{3z} \end{pmatrix}
\end{aligned}
$$

2. Generation of points in $3D$
   Point sampling on the $1D$ boundary line
   Point sampling on the $2D$ boundary surface
   Point sampling in the $3D$ domain
3. Straight-sided mesh generation and adaptation in $3D$
   Triangulation/tetrahedronation in $3D$
   Straight-sided edges swap in $3D$
   Straight-sided faces swap in $3D$
4. Curvilinear mesh generation and adaptation in $3D$
   Straight-sided edges curving in $3D$
   Curvilinear edges swap in $3D$
   Curvilinear faces swap in $3D$
   Curvilinear Small Polygon/Polyhedron Reconnection in $3D$

To extend two important *mathematical tools* in $3D$:

1. *Geodesic of given length in $3D$*

   In a $3D$ Riemannian metric space, given one point $\mathbf{x} = \{x^1, x^2, x^3\}$ and one direction $\boldsymbol{v} = \{v^1, v^2, v^3\}$, to find the geodesic starting from the point $\mathbf{x}$ in this direction $\boldsymbol{v}$, and the length of geodesic equal to a given length $\mathcal{L}_C$. We formulate the problem of finding geodesics in a Riemannian metric space as the problem of solving a system of ordinary differential equations, i.e, if we know two initial conditions, a start point $\mathbf{x} = \{x^1, x^2, x^3\}$ and an initial direction $\boldsymbol{v} = \{v^1, v^2, v^3\}$, we will be able to find geodesic $\mathcal{C}$ by integrating with those two initial conditions by taking $\mathbf{x} = \{x^1, x^2, x^3\}$ as the initial position and $\boldsymbol{v} = \{v^1, v^2, v^3\}$ as the initial velocity.

2. *Geodesic between two given points in $3D$*

   We attempt to compute a geodesic from a point $\mathbf{x}_{start}\left(x_1^1, x_1^2, x_1^3\right)$ to another point $\mathbf{x}_{end}\left(x_2^1, x_2^2, x_2^3\right)$, for $\forall \mathbf{x}_{start}, \mathbf{x}_{end} \in R^3$. In other words, given two points $\mathbf{x}_{start}$ and $\mathbf{x}_{end}$ in a $3D$ Riemannian metric space, the task is to find a geodesic joining $\mathbf{x}_{start}$ and $\mathbf{x}_{end}$. Equivalently, we want to find a path C whose coordinates satisfy the differential equations

$$\frac{d^2x^i}{dt^2} + \Gamma^i_{jkl}\frac{dx^j}{dt}\frac{dx^k}{dt}\frac{dx^l}{dt} = 0.$$

   Assuming the geodesic to be a parabola, with

$$\dot{\mathbf{x}} = \left((\mathbf{x}_{end} - \mathbf{x}_{start}) - \frac{\ddot{\mathbf{x}}}{2}\right)$$

   we have

$$\mathbf{x}(t) = \mathbf{x}_{start} + \left((\mathbf{x}_{end} - \mathbf{x}_{start}) - \frac{\ddot{\mathbf{x}}}{2}\right) t + \ddot{\mathbf{x}}\frac{t^2}{2}.$$

   In order to simplify even more our formulation, we assume that the midpoint $\mathbf{x}_{mid}(x^1, x^2, x^3)$ on the geodesic parabola C_{12} between $\mathbf{x}_{strat}(x^1, x^2, x^3)$ and $\mathbf{x}_{end}(x^1, x^2, x^3)$ is located on the orthogonal bisector plane $S_3(x^1, x^2, x^3)$ of segment $\mathbf{x}_{strat}\mathbf{x}_{end}$.

   The point $\mathbf{x}_{mid}(x^1, x^2)$ is computed by minimizing the length of that parabola

$$\begin{aligned}\mathbf{x}_{mid}(x^1, x^2, x^3) &= arg\min \mathcal{L}_C \\ &= \int_0^1 \sqrt{g_{ijk}(t)\,\mathrm{d}x^i(t)\,\mathrm{d}x^j(t)\,\mathrm{d}x^k(t)}\,, \quad i, j, k = 1, 2, 3\end{aligned}$$

Perspectives Numerical simulation has become an integral part of the design process in science and engineering Alauzet and Loseille (2016). The computational pipeline emphasizes the central role of meshing in scientific computing:

$$CAD \rightarrow MESH \rightarrow SOLVER \rightarrow VISUALIZATION/ANALYSIS.$$

From a rational point of view, we want to completely tract the problems, but in the field of computational science, being able to predict numerically all the features of complex physical phenomena with complex geometries remains an unachieved goal - such an ability would bring opportunities in a better understanding of complex physical phenomena. In the context of scientific computing, the mesh is used as a discrete support for the considered numerical methods. As a consequence, the mesh greatly impacts the efficiency, the stability and the accuracy of numerical methods. The goal of anisotropic mesh adaptation is to generate a mesh which fits the application and the numerical scheme in order to achieve the best possible solution. It is thus an active field of research which is progressing continuously.

The development of high-order numerical technologies for engineering analysis has been underway for many years now. Compared to standard second-order-accurate numerical schemes, high-order methods exhibit superior efficiency in problems with high resolution requirements, because they reach the required accuracy with much coarser grids. In many practical applications, high-order methods - show high accuracy with a lower computational cost and converge exponentially with the order of the approximating polynomial - have attracted considerable attention.

In finite element methods, high-order methods can provide a higher rate of solution convergence than their lower-order methods, have the potential to achieve higher accuracy with reduced computational cost; but a piecewise linear mesh at curved domains may introduce inaccuracies in shape, finite element solutions, and surface normal vectors (which are important for computer graphics); thus, some applications require curvilinear mesh to match a curved domain. Alignment and orthogonality of elements can be highly desired near the boundaries of the geometry, it is needed to curve to high order meshes and ensure that the resulting mesh is valid and matches the boundary geometry. To generate a high-quality mesh is the first step in numerical simulations of *PDEs*. In order to produce good quality meshes, the method to generate high-order curvilinear mesh - the curvilinear elements well approximate the curved boundaries - is needed, i.e. curved domains are approximated with elements whose faces are described by parametrized quadratic, cubic, bilinear, or trilinear patches. It is also need to match the feature of numerical simulations of *PDEs* which always be complex and curved.

Since the end of last century, metric-based anisotropic mesh adaptation have proved the efficiency to improve the ratio between solution accuracy and the number of degrees of freedom on many real-life problems; now, curvilinear mesh generation and adaptation have also proved a better efficiency.

But we still face both theoretical and practical difficulties to attain efficient adaptive methods for numerical computations and to assess adaptive computations, such as the adaptive detection and capturing of all the features of the solution, optimal mesh; for curvilinear mesh, we face more difficulties, such as adaptive all the curved features of the solution.

This means the story of mesh generation need and will continue, and its substory - curvilinear mesh generation and adaptation - just began and will become an important and excellent in the near future. Base on our work until now, it can be continue in the line:

For the problems with curved anisotropic features, curvilinear anisotropic mesh adaptation - improves computational efficiency and enhances the solution accuracy - adapts the mesh size, shape and orientation and involves three key factors: error estimates, metric construction and curvilinear anisotropic mesh generation.

1. *Derivatives recovery*

The kth polynomials interpolation error depends on the $k+1$th derivatives of the interpolated function u, and accurately understanding the anisotropic behavior of $\bigtriangledown^{k+1}u$ To handle k (an arbitrary integer) high order mesh adaptationin, - it is an extension of the P_1 adaptation metric based methods to the case of the P_k - the main issue lies in extend the interpretation of the $k+1$ differential of the solution u into an appropriate metric filed in the Riemannian metric space, with respect to which we then generate a unit mesh. Since only the discrete solution $u_h$is known, we never be able to use the exact derivatives of the function f, only its point-wise values are used to recover the differential form of order $k+1$. Once the numerical $k+1$ differential form of the smooth solution f is recovered, we can use it to construct the optimal metric $M(k)$.

For recovery procedure, L_2-projection operator can be applied to these hessians Clément (1975); Zienkiewicz and Zhu (1992a,b), double L_2-projection, the least square method or eventually the Green formula based approach Frey and Alauzet (2005) can also be applied; Buscaglia et al. (1998) discussed Hessian-based adaptivity (HBA) based on the recovery of the Hessian of the exact solution, reported several recent advances in HBA methods and extended to $3D$, and discussed the justification and better boundary treatment of Hessian recovery; Picasso et al. (2010, 2011) studied numerical methods to approach the second order derivatives of the exact solution f using the piece-wise linear finite element approximation f_h. In the framework of the Laplace problem and the Poisson problem with continuous, piece-wise linear finite elements, the convergence of numerical methods to approach second derivatives has been investigated. Numerical results show that the quality of the results of all methods considered strongly linked to the mesh topology and that no convergence can be insured in general in $2D$ and $3D$, but there is no blow up and the values obtained are probably accurate enough in order to be used as refinement or coarsening criteria in adaptive algorithm. Rusinkiewicz (2004) presents a finite-differences approach for estimating curvatures on irregular triangle meshes that may be thought of as an extension of a common method for estimating per-vertex normals, generalizes naturally to computing derivatives of curvature and higher-order surface differentials and is efficient in space and time. The results in significantly fewer outlier estimates while more broadly offering accuracy comparable to existing methods. Zhang and Naga introduced in Zhang and Naga (2005) a new gradient recovery operator $G_h : S_h \to S_h \times S_h$ that computes a high order version of the two components of $g_x(x,y) = \partial_x u_h$ and $g_y(x,y) = \partial_y u_h$ of the gradient of u_h in such a way that $g_x \in S_h$ and $g_y \in S_h$. We still need to find better derivatives recovery methods in order to have an exact derivatives of numerical solution.

2. *Error estimation*

The aim of mesh adaptation is to seek for the optimal continuous mesh minimizing the interpolation error for a given function (i.e, a given analytic metric field, a given analytic function, a given numerical solution of PDEs), and thus are relative to the prediction of the interpolation error both in magnitude and rate of convergence. In a general form, the problem is to find the optimal mesh that minimizes the error for a given function and require the simultaneous optimization of both the mesh geometry and topology, and thus is a global combinatorial problem cannot be considered practically and need to simply to approximate the solution. A common simplification is to perform a local analysis of the error instead of considering the global problem, such as: Formaggia and Perotto (2001); Picasso (2003); Frey and Alauzet (2005); Huang (2005) derive a local bound of the interpolation error and transform it into a metric-based estimate; Cao (2005) derives the optimal element shape. A local problem as they act in the vicinity of an element, if consider directly the minimization on a discrete mesh, such error minimization is equivalent to a steepest descent algorithm that converges only to a local minimum with poor convergence properties. Contrary to discrete-based study, the continuous formulation succeeds in solving globally the optimal interpolation error problem by using powerful mathematical tools such as calculus of variations. Unicity of the solution along with an optimal bound of the interpolation error are deduced from this analysis. It is critical to definite a good local error and a good global error in a numerical solution, thus we need to research different error models respect to different problems.

3. *Problem-based metric construction*

   Based on and taking advantage of well-defined Riemannian differential geometry, metric-based mesh generation and adaptation is to generate a quasi-uniform mesh - specifying the shape, size, orientation of elements with respect to a metric filed and all geometric operations are performed in the Riemannian metric space. Thus it is crucial to construct an appropriate metric field in the Riemannian metric space.

   From a theoretical point of view, a continuous metric field could be a direct way to represent the underlying Riemannian space, in which the measurement of length varies at each point and direction. Such a metric tensor is often given on a background mesh, either prescribed by the user or chosen as the mesh from the previous iteration in an adaptive solver.

   In mesh generation and finite element analysis, measuring their anisotropic behavior is the key for anisotropic mesh design and refinement Apel (1999a) and in order to determine an ideal element orientation, shape and size, one needs to define the *principal direction* and the *strength* to characterize the anisotropic behavior of the derivative tensors; in order to minimize linear interpolation error, the eigenvalues and eigenvectors of Hessian matrices can be used to determine the element aspect ratio and mesh alignment direction for anisotropic mesh generation or refinement, e.g., Borouchaki et al. (1997a,b); Cao (2005);

D'Azevedo and Simpson (1991); Formaggia and Perotto (2001); Habashi et al. (2000); Ait-Ali-Yahia et al. (2002); Dompierre et al. (2002); Nadler (1985); Rippa (1992); Shewchuk (2002); Simpson (1994); in the case of quadratic or higher-order interpolations, the error is determined by the third-order or higher-order partial derivatives of the interpolated functions that very few work has been done, such as Cao (2008) developed a method to measure the orientation and anisotropic ratio of the higher-order derivative tensors for two-dimensional functions. The technique is based on decomposing the homogeneous polynomials for directional derivatives into the product of linear and non-negative quadratic polynomials, then the anisotropic measure is defined by the directions of the lines and ellipses corresponding to those factors, an interpolation error estimate is further derived on anisotropic meshes that are quasi-uniform under given metrics, and optimal mesh metrics can be identified to minimize the error bound in various norms.

In practice, the most well established error analysis enables to calculate a metric tensor on an element basis, but it is also able to to calculate a metric tensor on an edge basis Duan Wang and Yan (2010); Coupez (2011) or an node basis, such as reference Coupez (2011) presents propose to build a metric field directly at the nodes of the mesh for a direct use in the meshing tools: the unit mesh metric is defined and well justified on a node basis, by using the statistical concept of length distribution tensors; the interpolation error analysis is performed on the projected approximate scalar field along the edges; the error estimate is established on each edge whatever the dimension is. It enables to calculate a stretching factor providing a new edge length distribution, its associated tensor and the corresponding metric and the optimal stretching factor field is obtained by solving an optimization problem under the constraint of a fixed number of edges in the mesh. For defining the metric tensor, the procedure developed on interpolation error also be applied to other types of error estimates, such as a posteriori error estimates and estimates for truncation error. It is also a good idea to construct a metric field with respect to problem: feature-based: to derive the best mesh to compute the characteristics of a given sensor; goal-oriented: to derive the best mesh to observe a given scalar functional.

4. *Mesh optimization strategies*

   It is well known the quality of meshes to affect both the efficiency and the accuracy of the numerical solution of application problems, especially for problems with complex geometric domain. Freitag and Ollivier-Gooch (2000) shows that the cost of mesh improvement is significantly less than the cost of solving the problem on a poorer quality mesh. To provide the applicability of the method to computational complex problems, great efforts have been made to ensure a good configuration of nodes and elements in all of mesh generating methods Li et al. (2001); Jianfei (2003); Chung et al. (2003); Lo and Wang (2005), lots of works Joe (1991a,b, 1995); Dari and Buscaglia (1994); Zavattieri et al. (1996); Freitag and Ollivier-Gooch (1997); Lo (1997); Freitag and Plassmann (2000); Sun and Liu (2003) and it still needed to improve the quality of mesh further.

Unlike some studies Kennon and Dulikravich (1986); Zhang and Trépanier (1994); Lo (1997); Jianfei et al. (2006) points out - in sense of quality, the *bad* elements produced in mesh generation only accounts for a small part in the whole mesh, and this small part of bad elements will often greatly deteriorate the accuracy of solution. Therefore, in this paper the quality of a mesh is defined as the quality value of the *worst* element in the mesh and the quality improvement begins from the worst element.

Mesh improvement procedure contain two main categorie: geometrical optimization - node repositioning or smoothing - relocates mesh points to improve mesh quality without changing mesh topology, Zavattieri et al. (1996); Lo (1997); Freitag and Ollivier-Gooch (1997, 2000); Sun and Liu (2003); Chen et al. (2004); Alliez et al. (2005); topological optimization - local transformation or reconnection - changes the topology of a mesh, i.e. node-element connectivity relationship, Zavattieri et al. (1996); Lo (1997); Freitag and Ollivier-Gooch (1997); Joe (1991a, 1995). Local mesh modifications typically involves: edge flipping, edge collapsing, edge splitting; and node removal, node repositioning and degree relaxation. Mesh quality can often be improved through the use of algorithms based on local reconnection schemes, node smoothing, and adaptive refinement or coarsening. Jianfei et al. (2006) recently proposed the strategy of optimal tetrahedralization for small polyhedron and corresponding small polyhedron reconnection (SPR) operation, which seeks the optimal tetrahedralization of a polyhedron with a certain number of vertexes and faces instead of choosing the best configuration from several possibilities within a small region that consists of a small number of tetrahedra. In this thesis, we extend *straight-sided edge flip* to *curvilinear edge flip* and *Small Polyhedron Reconnection (SPR)* to *Curvilinear Small Polyhedron Reconnection (CSPR)* with respect to a metric filed. It is still a very important topic to research more very robust mesh optimization strategies to improve mesh quality.

5. *Validity and quality measure*

   What is a good mesh? What is a good element? Provably good mesh generation is needed to have a good solution. The accuracy of the approximation depends on the sizes and shapes of the elements, the quality of the created elements is certainly one of the most important characteristics in mesh generation and thus for methods to assess validity and quality of such meshes.

   For the quality of the finite elements, there are geometric quality measures and Jacobian-based quality measures. Geometric quality measures are constructed from geometric characteristics, such as the length of the edges, the $2D$ area of the elements, the $3D$ volume of the elements, the radii of the inscribed $2D$

circumscribed circles and the radii of the inscribed $3D$ circumscribed spheres; Jacobian matrix of the mapping between the reference element and the physical element contains all the distortion information and is defined for every order and every type of element, thus Jacobian-based measures are essentially pointwise within the element and are a more natural fit to define quality measures Field (2000); Shewchuk (2002). More sharp and exact *validity and quality measure* is still very needed in the mesh research in the future, even for high-order and curvilinear mesh.

6. $3D$ *problems*

   $3D$ meshing is much more complicated and the bare existence of such $3D$ meshes is not guaranteed, thus $3D$ anisotropic mesh adaptation is even more complicated. "Mesh generation in three dimensions is a difficult enough task in the absence of mesh adaptation, and it is only recently that satisfactory three-dimensional mesh generators have become available. Even now, they do not exhibit the robustness and reliability that one has come to expect in two dimensions. Mesh alteration in three dimensions is therefore a rather perilous procedure that should be undertaken with care." Baker (1997). In $3D$, only a few works exist, and it is needed to fully extend the overall methodology implemented in a $2D$ process to $3D$ process.

7. *Space-time problems*

   More complex industrial unsteady problems - *space-time problems* - involve moving geometries. These simulations are time consuming. Using curvilinear mesh adaptation is a way to enhance the accuracy and reduce the cost. Extending curvilinear mesh adaptation to this context requires to take into account the mesh motion inside the derivatives recovery, the error estimate, the metric construction, the adapted mesh generation process, the validity and quality measure and mesh improvement/optimization strategies.

There is still a very long road to explore in the way of mesh generation. The aim, from a theoretical and practical point of view, in the end of the story is to build a optimization - *scalability, robustness, accuracy* - framework of curvilinear mesh generation and adaptation to predict the behavior of any physical problems - *complex space-time problems* - in $2D$ and $3D$.

Bibliography

Ainsworth, M., Oden, J.T., 2011. A posteriori error estimation in finite element analysis. volume 37. John Wiley & Sons.

Ait-Ali-Yahia, D., Baruzzi, G., Habashi, W.G., Fortin, M., Dompierre, J., Vallet, M.G., 2002. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent cfd. part ii. structured grids. International Journal for Numerical Methods in Fluids 39, 657–673.

Alauzet, F., Loseille, A., 2016. A decade of progress on anisotropic mesh adaptation for computational fluid dynamics. Computer-Aided Design 72, 13–39.

Alauzet, F., Loseille, A., Dervieux, A., Frey, P., 2006. Multi-dimensional continuous metric for mesh adaptation, in: Proceedings of the 15th International Meshing Roundtable, Springer. pp. 191–214.

Alliez, P., Cohen-Steiner, D., Yvinec, M., Desbrun, M., 2005. Variational tetrahedral meshing, in: ACM SIGGRAPH 2005 Papers, pp. 617–625.

Almeida, R.C., Feijóo, R.A., Galeao, A.C., Padra, C., Silva, R.S., 2000. Adaptive finite element computational fluid dynamics using an anisotropic error estimator. Computer Methods in Applied Mechanics and Engineering 182, 379–400.

Apel, T., 1999a. Anisotropic finite elements: local estimates and applications. Teubner Stuttgart.

Apel, T., 1999b. Interpolation of non-smooth functions on anisotropic finite element meshes. ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique 33, 1149–1185.

Baker, T.J., 1997. Mesh adaptation strategies for problems in fluid dynamics. Finite Elements in Analysis and Design 25, 243–273.

Bassi, F., Rebay, S., 1997. High-order accurate discontinuous finite element solution of the 2d euler equations. Journal of computational physics 138, 251–285.

Baudouin, T.C., Remacle, J.F., Marchandise, E., Henrotte, F., Geuzaine, C., 2014. A frontal approach to hex-dominant mesh generation. Advanced Modeling and Simulation in Engineering Sciences 1, 8.

Beaufort, P.A., Lambrechts, J., Henrotte, F., Geuzaine, C., Remacle, J.F., 2017. Computing cross fields a pde approach based on the ginzburg-landau theory. Procedia engineering 203, 219–231.

Beckmann, N., Kriegel, H.P., Schneider, R., Seeger, B., 1990. The r*-tree: an efficient and robust access method for points and rectangles, in: Acm Sigmod Record, Acm. pp. 322–331.

Benzley, S.E., Perry, E., Merkley, K., Clark, B., Sjaardama, G., 1995. A comparison of all hexagonal and all tetrahedral finite element meshes for elastic and elasto-plastic analysis, in: Proceedings, 4th international meshing roundtable, Citeseer. pp. 179–191.

Bernard, P.E., Remacle, J.F., Legat, V., 2009. Boundary discretization for high-order discontinuous galerkin computations of tidal flows around shallow water islands. International Journal for Numerical Methods in Fluids 59, 535–557.

Blacker, T.D., Stephenson, M.B., 1991. Paving: A new approach to automated quadrilateral mesh generation. International journal for numerical methods in engineering 32, 811–847.

Boissonnat, J.D., Devillers, O., Pion, S., Teillaud, M., Yvinec, M., 2002. Triangulations in cgal. Computational Geometry 22, 5–19. 16th ACM Symposium on Computational Geometry.

Bommes, D., Campen, M., Ebke, H.C., Alliez, P., Kobbelt, L., 2013. Integer-grid maps for reliable quad meshing. ACM Transactions on Graphics (TOG) 32, 1–12.

Borouchaki, H., George, P.L., Hecht, F., Laug, P., Saltel, E., 1997a. Delaunay mesh generation governed by metric specifications. part i. algorithms. Finite elements in analysis and design 25, 61–83.

Borouchaki, H., George, P.L., Mohammadi, B., 1997b. Delaunay mesh generation governed by metric specifications part ii. applications. Finite Elements in Analysis and Design 25, 85–109.

Bottasso, C.L., 2004. Anisotropic mesh adaption by metric-driven optimization. International Journal for Numerical Methods in Engineering 60, 597–639.

Botti, L., 2012. Influence of reference-to-physical frame mappings on approximation properties of discontinuous piecewise polynomial spaces. Journal of Scientific Computing 52, 675–703.

Bruneau, C.H., Saad, M., 2006. The 2d lid-driven cavity problem revisited. Computers & fluids 35, 326–348.

Buscaglia, G.C., Agouzal, A., Ramirez, P., Dari, E.A., 1998. On hessian recovery and anisotropic adaptivity .

Buscaglia, G.C., Dari, E.A., 1997. Anisotropic mesh optimization and its application in adaptivity. International Journal for Numerical Methods in Engineering 40, 4119–4136.

Cao, W., 2005. On the error of linear interpolation and the orientation, aspect ratio, and internal angles of a triangle. SIAM journal on numerical analysis 43, 19–40.

Cao, W., 2007. An interpolation error estimate on anisotropic meshes in r^n and optimal metrics for mesh refinement. SIAM journal on numerical analysis 45, 2368–2391.

Cao, W., 2008. An interpolation error estimate in r^2 based on the anisotropic measures of higher order derivatives. Mathematics of computation 77, 265–286.

Castro-Díaz, M., Hecht, F., Mohammadi, B., Pironneau, O., 1997. Anisotropic unstructured mesh adaption for flow simulations. International Journal for Numerical Methods in Fluids 25, 475–491.

Chen, L., Sun, P., Xu, J., 2007. Optimal anisotropic meshes for minimizing interpolation errors in l^p -norm. Mathematics of Computation 76, 179–204.

Chen, L., Xu, J.c., 2004. Optimal delaunay triangulations. Journal of Computational Mathematics , 299–308.

Chen, Z., Tristano, J.R., Kwok, W., 2004. Construction of an objective function for optimization-based smoothing. Engineering with Computers 20, 184–192.

Chung, S.W., Choi, Y.J., Kim, S.J., 2003. Computational simulation of localized damage by finite element remeshing based on bubble packing method. Computer Modeling in Engineering and Sciences 4, 707–718.

Ciarlet, P.G., Raviart, P.A., 1972. Interpolation theory over curved elements, with applications to finite element methods. Computer Methods in Applied Mechanics and Engineering 1, 217–249.

Clément, P., 1975. Approximation by finite element functions using local regularization. Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique 9, 77–84.

Cockburn, B., Karniadakis, G.E., Shu, C.W., 2000. The development of discontinuous galerkin methods, in: Discontinuous Galerkin Methods. Springer, pp. 3–50.

Cockburn, B., Shu, C.W., 1989. Tvb runge-kutta local projection discontinuous galerkin finite element method for conservation laws. ii. general framework. Mathematics of computation 52, 411–435.

Comon, P., Mourrain, B., 1996. Decomposition of quantics in sums of powers of linear forms. Signal Processing 53, 93–107.

Coulaud, O., Loseille, A., 2016. Very high order anisotropic metric-based mesh adaptation in 3d. Procedia engineering 163, 353–365.

Coupez, T., 2011. Metric construction by length distribution tensor and edge based error for anisotropic adaptive meshing. J. Comput. Physics 230, 2391–2405.

Coupez, T., 2017. On a basis framework for high order anisotropic mesh adaptation.

Coupez, T., Digonnet, H., Hachem, E., Laure, P., Silva, L., Valette, R., 2010. Multidomain finite element computations: Application to multiphasic problems.

Courty, F., Leservoisier, D., George, P.L., Dervieux, A., 2006. Continuous metrics and mesh adaptation. Applied Numerical Mathematics 56, 117–145.

Dari, E., Buscaglia, G., 1994. Mesh optimization: how to obtain good unstructured 3d finite element meshes with not-so-good mesh generators. Structural Optimization 8, 181–188.

D'Azevedo, E.F., Simpson, R.B., 1991. On optimal triangular meshes for minimizing the gradient error. Numerische Mathematik 59, 321–348.

Dey, S., 1997. Geometry-based three-dimensionalhp-finite element modelling and computations. Rensselaer Polytechnic Institute.

Dey, S., Shephard, M.S., Flaherty, J.E., 1997. Geometry representation issues associated with p-version finite element computations. Computer methods in applied mechanics and engineering 150, 39–55.

Dobrzynski, C., 2012. MMG3D: User guide. Ph.D. thesis. INRIA.

Dompierre, J., Vallet, M.G., Bourgault, Y., Fortin, M., Habashi, W.G., 2002. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent cfd. part iii. unstructured meshes. International journal for numerical methods in fluids 39, 675–702.

Dompierre, J., Vallet, M.G., Fortin, M., Bourgault, Y., Habashi, W., 1997. Anisotropic mesh adaptation-towards a solver and user independent cfd, in: 35th Aerospace Sciences Meeting and Exhibit, p. 861.

Duan Wang, R.L., Yan, N., 2010. An edge-based anisotropic mesh refinement algorithm and its application to interface problems .

Ern, A., Guermond, J.L., 2013. Theory and practice of finite elements. volume 159. Springer Science & Business Media.

Field, D.A., 2000. Qualitative measures for initial meshes. International Journal for Numerical Methods in Engineering 47, 887–906.

Fischer, P.F., 1997. An overlapping schwarz method for spectral element solution of the incompressible navier–stokes equations. Journal of Computational Physics 133, 84–101.

Formaggia, L., Micheletti, S., Perotto, S., 2004. Anisotropic mesh adaptation in computational fluid dynamics: application to the advection–diffusion–reaction and the stokes problems. Applied Numerical Mathematics 51, 511–533.

Formaggia, L., Perotto, S., 2001. New anisotropic a priori error estimates. Numerische Mathematik 89, 641–667.

Fortunato, M., Persson, P.O., 2016. High-order unstructured curved mesh generation using the winslow equations. Journal of Computational Physics 307, 1–14.

Freitag, L.A., Ollivier-Gooch, C., 1997. Tetrahedral mesh improvement using swapping and smoothing. International Journal for Numerical Methods in Engineering 40, 3979–4002.

Freitag, L.A., Ollivier-Gooch, C., 2000. A cost/benefit analysis of simplicial mesh improvement techniques as measured by solution efficiency. International Journal of Computational Geometry & Applications 10, 361–382.

Freitag, L.A., Plassmann, P., 2000. Local optimization-based simplicial mesh untangling and improvement. International Journal for Numerical Methods in Engineering 49, 109–125.

Frey, P.J., Alauzet, F., 2005. Anisotropic mesh adaptation for cfd computations. Computer methods in applied mechanics and engineering 194, 5068–5082.

George, P.L., Hecht, F., Saltel, É., 1990. Fully automatic mesh generator for 3d domains of any shape. IMPACT of Computing in Science and Engineering 2, 187–218.

Georgiadis, C., Beaufort, P.A., Lambrechts, J., Remacle, J.F., 2017. High quality mesh generation using cross and asterisk fields: Application on coastal domain, Research Notes, 26th International Meshing Roundtable, Sandia National Laboratories, September 18-21 2017.

Geuzaine, C., Remacle, J.F., 2009. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. International journal for numerical methods in engineering 79, 1309–1331.

Gilbert Strang, E.â.H., Mar 30, 2016. Calculus Volume 3. OpenStax.

Gruau, C., Coupez, T., 2005. 3d tetrahedral, unstructured and anisotropic mesh generation with adaptation to natural and multidomain metric. Computer Methods in Applied Mechanics and Engineering 194, 4951–4976.

Gu, X.D., Yau, S.T., 2008. Computational conformal geometry. volume 3 of *Advanced Lectures in Mathematics (ALM)*. International Press, Somerville, MA; Higher Education Press, Beijing. With 1 CD-ROM (Windows, Macintosh and Linux).

Habashi, W.G., Dompierre, J., Bourgault, Y., Ait-Ali-Yahia, D., Fortin, M., Vallet, M.G., 2000. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent cfd. part i: general principles. International Journal for Numerical Methods in Fluids 32, 725–744.

Hartmann, R., Leicht, T., 2016. Generation of unstructured curvilinear grids and high-order discontinuous galerkin discretization applied to a 3d high-lift configuration. International Journal for Numerical Methods in Fluids 82, 316–333.

Hecht, F., Kuate, R., 2014. An approximation of anisotropic metrics from higher order interpolation error for triangular mesh adaptation. Journal of Computational and Applied Mathematics 258, 99–115.

Hecht, F., Mohammadi, B., Hecht, F., Mohammadi, B., 1997. Mesh adaption by metric control for multi-scale phenomena and turbulence, in: 35th Aerospace Sciences Meeting and Exhibit, p. 859.

Hiester, H., Piggott, M., Farrell, P., Allison, P., 2014. Assessment of spurious mixing in adaptive mesh simulations of the two-dimensional lock-exchange. Ocean Modelling 73, 30–44.

Huang, W., 2005. Metric tensors for anisotropic mesh generation. Journal of computational physics 204, 633–665.

Huang, W., Kamenski, L., Lang, J., 2010. A new anisotropic mesh adaptation method based upon hierarchical a posteriori error estimates. Journal of Computational Physics 229, 2179–2198.

Huang, W., Sun, W., 2003. Variational mesh adaptation ii: error estimates and monitor functions. Journal of computational physics 184, 619–648.

Huang, Y., Su, Y., Wei, H., Yi, N., 2013. Anisotropic mesh generation methods based on acvt and natural metric for anisotropic elliptic equation. Science China Mathematics 56, 2615–2630.

Ims, J., Duan, Z., Wang, Z.J., 2015. meshcurve: An automated low-order to high-order mesh generator.

Jezdimirovic, J., Chemin, A., Reberol, M., Henrotte, F., François Remacle, J., 2021. Quad layouts with high valence singularities for flexible quad meshing. arXiv e-prints .

Jianfei, L., 2003. Automatic mesh generation of 3-d geometric models. Acta Mechanica Sinica 19, 285–288.

Jianfei, L., Shuli, S., Dachuan, W., 2006. Optimal tetrahedralization for small polyhedron: a new local transformation strategy for 3-d mesh generation and mesh improvement. Computer Modeling in Engineering and Sciences 14, 31.

Joe, B., 1991a. Construction of three-dimensional delaunay triangulations using local transformations. Computer Aided Geometric Design 8, 123–142.

Joe, B., 1991b. Delaunay versus max-min solid angle triangulations for three-dimensional mesh generation. International Journal for Numerical Methods in Engineering 31, 987–997.

Joe, B., 1995. Construction of three-dimensional improved-quality triangulations using local transformations. SIAM Journal on Scientific Computing 16, 1292–1307.

Johnen, A., Remacle, J.F., Geuzaine, C., 2013. Geometrical validity of curvilinear finite elements. Journal of Computational Physics 233, 359–372.

Karman, S.L., Erwin, J.T., Glasby, R.S., Stefanski, D., 2016. High-order mesh curving using wcn mesh optimization, in: 46th AIAA Fluid Dynamics Conference, p. 3178.

Kennon, S.R., Dulikravich, G.S., 1986. Generation of computational grids using optimization. AIAA journal 24, 1069–1073.

Kobayashi, K., Tsuchiya, T., 2014. A priori error estimates for lagrange interpolation on triangles. Applications of Mathematics 60, 485–499.

Kroll, N., 2006. Adigma: A european project on the development of adaptive higher order variational methods for aerospace applications, in: 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, p. 176.

Kroll, N., 2010. The adigma project, in: ADIGMA-A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications. Springer, pp. 1–9.

Kroll, N., Bieler, H., Deconinck, H., Couaillier, V., Van der Ven, H., Sorensen, K., 2010. ADIGMA–A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications: Results of a Collaborative Research Project Funded by the European Union, 2006-2009. volume 113. Springer Science & Business Media.

Kroll, N., Hirsch, C., Bassi, F., Johnston, C., Hillewaert, K., 2015. IDIHOM: Industrialization of High-Order Methods-A Top-Down Approach: Results of a Collaborative Research Project Funded by the European Union, 2010-2014. volume 128. Springer.

Kuhlmann, H.C., Romanò, F., 2019. The lid-driven cavity. Computational Modelling of Bifurcations and Instabilities in Fluid Dynamics , 233–309.

Ladevèze, P., Pelle, J., Rougeot, P., 1991. Error estimation and mesh optimization for classical finite elements. Engineering Computations 8, 69–80.

Lawson, C.L., 1972. Generation of a triangular grid with applications to contour plotting. Jet Propul. Lab. Techn. Memo 299, 2.

Lévy, B., . Geogram, 2015. URL http://alice. loria. fr/index. php/software/4-library/75-geogram. html .

Li, X., Shephard, M.S., Beall, M.W., 2005. 3d anisotropic mesh adaptation by mesh modification. Computer methods in applied mechanics and engineering 194, 4915–4950.

Li, X.Y., Teng, S.H., Wan, P.J., 2001. Generate good triangular meshes, conforming to control spacing requirements .

Liu Jianfei, Sun Shuli, W.D., 2006. Optimal tetrahedralization for small polyhedron: A new local transformation strategy for 3-d mesh generation and mesh improvement. Computer Modeling in Engineering & Sciences 14, 31–44.

Lo, S., 1997. Optimization of tetrahedral meshes based on element shape measures. Computers & structures 63, 951–961.

Lo, S., Wang, W., 2005. Generation of tetrahedral mesh of variable element size by sphere packing over an unbounded 3d domain. Computer Methods in Applied Mechanics and Engineering 194, 5002–5018.

Loseille, A., Alauzet, F., 2011a. Continuous mesh framework part i: well-posed continuous interpolation error. SIAM Journal on Numerical Analysis 49, 38–60.

Loseille, A., Alauzet, F., 2011b. Continuous mesh framework part ii: validations and applications. SIAM Journal on Numerical Analysis 49, 61–86.

Marcum, D.L., Alauzet, F., 2013. Unstructured mesh generation using advancing layers and metric-based transition for viscous flowfields, in: 21st AIAA Computational Fluid Dynamics Conference, p. 2710.

Marot, C., Pellerin, J., Remacle, J.F., 2019. One machine, one minute, three billion tetrahedra. International Journal for Numerical Methods in Engineering 117, 967–990.

Marot, C., Remacle, J.F., 2020. Quality tetrahedral mesh generation with hxt. arXiv preprint arXiv:2008.08508 .

Marot, C., Verhetsel, K., Remacle, J.F., 2020. Reviving the search for optimal tetrahedralizations. Proceedings of the 28th International Meshing Roundtable .

Mbinky, E., Alauzet, F., Loseille, A., 2012. Higher-order interpolation for mesh adaptation .

Moxey, D., Ekelschot, D., Keskin, Ü., Sherwin, S.J., Peiró, J., 2016. High-order curvilinear meshing using a thermo-elastic analogy. Computer-Aided Design 72, 130–139.

Nadler, E., 1985. Piecewise linear approximation on triangulations of a planar region .

Pain, C., Umpleby, A., De Oliveira, C., Goddard, A., 2001. Tetrahedral mesh optimisation and adaptivity for steady-state and transient finite element calculations. Computer Methods in Applied Mechanics and Engineering 190, 3771–3796.

Pascal, F., Marechal, J., 1998. Fast adaptive quadtree mesh generation. 1998 International Meshing Roundtable .

Picasso, M., 2003. An anisotropic error indicator based on zienkiewicz–zhu error estimator: Application to elliptic and parabolic problems. SIAM Journal on Scientific Computing 24, 1328–1355.

Picasso, M., Alauzet, F., Borouchaki, H., George, P.L., 2010. A numerical study of some Hessian recovery techniques on isotropic and anisotropic meshes. Technical Report RR-7202. INRIA.

Picasso, M., Alauzet, F., Borouchaki, H., George, P.L., 2011. A numerical study of some hessian recovery techniques on isotropic and anisotropic meshes. SIAM Journal on Scientific Computing 33, 1058–1076.

Piegl, L., Tiller, W., 1997. Symbolic operators for nurbs. Computer-Aided Design 29, 361–368.

Puso, M., Solberg, J., 2006. A stabilized nodally integrated tetrahedral. International Journal for Numerical Methods in Engineering 67, 841–867.

Quan, D.L., Toulorge, T., Marchandise, E., Remacle, J.F., Bricteux, G., 2014. Anisotropic mesh adaptation with optimal convergence for finite elements using embedded geometries. Computer Methods in Applied Mechanics and Engineering 268, 65–81.

Radzevich, S.P., 2013. Geometry of Surfaces: A practical Guide for Mechan-ical Engineers. Wiley.

Remacle, J.F., Gandham, R., Warburton, T., 2016. Gpu accelerated spectral finite elements on all-hex meshes. Journal of Computational Physics 324, 246–257.

Remacle, J.F., Henrotte, F., Carrier-Baudouin, T., Béchet, E., Marchandise, E., Geuzaine, C., Mouton, T., 2013a. A frontal delaunay quad mesh generator using the lâĹđ norm. International Journal for Numerical Methods in Engineering 94, 494–512.

Remacle, J.F., Lambrechts, J., Seny, B., Marchandise, E., Johnen, A., Geuzainet, C., 2012. Blossom-quad: A non-uniform quadrilateral mesh generator using a minimum-cost perfect-matching algorithm. International journal for numerical methods in engineering 89, 1102–1119.

Remacle, J.F., Toulorge, T., Lambrechts, J., 2013b. Robust untangling of curvilinear meshes, in: Proceedings of the 21st International meshing roundtable. Springer, pp. 71–83.

Rippa, S., 1992. Long and thin triangles can be good for linear interpolation. SIAM Journal on Numerical Analysis 29, 257–270.

Ruiz-Gironés, E., Roca, X., Sarrate, J., 2016. High-order mesh curving by distortion minimization with boundary nodes free to slide on a 3d cad representation. Computer-Aided Design 72, 52–64.

Rusinkiewicz, S., 2004. Estimating curvatures and their derivatives on triangle meshes, in: 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on, IEEE. pp. 486–493.

Schall, E., Leservoisier, D., Dervieux, A., Koobus, B., 2004. Mesh adaptation as a tool for certified computational aerodynamics. International journal for numerical methods in fluids 45, 179–196.

Scott, L.R., Zhang, S., 1990. Finite element interpolation of nonsmooth functions satisfying boundary conditions. Mathematics of Computation 54, 483–493.

Sevilla, R., Fernández-Méndez, S., Huerta, A., 2008a. Nurbs-enhanced finite element method for euler equations. International Journal for Numerical Methods in Fluids 57, 1051–1069.

Sevilla, R., Fernández-Méndez, S., Huerta, A., 2008b. Nurbs-enhanced finite element method (nefem). International Journal for Numerical Methods in Engineering 76, 56–83.

Shewchuk, J., 2002. What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures (preprint). University of California at Berkeley 73, 137.

Shewchuk, J.R., 1996. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator, in: Applied computational geometry towards geometric engineering. Springer, pp. 203–222.

Si, H., 2015. Tetgen, a delaunay-based quality tetrahedral mesh generator. ACM Transactions on Mathematical Software (TOMS) 41, 1–36.

Simpson, R.B., 1994. Anisotropic mesh transformations and optimal error control. Applied Numerical Mathematics 14, 183–198.

Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., Mavriplis, D., 2014. Cfd vision 2030 study: a path to revolutionary computational aerosciences .

Sun, S., Liu, J., 2003. An efficient optimization procedure for tetrahedral meshes by chaos search algorithm. Journal of Computer Science and Technology 18, 796–803.

Tam, A., Ait-Ali-Yahia, D., Robichaud, M., Moore, M., Kozel, V., Habashi, W., 2000. Anisotropic mesh adaptation for 3d flows on structured and unstructured grids. Computer Methods in Applied Mechanics and Engineering 189, 1205–1230.

Thompson, J.F., Soni, B.K., Weatherill, N.P., 1998. Handbook of grid generation. CRC press.

Toulorge, T., Desmet, W., 2010. Curved boundary treatments for the discontinuous galerkin method applied to aeroacoustic propagation. AIAA journal 48, 479–489.

Toulorge, T., Geuzaine, C., Remacle, J.F., Lambrechts, J., 2013. Robust untangling of curvilinear meshes. Journal of Computational Physics 254, 8–26.

Wang, Z.J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H.T., et al., 2013. High-order cfd methods: current status and perspective. International Journal for Numerical Methods in Fluids 72, 811–845.

Wikipedia contributors, 2022. Types of mesh — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Types_of_mesh&oldid=1066891597. [Online; accessed 1-February-2022].

Zavattieri, P.D., Dari, E.A., BUSCAGLIA, G.C., 1996. Optimization strategies in unstructured mesh generation. International Journal for Numerical Methods in Engineering 39, 2055–2071.

Zhang, H., Trépanier, J.Y., 1994. An algorithm for the optimization of directionally stretched triangulations. International journal for numerical methods in engineering 37, 1481–1497.

Zhang, R., Johnen, A., Remacle, J.F., 2018. Curvilinear mesh adaptation, in: International Meshing Roundtable, Springer. pp. 57–69.

Zhang, Z., Naga, A., 2005. A new finite element gradient recovery method: superconvergence property. SIAM Journal on Scientific Computing 26, 1192–1213.

Zienkiewicz, O.C., Zhu, J.Z., 1992a. The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique. International Journal for Numerical Methods in Engineering 33, 1331–1364.

Zienkiewicz, O.C., Zhu, J.Z., 1992b. The superconvergent patch recovery and a posteriori error estimates. part 2: Error estimates and adaptivity. International Journal for Numerical Methods in Engineering 33, 1365–1382.