POLITECNICO DI TORINO
Repository ISTITUZIONALE

Mesh Quality Meets The Virtual Element Method

Original
Mesh Quality Meets The Virtual Element Method / Sorgente, T., Vicini, F., Cabiddu, D., Biasotti, S., Spagnuolo, M., Manzini, G., Berrone, S.. - (2024), pp. 1-93. (2024 SIGGRAPH Asia 2024 Courses, SA Courses 2024 Tokyo, Japan 2024) [10.1145/3680532.3689578].

Availability:
This version is available at: 11583/3015708 since: 2026-09-19T21:31:07Z

Publisher:
Association for Computing Machinery - ACM

Published
DOI:10.1145/3680532.3689578

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

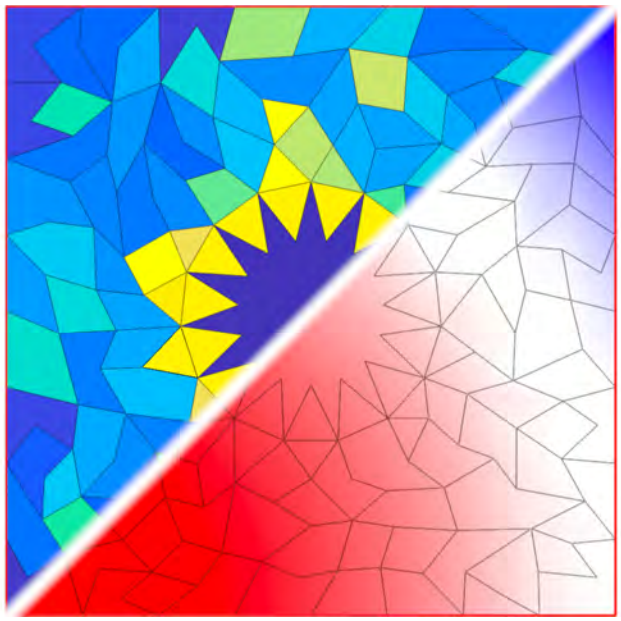
(Article begins on next page)

18 November 2026

SIGGRAPH Asia 2024 Courses, December 03-06, 2024, Tokyo, Japan

Mesh Quality Meets The Virtual Element Method

Tommaso Sorgente, Fabio Vicini, Daniela Cabiddu, Silvia Biasotti, Michela Spagnuolo, Gianmarco Manzini, Stefano Berrone



Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s). SA Courses '24, December 03-06, 2024, Tokyo, Japan, © 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1135-0/24/12, https://doi.org/10.1145/3680532.3689578

Contents

1 Introduction **4**
- 1.1 Simulations, Meshes and Computer Graphics . . . 4
- 1.2 Contribution . . . 5
- 1.3 Organization . . . 5

2 Mesh Quality **7**
- 2.1 Background and Notation . . . 8
  - 2.1.1 Mesh Generalities . . . 8
  - 2.1.2 Polytopes Generalities . . . 10
  - 2.1.3 Spectral Operators . . . 11
  - 2.1.4 Quality Indicators . . . 12
  - 2.1.5 Mesh Quality and Finite Element Methods . . . 14
  - 2.1.6 The First Example . . . 16
- 2.2 Element Quality Indicators . . . 18
  - 2.2.1 Simplicial Elements . . . 18
  - 2.2.2 Quads and Hexes . . . 21
  - 2.2.3 Polytopes . . . 24
  - 2.2.4 Discussion . . . 28
- 2.3 Mesh Quality Indicators . . . 33
  - 2.3.1 Consistency . . . 33
  - 2.3.2 Structure . . . 34
  - 2.3.3 Distribution . . . 35
- 2.4 Available Resources . . . 36

3 The Virtual Elements Method **38**
- 3.1 Motivations . . . 38
  - 3.1.1 VEM vs FEM . . . 39
- 3.2 Definitions . . . 40
  - 3.2.1 Problem Formulation . . . 40
  - 3.2.2 VEM Space . . . 41
  - 3.2.3 VEM Operators . . . 43
  - 3.2.4 VEM Error Estimation . . . 44
- 3.3 Mesh Requirements for the VEM . . . 44
  - 3.3.1 Assumption G1 . . . 44

3.3.2 Assumption G2 46
3.3.3 Assumption G3 47
3.3.4 Assumption G4 48
3.4 Mesh Quality Indicator for the VEM 49
3.4.1 Correlation With VEM Performance 52

4 Optimizing Mesh Quality **56**
4.1 How to Get a Quality Mesh 56
4.1.1 Optimization by Geometric Operations 58
4.1.2 Optimization by Topological Operations 60
4.1.3 Recent Trends 61
4.2 Mesh Optimization Driven by Quality Indicators 63
4.2.1 Mesh Partitioning 63
4.2.2 Mesh Agglomeration 65
4.2.3 Optimization in Action 66
4.2.4 Applications 68

5 PEMesh Software **73**
5.1 Mesh Generation 74
5.2 Quality Evaluation 75
5.3 PDE Solver 76
5.4 Mesh Quality Optimization 77

Chapter 1

Introduction

This course offers a dive into the concept of mesh quality in the context of and in relation to numerical methods for simulation problems, techniques that are at the heart of a variety of computer graphics tasks. It is targeted to graduate students, researchers, and practitioners interested in mesh processing and generation oriented to numerical simulations with the Virtual Element Method (VEM), who want to learn how to assess and optimize the quality of any kind of polytopal mesh and are willing to explore the potentialities of the VEM in Computer Graphics. It is assumed that the reader has basic mathematical and geometrical notions and is familiar with mesh processing and geometry processing, i.e. knows what a mesh is and how to manipulate it. Some knowledge about numerical analysis is also required, in particular on the Finite Element Method.

1.1 Simulations, Meshes and Computer Graphics

Numerical methods and simulations are crucial in computer graphics, enabling the creation of realistic animations, deformations and fluid dynamics simulations. For instance, from a panoramic view of the papers published in the range 2003-2023 in the ACM Transactions on Graphics journal, there are 158 papers containing *physical simulation* as a keyword, with almost a third of them (50) published in the years 2021-2023. This thus reveals a clear upward trend of the computer graphics community for simulation-related aspects.

By leveraging numerical methods, we can approximate solutions to otherwise analytically intractable problems, making it possible to simulate real-world phenomena with remarkable accuracy and efficiency. The role of meshing in the simulation pipelines is also crucial whenever the physical domain needs to be represented by a *good mesh*, whatever this means, supporting the achievement of accurate simulation results while making them computationally tractable. The mesh becomes particularly relevant in approximating the domain boundary, as

complex shapes require appropriate tessellation strategies.

We are recently witnessing a growing interest in non-standard meshes containing generic polytopal elements, thus not limited to simplicial tessellations or grids. This shift of attention is present both in computer graphics, with the definition of adaptive meshes, and in numerical simulations, with the development of polytopal element methods such as the VEM.

1.2 Contribution

Constructing a *good mesh* for approximating a Partial Differential Equation (PDE) is a well-studied area: the literature on PDE solvers has plenty of definitions and criteria to measure the quality of elements and meshes. Both engineering and the computer graphics communities have contributed significantly to this field in the last decades, proposing seminal ideas, theoretical insights, and practical algorithms. The course is structured to cover a systematic review of mesh quality indicators in a very general setting, and to analyze the impact of quality on PDE solvers that accommodate for a general class of polytopal elements, the Virtual Element Method. Finally, the course will use the notions discussed to present how quality measures can be exploited to build optimal meshes for this particular PDE solver.

Besides being up to date with the most advanced notions and techniques, we believe the value of this work can be enclosed by the following points:

- it considers the widest possible class of meshes: planar, surface and volumetric, structured and unstructured, from simplicial meshes to quadrangular/hexahedral grids and generic polytopal discretizations;
- it analyzes a similarly wide range of quality indicators: local and global, geometric and algebraic. It highlights connections between such indicators and theoretical error bounds estimates or algorithms for quality optimization;
- it is the first tutorial presented at SIGGRAPH about the Virtual Element Method, a topic that is gaining more and more importance and interest in the Computer Graphics community;
- the freely available PEMesh software and the exercises performed at the end of each module are helpful tools that attendees can use also after the conference, for their daily work.

1.3 Organization

The work is organized in four chapters. Chapter 2 is about the definition and the analysis of the quality of a mesh. In Section 2.1 we set the notation and define some basic notions and properties about meshes, polytopes, spectral operators,

and quality indicators. We show how the geometric features of a mesh come into play in a finite elements simulation of a PDE, and briefly review some of the historical results on the angle conditions that must be satisfied by a triangular mesh to ensure a good approximation. In Section 2.2 we present a comparative analysis of the local quality indicators available in the literature. Such local indicators measure the geometric and algebraic properties of the single mesh elements and deal more naturally with finite element simulations. The quality of a mesh will be here related to the performance of a numerical scheme over it, in terms of speed and accuracy. In addition to the local indicators, in Section 2.3 we present some examples of global quality indicators that observe the quality of the mesh from a more general perspective. They measure aspects related to the consistency of the mesh with the physical domain, the structure of the mesh, or its connectivity, and are therefore more commonly used in computer graphics problems. Section 2.4 contains a summary of the most common libraries and datasets for assessing the quality of a mesh and testing a quality indicator.

In Chapter 3 we introduce the Virtual Element Method (VEM) as a powerful alternative to standard FEM. In Section 3.1 we present the motivations behind the development of this new method, its similarities and differences with the FEM and its advantages to the standard techniques. In Section 3.2 we formulate the VEM space and the VEM operators for a second-order elliptic differential problem defined on a two or three-dimensional domain. In Section 3.3 we report the main geometrical assumptions typically made on polygonal and polyhedral meshes to ensure optimal convergence rates of the VEM. Based on these assumptions, in Section 3.4 we present a new mesh quality indicator for polytopal meshes, specifically designed to measure the geometrical quality of a tessellation from the particular point of view of a VEM solver.

In Chapter 4 we discuss the optimization of a mesh. In Section 4.1 we review existing techniques for improving or optimizing the quality of a mesh, classified as geometrical methods that modify the position of the mesh nodes and topological methods that change the mesh connectivity. In Section 4.2 we introduce and detail a particular optimization algorithm driven by a local quality indicator, to be chosen among those presented before. We also present possible applications of this algorithm to constrained and time-dependent problems, showing its utility and effectiveness.

In Chapter 5 we present the software used in this course, PEMesh, both for the production of these notes and for the live exercises. We explain how to get it and how to exploit its potential by describing its main features and showing examples of its usage.

Chapter 2

Mesh Quality

This Chapter is a report of the work presented in [159]. What constitutes a good quality mesh, and exactly how fine the mesh should be, are questions that have been around since the first mesh was generated. Despite decades of research, a high variety can be found in the literature on concepts related to the *quality* of a mesh. The surveys [149, 169] cover early papers on mesh generation. In these surveys, there is little explicit mention of how to implement the first theoretical results, such as the well-celebrated angle conditions by Zlamal [181], and Babuska and Aziz [15]. However, in [169] it is explicitly stated that elements should be roughly equilateral or instability may ensue. Among the first applications, it is worth mentioning the studies by Marchand, Weatherill, and Hassan on the adaptation of unstructured tetrahedral meshes for transonic viscous flow simulation [117]. Liu and Joe [108] developed similar geometric mesh quality indicators in the case of tetrahedral meshes, which can effectively spot geometric flaws in the mesh that can affect the numerical resolution of a PDE. Subsequent studies, cf. [124, 50, 76, 121], provided a more in-depth treatment of the matter, which impacted the design and development of the first mesh-generation codes. More recent surveys exist on mesh quality indicators [91, 164] and mesh generation and processing [38, 13, 35, 130].

One thing we have learned from decades of research and dozens of surveys is that the concept of quality depends on the application context. The goodness of a tool principally depends on what that tool is used for, therefore a mesh considered "good" for one specific application may not work as fine for another. On the one hand, in computer graphics, one generally thinks of mesh quality for model reconstruction and processing, e.g. for guiding repairing operations with dihedral angles or for simplifying the model while keeping the mesh faithful to the acquired set of points [38, 13]. On the other hand, in the context of simulations by means of finite element methods (FEM), element quality can have a crucial impact on error estimates and convergence rates, thus simulation speed and accuracy [55], and relevant quality indicators depend on the type of simulation [107]. The understanding of how the accuracy of a PDE approximation depends on the mesh is still an active field of research.

At the same time, the breathtaking explosion of the computational power available in modern computers seen in the last years opens up new possibilities, unimaginable so far. Since one of the few concepts universally accepted about mesh quality is that the finer the mesh is, the more accurate results it is likely to produce [101], the idea of generating finer and finer meshes, regardless of the notion of quality, becomes terribly tempting. However, there are still situations in which simply decreasing the size of the elements is not enough to guarantee accurate results. The standard theory of spline approximation (cf. e.g. [100]) and the finite element method (cf. e.g. [42]) require the size of an underlying partition to go to zero to demonstrate the accuracy as well as the convergence of a numerical algorithm for solving a PDE. This fact, however, is not enough to estimate how small the elements must become to guarantee accurate results. Just to give a trivial example, let us imagine covering a triangular domain with some squared cells: no matter how small the squares are, they will never exactly cover the domain. Decreasing the size of the cells surely leads to a better approximation, but there is no way we could ever reach the same result as if we used triangular cells instead. In a more realistic scenario, in presence of particularly complex domains or problems, we may need to exponentially increase the number of elements in order to slightly improve the accuracy of the approximation. In such contexts, it may be worth considering spending some time analyzing the domain and the problem, to generate meshes with higher quality (whatever it means) and fewer elements.

2.1 Background and Notation

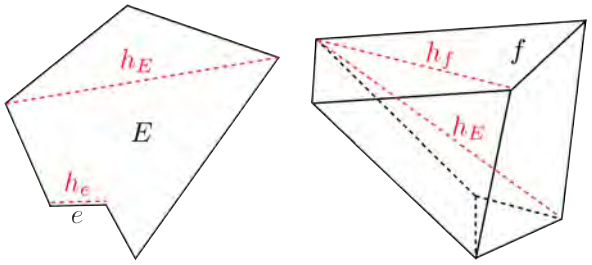
We introduce some basic concepts and properties about meshes, polytopes, and operators that will be used throughout the chapter. We focus on two- and three-dimensional meshes containing any kind of polytopal elements with straight edges and planar faces. As a consequence, meshes with curved elements are out of the scope of the present work. Besides being of great interest for some types of applications (e.g. isogeometric analysis [83]), the literature related to the quality of such types of meshes is still scarce.

2.1.1 Mesh Generalities

A *mesh*, often also called *tessellation*, *domain discretization*, *domain partition*, or *grid* (e.g., in the literature related to finite element methods), is a discrete approximation of an object or a domain $\Omega \subset \mathbb{R}^d$, partitioned into a finite collection of disjoint closed cells. The mesh is called *planar* if $d = 2$, or *volumetric* if $d = 3$. There exists a half-way case, i.e., meshes in $\mathbb{R}^3$ made by planar cells, also known as *surface* meshes. We consider them together with planar meshes, as they are locally planar, and refer to both as "2D-meshes", as opposed to "3D-meshes", which refers to volumetric ones.

The *cells* (or *elements*) of the mesh are subsets of $\mathbb{R}^d$ with no holes and no self-intersections. The boundary of a cell is composed of 2-dimensional *faces*

(in 3D-meshes), 1-dimensional *edges* and 0-dimensional *nodes* (or *vertices*), and two cells in a mesh can only share faces (in 3D-meshes), edges, and nodes. A mesh relative to a domain Ω with maximum edge length h (also called the *mesh size*) is noted Ω_h. We note by h_E the *diameter* of an element $E \in \Omega_h$, defined as the maximum point-to-point Euclidean distance in E. Similarly, we note by h_f the diameter of a face $f \in \partial E$ and by h_e the length of an edge $e \in \partial f$, see Figure 2.1. We also indicate with e_{ij} the edge shared by the faces f_i and f_j and with f_{ij} the face shared by the elements E_i and E_j.



(a) (b)

Figure 2.1: (a) polygon E with diameter h_E and its edge e with length h_e; (b) polyhedron E with diameter h_E and its face f with diameter h_f.

A mesh is called *pure* if all its elements are of the same type, i.e., they have the same number of edges in 2D or faces in 3D. Common examples are triangular, quadrangular, tetrahedral, and hexahedral meshes. The number of cells adjacent to a node (equivalently, the number of incident edges in 2D, or incident faces in 3D) is called its *valence*. Meshes typically have a majority of *regular* nodes with a fixed valence and a certain number of *singular* or *irregular* nodes with different ones. A mesh is said to be *structured* if every internal node is regular; otherwise, the mesh is called *unstructured*. A mesh is said to be *conforming* if two adjacent cells can only share a node, a whole edge, or a whole face. Otherwise, the mesh is called *non-conforming*, and the intersections inside edges or faces are called *T-junctions*, or hanging nodes (see Figure 2.2).

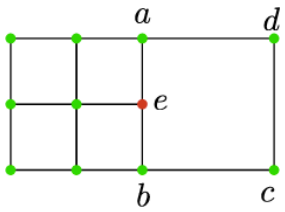


Figure 2.2: If the rightmost cell is defined as "*aebcd*", then the mesh is conforming but not purely quadrangular. If it is defined as "*abcd*" instead, the mesh is pure but non-conforming, and the node e is a T-junction.

2.1.2 Polytopes Generalities

The elements in a mesh are *polytopes*, i.e., polygons for 2D-meshes and polyhedra for 3D-meshes. Following [132], a *polygon* is defined by a finite set of line segments such that every segment extreme is shared by exactly two edges and no subset of edges has the same property. A *polyhedron* is a finite set of plane polygons such that every edge of a polygon is shared by exactly one other polygon and no subset of polygons has the same property. As commonly assumed [132], we consider polyhedra such that no pair of nonadjacent faces share a point.

A polytope P is said to be *convex* if, given any two points p_1 and p_2 in P, the line segment connecting p_1 and p_2 is entirely contained in P. Two points p_1 and p_2 in P are said to be *visible* from each other if the segment (p_1, p_2) does not intersect the boundary of P. The *kernel* of P is the set of points in the interior of P from which all the points in P are visible. The first obvious consideration is that the kernel of a polytope is a convex polytope. If P is convex, its kernel coincides with its interior, because any two points inside a convex polytope are visible from each other. A polytope may also not have a kernel at all; in this case, we say that its kernel is *empty*. A polytope P is called *star-shaped* if there exists a ball from which all the points in P are visible, see Figure 2.3. Note that the requirement of 'not intersecting the boundary' in the definition of visibility implies that a star-shapedness ball cannot have radius zero. Consider the case of a polygon $\tilde{P}$ star-shaped with respect to a ball $\tilde{B}$ with zero radius, i.e., a segment or a point. In particular, this means that $\tilde{P}$ contains at least one edge $\tilde{e}$ aligned with the points in $\tilde{B}$, and therefore it is not possible to connect a point in $\tilde{e}$ to a point in $\tilde{B}$ without touching the other points of the boundary of $\tilde{P}$ (the same argument holds for a polyhedron). A polytope is star-shaped if and only if its kernel is not empty, therefore star-shapedness can be thought of as an indicator of the existence of a kernel. Star-shapedness is weaker than convexity, and it is often used in the literature as many theoretical results in the theory of polynomial approximation in Sobolev spaces rely on this condition [42, 63].

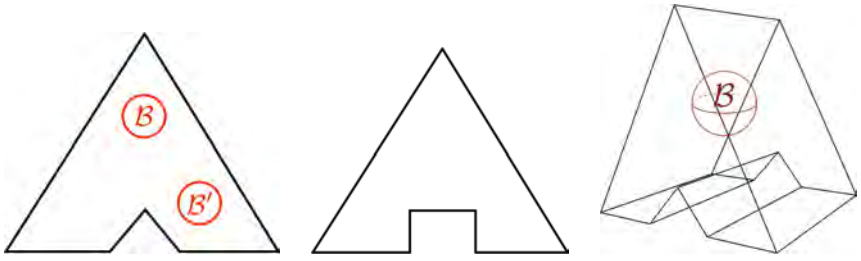


Figure 2.3: The polygon (a) is star-shaped with respect to the disk $\mathcal{B}$ but not with respect to the disk $\mathcal{B}'$; the polygon (b) is not star-shaped with respect to any disk; the polyhedron (c) is star-shaped with respect to the ball $\mathcal{B}$.

For every polytope P, we can define an *inradius* r_P and an *circumradius* R_P, which are, respectively, the radii of the maximum inscribed and the minimum circumscribed circle (or sphere). We note the area and the perimeter of P as A_P and p_P respectively, and every time there is no place for ambiguity we omit the subscript P. The *solid angle* θ_i at vertex $\mathbf{v}_i$ of a tetrahedron $T(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4)$ is defined to be the surface area formed by projecting each point on the face opposite to $\mathbf{v}_i$ to the unit sphere centered at $\mathbf{v}_i$, see Figure 2.4. It measures how large an object appears to an observer looking at it from a point, with a natural geometrical relationship to object visualization. In contrast, the *dihedral angle* between two adjacent faces of a tetrahedron is the angle between the intersection of the two faces and a plane perpendicular to their common edge.

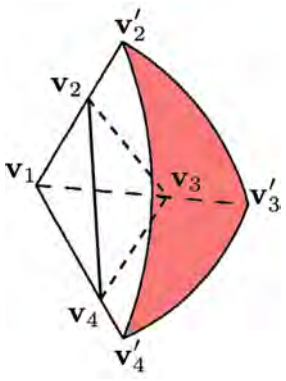


Figure 2.4: The solid angle θ_1 at vertex $\mathbf{v}_1$ of a tetrahedron $T(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4)$ is the surface area of the spherical triangle $T'(\mathbf{v}_2', \mathbf{v}_3', \mathbf{v}_4')$ (colored in red), where the $\mathbf{v}_i'$ are the projections of the $\mathbf{v}_i$ on the unit sphere centered at $\mathbf{v}_1$.

2.1.3 Spectral Operators

Consider a triangle with vertices $\mathbf{v}_i = (x_i, y_i)$, for $i = 0, 1, 2$ in physical space. Let ξ_i, with $0 \leq \xi_i \leq 1$ and $\xi_0 + \xi_1 + \xi_2 = 1$, be three logical space coordinates and define the mapping from logical to physical space by $\mathbf{x}(\xi) = \sum \xi_i \mathbf{v}_i$ with $\mathbf{x} \in \mathbb{R}^2$. This mapping can be explicitly written as

$$\mathbf{x} = (1 - \xi_1 - \xi_2)\mathbf{v}_0 + \xi_1 \mathbf{v}_1 + \xi_2 \mathbf{v}_2 = \mathcal{A}_0 \mathbf{u}_0 + \mathbf{v}_0,$$

with $\mathbf{u}_0 = (\xi_1, \xi_2)^t$ and

$$\mathcal{A}_0 = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{bmatrix}$$

We refer to $\mathcal{A}_0$ as the *Jacobian matrix* of the triangle relative to its vertex $\mathbf{v}_0$, because the columns of the matrix form the Jacobian of the affine map with respect to the logical variables, i.e., $\mathcal{A}_{ij} = d\mathbf{v}_i/d\xi_j$. We can define a Jacobian

matrix for each vertex of the triangle:

$$\mathcal{A}_i := \begin{bmatrix} x_{i+1} - x_i & x_{i+2} - x_i \\ y_{i+1} - y_i & y_{i+2} - y_i \end{bmatrix}$$

where the indices are taken modulo 3. The Jacobian matrix is not independent of the node on which it is computed, but the determinants of the three $\mathcal{A}_i$ coincide (and they equal twice the area of the triangle). Therefore we omit the subscript and note $\mathcal{J} := \det(\mathcal{A}_i)$, the *Jacobian* of the element. Analogous matrices can be defined for a tetrahedron with vertices $\mathbf{v}_i = (x_i, y_i, z_i)$, for $i = 0, 1, 2, 3$. However, in order to maintain the invariance of the determinant with respect to the order of the vertices, an alternate minus sign is added:

$$\mathcal{A}_i := (-1)^i \begin{bmatrix} x_{i+1} - x_i & x_{i+2} - x_i & x_{i+3} - x_i \\ y_{i+1} - y_i & y_{i+2} - y_i & y_{i+3} - y_i \\ z_{i+1} - z_i & z_{i+2} - z_i & z_{i+3} - z_i \end{bmatrix}$$

The determinant $\mathcal{J}$ of the Jacobian matrix of a tetrahedron gives six times its volume, no matter which of the four matrices is used [69]. The concept can be extended to non-simplicial polytopes like quadrangles and hexahedra, but in this case, the invariance is not guaranteed. A possible solution is to consider the different matrices $\mathcal{A}_i$ at the corners and at the center of the polytope, and define the Jacobian as $\mathcal{J} := \min_i \det(\mathcal{A}_i)$ or as a harmonic or geometric mean of the values [92].

According to [88], the Jacobian matrix is the fundamental theoretical object in structured mesh generation. It is fundamental because the columns of the matrix are vectors that point in the direction of the tangents to coordinate lines. Their lengths control the edge lengths of the mesh, their dot products are related to the angles between coordinate lines, and the determinant of the Jacobian matrix is simply related to the volume of the mesh element. Several quality indicators in Section 2.2 will be based on the condition number $\kappa(\mathcal{A}_i) = \|\mathcal{A}_i\| \, \|\mathcal{A}_i^{-1}\|$, or on the Jacobian $\mathcal{J}$. For instance, if $\mathcal{J} \leq 0$, the implied element is said to be *irregular*, and it is considered invalid in the context of finite element methods [89, 120]. Sometimes a distinction is made between degeneration ($\mathcal{J} = 0$) and inversion or fold-over ($\mathcal{J} < 0$). Depending on the concrete setting, such elements may lead to "inaccurate solutions or no solutions at all" [18], "invalidated" solutions [137], or situations in which "calculations cannot be continued" [141]. Other useful entities are the *metric tensors* $\mathcal{A}_k^T \mathcal{A}_k$ relative to the Jacobian matrix $\mathcal{A}_k$ at each vertex $\mathbf{v}_k$. These matrices are symmetric, and we note λ_{ij}^k the ij-th component of the k-th metric tensor. At the k-th node, the λ_{ii}^k are the squares of the lengths of the edges incident to $\mathbf{v}_k$, and they coincide with the eigenvalues λ_i^k of $\mathcal{A}_k^T \mathcal{A}_k$ [92].

2.1.4 Quality Indicators

We introduce a generic notion of a quality indicator that anticipates the more rigorous and particular definitions presented in Section 2.2.

A quality indicator is a function defined over a mesh, capable of giving insights on the accuracy and the convergence speed of a finite element scheme applied on that mesh, before solving any numerical problem.

As we will see in the next section, quality indicators can assume very different shapes and can be formulated in terms of different properties, with different approaches. However, there has been in the literature an effort to determine some basic properties that any "good" indicator should satisfy. First of all, we point out the difference between an indicator, a measure, and a metric. We highlight certain freedom in the literature, related to this terminology. *Indicator* is a generic term, that stands for a map $f : \Omega_h \to \mathbb{R}$ defined on the mesh, its elements, or its nodes. By using the term *measure* we assume (among other more obvious properties) that f is positive and bounded. For instance, all indicators taking values in the range $[-1, 1]$ can not be called quality measures unless re-scaling them opportunely. Similarly, indicators that assume infinite values for degenerate elements are not accepted. A *metric* (or *distance*) instead, is required to be positive but not bounded, and to satisfy the triangle inequality. The triangle inequality is a particularly strong condition, that makes several indicators not be metrics but just *semi-metrics* [168]. It is not so obvious however that this property is more important than others in relation to the concept of mesh quality, therefore it is not always required.

A general property, that turns out to be particularly useful in surveys and comparative works, is the universality with respect to the spatial dimension and the element type. An ideal indicator should be a function computable on both triangles, tetrahedra, quadrangles, and hexahedra, as well as on generic polytopes.

Fair indicators are defined in [68] to be those that possess four attributes: the ability to detect all degenerate elements, size invariance, boundedness, and normalization. An ability to detect all degenerate elements means that f yields a value of zero for two- (three-) dimensional elements that have no area (volume). Size invariance means that similar triangles (tetrahedra) or quadrilaterals (hexahedra) should yield the same value of f. Boundedness means that f cannot yield arbitrarily large values. Normalization forces f to take on positive values between zero and one, allowing better comparisons among indicators.

More properties are listed in [91, 92], like the invariance with respect to the orientation of the element, to translations, to the nodes indexing. It is also required that f is referenced to an ideal element that describes the desired geometric configuration of the physical element and that $0 \leq f \leq 1$, with $f = 1$ if and only if the physical element attains the ideal node configuration and $f = 0$ if and only if the physical element is degenerate. It is worth noting that size invariance may be required as a property for a good indicator, but the size of the elements (area or volume) may as well be considered a quality indicator itself.

2.1.5 Mesh Quality and Finite Element Methods

Finite element methods (FEMs) [42] and their many old and recent variants, such as the Discontinuous Galerkin method [45], the Mimetic Finite Difference (MFD) method [25], the Polygonal Finite Element Method (PFEM) [165], the Virtual Element Method (VEM) [20], the Hybrid-High Order (HHO) Method [59], etc., as well as the Finite Volume Methods [32, 33] provide very effective numerical approximation strategies on a wide range of problems involving partial differential equations. All these methods have in common that their formulation requires a family of properly refined meshes. The numerical solution accuracy depends on several factors: the degree of the approximation schemes, how the mesh approximates the geometry of the computational domain, the number of elements in the mesh, their size, and their shape. The interplay between the continuous formulation, its discretized form and the mesh has a fundamental role in determining how accurate the numerical solution is.

To discuss what a "good mesh" is in an application of the FEM, we first make a distinction between a-priori and a-posteriori analysis. A-posteriori analysis determines the critical regions of the computational domain where a mesh refinement could be beneficial to improve the accuracy of the numerical solution after such a solution has been computed. We address this topic in Section 4.1. In this section instead, we discuss a-priori analysis, which establishes the accuracy of the FEM depending on the degree of the method and the regularity of the solution. For the notation and specific details concerning the FEM we refer to [42].

Discretization and Interpolation Errors Let u be the exact solution of an elliptic problem in variational form, and u_h its numerical approximation. The *discretization error* is the difference between the "true" solution u and its finite element approximation u_h. We define the discretization error using the L^2 norm and the H^1 seminorm

$$\|u - u_h\|_{0,\Omega}^2 := \int_\Omega |u - u_h|^2 \, d\mathbf{x},$$

$$|u - u_h|_{1,\Omega}^2 := \int_\Omega |\nabla u - \nabla u_h|^2 \, d\mathbf{x},$$

and the H^1-norm

$$\|u - u_h\|_{1,\Omega}^2 := \|u - u_h\|_{0,\Omega}^2 + |u - u_h|_{1,\Omega}^2.$$

In the a-priori analysis, we search for an upper bound to these quantities. A standard result from the theory of the finite element method applied to the variational formulation of a self-adjoint second-order PDE indicates that the discretization error can be bounded by *the interpolation error*. The interpolation error is the error committed when we approximate some "true" function u (not necessarily the solution of a PDE) with an *interpolating function* u_I that belongs to the finite element space. The interpolation u_I shares the degrees of

freedom with u (for example, the values at the mesh vertices), assuming that u is sufficiently regular. An application of the Céa's Lemma (cf. [42]), allows us to write the inequality

$$|u - u_h|_{1,\Omega} \leq C \inf_{v_h \in \mathcal{V}_h} |u - v_h|_{1,\Omega} \leq C|u - u_{\mathrm{I}}|_{1,\Omega}. \tag{2.1}$$

Similarly, by using a duality argument usually referred to in the literature as the Aubin-Nische's trick [42], we can prove that

$$\|u - u_h\|_{0,\Omega} \leq Ch|u - u_h|_{1,\Omega}. \tag{2.2}$$

Therefore, if we know an upper bound on the interpolation error in the right-hand side of (2.1), we estimate the discretization error in the H^1-norm still from (2.1). Then, using this upper bound, we estimate the discretization error in the L^2-norm using (2.2).

In other words, inequality (2.1) makes it possible to control the discretization error through the interpolation error. However, the dependence of these two types of error on the shape and the size of the mesh elements is different. On one hand, the discretization error is strongly related to the stiffness matrix, i.e., the matrix that represents the system of linear equations to be solved. Indeed, poorly conditioned matrices affect linear equation solvers by slowing them down or introducing large roundoff errors into their results. The relationship between element shape and matrix conditioning depends on the PDE being solved and the basis and test functions used to discretize it, and is reflected in the value of the constant factors C in the estimates. On the other hand, the interpolation error only depends on the approximation properties of the finite element space $\mathcal{V}_h$, and is, thus, independent of the partial differential equation to be solved numerically. Since the finite element space $\mathcal{V}_h$ where we seek u_I, is built on a specific mesh (or sequence of meshes), the interpolation error strongly depends on the geometric properties of the elements (edges, angles, area, inradius, ...). Following on the path drawn in [152], in Section 2.2 we will therefore assess mesh quality in relation to the discretization error and the interpolation error.

Role of the C Constant In the H^1 and L^2 a-priori errors estimates (2.1)–(2.2) we explicit the dependence on the mesh parameter h to establish a convergence rate, and the dependence on a higher seminorm of the exact solution u to show the minimal regularity that u must have for this estimate to be valid. However, the dependence on other mesh factors or parameters is not explicited. In fact, the quality of the mesh is normally hidden in the various constants C, independent of h and u, that "infest" such estimates. From such convergence estimates, we may also (erroneously) assume that decreasing the mesh size, i.e., considering a sequence of meshes with smaller and smaller elements, is sufficient to improve the accuracy of the solution. However, the constants C still depend on the geometrical shape of the elements. This latter one cannot be arbitrary, because, otherwise, the value of C, which is uniform on the whole sequence of refined meshes, would deteriorate.

We can describe the shape of an element in terms of geometrical quantities such as the internal angles, the radius of the circle or sphere that circumscribes the element, etc. As noted in [152], several Authors in the literature were successful in determining a relationship between the constant C and the geometrical quantities of a given element, thus paving the way for the formulation of mesh indicators, many of which we will review in the next sections.

2.1.6 The First Example

The first historical results in mesh quality assessment are related to the internal angles of a triangular mesh. This investigation begins with Zlamal [181], Babuska and Aziz [15], that prove results of the form

$$\|u - u_h\|_{1,\Omega} \leq C(\theta) h |u|_{2,\Omega} \quad \text{where} \quad |u|^2_{2,\Omega} := \sum_{i,j=1}^{2} \left| \frac{\partial^2 u}{\partial x^i \partial y^j} \right|^2 ,$$

and $C(\theta)$ is a constant factor that depends on the internal angles of the triangles forming the mesh. Zlamal demonstrates that $C(\theta) \simeq 1/\sin(\theta_{min})$, where $\theta_{min} = \min(\theta_1, \theta_2, \theta_3)$, and θ_i, $i = 1, 2, 3$, is the angle of the i-th vertex of a triangular cell. Babuska and Aziz prove that $C(\theta) \simeq 1/\Psi(\theta)$, where $\Psi(\theta)$ is a positive and continuous function such that $\Psi(\theta) \geq \Psi(\gamma)$ for $\theta \leq \gamma < \pi$. Here, γ is a bound on the maximum angle of the mesh, so that the proper condition for triangles is the absence of large angles. So, according to conventional thinking, triangular elements should have no small or large angles wherever possible. However, internal angles impact differently on the discretization and the interpolation error.

The interpolation error requires a maximum angle condition and is relatively insensitive to small angles. This fact had a significant impact on using anisotropic meshes for solving elliptic problems with strong anisotropies. Based on interpolation errors, Rippa [133] argues convincingly that long, thin triangles provide a proper mesh for solving elliptic problems with highly anisotropic coefficients. Apel and Dobrowolski [9] offer a unified theory of interpolation error estimates on nonuniform meshes and application to finite elements for anisotropic problems.

The situation is completely different for the discretization error, since small angles have a major impact on the conditioning of the stiffness matrix. Generally speaking, poor quality elements significantly affect the convergence and accuracy of the Krylov iterative solvers and the direct solvers that we apply to the system of linear equations resulting from the finite element discretization. The parameter measuring this difficulty is the condition number, and we expect that the higher its value, the harder it is to solve such linear systems. In fact, the condition number may arbitrarily increase if just one single angle of a triangular mesh approaches zero.

Let K denote the global matrix from an application of the finite element method to the bilinear form of the variational formulation of an adjoint, second-degree PDE like, for example, the Poisson equation with homogeneous boundary

conditions, and $\lambda_{max}(K)$ and $\lambda_{min}(K)$ the largest and smallest eigenvalues of K. If we include the boundary conditions, this bilinear form is symmetric, coercive, and continuous, the corresponding matrix K is symmetric and positive definite, and its eigenvalues are real, positive numbers. The condition number of K is defined as $\kappa(K) = \lambda_{max}(K)/\lambda_{min}(K)$.

The minimum eigenvalue $\lambda_{min}(K)$ is related to the size of the elements and the physical system that we want to model. In [73], we find an upper bound for $\lambda_{min}(K)$ that is proportional to the area or volume of the biggest element of the mesh, and a lower bound for $\lambda_{min}(K)$ that is proportional to the area or volume of the smallest element. Therefore, we can claim that, in general, reducing the size of the elements reduces its value. However, the value of the minimum eigenvalue is not very sensitive to the elemental shape, particularly the elemental angles.

Instead, the maximum eigenvalue $\lambda_{max}(K)$ can attain large values even if there is just a single badly shaped element E in the mesh. Let $\lambda_{max}(K^E)$ denote the biggest eigenvalue of the elemental stiffness matrix K^E. From [73], we also know that

$$\max_E \lambda_{max}(K^E) \leq \lambda_{max}(K) \leq m \ \max_E \lambda_{max}(K^E),$$

where m is the maximum number of elements meeting at a single vertex of the mesh. Consequently, the condition number $\kappa(K)$ is roughly proportional to the biggest eigenvalue of the elemental stiffness matrices.

We can exemplify this situation by writing the elemental stiffness matrix for a triangle as

$$K^E = \frac{1}{2}\begin{bmatrix} \cot\theta_2 + \cot\theta_3 & -\cot\theta_3 & -\cot\theta_2 \\ -\cot\theta_3 & \cot\theta_1 + \cot\theta_3 & -\cot\theta_1 \\ -\cot\theta_2 & -\cot\theta_1 & \cot\theta_1 + \cot\theta_2 \end{bmatrix}$$

where θ_i, $i = 1, 2, 3$, are the internal angles of E. We can easily see that $\lambda_{max}(K^E)$ and $\lambda_{max}(K)$ approach infinity if one of the angles θ_i, with $i = 1, 2, 3$, approaches 0 or π, because in these cases the value of $\cot\theta_i$ becomes arbitrarily large. We infer that both small and large angles deteriorate the matrix conditioning.

Conversely, suppose that there are no badly shaped triangles in the mesh, so that for every triangle E, we can bound $\lambda_{max}(K^E)$ from below by some small constant. In this case, we can bound $\lambda_{max}(K)$ from below by a small constant, and note that the lower bound on the smallest global eigenvalue $\lambda_{min}(K)$ is proportional to the area A_{min} of the smallest triangle, and $\kappa = \mathcal{O}(1/A_{min})$. If the triangles are of uniform size, it follows that $\kappa(K)$ is proportional to ℓ^{-2}, where we denote by ℓ the typical edge length. This deterioration of the conditioning of the matrix is, however, unavoidable, and shows that, due to the finite precision of the arithmetics, reducing the mesh size does not necessarily lead to an improvement of the numerical accuracy.

2.2 Element Quality Indicators

Element quality indicators are defined element-wise and then collected into a single mesh quality score in some kind of average. As a consequence, they are often specific to particular types of elements. In this section, we separately analyze indicators defined for the following types of elements:

- *Simplicial elements (triangles and tets):* triangular elements in a 2D-mesh with 3 nodes and 3 edges; tetrahedral elements in a 3D-mesh with 4 nodes, 6 edges, and 4 triangular faces;
- *Quads and hexes:* quadrangular elements in a 2D-mesh with 4 nodes and 4 edges; hexahedral elements in a 3D-mesh with 8 nodes, 12 edges, and 6 quadrangular faces;
- *Polytopes:* polygonal elements in a 2D-mesh that are not triangles nor quads; polyhedral elements in a 3D-mesh that are not tets nor hexes.

While a common requirement is that all mesh elements are non-degenerate (i.e., they have a positive area or volume), different numerical schemes may demand the fulfillment of additional requirements. For each type of element, we further distinguish between indicators that measure the interpolation error and those that measure the discretization error. The former are derived straightforwardly from error bounds or by computing some sort of distance from an ideal element (e.g., the equilateral triangle or the square). They typically involve geometric quantities like edges, areas, volumes, or diameters. The latter are derived from information about the stiffness matrix and therefore are formulated in terms of condition numbers, eigenvalues, or matrix norms.

2.2.1 Simplicial Elements

We start our analysis from simplicial elements because several indicators are firstly defined over them and then extended to quads/hexes or polytopes. The quality of a simplex is typically intended as its deviation from an equilateral triangle or tetrahedron (at least for isotropic problems). Pathological triangular elements can be either "needles" with one and only one angle close to 0, see Figure 2.5(a), or "flat elements" with one angle close to π, see Figure 2.5(b). The major difference between a triangle and a tet is that a new entity comes to play in the latter, namely, the solid angle. Pathological tetrahedra can be "needles" with one and only one solid angle close to 0, see Figure 2.5(c), "wedges" with one face with a very small area, see Figure 2.5(d), or "slivers" with a vertex almost on the plane containing its opposite face, see Figure 2.5(e).

Interpolation The simplest indicators for triangles are provided by the ratio between the inradius r and the circumradius R of the element, also called *radius ratio* or *aspect ratio*, or by the ratio between the longest edge $h_{e\,\max}$ and the

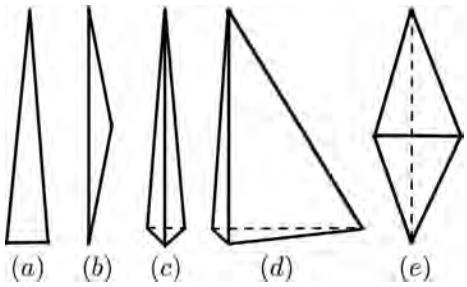


Figure 2.5: Examples of poorly shaped triangles and tetrahedra: (a,c) needles, (b) flat element, (d) wedge, (e) sliver.

inradius:

$$\varepsilon_1 = \frac{r}{R}, \quad \varepsilon_2 = \frac{r}{h_{e\,\max}}, \quad \varepsilon_3 = \frac{h_{e\,\max}}{R}. \tag{2.3}$$

A deep analysis of these indicators is provided by [129], where the authors study their extremum values and asymptotic behaviors. A direct formula for computing ε_3 for a tetrahedron in terms of edge lengths and face areas is provided in [108]. The edge ratio instead, intended as the ratio between the shortest and the longest edge, is not considered an interesting quality indicator as it does not vanish over flat elements like Figure 2.5(b) [14].

Equally important is the angle regularity, which imposes that the internal angles are not too different from each other. For triangles it can be devised by measuring the sinus of the smallest angle $\theta_{\min}$ or the widest angle $\theta_{\max}$, or a ratio between these two:

$$\vartheta_1 = \sin(\theta_{\min}), \quad \vartheta_2 = \sin(\theta_{\max}), \quad \vartheta_3 = \frac{\vartheta_1}{\vartheta_2}. \tag{2.4}$$

Several variations of the above indicators are overviewed in [68] and [164]. The computation of the minimum solid angle of a tetrahedron is not intuitive; a widely used formula is the one introduced in [108]:

$$\vartheta_4 = \min_{i=1,\dots,4} \{\sin(\theta_i/2)\}, \quad \text{where} \tag{2.5}$$
$$\sin(\theta_i/2) = \frac{12V}{\sqrt{\prod_{j,k\neq i,\ 1\leq j<k\leq 4} \left[(h_{eij} + h_{eik})^2 - h_{ejk}^2\right]}}.$$

The minimum dihedral angle can be measured as well, but it cannot detect the degenerate case of a needle-shaped tet like Figure 2.5(c) [14]. In fact, the dihedral angles of a tetrahedron remain more or less the same as the triangular face opposite to the pointed node is getting smaller and smaller. As already noted in Section 2.1.6, big angles are deleterious to the interpolation accuracy and the discretization error, while small angles may also be deleterious to the condition number of the stiffness matrix in finite element simulations [62].

Starting from tight error bounds for the interpolated function and its gradient, accurate quality indicators are derived in [152], for triangles:

$$\varrho_1 = \frac{A}{R^2}, \quad \varrho_1^{\nabla} = \frac{A}{(h_{e1}h_{e2}h_{e3})^{2/3}}, \tag{2.6}$$

and for tets:

$$\varrho_2 = \frac{V}{R^3}, \quad \varrho_2^{\nabla} = \frac{V \ \sum_{m=1}^{4} A_m}{\sum_{1\leq i<j\leq 4} A_i A_j h_{eij}}, \tag{2.7}$$

where the index ∇ indicates the quality indicator relative to the quantity $\|\nabla f - \nabla g\|_\infty$. We only report here the smooth and scale-invariant versions of such measures, referring to the original paper for a more detailed analysis.

Discretization Indicators based on the Jacobian matrix have been identified as the key measures relevant in the context of FEM [64] and can be also used for curvilinear elements [116]. The algebraic shape indicator from [108] is expressed as the ratio of the geometric mean to the arithmetic mean of the eigenvalues λ_i of $\mathcal{A}^T\mathcal{A}$:

$$\varsigma_1 = \frac{3\mathcal{J}^{2/3}}{\text{trace}(\mathcal{A}^T\mathcal{A})} = \frac{3\sqrt[3]{\lambda_1\lambda_2\lambda_3}}{\lambda_1+\lambda_2+\lambda_3} \tag{2.8}$$

For this reason, it can be found under the name of *mean ratio* [14]. An equivalent version is the *Frobenius ratio*, which consists of the inverse of the condition number of matrix $\mathcal{A}$ in the Frobenius norm [91]:

$$\varsigma_2 = \frac{d}{\kappa(\mathcal{A})} = \frac{d}{|\mathcal{A}|_F \ |\mathcal{A}^{-1}|_F}, \quad |\mathcal{A}|_F = \sqrt{\text{trace}(\mathcal{A}^T\mathcal{A})} \tag{2.9}$$

where d is the size of the matrix, i.e., the space dimension. The same indicator can be defined, with analog results, using any other type of matrix norm; see [129] for a comparison. Other specific shape quality indicators for triangles and tetrahedra based on the jacobian matrix are given in [92] under the formulas

$$\begin{aligned} \varsigma_4^{2\text{D}} &= \frac{\sqrt{3}\mathcal{J}}{\lambda_{11}+\lambda_{22}-\lambda_{12}} \\ \varsigma_4^{3\text{D}} &= \frac{3\sqrt[3]{2}\ \mathcal{J}^{2/3}}{\frac{3}{2}(\lambda_{11}+\lambda_{22}+\lambda_{33})-(\lambda_{12}+\lambda_{23}+\lambda_{13})}. \end{aligned} \tag{2.10}$$

In [152], two precise quality indicators are derived from tight bounds on the minimum and maximum eigenvalues of the stiffness matrix, respectively for triangles and tets:

$$\varsigma_3^{2\text{D}} = \frac{3A}{\sum_{i=1}^{3} h_{ei}^2}, \quad \varsigma_3^{3\text{D}} = \frac{4V}{\left(\sum_{i=1}^{4} A_i^2\right)^{3/4}}. \tag{2.11}$$

Similar indicators have been found in other works: in [17] the quality of a triangle is set as $4\sqrt{3}A/(h_{e1}^2 + h_{e2}^2 + h_{e3}^2)$, while [143] opts for the formulation $12\sqrt{3}A/(h_{e1}+h_{e2}+h_{e3})^2$. Analogously, the quality of a tet is measured in [111] by the formula $12\sqrt{3}V/(h_{e1}^2+h_{e2}^2+h_{e3}^2)^{3/2}$. In [62] a bound on the determinant of the stiffness matrix in the more general setting of general finite element spaces and general model equations is given by $\sum_{i=1}^{n+1} A_i^2/V$.

2.2.2 Quads and Hexes

While quads may be more efficient than triangles from a computational point of view, their properties make them a more difficult primitive to handle compared to triangles. In addition to edge lengths, quadrilaterals have one (and only one) more degree of freedom, provided by any of their four angles [128]. This extra degree of freedom gives birth to many pathological configurations, and even simple operations present a greater challenge. In Figure 2.6 we introduce some quantities specific to quadrilateral elements or faces. The *normal versor* associated to each vertex $\mathbf{v}_i$ is defined as:

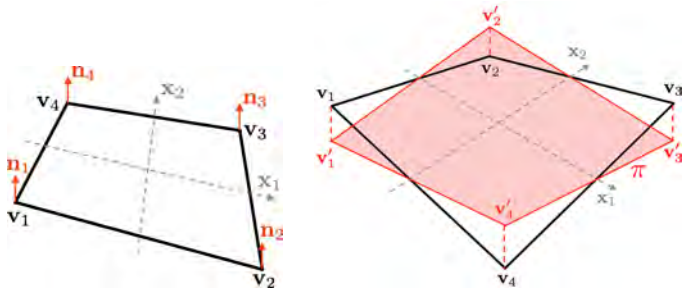
$$\mathbf{n}_i = \frac{e_i \times e_{i+1}}{\|e_i \times e_{i+1}\|},$$

while the two *principal axes* of the quad are given by:

$$\mathbf{x}_1 = (\mathbf{v}_2 - \mathbf{v}_1) + (\mathbf{v}_3 - \mathbf{v}_4), \quad \mathbf{x}_2 = (\mathbf{v}_3 - \mathbf{v}_2) + (\mathbf{v}_4 - \mathbf{v}_1).$$

If the quad is not planar, the principal axes individuate a *projection plane* π. The *cross derivatives* of the map from parametric to world space are oriented along

$$\mathbf{x}_{12} = (\mathbf{v}_1 - \mathbf{v}_2) + (\mathbf{v}_3 - \mathbf{v}_4) = (\mathbf{v}_1 - \mathbf{v}_4) + (\mathbf{v}_3 - \mathbf{v}_2) = \mathbf{x}_{21}.$$



(a) (b)

Figure 2.6: Elements of a quadrangle: (a) normal versors $\mathbf{n}_i$ and principal axes $\mathbf{x}_i$; (b) projection plane π defined by the principal axes.

First of all, a quad (intended as an element itself, or as a face of a hexahedral element) is not necessarily flat, and even planar quads may be non-convex. This

can be particularly crucial in architectural applications, where quad-meshes featuring this property are often categorized as *Planar-Quad* [35]. Planarity and convexity can be simply imposed as prerequisites for all the elements in a mesh, or be the subject of some quality indicators. Robinson's pioneering work [134] individuates four shape parameters that define a planar quadrilateral, shown in Figure 2.7(a-c): the *aspect ratio*, the *skew* angle and two distinct *tapers*. If the quad is not planar we can add a fifth parameter, namely *warpage*, to measure planarity (Figure 2.7(d)).

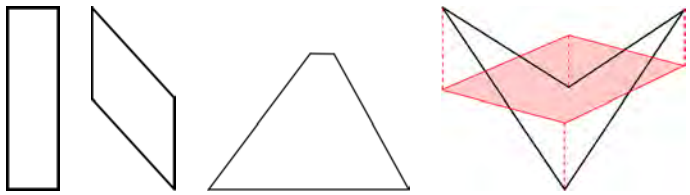


Figure 2.7: Quality of a quadrangular element: (a) low aspect ratio, (b) skewed element, (c) element with a significant taper along one direction, (d) warped element.

Interpolation The *warping* of a quad measures its distance from being planar. Given the projection plane π_Q defined by the principal axes, the warping of Q is measured by the distance between its vertices $\mathbf{v}_1, \ldots, \mathbf{v}_4$ and their projections $\mathbf{v}'_1, \ldots, \mathbf{v}'_4$ onto π_Q, see Figure 2.6(b):

$$\varphi_1 = \max_{i=1,\ldots,4} \left\{ \sin^{-1} \left(\frac{\|\mathbf{v}_i - \mathbf{v}'_i\|}{l} \right) \right\}, \tag{2.12}$$

where l is the half-length of the shortest between the two edges around $\mathbf{v}_i$ [135]. In the simplest sense, $\|\mathbf{v}_i - \mathbf{v}'_i\|$ is already an estimate of the warpage [136]. Alternatively, warping is measured in [164] through the curvatures of the coordinate surface on which the quad lies:

$$\varphi_2 = 1 - \min \left\{ (\mathbf{n}_1 \cdot \mathbf{n}_3)^3, (\mathbf{n}_2 \cdot \mathbf{n}_4)^3 \right\}. \tag{2.13}$$

Other formulations involve the coefficients of the second fundamental form or the mean and Gaussian curvatures of a coordinate surface containing the quad [107].

A formula for the *aspect ratio* of a quad can be derived from Indicator (2.3) for triangles. The opposite sides of each quad should have approximately equal lengths (this can be particularly crucial in contexts like physical simulations), or, for anisotropic approximation, a ratio best for approximation quality. However, the simple ratio between edges is not able to capture all kinds of pathologies, for instance, skewness. Another formula specific to quadrilateral quality measure

has been proposed by [135] and further optimized in [68]:

$$\varepsilon_1 = \max\left\{\frac{\|\mathbf{x}_1\|}{\|\mathbf{x}_2\|}, \frac{\|\mathbf{x}_2\|}{\|\mathbf{x}_1\|}\right\}, \tag{2.14}$$

while the quality measured by [72] involves the quad diagonals d_1, d_2 and the areas A_i^t of the triangles formed by the two edges incident to the i-th vertex:

$$\varepsilon_2 = \frac{\sqrt{\sum_{i=1}^4 h_{ei}^2}\ \max\{h_{\max}, d_1, d_2\}}{\min_i\{A_i^t\}}. \tag{2.15}$$

In [128] we find a specific formula to measure the aspect regularity, intended as the departure of the cell from a rhombus:

$$\varepsilon_3 = \frac{1}{1 - \left|1 - \frac{2\alpha}{\pi}\right|}, \tag{2.16}$$

where α is the average between two opposite angles, no matter which. A similar *stretch* indicator is given by the ratio between the minimum edge and the maximum diagonal [164]. In [107] the stretch is also formulated in terms of the covariant metric tensor.

Regarding the angle regularity, or *skewness*, internal angles should be close to 90 degrees, therefore we need to characterize the departure of a quad from a rectangle. This is measured by the absolute value of the cosine of the angle between the principal axes [135]:

$$\vartheta_1 = \left|\frac{\mathbf{x}_1}{\|\mathbf{x}_1\|} \cdot \frac{\mathbf{x}_2}{\|\mathbf{x}_2\|}\right|. \tag{2.17}$$

It can also be described through the cosine (or the cotangent) squared of the angle between the two tangent vectors defining the quad [107]. As pointed out in [92], such skew indicators reach their maximum also over elements that are not rectangles. To avoid this issue, the alternative indicator is proposed:

$$\vartheta_2 = \frac{4}{\sum_{i=1}^4 1/\sin(\theta_i)}. \tag{2.18}$$

A further quantity that can be measured in a quad is the so-called *taper*, conceived as the difference in length between opposite edges. The tapers relative to the two couples of edges in a quad are conventionally called the x- and y-component [134]. It can be quantified by the maximum ratio of cross derivative magnitude to principal axis magnitude [164]:

$$\tau = \frac{\|\mathbf{x}_{12}\|}{\min\{\|\mathbf{x}_1\|, \|\mathbf{x}_2\|\}}. \tag{2.19}$$

The quality indicators defined for quadrangular elements are straightforwardly extended to hexahedral cells. In [136] Robinson defines three skew ratios, three tapers, and six warpages for a hexahedral element, with the same procedure used for quads. Warping, stretching, skewness, and aspect regularity of a hex can be defined by the sum, the average, or the minimum/maximum of the respective indicators on its faces [107, 164].

Discretization Robinson's efforts amply demonstrate that quadrilaterals present much more complex geometric shapes to assess than triangles. This complexity perhaps motivates the predominant practice of using Jacobian determinants rather than purely geometric indicators [68].

For the *shape regularity*, we already saw how the Jacobian can be simply extended by considering the minimum between the values of $\mathcal{J}$ at the vertices and at the center of the quad. In this sense, it can be used in the same formulas as for triangles. In [91] we have the equivalent of Indicator (2.9) for quads ($n = 4$) and hexes ($n = 8$), together with an alternative formulation ς_2:

$$\varsigma_1 = \min_{i=1,\dots,n-1} \left\{ \frac{d}{\kappa(A_i)} \right\}, \quad \varsigma_2 = \frac{9n}{\sum_{i=1}^{n-1} \kappa(A_i)^2}. \tag{2.20}$$

More advanced Jacobian-based indicators exist under the names of *weighted* or *scaled* Jacobian [90, 89]:

$$\varsigma_3 = \frac{\min_i\{\mathcal{J}_i\}}{\max_i\{\mathcal{J}_i\}}, \quad \varsigma_4 = \min_i \left\{ \frac{\mathcal{J}_i}{h_{e_{i-1}} h_{e_i}} \right\}, \tag{2.21}$$

or the ones proposed in [92] for quads and hexes:

$$\varsigma_5^{\mathrm{2D}} = \frac{8}{\sum_{i=1}^{4} (\lambda_{11}^i + \lambda_{22}^i)/\mathcal{J}_i}, \tag{2.22}$$

$$\varsigma_5^{\mathrm{3D}} = \frac{24}{\sum_{i=1}^{8} (\lambda_{11}^i + \lambda_{22}^i + \lambda_{33}^i)/\mathcal{J}_i^{2/3}}. \tag{2.23}$$

Such indicators provide a measure of the variation in the Jacobian across the element, and they return a general measure of the quad shape.

Another diffused shape indicator is the *Oddy metric* [125], which measures the maximum deviation of the metric tensor at the corners of the quad. It can be defined in terms of the Jacobian matrices, or through the more explicit formula given in [164] for quads (and then extended to hexes):

$$\omega = \max_{i=1,\dots,4} \left\{ \frac{(h_{ei}^2 - h_{ei+1}^2)^2 + 4(h_{ei} h_{ei+1})^2}{2\|e_{i-1} \times e_i\|^2} \right\}. \tag{2.24}$$

2.2.3 Polytopes

The concept of the geometric quality of a cell becomes quite vague when the cell is a generic polygon or polyhedron. Most of the quality indicators defined above become meaningless when the elements have a generic number of vertices or faces, or they are at least very difficult to extend. For instance, the Jacobian can be extended to generic convex polygons (similarly to how we did for quads), but the Jacobian matrix in a polyhedron vertex is defined only if such vertex has three incident edges (e.g., a prism is fine, but a pyramid is not), and in general it is not defined for non-convex polytopes [91]. Trying to delimit the

field by excluding unlikely configurations, most mesh generators, and most numerical schemes only allow convex or star-shaped elements. On the other side, indicators defined for generic polygons can also be used to analyze the quality of triangular and quadrangular elements, being triangles and quads particular cases of polygons, and the same holds for polyhedra and tets/hexes.

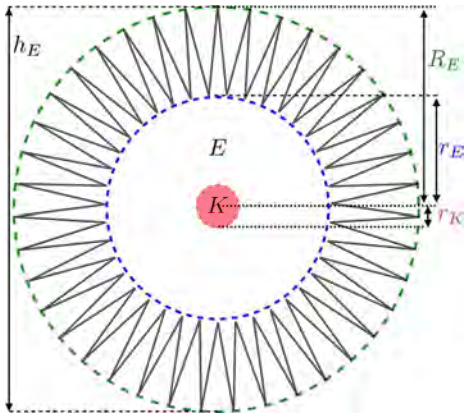


Figure 2.8: Polygonal element E with diameter h_E, inradius r_E, and circumradius R_E. Its kernel K has kernel radius r_K.

Interpolation The concept of *warping* can be extended naturally to polytopes. Indicator (2.12) remains valid, but computing the plane on which projecting the polygon (or the polygonal faces) is less immediate when the polygon is not a quad. Indicator (2.13) instead, can be extended to polygons by adding a proper number of pairs of normal versors $\mathbf{n}_i$.

The *aspect ratios* for polytopes are typically derived from Indicator (2.3), using the inradius r, the circumradius R, and opportunely replacing the maximum edge lengths with the diameter h_E [78], see Figure 2.8:

$$\varepsilon_1 = \frac{r}{R}, \quad \varepsilon_2 = \frac{2r}{h_E}, \quad \varepsilon_3 = \frac{h_E}{2R}. \tag{2.25}$$

Note that since we use the diameter instead of the edge, we should compare it with twice the radius. A number of variations of the above indicators (only for polygons) are overviewed in [11], where the authors analyze the impact of the different indicators on the performance of the Virtual Element Method. Alternatively, it is always possible to decompose a polytope into triangular or tetrahedral elements obtained connecting its vertices (generating a "matching simplicial submesh" [59]) and apply the simplicial shape ratio indicators.

The angle regularity of a generic polytope cannot be inferred by Indicators (2.4) and Indicator (2.5). In particular, problems occur with angles greater than π because the sinus becomes negative. A possible fix to this is to get rid

of the sinus and simply measuring the smallest and largest angles, in radians:

$$\vartheta_1 = \theta_{\min}, \quad \vartheta_2 = \theta_{\max}, \quad \vartheta_3 = \frac{\vartheta_1}{\vartheta_2}. \tag{2.26}$$

If we allow for the presence of non-convex polytopes, the ratio between the area and the perimeter $2\pi A_E/p_E^2$ (or the volume and the surface area) becomes much more interesting than it was for simplicial elements [11]. A number of indicators involve the *kernel* K of the element:

$$\varphi_1 = \frac{A_K}{A_E}, \quad \varphi_1 = \frac{V_K}{V_E}, \tag{2.27}$$

which returns 1 for convex polytopes, a value in $(0, 1)$ for concave polytopes, and 0 for star-shaped polytopes. It is also possible to measure the ratio between the inradius r_K of the circle inscribed in K and the circumradius R_E, see Figure 2.8.

Some works on polygonal element methods also require that the elements in the mesh have no "short edges", meaning that the ratio between the diameter and the shortest edge of the element is uniformly bounded across the mesh [20, 2, 26, 41]. Recently, researchers noticed that the "no short edge" requirement can be dropped in the case of Virtual Element Methods [43, 23]. But for other numerical methods, the situation is not completely clear and no short edges are still required at least in some theoretical analyses [78]. This, therefore, arises as a sufficient condition for the good behavior of the numerical method, but it can be also intended as a quality indicator. This is the approach of [160, 157], where the authors propose a mesh quality indicator specifically designed to work in couple with the Virtual Element Method (see Section 3). It is derived from the four geometrical assumptions required on a mesh for the convergence of the VEM:

$$\varrho_1 = A_K/A_E, \quad \varrho_2 = \min\{\sqrt{A_E}, h_{e\,\min}\}/h_E$$
$$\varrho_3 = 3/\#\left\{e \in \partial E\right\}, \quad \varrho_4 = \min_j \frac{\min_{e\in\mathcal{I}_E^j} h_e}{\max_{e\in\mathcal{I}_E^j} h_e},$$

and it is formulated as

$$\varrho = \sqrt{\frac{\varrho_1\varrho_2 + \varrho_1\varrho_3 + \varrho_1\varrho_4}{3}}. \tag{2.28}$$

Its 3D version is derived straightforwardly in [156], by simply extending the indicators to volumetric elements. This indicator will be the subject of Section 3.4.

Numerous attempts have been made to measure the quality of a polygon or polyhedron as "how far it is from a regular n-gon or n-hedron" [51, 57, 182] because a unitary regular n-gon (or n-hedron) is usually considered to be of high quality in the Euclidean metric. This implies, for instance, considering the ratio between the area (volume) of an element P and the area (volume) of a regular n-gon (n-hedron) with the same perimeter (surface area) as P, or

the difference between the angles of P and those of the ideal element. Unlike triangles, however, n-gons with $n > 3$ generally are not affine similar to a single reference n-gon [82]. Hence the idea of measuring the quality of a polygonal mesh by comparing its elements to regular n-gons through affine mappings does not work in general. To make it work, one needs to build proper mappings connecting arbitrary polygons with the reference ones, or re-define reference polygons of "good quality", or do both. However, polytopal element methods are defined to work over extremely generic elements, therefore there is apparently no reason to believe that the quality of a regular n-gon is the highest possible, or that they are the only configurations able to reach that quality level.

Discretization As already noted, the Jacobian matrix in a polyhedron vertex is defined only if such vertex has three incident edges. An exception can be made for the particular case of pyramidal elements, where the only vertex with valence higher than three is the top one. As a pyramid is completely defined by the tetrahedra associated with its base vertices, it is possible to define the Jacobian as an average of the Jacobians computed at the base vertices, which are all well-defined. It is therefore possible to compute Indicator (2.8) or (2.9) over "hybrid" meshes, that are mostly tetrahedral or hexahedral, with a small number of pyramids or prisms/wedges (whose vertices have all three incident edges) as transition elements to connect triangular and quadrilateral element faces [173, 99, 114].

The *Target-matrix paradigm* is a generalization of this Jacobian-based algebraic approach to mesh quality [93, 94]. The Jacobian matrices A_k are computed on a set of sample points and compared to the relative "Target" matrices W_k, which provide the local definition of quality. Its generality stems from the fact that the sample points can be defined for any kind of mesh and the quality indicator can be customized to match users' requirements. In the easiest case, the sample points coincide with the vertices of a simplicial mesh, and the indicator is built on the matrix $T = AW^{-1}$ as Indicator (2.9):

$$\varsigma = \frac{d}{\kappa(T)} = \frac{d}{|T|_F \ |T^{-1}|_F}. \tag{2.29}$$

A similar approach is adopted in [153], encoding the desired target size and shape of each element into a Riemannian metric instead of a matrix.

Another common type of polytopal meshes is Centroidal Voronoi Tessellations (CVT), which contain convex Voronoi cells with an undefined number of edges and faces. The quality of a CVT is often measured through some ad-hoc *CVT energy*. Given a CVT with centroids $\mathbf{x}_i$, $i = 1, \ldots, n$, defined over a domain $\Omega = \bigcup_{i=1}^{n} \Omega_i$ endowed with a density function $\rho(\mathbf{x}) > 0$, its global energy is defined as [60]:

$$\sum_{i=1}^{n} \int_{\Omega_i} \rho(\mathbf{x}) \|\mathbf{x} - \mathbf{x}_i\|^2 d\sigma.$$

The CVT energy accounts for the compactness of the cells, but it depends both on the number of cells and on the size of the shape. To overcome this problem, in [174] the author proposes to define a local quality indicator derived from the dimensionless second moment (or variance) of the single polytope, which measures how far the points inside E are spread out relative to its centroid $\tilde{x}$:

$$\vartheta = \frac{\int_E \|x - \tilde{x}\|^2 \mathrm{dx}}{d \ \left(\int_E \mathrm{dx}\right)^{(d+2)/d}}. \tag{2.30}$$

In this approach, the optimal element turns out to be the hexagon in 2D and the truncated octahedron in 3D.

2.2.4 Discussion

In Table 2.1 and Table 2.2 we summarize the main quality indicator found for 2D and 3D elements, respectively. The tables are not exhaustive: we reported a representative selection of the vast amount of the existing variants for each indicator. Moreover, we occasionally had to "force" the notation in favor of readability. In particular, the rows relative to "interpolation quality" and "shape regularity" contain all the indicators from the Interpolation and Discretization paragraphs (respectively) that did not fit into any other category.

We briefly comment on the two tables, also with respect to the considerations made in Section 2.1.4. Despite the name with which the indicators have been originally introduced, few of them are actually measures or metrics. For instance, they take values in ranges different from $[0, 1]$, but a deeper analysis of this aspect is beyond the scope of this work. However, according to the results presented in the respective papers, this does not prevent them to be accurate and reliable in assessing the quality of an element. Indicators reported in the rightmost columns are the most general: they are defined over any type of cell and they are often very similar between their 2D and 3D formulation. It is also notable how almost all the quality indicators for hexes are simply defined as a function of the quads quality indicators applied to their faces, exceptions made for those based on the Jacobian.

Many indicators have been shown to be equivalent when applied to the same type of elements. For instance, [108] proved that radius ratio, mean ratio, and sine of solid angle are equivalent over tets, meaning that they perform similarly well (or bad) over well (or bad) shaped elements. It has also been shown how the mean ratio performs equivalently to the Frobenius ratio [91], which in its turn is equivalent to the same ratio using any other matrix norm [129]. This is the reason why they belong to the same row in our tables. Chances are that more equivalences can be proven among other indicators, taking advantage of formulas that allow expressing algebraic quantities in terms of geometric ones and vice versa. Excluded from this point are case-specific indicators like the CVT energy or the VEM quality indicator.

The computational efforts required for some of the presented indicators are discussed in [127]. There exist a few works that compare the impact of the

Table 2.1: Taxonomy of quality indicators for 2D elements.

	triangles	quads	polygons
warping	-	$1-\min\left\{(\mathbf{n}_1\cdot\mathbf{n}_3)^3,(\mathbf{n}_2\cdot\mathbf{n}_4)^3\right\}$	$\max_{i=1,\ldots,n}\left\{\sin^{-1}\left(\frac{\|\mathbf{v}_i-\mathbf{v}_i'\|}{l}\right)\right\}$
aspect ratio	$\frac{r}{R}$, $\frac{r}{h_{e\,\max}}$, $\frac{h_{e\,\max}}{R}$	$\max\left\{\frac{\mathbf{x}_1}{\mathbf{x}_2},\frac{\mathbf{x}_2}{\mathbf{x}_1}\right\}$, $\left(1-\left\|1-\frac{2\alpha}{\pi}\right\|\right)^{-1}$	$\frac{r}{R}$, $\frac{2r}{h_E}$, $\frac{h_E}{2R}$, $\frac{r_K}{r_E}$, $\frac{A_K}{A_E}$
skewness	$\sin(\theta_{\min})$, $\sin(\theta_{\max})$, $\frac{\sin(\theta_{\min})}{\sin(\theta_{\max})}$	$\left\|\frac{\mathbf{x}_1}{\|\mathbf{x}_1\|}\cdot\frac{\mathbf{x}_2}{\|\mathbf{x}_2\|}\right\|$, $\frac{4}{\sum_{i=1}^{4}1/\sin(\theta_i)}$	$\theta_{\min}$, $\theta_{\max}$, $\frac{\theta_{\min}}{\theta_{\max}}$
taper	-	$\frac{\|\mathbf{x}_{12}\|}{\min\{\|\mathbf{x}_1\|,\|\mathbf{x}_2\|\}}$	-
interpolation quality	$\frac{A}{R^2}$, $\frac{A}{(h_{e1}h_{e2}h_{e3})^{2/3}}$	$\max_{i=1,\ldots,4}\left\{\frac{(h_{ei}^2-h_{ei+1}^2)^2+4(h_{ei}h_{ei+1})^2}{2\|e_{i-1}\times e_i\|^2}\right\}$	$\sqrt{\frac{g_1g_2+g_1g_3+g_1g_4}{3}}$
mean ratio	$\frac{\mathcal{J}}{\mathrm{trace}(\mathcal{A}^T\mathcal{A})}$, $\frac{2}{\kappa(\mathcal{A})}$	$\min_{i=1,\ldots,4}\left\{\frac{2}{\kappa(A_i)}\right\}$, $\frac{36}{\sum_{i=1}^{4}\kappa(A_i)^2}$	$\min_k\left\{\frac{2}{\kappa(T_k)}\right\}$
shape regularity	$\frac{3A}{\sum_{i=1}^{3}h_{ei}^2}$, $\frac{\sqrt{3}\mathcal{J}}{\lambda_{11}+\lambda_{22}-\lambda_{12}}$	$\frac{\min_i\{\mathcal{J}_i\}}{\max_i\{\mathcal{J}_i\}}$, $\frac{8}{\sum_{i=1}^{4}(\lambda_{11}{}^i+\lambda_{22}{}^i)/\mathcal{J}_i}$	$\frac{\int_E\|x-\tilde{x}\|^2\mathrm{d}x}{2\left(\int_E\mathrm{d}x\right)^2}$

Table 2.2: Taxonomy of quality indicators for 3D elements.

	tetrahedra	hexahedra	polyhedra
warping	-	sum/average/min/max of the faces	$\max_{i=1,\dots,n}\left\{\sin^{-1}\left(\frac{\|\mathbf{v}_i-\mathbf{v}_i'\|}{l}\right)\right\}$
aspect ratio	$\frac{r}{R}$, $\frac{r}{h_{e\max}}$, $\frac{h_{e\max}}{R}$	sum/average/min/max of the faces	$\frac{r}{R}$, $\frac{2r}{h_E}$, $\frac{h_E}{2R}$, $\frac{r_K}{r_E}$, $\frac{V_K}{V_E}$
skewness	$\min_{i=1,\dots,4}\{\sin(\theta_i/2)\}$	sum/average/min/max of the faces	$\min_{i=1,\dots,n}\{\theta_i\}$
taper	-	sum/average/min/max of the faces	-
interpolation quality	$\frac{V}{R^3}$, $\frac{V\ \sum_{m=1}^{4} A_m}{\sum_{1\le i<j\le 4} A_i A_j h_{eij}}$	sum/average/min/max of the faces	$\sqrt{\frac{\varrho_1\varrho_2+\varrho_1\varrho_3}{2}}$
mean ratio	$\frac{3\mathcal{J}^{2/3}}{\mathrm{trace}(\mathcal{A}^T\mathcal{A})}$, $\frac{3}{\kappa(\mathcal{A})}$	$\min_{i=1,\dots,8}\left\{\frac{3}{\kappa(A_i)}\right\}$, $\frac{72}{\sum_{i=1}^{8}\kappa(A_i)^2}$	$\min_k\left\{\frac{3}{\kappa(T_k)}\right\}$
shape regularity	$\frac{4V}{\left(\sum_{i=1}^{4} A_i^2\right)^{3/4}}$, $\frac{3\sqrt[3]{2}\ \mathcal{J}^{2/3}}{\frac{3}{2}(\lambda_{11}+\lambda_{22}+\lambda_{33})-(\lambda_{12}+\lambda_{23}+\lambda_{13})}$	$\frac{\min_i\{\mathcal{J}_i\}}{\max_i\{\mathcal{J}_i\}}$, $\frac{24}{\sum_{i=1}^{8}(\lambda_{11}^i+\lambda_{22}^i+\lambda_{33}^i)/\mathcal{J}_i^{2/3}}$	$\frac{\int_E \|x-\bar{x}\|^2 \mathrm{dx}}{3\left(\int_E \mathrm{dx}\right)^{5/2}}$

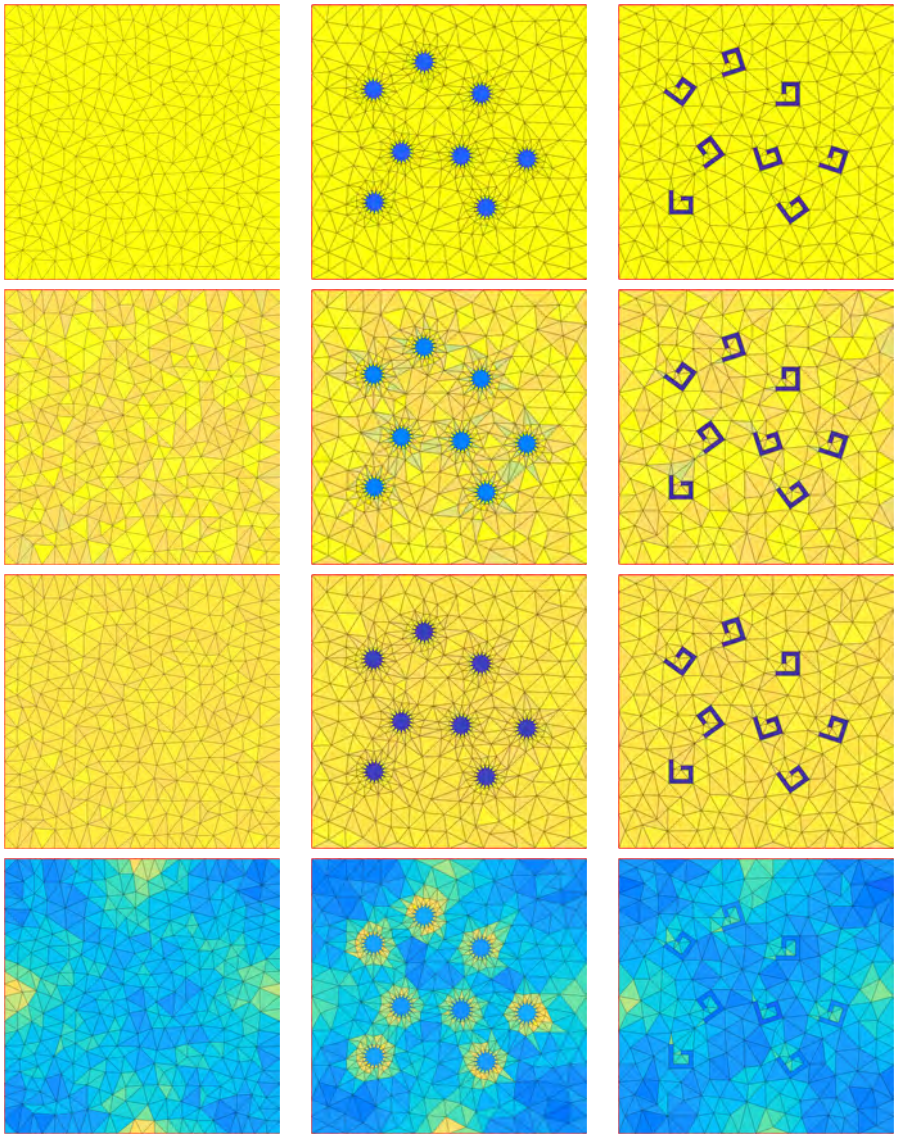
quality indicators on various steps of the simulation process, to evaluate how much the a-priori indicators actually correlate with the eventually observed performance of a mesh. A comparison of the performance of some specific quality indicators for hexahedral meshes in terms of stability and accuracy is given in [75], with both application-dependent and independent studies. The PEMesh graphical framework [48] allows the study of the correlations between the geometric properties of polygonal meshes and the numerical performances of PEM solvers. The user can design sets of polygonal meshes that increasingly stress some geometric properties and study the correlation between the performances of an external PEM solver and the geometric properties of the input mesh. More details on available software and datasets are provided in Section 2.4.

In Figure 2.9 we present a visual comparison of the differences between some quality indicators, and their relationship with the errors produced in a numerical simulation. In column (a) we consider a mesh containing only triangular elements, in (b) we introduce some star-shaped but non-convex polygons, and in (c) some non-star-shaped ones. In the top three rows, each element is colored with respect to its quality according to, respectively, the kernel-area ratio φ_1 from (2.27), the angle ratio ϑ_3 from (2.4), and the VEM quality indicator ρ from (2.28). Such indicators were chosen as representatives of the aspect ratio, skewness, and interpolation quality classes in Table 2.1. All indicators are re-scaled in the range $[0, 1]$, value 0 (low quality) is associated with the color blue, and value 1 (high quality) with the color yellow. In the bottom row, we color the elements with respect to the H^1 error produced by the solution of a Poisson problem with homogeneous Dirichlet boundary conditions through the linear VEM (the L^2 and the L^∞ errors produce very similar results):

$$\epsilon(E) = -\log \frac{|u - u_h|_{1,E}}{|u|_{1,\Omega}}, \quad \forall E \in \Omega_h,$$

where the negative sign is introduced so that high error values correspond in color to low-quality elements. Moreover, ϵ values are re-scaled in the range $[\min_{E\in\Omega_h} \epsilon(E), \max_{E\in\Omega_h} \epsilon(E)]$ in order to highlight differences between the elements.

We can immediately observe how all the considered quality indicators perfectly "recognize" the pathological elements in (a) and (b), assigning them a deep blue color. Regarding the triangular elements, the kernel-area ratio holds 1 for all the triangles in all three meshes, while the angle ratio and the VEM indicator exhibit some differences among elements, more accentuated in the first case. The errors in the bottom row present similar color patterns, which depend on the function we are approximating (in this case, a sinusoidal function). Besides this, it is still appreciable how poor-quality elements produce higher errors than their neighbors, highlighting a correlation between the quality indicators and the numerical performance.



(a) (b) (c)

Figure 2.9: Comparison between different quality indicators over three polygonal meshes (a-c): each element is colored with respect to its quality from blue (low) to yellow (high). From top to bottom row: kernel-area ratio; angle ratio; VEM quality indicator; H^1 norm of the approximation error produced by the linear VEM.

2.3 Mesh Quality Indicators

Mesh quality indicators observe the features of the mesh from a more general perspective. In addition to all the possible functions that we can define collecting element indicators in some sort of average or weighted norm across all the cells, there are indicators related to features that cannot be measured element-wise. In this section, we discuss possible strategies that are in the literature to define global quality indicators. Without providing an exhaustive survey of all the existing papers, we rather give a general taste of the aspects of a mesh that may affect its quality. We refer the interested reader to the original cited literature for the details and the precise formulation of such indicators.

2.3.1 Consistency

We can talk about different types of consistency [106]. A mesh is considered consistent with the *geometry* of the domain if it faithfully represents its boundary and eventual sharp features. It is consistent with the *topology* of the domain if the numbers of holes and connected components are preserved. It is consistent with the *solution* of the physical problem if it reflects the features of the function to be approximated.

For the accurate computation of the numerical solution of a PDE defined over the domain, the requirement of consistency with the geometry is indispensable. The mesh nodes must adequately approximate the original geometry, that is, the distance between any point of the domain and the nearest mesh node must not be too large, and this distance must approach zero when the number of nodes tends to infinity. This is particularly important for nodes, edges, or faces representing the boundary of the physical domain, as it allows the boundary conditions to be applied more easily and accurately.

The concept of topology consistency traditionally refers to the connectedness properties, or which node belongs to which element. When dealing with topological consistency the location in space of each node is irrelevant, while it is crucial to have correct adjacency rules among mesh elements [131]. In addition, the topology of the mesh must be coherent with that of the model. Besides simple consistency rules such as the Euler relation, the analysis of evolving surfaces, or model deformations, requires guaranteeing the consistency of the mesh with the model during the whole simulation process. This implies sampling a model consistently [54]; identifying small defects, holes, and handles [162, 13]; removing the topological noise [176]; and even topologically replicating the boundaries during optimization [148].

Another type of consistency is the one with the solution to the physical problem. The distribution of the mesh nodes and the form of the mesh cells should be dependent on the features of the solution, such as preferred directions (e.g., streamlines or vector fields), localized regions of very rapid variations (i.e., regions of high gradients), boundary and interior layers (e.g., in fluid dynamics, combustion, solidification, solid mechanics, and wave propagation), areas of high solution error of the numerical approximation [107]. In all these cases, it

may be helpful to subdivide the domain into smaller parts and mesh it locally because the uniform refinement of the entire domain may be very costly for multidimensional computations. Features lines, when present, should be explicitly represented as edge sequences. For example, sharp crease lines in mechanical objects, or lines where some attribute other than normals (e.g., color) varies. In this sense, the consistency with the solution is related to the consistency with the geometry: incorporating an input feature network into the mesh is not possible if the connectivity does not align with it, even refining the mesh [110] (see Figure 2.10).

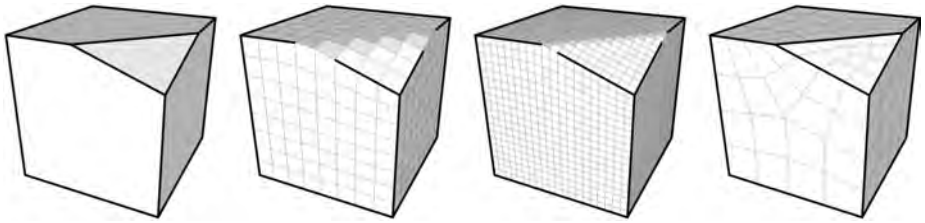


Figure 2.10: Key to feature preservation is the ability to align surface edges to the input network, carefully positioning mesh singularities (image from [110]).

Consistency indicators should measure the distance between the mesh and the domain, or the domain features. This is, for instance, a common theme in many mesh simplification strategies, where an explicit or implied geometric distance measure between the original and the simplified model is used to drive the simplification process [105]. A simple example of such measure is given by the Hausdorff distance [58] that was used to guarantee that the simplified model is less than a certain upper bound from the original model [139, 96]. Many other energy functions have been considered in the literature, as examples, we refer to [80, 179] and many of the optimization methods listed in Section 4.1. Often, approaches based on this paradigm admit both modifications of the local geometry and topology of the mesh.

2.3.2 Structure

By the structure of a mesh, we mean the way its elements are connected to each other. A common classification of the mesh structures is the one originally developed for quad meshes [35] and then extended to hex and hybrid meshes [130]. The authors call *regular* and *irregular* the structured and unstructured meshes already introduced in Section 2.1.1, and they also introduce two intermediate configurations. The *semi-regular* meshes are obtained by gluing in a conforming way a number of regular blocks. All vertices that are internal to a block are regular, while vertices at the edges or corners of a block may be irregular. The *valence semi-regular* meshes contain a limited amount of irregular vertices, but they are not connected in a way that induces a coarse block decomposition into a few regular blocks. In some sense, irregular, valence semi-regular,

semi-regular, and regular meshes can be seen as a continuum of cases, with an increasing degree of structure regularity. Depending on the applications, either a lower or higher degree of structure regularity is required, and this can be measured by the number of singular nodes in the mesh. A low number of singular nodes implies a simpler singular structure, which is more likely to allow for a block-structured mesh. Often, also the positioning of singular nodes is crucial: straight sequences of edges stemming from them (a.k.a. separatrices) should connect them in a graph that is as simple as possible, and they should appear in regions with a strong negative or positive Gaussian curvature (other than where it is needed to change resolution) [35]. Other works weaken the concept of regular node, considering regular also nodes with valence close to the fixed value [1].

Mesh connectivity is typically assumed to be conforming (i.e., free from T-junctions, see Figure 2.2) and pure, see Section 2.1.1. T-junctions are problematic in a number of ways, for instance, they lead to configurations in which it is not trivial to enforce continuity of the finite element functions across the edges or faces. In this sense, quality indicators can be provided by the number of T-junctions, or the number of non-standard elements (for instance, the number of non-quad cells in a quad-dominant mesh). The two above requirements can sometimes be loosened [123], but such topological freedom is not unlimited and may be bounded by the specific application [130].

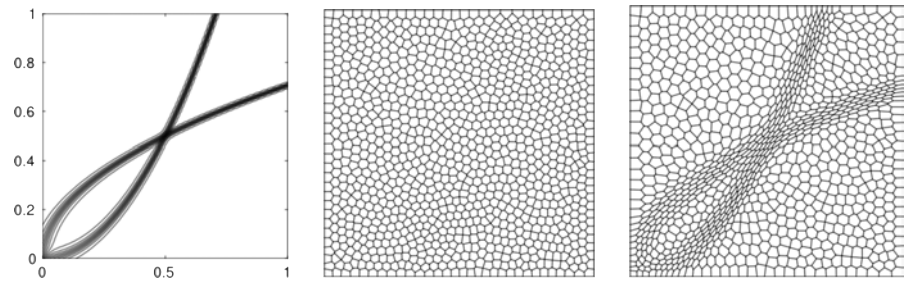
2.3.3 Distribution

Since nodes and elements represent the only contact points between the real world and the approximated copy that is going to be reproduced through the mesh, the way they are distributed in the domain affects the quality of the final result. If a uniform distribution of the information over the domain is needed, the element size (area, volumes, or sum of the faces areas) has to be similar across the mesh. If the mesh is pure, it is possible to measure the balance (or concentration, or density) through simpler quantities like edge lengths or diameters. In particular, in structured meshes, the change of cell size in a certain direction or along a curve can become an indicator [107].

However, it is often beneficial to let the tessellation density vary over the mesh, to adapt to local shape complexity, or to the solution of the numerical problem. In this case, the size of the elements cannot be required to be uniform across the mesh, but it is possible to ask at least for a smooth transition by imposing a gradual size change (neighboring elements cannot have too different sizes) [4], also called mesh gradation control [37]. Moreover, in order to allow for spatial transition through different levels of resolution, extra singular nodes must be included, unless T-junctions are introduced. Therefore, in many contexts, it is desirable to achieve the right trade-off between adaptivity and connectivity.

If the physical problem itself is anisotropic, it requires mesh generation to be guided by a prescribed anisotropy field. For instance, in computational fluid dynamics, it is desirable to squeeze the elements in the direction normal to the wing of a plane since the most significant physics occurs in the limit

layer [133, 82]. If the exact solution is of the type shown in Figure 2.11(a), a discretization like Figure 2.11(c) is likely to perform much better than the one in Figure 2.11(b), despite elements in the latter are more similar to regular polygons.



(a) (b) (c)

Figure 2.11: Example of anisotropic problem: (a) contour plot of the exact solution, (b,c) two possible discretizations [82].

2.4 Available Resources

This section summarizes several resources that are currently available for investigating the quality of a mesh, according to the different quality indicators. We also include some datasets publicly available online, that can be used to test the accuracy of a quality indicator in predicting the performance of a simulation. These lists are, obviously, in constant evolution.

Libraries

A vast amount of the quality indicators presented in Section 2.2.1 and Section 2.2.2 is detailed in the Verdict Geometric Quality Library [95]. The Verdict library became so popular that its indicators are currently implemented also in more general mesh processing toolkits like Cinolib [109] or HexaLab [40]. An implementation of the target-matrix paradigm is available in Mesquite [44]. Popular geometry processing libraries like CGAL [67], GMSH [77], or Geogram [103], also implement routines for some indicators, but they are often only defined for specific types of elements. The Mesh Repair website [12] contains a list of freely available software tools that perform the repairing of polygonal meshes with various sorts of defects and flaws. This list was published in the context of mesh repairing [13], which is somehow preliminary to mesh quality optimization, but we report it here for completeness.

The availability of commercial and open software libraries for FEMs and the related mesh generation methods is very wide, being it an established field with over 70 years of development on the subject. By way of example, we mention the *Discrete Methods & Related Tools* website [84], which contains links to a wide

range of commercial or open software (133 links), classified according to their functionality (FEM modelers, FEM solvers, creation of surface and volumetric meshes and grids, type of application, etc.).

More specific software that handles polygons and polyhedra are OpenMesh [34], OpenVolumeMesh [98], and PolyMesher [167]. The PEMesh framework [48] allows the computation of the quality indicators analyzed in [11] for generic polygons. The work by Schneider et al. proposed in the papers [146, 144, 147] is condensed into the PolyFEM library [145], a polyvalent C++ and Python FEM library. The Voro++ library is a C++ open-source software for the computation of the Voronoi diagram [140]. It contains the routines for constructing a single Voronoi cell, which is represented as a collection of vertices that are connected by edges, and to iterate the creation of cells.

Datasets

The ABC dataset [97] is a collection of about one million computer-aided design models proposed for research in geometric deep learning and applications, provided in different formats. Models were originally created with Onshape, and come in STEP and Parasolid format. Models in the Parasolid format can then be tessellated and saved in the STL format, obtaining triangle meshes with high resolution that carefully approximate the original geometry, but with low mesh quality, as the triangles may have bad aspect ratios and the sampling can be highly non-uniform. The GMSH library is then used to create a high-quality triangle mesh in OBJ format, with a uniform vertex distribution and most triangles have angles close to 60 degrees.

A popular reference for the generation of tetrahedral meshes from a triangle soup is TetWild [81]. Despite being a tetrahedral meshing technique, TetWild is also a tetrahedral dataset, since the authors provide the output of their algorithm applied to Thingi10k [180], a dataset of triangular meshes containing ten thousand models. A recent dataset, SimJEB [175], provides 381 tetrahedral meshes from CAD models. The HexaLab project [40] is a unified portal to visualize hexahedral meshes directly in a web browser as well as to download them. The recent survey [130] presents a wide list of 3D datasets that mainly contains pure hexahedral or hex-dominant meshes. Most of them are already included in Hexalab, notable exceptions are the HexMe tetrahedral dataset [19] and the MAMBO project [102], which stores geometric models for the study and comparison of hexahedral block meshing algorithms.

Regarding polytopal meshes, the Voro++ library also contains some examples of models tessellated with Voronoi cells. Similarly, the CVT dataset [46] is a FORTRAN90 open-source software that generates a Centroidal Voronoi Tessellation from a set of vertices based on the works [61, 47]. A dataset of polygonal meshes used for testing Indicator (2.28) is available on GitHub at [155], together with its 3D counterpart [158].

Chapter 3

The Virtual Elements Method

We introduce here the Virtual Element Method (VEM) in its enhanced form. All the details contained in the chapter can be retrieved in [21] and in [160]. However, we encourage the interested reader to find further details, extensions, and applications of VEM in [22].

3.1 Motivations

Since the first publication in 2013 and until today, the fundamental concept behind virtual elements is to associate each set of degrees of freedom (DOFs) with a smooth function, obtained as a solution of a local partial differential equation (PDE) within the element of the domain's tesselation. These functions, not necessarily polynomials, are typically not computable, and for this reason *virtual*, as it would be impractical to solve multiple PDE problems within each element. However, their projections onto polynomial spaces can be cheaply computed and subsequently utilized in formulating the discretized problem.

This powerful concept makes the VEM a Galerkin method with a large list of advantages, such as:

- the mesh elements have few geometrical restrictions. Indeed, the tesselation may contain generic polygon/polyhedrons, even concave and with aligned edges;
- mesh refinement or agglomeration operations result simpler and more natural, thanks to the capability of the virtual element to handle generic polytopal elements;
- with a proper definition of the local *virtual* functions, the discrete problem may retain selected properties of the original continuous problem. This

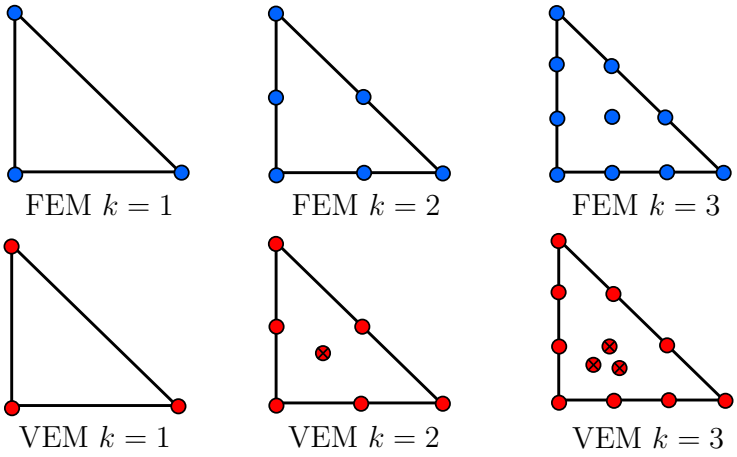


Figure 3.1: Degrees of freedom for FEM and VEM on triangles.

approach contrasts with standard methods, where these properties are only imposed in a weak sense;

- VEM can be seen as an extension of the classical Finite Element Method (FEM), thus the two methods may coexist in the same tesselation.

Due to these benefits and their remarkable flexibility, virtual elements have garnered significant interest within the engineering communities and are suitable for a large variety of applications.

3.1.1 VEM vs FEM

As highlighted in the previous section, the VEM we are considering in this course can be seen as an extension of the classic FEM.

By analogy, both VEM and FEM, as the other Galerkin methods, use a finite decomposition of the domain, associate a finite-dimensional space, and create a discretized bilinear form and a linear functional. Like FEM, VEM guarantees the uniqueness of the discrete solution of the PDE and optimal error estimates. For triangular (and quadrilateral) elements, finite and virtual element DOFs are the same on the border of the cell, see Figure 3.1; indeed, the VEM discrete approximation of order $k = 1$ on triangular mesh cells coincides with the continuous piece-wise linear finite element approximation. Moreover, as in FEM, VEM global bilinear form and right-hand side are defined by the sum of local contributions. Therefore, the final discrete linear system can be built with the typical FEM *local-global* scheme, where local matrices are created on each element of the tessellation and then assembled in the global matrix.

By contrast with the FEM, VEM is able to tackle tessellation with almost generic polygonal/polyhedral elements, with an arbitrary number of edges/faces. The direct consequence is that the finite element reference shape can not be used

in the virtual element implementation, as well as a unique quadrature formula shared by all the mesh elements. Furthermore, unlike the finite element well-known polynomial basis functions, the virtual basis functions are never computed on the elements and only their polynomial projection can be determined and used. Finally, virtual element approximation may preserve some selected properties of the continuous problem point-wisely, whereas finite element approximation can only impose them in a relaxed way.

Even on the meshes where FEM can be formulated, e.g., triangular and quadrilateral in 2D, comparing the performance of VEM and FEM is a non-trivial task. As illustrated in Figure 3.1, the local and global DOFs of the two spaces do not coincide when the order exceeds one. As a consequence, either we compare the two formulations over the same mesh but with different numbers of DOFs (higher for the VEM), or we compare them over two different meshes with the same number of DOFs. One should therefore choose, according to the context, which approach provides a more fair comparison.

In summary, as a first impression, VEM may frighten developers or consumers familiar with the simplicity of the finite element scheme, due to the introduction of projectors and additional geometrical information. However, with this course, we will see that the virtual element implementation and its use are not too far and more complex than the finite element one, thanks to the direct experience of the virtual element theory and practice.

3.2 Definitions

3.2.1 Problem Formulation

With the purpose of a gentle introduction to VEM, we examine the simple diffusion second-order elliptic differential problem

$$-\Delta u = f, \tag{3.1}$$

within a sufficiently smooth domain $\Omega \subset \mathbb{R}^d$, where d can be either 2 or 3. A zero Dirichlet boundary condition is imposed on $\Gamma_D := \partial\Omega$.

We express the equation in its variational form: given a functional space V, we seek $u \in V$ that satisfies

$$a(u, v) := \int_\Omega \nabla u \cdot \nabla v = \int_\Omega fv := F(v) \quad \forall v \in V, \tag{3.2}$$

where:

- V is the Sobolev space $H^1_0(\Omega) := \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_D\}$;
- $a : V \times V \to \mathbb{R}$ can be shown to be a bilinear form continuous and coercive on Ω;
- $F : V \to \mathbb{R}$ is a continuous linear functional if $f \in L^2(\Omega)$.

Under these assumptions, the Lax-Milgram lemma ensures that the problem (3.2) is well-posed (see, for example, [42]).

Virtual element discrete version of Problem (3.2) introduces a finite discrete approximation of Ω, namely $\Omega_h := \{E\}$ with $\bigcup_{E\in\Omega_h} E = \bar{\Omega}$, where E is a generic polygon/polyhedron in $\mathbb{R}^d$ and $\bar{\Omega}$ is the closure of Ω in $\mathbb{R}^d$. Every mesh Ω_h is labeled by the mesh size parameter $h := \max_{E\in\Omega_h}$, $h_E := \max_{x,y\in E} |x-y|$. We denote the set of all edges and faces on the boundary of E by $\mathcal{E}_E$ and $\mathcal{F}_E$. We set a positive integer k as the order of the discrete approximation, and we let V_k denote the finite-dimensional approximation space of order k. This space will be mathematically defined in the next subsection; for the moment we just anticipate that V_k contains the set of d-dimensional polynomials of degree less than or equal to k on E, which is denoted by $\mathbb{P}^d_k(E)$.

The discrete problem reads as: *Find $u_h \in V_k$, finite-dimensional subspace of V, such that:*

$$a_h(u_h, v_h) := \sum_{E\in\Omega_h} a_h^E(u_h, v_h) = \sum_{E\in\Omega_h} F_h^E(v_h) =: F_h(v_h) \quad \forall v_h \in V_k, \tag{3.3}$$

where $F_h^E(v_h)$ is the local approximation of $F(v_h)$ on each mesh element E. The bilinear discrete form $a_h^E(\cdot,\cdot)$ is a local approximation of $a^E(\cdot,\cdot) := a(\cdot,\cdot)_{|_E}$, which must satisfy the following assumptions:

- *k-consistency*: $\forall v \in V_k$ and $\forall p \in \mathbb{P}^d_k(E)$ it must hold
$$a_h^E(v,p) = a^E(v,p)_{|_E};$$
- *stability*: $\exists\ \alpha^*$, α_* constants, independent of h, s.t. $\forall v \in V_k$
$$\alpha_* a^E(v,v) \le a_h^E(v,v) \le \alpha_* a^E(v,v).$$

Under these assumptions, it is possible to prove that $a_h^E(\cdot,\cdot)$ is continuous and that the solution of discrete problem (3.3) is unique and convergent to the continuous solution u for h going to zero.

3.2.2 VEM Space

Now, all that remains is to define V_k for (enhanced) VEM. We discretize this variational problem (3.3) using the d-dimensional framework detailed in [2, 21]. Let E be a d-dimensional element in the domain partition Ω_h. We focus on the virtual element implementation which uses the standard basis of scaled monomials for all polynomial spaces, see [20].

We introduce the two elliptic projection operators on E that are defined for every integer $k > 0$ by:

- $\Pi_k^\nabla : H^1(E) \to \mathbb{P}^d_k(E)$,
$$\begin{cases} \big(\nabla p, \nabla(\Pi_k^\nabla v - v)\big)_E = 0, & \forall p \in \mathbb{P}^d_k(E), \\ \big(1, \Pi_k^\nabla v - v\big)_{\partial E} = 0, \end{cases}$$

- $\Pi_k^0 : L^2(E) \to \mathbb{P}_k^d(E)$,

$$\left(p, \Pi_k^0 v - v\right)_E = 0, \quad \forall p \in \mathbb{P}_k^d(E).$$

The local virtual element space of order k is defined on each $E \in \Omega_h$ as follows:

- For $d = 2$,

$$V_k^E = \big\{ v \in H^1(E) : \Delta v \in \mathbb{P}_k^2(E),\ v|_e \in \mathbb{P}_k^1(e)\ \forall e \in \mathcal{E}_E,\ v|_{\partial E} \in C^0(\partial E), \\ (p, v - \Pi_k^\nabla v)_E = 0\ \forall p \in \mathbb{P}_k^2(E)\ \setminus \mathbb{P}_{k-2}^2(E) \big\},$$

- For $d = 3$,

$$V_k^E = \big\{ v \in H^1(E) : \Delta v \in \mathbb{P}_k^3(E),\ v|_f \in V_k^f\ \forall f \in \mathcal{F}_E,\ v|_{\partial E} \in C^0(\partial E), \\ (p, v - \Pi_k^\nabla v)_E = 0\ \forall p \in \mathbb{P}_k^3(E)\ \setminus \mathbb{P}_{k-2}^3(E) \big\}.$$

Here, $\mathbb{P}_k^d(E)\ \setminus \mathbb{P}_{k-2}^d(E)$ denotes polynomials generated by monomials of degree k and $k-1$, and V_k^f is the virtual element space on a 2D face f.

Every function $v \in V_k^E$ can be uniquely identified by its DOFs:

- the value of v at each vertex of E;
- for $k > 1$, the value of v at the internal $k-1$ Gauss quadrature nodes of each edge $e \in \mathcal{E}_E$;
- if $k > 1$ and $d = 3$, the value of the internal scaled moments of v of order $k-2$ on each face $f \in \mathcal{F}_E$, namely

$$\int_f v\, p_{k-2} \quad \forall p_{k-2} \in \mathbb{P}_{k-2}^3(f);$$

- if $k > 1$, the value of the internal scaled moments of v of order $k-2$ on E, namely

$$\int_E v\, p_{k-2} \quad \forall p_{k-2} \in \mathbb{P}_{k-2}^3(E).$$

A representation of the DOFs for $d = 2$ on simple triangular cells is given in Figure 3.1, where the internal dots constitute the internal scaled moments.

Both projection operators Π_k^∇ and Π_k^0 can be computed on each E from these local DOFs. For example, the Π_k^∇ term $(\nabla p, \nabla v)_E$ can be determined by the DOFs integrating by parts:

$$(\nabla p, \nabla v)_E = \int_E \nabla p \cdot \nabla v = -\int_E v \Delta p + \int_{\partial E} v \nabla p \cdot n_e, \quad p \in \mathbb{P}_k^d(E)$$

and the two new integrals can be computed using the known moments of v inside the element and the piece-wise polynomial restriction of v on the edges.

We remark that the construction of Π_k^∇ on E requires recursively constructing Π_k^∇ on each face $f \subset \partial E$.

Consequently, we define the global virtual element space as:

$$V_k = \left\{ v \in C^0(\Omega) : v|_E \in V_k^E \ \forall E \in \Omega_h \right\}. \tag{3.4}$$

Each function $v \in V_k \subset V$ can be uniquely identified by the DOFs:

- The value of v at each mesh vertex;
- If $k > 1$, the value of v at the internal $k-1$ Gauss quadrature nodes of each mesh edge;
- If $k > 1$ and $d = 3$, the value of the internal scaled moments of v of order $k-2$ on each mesh face;
- If $k > 1$, the value of the internal scaled moments of v of order $k-2$ on each $E \in \Omega_h$.

3.2.3 VEM Operators

We approach the discretization of Problem (3.2) by using the virtual element space introduced in (3.4). The local bilinear discrete form $a_h^E : V_k^E \times V_k^E \to \mathbb{R}$ and the local linear form $F_h^E : V_k^E \to \mathbb{R}$ of discrete problem (3.3) can be defined for $k \geq 1$ as:

$$a_h^E(u, v) := a^E(\Pi_{k-1}^0 \nabla u, \Pi_{k-1}^0 \nabla v) + S^E(u - \Pi_k^\nabla u, v - \Pi_k^\nabla v); \tag{3.5}$$

and

$$F_h^E(v) := (f, \Pi_k^0 v).$$

The bilinear form $S^E(\cdot, \cdot)$ serves as the *stabilization term*, which can be any computable, symmetric, positive definite bilinear form that satisfies the stability condition

$$c_* a^E(v, v) \leq S^E(v, v) \leq c^* a^E(v, v), \ \forall v \in V_k^E \cap \ker(\Pi_k^\nabla),$$

for some pair of real, positive constants c_* and c^*. This term is fundamental to show the assumptions of k-consistency and stability of the bilinear form introduced in (3.5). Different versions of the S^E can be found in the VEM literature. In our numerical implementation, we use the standard and simple *dofi-dofi* stabilization proposed in [20]

$$S^E(v, w) := \sum_{i=1}^{\#\text{DOFs}} \text{dof}_i(v)\text{dof}_i(w)$$

where $\text{dof}_i(v)$ represents the i-th local DOF of function $v \in V_k^E$.

It is well known that the arbitrary choice of the stabilization term is one of the main factors that discourages classic FEM users to move to VEM. Indeed, in the last years, numerical researchers have proposed new virtual element frameworks *stabilization-free*, such as the one proposed in [28].

3.2.4 VEM Error Estimation

As introduced for FEM in the previous chapter, also for VEM the distance between the exact solution u of Problem (3.2) and virtual element approximation $u_h \in V_k$ solution of Problem (3.3) is controlled by

$$||u - u_h||_{0,\Omega} \leq C_0 h^{k+1} |u|_{k+1,\Omega}, \quad |u - u_h|_{1,\Omega} \leq C_1 h^k |u|_{k+1,\Omega}, \tag{3.6}$$

under sufficient regularity of the solution u and of the mesh Ω_h. See for example Theorem 3.7 in [22] for the general proof.

It is important to remark that, if we want to verify the error estimation (3.6) on numerical tests, in virtual elements we need to be careful because the basis functions are unknown inside the mesh elements E. For this reason, in contrast to FEM, the global and local relative numerical virtual element errors are redefined by

$$L^2\text{-Error} := \frac{\sum_{E \in \Omega_h} ||u - \Pi_k^0 u_h||_{0,E}}{||u||_{0,\Omega}}, \tag{3.7}$$

and by

$$H^1\text{-Error} := \frac{\sum_{E \in \Omega_h} ||\nabla u - \Pi_{k-1}^0 \nabla u_h||_{0,E}}{|u|_{1,\Omega}}. \tag{3.8}$$

Other equivalent definitions of numerical errors can be considered, exploiting the other virtual element projectors, e.g. Π_k^∇, see for example [28].

3.3 Mesh Requirements for the VEM

Various geometrical (or *regularity*) assumptions have been proposed in the literature to ensure the convergence of the VEM and optimal estimates of the approximation error with respect to different norms. These assumptions guarantee that all elements of all meshes in a given mesh family in the refinement process are sufficiently regular. They are defined for a single mesh Ω_h, but the conditions contained in them are required to hold independently of the mesh size h. As a consequence, when an assumption is imposed to a mesh family $\mathcal{T} = \{\Omega_h\}_h$, it has to be verified simultaneously by every $\Omega_h \in \mathcal{T}$.

These assumptions simply delimit the class of acceptable meshes for the VEM and are not meant to provide a quality score of a mesh or an element. To do this, from each geometrical assumption we derive an element quality indicator like the ones of Section 2.2, which measures how well a polytope $E \in \Omega_h$ meets the requirements of the assumption. The combination of these indicators will give us a mesh quality indicator specific to the VEM. For more details about the geometrical assumptions and the derived quality indicators, we refer the reader to the original papers [157, 160, 156].

3.3.1 Assumption G1

The main assumption, which systematically recurs in every VEM paper, is the star-shapedness of the mesh elements, already introduced in Section 2.2 among

the polytopal quality indicators.

Definition 3.3.1 ($\mathbf{G_1^{2D}}$) *There exists a real number $\rho \in (0,1)$, independent of h, such that every polygon $E \in \Omega_h$ is star-shaped with respect to a disc with radius*

$$r_E \geq \rho h_E.$$

We denote r_E the radius of the greatest possible inscribed disk in E and *star center* the center of such disk, where it exists. We stress the fact that being both ρ and h_E greater than zero, the radius r_E needs to be positive too, in accordance with the definition of star-shapedness. Assumption $\mathbf{G}_1^{2\mathrm{D}}$ is nothing but the polygonal extension of the classical *shape regularity condition* for triangular meshes. In fact, any triangular element E is star-shaped with respect to its maximum inscribed disk (the one with radius r_E), and the diameter h_E coincides with its longest edge. Assumption $\mathbf{G}_1^{2\mathrm{D}}$ is naturally extended to polyhedral meshes by imposing the star-shapedness both to the interior of an element and to each of its faces, with the same constant ρ.

Definition 3.3.2 ($\mathbf{G_1^{3D}}$) *There exists a real number $\rho \in (0,1)$, independent of h, such that every polyhedron $E \in \Omega_h$ is star-shaped with respect to a ball with radius*

$$r_E \geq \rho h_E,$$

and every face $f \in \partial E$ is star-shaped with respect to a disc with radius

$$r_f \geq \rho h_f.$$

Assumption $\mathbf{G}_1^{2\mathrm{D}}$ (and $\mathbf{G}_1^{3\mathrm{D}}$) plays a key role in most of the theoretical results regarding polytopal methods. For instance, it is required among the hypotheses of the Bramble-Hilbert lemma [42], an important result on polynomial approximation that is often used for building approximation estimates.

The $\mathbf{G_1}$-based Indicator From assumption $\mathbf{G}_1^{2\mathrm{D}}$ we derive the indicator $\varrho_1^{2\mathrm{D}}$, defined as the ratio between the area of the kernel and the total area of the element, already defined in Section 2.2 as *kernel ratio*:

$$\varrho_1^{2\mathrm{D}}(E) = \frac{|k(E)|}{|E|}.$$

We have $\varrho_1^{2\mathrm{D}}(E) = 1$ if E is convex, $\varrho_1^{2\mathrm{D}}(E) = 0$ if E is not star-shaped and $\varrho_1^{2\mathrm{D}}(E) \in (0,1)$ if E is concave but star-shaped. Therefore, $\varrho_1^{2\mathrm{D}}$ can be interpreted as an estimate of the value of the constant ρ from $\mathbf{G}_1^{2\mathrm{D}}$ on the polygon E. From assumption $\mathbf{G}_1^{3\mathrm{D}}$ instead, we derive the three-dimensional counterpart $\varrho_1^{3\mathrm{D}}$ of $\varrho_1^{2\mathrm{D}}$. The volumetric component is expressed through a kernel ratio that involves the volume of the kernel and the elemental volume. We want to have $\varrho_1^{3\mathrm{D}}(E) = 1$ if E and all its faces are convex, $\varrho_1^{3\mathrm{D}}(E) = 0$ if E or one of its faces

are not star-shaped, and $\varrho_1^{3\mathrm{D}}(E) \in (0,1)$ if E and all its faces are concave and star-shaped.

$$\varrho_1^{3\mathrm{D}}(E) = \frac{|k(E)|}{|E|} \prod_{f \in \partial E} \varrho_1^{2\mathrm{D}}(f).$$

The volumetric and the boundary components (the kernel ratio of each face) are multiplied so that, even if only one of them is zero (if a single face is not star-shaped), the whole product vanishes. We stress the fact that $\varrho_1^{2\mathrm{D}}$ and $\varrho_1^{3\mathrm{D}}$ are constant 0 for non-star-shaped cells, regardless of how far they are from being star-shaped, as the regularity constant ρ that we are approximating is zero in all those cases. It could be interesting to also measure the entity of the eventual non-star-shapedness of an element, but this information would have no relation with ρ.

3.3.2 Assumption G2

In the first VEM papers [20, 2, 26, 41], assumption $\mathbf{G}_1^{2\mathrm{D}}$ was followed by another condition on the length of the elemental edges:

Definition 3.3.3 ($\mathbf{G}_2^{\mathbf{2D}}$) *There exists a real number $\rho \in (0,1)$, independent of h, such that for every polygon $E \in \Omega_h$, the length h_e of every edge $e \in \partial E$ satisfies*

$$h_e \geq \rho h_E.$$

The Authors consider a single constant ρ for both assumption $\mathbf{G}_1^{2\mathrm{D}}$ and $\mathbf{G}_2^{2\mathrm{D}}$ and refer to it as the *mesh regularity constant* or *parameter*. Under assumption $\mathbf{G}_{1w}^{2\mathrm{D}}$ and $\mathbf{G}_2^{2\mathrm{D}}$, it can be proved [41] that the simplicial triangulation of E determined by the star-centers of $E_1, \ldots, E_N$ satisfies the *shape regularity* and the *minimum angle* conditions. The same holds under assumptions $\mathbf{G}_1^{2\mathrm{D}}$ and $\mathbf{G}_2^{2\mathrm{D}}$, as a special case of the previous statement. Moreover, assumption $\mathbf{G}_2^{2\mathrm{D}}$ implies that for $1 \leq j, k \leq N$ it holds $h_{E_j}/h_{E_k} \leq \rho^{-|j-k|}$. For a volumetric version of $\mathbf{G}_2^{2\mathrm{D}}$, we set a condition $h_f \geq \rho h_E$ on the element, and substitute it into the analogous condition on the faces $h_e \geq \rho h_f$, obtaining a triple inequality.

Definition 3.3.4 ($\mathbf{G}_2^{\mathbf{3D}}$) *There exists a real number $\rho \in (0,1)$, independent of h, such that for every polyhedron $E \in \Omega_h$, the length h_e of every edge e of every face f satisfies*

$$h_e \geq \rho h_f \geq \rho^2 h_E.$$

The $\mathbf{G}_2$-based Indicator The function $\varrho_2^{2\mathrm{D}}$ returns an estimate of the constant ρ introduced in $\mathbf{G}_2^{2\mathrm{D}}$. It is expressed through the ratio $|e|/h_E$, with the insertion of the quantity $\sqrt{|E|}$ in order to avoid pathological situations.

$$\varrho_2^{2\mathrm{D}}(E) = \frac{\min(\sqrt{|E|},\ \min_{e \in \partial E} |e|)}{h_E}.$$

If E is an equilateral triangle, we have that $\sqrt{|E|} = \frac{\sqrt[4]{3}}{2}|e|$ and $\min_{e\in\partial E}|e| = h_E = |e|$, that lead to $\varrho_2^{2\mathrm{D}}(E) = \frac{\sqrt[4]{3}}{2} \sim 0.65$. Likewise, in the case of a square, we have $\sqrt{|E|} = \min_{e\in\partial E}|e| = |e|$ and $h_E = \sqrt{2}\,|e|$, obtaining $\varrho_2^{2\mathrm{D}}(E) = \frac{1}{\sqrt{2}} \sim 0.7$. Therefore, from the point of view of this indicator, the quality of a squared element is slightly higher than that of an equilateral triangle. On the other side, $\varrho_2^{2\mathrm{D}}(E) \to 0$ if the length of at least one edge of E approaches 0 while the diameter does not vanish. The function $\varrho_2^{3\mathrm{D}}$ is an average of the volumetric constant ρ from $\mathbf{G}_2^{3\mathrm{D}}$, expressed through the ratio h_f/h_E, and all the boundary constants represented by the two-dimensional indicators $\varrho_2^{2\mathrm{D}}$:

$$\varrho_2^{3\mathrm{D}}(E) = \frac{1}{2}\left[\frac{\min(\sqrt[3]{|E|},\, \min_{f\in\partial E} h_f)}{h_E} + \frac{1}{\#\{f\in\partial E\}}\sum_{f\in\partial E}\varrho_2^{2\mathrm{D}}(f)\right].$$

3.3.3 Assumption G3

In the subsequent papers [26] and [43], assumption $\mathbf{G}_1^{2\mathrm{D}}$ is preserved, but assumption $\mathbf{G}_2^{2\mathrm{D}}$ is substituted by the alternative version:

Definition 3.3.5 ($\mathbf{G}_3^{2\mathbf{D}}$) *There exists a positive integer N, independent of h, such that the number of edges of every polygon $E \in \Omega_h$ is (uniformly) bounded by N.*

The Authors show how assumption $\mathbf{G}_2^{2\mathrm{D}}$ implies assumption $\mathbf{G}_3^{2\mathrm{D}}$, but assumption $\mathbf{G}_3^{2\mathrm{D}}$ is weaker than assumption $\mathbf{G}_2^{2\mathrm{D}}$, as it allows for edges arbitrarily small with respect to h_E. Both combinations $\mathbf{G}_1^{2\mathrm{D}}$+$\mathbf{G}_2^{2\mathrm{D}}$ and $\mathbf{G}_1^{2\mathrm{D}}$+$\mathbf{G}_3^{2\mathrm{D}}$ imply that the number of vertices of E and the minimum angle of the simplicial triangulation of E obtained by connecting all the vertices of E to its star-center are controlled by the constant ρ. Assumption $\mathbf{G}_3^{3\mathrm{D}}$ extends the bound from the number of edges to the number of edges and faces of every element and can be derived from $\mathbf{G}_1^{3\mathrm{D}}$ and $\mathbf{G}_2^{3\mathrm{D}}$ as well [2, Remark 11].

Definition 3.3.6 ($\mathbf{G}_3^{3\mathbf{D}}$) *There exists a positive integer N, independent of h, such that the number of edges and faces of every polyhedron $E \in \Omega_h$ is (uniformly) bounded by N.*

The $\mathbf{G}_3$-based Indicator Function $\varrho_3^{2\mathrm{D}}$ is a simple counter of the number of edges of the polygon, which penalizes elements with numerous edges as required by $\mathbf{G}_3^{2\mathrm{D}}$:

$$\varrho_3^{2\mathrm{D}}(E) = \frac{3}{\#\{e\in\partial E\}}.$$

It returns 1 if E is a triangle, which is therefore the optimal shape for this indicator, and it tends to 0 as the number of edges increases. Function $\varrho_3^{3\mathrm{D}}$ measures the number of edges and faces of a polyhedron, penalizing elements with numerous edges and faces as required by $\mathbf{G}_3^{3\mathrm{D}}$. In analogy to $\varrho_3^{2\mathrm{D}}$, it returns

1 if E is a tetrahedron, and it tends to 0 when the number of faces and edges increases.

$$\varrho_3^{3\mathrm{D}}(E) = \frac{1}{2}\left[\frac{4}{\#\{f \in \partial E\}} + \frac{1}{\#\{f \in \partial E\}} \sum_{f \in \partial E} \varrho_3^{2\mathrm{D}}(f)\right].$$

3.3.4 Assumption G4

Another step forward in the refinement of the geometrical assumptions was made by Beirão Da Veiga and Vacca in [23]. Besides assuming $\mathbf{G}_1^{2\mathrm{D}}$, the Authors imagine to "unwrap" the boundary ∂E of each element $E \in \Omega_h$ onto an interval of the real line, obtaining a one-dimensional mesh $\mathcal{I}_E$. The mesh $\mathcal{I}_E$ can be subdivided into a number of disjoint sub-meshes $\mathcal{I}_E^1, \ldots, \mathcal{I}_E^N$, corresponding to the edges of E. Then, the following condition is assumed.

Definition 3.3.7 ($\mathbf{G}_4^{2\mathrm{D}}$) *There exists a real number $\delta > 0$, independent of h, such that for every polygon $E \in \Omega_h$:*

- *the one-dimensional mesh $\mathcal{I}_E$ can be subdivided into a finite number N of disjoint sub-meshes $\mathcal{I}_E^1, \ldots, \mathcal{I}_E^N$;*
- *for each sub-mesh $\mathcal{I}_E^i$, $i = 1, \ldots, N$, it holds that*

$$\frac{\max_{e \in \mathcal{I}_E^i} |e|}{\min_{e \in \mathcal{I}_E^i} |e|} \leq \delta.$$

Each polygon E corresponds to a one-dimensional mesh $\mathcal{I}_E$, but a sub-mesh $\mathcal{I}_E^i \subset \mathcal{I}_E$ might contain more than one edge of E, cf. Figure 3.2. Therefore assumption $\mathbf{G}_4^{2\mathrm{D}}$ does not require a uniform bound on the number of edges in each element and does not exclude the presence of small edges. Mesh families created by agglomeration, cracking, gluing, etc.. of existing meshes are admissible according to $\mathbf{G}_4^{2\mathrm{D}}$.

We do not extend $\mathbf{G}_4^{2\mathrm{D}}$ to the three-dimensional case because, to the best of our knowledge, such extension has not yet been considered in the analysis of the three-dimensional formulation of the VEM. We point out that there is a fundamental difference between $\mathbf{G}_1^{2\mathrm{D}}$ ($\mathbf{G}_1^{3\mathrm{D}}$) on the one hand and the other assumptions on the other hand. For a fixed h, $\mathbf{G}_2^{2\mathrm{D}}$, $\mathbf{G}_3^{2\mathrm{D}}$, and $\mathbf{G}_4^{2\mathrm{D}}$ (or their respective 3D versions) are always satisfied, as a suitable ρ, N, or δ always exists for any given element. Such assumptions are therefore all about h-independent bounds for sequences of meshes. $\mathbf{G}_1^{2\mathrm{D}}$ and $\mathbf{G}_1^{3\mathrm{D}}$, by contrast, can be violated already by an individual non-star-shaped element. This difference will play a role in the definition of the mesh quality indicator in Section 3.4.

The $\mathbf{G}_4$-based Indicator The indicator $\varrho_4^{2\mathrm{D}}$ returns the minimum ratio between the smallest and the largest element in every $\mathcal{I}_E$, which is a measure of

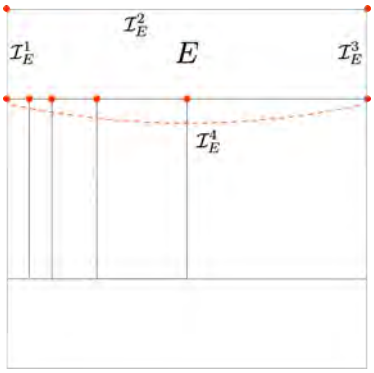


Figure 3.2: 1-dimensional mesh $\mathcal{I}_E = \{\mathcal{I}_E^1, \mathcal{I}_E^2, \mathcal{I}_E^3, \mathcal{I}_E^4\}$ relative to element E of a polygonal mesh. The sub-mesh $\mathcal{I}_E^4$ contains the sequence of all the aligned edges in the bottom of E.

the quasi-uniformity of $\mathcal{I}_E$ imposed by $\mathbf{G}_4^{2\mathrm{D}}$:

$$\varrho_4^{2\mathrm{D}}(E) = \min_i \frac{\min_{e \in \mathcal{I}_E^i} |e|}{\max_{e \in \mathcal{I}_E^i} |e|}.$$

If an element E does not present aligned edges, every sub-mesh $\mathcal{I}_E^i$ contains exactly one edge, and $\varrho_4^{2\mathrm{D}}$ is constantly equal to 1. In the presence of at least one sequence of aligned edges, $\varrho_4^{2\mathrm{D}}(E) = 1$ if all the edges in all the sequences have equal lengths. Last, $\varrho_4^{2\mathrm{D}}(E) \to 0$ as the edge lengths in even one single sequence (as we are considering the minimum among all the sequences) become more and more unbalanced. We do not consider $\varrho_4^{3\mathrm{D}}$, since assumption $\mathbf{G}_4^{2\mathrm{D}}$ has not been extended to the three-dimensional scenario.

3.4 Mesh Quality Indicator for the VEM

We can now define our mesh quality indicator, that is, a scalar function capable of providing insights into the behavior of the VEM over a particular sequence of meshes, before actually computing the approximated solutions. The quality indicator for two-dimensional meshes has been introduced in [157], and its three-dimensional version in [156]. The driving idea is that, instead of imposing an absolute condition (the geometrical assumptions) that a mesh can only satisfy or violate, we want to measure *how much* the mesh satisfies that condition. This approach is more accurate, as it captures small quality differences between meshes, and it does not exclude a priori all the particular cases of meshes only slightly outside the geometrical assumptions.

In the previous section, from each geometrical assumption $\mathbf{G}_i^{2\mathrm{D}}$, $i = 1, \ldots, 4$, we derived a scalar function $\varrho_i^{2\mathrm{D}} : \{E \subset \Omega_h\} \to [0, 1]$ which measures how well a polytope $E \in \Omega_h$ meets the requirements of $\mathbf{G}_i^{2\mathrm{D}}$. We set $\varrho^{2\mathrm{D}} = 0$ if E does not

respect $\mathbf{G}_i^{2\mathrm{D}}$, and the higher $\varrho^{2\mathrm{D}}$ the better E is shaped with respect to $\mathbf{G}_i^{2\mathrm{D}}$. Similar scalar functions are derived from the assumptions $\mathbf{G}_i^{3\mathrm{D}}$, $i = 1, \ldots, 3$ for three-dimensional meshes. In this case, $\varrho_i^{3\mathrm{D}} : \{E \subset \Omega_h\} \to [0, 1]$ measures both the quality of the interior of E (through a new volumetric operator) and the quality of its faces ∂E using the indicators $\varrho_i^{2\mathrm{D}}$.

Combining together $\varrho_1^{2\mathrm{D}}$, $\varrho_2^{2\mathrm{D}}$, $\varrho_3^{2\mathrm{D}}$ and $\varrho_4^{2\mathrm{D}}$, we define a global function $\varrho^{2\mathrm{D}} : \{E \subset \Omega_h\} \to [0, 1]$, which measures the overall quality of an element $E \in \Omega_h$, and identically for $\varrho^{3\mathrm{D}}$. In particular, we combine the indicators with the formula $\varrho_1^{2\mathrm{D}}\,\varrho_2^{2\mathrm{D}} + \varrho_1^{2\mathrm{D}}\,\varrho_3^{2\mathrm{D}} + \varrho_1^{2\mathrm{D}}\,\varrho_4^{2\mathrm{D}}$, as it reflects the way in which these assumptions are typically imposed: $\mathbf{G}_1^{2\mathrm{D}}$ and $\mathbf{G}_2^{2\mathrm{D}}$, $\mathbf{G}_1^{2\mathrm{D}}$ and $\mathbf{G}_3^{2\mathrm{D}}$ or $\mathbf{G}_1^{2\mathrm{D}}$ and $\mathbf{G}_4^{2\mathrm{D}}$ (but not, for instance, $\mathbf{G}_2^{2\mathrm{D}}$ and $\mathbf{G}_3^{2\mathrm{D}}$ simultaneously):

$$\begin{aligned} \varrho^{2\mathrm{D}}(E) &= \sqrt{\frac{\varrho_1^{2\mathrm{D}}(E)\varrho_2^{2\mathrm{D}}(E) + \varrho_1^{2\mathrm{D}}(E)\varrho_3^{2\mathrm{D}}(E) + \varrho_1^{2\mathrm{D}}(E)\varrho_4^{2\mathrm{D}}(E)}{3}} \\ \varrho^{3\mathrm{D}}(E) &= \sqrt{\frac{\varrho_1^{3\mathrm{D}}(E)\varrho_2^{3\mathrm{D}}(E) + \varrho_1^{3\mathrm{D}}(E)\varrho_3^{3\mathrm{D}}(E)}{2}} \end{aligned} \tag{3.9}$$

We also tried alternative formulations for $\varrho^{2\mathrm{D}}$, for instance considering only $\varrho_1^{2\mathrm{D}}$, $\varrho_3^{2\mathrm{D}}$, and $\varrho_4^{2\mathrm{D}}$ (as, technically, $\mathbf{G}_3^{2\mathrm{D}}$ is meant to replace $\mathbf{G}_2^{2\mathrm{D}}$), adding weights to give more importance to a specific indicator, or multiplying all the indicators instead of summing them. Among all these possible variants, however, the formulation (3.9) was the one providing the highest correlation between the quality score of a mesh and the numerical results of the VEM over it.

In Figure 3.3 we show a bunch of polygonal elements colored with respect to their VEM quality, i.e., the value of $\varrho^{2\mathrm{D}}$ over them. We have $\varrho^{2\mathrm{D}}(E) \approx 1$ if E is a perfectly-shaped element, e.g. an equilateral triangle or a square, even though squares are slightly penalized by indicator $\varrho_3^{2\mathrm{D}}$. As soon as the polygon gets stretched, or the number of edges grows, its quality decreases. In particular, note that the upper-rightmost rectangle has many vertices on its left side, which cause low values of $\varrho_2^{2\mathrm{D}}$ and $\varrho_4^{2\mathrm{D}}$. The lowest values are obtained for non-convex polygons, and we have $\varrho^{2\mathrm{D}}(E) = 0$ if and only if E is not star-shaped.

Similarly, in Figure 3.4 we show the quality of polyhedral elements. The highest quality is reached by the equilateral tetrahedron and the cube, even though cubes are slightly penalized by indicator $\varrho_3^{3\mathrm{D}}$. As soon as the polyhedron gets stretched, or the number of faces grows, its quality decreases. If the polyhedron is non-convex, we get low $\varrho_1^{3\mathrm{D}}$ values even if all its faces are convex, see the bottom-rightmost element.

We can also define a global function $\varrho : \{\Omega_h\}_h \to [0, 1]$ which measures the overall quality of a mesh Ω_h:

$$\varrho(\Omega_h) = \sqrt{\frac{1}{\#\{E \in \Omega_h\}} \sum_{E \in \Omega_h} \varrho(E)}, \tag{3.10}$$

using the general notation ϱ for indicating $\varrho^{2\mathrm{D}}$ or $\varrho^{3\mathrm{D}}$ according to the context. We have $\varrho(\Omega_h) \to 1$ if Ω_h contains only perfectly-shaped elements, $\varrho(\Omega_h) = 0$

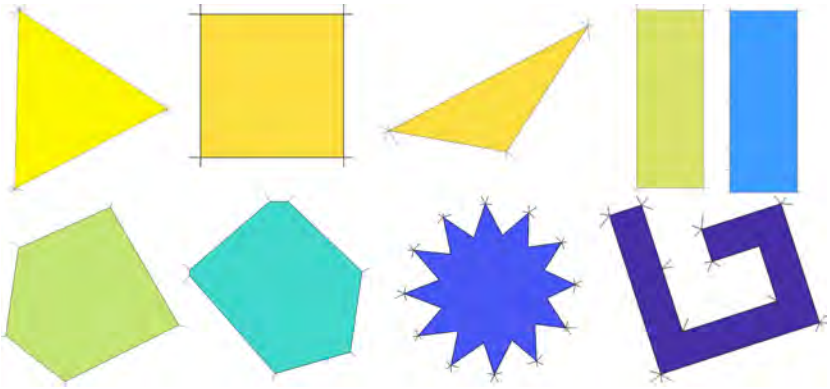


Figure 3.3: Polygonal elements colored with respect to their VEM quality, from blue ($\varrho^{2D} = 0$) to yellow ($\varrho^{2D} = 1$).

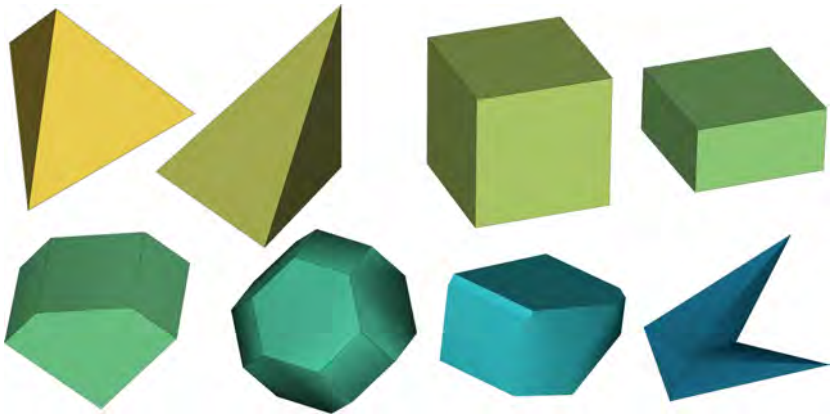


Figure 3.4: Polyhedral elements colored with respect to their VEM quality, from blue ($\varrho^{3D} = 0$) to yellow ($\varrho^{3D} = 1$).

if and only if Ω_h contains only non-star-shaped polytopes (or with non-star-shaped faces), and $0 < \varrho(\Omega_h) < 1$ otherwise. We point out that the indicator ϱ only depends on the geometrical properties of the mesh elements (because the same holds for all the $\varrho_i^{2\mathrm{D}}$ and $\varrho_i^{3\mathrm{D}}$), and it is problem-independent (but $\varrho_2^{2\mathrm{D}}$ and $\varrho_2^{3\mathrm{D}}$ penalize anisotropic configurations). Therefore, they can be computed before applying the VEM or any other numerical scheme. Given a dataset $\mathcal{D}$, we can study the behavior of $\varrho(\Omega_h)$ for $h \to 0$ and determine the quality of the meshes through the refinement process without having to solve the numerical problem from time to time.

The construction of ϱ is easily upgradeable to future developments. Whenever new assumptions on the features of a mesh should come up, one simply needs to introduce a new function $\varrho_i^{2\mathrm{D}}$ or $\varrho_i^{3\mathrm{D}}$ that measures the fulfillment of the new assumption and opportunely insert it into (3.9). This could be done, for instance, if a valid extension of $\mathbf{G}_4^{2\mathrm{D}}$ to the three-dimensional case should appear in the literature. Similarly, the global indicators are easily extendable to other numerical schemes by substituting the assumptions designed for the VEM with the assumptions on the new scheme, and defining new relative indicators $\varrho_i^{2\mathrm{D}}$ and $\varrho_i^{3\mathrm{D}}$.

3.4.1 Correlation With VEM Performance

We conclude this section by reporting some results from [157] and [156] that show the correlation between the quality indicator and the performance of the VEM. In those papers, the former for 2D meshes and the latter for 3D meshes, we consider several sequences of meshes with decreasing mesh size, called *datasets*.

To test the accuracy of the quality indicator ϱ defined in (3.10) we evaluate it over each mesh of each dataset, and according to its behavior we make some predictions on how the VEM will perform in solving the discrete Poisson problem. We recall that for an ideal dataset made by meshes containing only equilateral triangles/tetrahedra, ϱ would be constantly equal to 1. We assume this value as a reference for the other datasets: the closer ϱ on a dataset to the line $y = 1$, the smaller the approximation error that we expect that dataset to produce. Moreover, the ϱ slope is indicative of the convergence rate. Since an ideal dataset would produce a horizontal line, the more negative the slope, the worse the convergence rate that we expect. A positive trend instead should indicate a convergence rate higher than the one obtained with an equilateral triangular/tetrahedral mesh, that is, higher than the theoretical estimates.

For each dataset we plot the approximation error $u - u_h$ as the number of DOFs increases using the H^1-seminorm and L^2-norm as defined in (3.7), (3.8), and the relative L^∞-norm properly redefined for virtual element approximation. The optimal convergence rate of the method, provided by the estimates (3.6), is indicated by the slope of the reference triangle. In the case of the L^∞-norm, we do not have such theoretical results.

The exercise we propose is to first analyze the values of ϱ on a dataset, computed before solving the problem, and make some predictions on the behavior

of the VEM over it in terms of convergence rate and error magnitude. Then, looking at the approximation errors actually produced by that dataset, search for correspondences between ϱ and the errors, checking the accuracy of the prediction. We report here only a significant selection of the considered datasets and the performance only for the VEM of order $k = 1$. We address the reader to the original papers for more details about the datasets, which can also be downloaded at [155, 158], and for the full analysis also with order $k = 2, 3$.

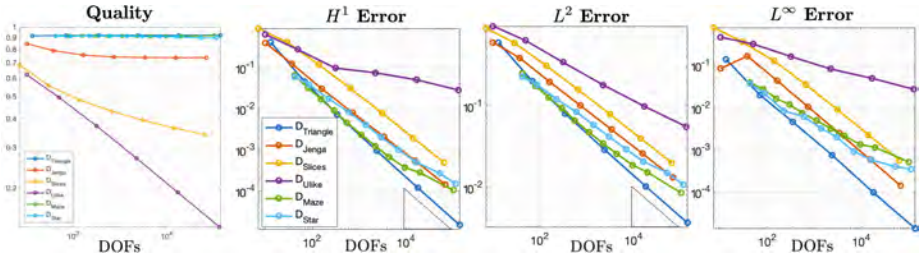


Figure 3.5: Indicator ϱ computed on 2D datasets and VEM performance over them.

For the 2D case, we report six datasets ($\mathcal{D}_{\text{Triangle}}$, $\mathcal{D}_{\text{Jenga}}$, $\mathcal{D}_{\text{Slices}}$, $\mathcal{D}_{\text{Ulike}}$, $\mathcal{D}_{\text{Maze}}$, and $\mathcal{D}_{\text{Star}}$) containing six polygonal meshes each. In Figure 3.5 we can see the mesh quality indicator ϱ and the H^1, L^2, and L^∞ errors produced by the VEM of order $k = 1$ solving the discrete Poisson problem over each mesh, using as ground-truth the function

$$u(x, y) = \frac{\sin(\pi x)\sin(\pi y)}{2\pi^2}, \quad (x, y) \in \Omega = (0, 1)^2.$$

The quality of meshes belonging to $\mathcal{D}_{\text{Triangle}}$ is almost constant and very close to 1, thus foreseeing excellent convergence rates and low errors. The quality of $\mathcal{D}_{\text{Maze}}$ and $\mathcal{D}_{\text{Star}}$ is close to that of $\mathcal{D}_{\text{Triangle}}$, hence we expect them to behave similarly. This is confirmed by the error plots: $\mathcal{D}_{\text{Maze}}$ and $\mathcal{D}_{\text{Star}}$ are almost coincident and very close to $\mathcal{D}_{\text{Triangle}}$ until the very last meshes. The $\mathcal{D}_{\text{Jenga}}$ quality anticipates a perfect convergence rate but greater error values with respect to the previous three, and again this behavior is respected in the error plots. The quality of dataset $\mathcal{D}_{\text{Slices}}$ is quite distant from the ideal value of 1. Importantly, it keeps decreasing from mesh to mesh, but the curve rate allows us to assume that it may flatten within a couple more meshes. We notice that this dataset produces an error significantly higher than the previous ones, but the convergence rates are very similar to the theoretical estimates. Last, the ϱ values predict huge errors and a completely wrong convergence rate for $\mathcal{D}_{\text{Ulike}}$. This dataset is actually the one with the worst performance, as it exhibits incorrect convergence rates for all types of norm (see in particular the H^1 and L^∞ errors).

For the 3D case, we report six datasets ($\mathcal{D}_{\text{tet-uniform}}$, $\mathcal{D}_{\text{tet-anisotropic}}$, $\mathcal{D}_{\text{tet-parallel}}$, $\mathcal{D}_{\text{tet-bcl}}$, $\mathcal{D}_{\text{tet-poisson}}$, and $\mathcal{D}_{\text{tet-random}}$) containing six tetrahedral meshes each.

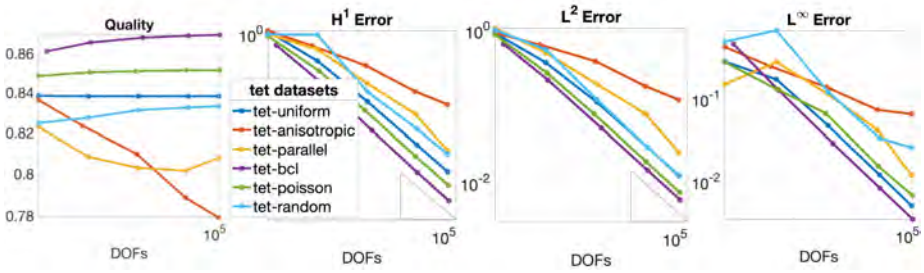


Figure 3.6: Indicator ϱ computed on 3D datasets and VEM performance over them.

In Figure 3.6 we can see the mesh quality indicator ϱ and the H^1, L^2, and L^∞ errors produced by the VEM of order $k = 1$ solving the discrete Poisson problem over each mesh, using as ground-truth the function

$$u(x, y, z) = x^3y^2z + x\sin(2\pi xy)\sin(2\pi yz)\sin(2\pi z),\ (x, y, z) \in \Omega = (0, 1)^3.$$

Looking at the quality plot (the leftmost) we would say that the VEM should converge with the optimal rate over almost all datasets, as their ϱ tends to get horizontal. One exception is $\mathcal{D}_{\text{tet}-\text{anisotropic}}$, which has a negative trend and therefore is not expected to converge properly. We can also observe how the slope of $\mathcal{D}_{\text{tet}-\text{parallel}}$ becomes positive in the last mesh: this should indicate a more than optimal convergence rate. Regarding the error magnitude, represented by the overall distance from the top of the plot, we can predict that $\mathcal{D}_{\text{tet}-\text{bcl}}$ will produce the smallest errors, being the one with the highest quality. This is reasonable because this dataset is composed mainly of equilateral tetrahedra. We can then order decreasingly the other datasets according to their quality: $\mathcal{D}_{\text{tet}-\text{poisson}}$, $\mathcal{D}_{\text{tet}-\text{uniform}}$, $\mathcal{D}_{\text{tet}-\text{random}}$, $\mathcal{D}_{\text{tet}-\text{parallel}}$, and $\mathcal{D}_{\text{tet}-\text{anisotropic}}$, and we expect the errors' magnitudes to behave accordingly.

The convergence rates of the tetrahedral datasets in the error plots faithfully respect all the above considerations. In both H^1 and L^2-norms, the method converges with the optimal rate (the one suggested by the reference triangle) over all datasets, except for $\mathcal{D}_{\text{tet}-\text{anisotropic}}$, and datasets $\mathcal{D}_{\text{tet}-\text{parallel}}$ superconverges in the last mesh. We checked the condition numbers for the linear systems that are built on the meshes of $\mathcal{D}_{\text{tet}-\text{anisotropic}}$ and we verified that their values are reasonably small, i.e., in the range $[1, 10^6]$. These values are comparable to the condition numbers seen in the other datasets. In the L^∞-norm, the situation is similar, even if $\mathcal{D}_{\text{tet}-\text{random}}$ has an unexpected peak in the last mesh. The error magnitudes, i.e. the distance of the line from the $y-$axis, perfectly follow the ordering suggested by the quality plot. The dataset which produces the smallest errors is $\mathcal{D}_{\text{tet}-\text{bcl}}$. After that, in the H^1 and L^2 plots we have $\mathcal{D}_{\text{tet}-\text{poisson}}$, $\mathcal{D}_{\text{tet}-\text{uniform}}$ and $\mathcal{D}_{\text{tet}-\text{random}}$, which tend to become very close, then $\mathcal{D}_{\text{tet}-\text{parallel}}$ and last $\mathcal{D}_{\text{tet}-\text{anisotropic}}$. The situation slightly changes if we look at the L^∞ error: in this case, $\mathcal{D}_{\text{tet}-\text{uniform}}$ performs better

than $\mathcal{D}_{\mathrm{tet-poisson}}$, probably due to a bunch of poor quality elements that do not particularly affect the overall accuracy of the method.

Summing up these results, we conclude that the mesh quality indicator is able to properly predict the behavior of the VEM, both in terms of convergence rate and error magnitude, up to a certain precision. It shows up to be particularly accurate when compared to the H^1 and the L^2-norms of the error, while it does not always manage to capture the oscillations of the L^∞-norm.

Chapter 4

Optimizing Mesh Quality

Despite extensive research and notable achievements, developing techniques for generating meshes with suitable geometrical properties remains an ongoing endeavor. Contemporary meshing algorithms typically generate an initial mesh, ideally dominated by well-shaped elements, and optionally execute optimization steps to enhance the geometrical quality of the elements [113]. Pivotal to these optimizations is the definition of the concept of "quality" of a mesh and its elements.

These operations are even more crucial when we deal with polytopal meshes. The potentially unlimited freedom that we have when creating a polytopal mesh, makes it complicated to make choices from time to time. In fact, the main methods for polytopal mesh generation from scratch are all based on Voronoi tessellations [178]. However, the development of numerical schemes that support generic polytopal elements [20, 45, 24, 165, 59, 32] and the great advantages that they provide over classical methods, make it more and more urgent to devise strategies for generating and optimizing polytopal meshes.

Polytopal mesh optimization is emerging [8, 7, 161], both as a way to improve the quality of a given mesh and as a way to generate new polytopal meshes starting from standard (triangular / tetrahedral / quadrangular / hexahedral) ones. This section collects a survey of the most common techniques for improving or optimizing the quality of a mesh, derived from [159]. In the second part we also present one particular mesh optimization algorithm introduced in [161, 154].

4.1 How to Get a Quality Mesh

The local and global quality indicators presented in Section 2.2 and Section 2.3 are commonly used to investigate and improve mesh quality. Some of them can be integrated into the mesh generation process (e.g., in advancing front methods [112]) so that at the end of the pipeline, the output mesh already reaches a certain quality level. This is particularly true for 2D-meshes and for indicators

concerning properties like structure or connectivity, which may be essential requirements in some applications. However, the new placement of vertices, edges, and faces can be implicitly treated as an optimization problem. For instance, the edge collapse operation can be stated as the problem of choosing the position of the new vertex such that an associated objective function (sometimes called the "edge cost") is minimized.

In this sense, in Computer Graphics there is a large literature of methods that set the problem of improving the quality of a mesh as an optimization problem with respect to a functional minimization. Among the others, we refer to the seminal work of Hoppe et al [80]. Similarly, the majority of volumetric meshing algorithms, employ a two-step process [130]. The first step generates an initial mesh, which is expected to be dominated by well-shaped elements, but often also contains some poorly-shaped or even degenerate elements. This step is typically followed by an optimization step, whose goal is to maximize the quality of the mesh elements and remove any degeneracy while keeping the meshed domain boundary intact. As a mesh is defined by the nodal connections of the elements and the coordinates of the nodes, optimization methods can be broadly grouped into two categories [113]: (i) geometrical methods that involve a change of element shapes by means of shifting of nodal points; and (ii) topological methods that involve a change in the element connections to the nodal points. However, there is no reason why topological and geometrical operations cannot be put together to form even more effective and well-balanced optimization schemes.

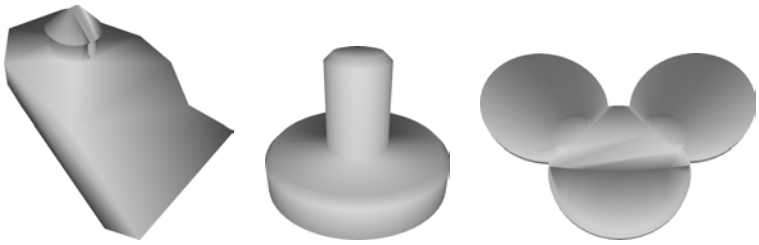


Figure 4.1: Impact of long and thin triangles on the model rendering (models taken from the Thingi dataset [180]). Artificial black parts are visible even in regions that are geometrically flat.

Broadly speaking, operations used for repairing and eliminating the noise and defects of a mesh are also included in the optimization of a mesh. In this paper we do not intend to go into detail about all processing techniques involved in mesh smoothing, repairing, etc.; for more details, the reader can consult [38, 13] and the Eurographics tutorial [49]. As a flavor of the relevance of mesh quality for practical applications, we mention rendering and manufacturing. For instance, the poor quality of some elements may generate a worsening of the 3D model rendering. Figure 4.1 shows the effect of some triangles with bad aspect ratio on the model rendering (the visualization is done with the

standard per-vertex rendering available in the Meshlab suite [56]). On this type of models, even the application of a texture would require remeshing to avoid image distortion. In the case of 3D printing, STL mesh models with poor quality elements could make the printing software either fail or produce machine instructions that lead to printing failures [10]. In the left of Figure 4.2 we show a complex model originally with several heterogeneous defects; the distorted elements highlighted in the red box are then remeshed in the picture on the right, thus making the model printable.



Figure 4.2: Left: a complex 3D model with some distorted elements (e.g., needles) highlighted in the red box. Right: after remeshing, the model can be printed. The original pictures are available in [10].

4.1.1 Optimization by Geometric Operations

Improvement methods that keep the mesh connectivity fixed while changing only the locations of the mesh vertices are commonly referred to as *geometric optimization*, *smoothing*, or *untangling* methods [126, 150]. The quality of a mesh can be improved by shifting the interior points within the domain boundary, and usually, boundary points have to be kept intact for compatibility with other meshes or for imposing the required boundary conditions for the finite element analysis.

Before the development of valid shape indicators, an early attempt to improve a triangular mesh was to shift each interior node in turn to the centroid of the polygon formed by all the nodes connected to it through an edge of the mesh. This iteration process is known as *Laplacian smoothing* [79], and such heuristic smoothing techniques have been successfully employed for tri and quad-meshes [66, 177]. Laplacian smoothing is computationally inexpensive and well-suited for meshes combining arbitrary element types. However, due to its quality-unaware node averaging scheme, Laplacian smoothing can lead to mesh quality deterioration and the generation of inverted elements.

Besides smoothing operations, vertex repositioning is able to address other needs such as equidistribution of volumes, shape improvement, or adaptive refinement [71]. Based on a shape quality indicator, we can define a cost function over the mesh, which can be maximized as a local/global optimization problem to improve the overall mesh quality [70, 142, 94]. In these works, the objective function is defined by measuring one or more mesh properties, such as in [86], and it can be optimized by repositioning the vertices, leading to an improve-

ment of the measured properties. The work in [71] analyzes the approaches for

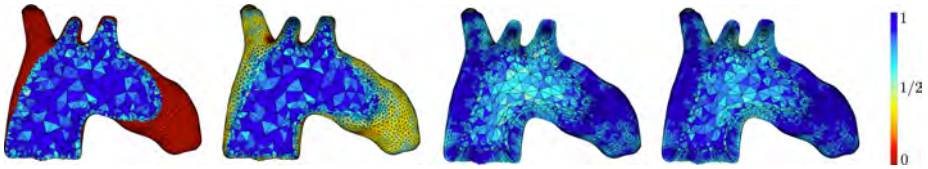


Figure 4.3: Initial and smoothed aorta meshes with elements colored according to their mean ratio [173]. From left to right: initial mesh with prismatic boundary layer; Laplace smoothing; global optimization; GETMe optimization.

the construction of a global objective function: an *all-vertex* method where the positions of all free vertices are moved simultaneously in a single iteration; and a *single-vertex* method where the position of only one vertex is modified at a time. The study, limited to tetrahedral meshes, sets the harmonic average of the mean ratio as the measure of the quality of the mesh and concludes that, of the two methods considered, the single-vertex method is better for making fast improvements, while the all-vertex one is preferable when an accurate solution of the optimization problem is necessary.

A comparison of six morphing methods that preserve the topology throughout the optimization process has been proposed in [163] and tested over four tet and four hex example problems. The quality of the elements is assessed using the scaled Jacobian indicator. Notably, the authors notice that hexahedral meshes tend to maintain higher element quality for longer than tetrahedral meshes during morphing. This fact is more important in hexahedral meshes because local remeshing of hexahedra is generally considered intractable while being straightforward for tetrahedral meshes. Since a global optimization approach might lead to a high computational effort, a combined approach of Laplacian smoothing and local optimization was proposed in [69], where optimization is only accomplished in problematic regions of the mesh. In addition, theoretical developments on the local optimization problem based on shape indicators for triangular and tetrahedral elements were presented by Aiffa and Flaherty [3].

Laplace smoothing and global optimization have been the dominating techniques in mesh optimization based on node shifting in the early days of mesh generation. A breakthrough was made when Vartziotis et al. [170] proposed the Geometric Element Transformation Method (GETMe), which is a purely geometric process to move the nodes of an element to improve its quality. The driving force behind GETMe smoothing is regularizing element transformations which, if applied iteratively, lead to more regular elements (i.e., with a higher mean ratio), see Figure 4.3. Such transformations for polygons with an arbitrary number of nodes have been proposed and analyzed in [172], and an extension to hexahedra and general polyhedral elements is detailed in [173].

4.1.2 Optimization by Topological Operations

Local topological operations such as edge/face swaps, and elimination of nodes, edges/faces, and elements, are effective means to improve the quality of a mesh. These operations can be carried out based on an appropriate shape indicator, and/or according to some topological considerations, to create a structure as balanced as possible in the connections among nodes, edges, faces, and elements. Numerous contributions address mesh-related questions as mesh optimization problems and deal with edge collapse and edge swap operations. For instance, the seminal work [80] minimizes an energy function that explicitly models the competing desires of conciseness of representation and fidelity to the data. Starting from this method, numerous works have addressed the simplification of a model as an optimization problem, e.g., [138, 65, 105]. The related literature is wide, and we only report here some examples to give a hint on how shape indicators can be combined for mesh optimization (or also *re-meshing*). We address the interested reader to [5, 13] for a more detailed treatment of the subject.

General optimization algorithms, for instance, automatically eliminate valence-three nodes (nodes with three incident edges) from a triangle mesh (Figure 4.4(a)). This is commonly considered a safe operation, as it reduces the number of elements and edges and the quality of the newly created triangle is typically higher than those of the old ones. Similarly, short edges smaller than a certain threshold can be eliminated by shrinking the two related triangles to line segments, and small triangles are eliminated by shrinking them to a point (Figure 4.5(b,c)). Analogous automatic operations can be defined for quad-, tet-, and hex-meshes [119], but only if there is little or no ambiguity on how to adjust the connectivity after an element is removed or modified.

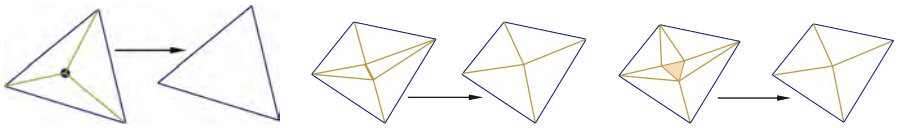


Figure 4.4: Triangle mesh optimization: (a) elimination of a valence-three node, (b) short edge shrinking, (c) small triangle shrinking.

The situation changes if we want to remove a valence-four node from a triangle mesh, as in Figure 4.5(a). In this case, we obtain a quadrilateral element that can be triangulated in two different ways (along the two diagonals), and we need a criterion to establish which one to choose. Quality indicators are employed in topological optimizations every time in which multiple configurations are acceptable, to indicate the best one. Other examples are the diagonal swap between couples of adjacent triangles or quads, and the elimination of a valence-three node in a quad-mesh, see Figure 4.5(b,c).

The preferred configurations for local transformations are those whose mean geometric qualities are maximized. Although the resulting mesh would certainly

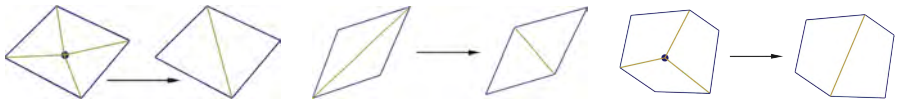


Figure 4.5: Triangle mesh optimization: (a) elimination of a valence-four node, (b) diagonal swap. Quad mesh optimization: (c) elimination of a valence-three node.

be different, the optimization by local transformations can be done with respect to any valid quality indicator. In three dimensions, there exist several methods for the optimization of tetrahedral meshes also using topological operations, e.g., [87, 53, 115], while much less is available for generic polyhedra. When possible, generic polyhedra are reduced to a set of tetrahedra [171] and quality is assessed on the mean volume of the equivalent tetrahedrization or ad-hoc strategies are adopted, for instance, using grid-meshes [166].

Pure hexahedral meshes are very often generated by mapping and sweeping methods, which are structured by the nature of their generation, and hence, there is no need to rebuild their topology. When hybrid elements are present, the application of mesh smoothing or untangling algorithms becomes more challenging, and a common assumption is that all the polyhedral elements admit a tetrahedrization [130].

Optimizing the element connections in polytopal meshes, in general, is difficult enough to leave any room for topological optimization by a change in element types. Indeed, in such situations, other possible solutions would have already been evaluated right at the spot when the problem first arises. In other words, topological optimization of polyhedral meshes might be possible; however, it is unlikely to be cost-effective as local element swaps can only be applied to very limited locations if any, even after rigorous and time-consuming analysis [113].

4.1.3 Recent Trends

Numerical solutions on complex geometries need to capture intricate physical features which in turn compel engineers to seek a balance between accuracy and computational cost, despite the lively growth in computing power. Unstructured meshes promise to yield this balance, offering accurate results and being preferred over their structured counterpart in certain applications [179]. On the one hand, to progress in this direction, methods are being investigated to improve the estimation of the optimization objective function; for instance, by leveraging the eigenanalysis of a Jacobian-based operator computed over proper subsets of vertices [179]. On the other hand, the increasing interest in developing polytopal element methods for the numerical discretizations of partial differential equations requires novel and effective algorithms to handle polygonal and polyhedral grids and to assess their quality.

The interest in using general polytopal meshes has grown over the years because these models are well suited for handling complex domains, featuring geometrical complexities for which the generation of good quality simplicial or hexahedral meshes can be particularly expensive or even unfeasible. Among the open problems, there is the issue of handling polytopal mesh refinement [101, 16, 27], i.e., partitioning mesh elements into smaller elements to produce a finer grid, and agglomeration strategies [52, 104, 7], i.e., merging mesh elements to obtain coarser grids. In both cases, refinement and agglomeration, it is crucial to preserve the quality of the underlying mesh, because this might affect the overall performance of the method in terms of stability and accuracy. Adaptivity is a key point that guarantees reliability and efficiency. For simplicity, algorithms often privilege dealing with convex polygons or polyhedra. For instance, the refinement algorithms in [27], start from a coarse mesh made of convex polygons created from a number of discrete fractures and use aspect ratio to select elements to be cut with planes, oriented according to specific criteria.

Application-oriented problems such as simulation on fracture networks call for approaches to mesh optimization able to deal with the geometrical complexity and the size of the computational domain. Indeed, fracture networks described by the Discrete Fracture Network (DFN) model correspond to sets of intersecting planar polygons arbitrarily oriented in a three-dimensional space, representing fractures in underground rock formations, and are typically generated starting from hydraulic and geological soil models. These models might count a large number of fractures, with dimensions ranging from centimeters to kilometers, and form an intricate network of intersections. To address this problem it is necessary to use methods able to scale to different sizes, incorporate refinement strategies when necessary, and agglomerate elements far from fractures. The methods developed in [31, 29] cut the elements of a tessellation with the intersection segments of two fracture plans, and use an optimization approach to further split the elements that are too thin and elongated. Vertex collapses are in place too, e.g., if two vertices are close under a threshold, as well as small shifts of vertices if the mesh quality locally improves. In the terms of the previous sections, we would classify this strategy as a geometry-topology-driven optimization. Further agglomerations are possible, in particular, far from the fracture constraints. Recently, an agglomeration approach driven by the quality indicator (2.28) has been proposed in [161]. Extensions to whole three-dimensional domains are currently under study, with preliminary results in [30].

A key aspect that is receiving increasing attention is how to keep elements aligned to some extent with the geometry of the solution. In this direction, previous studies on simplicial meshes proved that a properly generated anisotropic mesh can lead to a much more accurate solution than an isotropic one with comparable size, see [82] and the references therein. In this view, the shape of the elements of the tessellation is not necessarily uniform, and mesh elements with a large aspect ratio are also admitted [82]. Based on the intuition that it is convenient to adapt the tessellation to the physical problem, recent studies

address the problem of how to deform a mesh to follow sharp physical interfaces, see for instance [122]. In this work, the topology of the tessellation does not change during the deformation process, rather it admits that elements decrease their quality measure until zero. These studies and the anisotropic mesh quality measures available in [82] focus on convex polygonal meshes.

Finally, we mention the ongoing investigation of the use of Convolutional Neural Networks (CNNs) for handling polygonal [8] and polyhedra [6] grid refinement. The idea behind the approach in [8] is to recognize an almost regular single polygon as the representative of a set of polygons and then apply a polygon refinement strategy to the recognized element. Currently, triangular, quadrangular, pentagonal, and hexagonal polygons are used as basic shapes for recognition. Similarly, when dealing with polyhedra, a k-means algorithm is used to learn a clustered representation of the elements of an initial tessellation, on the basis of some basic convex polyhedral classes (cube, prism, tetrahedron, and other). The type of the recognized element determines the partition rule [6], thus improving and replacing the elements of the original mesh. Being these approaches supervised, the quality of the polytopes recognized by the neural network is demanded to the variety and representativeness of the polytopes used to train the model.

4.2 Mesh Optimization Driven by Quality Indicators

We introduce the mesh optimization strategy for generic polytopal meshes defined in [161, 154]. It falls under the category of topological optimization methods, as it does not modify the position of mesh vertices, and it is driven by a quality indicator that "indicates" the topological operations to be performed. It involves two distinct steps:

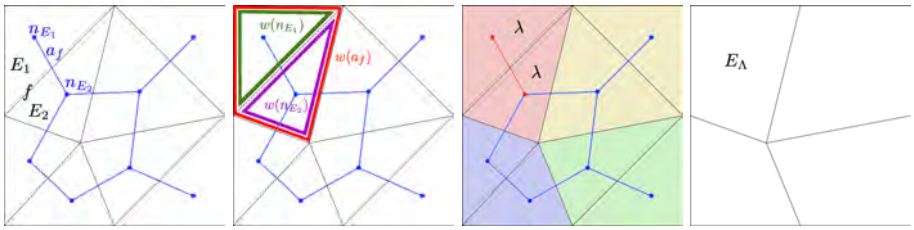
1. evaluation of the quality of the mesh elements through a local quality indicator, opportunely grouping them with tags;

2. agglomeration of elements that bear identical tags, generating a new mesh.

The algorithm was originally formulated using the VEM quality indicator of Section 3.4, but we can choose any of the polytopal indicators defined in Section 2.2.3. Indeed, even if the starting mesh is triangular, we will need to measure the quality of groups of agglomerated elements, which are therefore generic polygons. We detail the process in parallel for two-dimensional ($d = 2$) and three-dimensional ($d = 3$) meshes. The entire workflow is shown in Figure 4.6.

4.2.1 Mesh Partitioning

METIS [85] is a collection of serial programs implementing graph partitioning algorithms and finite element meshes, and producing fill-reducing orderings for



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

Figure 4.6: Optimization pipeline: (a) input mesh (black), with elements E_1, E_2 and their shared edge f, and mesh dual graph (blue), with nodes n_{E_1}, n_{E_2} and arc a_f; (b) weights assignment: $w(n_{E_1})$ is the quality of the green polygon, $w(n_{E_2})$ is the quality of the purple polygon and $w(a_f)$ is the quality of the red polygon; (c) input mesh colored with respect to METIS labeling: nodes n_{E_1} and n_{E_2} are assigned the same label λ; (d) agglomeration of the elements with the same label: elements E_1 and E_2 become a single element E_Λ.

sparse matrices. In the context of our work, we use it to perform a K-way partitioning of the dual graph of a mesh. We construct this graph by representing each mesh element as a node and establishing arcs between adjacent elements, as depicted in Figure 4.6a. Additionally, we assign weights to both nodes and arcs based on the chosen element quality indicator $\mathcal{Q}$, cf. Figure 4.6b. In detail:

- Each mesh element $E \in \Omega_h$ becomes a graph node n_E with the associated weight $w(n_E) = \mathcal{Q}(E)$;
- Each internal mesh $(d-1)$-dimensional object (which correspond to a face $f \in \mathcal{F}_h$ if $d = 3$ or an edge $e \in \mathcal{E}_h$ if $d = 2$), shared by elements E_1 and E_2, becomes a graph arc a_f with weight $w(a_f) = \mathcal{Q}(E_1 \cup E_2)$;
- $(d-1)$-dimensional boundary objects (boundary faces if $d = 3$ or boundary edges if $d = 2$) are not part of the graph, as they are only shared by one element.

We store the weights in two arrays $\boldsymbol{w_n}$, $\boldsymbol{w_a}$ that will be passed to METIS together with the mesh. To maximize their impact on the mesh partition process, and also because METIS only accepts integer weights, we re-scale the weights in $\boldsymbol{w_n}$ from $[0, 1]$ onto the interval

$$\left[1 + \left\lfloor \min(\boldsymbol{w_n}) \frac{\#\boldsymbol{w_n}}{c_w} \right\rfloor,\ 1 + \left\lfloor \max(\boldsymbol{w_n}) \frac{\#\boldsymbol{w_n}}{c_w} \right\rfloor\right]$$

and similarly for $\boldsymbol{w_a}$. We add a constant 1 because zero-weights are not accepted, and divide by $c_w \in \mathbb{R}$ to avoid extremely big weights in very large meshes which may cause overflow problems. For the meshes in this work, we used $c_w = 10$.

The routine *METIS-PartGraphKway* partitions a graph into K parts using multilevel K-way partitioning. We compute the number of required partitions

K as a percentage of the number of elements in the mesh $\#\Omega_h$. This percentage is called the *optimization parameter* $\mathcal{K}$ and it is defined by:

$$K = \mathcal{K}\%(\#\Omega_h) := \mathcal{K}\,\frac{\#\Omega_h}{100}.$$

If we set $\mathcal{K} = 20$ we are asking the optimization to partition the mesh into $K = 20\%(\#\Omega_h)$ parts, i.e., to generate a mesh with $1/5$ of the elements of Ω_h. In our experiments, we could observe that values of $\mathcal{K}$ below 5 lead to collapsing all the elements into a single one, while for $\mathcal{K}$ values higher than 40 the number of partitions computed by METIS does not increase proportionally. These observations could highlight possible limitations of the partitioning algorithm that are independent of the quality indicator used. The *METIS-PartGraphKway* routine can be configured with a number of flags. In particular, we set *PartitionType: CutBalancing* and activated the flags *ContigousPartitions*, *CompressGraph*, and *MinimizeConnectivity* (see the METIS documentation for the details [85]).

4.2.2 Mesh Agglomeration

The partition computed by METIS consists of an array containing a flag $\lambda_E \in [1, K]$ for each element E of the mesh, see Figure 4.6c. We aim to pick all the elements with the same flag and replace them with their union, which will be treated as a single element. In particular, for each flag $\lambda = 1, \dots, K$ we consider the set $\Lambda := \{E \in \Omega_h : \lambda_E = \lambda\}$ and build a new element E_Λ defined by all the faces (edges if $d = 2$) that are shared by only one element in Λ. While doing this operation, we check that the arising element is still a manifold. If not, we stop the agglomeration and skip to the next label, maintaining the old elements. This situation is extremely rare, but a single non-manifold element may compromise the whole mesh for certain applications.

At the end of the optimization process with input parameter $\mathcal{K}$, the resulting mesh will have approximately K elements optimized for the computation of the virtual element basis functions. We will have removed around $\#\Omega_h - K$ elements from the mesh and at least $\#\Omega_h - K$ faces (edges if $d = 2$) between them, see Figure 4.6d. Given that a reasonable range for $\mathcal{K}$ is $[5, 40]$, as noted above, we can generate new meshes containing from 5% to 45% of the elements w.r.t. the original mesh. Moreover, if K is particularly low, we may happen to remove also edges (if $d = 3$) and vertices, but in a smaller number. We point out that the primary goal of the algorithm is to reduce the number of elements to the expected percentage. In this sense, the global mesh quality is likely to decrease after the optimization. However, the loss of quality is optimal with respect to the number of elements removed. In other words, the optimized mesh has the maximum quality we can achieve by removing that number of elements.

If we have a discrete numerical problem defined on the mesh, the optimization of elements and faces naturally translates into a reduction of the number of degrees of freedom. For instance, recalling the DOFs involved in the VEM formulation of order k (see Section 3.2), we have a reduction of:

- 1 DOF for each removed vertex;
- if $k > 1$, $k - 1$ DOFs for each removed edge;
- if $k > 1$, $k - 2$ DOFs for each removed face (if $d = 3$);
- if $k > 1$, $k - 2$ DOFs for each removed element.

This reduction has a significant impact on computational and storage costs, in particular for high-order formulations. In [161, 154] we show how, when combined with the VEM quality indicator, the optimization does not deteriorate the VEM convergence rate. Optimized meshes produce approximation errors comparable or improved to those produced by the original mesh. In fact, we were even able to recover the optimal convergence rate in low-quality meshes by removing pathological elements, i.e., elements that yield a system matrix with a very large condition number, which may cause the VEM to diverge on the original tessellation.

4.2.3 Optimization in Action

To give a practical example of how the optimization works, we test it over a mesh using different quality indicators. We consider a triangular discretization of the unit square, see Figure 4.7(a), and optimize it with input parameter $\mathcal{K} = 40$ using six different element indicators taken from Section 2.2:

$$\mathcal{Q}_1(E) = \frac{2r}{h_E} \qquad \text{Inradius Ratio}$$

$$\mathcal{Q}_2(E) = \frac{2\pi A}{p^2} \qquad \text{Area/Perimeter Ratio}$$

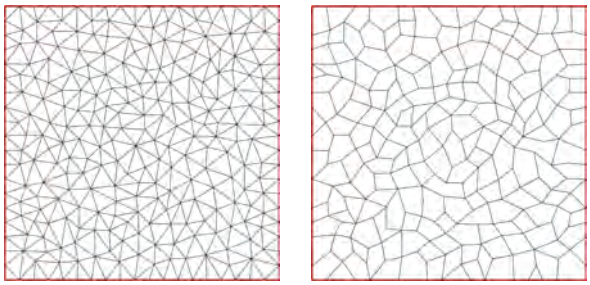
$$\mathcal{Q}_3(E) = \frac{\theta_{\min}}{\theta_{\max}} \qquad \text{Angle Ratio}$$

$$\mathcal{Q}_4(E) = \sqrt{\frac{\varrho_1\varrho_2 + \varrho_1\varrho_3 + \varrho_1\varrho_4}{3}} \qquad \text{VEM Indicator}$$

$$\mathcal{Q}_5(E) = \min_i \frac{\det(\mathcal{A}_i)}{\|e_i\| \ \|e_{i+1}\|} \qquad \text{Scaled Jacobian}$$

$$\mathcal{Q}_6(E) = \min_i \frac{2}{|\mathcal{A}_i|_F \ |\mathcal{A}_i^{-1}|_F} \qquad \text{Frobenius Norm}$$

where $\mathcal{A}_i$ is the Jacobian matrix at the i-th vertex v_i of the polygon E, and e_i, e_{i+1} are the two edges incident to v_i. After the optimization we obtain six different meshes, but all with the same number of elements (the 40% of the original mesh), and similar numbers of edges and vertices. We note as $\Omega_h + \mathcal{Q}_i$ the mesh generated by optimizing mesh Ω_h with indicator $\mathcal{Q}_i$, and in Figure 4.7(b) we present $\Omega_h + \mathcal{Q}_1$.



(a) (b)

Figure 4.7: (a) Original mesh: 284 vertices, 773 edges, and 490 triangles; (b) mesh optimized with $\mathcal{K} = 40$ and Inradius Ratio $\mathcal{Q}_1$: 264 vertices, 459 edges, and 196 polygons.

To analyze the results, we compute the global quality of the optimized meshes with respect to the mesh quality indicators:

$$\bar{\mathcal{Q}}_i(\Omega_h) := \frac{\sum_{E\in\Omega_h} \mathcal{Q}_i(E)}{\#\Omega_h}, \quad i = 1, \ldots, 6,$$

which are nothing but the element indicators $\mathcal{Q}_i$ averaged across the whole mesh. What we expect is that a mesh optimized with local indicator $\mathcal{Q}_i$ will have a high quality score when measured with the respective global indicator $\bar{\mathcal{Q}}_i$. In Table 4.1 we present the quality scores of the original mesh and the six optimized ones. Each column represents a mesh, and each line represents the quality score with respect to a certain global indicator. The bold entries highlight the quality of the mesh optimized with the same indicator, i.e., the one in which $\bar{\mathcal{Q}}_i$ should be particularly high. Note that the table should be read by lines and not by columns, as the indicators may have different scales (besides being all valued in $[0, 1]$).

Table 4.1: Comparison of the optimization through different quality indicators: mesh Ω_h is optimized with the local indicators $\mathcal{Q}_i$ generating mesh $\Omega_h + \mathcal{Q}_i$, then its quality is measured with global indicators $\bar{\mathcal{Q}}_j$, for $i, j = 1, \ldots, 6$.

	Ω_h	$\Omega_h + \mathcal{Q}_1$	$\Omega_h + \mathcal{Q}_2$	$\Omega_h + \mathcal{Q}_3$	$\Omega_h + \mathcal{Q}_4$	$\Omega_h + \mathcal{Q}_5$	$\Omega_h + \mathcal{Q}_6$
$\bar{\mathcal{Q}}_1$	0.45	**0.54**	0.55	0.55	0.52	0.54	0.52
$\bar{\mathcal{Q}}_2$	0.28	0.36	**0.36**	0.36	0.35	0.35	0.35
$\bar{\mathcal{Q}}_3$	0.56	0.51	0.53	**0.56**	0.47	0.51	0.47
$\bar{\mathcal{Q}}_4$	0.92	0.84	0.85	0.85	**0.83**	0.84	0.84
$\bar{\mathcal{Q}}_5$	0.68	0.65	0.69	0.67	0.60	**0.65**	0.61
$\bar{\mathcal{Q}}_6$	0.67	0.65	0.68	0.67	0.63	0.65	**0.64**

The main consideration arising from Table 4.1 is that the selected indicators produce very similar results. Indeed, besides small oscillations, the quality is

approximately constant in each line, meaning that the optimized meshes are similar to each other. This can be partly due to how the optimization algorithm works because it only compares elements in couples along their common edge. At the same time, this confirms what is already noted in Section 2.2.4, that is, many of the most famous indicators are equivalent. The only indicator exhibiting a significantly higher quality when optimized with respect to itself is the Angle Ratio $\mathcal{Q}_3$. The qualities measured by the global Scaled Jacobian $\bar{\mathcal{Q}}_5$ and Frobenius Norm $\bar{\mathcal{Q}}_6$ are very close for all meshes, suggesting that these indicators are particularly similar. As already noted, we do not expect the optimization algorithm to improve the quality with respect to the original mesh. This only happens for the Inradius Ratio and Area/Perimeter Ratio, in which the quality of mesh $\Omega_h + \mathcal{Q}_i$ measured with $\bar{\mathcal{Q}}_i$ is greater than that of Ω_h. In conclusion, it is worth noting that not all indicators correlate equally with the accuracy of a PDE simulation or any other computation we may be interested in. Therefore a quality difference between two meshes of 1% for one indicator may impact more (or less) than the same difference for another indicator.

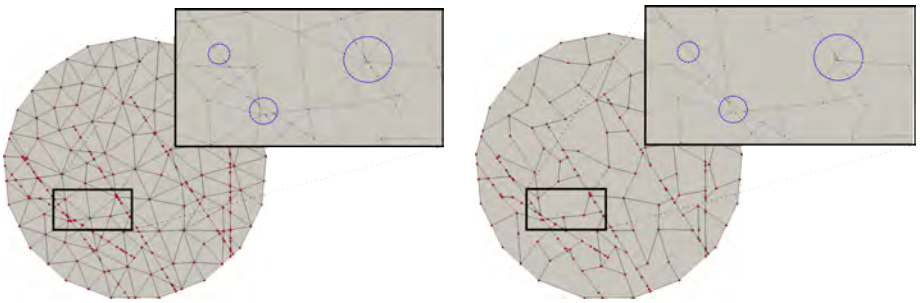
4.2.4 Applications

We present here two contexts in which we applied our optimization algorithm, combined with the VEM quality indicator. They serve as a taste of the variety of situations in which reducing the size of a mesh while taking care of its quality may provide significant advantages to the user.

Constrained Problems The first scenario is the discretization of a domain containing small-size boundary details and constrained edges and vertices that the mesh should conform to. Under such constraints, standard triangular meshing algorithms are forced to generate tons of small and badly shaped elements, therefore it may be useful to agglomerate some of them into larger polygons.

A famous example is the flow simulation in underground fractured porous media, a fundamental task for a huge number of applications such as carbon capture processes, the monitoring of aquifers, or the contaminant transport in the subsoil. Among the existing approaches [74], the Discrete Fracture Network (DFN) model is particularly suited for scenarios in which the permeability of the surrounding media is negligible with respect to the fracture transmissivities. Indeed, in DFN simulations the porous rock matrix is neglected and the fluid is confined to move on the rock discontinuities, called *fractures*, which act as channels for the flow. The DFN model is characterized by a dimensional reduction of each fracture to a planar polygon immersed in a three-dimensional space, and the coupling conditions are imposed on fracture segment intersections [118]. Due to the deficiency of direct measurements of the subsoil properties, the position, orientation, size, and aperture of each polygonal fracture are generated according to different probability distributions. Furthermore, real fractured media are characterized by a huge computational domain, that involves a large number of unknowns, and often present an intricate system of intersections at different scales, e.g., large faults are crossed by small fractures.

The generation of a polygonal tessellation of a DFN conforming to the fracture interfaces is not a trivial task, due to the large number of fractures typically found in real applications, the different sizes of neighboring domains, and the high complexity of their intersections. In particular, it frequently leads to a high number of degrees of freedom in the discrete problem, and to the generation of numerous elongated elements with aligned small edges. Such "bad-shaped" cells impact the accuracy of the simulation, producing high condition numbers and numerical errors, while the huge number of DOFs translates into a significant computational cost and, again, an increase of the condition numbers.



(a) (b)

Figure 4.8: Focus on a single fracture of a DFN. Small edges and cells in the original fracture (a) are removed after the optimization process (b). Blue circles compare three examples of mesh quality improvements.

For these reasons, in [161] we proposed the application of the mesh optimization algorithm locally on each fracture of the network to reduce the total unknowns of the discrete problem with a control on the geometric quality of the elements (see Figure 4.8). The optimization was driven by the VEM quality indicator presented in Section 3.4, and the labeling part was performed using GraphCut [39] instead of METIS. To evaluate the effects of the optimization, we ran the algorithm over a DFN and solved a numerical problem over the original and the optimized version. Then, we compared the results produced by the VEM over the different networks in terms of computational cost, approximation error, and convergence rate. We remark that the optimization was performed independently on each fracture of the network, thus its application in larger and more complex DFN configurations is perfectly feasible.

The quality optimization strategy has proved to be particularly effective in the DFN setting, being able to reduce the total number of mesh elements (up to 65%) and the number of DOFs (up to 50%), while preserving the VEM optimal convergence rates. The former effect leads to a remarkable reduction in the computational effort for the discrete system assembly process; the latter effect translates into a significant gain in terms of computational cost to obtain the numerical solution. Finally, a slight improvement in the quality of the discretization was observed, both locally on the VEM projector approximation

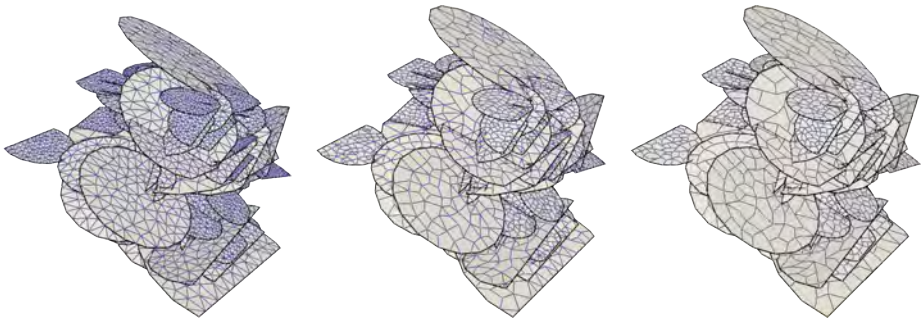


Figure 4.9: A complete DFN (left) optimized with a lower parameter value (middle) and with a higher parameter value (right).

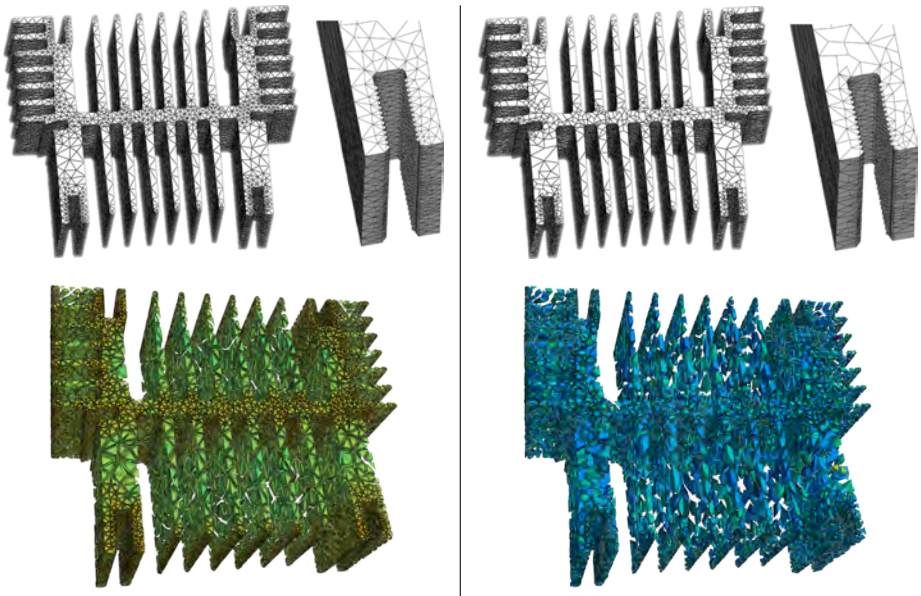
errors measured on each mesh element and globally on the condition number of the stiffness matrix.

Time-Dependent Problems The second scenario is the discretization of a domain on which we need to solve a time-dependent problem, i.e., to solve the same PDE multiple times with some time-dependent parameters changing at every iteration. In this case, the main goal of optimization is to reduce the size of the mesh as much as possible, while preserving a quality sufficient to the correct working of the numerical method. Each element, face, edge, or vertex, removed in the optimization process represents a gain in terms of computational time and memory storage that will be multiplied by the number of time steps of the problem.

In [154] we tested our algorithm over a tetrahedral CAD model of a *heat sink*, downloaded from the repository *Traceparts*[1]. The considered model is particularly complex because it presents three different levels of detail, which determine the generation of elements with variable sizes. The optimization was driven by the VEM quality indicator ϱ described in Section 3.4. The resulting mesh, noted *Cad*, is visible in Figure 4.10, together with its optimization with parameter $\mathcal{K} = 20$. In the "exploded" version, elements are colored with respect to their quality, from yellow ($\varrho = 1$) to blue ($\varrho = 0$). While *Cad* contains only tets, which are very high quality, in Cad_{20} the 80% of the elements have been agglomerated into generic polyhedra. The local quality of Cad_{20} is therefore generally lower, and this is an unavoidable side-effect when optimizing high-quality meshes. However, our guess is that the loss in local quality is negligible when compared to the gain in terms of DOFs.

We solved a time-dependent diffusion problem defined on *Cad* and Cad_{20} using a backward Euler scheme for a variable number of time steps. In Table 4.2 we report some statistics concerning the differences between *Cad* and Cad_{20} for $k = 1, \dots, 4$. In particular, we measure the difference in terms of number

[1] www.traceparts.com/it/product/fischer-elektronik-gmbh-co-kg-strangkuhlkorper-fur-einrasttransistorhaltefeder?CatalogPath=TRACEPARTS%3ATP014002&Product=34-23062017-133913&PartNumber=SK%20593%2025&corid=0737fe6b-01e1-4714-6dec-a98692f099d7



(a) *Cad*, 122882 elements (b) Cad_{20}, 24715 elements

Figure 4.10: *Cad* model optimized with $\mathcal{K} = 20$, with a close-up on a small-scale detail (top) and elements colored w.r.t. their quality (bottom).

Table 4.2: Differences between *Cad* and Cad_{20} for $k = 1, 2, 3, 4$.

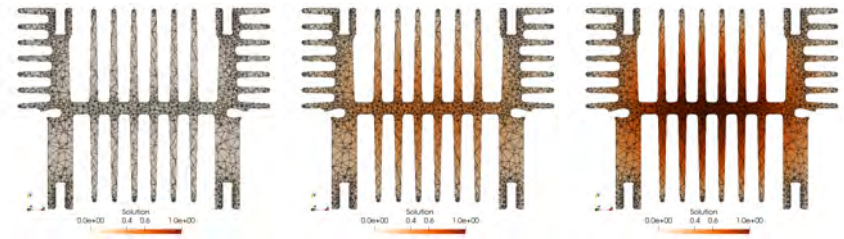
k	$\Delta\mathcal{T}_h$	ΔDOFs	$\Delta\mathcal{E}_{L^2}$	$\Delta\mathcal{E}_{H^1}$
1	$8.51 \cdot 10^4$	$4.00 \cdot 10^0$	$-3.33 \cdot 10^{-3}$	$+1.25 \cdot 10^{-2}$
2	$8.51 \cdot 10^4$	$1.91 \cdot 10^5$	$-1.27 \cdot 10^{-3}$	$-1.19 \cdot 10^{-2}$
3	$8.51 \cdot 10^4$	$6.47 \cdot 10^5$	$-7.10 \cdot 10^{-5}$	$-9.75 \cdot 10^{-4}$
4	$8.51 \cdot 10^4$	$1.45 \cdot 10^6$	$-1.15 \cdot 10^{-6}$	$-1.89 \cdot 10^{-5}$

of elements $\mathcal{T}_h$, which is constant for all k values, and of number of DOFs, which instead changes significantly. Both deltas are positive, meaning that the optimized mesh contains fewer elements and DOFs than the original one. Moreover, we measure the differences in the L^2 and H^1 errors, ε_{L^2} and ε_{H^1}. These deltas are mainly negative, i.e., the accuracy of the optimized mesh is lower, but they decrease exponentially with k, and therefore their influence is smaller and smaller.

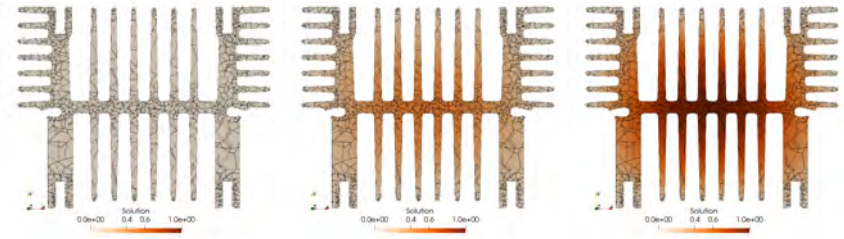
Last, we measure the difference in computational time required to perform the whole process, including the optimization step in the case of Cad_{20}. On one side, the reduction of the number of elements and DOFs leads to a smaller number of operations to compute. On the other side, some quantities may be more difficult to compute on agglomerated elements than on the original tetrahedra (e.g., the element volume). The final balance is positive for the

optimized mesh, but not enough to cover the cost of the optimization step. This is where the time-dependence comes into play because the optimization is performed only once while the gain in computational time is multiplied by the number of time steps. Therefore we have a critical number $\bar{t}$ such that for less than $\bar{t}$ steps, computations are faster on the original *Cad*, while for more than $\bar{t}$ steps the optimized Cad_{20} becomes advantageous. This number decreases as k grows: we have $\bar{t} = 2.63 \cdot 10^4$ for $k = 2$, $\bar{t} = 1.25 \cdot 10^3$ for $k = 3$, and $\bar{t} = 7.63 \cdot 10^2$ for $k = 4$. For $k = 1$ times are essentially equivalent, thus we could not identify a single $\bar{t}$ value. However, the presented calculations were made using a code written and optimized for meshes with convex elements, therefore we believe further improvements can be made with solvers optimized for generic non-convex cells.

In conclusion, for $k > 1$ Cad_{20} contains drastically fewer DOFs than *Cad* and produces results in a shorter time and with very similar accuracy, thus making the optimization convenient. In Figure 4.11 we present a visual result of the simulation at different times.



(a) *Cad*



(b) Cad_{20}

Figure 4.11: Simulation with $k = 1$ over models *Cad* (top) and Cad_{20} (bottom). From left to right, $t = 0$, $t = 0.5$, $t = 1$.

Chapter 5

PEMesh Software

This section is devoted to the presentation of PEMesh, the software used during the course for live exercises and also in these notes for generating practical examples. The current implementation includes major updates with respect to the original version [48]. PEMesh is an open-source software tool designed to help researchers perform experiments on the analysis and design of polytopal meshes for Polytopal Element Methods (PEM). It is an advanced graphical tool that seamlessly integrates geometric design pipelines and PEM simulations.

PEMesh is composed of four modules for generating or loading a mesh (Section 5.1), analyzing its quality (Section 5.2), solving a PDE with a PEM over it (Section 5.3), and possibly optimizing the input mesh according to a user-selected quality indicator (Section 5.4). Specifically, PEMesh supports the design, generation, and optimization of complex input polygonal meshes by stressing geometric properties while providing the possibility to run a simulation with a PEM on the generated meshes. Furthermore, it allows the user to correlate one or more quality indicators on the input polytopal mesh with the performance of the PEM solver, and to visualize the results through customizable and interactive plots. Meshes can then be optimized with respect to one of the available quality indicators, generating new meshes with fewer elements but similar quality. The last two steps, simulation and optimization, are interchangeable and repeatable, i.e., it is possible to optimize a mesh multiple times and compare the performance of the simulation after each step.

PEMesh is released as an open-source project (`https://github.com/DanielaCabiddu/PEMesh`). The source code is written in C++ and depends on some external libraries. Specifically, It exploits Qt libraries (version 6.7.0, including Qt Charts module) for the design and implementation of the graphical user interface, Cinolib [109] to generate, analyze and visualize meshes, and Metis [85] to optimize them. Additional dependencies are necessary, such as Triangle [151] to generate 2D meshes and Boost [36] for some specific geometric utilities. A CMake configuration is available in the Github repository to make the building process automatic and feasible on multiple platforms, including the download and installation of the above-mentioned dependencies. Pre-compiled binaries are

available for the most recent operating system (Windows 11, Ubuntu 24.10, and MacOS 13).

5.1 Mesh Generation

The Mesh Generation module is started as soon as the application is run; its aspect is presented in Figure 5.1. It provides the possibility to load an existing polygonal mesh (encoded either as an *.off*, *.obj*, or *.stl* file) or generate a new one from scratch.

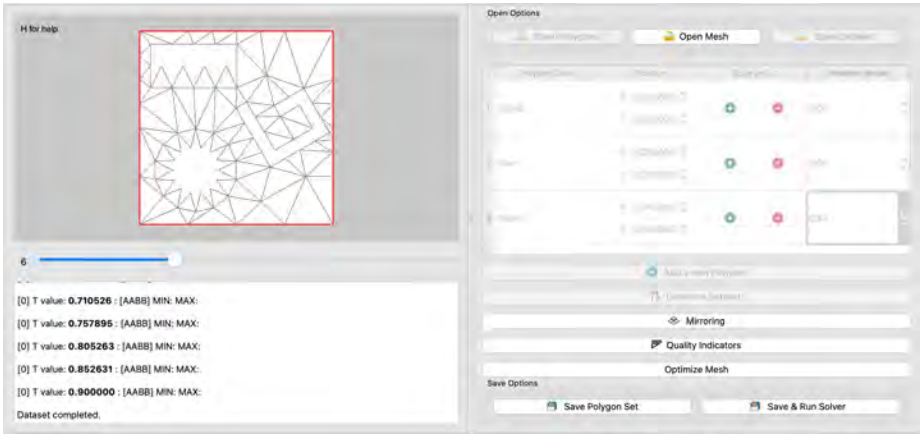
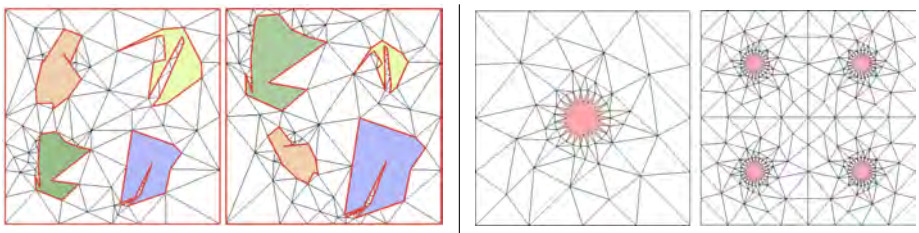


Figure 5.1: Main window. On the left, a canvas allows visualizing the input mesh(es) and the bottom text box reports some logs. On the right, an advanced panel allows setting multiple parameters to both guide the dataset generation according to desired geometric features and start the geometric analysis. Also, a button is provided to start the PEM solver and run the simulation quickly.

The mesh generation method takes a cue from the approach described in [11], where the domain is supposed to be a squared canvas, and the area of the domain which is not covered by a polygon is filled with triangles using [151]. Differently from [11], our framework supports the generation of meshes with more than one non-triangle polygon, whose position, scale, and rotation are chosen by the user before applying the triangulation of the external domain (Figure 5.2(a)). Furthermore, triangulation parameters are set according to the user needs (e.g., bounding the area of the triangles or the minimum angle). An additional feature is the possibility to generate a sequence of progressively finer grids, obtained by mirroring the squared domains multiple times, see Figure 5.2(b).



(a) (b)

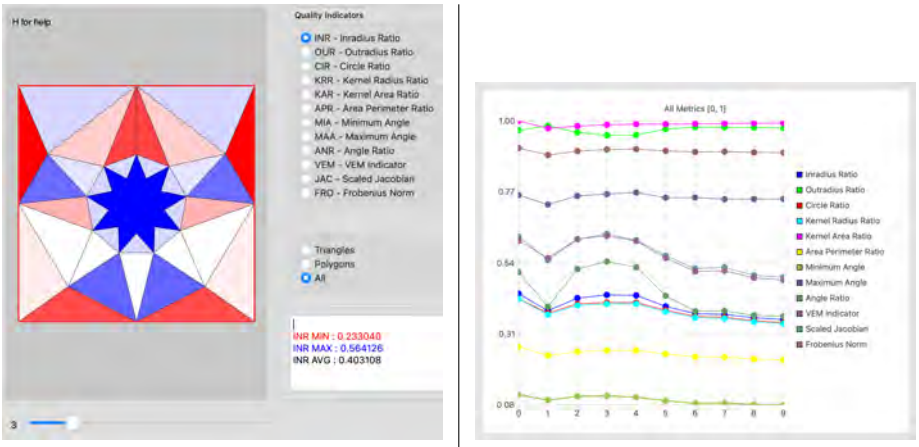
Figure 5.2: (a) Meshes generated by selecting the same set of polygons, but editing each element differently. (b) A mesh hierarchy is obtained by mirroring the meshes, thus reducing the average edge length and enabling convergence studies.

5.2 Quality Evaluation

When one or more polygonal meshes are available, either generated from scratch or loaded from disk, PEMesh allows a deep geometric analysis and provides a visual summary of geometric properties. Specifically, it considers a selection of the element quality indicators defined in Section 2.2, all valued in $[0, 1]$ (where 0 corresponds to low quality and 1 to high quality) and scale-independent:

- Aspect Ratios [Eq. (2.25)]: Inradius Ratio (INR), Outradius Ratio (OUR), and Circle Ratio (CIR), defined as the ratios between, respectively, twice the inradius and the diameter, the diameter and twice the outradius, the inradius and the outradius;
- Angle Ratios [Eq. (2.26)]: Minimum Angle (MIA), Maximum Angle (MAA), Angle Ratio (ANR), defined as the minimum, the maximum angle in the polygon, and their ratio;
- Shape Ratios [Eq. (2.27)]: Kernel Radius Ratio (KRR), Kernel Area Ratio (KAR), Area Perimeter Ratio (APR), defined as the ratios between, respectively, the kernel inradius and the polygon outradius, the kernel area and the polygon area, the square root of the polygon area and the polygon perimeter;
- VEM Indicator (VEM) [Eq. (2.28)], defined as in Eq. (3.9);
- Mean Ratios [Eq. (2.29)]: Scaled Jacobian (JAC) and Frobenius Norm (FRO), defined as the minimum of the determinants, or the Frobenius norms, of the Jacobian matrices computed at the vertices of the polygon.

Different visualizations of the results of the geometric analysis are available. When we focus on the polygonal mesh as a standalone object, each available mesh is colored with respect to its quality, see Figure 5.3(a). The color of each polygon is chosen based on the value of a specific quality indicator selected



(a) (b)

Figure 5.3: Graphical visualization of geometric polygonal metrics. (a) Given a polygonal mesh and a selected geometric metric, the mesh elements are colored with respect to their quality score. Numerical values are reported in the bottom text box. (b) Quality indicators are compared across all meshes of the sequence.

by the user. If needed, the module also allows visually highlighting the element (either triangle or polygon) where the minimum and the maximum values of the selected quality indicator are located, together with textual information about them (e.g., the numerical values). This visualization enables the detection of possible geometric degeneracies.

The other visualization approach consider the full dataset as a single object and plot how the evaluated quality indicators evolve in the dataset, and how they correlate with each other. PEMesh generates both a global chart where all the indicators are compared, as shown in Figure 5.3(b), and single charts for each indicator showing the plot of the minimum, maximum and average value of the indicator on the elements, possibly distinguishing between triangular and polygonal elements.

5.3 PDE Solver

PEMesh provides the possibility to compute the numerical approximation of a PDE on the available meshes with a PEM and to analyze the solver performance. To this end, the PEM solver is not part of the tool but is rather considered an external resource that is invoked from the graphical interface. Without loss of generality, our framework assumes that the selected solver takes in input a polygonal mesh and returns both the solution and the groundtruth (if any) of a PDE, together with statistics like the approximation error in different norms and conditioning of the stiffness matrix.

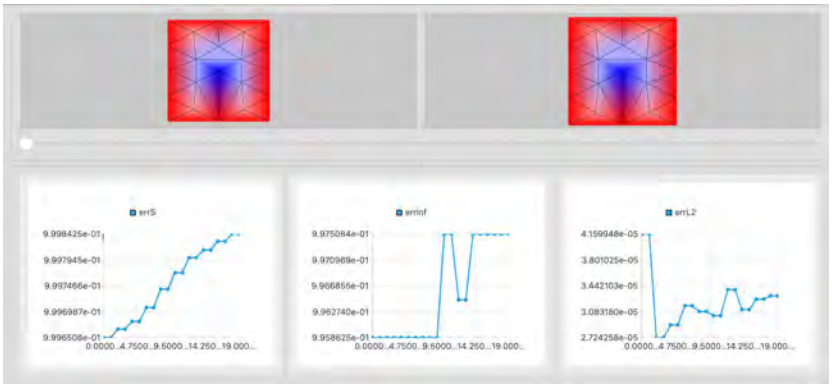


Figure 5.4: Visualization of PEM solver results. On the top, both the solver output and the groundtruth solution are color-mapped on the input polygon meshes, while on the bottom a set of linear plots show how the solver performance vary across the dataset.

5.4 Mesh Quality Optimization

The mesh optimization tool allows the optimization of the current mesh(es) through the optimization algorithm defined in Section 4.2. The user is required to select the quality indicator $\mathcal{Q}$ among the ones implemented in the software (listed in Section 5.2), the optimization parameter $\mathcal{K}$, and whether to assign weights to nodes and/or arcs of the dual graph (see Section 4.2.1 for the algorithmic details). There is also the possibility to decide whether the optimized meshes should replace the original ones, or if they should be appended to the mesh sequence.

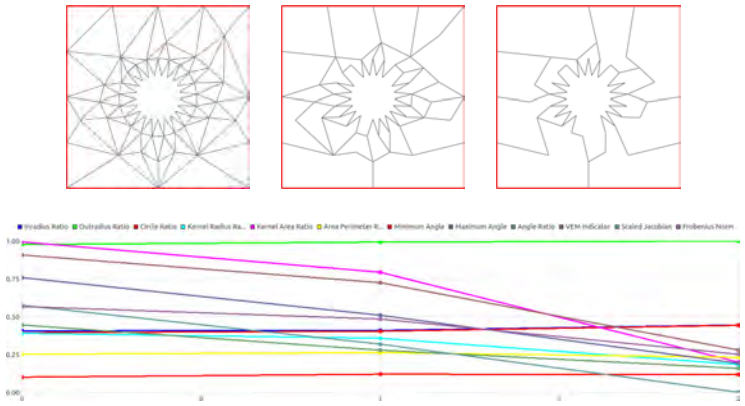


Figure 5.5: Optimization and quality analysis of a mesh. In the top row, a mesh optimized recursively with respect to the Angle Ratio, on the bottom the quality indicators computer over the three meshes.

After the optimization, PEMesh prints statistics about the number of removed elements, edges, and vertices, and we can visually inspect and compare the newly generated meshes, as shown in Figure 5.5. Moreover, we can use the quality evaluation tool to check how quality is changed after the optimization with respect to the different indicators. Finally, we can use the PDE solver and compare the performance of the VEM (or another numerical method) over the original and the optimized meshes. This last feature completes the pipeline and makes PEMesh a complete tool, capable of generating, analyzing, optimizing a mesh, and validating its quality.

Acknowledgment

Funded by the European Union - NextGenerationEU and by the Ministry of University and Research (MUR), National Recovery and Resilience Plan (NRRP), Mission 4, Component 2, Investment 1.5, project "RAISE - Robotics and AI for Socio-economic Empowerment" (ECS00000035). T. Sorgente, D. Cabiddu, S. Biasotti, and M. Spagnuolo are part of RAISE Innovation Ecosystem. F. Vicini is a member of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM) and thanks the partial support by INdAM - GNCS Project CUP_E53C23001670001.

Bibliography

[1] Aghdaii, N., Younesy, H., and Zhang, H. 5–6–7 meshes: Remeshing and analysis. *Computers & Graphics 36*, 8 (2012), 1072–1083.

[2] Ahmad, B., Alsaedi, A., Brezzi, F., Marini, L. D., and Russo, A. Equivalent projectors for virtual element methods. *Computers & Mathematics with Applications 66* (September 2013), 376–391.

[3] Aiffa, M., and Flaherty, J. A geometrical approach to mesh smoothing. *Computer methods in applied mechanics and engineering 192*, 39-40 (2003), 4497–4514.

[4] Alliez, P., Cohen-Steiner, D., Yvinec, M., and Desbrun, M. Variational tetrahedral meshing. In *ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), SIGGRAPH '05, Association for Computing Machinery, pp. 617–625.

[5] Alliez, P., Ucelli, G., Gotsman, C., and Attene, M. Recent advances in remeshing of surfaces. *Shape analysis and structuring* (2008), 53–82.

[6] Antonietti, P., Dassi, F., and Manuzzi, E. Machine learning based refinement strategies for polyhedral grids with applications to virtual element and polyhedral discontinuous galerkin methods. *Journal of Computational Physics 469* (2022), 111531.

[7] Antonietti, P. F., Berrone, S., Busetto, M., and Verani, M. Agglomeration-based geometric multigrid schemes for the virtual element method. *SIAM Journal on Numerical Analysis 61*, 1 (2023), 223–249.

[8] Antonietti, P. F., and Manuzzi, E. Refinement of polygonal grids using convolutional neural networks with applications to polygonal discontinuous galerkin and virtual element methods. *Journal of Computational Physics 452* (2022), 110900.

[9] Apel, T., and Dobrowolski, M. Anisotropic interpolation with applications to the finite element method. *Computing 47* (1992), 277–293.

[10] Attene, M. As-exact-as-possible repair of unprintable stl files. *Rapid Prototyping Journal* (2018).

[11] Attene, M., Biasotti, S., Bertoluzza, S., Cabiddu, D., Livesu, M., Patané, G., Pennacchio, M., Prada, D., and Spagnuolo, M. Benchmarking the geometrical robustness of a virtual element Poisson solver. *Mathematics and Computers in Simulation 190* (2021), 1392–1414.

[12] Attene, M., Campen, M., and Kobbelt, L. Mesh Repair. `http://www.meshrepair.org`, 2013.

[13] Attene, M., Campen, M., and Kobbelt, L. Polygon mesh repairing: An application perspective. *ACM Computing Surveys (CSUR) 45*, 2 (2013), 1–33.

[14] Au, P., Dompierre, J., Labbé, P., Labb, P., Guibault, F., Guibault, F., and Camarero, R. Proposal of benchmarks for 3d unstructured tetrahedral mesh optimization. In *In Proceedings of the 7th International Meshing RoundTable'98* (1998), Citeseer.

[15] Babuška, I., and Aziz, A. K. On the angle condition in the finite element method. *SIAM Journal on numerical analysis 13*, 2 (1976), 214–226.

[16] Bakhvalov, P., and Surnachev, M. Method of averaged element splittings for diffusion terms discretization in vertex-centered framework. *Journal of Computational Physics 450* (2022), 110819.

[17] Bank, R. E., and Xu, J. An algorithm for coarsening unstructured meshes. *Numerische Mathematik 73*, 1 (1996), 1–36.

[18] Barrett, K. Jacobians for isoparametric finite elements. *Communications in numerical methods in engineering 12*, 11 (1996), 755–766.

[19] Beaufort, P. A., Reberol, M., Kalmykov, D., Liu, H., Ledoux, F., and Bommes, D. Hex me if you can. *Computer Graphics Forum 41*, 5 (2022).

[20] Beirão da Veiga, L., Brezzi, F., Cangiani, A., Manzini, G., Marini, L. D., and Russo, A. Basic principles of virtual element methods. *Mathematical Models & Methods in Applied Sciences 23* (2013), 119–214.

[21] Beirão da Veiga, L., Brezzi, F., Marini, L., and Russo, A. Virtual element method for general second-order elliptic problems on polygonal meshes. *Mathematical Models and Methods in Applied Sciences 26*, 4 (2016), 729–750.

[22] Beirão da Veiga, L., Brezzi, F., Marini, L., and Russo, A. The virtual element method. *Acta Numerica 32* (2023), 123–202.

[23] Beirão da Veiga, L., and Vacca, G. Sharper error estimates for virtual elements and a bubble-enriched version. *arXiv preprint arXiv:2005.12009* (2020).

[24] Beirão da Veiga, L., Lipnikov, K., and Manzini, G. *The Mimetic Finite Difference Method*, I ed., vol. 11 of *MS&A. Modeling, Simulations and Applications*. Springer, 2014.

[25] Beirão da Veiga, L., Lipnikov, K., and Manzini, G. *The mimetic finite difference method for elliptic problems*, vol. 11. Springer, 2014.

[26] Beirão da Veiga, L., Lovadina, C., and Russo, A. Stability analysis for the virtual element method. *Mathematical Models and Methods in Applied Sciences 27*, 13 (2017), 2557–2594.

[27] Berrone, S., Borio, A., and D'Auria, A. Refinement strategies for polygonal meshes applied to adaptive vem discretization. *Finite Elements in Analysis and Design 186* (2021), 103502.

[28] Berrone, S., Borio, A., and Marcon, F. A stabilization-free virtual element method based on divergence-free projections. *Computer Methods in Applied Mechanics and Engineering 424* (2024), 116885.

[29] Berrone, S., Grappein, D., Pieraccini, S., and Scialó, S. A three-field based optimization formulation for flow simulations in networks of fractures on nonconforming meshes. *SIAM Journal on Scientific Computing 43*, 2 (2021), B381–B404.

[30] Berrone, S., Grappein, D., and Scialò, S. 3d-1d coupling on non conforming meshes via a three-field optimization based domain decomposition. *Journal of Computational Physics 448* (2022), 110738.

[31] Berrone, S., Pieraccini, S., and Scialò, S. A pde-constrained optimization formulation for discrete fracture network flows. *SIAM Journal on Scientific Computing 35*, 2 (2013), B487–B510.

[32] Bertolazzi, E., and Manzini, G. A cell-centered second-order accurate finite volume method for convection–diffusion problems on unstructured meshes. *Mathematical Models and Methods in Applied Sciences 14*, 08 (2004), 1235–1260.

[33] Bertolazzi, E., and Manzini, G. On vertex reconstructions for cell-centered finite volume approximations of 2-D anisotropic diffusion problems. *Math. Mod. Methods Appl. Sci. 17*, 1 (2007), 1–32.

[34] Bischoff, B. S., Botsch, M., Steinberg, S., Bischoff, S., Kobbelt, L., and Aachen, R. Openmesh–a generic and efficient polygon mesh data structure. In *In OpenSG Symposium* (2002).

[35] Bommes, D., Lévy, B., Pietroni, N., Puppo, E., Silva, C., Tarini, M., and Zorin, D. Quad-mesh generation and processing: A survey. In *Computer Graphics Forum* (2013), vol. 32-6, Wiley Online Library, pp. 51–76.

[36] Boost. Boost C++ Libraries. `http://www.boost.org/`, 2021.

[37] Borouchaki, H., Hecht, F., and Frey, P. J. Mesh gradation control. *International Journal for Numerical Methods in Engineering 43*, 6 (1998), 1143–1165.

[38] Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Lévy, B. *Polygon mesh processing*. CRC press, 2010.

[39] Boykov, Y., Veksler, O., and Zabih, R. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence 23*, 11 (2001), 1222–1239.

[40] Bracci, M., Tarini, M., Pietroni, N., Livesu, M., and Cignoni, P. Hexalab. net: An online viewer for hexahedral meshes. *Computer-Aided Design 110* (2019), 24–36.

[41] Brenner, S. C., Guan, Q., and Sung, L.-Y. Some estimates for virtual element methods. *Computational Methods in Applied Mathematics 17*, 4 (2017), 553–574.

[42] Brenner, S. C., and Scott, L. R. *The mathematical theory of finite element methods*, 3 ed. Texts in applied mathematics 15. Springer-Verlag, New York, 2008.

[43] Brenner, S. C., and Sung, L.-Y. Virtual element methods on meshes with small edges or faces. *Mathematical Models and Methods in Applied Sciences 28*, 07 (2018), 1291–1336.

[44] Brewer, M. L., Diachin, L. F., Knupp, P., Leurent, T., and Melander, D. J. The mesquite mesh quality improvement toolkit. In *IMR* (2003).

[45] Brezzi, F., Manzini, G., Marini, D., Pietra, P., and Russo, A. Discontinuous Galerkin approximations for elliptic problems. *Numerical Methods for Partial Differential Equations 16*, 4 (2000), 365–378.

[46] Burkardt, J. CVT dataset. `https://people.sc.fsu.edu/~jburkardt/datasets/cvt/cvt.html`, 2005.

[47] Burkardt, J., Gunzburger, M., Peterson, J., and BRANNON, R. M. User manual and supporting information for library of codes for centroidal voronoi point placement and associated zeroth, first, and second moment determination. Tech. rep., Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2002.

[48] Cabiddu, D., Patané, G., and Spagnuolo, M. A Graphical Framework to Study the Correlation between Geometric Design and Simulation. In *Smart Tools and Applications in Graphics - Eurographics Italian Chapter Conference* (2022), D. Cabiddu, T. Schneider, D. Allegra, C. E. Catalano, G. Cherchi, and R. Scateni, Eds., The Eurographics Association.

[49] Campen, M., Attene, M., and Kobbelt, L. A practical guide to polygon mesh repairing. In *Eurographics (Tutorials)* (2012), p. 32.

[50] Carey, G. F. *Computational Grids: Generations, Adaptation & Solution Strategies*, 1st edition ed. Computational and Physical Processes in Mechanics and Thermal sciEnces. Taylor and Francis, Washington, DC, 1997. ISBN-10: 1560326352.

[51] Chalmeta, R., Hurtado, F., Sacristán, V., and Saumell, M. Measuring regularity of convex polygons. *Computer-Aided Design 45*, 2 (2013), 93–104.

[52] Chan, T. F., Xu, J., and Zikatanov, L. T. An agglomeration multigrid method for unstructured grids. *Contemporary mathematics* (1998).

[53] Chen, J., Zheng, J., Zheng, Y., Xiao, Z., Si, H., and Yao, Y. Tetrahedral mesh improvement by shell transformation. *Engineering with Computers 33*, 3 (2017), 393–414.

[54] Cheng, S.-W., Dey, T. K., Ramos, E. A., and Ray, T. Sampling and meshing a surface with guaranteed topology and geometry. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry* (New York, NY, USA, 2004), SCG '04, Association for Computing Machinery, p. 280–289.

[55] Ciarlet, P. G. *The finite element method for elliptic problems*. SIAM, 2002.

[56] Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G. MeshLab: an Open-Source Mesh Processing Tool. In *Eurographics Italian Chapter Conference* (2008), V. Scarano, R. D. Chiara, and U. Erra, Eds., The Eurographics Association.

[57] Coxeter, H. S. M. Regular skew polyhedra in three and four dimension, and their topological analogues. *Proceedings of the London Mathematical Society 2*, 1 (1938), 33–62.

[58] Deza, M. M., and Deza, E. *Encyclopedia of Distances*. Springer Science & Business Media, 2012.

[59] Di Pietro, D. A., and Droniou, J. The hybrid high-order method for polytopal meshes. *Number 19 in Modeling, Simulation and Application* (2020).

[60] Du, Q., Faber, V., and Gunzburger, M. Centroidal voronoi tessellations: Applications and algorithms. *SIAM review 41*, 4 (1999), 637–676.

[61] Du, Q., Faber, V., and Gunzburger, M. Centroidal voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4 (1999), 637–676.

[62] Du, Q., Wang, D., and Zhu, L. On mesh geometry and stiffness matrix conditioning for general finite element spaces. *SIAM journal on numerical analysis 47*, 2 (2009), 1421–1444.

[63] Dupont, T., and Scott, R. Polynomial approximation of functions in sobolev spaces. *Mathematics of Computation 34*, 150 (1980), 441–463.

[64] Engvall, L., and Evans, J. A. Mesh quality metrics for isogeometric bernstein–bézier discretizations. *Computer Methods in Applied Mechanics and Engineering 371* (2020), 113305.

[65] Erikson, C., and Manocha, D. Gaps: General and automatic polygonal simplification. In *Proceedings of the 1999 symposium on Interactive 3D graphics* (1999), pp. 79–88.

[66] Erten, H., Üngör, A., and Zhao, C. Mesh smoothing algorithms for complex geometric domains. In *Proceedings of the 18th international meshing roundtable*. Springer, 2009, pp. 175–193.

[67] Fabri, A., and Pion, S. Cgal: The computational geometry algorithms library. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (2009), pp. 538–539.

[68] Field, D. A. Qualitative measures for initial meshes. *International Journal for Numerical Methods in Engineering 47*, 4 (2000), 887–906.

[69] Freitag, L. A. On combining laplacian and optimization-based mesh smoothing techniques. Tech. rep., Argonne National Lab., IL (United States), 1997.

[70] Freitag, L. A., and Plassmann, P. Local optimization-based simplicial mesh untangling and improvement. *International Journal for Numerical Methods in Engineering 49*, 1-2 (2000), 109–125.

[71] Freitag Diachin, L., Knupp, P., Munson, T., and Shontz, S. A comparison of two optimization methods for mesh quality improvement. *Engineering with Computers 22*, 2 (2006), 61–74.

[72] Frey, P., and George, P. Mesh generation. application to finite elements. hermes science publ., paris, 2000.

[73] Fried, I. Condition of finite element matrices generated from nonuniform meshes. *AIAA Journal 10* (1972). Technical Note.

[74] Fumagalli, A., and Berre, I. *Numerical methods for processes in fractured porous media*. Birkhäuser, Cham, 2019.

[75] Gao, X., Huang, J., Xu, K., Pan, Z., Deng, Z., and Chen, G. Evaluating hex-mesh quality metrics via correlation analysis. In *Computer Graphics Forum* (2017), vol. 36-5, Wiley Online Library, pp. 105–116.

[76] George, P.-L., and Borouchaki, H. *Delaunay Triangulation and Meshing: Application to Finite Elements*. Technology & Engineering. Hermès, 1998. ISBN: 2866016920, 9782866016920.

[77] Geuzaine, C., and Remacle, J.-F. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering 79*, 11 (2009), 1309–1331.

[78] Gillette, A., Rand, A., and Bajaj, C. Error estimates for generalized barycentric interpolation. *Advances in computational mathematics 37*, 3 (2012), 417–439.

[79] Herrmann, L. R. Laplacian-isoparametric grid generation scheme. *Journal of the Engineering Mechanics Division 102*, 5 (1976), 749–756.

[80] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. Mesh optimization. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1993), SIGGRAPH '93, Association for Computing Machinery, p. 19–26.

[81] Hu, Y., Zhou, Q., Gao, X., Jacobson, A., Zorin, D., and Panozzo, D. Tetrahedral meshing in the wild. *ACM Trans. Graph. 37*, 4 (2018), 60–1.

[82] Huang, W., and Wang, Y. Anisotropic mesh quality measures and adaptation for polygonal meshes. *Journal of Computational Physics 410* (2020), 109368.

[83] Hughes, T. J., Cottrell, J. A., and Bazilevs, Y. Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer methods in applied mechanics and engineering 194*, 39-41 (2005), 4135–4195.

[84] Jou, H.-J. Sas - discrete methods & related tools. `http://www.sai.msu.su/sal/B/2/`, 2001.

[85] Karypis, G., and Kumar, V. Metis: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. *University of Minnesota Digital Conservancy* (1997).

[86] Kim, J., Panitanarak, T., and Shontz, S. M. A multiobjective mesh optimization framework for mesh quality improvement and mesh untangling. *International Journal for Numerical Methods in Engineering 94*, 1 (2013), 20–42.

[87] Klingner, B. M., and Shewchuk, J. R. Aggressive tetrahedral mesh improvement. In *Proceedings of the 16th International Meshing Roundtable* (Berlin, Heidelberg, 2008), M. L. Brewer and D. Marcum, Eds., Springer Berlin Heidelberg, pp. 3–23.

[88] Knupp, P. Matrix norms and the condition number: A general framework to improve mesh quality via node-movement. Tech. rep., Sandia National Lab.(SNL-NM), Albuquerque, NM (United States); Sandia ..., 1999.

[89] Knupp, P. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part ii—a framework for volume mesh optimization and the condition number of the jacobian matrix. *International Journal for numerical methods in engineering 48*, 8 (2000), 1165–1185.

[90] Knupp, P. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part i—a framework for surface mesh optimization. *International Journal for Numerical Methods in Engineering 48*, 3 (2000), 401–420.

[91] Knupp, P. Algebraic mesh quality metrics. *SIAM journal on scientific computing 23*, 1 (2001), 193–218.

[92] Knupp, P. Algebraic mesh quality metrics for unstructured initial meshes. *Finite Elements in Analysis and Design 39*, 3 (2003), 217–241.

[93] Knupp, P. Formulation of a target-matrix paradigm for mesh optimization. Tech. rep., Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2007.

[94] Knupp, P. Introducing the target-matrix paradigm for mesh optimization via node-movement. *Engineering with Computers 28*, 4 (2012), 419–429.

[95] Knupp, P., Ernst, C., Thompson, D. C., Stimpson, C., and Pebay, P. P. The verdict geometric quality library. Tech. rep., Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA ..., 2006.

[96] Kobbelt, L., Campagna, S., and Seidel, H.-P. A general framework for mesh decimation. In *Proceedings of the Graphics Interface 1998 Conference, June 18-20, 1998, Vancouver, BC, Canada* (June 1998), pp. 43–50.

[97] Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., and Panozzo, D. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019), pp. 9601–9611.

[98] Kremer, M., Bommes, D., and Kobbelt, L. Openvolumemesh–a versatile index-based data structure for 3d polytopal complexes. In *Proceedings of the 21st International Meshing Roundtable*. Springer, 2013, pp. 531–548.

[99] Kwok, W., and Chen, Z. A simple and effective mesh quality metric for hexahedral and wedge elements. In *IMR* (2000), pp. 325–333.

[100] Lai, M.-J., and Schumaker, L. L. *Spline functions on triangulations*, vol. 110. Cambridge University Press, 2007.

[101] Lai, M.-J., and Slavov, G. On recursive refinement of convex polygons. *Computer Aided Geometric Design 45* (2016), 83–90.

[102] Ledoux, F. MAMBO. `https://gitlab.com/franck.ledoux/mambo`, 2022.

[103] Lévy, B., and Filbois, A. Geogram: a library for geometric algorithms, 2015.

[104] Li, Z., Wang, Z., Cao, W., and Yao, L. An aspect ratio agglomeration multigrid for unstructured grids. *International Journal for Numerical Methods in Fluids 72*, 10 (2013), 1034–1050.

[105] Lindstrom, P., and Turk, G. Image-driven simplification. *ACM Trans. Graph. 19*, 3 (jul 2000), 204–241.

[106] Liseikin, V. D. *A computational differential geometry approach to grid generation*. Springer Science & Business Media, 2006.

[107] Liseikin, V. D. *Grid generation methods*, vol. 1. Springer, 2017.

[108] Liu, A., and Joe, B. Relationship between tetrahedron shape measures. *BIT Numerical Mathematics 34*, 2 (1994), 268–287.

[109] Livesu, M. cinolib: a generic programming header only C++ library for processing polygonal and polyhedral meshes. *Transactions on Computational Science XXXIV* (2019). https://github.com/mlivesu/cinolib/.

[110] Livesu, M., Pietroni, N., Puppo, E., Sheffer, A., and Cignoni, P. Loopycuts: Practical feature-preserving block decomposition for strongly hex-dominant meshing. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 121–1.

[111] Lo, D. S. Optimization of tetrahedral meshes based on element shape measures. *Computers & structures 63*, 5 (1997), 951–961.

[112] Lo, D. S. Dynamic grid for mesh generation by the advancing front method. *Computers & Structures 123* (2013), 15–27.

[113] Lo, D. S. *Finite element mesh generation*. CRC Press, 2014.

[114] Lobos, C., Arenas, C., Daines, E., and Hitschfeld, N. Measuring geometrical quality of different 3d linear element types. *Numerical Algorithms 90*, 1 (2022), 423–446.

[115] Ma, Y., and Wang, M. An efficient method to improve the quality of tetrahedron mesh with mfrc. *Scientific Reports 11*, 1 (2021), 1–20.

[116] Mandad, M., and Campen, M. Guaranteed-quality higher-order triangular meshing of 2d domains. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–14.

[117] Marchant, M. J., Weatherill, N. P., and Hassan, O. The adaptation of unstructured grids for transonic viscous flow simulation. *Finite Elements in Analysis and Design 25*, 3 (1997), 199–218. Adaptive Meshing, Part 2.

[118] Martin, V., Jaffré, J., and Roberts, J. Modeling fractures and barriers as interfaces for flow in porous media. *SIAM Journal on Scientific Computing 26*, 5 (2005), 1667–1691.

[119] Misztal, M. K., Bærentzen, J. A., Anton, F., and Erleben, K. Tetrahedral mesh improvement using multi-face retriangulation. In *Proceedings of the 18th international meshing roundtable*. Springer, 2009, pp. 539–555.

[120] Mitchell, A., Phillips, G., and Wachspress, E. Forbidden shapes in the finite element method. *IMA Journal of Applied Mathematics 8*, 2 (1971), 260–269.

[121] Mitchell, S., and Vavasis, S. Quality mesh generation in three dimensions. *Proceedings of the ACM Computational Geometry Conference* (1999).

[122] Moës, N., Remacle, J.-F., Lambrechts, J., Lé, B., and Chevaugeon, N. The extreme mesh deformation approach (x-mesh). *arXiv preprint arXiv:2111.04179* (2023).

[123] Myles, A., Pietroni, N., Kovacs, D., and Zorin, D. Feature-aligned t-meshes. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 1–11.

[124] Nielson, G. M. Tools for triangulation and tetrahedrizations and constructing functions defined over them. In *Scientific Visualization*, G. M. Nielson, H. Hagen, and H. Müller, Eds. IEEE Computer Society, Los Alamitos, CA, 1997, pp. 429–525.

[125] Oddy, A., Goldak, J., McDill, M., and Bibby, M. A distortion metric for isoparametric finite elements. *Transactions of the Canadian Society for Mechanical Engineering 12*, 4 (1988), 213–217.

[126] Owen, S. J. A survey of unstructured mesh generation technology. *IMR 239* (1998), 267.

[127] Parthasarathy, V., Graichen, C., and Hathaway, A. A comparison of tetrahedron quality measures. *Finite Elements in Analysis and Design 15*, 3 (1994), 255–261.

[128] Pébay, P. Planar quadrilateral quality measures. *Engineering with Computers 20*, 2 (2004), 157–173.

[129] Pébay, P., and Baker, T. Analysis of triangle quality measures. *Mathematics of computation 72*, 244 (2003), 1817–1839.

[130] Pietroni, N., Campen, M., Sheffer, A., Cherchi, G., Bommes, D., Gao, X., Scateni, R., Ledoux, F., Remacle, J.-F., and Livesu, M. Hex-mesh generation and processing: a survey. *ACM Transactions on Graphics* (2022).

[131] Preiss, K. Topological consistency rules for general finite element meshes. In *CAD82*, A. Pipes, Ed. Butterworth-Heinemann, 1982, pp. 453–460.

[132] Preparata, F. P., and Shamos, M. I. *Computational Geometry: An Introduction*. Springer-Verlag, Berlin, Heidelberg, 1985.

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

[134] Robinson, J. Cre method of element testing and the jacobian shape parameters. *Engineering Computations* (1987).

[135] Robinson, J. Some new distortion measures for quadrilaterals. *Finite Elements in Analysis and Design 3*, 3 (1987), 183–197.

[136] Robinson, J. Quadrilateral and hexahedron shape parameters. *Finite elements in analysis and design 16*, 1 (1994), 43–52.

[137] Roca, X., Gargallo-Peiró, A., and Sarrate, J. Defining quality measures for high-order planar triangles and curved mesh generation. In *Proceedings of the 20th International Meshing Roundtable*. Springer, 2011, pp. 365–383.

[138] Ronfard, R., and Rossignac, J. Full-range approximation of triangulated polyhedra. In *Computer Graphics Forum* (1996), vol. 15-3, Wiley Online Library, pp. 67–76.

[139] Rossignac, J., and Borrel, P. Multi-resolution 3d approximations for rendering complex scenes. In *Modeling in Computer Graphics* (Berlin, Heidelberg, 1993), B. Falcidieno and T. L. Kunii, Eds., Springer Berlin Heidelberg, pp. 455–465.

[140] Rycroft, C. Voro++. `https://math.lbl.gov/voro++/about.html`, 2008.

[141] Salagame, R. R., and Belegundu, A. D. Distortion, degeneracy and rezoning in finite elements—a survey. *Sadhana 19*, 2 (1994), 311–335.

[142] Sastry, S. P., and Shontz, S. M. A comparison of gradient-and hessian-based optimization methods for tetrahedral mesh quality improvement. In *Proceedings of the 18th International Meshing Roundtable*. Springer, 2009, pp. 631–648.

[143] Scarlatos, L. L., and Pavlidis, T. Optimizing triangulations by curvature equalization. In *Proceedings of the 3rd conference on Visualization'92* (1992), pp. 333–339.

[144] Schneider, T., Dumas, J., Gao, X., Botsch, M., Panozzo, D., and Zorin, D. Poly-spline finite-element method. *ACM Transactions on Graphics (TOG) 38*, 3 (2019), 1–16.

[145] Schneider, T., Dumas, J., Gao, X., Zorin, D., and Panozzo, D. PolyFEM. `https://polyfem.github.io/`, 2019.

[146] Schneider, T., Hu, Y., Dumas, J., Gao, X., Panozzo, D., and Zorin, D. Decoupling simulation accuracy from mesh quality. *ACM Transactions on Graphics 37*, 6 (10 2018).

[147] Schneider, T., Hu, Y., Gao, X., Dumas, J., Zorin, D., and Panozzo, D. A large-scale comparison of tetrahedral and hexahedral elements for solving elliptic pdes with the finite element method. *ACM Trans. Graph. 41*, 3 (mar 2022).

[148] Shakour, E., and Amir, O. Topology optimization with precise evolving boundaries based on iga and untrimming techniques. *Computer Methods in Applied Mechanics and Engineering 374* (2021), 113564.

[149] Shephard, M. S. Approaches to the automatic generation and control of finite element meshes. *Applied Mechanics Reviews 41*, 4 (1988), 169–185.

[150] Shepherd, J. F., and Johnson, C. R. Hexahedral mesh generation constraints. *Engineering with Computers 24*, 3 (2008), 195–213.

[151] Shewchuk, J. R. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. In *Workshop on applied computational geometry* (1996), Springer, pp. 203–222.

[152] Shewchuk, J. R. What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures. *University of California at Berkeley 2002* (2002).

[153] Sirois, Y., Dompierre, J., Vallet, M.-G., and Guibault, F. Hybrid mesh smoothing based on riemannian metric non-conformity minimization. *Finite elements in analysis and design 46*, 1-2 (2010), 47–60.

[154] Sorgente, T., Berrone, S., Biasotti, S., Manzini, G., Spagnuolo, M., and Vicini, F. Mesh optimization for the virtual element method: How small can an agglomerated mesh become? *arXiv preprint arXiv:2404.11484* (2024).

[155] Sorgente, T., Biasotti, S., Manzini, G., and Spagnuolo, M. VEM 2D Dataset. `https://github.com/TommasoSorgente/vem-2D-quality-dataset`, 2021.

[156] Sorgente, T., Biasotti, S., Manzini, G., and Spagnuolo, M. Polyhedral mesh quality indicator for the virtual element method. *Computers & Mathematics with Applications 114* (2022), 151–160.

[157] Sorgente, T., Biasotti, S., Manzini, G., and Spagnuolo, M. The role of mesh quality and mesh quality indicators in the virtual element method. *Advances in Computational Mathematics 48*, 1 (2022), 1–34.

[158] Sorgente, T., Biasotti, S., Manzini, G., and Spagnuolo, M. VEM 3D Dataset. `https://github.com/TommasoSorgente/vem-3D-quality-dataset`, 2022.

[159] Sorgente, T., Biasotti, S., Manzini, G., and Spagnuolo, M. A survey of indicators for mesh quality assessment. In *Computer Graphics Forum* (2023), vol. 42-2, Wiley Online Library, pp. 461–483.

[160] Sorgente, T., Prada, D., Cabiddu, D., Biasotti, S., Patané, G., Pennacchio, M., Bertoluzza, S., Manzini, G., and Spagnuolo, M. *VEM and the Mesh*, vol. 31 of *SEMA SIMAI Springer series*. Springer International Publishing, 2022, ch. 1, pp. 1–57.

[161] Sorgente, T., Vicini, F., Berrone, S., Biasotti, S., Manzini, G., and Spagnuolo, M. Mesh quality agglomeration algorithm for the virtual element method applied to discrete fracture networks. *Calcolo 60*, 2 (2023), 27.

[162] Stander, B. T., and Hart, J. C. Guaranteeing the topology of an implicit surface polygonization for interactive modeling. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., p. 279–286.

[163] Staten, M. L., Owen, S. J., Shontz, S. M., Salinger, A. G., and Coffey, T. S. A comparison of mesh morphing methods for 3d shape optimization. In *Proceedings of the 20th international meshing roundtable*. Springer, 2011, pp. 293–311.

[164] Stimpson, C., Ernst, C., Knupp, P., Pébay, P., and Thompson, D. The verdict library reference manual. *Sandia National Laboratories Technical Report 9*, 6 (2007).

[165] Sukumar, N., and Tabarraei, A. Conforming polygonal finite elements. *International Journal for Numerical Methods in Engineering 61*, 12 (2004), 2045–2066.

[166] Sun, L., Zhao, G., and Ma, X. Quality improvement methods for hexahedral element meshes adaptively generated using grid-based algorithm. *International Journal for Numerical Methods in Engineering 89*, 6 (2012), 726–761.

[167] Talischi, C., Paulino, G. H., Pereira, A., and Menezes, I. F. Polymesher: a general-purpose mesh generator for polygonal elements written in matlab. *Structural and Multidisciplinary Optimization 45*, 3 (2012), 309–328.

[168] Tangelder, J. W., and Veltkamp, R. C. A survey of content based 3d shape retrieval methods. *Multimedia tools and applications 39*, 3 (2008), 441–471.

[169] Thacker, W. C. A brief review of techniques for generating irregular computational grids. *International Journal for Numerical Methods in Engineering 15*, 9 (1980), 1335–1341.

[170] Vartziotis, D., Athanasiadis, T., Goudas, I., and Wipper, J. Mesh smoothing using the geometric element transformation method. *Computer Methods in Applied Mechanics and Engineering 197*, 45-48 (2008), 3760–3767.

[171] Vartziotis, D., and Himpel, B. Efficient mesh optimization using the gradient flow of the mean volume. *SIAM J. Numer. Anal. 52*, 2 (2014), 1050–1075.

[172] Vartziotis, D., and Wipper, J. Characteristic parameter sets and limits of circulant hermitian polygon transformations. *Linear algebra and its applications 433*, 5 (2010), 945–955.

[173] Vartziotis, D., and Wipper, J. Fast smoothing of mixed volume meshes based on the effective geometric element transformation method. *Computer methods in applied mechanics and engineering 201* (2012), 65–81.

[174] Wang, L. *Algorithms and Criteria for Volumetric Centroidal Voronoi Tessellations*. PhD thesis, Université Grenoble Alpes, 2017.

[175] Whalen, E., Beyene, A., and Mueller, C. Simjeb: Simulated jet engine bracket dataset. In *Computer Graphics Forum* (2021), vol. 40-5, Wiley Online Library, pp. 9–17.

[176] Wood, Z., Hoppe, H., Desbrun, M., and Schröder, P. Removing excess topology from isosurfaces. *ACM Transactions on Graphics (TOG) 23*, 2 (2004), 190–208.

[177] Xu, H., and Newman, T. S. An angle-based optimization approach for 2d finite element mesh smoothing. *Finite Elements in Analysis and Design 42*, 13 (2006), 1150–1164.

[178] Yan, D.-M., Wang, W., Lévy, B., and Liu, Y. Efficient computation of clipped voronoi diagram for mesh generation. *Computer-Aided Design 45*, 4 (2013), 843–852.

[179] Zandsalimy, M., and Ollivier-Gooch, C. A novel approach to mesh optimization to stabilize unstructured finite volume simulations. *Journal of Computational Physics 453* (2022), 110959.

[180] Zhou, Q., and Jacobson, A. Thingi10k: A dataset of 10,000 3D-printing models, 2016.

[181] Zlámal, M. On the finite element method. *Numerische Mathematik 12*, 5 (1968), 394–409.

[182] Zunic, J., and Rosin, P. L. A new convexity measure for polygons. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 7 (2004), 923–934.