High-order CAD-based surface mesh adaptation with the log-simplex method

Olivier Coulaud

Abstract

The present article studies the problem of approximating 3D surface models with meshes composed of curved triangles of arbitrary order. The considered process derives from a high-order solution-based mesh adaptation method called log-simplex method. In this case, it is locally applied on a specific 2D high-order solution, which is built from the features of the model and is defined on the tangent plane to the surface. This way, for a given mesh complexity, an optimal metric field is computed, which then drives the mesh adaptation procedure.

1 Introduction

When numerically computing a physical phenomenon by using a finite elements method, one of the most significant prerequisite lies in the ability to generate a mesh which both represents well the computational domain and makes the numerical method converge and approximate well the true physical solution of the problem. In practice, in most of the challenging numerical computations, the most human time consuming task is the generation of a suitable mesh, which usually requires to manually tune meshing parameters, or manually define specific areas where the mesh should be refined or coarsened, based on engineer knowledge. In order to make this work automatic, mesh adaptation techniques have been developed through the years [8, 14, 17, 16, 25, 26]. The main idea behind those methods consists in modifying an initial mesh so that it minimizes the interpolation error of a solution, induced by the projection of the analytical solution onto a finite elements space, which is usually locally polynomial. Among those methods, the Hessian-based methods (see for instance [16, 25, 26]), for which the mesh adaptation process is driven by a metric field derived from the Hessian matrix of the solution, have been largely used in both academical and industrial cases [5, 28, 29, 36]. However, when high-order finite elements are considered, that is to say the solution is approximated by polynomials of degree k, $k \geq 2$, the problem is more complicated. Actually, as shown in [10], the interpolation error depends on the high-order $k+1$ differential of the solution, which is a homogeneous polynomial of degree $k+1$. Unfortunately, on the contrary to the Hessian-based methods, this high-order differential has no direct transcription in terms of metric space, and the problem has been remained open for quite a long time. Recently, several usable ways to extend the Hessian-based methods to the high-order case have been introduced [9, 12, 13, 31], finally enabling to derive a metric-field from the high-order differential of the solution. This way, when using meshes composed of simplicial linear elements (straight triangles or tetrahedra), these methods enable to generate meshes adapted to arbitrary high-order solutions, improving the robustness of the high-order numerical schemes and drastically reducing the massive numerical cost induced by the use of high-order finite elements.

As an additional issue to the high-order case, a key problem has been exhibited by Bassi and Rebay in [7]. According to these authors, high-order schemes such as Discontinuous Galerkin [21] or Spectral Differences [22] methods applied on a domain represented by straight triangles or tetrahedra may not converge, independently from the level of mesh refinement. In other words, adapting a mesh with respect to a high-order solution may not be relevant if the mesh is composed of straight elements, especially at the boundary of the domain. One naive idea would be just curving the boundary mesh through projection on an analytical boundary, but this would produce very low-quality meshes, usually involving non-valid curved elements. Consequently, there is a need to truly adapt the boundary mesh with respect to the features of the analytical boundary. Though the case of usual straight elements has been largely investigated [6, 11, 23, 34, 37], the high-order case is quite a new field, where plenty of specific aspects need to be addressed. The present article gives an answer to this later problem. More precisely, from a parameterized surface model and an initial mesh composed of straight triangles, a mesh adaptation procedure is proposed, enabling to optimally approximate the surface model by a mesh composed of high-order curved triangles, under the constraint of a given mesh complexity. In some sense, this article is a completion to the work presented in [12] and [13], in which the problem of adapting a mesh with respect to high-order finite elements solutions by minimizing the high-order interpolation error is addressed. Combining the two approaches would enable to fully exploit the high-order numerical methods, and take benefit of the high-precision they provide. Actually, the surface mesh adaptation method presented in this article takes advantage of the high-order mesh adaptation technique introduced in [12, 13], which is called the Log-simplex method, and is a way to derive an optimal metric field from a high-order solution, which then drives the adaptation procedure into the generation of an optimal mesh which minimizes the interpolation error. This method is a natural extension to the classical Hessian-based methods described above, which are devoted to the mesh adaptation with respect to $\mathbb{P}_1$ finite elements spaces. In order to address the problem emphasized by Bassi and Rebay in [7], the Log-simplex method is used in a very specific manner, which enables to control the error induced by the representation of a Computer Aided Design (CAD) by a mesh composed by a restricted number of curved triangles with arbitrary polynomial degree $k \geq 1$. The main idea consists in locally considering a 2D high-order solution on the tangent plane of the surface, which measures the signed distance between the surface and the tangent-plane, in the normal direction to the surface. From this high-order solution, the Log-simplex method enables to derive an optimal 2D metric field, which is rescaled and remapped into the 3D physical space. Finally, using this 3D optimal metric field, an optimal high-order curved mesh is produced through local mesh modifications, such as edge splits, collapses or swaps. Notice that a first extension of the Log-simplex method to the high-order CAD-based mesh adaptation has been introduced in [18], following a slightly different approach. The method presented in this paper allows to handle complex geometries, even with low-quality CAD involving singularities in their parameters spaces.

This article is organised as follows. In order to keep this article as self-contained as possible, Section 2 gives some recalls about the metric-based adaptation techniques. Then, the complete error-model is described in Section 3, together with its link with the Log-simplex method, which is briefly explained. The numerical implementation is detailed in Section 4, after what the numerical results are exposed in Section 5. Finally, Section 6 gives a conclusion to this work, and some future developments are proposed.

2 Metric-based mesh adaptation

In the present article, the mesh adaptation process which is enlighten is based on the generation of a metric-field. As it is described in Section 3, the surface mesh problem can be reduced to the generation of an adapted mesh with respect to a solution u, interpolated on a polynomial finite elements space of order $k \geq 1$. In this section, we give some recalls about the metric-based mesh adaptation, following the framework of the continuous mesh model, which is for instance described by Alauzet and Loseille in [25, 26]. The main principle of such a mesh adaptation method relies on the definition of the geometrical quantities through a distance linked to a metric $\mathcal{M}$, that is to say a symmetric positive definite matrix. More precisely for a d-dimensional metric $\mathcal{M}$, the scalar product $\langle ., .\rangle_{\mathcal{M}}$ is considered, given by

$$\langle x,\, y\rangle_{\mathcal{M}} = x^t \mathcal{M}\, y, \quad \text{for all} \quad x, y \in \mathbb{R}^d. \tag{1}$$

Likewise, a $\mathcal{M}$-dependent norm $|.|_{\mathcal{M}}$ and a derived distance $d_{\mathcal{M}}$ may be deduced from $\langle ., .\rangle_{\mathcal{M}}$, which are used to compute all the geometrical values, such as the lengths, areas or volumes.

When $\mathcal{M}$ is fixed throughout the domain $\Omega \subset \mathbb{R}^d$, then the metric space $(\Omega, d_{\mathcal{M}})$ is an Euclidean metric-space. A natural extension to the Euclidean metric-space lies in the definition of continuous variable metric map $\mathbf{M} = (\mathcal{M}(x))_{x\in\Omega}$ over Ω. In this case, we talk about a Riemannian metric space and the length of a segment ab with respect to $\mathbf{M}$ may be computed through an integral formulation. Indeed, let $\gamma : [0,1] \to \mathbb{R}^d$ such that $\gamma(s) = (1-s)a + sb$ be the curve travelling along the segment ab. The length of ab with respect to $\mathbf{M}$ is then given by

$$\ell_{\mathbf{M}}(ab) = \int_0^1 |\gamma'(s)|_{\mathcal{M}(s)} ds = \int_0^1 \sqrt{ab^t \mathcal{M}(\gamma(s)) ab}\, ds. \tag{2}$$

Likewise, the notions of area or volume have their respective definitions in terms of Riemannian metric spaces, also related to integral formula. The mesh adaptation which is considered in this paper is then related to this framework. For instance, the admissible length of an edge is always related to a specific metric-space, and the decision to split or collapse this later strongly depends on $\mathbf{M}$ around this edge. More precisely, the link between a mesh and a metric-field is then established by the definition of a unit element with respect to a metric.

Definition 2.1. *A simplex element $\mathcal{T}$ (a triangle in 2D, a tetrahedron in 3D) is said to be unit with respect to a metric space $\mathbf{M} = (\mathcal{M}(x))_{x\in\Omega}$ if every edge e of $\mathcal{T}$ satisfies $\ell_{\mathbf{M}}(e) = 1$.*

Likewise, a unit mesh with respect to a Riemannian metric-space $\mathbf{M}$ is mesh composed of unit elements with respect to $\mathbf{M}$. Notice that, from a practical point of view, a strict unit mesh is never considered, but replaced with the relaxed notion of quasi-unit mesh, composed of elements whose edges lengths are close to 1. In this paper, every presented adapted mesh is quasi-unit in the sense that its edges lengths belong to the interval $\left[\sqrt{2}, \frac{\sqrt{2}}{2}\right]$. Actually, this quasi-unit compromise is even necessary in 3D, since it is in general not possible to mesh an arbitrary volume with equilateral tetrahedra. The duality between a mesh and a metric-field is closer than it looks, since most of the notions usually devoted to the description of a mesh can be translated in terms of metric-fields. For instance, a triangle $\mathcal{T}$ which is unit with respect 2D metric $\mathcal{M}$ satisfies $|\mathcal{T}| = \frac{\sqrt{3}}{4}\det(\mathcal{M})^{-\frac{1}{2}}$. Likewise, the complexity of a mesh has its translation in terms of metric-space. It is given by

$$\mathcal{C}(\mathcal{M}) = \int_\Omega \sqrt{\det(\mathcal{M}(x))} dx. \tag{3}$$

According to (3), the complexity N increases together with the determinant of $\mathcal{M}$. Since the area of the unit ball $\mathcal{B}_{\mathcal{M}}$ of a metric $\mathcal{M}$ is given by $|\mathcal{B}_{\mathcal{M}}| = \pi (\det(\mathcal{M}))^{-\frac{1}{2}}$, a large determinant actually means a small unit ball. Thus, (3) can be written

$$\mathcal{C}(\mathcal{M}) = \pi \int_\Omega \frac{1}{|\mathcal{B}_{\mathcal{M}}(x)|} dx. \tag{4}$$

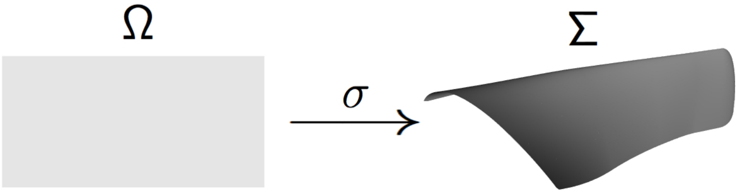

Figure 1: A regular surface Σ of $\mathbb{R}^3$ and its parameters space Ω.

Hence, the analogy between the complexity of a metric-space and the number of elements of a unit mesh with respect to it is actually quite natural.

3 High-order surface error model

In this section, we consider a regular parameterised surface $\Sigma \subset \mathbb{R}^3$, whose parameters space is denoted $\Omega \subset \mathbb{R}^2$. It is defined by σ, its smooth parameterisation function, such that

$$\begin{array}{rcccl} \sigma: & \Omega & \longrightarrow & \Sigma & \\ & (u,v) & \longmapsto & \sigma(u,v) & = \begin{pmatrix} \sigma_1(u,v) \\ \sigma_2(u,v) \\ \sigma_3(u,v) \end{pmatrix}. \end{array} \tag{5}$$

This surface representation is shown in Fig. 1. In the present paper, we look for the best representation of such a surface by a triangular mesh with given complexity $N > 0$, whose elements have an arbitrary order. One way to address this problem is to first consider an adapted 2D mesh of the parameters space Ω, and then to use the mapping σ to obtain an adapted 3D mesh of Σ. This way, it is quite easy to curve the mesh and reach a given order, by inserting nodes in the straight elements of the 2D mesh and map these nodes by applying σ. Of course, there still is the problem of how to produce a good adapted mesh of Ω, such that its mapping gives a good representation of Σ, once mapped onto it. This kind of methods has been notably studied in [18], where the authors define a specific metric-field in the parameters space, which they use to produce an adapted mesh of Ω. The definition of such a metric space is locally obtained by making a high-order Taylor expansion of the second fundamental form at a given point of Ω, which measures the gap between the surface and its tangent plane. In particular, the metric space they obtain is independent from the parameterization, and only depends on geometric features of the surface.

One major issue of considering a mesh in the parameters space before mapping it onto Σ, comes when the geometric model is more complex and consists in several surfaces linked between them by curves. In this particular case, each surface owns its parameters space, whose boundaries are generally in common with other surfaces of the model. Consequently, when making mesh adaptation in these parameters spaces, the boundaries must be constrained, which needs to first consider a mesh adaptation on the lines of the model only. In the present paper, the opposite approach is considered: a 3D metric space is defined, which is then transcribed into a 2D one in the parameters space, when needed (for instance for computing edge lengths). In particular, all the adaptation work is done in the 3D framework, which allows to consider the surfaces and the boundary lines at the same time. This enables to produce high-order adapted meshes on complex geometries, as shown in Section 5.

3.1 Alternative surface mapping

The considered error model of this article relies on the construction of an alternate local mapping $\bar{\sigma} : \mathcal{U} \subset \mathbb{R}^2 \to \Sigma$, where $\mathcal{U}$ is a 2D neighborhood of $(0,0)$, which depends on the location on Σ. This mapping is built through the use of the initial mapping σ, but only the point evaluation of σ is called, and not the derivatives. Indeed, these later may vanish and lead to a dead-end. In the present article, the mapping is locally built on the tangent plane to Σ, leading to the definition of the smooth $\bar{\sigma}$ on $\mathcal{U}$, which truly depends on the geometrical features of Σ. Then, once this mapping is computed, all the adaptation metric-field computation is done with respect to it. In order to exhibit this alternative mapping, a local gap-function is considered, which measures the distance between the tangent plane and the surface itself. More precisely, if σ is differentiable at $x_0 = \sigma(u_0, v_0)$ and $\partial_u \sigma(u_0, v_0)$ and $\partial_v \sigma(u_0, v_0)$ are non-colinear vectors, then the tangent plane $\mathcal{P}(x_0)$ with respect to Σ is spanned by the orthonormal vectors $\vec{u}$ and $\vec{v}$, given by

$$\begin{aligned} \vec{u} &= \frac{\partial_u \sigma(u_0, v_0)}{|\partial_u \sigma(u_0, v_0)|_2}, \\ \vec{v} &= \frac{\partial_v \sigma(u_0, v_0) - \langle \partial_v \sigma(u_0, v_0), \vec{u} \rangle \, \vec{u}}{|\partial_v \sigma(u_0, v_0) - \langle \partial_v \sigma(u_0, v_0), \vec{u} \rangle \, \vec{u}|_2}, \end{aligned} \tag{6}$$

where $|.|_2$ denotes the Euclidean norm and $\langle ., . \rangle$ the usual 3D scalar product. Notice that, in practical, the derivatives of σ are not used to compute $(\vec{u}, \vec{v})$, but one relies on the underlying mesh instead. The couple $(\vec{u}, \vec{v})$ can be completed into an orthonormal basis of $\mathbb{R}^3$, by introducing the normal vector

$$\vec{n} = \frac{\vec{u} \times \vec{v}}{|\vec{u} \times \vec{v}|_2}, \tag{7}$$

where × stands for the usual cross product. From this construction, every point $x \in \mathbb{R}^3$ can be expressed in the coordinates system $(x_0, \vec{u}, \vec{v}, \vec{n})$ and $(\bar{u}, \bar{v}, \bar{n})_{\mathcal{P}}$ denotes its coordinates. In this change of variables, the tangent plane corresponds to $\mathcal{P}(x_0) = \{(\bar{u}, \bar{v}, 0)_{\mathcal{P}} : (\bar{u}, \bar{v}) \in \mathbb{R}^2\}$, and the point x_0 is given by $(0,0,0)_{\mathcal{P}}$.

The main idea of this paper relies on the definition of an error estimate on $\mathcal{P}(x_0)$. More precisely, for a given point $x_0 = \sigma(u_0, v_0) \in \Sigma$, let $\xi : \mathcal{P}(x_0) \to \mathbb{R}$ denotes the function which measures the gap between $\mathcal{P}(x_0)$ and Σ, in the direction of $\vec{n}$. Then, the local alternate mapping $\bar{\sigma} : \mathcal{U} \to \Sigma$ is defined by

$$\bar{\sigma}(\bar{u}, \bar{v}) = x_0 + \bar{u}\vec{u} + \bar{v}\vec{v} + \xi(\bar{u}, \bar{v})\vec{n}. \tag{8}$$

Figure 2 gives a representation of such a gap function ξ. Though there is no explicit formula for ξ, its existence is ensured by the local inverse function theorem. Indeed, it suffices to show that the orthogonal projection onto $\mathcal{P}(x_0)$, given by

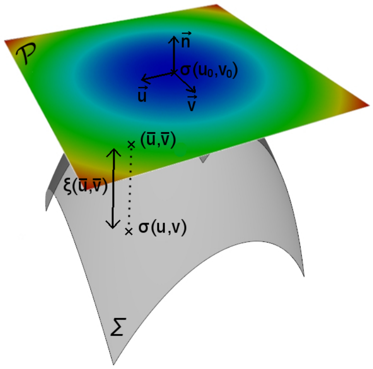

Figure 2: Parameterised surface Σ and the error function ξ on its tangent plane $\mathcal{P}(x_0)$ at the point $x_0 = \sigma(u_0, v_0)$. The colours correspond to the values of ξ, which actually measures the distance from $\mathcal{P}(x_0)$ to Σ, according to the normal direction $\vec{n}$.

$$\begin{array}{rccl} \Pi_{\mathcal{P}} : & \Omega & \to & \mathcal{P}(x_0) \\ & (u,v) & \mapsto & (\bar{u}, \bar{v}) = \begin{pmatrix} \langle \sigma(u,v) - x_0, \vec{u} \rangle \\ \langle \sigma(u,v) - x_0, \vec{v} \rangle \end{pmatrix} \end{array}$$

is invertible in a neighbourhood $\mathcal{U} \subset \Omega$ of (u_0, v_0). Obviously, $\Pi_{\mathcal{P}}$ has the same regularity as σ. Moreover, if σ is differentiable and $\partial_u \sigma(u_0, v_0)$ and $\partial_v \sigma(u_0, v_0)$ are non-colinear, we can easily check that the Jacobian matrix

$$\nabla \Pi_{\mathcal{P}}(u_0, v_0) = \begin{pmatrix} \langle \partial_u \sigma(u_0, v_0), \vec{u} \rangle & \langle \partial_v \sigma(u_0, v_0), \vec{u} \rangle \\ \langle \partial_u \sigma(u_0, v_0), \vec{v} \rangle & \langle \partial_v \sigma(u_0, v_0), \vec{v} \rangle \end{pmatrix}$$

is invertible. Consequently, according to the usual local inverse function theorem, there exist a neighbourhood $\mathcal{V}(u_0, v_0) \subset \Omega$ of (u_0, v_0) and a neighbourhood $\mathcal{U}(0, 0) \subset \mathcal{P}(x_0)$ of $(0, 0)$ such that $\Pi_{\mathcal{P}} : \mathcal{V}(u_0, v_0) \to \mathcal{U}(0, 0)$ is invertible. Moreover, the inverse mapping $\Pi_{\mathcal{P}}^{-1}$ has the same regularity as $\Pi_{\mathcal{P}}$. This way, ξ can be written

$$\xi(\bar{u}, \bar{v}) = \left\langle \sigma(\Pi_{\mathcal{P}}^{-1}(\bar{u}, \bar{v})) - (x_0 + \bar{u}\vec{u} + \bar{v}\vec{v}) \, , \vec{n} \right\rangle . \tag{9}$$

As explained above, though the theoretical justification of the existence of ξ and $\bar{\sigma}$ assumes that the derivatives of σ are well defined and non-colinear, the tangent plane and its constitutive vectors are obtained from the initial mesh, in practise. This way, even if the actual mapping σ is not regular, it allows to build a smooth version of ξ, by using the piecewise evaluation of σ only.

3.2 Local error model

In order to minimize the interpolation error due to the discretization of the surface Σ by a mesh, which is basically the difference between the surface and the mesh representing it, it is shown in this section that it suffices to locally control the interpolation error of ξ on the tangent plane to Σ. More precisely, let $x_0 = \sigma(u_0, v_0)$ be a vertex of the mesh $\mathcal{H}$, $\mathcal{P}(x_0)$ be the tangent plane to σ at x_0 and ξ be the local gap function described in Section 3.1. Then, let consider $H_{\mathcal{P}}$, the projection of the sub-mesh composed of the triangles surrounding x_0 onto $\mathcal{P}(x_0)$ with respect to the normal vector $\vec{n}$. Notice that the validity of $\mathcal{H}_{\mathcal{P}}$ notably depends on the refinement level of $\mathcal{H}$ around x_0. Throughout this theoretical section, a validity assumption is done on $\mathcal{H}_{\mathcal{P}}$, but this issue may occur in a numerical implementation, when dealing with coarse meshes. On the local planar mesh $\mathcal{H}_{\mathcal{P}}$, consider $\bar{\sigma}$ as defined by (8) and its k-order polynomial interpolation $\Pi_k \bar{\sigma}$ over $\mathcal{H}_{\mathcal{P}}$. More precisely, each of the three components of $\bar{\sigma}$ is approximated by a k-degree polynomial representation, obtained by the projection of $\bar{\sigma}$ onto the finite elements space

$$\mathbb{P}_k(\mathcal{H}_{\mathcal{P}}) = \left\{ \varphi : \mathcal{P}(x_0) \cap \mathcal{H}_{\mathcal{P}} \to \mathbb{R} \,|\, \varphi_{|\mathcal{T}} \text{ is a polynomial of degree k} \right\} .$$

Performing a $(k + 1)$-order Taylor expansion of ξ and injecting it into (8), it comes

$$\bar{\sigma}(\bar{u}, \bar{v}) = x_0 + \bar{u}\vec{u} + \bar{v}\vec{v} + \left(\sum_{i=0}^{k} d^{(i)}\xi(0,0)(\bar{u}, \bar{v})\vec{n} + d^{(k+1)}\xi(0,0)(\bar{u}, \bar{v}) + o\left(|(\bar{u}, \bar{v})|_2^{k+1} \right) \right) \vec{n}, \tag{10}$$

where $d^{(i)}\xi(0,0)$ stands for the i-order differential of ξ at $(0,0)$, which is a homogeneous polynomial of degree i given by

$$d^{(i)}\xi(0,0)(\bar{u},\bar{v}) = \sum_{j=0}^{i} \left(\begin{array}{c} i \\ j \end{array} \right) \frac{\partial}{\partial^j \bar{u}\, \partial^{i-j} \bar{v}} \xi(0,0) \bar{u}^j \bar{v}^{i-j}. \tag{11}$$

Then, for $k \geq 1$, since $|\vec{n}|_2 = 1$, the difference between $\bar{\sigma}$ and its k-order interpolation $\Pi_k \bar{\sigma}$ leads to

$$|(\bar{\sigma} - \Pi_k \bar{\sigma})(\bar{u},\bar{v})|_2 \leq C \left| d^{(k+1)}\xi(0,0)(\bar{u},\bar{v}) \right| + o\left(|(\bar{u},\bar{v})|_2^{k+1} \right), \tag{12}$$

where C is a positive constant. Consequently, the interpolation error of $\bar{\sigma}$ induced by its projection onto $\mathbb{P}_k(\mathcal{H}_{\mathcal{P}})$ is actually governed by the differential form of ξ of order $k+1$. Notice that Inequality (12) actually leads the high-order mesh adaptation with respect to the 2-dimensional function ξ, as explained in [12]. Therefore, minimizing this interpolation error is equivalent to minimizing the interpolation error of order k of the 2-dimensional solution ξ. The main difficulty of the anisotropic mesh adaptation with respect to high-order solutions stands in the translation of the inequality (12) in terms of metric space. Indeed, when dealing with $\mathbb{P}_1$ elements, then the $k+1$ differential of the solution is nothing but the Hessian matrix, which almost writes as a metric. On the contrary, if $k > 1$, there is no obvious geometrical interpretation of the differential, which is why the log-simplex method has been introduced in [12]. The principle consists in finding a metric $\mathcal{Q} \in \mathcal{M}_{2,2}(\mathbb{R})$ which bounds the right hand side of (12), that is to say

$$\left| d^{(k+1)}\xi(0,0)(\bar{u},\bar{v}) \right| \leq \left| {}^t(\bar{u},\bar{v})\, \mathcal{Q}\, (\bar{u},\bar{v}) \right|^{\frac{k+1}{2}}, \quad \text{for all} \quad (\bar{u},\bar{v}) \in \mathbb{R}^2. \tag{13}$$

Though the right hand side of (13) is not polynomial, notice that it is homogeneous of order $k+1$, just like the left hand side. In Section (3.4), some recalls about the log-simplex method are given, but we refer to [12] for a full theoretical and numerical description of it.

3.3 Global error model

Once the metric $\mathcal{Q}$ has been computed for each vertex of the mesh $\mathcal{H}$, it is time to derive the optimal planar metric $\mathcal{M}_{opt}^{\mathcal{P}}$ in the 3D physical space. This later will not be directly used to perform mesh adaptation, since it is now flat, and needs to be inflated in the normal direction to the tangent plane. Since this is a 2D metric space, obtained by minimizing the interpolation error of the function ξ, it has to be globally normalized following the 2D version of the framework given by [12, 24]. According to those references, considering a mesh $\mathcal{H}$ of

$\Omega \subset \mathbb{R}^2$ which is unit with respect to a metric space $(\mathcal{M}(x))_{x\in\Omega}$ and a function $u : \Omega \to \mathbb{R}$, the L^p interpolation error $\|u - \Pi_k u\|_{L^p(\Omega)}$ of order $k+1$ is equivalent to the functional

$$\mathcal{E}(\mathcal{M}) = \left(\int_\Omega \left| \text{trace} \left(\mathcal{M}^{-\frac{1}{2}} \, \mathcal{Q} \, \mathcal{M}^{-\frac{1}{2}} \right)^{\frac{p(k+1)}{2}} \right| \right)^{\frac{1}{p}}, \tag{14}$$

where $\mathcal{Q}$ satisfies (13). Therefore, assuming that $\mathcal{M}$ has a fixed complexity $\mathcal{C}(\mathcal{M}) = N > 0$, the planar optimal metric field $\mathcal{M}^{\mathcal{P}}_{opt}$ minimizing (14) is obtained through a calculus of variations described in [12, 24]. It comes

$$\mathcal{M}^{\mathcal{P}}_{opt} = C(N) \left(\det(\mathcal{Q})\right)^{\frac{-1}{p(k+1)+2}} \mathcal{Q}, \tag{15}$$

where $C(N)$ is a positive constant chosen so that $\mathcal{C}(\mathcal{M}_{opt}) = N$. Notice that, if $\mathcal{M}$ is replaced by $\mathcal{M}_{opt}$ in the equation (14), then we have

$$\mathcal{E}(\mathcal{M}_{opt}) = \mathcal{O}\left(\frac{1}{N^{\frac{k+1}{2}}} \right), \quad \text{when} \quad N \to +\infty. \tag{16}$$

3.4 Log-simplex method

In what follows, a few recalls about how the log-simplex method enables to accurately solve (13) will be given, but we refer to [13] for a more exhaustive overview. It relies on two principles. First, the inequation (13) is written in terms of the log matrix $\mathcal{L} = \log(\mathcal{Q})$. Then, by introducing approximated linear constraints on $\mathcal{L}$, the non-linear problem of finding the best metric $\mathcal{Q}$ satisfying (13) is reduced to a sequence of linear ones, which are solved by a classical simplex optimization algorithm. Since $d^{(k+1)}\xi(0,0)$ is a homogeneous polynomial of degree $k+1$, the problem actually reduces to find a metric $\mathcal{Q}$ whose unit ball $\mathcal{B}_{\mathcal{Q}} = \{(\bar{u},\bar{v}) : {}^t(\bar{u},\bar{v}) \, \mathcal{Q} \, (\bar{u},\bar{v}) < 1\}$ remains included into the iso-line of level 1 of $d^{(k+1)}\xi(0,0)$, as stated by the next lemma.

Lemma 3.1. *Let $P : \mathbb{R}^2 \to \mathbb{R}$ be a homogeneous polynomial of degree $k+1$ and $\mathcal{Q}$ be a 2D metric. Assume that, for $x \in \mathbb{R}^2$, the following assumption is satisfied:*

$$P(x) = 1 \quad \Longrightarrow \quad {}^t x \, \mathcal{Q} \, x \geq 1.$$

Then, for all $x \in \mathbb{R}^2$,

$$|P(x)| \leq \left({}^t x \, \mathcal{Q} \, x \right)^{\frac{k+1}{2}}.$$

Proof. Let $x \in \mathbb{R}^2 \setminus \{0\}$. Since P is homogeneous, $P(x) \neq 0$ and we can set $y = \dfrac{x}{|P(x)|^{\frac{1}{k+1}}}$. The degree of P being $k+1$, it comes $P(y) = 1$ and the assumption of the proposition gives ${}^t y \, \mathcal{Q} \, y \geq 1$. Equivalently, it writes

$$1 \leq \frac{{}^t x \, \mathcal{Q} \, x}{|P(x)|^{\frac{2}{k+1}}} \quad \text{and thus} \quad |P(x)| \leq \left({}^t x \, \mathcal{Q} \, x\right)^{\frac{k+1}{2}}.$$

The case $x = 0$ being obvious, it achieves the proof of this lemma. □

The log-simplex method enables to find a metric which satisfies the assumption of Lemma 3.1, with the largest possible unit ball $\mathcal{B}_{\mathcal{Q}}$, in terms of area (or volume in 3D). Since it has to be numerically implemented, only a finite set of points $\{x_1, ..., x_n\}$ such that $P(x_i) = 1$, $i = 1, ..., n$ is actually considered, and the problem reduces to the optimization problem below.

Problem 1. *Given a set of points $\{x_1, ..., x_n\}$ such that $|P(x_i)| = 1$ for all $i \in \{1, ..., n\}$, find a metric $\mathcal{Q}$ of $\mathbb{R}^2$ such that*

$$\left\{ \begin{array}{l} \det(\mathcal{Q}) \quad \textit{is minimal}, \\ 1 \leq {}^t x_i \, \mathcal{Q} \, x_i, \quad \textit{for all} \quad i \in \{1, ..., n\} \, . \end{array} \right. \tag{17}$$

Indeed, the area $|\mathcal{B}_{\mathcal{Q}}|$ of the unit-ball $\mathcal{B}_{\mathcal{Q}}$ only depends on the determinant of $\mathcal{Q}$, and is given by the formula

$$|\mathcal{B}_{\mathcal{Q}}| = \pi \det \left(\mathcal{Q}\right)^{-\frac{1}{2}}.$$

Solving the optimization problem 1 is actually not obvious, since the cost function is the determinant, which is not linear in $\mathcal{Q}$. In order to overcome this issue, the problem is translated in terms of the symmetric matrix $\mathcal{L} = \log(\mathcal{Q})$. Since $\det(\mathcal{Q}) = \exp(\mathrm{trace}(\mathcal{L}))$, the cost function becomes the trace of $\mathcal{L}$, which is linear in $\mathcal{L}$. Unfortunately, the linear constraints on $\mathcal{Q}$ become non-linear when expressed in $\mathcal{L}$. But, as shown in [12], if $x \in \mathbb{R}^2$ satisfies ${}^t x \, \mathcal{L} \, x \geq -|x|_2 \log(|x|_2)$, then is also satisfies ${}^t x \, \mathcal{Q} \, x \geq 1$, and the optimization problem 1 is then replaced by the following linear one.

Problem 2. *Given a set of points $\{x_1, ..., x_n\}$ such that $|P(x_i)| = 1$ for all $i \in \{1, ..., n\}$, find a symmetric matrix $\mathcal{L}$ of $\mathbb{R}^2$ such that*

$$\left\{ \begin{array}{l} \textit{trace}(\mathcal{L}) \quad \textit{is minimal}, \\ {}^t x_i \, \mathcal{L} \, x_i \geq -|x_i|_2 \log(|x_i|_2), \quad \textit{for all} \quad i \in \{1, ..., n\} \, . \end{array} \right. \tag{18}$$

The main advantage of considering Problem 2 instead of 1 comes from the fact that both the cost-function and the constraints on $\mathcal{L}$ are linear. Thus, having a first admissible symmetric matrix $\mathcal{L}_0$ (for instance the matrix $\mathcal{L}_0 = \alpha I_2$ with α a well chosen scalar), it is possible to solve this problem with usual linear programming (see [15] for more details). As a counterpart of such an approximation in the constraints, only solving the log-problem does not give satisfying results, in the sense that, though it is satisfied, the inequality (13) is in general quite far from being an equality. Finally, this last difficulty is dealt with an iterative process in which a sequence of symmetric matrices $(\mathcal{L}_k)_{k\in\mathbb{N}}$ is computed. In order to go from $\mathcal{L}_k$ to $\mathcal{L}_{k+1}$, it consists in stretching the 2D space with respect to $\mathcal{Q}_k = \exp(\mathcal{L}_k)$. As shown in [12], if $(\mathcal{L}_k)_{k\in\mathbb{N}}$ converges to a symmetric matrix $\mathcal{L}$, then complying the constraints of Problem 2 for $\mathcal{L}$ is equivalent to complying the constraints of Problem 1 for $\mathcal{Q} = \exp(\mathcal{L})$. The algorithm 1 describes this process. Usually, the points $\{x_1, ..., x_n\}$ are obtained from a set of points lying on the unit circle (or unit sphere in 3D), which are projected onto the iso-line of $P \circ Q^{-\frac{1}{2}}$. Notice that, when numerically implementing this method, other tricky aspects have to be take into account. For instance, it may happen that the process described by the algorithm 1 does not converge, resulting in a sequence of metrics $(\mathcal{Q}_k)_{k\in\mathbb{N}}$ whose eigenvalues go to zero. To overcome this issue, the initial polynomial P is actually modified, so that there no infinite branch into its level-set of level 1. As written earlier, the purpose of the present article is not to give a full description of the log-simplex method but to exhibit a way to use it in a surface mesh adaptation context. The theoretical justifications and the details needed for an effective numerical implementation can be found in [12].

```
input  : Homogeneous polynomial $P$

output: Metric $\mathcal{Q}$ solving the problem 1

$\mathcal{Q}_0 = I_2$
repeat
    Compute $\{x_1, ..., x_n\}$ such that $|P(\mathcal{Q}_j^{-\frac{1}{2}} x_i)| = 1$, for all $i \in \{1, ..., n\}$
    Get $\mathcal{L}_j$ through log-problem 2 with input $\{y_1, ..., y_n\}$
    Compute $\mathcal{Q}_{j+1} = \mathcal{Q}_j^{\frac{1}{2}} \exp(\mathcal{L}_j) \mathcal{Q}_j^{\frac{1}{2}}$
    Replace $\mathcal{Q}_j$ by $\mathcal{Q}_{j+1}$
until convergence;
```

Algorithm 1: Log-simplex algorithm

4 Numerical implementation

In this section, every aspect of the numerical implementation is discussed. More precisely, from a valid initial mesh $\mathcal{H}$ of the domain Σ, the purpose is to numerically obtain the optimal metric space $(\mathcal{M}(x))_{x\in\mathcal{H}}$ described above. Notice that it is now assumed that Σ is actually composed of several surfaces, connected together with geometrical lines, themselves ended by geometrical nodes. Though the adapted mesh is devoted to be curved, the initial mesh $\mathcal{H}$ is composed of linear elements, and curved at the very end of the process. It is also assumed that, for every vertex of $\mathcal{H}$ which lies on a geometrical curve or surface, the coordinates in the parametric space are known.

4.1 Local gap-function high-order construction

The first thing to do is the numerical computation of $d^{(k+1)}\xi(x_0)$ for each vertex x_0 of $\mathcal{H}$. Naturally, the first step comes with the construction of the tangent plane $\mathcal{P}(x_0)$, which is equivalent to the computation of the two spanning vectors $(\vec{u}, \vec{v})$ described in Section 3.1. When a vertex lies on a parameterized surface, the natural choice would be applying the identity (6) by calling the mapping σ. Unfortunately, this formula involves the first derivatives of σ which may vanish if the surface is not entirely smooth. Therefore, for robustness considerations, it is a way better idea to compute those vectors from the mesh itself. Indeed, for every vertex of the mesh, the normal vector $\vec{n}$ to the surface is then built from the averaging of the normals of the neighboring triangles. Let $\mathcal{T}(x_0) = \{T_1, ..., T_m\}$ be the triangles surrounding the vertex x_0 and $\{n_1, ..., n_m\}$ be their normals, oriented in the same direction. Then, the normal vector $\vec{n}$ to the surface is just the average vector

$$\vec{n} = \frac{1}{m}\sum_{i=1}^{m} n_i.$$

The two orthonormal vectors $(\vec{u}, \vec{v})$ are then computed as defining the orthogonal plan with respect to $\vec{n}$. One additional advantage of retrieving $\mathcal{P}(x_0)$ from the mesh comes from the fact that it is feasible even for vertices which lie on geometrical lines or points, without having to call any surface or line parameterisation. This will be notably used to compute the optimal metric for line vertices.

Once $\mathcal{P}(x_0)$ is set, a $k+1$ order polynomial representation of ξ is built. To this end, the neighbouring elements $\mathcal{T}(x_0)$ of x_0 are "curved" up to the order $k+1$, and then projected onto $\mathcal{P}(x_0)$. Let T be one triangle of $\mathcal{T}(x_0)$, whose corners are denoted $\{x_1, x_2, x_3\}$, one of them actually being x_0. Let Σ be the parameterised surface on which T lies, σ be a mapping of Σ and $(u_i, v_i), i = 1, ..., 3$ the parameters coordinates such that $\sigma(u_i, v_i) = x_i$, for all $i \in \{1, ..., 3\}$. In order to curve T, Lagrangian interpolation nodes are added to T, whose parameters coordinates are interpolated from the three corners ones. Since the number of nodes to define a complete triangle of order $k+1$ is $\frac{(k+1)(k+2)}{2}$, the idea only consists in completing the three corner points by setting a set of barycentric coordinates $\left\{(\alpha_i, \beta_i, \gamma_i) : \alpha_i + \beta_i + \gamma_i = 1, i = 1, ..., \frac{(k+1)(k+2)}{2}\right\}$, the three first ones being $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$, which correspond to x_1, x_2 and x_3 respectively. This way, the nodes parameters are given by

$$(u_i, v_i) = \alpha_i(u_1, v_1) + \beta_i(u_2, v_2) + \gamma_i(u_3, v_3), \quad \text{for all } i \in \{1, ..., \tfrac{(k+1)(k+2)}{2}\}.$$

For instance, for a second order triangle, which requires 6 nodes to be defined, a uniform set of barycentric coordinates would be

$$\begin{array}{ll} (\alpha_1, \beta_1, \gamma_1) = (1, 0, 0), & (\alpha_4, \beta_4, \gamma_4) = \left(\frac{1}{2}, \frac{1}{2}, 0\right), \\ (\alpha_2, \beta_2, \gamma_2) = (0, 1, 0), & (\alpha_5, \beta_5, \gamma_5) = \left(0, \frac{1}{2}, \frac{1}{2}\right), \\ (\alpha_3, \beta_3, \gamma_3) = (0, 0, 1), & (\alpha_6, \beta_6, \gamma_6) = \left(\frac{1}{2}, 0, \frac{1}{2}\right). \end{array}$$

Once all the parameters coordinates are known, the triangle T is curved by calling the mapping σ. This way, the coordinates of all the interpolation nodes are obtained by the formula $x_i = \sigma(u_i, v_i)$. Notice that there is no need to apply σ to the first three corners of T, since their parameters already correspond to their physical coordinates. Then, all these nodes are projected onto $\mathcal{P}(x_0)$, and their coordinates in $\mathcal{P}(x_0)$ are obtained by the projection identity

$$\begin{pmatrix} \bar{u}_i \\ \bar{v}_i \end{pmatrix} = \begin{pmatrix} \langle x_i - x_0, \vec{u} \rangle \\ \langle x_i - x_0, \vec{v} \rangle \end{pmatrix}, \quad \text{for all} \quad i \in \left\{1, ..., \frac{(k+1)(k+2)}{2}\right\}.$$

Likewise, the value of ξ at each $(\bar{u}_i, \bar{v}_i)$ is nothing more than the dot product $\xi(\bar{u}_i, \bar{v}_i) = \langle x_i - x_0, \vec{n} \rangle$. Finally, a polynomial approximate of ξ is built as being the only polynomial P_T^{k+1} of order $k + 1$ such that

$$P_T^{k+1}(\bar{u}_i, \bar{v}_i) = \xi(\bar{u}_i, \bar{v}_i), \quad \text{for all} \quad i \in \left\{1, ..., \frac{(k+1)(k+2)}{2}\right\}. \tag{19}$$

Actually, the identity (19) is a linear system on the $\frac{(k+1)(k+2)}{2}$ coefficients of P_T^{k+1}, which has a unique solution if the interpolation nodes are well located. In the present article, the choice of uniform Lagrangian interpolation nodes has been made, but other choices may be suitable as well. In the end, for each triangle T_i, $i = 1, ..., m$ of $\mathcal{T}(x_0)$, a polynomial $P_{T_i}^{k+1}$ is built. The final polynomial used to approximate ξ around x_0 is given by the average formula

$$P_{x_0}^{k+1} = \frac{1}{m} \sum_{i=1}^{m} P_{T_i}^{k+1}.$$

Line vertices special treatment When a vertex lies on a line, an additional correction is done, in order to take into account the curvature of the curve itself. Indeed, the surface metric as described above is not sufficient to ensure a good representation of the line, when performing mesh adaptation. For instance, a plane surface could be separated by a curved line onto it. Then, since the surface is flat, the obtained surface metric would be the largest possible, but would not be suitable to properly catch the line. The 1D process is very similar to what is done for surface vertices, but in one dimension of space. More precisely, the same alternative mapping as the one described in Section 3.1 is built. Let $\gamma : [a, b] \rightarrow \mathbb{R}^3$ being the mapping of a 3D curve Γ, with $a < b \in \mathbb{R}$. Then, around a regular point $x_0 = \gamma(u_0)$ are computed the tangent and normal unit vectors

$$\begin{aligned} \vec{u} &= \frac{\gamma^{'}(u_0)}{|\gamma^{'}(u_0)|_2}, \\ \vec{v} &= \frac{\gamma^{''}(u_0) - \left\langle \gamma^{''}(u_0), \vec{u} \right\rangle \vec{u}}{|\gamma^{''}(u_0) - \left\langle \gamma^{''}(u_0), \vec{u} \right\rangle \vec{u}|_2}. \end{aligned} \tag{20}$$

Very similarly to the surface case, the function ξ_σ which measure the gap between the curve described by γ and its tangent line at x_0 in the normal direction $\vec{v}$ is considered. Following the framework of Section 3, the local k interpolation error is governed by the $k + 1$ derivative of ξ_γ, but the 1D version is much simpler in the sense that the 1 dimensional optimal metric is just a constant which is proportional to $|\xi^{(k+1)}(x_0)|$. According to the 1 dimensional local error model, the 1D "metric" $|\xi^{(k+1)}(x_0)|^{\frac{p(k+1)}{p(k+1)+1}}$ is computed for vertices lying on geometric lines. In addition, the vertex is also given a metric corresponding to each surface surrounding it. The local metric at x_0 is finally obtained by intersecting all those metrics. The $k + 1$ derivative of ξ_γ is obtained by building a $k + 1$ polynomial approximation of ξ_γ, using the actual CAD. In practise, for robustness considerations, $\vec{u}$ is usually computed from the CAD and $\vec{v}$ from the mesh.

4.2 From planar to 3D metric

Once the optimal planar metric $\mathcal{M}^{\mathcal{P}}_{opt}$ is obtained through the identity (15) or by its 1D equivalent in case of vertices lying on geometric lines, it has to be extended to the 3 dimensional space. At this point, the metric $\mathcal{M}^{\mathcal{P}}_{opt}$ is flat in the normal direction $\vec{n}$ to the tangent plan to the surface Σ. Let (μ_1, μ_2) denotes the eigenvalues of $\mathcal{M}^{\mathcal{P}}_{opt}$ along the directions $\vec{u}$ and $\vec{v}$, then the third eigenvalue μ_3 along the normal direction $\vec{n}$ is set to $\mu_3 = \max(\mu_1, \mu_2)$ and the 3D optimal metric is given by

$$\mathcal{M}_{opt}(x_0) = R \begin{pmatrix} \mu_1 & 0 & 0 \\ 0 & \mu_2 & 0 \\ 0 & 0 & \max(\mu_1, \mu_2) \end{pmatrix} R^t, \tag{21}$$

where R is the rotation matrix whose columns are composed by the unit vectors $\vec{u}$, $\vec{v}$ and $\vec{n}$. This choice corresponds to taking the smallest length in the normal direction. It is motivated by the will to preserve the anisotropy of the 2D metric in the 3 dimensional space, particularly at the interface between two distinct surfaces.

When a vertex x_0 lies on a geometrical line Γ being the interface of several surfaces $\Sigma_1, ..., \Sigma_n$, $n \in \mathbb{N}$, the average surface matrix $\overline{\mathcal{M}}_{opt}(x_0) = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} \mathcal{M}^i_{opt}(x_0)$ is computed, where $\mathcal{M}^i_{opt}(x_0)$ is the optimal matrix obtained for the surface i. Besides, the 1D metric computed for Γ on the tangent line defined through the tangent vector $\vec{u}$ is completed in the orthogonal plane to $\vec{u}$ by the smallest possible eigenvalue μ_{min}, which corresponds to the largest possible length $h_{max} = \frac{1}{\sqrt{\mu_{min}}}$. This way, the 3D metric computed on the line Γ is given by

$$\mathcal{M}^{\Gamma}_{opt}(x_0) = R_\Gamma \begin{pmatrix} \mu & 0 & 0 \\ 0 & \mu_{min} & 0 \\ 0 & 0 & \mu_{min} \end{pmatrix} R^t_\Gamma, \tag{22}$$

where R_Γ is the rotation matrix whose columns are $\vec{u}$, $\vec{v}$ and $\vec{u} \wedge \vec{v}$. Finally, the optimal metric $\mathcal{M}_{opt}(x_0)$ is obtained by metric intersection $\mathcal{M}_{opt}(x_0) = \mathcal{M}^{\Gamma}_{opt}(x_0) \cap \overline{\mathcal{M}}_{opt}(x_0)$. In this article, the simultaneous reduction method is used for metric intersection, as in [4]. At the very end of the process, a metric smoothing is performed through a metric gradation, following the framework of [4].

4.3 Full mesh adaptation algorithm

Finally, the whole surface mesh adaptation schedule follows Algorithm 2 below. The most notable observation that can be done from this algorithm is the fact that, during the whole process, the mesh is kept linear. This enables to speed up the computation and reduce the memory cost during all the mesh adaptation part, when new triangles are introduced and deleted. Indeed, directly dealing with high-order triangles would imply the creation and destruction of a massive amount of useless interpolation nodes. When new vertices are introduced through edge splitting, their parameters coordinates are computed on the fly by interpolation from neighboring vertices, and the same mechanism is used when curving the mesh through the introduction of interpolation nodes, at the very end of the process. Actually, everything is made in order to avoid the projection of vertices onto the CAD, since these are very costly. Indeed, these projections are only used at the very first step of matching the initial mesh with the CAD, which is not always necessary, depending on the ability to keep the parameters coordinates when generating the initial mesh.

5 Numerical results

In this section are exhibited several application cases, on various geometric models. The initial meshes are generated by the mesh generator Gmsh [19]. The mesh-adaptation work,

```
input :
- Parameterized surface model
- Initial linear mesh $\mathcal{H}_0$
- Complexity $N > 0$
- Mesh elements degree $k$

output: Adapted curved mesh with complexity $N$

1. Match CAD and $\mathcal{H}_0$: recover surface or line parameters for each vertex
2. repeat
       for all $x_0 \in \mathcal{H}$ do
               - Locally approximate $\Sigma$ with polynomial of degree $k$ onto $\mathcal{P}(x_0)$
               - Apply 2D Log-simplex algorithm and get $\mathcal{M}^{\mathcal{P}}_{opt}(x_0)$
               - Extend $\mathcal{M}^{\mathcal{P}}_{opt}(x_0)$ to 3D local metric $\mathcal{M}_{opt}(x_0)$
       end
           - Normalize metric-field and get $\mathcal{M}_{opt}$ with complexity $N$
           - Adapt the mesh
   until convergence;
3. Upgrade mesh elements to degree $k$
```

Algorithm 2: Surface mesh adaptation algorithm

which includes the optimal metric-field computation, the mesh local modifications and the elements curving process is done with the open source library MAdLib [1] (for Mesh Adaptation Library). For this work, MAdLib has been linked with the CAD handling library Open Cascade [2] All the mesh pictures shown in this section have been generated by the visualization software

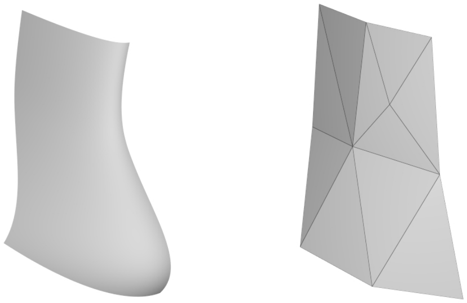

Figure 3: Surface and coarse initial linear mesh.

Vizir [27].

5.1 Academical validation case

In order to validate the good behavior of the mesh adaptation process, and to ensure that the error between the CAD model and the curved adapted is minimized, a simple model containing a single surface is considered (see Figure 3). On this case, a very coarse mesh is generated and the mesh adaptation procedure described above is iteratively performed, increasing the complexity of the mesh at each iteration. This is done for triangular elements, from degree 1 to 4, giving the adapted meshes shown by Figure 4. Though the differences in the adapted meshes between the different orders are not truly spectacular in this case, we still are able to see some variations in the orientations of the elements, or in the refined or coarsened zones.

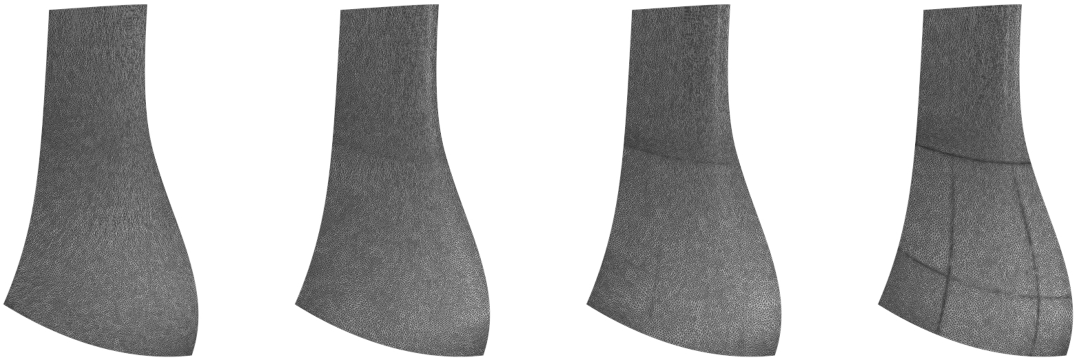

Figure 4: Adapted high-order meshes adapted to a surface. Left to right: meshes composed of $\mathbb{P}_1$, $\mathbb{P}_2$, $\mathbb{P}_3$ and $\mathbb{P}_4$ triangles. Each mesh is composed of around 33 000 triangles.

Using a classical fourth order Legendre quadrature, the interpolation error is then computed. It consists in computing the L^2-norm of the distance between the surface and the curved mesh. More precisely, for each triangle $\mathcal{T}$ of the mesh, let $s_{\mathcal{T}}$ be the mapping from a reference triangle T_{ref} to $\mathcal{T}$

$$\begin{array}{rccc} s_{\mathcal{T}}: & T_{ref} & \rightarrow & \mathcal{T} \\ & (u,v) & \mapsto & s_{\mathcal{T}}(u,v) \end{array} \tag{23}$$

Each component of $s_{\mathcal{T}}$ is a polynomial of arbitrary degree k, and the local interpolation error can be computed as the surface integral

$$\mathcal{E}(\mathcal{T}) = \int_{T_{ref}} |s_{\mathcal{T}}(u,v) - \Pi_{\Sigma}\left(s_{\mathcal{T}}(u,v)\right)|^2 \, |n_{\mathcal{T}}(u,v)|_2 \, dudv, \tag{24}$$

where Π_{Σ} is the orthogonal projection onto Σ and $n_{\mathcal{T}}(u,v) = \dfrac{\partial s_{\mathcal{T}}}{\partial_u}(u,v) \wedge \dfrac{\partial s_{\mathcal{T}}}{\partial_v}(u,v)$ is the normal vector to the triangle at (u,v). Finally, the global L^2 error for the mesh $\mathcal{H}$ is given by

$$E(\mathcal{H}) = \left(\sum_{\mathcal{T}\in\mathcal{H}} \mathcal{E}(\mathcal{T})\right)^{\frac{1}{2}} = \left(\sum_{\mathcal{T}\in\mathcal{H}} \int_{T_{ref}} |s_{\mathcal{T}}(u,v) - \Pi_{\Sigma}\left(s_{\mathcal{T}}(u,v)\right)|^2 \, |n_{\mathcal{T}}(u,v)|_2 \, dudv\right)^{\frac{1}{2}}. \tag{25}$$

Figure 5 shows the interpolation error (25) with respect to the number of degrees of freedom, for the adapted meshes and for uniformly refined meshes. First, the expected rate of convergence is recovered for every order, when the mesh is refined enough. More significantly, the comparison with uniform meshes shows an obvious improvement when considering adapted meshes. For the $\mathbb{P}_4$ case, for instance, there are almost 2 order of magnitude between the errors induced by the most refined uniform and adapted meshes. Furthermore, the error curve of the uniform $\mathbb{P}_4$ meshes does not even follow the theoretical rate of convergence.

5.2 Complex geometries

In this section are exhibited curved meshes complying to more complex geometries, emphasizing the differences of the mesh adaptation process, depending on the order of the triangles. In Figure 6 are shown adapted $\mathbb{P}_1$, $\mathbb{P}2$ and $\mathbb{P}_3$ meshes with respect to a rotor blade with around 150 000 triangles. Interestingly, the meshes differs in the way they are refined in the leading edge of the blade and at the foot of the blade. For instance, the $\mathbb{P}_1$ mesh is very anisotropic at very end of the leading edge of the blade, whereas the mesh is anisotropic a little further from this area. The $\mathbb{P}_3$ follows a third different configuration. At the foot of the blade, the $\mathbb{P}_1$ mesh is anisotropic, on the contrary to the other ones, for which the curvature of the elements fits well the surface without specific alignment.

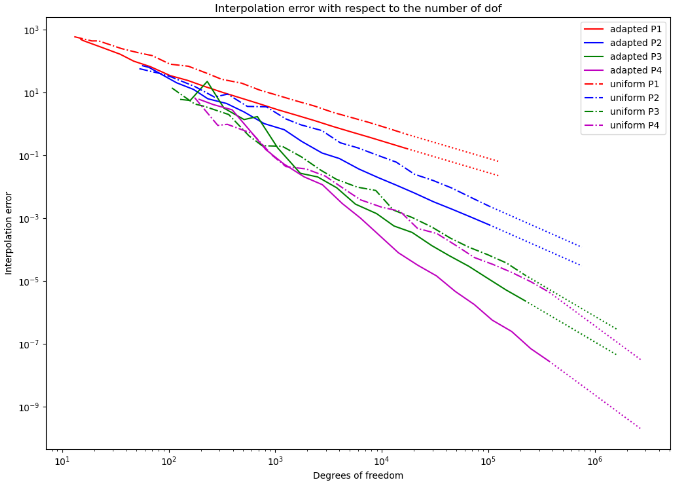

Figure 5: Interpolation error with respect to the number of degrees of freedom. Comparison with the error for uniform meshes. Plan lines: adapted meshes. Dashed lines: uniform meshes. Dot lines: expected theoretical rate of convergence given by (16).

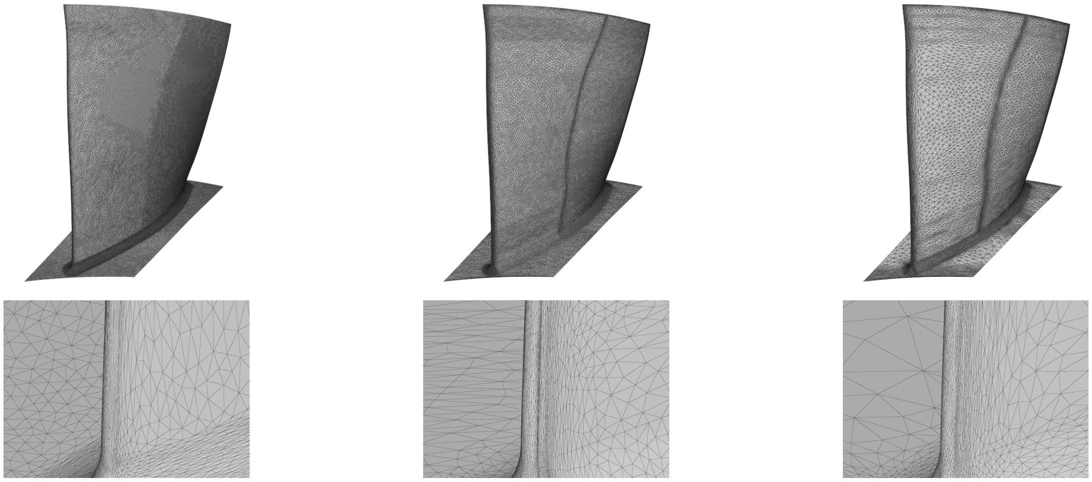

Figure 6: Adapted meshes with respect to a rotor 37 geometry and zooms on the leading edge. Left to right: $\mathbb{P}_1$, $\mathbb{P}_2$ and $\mathbb{P}_3$ meshes composed of around 150 000 triangles.

Figure 7 shows curved meshes adapted to a CPU fan (without the blades), from $\mathbb{P}_1$ to $\mathbb{P}_3$

elements, starting from an initial with 4 446 linear triangles. Two distinct mesh adaptation procedures have been performed, the first one aiming the reduction of the number of triangles, up to around 2 000 triangles and the second one increasing the number of triangles up to 300 000 triangles. Similarly to the rotor 37 geometry, we can see differences between the order of interpolations, especially on the cylinders contained in the geometry.

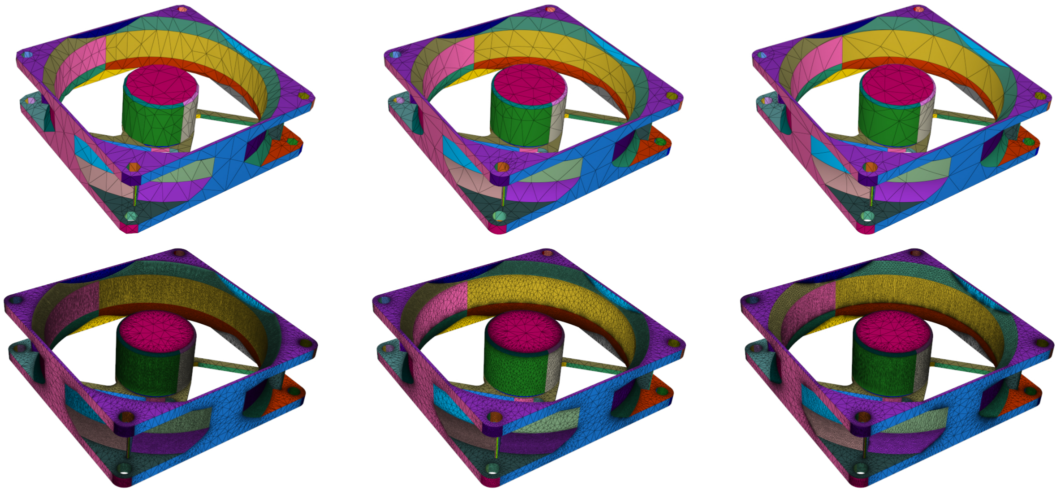

Figure 7: Adapted curved meshes with respect to a fan geometry, with very coarse and very refined meshes. Left to right: $\mathbb{P}_1$, $\mathbb{P}_2$ and $\mathbb{P}_3$ meshes. Each mesh of the top line is composed of around 2 000 triangles, each mesh of the bottom line is composed of around 300 000 triangles.

6 Conclusions and perspectives

A surface high-order mesh adaptation method has been proposed in this article, based on the log-simplex method, which enables to produce adapted high-order meshes with respect to complex geometries. This feature is one of the required steps towards the practical use of high-order finite elements methods, which still suffers from the lack of robust high-order mesh generation methods. In the continuation of this work, several improvements could be done, in order to efficiently couple this adaptation method with high-order finite elements methods computations. First of all, in the present paper, no control on the curved triangles qualities is considered, which is a key issue that has to be taken into account. Even though the mesh is adapted to the surface, minimizing the amount of low-quality elements, a high-order quality optimization process would be a significant improvement. Several approaches already exist to deal with this issue (see for instance [20, 35]), based on the optimization of the interpolation nodes locations. In addition, though this article only deals with the surface mesh, it is obvious that the case of the generation of high-order volume meshes also needs to be addressed. In this case, the minimum that could be expected would be the diffusion of the curvature of the surface elements to the volume elements, for instance through elasticity methods [3, 30, 38]. Another interesting approach, that has notably been investigated in [32, 33], lies in the generation of high-order volume elements by considering edges (or faces) following geodesic curves given by a metric-field. This way, the mesh does not only comply to the curvature induced by the boundary elements but may also minimize the interpolation error due to the projection of a solution onto a high-order finite elements space. In further works, coupling all these mesh adaptation techniques with the log-simplex high-order solution-based adaptation process introduced in [12, 13] would enable to address the problem of the efficiency of the high-order finite elements methods, by adapting an high-order mesh both to the geometric model and to the solution.

References

[1] Madlib `https://sites.uclouvain.be/madlib/`.

[2] Open cascade technology `www.opencascade.com`.

[3] R. Abgrall, C. Dobrzynski, and A. Froehly. A method for computing curved meshes via the linear elasticity analogy, application to fluid dynamics problems. *International Journal for Numerical Methods in Fluids*, 76(4):246–266, 2014.

[4] F. Alauzet. Size gradation control of anisotropic meshes. *Finite Elements in Analysis and Design*, 46(1):181–202, 2010. Mesh Generation - Applications and Adaptation.

[5] F. Alauzet, L. Frazza, and D. Papadogiannis. Periodic adjoints and anisotropic mesh adaptation in rotating frame for high-fidelity rans turbomachinery applications. *Journal of Computational Physics*, 450, 2022.

[6] R. Aubry, S. Dey, E.L. Mestreau, B.K. Karamete, and D. Gayman. A robust conforming nurbs tessellation for industrial applications based on a mesh generation approach. *Computer-Aided Design*, 63:26–38, 2015.

[7] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2d euler equations. *Journal of Computational Physics*, 138(2):251–285, 1997.

[8] M. J. Castro-Díaz, F. Hecht, B. Mohammadi, and O. Pironneau. Anisotropic unstructured mesh adaption for flow simulations. *International Journal for Numerical Methods in Fluids*, 25(4):475–491, 1997.

[9] A. Chakraborty, A. Rangarajan, and G. May. An anisotropic h-adaptive strategy for discontinuous petrov-galerkin schemes using a continuous mesh model. *Computers & Mathematics with Applications*, 106:1–17, 2022.

[10] P. G. Ciarlet. *The finite element method for elliptic problems*, volume 40 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 2002.

[11] H. L. Cougny and M.S. Shephard. Surface meshing using vertex insertion. In *Proceedings of the 5th International Meshing Roundtable*, pages 243–256, 1996.

[12] O. Coulaud and A. Loseille. Very high order anisotropic metric-based mesh adaptation in 3d. *Procedia Engineering*, 163:353–364, 2016.

[13] O. Coulaud, A. Loseille, and P. Schrooyen. Anisotropic mesh adaptation for high-order finite elements spaces with the log-simplex method. application to discontinuous galerkin methods. *Journal of Computational Physics*, 501:112774, 2024.

[14] T. Coupez. Génération de maillage et adaptation de maillage par optimisation locale. *Revue Européenne des Éléments Finis*, 9, 01 2000.

[15] G. B. Dantzig and M. N. Thapa. *Linear Programming 2: Theory and Extensions*. Springer-Verlag, 2003.

[16] C. Dobrzynski and P. Frey. Anisotropic delaunay mesh adaptation for unsteady simulations. In Rao V. Garimella, editor, *Proceedings of the 17th International Meshing Roundtable*, pages 177–194, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[17] V. Dolejší. Anisotropic mesh adaptation for finite volume and finite element methods on triangular meshes. *Computing and Visualization in Science*, 1:165–178, 1998.

[18] R. Feuillet, O. Coulaud, and A. Loseille. Anisotropic error estimate for high-order parametric surface mesh generation. *International Meshing Roundtable*, 28, 2019.

[19] C. Geuzaine and J. F. Remacle. 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):1309–1331, 2009.

[20] M.D. Green, K.S. Kirilov, M. Turner, J. Marcon, J. Eichstädt, E. Laughton, C.D. Cantwell, S.J. Sherwin, J. Peiró, and D. Moxey. Nekmesh: An open-source high-order mesh generation framework. *Computer Physics Communications*, 298:109089, 2024.

[21] K. Hillewaert. *Development of the discontinuous Galerkin method for high-resolution, large scale CFD and acoustics in industrial geometries.* PhD thesis, 02 2013.

[22] D. A. Kopriva and J. H. Kolias. A conservative staggered-grid chebyshev multidomain method for compressible flows. *Journal of Computational Physics*, 125(1):244–261, 1996.

[23] P. Laug. Some aspects of parametric surface meshing. *Finite Elements in Analysis and Design*, 46(1):216–226, 2010. Mesh Generation - Applications and Adaptation.

[24] A. Loseille. *Phd Thesis: Adaptation de maillage anisotrope 3D multi-échelles et ciblée à une fonctionnelle pour la mécanique des fluides : Application à la prédiction haute-fidélité du bang sonique.* PhD thesis, Université Pierre et Marie Curie - Paris VI, 2008.

[25] A. Loseille and F. Alauzet. Continuous mesh framework part I: well-posed continuous interpolation error. *SIAM J. Numer. Anal.*, 49(1):38–60, 2011.

[26] A. Loseille and F. Alauzet. Continuous mesh framework part II: validations and applications. *SIAM J. Numer. Anal.*, 49(1):61–86, 2011.

[27] A. Loseille and R. Feuillet. Vizir: High-order mesh and solution visualization using opengl 4.0 graphic pipeline. In Xiangmin Jiao and Jean-Christophe Weill, editors, *56th AIAA Aerospace Sciences Meeting, AIAA Scitech.*, 2008.

[28] A. Loseille and V. Menier. Serial and parallel mesh modification through a unique cavity-based primitive. In Josep Sarrate and Matthew Staten, editors, *Proceedings of the 22nd International Meshing Roundtable*, pages 541–558, Cham, 2014. Springer International Publishing.

[29] D. Marcum and F. Alauzet. Control of element shape and alignment for 3d solution adaptive mesh generation. *Computer-Aided Design*, 119:102750, 2020.

[30] D. Moxey, D. Ekelschot, Ü. Keskin, S. Sherwin, and J. Peiro. High-order curvilinear meshing using a thermo-elastic analogy. *Computer-Aided Design*, 72, 10 2015.

[31] Ajay Rangarajan, Aravind Balan, and Georg May. Mesh optimization for discontinuous galerkin methods using a continuous mesh model. *AIAA Journal*, 56(10):4060–4073, 2018.

[32] L. Rochery and A. Loseille. P2 Cavity Operator with Metric-Based Volume and Surface Curvature. In *29th International Meshing Roundtable (IMR)*, Virtual, France, June 2021.

[33] L. Rochery and A. Loseille. Fast high-order mesh correction for metric-based cavity remeshing and a posteriori curving of p2 tetrahedral meshes. *Computer-Aided Design*, 163:103575, 2023.

[34] D. Siqueira, J. B. Cavalcante-Neto, C. A. Vidal, and R. J. Silva. A hierarchical adaptive mesh generation strategy for parametric surfaces based on tree structures. In *2010 23rd SIBGRAPI Conference on Graphics, Patterns and Images*, pages 79–86, 2010.

[35] T. Toulorge, J. Lambrechts, and J. F. Remacle. Optimizing the geometrical accuracy of curvilinear meshes. *Journal of Computational Physics*, 310:361–380, 2016.

[36] Guglielmo Vivarelli, Ning Qin, and Shahrokh Shahpar. *Combined Hessian and Adjoint Error-Based Anisotropic Mesh Adaptation for Turbomachinery Flows.* 2017.

[37] D. Wang, O. Hassan, K. Morgan, and N. Weatherill. Eqsm: An efficient high quality surface grid generation method based on remeshing. *Computer Methods in Applied Mechanics and Engineering*, 195(41):5621–5633, 2006. John H. Argyris Memorial Issue. Part II.

[38] Z. Xie, R. Sevilla, O. Hassan, and K. Morgan. The generation of arbitrary order curved meshes for 3d finite element analysis. *Computational Mechanics*, 51:361–374, 01 2014.