OPTIMALLY CONVERGENT ISOPARAMETRIC P^2 MESH GENERATION

Arthur Bawin[1] André Garon[2] Jean-François Remacle[3]

[1] *Polytechnique Montréal, Montréal, QC, Canada and UCLouvain, Louvain-la-Neuve, Belgium arthur.bawin@uclouvain.be*
[2] *Polytechnique Montréal, Montréal, QC, Canada andre.garon@polymtl.ca*
[3] *UCLouvain, Louvain-la-Neuve, Belgium jean-francois.remacle@uclouvain.be*

ABSTRACT

This paper describes a framework for the generation of P^2 isoparametric meshes in two dimensions. It is an extension of existing metric-based methods for high-order straight-sided mesh adaptation to curved meshes. Starting with an interpolation error estimate for curved trajectories, we compute an optimal metric field using a generalization of the log-simplex algorithm for high-order metrics. A straight quasi-unit mesh is generated in a frontal approach, then the edges are curved to minimize their length in the chosen metric. Convergence studies are performed on simple test cases. In particular, optimal convergence rates (third order) are reached even when the edges curvature decreases at the same rate as the edge length.

Keywords: mesh generation, high-order mesh, interpolation error, anisotropic mesh, metric

1. INTRODUCTION

The generation of curvilinear meshes was initially intended to improve the approximation of the boundary geometry of the domains to be modeled by finite elements [1, 2]. Recently, a new trend has emerged: the use of curved meshes inside the domain to approximate the solution as well as possible [3, 4, 5, 6, 7]. This problem is the one of curvilinear mesh adaptation, where anisotropic but also curvilinear elements are allowed.

As for anisotropic mesh adaptation, the metric tensor plays a central role in curvilinear mesh adaptation. However, to the best of our knowledge, a solution-based metric tensor field tailored for curved elements is not yet available in the literature. In [7], the two- and three-dimensional metric is either induced from the curvature of the geometry, computed from solutions of PDE on straight-sided meshes, or a combination thereof. But the a priori error model uses straight line parameterizations to write local upper bounds on the interpolation error, and does not let the elements bend to follow the solution, yielding potentially shorter elements. In [4, 5], the target metric is chosen to align the mesh to specified curves or surfaces, but does not rely on the solution of a PDE.

The motivation to this paper is thus to describe the structure of the interpolation error if we allow to curve the mesh elements, by extending the framework proposed by Alauzet & Loseille [8, 9] to curvilinear meshes in two dimensions.

Another question remains open: how does the curvature of an element influence the interpolation error? Papers dating from the beginning of the finite element era have tried to give an answer to this question. In Ciarlet & Raviart [10], the authors show that, to preserve the optimal convergence of finite elements, the radius of curvature of the edges of the elements should decrease as h^2 if h is the size of the elements. This means that in order to preserve the optimal convergence, it is necessary to curve the elements relatively

less and less when the edges' length is decreased. Here, we show, using numerical test cases, that this assumption is too strong and that homothetically refined elements i.e. whose relative curvature remains constant allow for optimal convergence.

The paper is structured as follows. In Section 2, we derive an interpolation error estimate tailored for curved elements, which is the main contribution of this paper. From this anisotropic estimate, a metric tensor field is obtained, based on previous work on high-order straight-sided meshes [11, 12, 13]. In particular, the extension of the log-simplex algorithm proposed in [13] to non-homogeneous error polynomial is a new contribution. We then introduce the metric tensor induced by the graph of the target field u and use it to establish the principal directions of the mesh. In Section 3, the mesh generation algorithm is described. In particular, we adapt the advancing front method introduced in [6] to use both the error metric and the induced metric to specify the target sizes and anisotropic directions of the mesh elements. An extension of the edge curving method proposed in [6] is also presented. Section 4 presents an application of this mesh adaptation framework to two simple test cases.

2. INTERPOLATION ERROR MODEL AND METRIC TENSOR

The mesh generation methodology described in this paper is based on interpolation error. We follow the steps presented in similar work on higher order (≥ 2) straight-sided anisotropic mesh adaptation [11, 12, 13]. First, we build an a priori error estimate based on a Taylor expansion and higher-order derivatives of an unknown scalar field $u(x, y)$. As we consider quadratic edges, second and third order derivatives essentially determine the error estimator. Then, we seek the metric tensor field that best translates this estimate in the shape of the ideal elements. It is well-known that for elements with interpolation functions of order $k \geq 2$, the natural connection between the Hessian matrix of u and the metric tensor driving the mesh adaptation process no longer holds. Indeed, the metric tensor is represented by an $n \times n$ matrix, with n the space dimension, whereas an error estimate based on a Taylor expansion features tensors of high order derivatives. Rather, a quadratic form represented by a symmetric positive-definite matrix which is a tight upper bound on the error estimate is sought. Finally, the obtained metric field is scaled to obtain a target number of vertices in the final mesh and perform convergence studies.

2.1 Curve parameterizations

We start by defining the curves and parameterizations we consider in this paper. When writing an error estimate around a vertex $\mathbf{x}_0$, initial conditions such as the starting point, initial direction and curvature are prescribed, but the final point of the curve is unknown. Working with a parameter $s \geq 0$ (not necessarily the arclength) thus makes sense in this case. On the other hand, when curving the mesh edges, Section 3.3, both extremities are known and a parameterization in $t \in [0, 1]$ is used.

2.1.1 Parameterization of paths for error estimation

Let $u(x, y)$ represent a real-valued scalar field and $\mathbf{x}_0 = (x_0, y_0) \in \mathbb{R}^2$ be a point around which we define an anisotropic error estimate. To write an error estimator, we consider curves leaving $\mathbf{x}_0$ and whose curvature is obtained from the derivatives of u. Indeed, for a given unit direction $\mathbf{v}$, there is infinitely many curved paths leaving $\mathbf{x}_0$ with arbitrary initial curvature κ, each endowed with an amount of error. Two particular ways of leaving $\mathbf{x}_0$ are to follow the gradient and the level curve of u at $\mathbf{x}_0$, i.e. to consider curves $\mathcal{C}_1 \equiv \mathbf{r}_1(s), \mathcal{C}_2 \equiv \mathbf{r}_2(s)$ everywhere tangent to ∇u and its orthogonal direction $\nabla u^{\perp}$. As we aim to generate quadratic edges, we propose to approach those two curves by their quadratic Taylor expansion of the form:

$$\mathbf{x}_i(s) = \mathbf{r}_i(0) + s\mathbf{v} + \frac{s^2}{2}\kappa\mathbf{v}^{\perp} + o(s^3), \quad i = 1, 2. \quad (1)$$

This expression is the projection of the local canonical form of the curve $\mathbf{r}_i(s)$ into its osculating plane, see e.g. Prop. 2.6 of [14]. Imposing the match up to order 2 between the curves $\mathcal{C}_i$ and their Taylor expansion yields the following tangent vectors and curvatures (see e.g. [6] for the full computation):

$$\begin{aligned} \mathbf{v}_1 &= \nabla u(\mathbf{x}_0)/\|\nabla u(\mathbf{x}_0)\|, \\ \mathbf{v}_2 &= \nabla u^{\perp}(\mathbf{x}_0)/\|\nabla u^{\perp}(\mathbf{x}_0)\|, \\ \kappa_1 &= \frac{\mathbf{v}_2^T H(\mathbf{x}_0)\mathbf{v}_1}{\|\nabla u(\mathbf{x}_0)\|}, \\ \kappa_2 &= -\frac{\mathbf{v}_2^T H(\mathbf{x}_0)\mathbf{v}_2}{\nabla u(\mathbf{x}_0)\cdot\mathbf{v}_1}, \end{aligned} \quad (2)$$

with H the Hessian matrix of u. It is worth noting that this approximation is *not* an arclength parameterization. Indeed, we have:

$$\|\mathbf{x}_i'(s)\| = \sqrt{1 + \kappa_i^2 s^2}, \quad (3)$$

which is unit only at $s = 0$. Hence the norm of the second derivative,

$$\|\mathbf{x}_i''(s)\| = \kappa_i, \quad (4)$$

represents the curvature of $\mathbf{x}_i(s)$ only at $s = 0$. In particular, approximation (1) does not have constant curvature. Computing the exact curvature of the parabola would require an arclength parameterization of (1), but the computation of

$$\tilde{s} \triangleq \int_0^s \|\mathbf{x}_i'(\sigma)\| \, d\sigma \tag{5}$$

yields

$$\tilde{s} = \frac{\sinh^{-1}(\kappa_i(0)s) + ks\sqrt{1+\kappa_i(0)^2 s^2}}{2\kappa_i(0)}, \tag{6}$$

which cannot be solved for $s(\tilde{s})$.

2.1.2 Parameterization of parabolic edges

As we use isoparametric P^2 finite elements, the mesh is made of 6-nodes triangles with parabolic edges and the reference-to-physical transformation $\mathbf{x}(\boldsymbol{\xi})$ is given by:

$$\mathbf{x}(\boldsymbol{\xi}) = \sum_{i=1}^{6} \mathbf{X}_i \phi_i(\boldsymbol{\xi}), \tag{7}$$

where $\mathbf{X}_i$ is the position of the vertices in the physical space and $\phi_i(\boldsymbol{\xi})$ denote the quadratic Lagrange basis functions (Fig. 1).

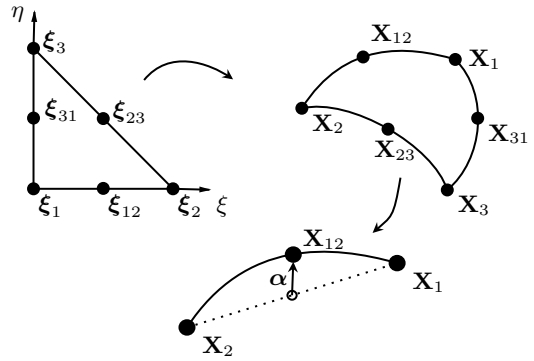


Figure 1: Reference-to-physical transformation and displacement vector.

The Lagrange functions satisfy $\phi_i(\Xi_j) = \delta_{ij}$, with Ξ_j the reference vertices and δ_{ij} the Kronecker delta. On each edge, the middle vertex (or midpoint) is defined by the displacement vector $\boldsymbol{\alpha} \in \mathbb{R}^2$:

$$\boldsymbol{\alpha} = \mathbf{X}_{12} - \frac{\mathbf{X}_1 + \mathbf{X}_2}{2}, \tag{8}$$

so that the edge parameterization writes for $t \in [0,1]$:

$$\mathbf{x}(t) = \mathbf{X}_1 + \boldsymbol{\gamma} t + L(t)\boldsymbol{\alpha}, \tag{9}$$

with $\boldsymbol{\gamma} \triangleq \mathbf{X}_2 - \mathbf{X}_1$ and $L(t) \triangleq 4t(1-t)$.

2.2 Interpolation error estimate

We now write an estimate of the interpolation error along curves $\mathbf{x}(s,\theta),\ \theta \in [0, 2\pi]$. These curves are obtained by sweeping the unit directions around $\mathbf{x}_0$ and interpolating the curves $\mathbf{x}_1(s)$ and $\mathbf{x}_2(s)$ accordingly:

$$\begin{aligned} \mathbf{x}(s,\theta) &= \mathbf{x}_0 + s\mathbf{v}(\theta) + \frac{s^2}{2}\kappa(\theta)\mathbf{v}^\perp(\theta), \\ \mathbf{v}(\theta) &= \mathbf{v}_1 \cos\theta + \mathbf{v}_2 \sin\theta, \\ \mathbf{v}^\perp(\theta) &= \mathbf{v}_1^\perp \cos\theta + \mathbf{v}_2^\perp \sin\theta, \\ \kappa(\theta) &= \kappa_1 \cos^2\theta + \kappa_2 \sin^2\theta. \end{aligned} \tag{10}$$

In the following, we drop the θ and write $\mathbf{x}(s)$ to ease the notation. We want to find an expression for the interpolation error:

$$e(s) = u(\mathbf{x}(s)) - \Pi^2 u(\mathbf{x}(s)), \tag{11}$$

with $\Pi^2 u$ is the interpolant of u using quadratic Lagrange functions. We consider Taylor's integral remainder for an interpolation of degree k written for a parameterization $\mathbf{x}(s)$, integrating until an arbitrary length $s = \bar{s}$:

$$e(\bar{s}) = \frac{1}{k}\int_0^{\bar{s}} (\bar{s}-s)^k D^{(k+1)} u(\mathbf{x}(s)) \, ds. \tag{12}$$

We examine the case $k = 2$ of quadratic interpolation. Using the chain rule, the third derivative of the composition $(u \circ \mathbf{x})(s)$ is given by:

$$D^{(3)}u = C_{ijk}(\mathbf{x}(s))\dot{\mathbf{x}}_i\dot{\mathbf{x}}_j\dot{\mathbf{x}}_k + 3H_{ij}(\mathbf{x}(s))\dot{\mathbf{x}}_i\ddot{\mathbf{x}}_j, \tag{13}$$

with $H_{ij} = \partial^2 u/\partial x_i \partial x_j$, $C_{ijk} = \partial^3 u/\partial x_i \partial x_j \partial x_k$ and Einstein's summation convention on repeated indices. The term in gradient of u is absent since the third derivative of the parameterization $\dddot{\mathbf{x}}$ vanishes.

Inserting this in the error estimate, we write:

$$\begin{aligned} e(\bar{s}) &= \frac{1}{2}\int_0^{\bar{s}} (\bar{s}-s)^2 D^{(3)} u(\mathbf{x}(s)) \, ds \\ &= \frac{1}{2}\int_0^{\bar{s}} (\bar{s}-s)^2 \Big(C_{ijk}(\mathbf{x}(s))(\mathbf{v}+\kappa\mathbf{v}^\perp s)_{i,j,k} \\ &\qquad + 3H_{ij}(\mathbf{x}(s))(\mathbf{v}+\kappa\mathbf{v}^\perp s)_i(\kappa\mathbf{v}^\perp)_j \Big) \, ds \\ &= \frac{C_{ijk}(\mathbf{x}_0)}{2}\int_0^{\bar{s}} (\bar{s}-s)^2 (\mathbf{v}+\kappa\mathbf{v}^\perp s)_{i,j,k} \, ds \\ &+ \frac{3H_{ij}(\mathbf{x}_0)}{2}\int_0^{\bar{s}} (\bar{s}-s)^2 (\mathbf{v}+\kappa\mathbf{v}^\perp s)_i(\kappa\mathbf{v}^\perp)_j \, ds \\ &\triangleq c_1\bar{s}^6 + c_2\bar{s}^5 + c_3\bar{s}^4 + c_4\bar{s}^3. \end{aligned} \tag{14}$$

We have neglected the higher order derivatives (> 3) and approximated $H_{ij}(\mathbf{x}(s))$ and $C_{ijk}(\mathbf{x}(s))$ by their value at $\mathbf{x}_0$ and took them outside of the integral. The result is a non-homogeneous polynomial of degree 6 in $\bar{s}$: the explicit form of the coefficients $c_i = c_i(\theta)$ is

given in the appendix. For a linear (i.e. straight) parameterization $\mathbf{x}(s) = \mathbf{x}_0 + \mathbf{v}s$, $\ddot{\mathbf{x}} = 0$ and $\dot{\mathbf{x}} = \mathbf{v}$ and e reduces to a homogeneous polynomial of order 3. Indeed, setting $\kappa = 0$, one has:

$$e(\bar{s}) = \frac{\bar{s}^3}{6} C_{ijk}(\mathbf{x}_0)\mathbf{v}_i\mathbf{v}_j\mathbf{v}_k \tag{15}$$

or

$$\frac{1}{6}\Bigg(\underbrace{C_{111}}_{a}(\bar{s}^3\mathbf{v}_1^3) + \underbrace{(C_{112}+C_{121}+C_{211})}_{b}(\bar{s}^3\mathbf{v}_1^2\mathbf{v}_2) + \underbrace{(C_{122}+C_{212}+C_{221})}_{c}(\bar{s}^3\mathbf{v}_1\mathbf{v}_2^2) + \underbrace{C_{222}}_{d}(\bar{s}^3\mathbf{v}_2^3)\Bigg).$$

Defining the endpoints $\bar{x} \triangleq \bar{s}\mathbf{v}_1$ and $\bar{y} \triangleq \bar{s}\mathbf{v}_2$, we write

$$e(\bar{x},\bar{y}) = \frac{1}{6}\left(a\bar{x}^3 + b\bar{x}^2\bar{y} + c\bar{x}\bar{y}^2 + d\bar{y}^3 \right) \tag{16}$$

which is the homogeneous error polynomial used for high-order straight-sided mesh adaptation in [11, 12, 13]. It is worth pointing out that for a curved parameterization, we cannot write the error estimate as a polynomial in $(\bar{x},\bar{y})$, since the path used to travel from (x_0,y_0) to $(\bar{x},\bar{y})$ changes the total interpolation error, contrary to straight-sided parameterization where only the endpoint matters.

2.3 Optimal metric

Following the approach of [11, 13], we now wish to find the quadratic form, represented by a matrix $\mathcal{Q}$, that is the best upper bound for this error polynomial. More precisely, we seek the symmetric positive-definite matrix $\mathcal{Q}$ such that

$$d_e(\mathbf{x},\mathbf{x}_0) \le d_{\mathcal{Q}}(\mathbf{x},\mathbf{x}_0), \qquad \forall\, \mathbf{x}. \tag{17}$$

In this expression, $d_{\mathcal{Q}}(.,.)$ is the Riemannian distance induced by the metric tensor associated to $\mathcal{Q}$ and $d_e(.,.)$ is a distance function induced by the error polynomial. These distance functions are a normalized way to compare $\mathcal{Q}$ and e, see e.g. Section 3.2.2 of [11]. The distance $d_{\mathcal{Q}}(.,.)$ is defined by the infimum of the metric-weighted length taken from all the (regular) curves joining two points $\mathbf{p}$ and $\mathbf{q}$:

$$d_{\mathcal{Q}}(\mathbf{p},\mathbf{q}) = \inf \text{length}(c), \tag{18}$$

where $c : [a,b] \to \mathbb{R}^2$ is a differentiable piecewise C^1 curve with $c(a) = \mathbf{p}$ and $c(b) = \mathbf{q}$ and with

$$\text{length}(c) = \int_a^b \sqrt{(c'(t),c'(t))_{\mathcal{Q}}}\, dt, \tag{19}$$

where $(\mathbf{u},\mathbf{v})_{\mathcal{Q}} = \mathbf{u}^T\mathcal{Q}\mathbf{v}$ is the dot product with respect to $\mathcal{Q}$. While computing the metric tensor at $\mathbf{x}_0$, we place ourselves in the tangent plane to $\mathbb{R}^2$ at $\mathbf{x}_0$ and hence consider a constant (unknown) metric $\mathcal{Q}$. This way, the geodesics of $\mathcal{Q}$ are straight lines and the infimum is obtained by looking only at $\mathbf{x} - \mathbf{x}_0$. This will obviously not be the case when we generate curved edges later on, since variations of the metric will determine the edges' curvature. Thus, we can still write

$$d_{\mathcal{Q}}(\mathbf{x},\mathbf{x}_0) = \sqrt{(\mathbf{x}-\mathbf{x}_0)^T\mathcal{Q}(\mathbf{x}-\mathbf{x}_0)} \tag{20}$$

as in the straight-sided case.

The error-based distance function is defined by:

$$d_e(\mathbf{x},\mathbf{x}_0) = |e(\bar{s}(\mathbf{x}))|^{\frac{1}{k+1}}. \tag{21}$$

We thus seek, in a frame centered at $\mathbf{x}_0$, the SPD matrix $\mathcal{Q}$ such that:

$$(\mathbf{x}(\bar{s})^T\mathcal{Q}\mathbf{x}(\bar{s}))^{\frac{k+1}{2}} \ge |e(\bar{s})|, \quad \forall\, \theta \in [0,2\pi],\ \bar{s} > 0 \tag{22}$$

and such that its unit ball has the maximum area, in order to minimize the interpolation error for a given number of mesh vertices, i.e. we seek $\mathcal{Q}$ with minimum determinant. Minimizing $\det\mathcal{Q}$ subject to the constraints (22) is impractical though, since it requires an expensive discretization of both θ and $\bar{s}$ to evaluate the constraints. For a linear parameterization $\mathbf{x}(s)$, the error polynomial is homogeneous and a scaling argument [11, 13] shows that it is sufficient to satisfy (22) on the level curve 1 of the error polynomial. The parameter $\bar{s}$ can then be obtained by $\bar{s} = e^{-1}(1)$, so the resolution requires a discretization of θ only. Here, the error polynomial (14) is non-homogeneous in $\bar{s}$ and the scaling argument does not hold anymore. As $\mathbf{v}$ and $\mathbf{v}^\perp$ are unit vectors, we can however write the following upper bound:

$$|e(\bar{s})| \le \bar{s}^3(1+|\kappa|\bar{s}) \ \times \left(\frac{1}{6}\sum_{i,j,k}|C_{ijk}(\mathbf{x}_0)| + \frac{|\kappa|}{2}\sum_{i,j}|H_{ij}(\mathbf{x}_0)| \right), \tag{23}$$

whose limit for small curvature κ is an homogeneous polynomial. We could thus use this bound in the definition of the error-based distance, which would justify the scaling argument for regions of low curvature. Using this bound might be too conservative, so we make the choice of discretizing only the level curve 1 of the non-homogeneous error polynomial (14) all the same. The impact of this trade-off between computational cost, practicality and accuracy is however hard to quantify, and better solutions might be possible.

At each vertex of the background mesh, the metric tensor $\mathcal{Q}$ is thus found by solving the optimization problem:

$$\begin{aligned} &\min_{a,b,c\,\in\,\mathbb{R}} \det\mathcal{Q} \\ &\mathbf{x}_i^T\mathcal{Q}\mathbf{x}_i \ge 1 \qquad \text{for } i = 1,...,n, \end{aligned} \tag{24}$$

where a, b, c are the coefficients of $\mathcal{Q}$ and the $\mathbf{x}_i$ are points lying on the level curve 1 of $e(\bar{s})$. As pointed out in [13], the problem is ill-posed since one can always fit an ellipse between constraint points whose determinant goes to 0. To solve this, we use the log-simplex algorithm proposed in [13] which consists of (i) solving the optimization problem for $\mathcal{L} = \log \mathcal{Q} = R \log(\Lambda) R^T$, the matrix logarithm of $\mathcal{Q} = R \Lambda R^T$, and for modified constraints, and (ii) apply iteratively the transformation $\tilde{\mathbf{x}} = \mathcal{Q}^{\frac{1}{2}} \mathbf{x}$ to converge to the initial constraints. We briefly recall the method: Starting with the identity matrix $\mathcal{Q}_0 = \mathcal{I}$, we solve iteratively

$$\begin{aligned} &\min_{a',b',c' \in \mathbb{R}} \text{trace } \mathcal{L}_j \\ &(\mathbf{y}_i^T)_j \mathcal{L}(\mathbf{y}_i)_j \geq -\|(\mathbf{y}_i)_j\|^2 \log(\|(\mathbf{y}_i)_j\|^2) \end{aligned} \tag{25}$$

for the $i = 1, ..., n$ constraint points. At iteration j, the metric $\mathcal{Q}_{j+1} = \mathcal{Q}_j^{\frac{1}{2}} \mathcal{L}_j \mathcal{Q}_j^{\frac{1}{2}}$ is recovered and the constraint points are updated using the transformation $(\mathbf{y}_i)_{j+1} = \mathcal{Q}_{j+1}^{\frac{1}{2}} \mathbf{x}_i$. Sweeping the angles $\theta_i \in [0, 2\pi]$ around the vertex $\mathbf{x}_0$, we write:

$$e_i(\bar{s}) \triangleq e(\bar{s}, \theta_i) = 1 \quad \rightarrow \quad \bar{s} = e_i^{-1}(1), \tag{26}$$

so that the points $\mathbf{x}_i = \mathbf{x}(e_i^{-1}(1))$ lie on the level curve 1 of the error polynomial. In [13], the error polynomial is homogeneous and finding the points $\mathbf{x}_i$ on the level curve 1 is trivial. Here, the error function is a non-homogeneous polynomial of degree 6, so unfortunately there is no closed-form solution available to solve $e_i(\bar{s}) = \pm 1$ and we must rely on a numerical root-finding algorithm to find the smallest positive real root, as $\bar{s}$ is a strictly positive length.

The convergence theorem provided in [13] still holds even for a non-homogeneous function. Indeed, if the sequence $(\mathcal{Q})_j$ converges to $\mathcal{Q}$ as $j \rightarrow \infty$, then (i) the objective function converges to a minimum ($\mathcal{L}$ converges to 0, the proof is unchanged) and (ii) the log-constraints converge to the initial set of constraints $\mathbf{x}_i^T \mathcal{Q} \mathbf{x}_i \geq 1$.

Without relying on the homogeneous character of e, the proof goes as follows: the transformation $(\mathbf{y}_i)_j = \mathcal{Q}_j^{\frac{1}{2}} \mathbf{x}(e_i^{-1}(1)) = \mathcal{Q}_j^{\frac{1}{2}} \mathbf{x}_i$ converges to $\mathbf{y}_i = \mathcal{Q}^{\frac{1}{2}} \mathbf{x}_i$ as $j \rightarrow \infty$. Since $\mathcal{L} \rightarrow 0$, constraint (25) converges to

$$\begin{aligned} 0 \geq &-\|\mathbf{y}_i\|^2 \log(\|\mathbf{y}_i\|^2) \\ &\iff 0 \leq \log(\|\mathbf{y}_i\|^2) \\ &\iff 1 \leq \|\mathbf{y}_i\|^2 \\ &\iff 1 \leq \|Q^{\frac{1}{2}} \mathbf{x}_i\|^2 = \mathbf{x}_i^T Q^{\frac{1}{2}} Q^{\frac{1}{2}} \mathbf{x}_i = \mathbf{x}_i^T \mathcal{Q} \mathbf{x}_i, \end{aligned}$$

which concludes the proof.

Solving the optimization problem at each vertex of the background mesh yields the metric field $\mathcal{Q}(\mathbf{x})$. Following the continuous mesh theory presented in [8, 15, 16], the metrics are then scaled to obtain roughly N vertices in the final mesh:

$$\mathcal{M}^e(\mathbf{x}) = C \left(\det \mathcal{Q}(\mathbf{x})\right)^{\frac{-1}{p(k+1)+n}} Q(\mathbf{x}), \tag{27}$$

with

$$C = N^{\frac{2}{n}} \left(\int_\Omega \left(\det \mathcal{Q}(\mathbf{x})\right)^{\frac{p(k+1)}{2(p(k+1)+n)}} dx \right), \tag{28}$$

with p translates in which L^p norm the interpolation error should be minimized, $k = 2$ is the polynomial degree of the interpolation and $n = 2$ is the space dimension. In the following, we set $p = 2$.

3. MESH GENERATION

With the metric field $\mathcal{M}^e(\mathbf{x})$ at hand, we now generate a mesh of P^2 triangles. We follow the unit mesh approach [17] and aim at generating edges with metric-weighted length in $[1/\sqrt{2}, \sqrt{2}]$, to ensure some form of error equidistribution over mesh elements. In our approach, the straight mesh is created using an advancing front of vertices, and is then curved one edge at a time. This is done in four main steps:

1. generate vertices at unit distance from one another and connect them;
2. curve the straight edges by moving the midpoint to minimize metric-weighted length;
3. make the curved mesh valid;
4. perform quality-enhancing topological operations (edge swaps).

3.1 Principal directions of the mesh

In straight-sided anisotropic mesh generation, orientation of the elements is generally not controlled, since all triangles inscribed in the unit ball of the local metric are part of an equivalence class [9]. To generate curved elements however, working in the tangent space is not sufficient, and we should account for the variation of the metric and thus follow metric-imposed directions, such as the orthogonal directions given by the eigenvectors of the metric field, similarly to [18]. Here, instead of taking the eigenvectors of the metric field $\mathcal{M}^e(\mathbf{x})$, we introduce the induced metric $\mathcal{M}^I(\mathbf{x})$, defined on the graph of $u(x, y)$. The graph G of u, denoted by $\mathbf{p} = (x, y, u(x, y))$, is a surface in $\mathbb{R}^3$. The restriction of the euclidian metric,

$$ds^2 = dx^2 + dy^2 + dz^2, \tag{29}$$

on the surface associated to the implicit function $f = z - u = 0$ yields the induced metric (also known as the

first fundamental form):

$$ds^2 = \frac{f_{,x}^2 + f_{,z}^2}{f_{,z}^2}\,dx^2 + \frac{2f_{,x}f_{,y}}{f_{,z}^2}\,dxdy + \frac{f_{,y}^2 + f_{,z}^2}{f_{,z}^2}\,dy^2, \tag{30}$$

with the notation $f_{,x} = \partial f/\partial x$. As $f_{,z} = 1$, $f_{,x} = -u_{,x}$ and $f_{,y} = -u_{,y}$, this writes:

$$ds^2 = (1+u_{,x}^2)\,dx^2 + 2u_{,x}u_{,y}\,dxdy + (1+u_{,y}^2)\,dy^2. \tag{31}$$

The matrix associated to this metric is:

$$\mathcal{M}^I = \begin{pmatrix} 1+u_{,x}^2 & u_{,x}u_{,y} \\ u_{,x}u_{,y} & 1+u_{,y}^2 \end{pmatrix}, \tag{32}$$

whose eigenvectors of the induced metric are

$$\mathbf{v}_1 = \left(\frac{u_{,x}}{u_{,y}}, 1\right), \qquad \mathbf{v}_2 = \left(\frac{-u_{,y}}{u_{,x}}, 1\right). \tag{33}$$

They are aligned respectively with $(u_{,x}, u_{,y})$ and $(-u_{,y}, u_{,x})$, the direction of the gradient and of the level curve of u at $\mathbf{p}$. Hence, the curves obtained by integrating along $\mathbf{v}_1$ and $\mathbf{v}_2$ are approximations of the gradient curves and the level curves of u.

There are thus two metric fields of interest: the metric obtained from the interpolation error analysis $\mathcal{M}^e(\mathbf{x})$ (the exponent e for *error* was added for emphasis) and the induced metric $\mathcal{M}^I(\mathbf{x})$ obtained from the geometry of the graph of u. Contrary to the error metric, the induced metric is intrinsic to the solution u and does not involved the interpolation scheme. For each metric field, two types of curves are of particular interest:

- the geodesics are locally distance-minimizing curves. They are defined as parameterized curves $g(t)$ with zero acceleration everywhere on the curve, that is, with $\nabla_{g'} g' = 0$, where $\nabla_{g'}$ denotes the covariant derivative in the direction $g'(t)$ [14]. For a given coordinate system, the components form of this relation writes:

$$\frac{d^2 g^i}{dt^2} + \Gamma^i{}_{jk} \frac{dg^j}{dt}\frac{dg^k}{dt} = 0, \tag{34}$$

  which is as second order ODE in $g(t)$ and where $\Gamma^i{}_{jk}$ are the Christoffel symbols of the second kind, defined as:

$$\Gamma^i{}_{jk} = \frac{1}{2}\mathcal{M}^{-1}_{im}\left(\mathcal{M}_{mj,k} + \mathcal{M}_{mk,j} - \mathcal{M}_{jk,m}\right). \tag{35}$$

  Geodesics can be obtained by numerically integrating (34) using e.g. a 4-th order explicit Runge-Kutta scheme, along with two initial conditions:

$$g(0) = \mathbf{x}_0, \qquad g'(0) = \mathbf{v}, \tag{36}$$

  with $\mathbf{v}$ a unit direction.

- the integral curves tangent to either $\mathbf{v}_1$ or $\mathbf{v}_2$, the eigenvectors of the matrix associated to the metric tensor.

For each of the two metric fields $\mathcal{M}^I$ and $\mathcal{M}^e$, one can make the following observations:

- The geodesics of the induced metric $\mathcal{M}^I$ are the projection on the $xy-$plane of the geodesics on the graph of u. While they minimize the euclidian distance on the graph, they do not have obvious properties in terms of error minimization, i.e. minimizing the error-weighted distance.

- The integral curves of the eigenvectors of $\mathcal{M}^I$ are approximations of the gradient and level curves of u. Let $c(\xi) : [-1, 1] \to \mathbb{R}^2$ be a (perfectly represented) piece of a level curve, such that $u(c(\xi)) = C$. The Lagrange interpolate of u on c of degree k with n_k basis functions writes:

$$\begin{aligned} \Pi^k u &= \sum_{i=1}^{n_k} \phi_i(\xi) u(c(\Xi_i)) \\ &= C\left(\sum_{i=1}^{n_k} \phi_i(\xi)\right) \\ &= C \end{aligned} \tag{37}$$

  since the basis functions sum to 1, with Ξ_i the Lagrange nodes. Thus, the pointwise interpolation error $u - \Pi^k u$ is zero on a level curve.

- The geodesics of the error metric $\mathcal{M}^e$ minimize the error-weighted distance and are thus good candidates for curves on which generate the vertices.

- The integral curves of the eigenvectors of $\mathcal{M}^e$ follow the directions of extreme error. They have been investigated e.g. in [18].

The integral curves of both metrics, as well as the geodesics of $\mathcal{M}^e$, exhibit valuable properties in terms of error minimization. It is however not clear to us what is the link between these curves, if there is any. From a practical point of view, integrating along the geodesics of the error metric has not shown to be very robust, mostly because small perturbations in the metric field and its derivatives yield quite different geodesics. In this work, we chose to integrate along the eigenvectors of the induced metric. Integrating along the eigenvectors of $\mathcal{M}^e$ are currently also being investigated.

3.2 Vertices generation and triangulation

We start by generating an anisotropic straight-sided mesh with respect to $\mathcal{M}^e(\mathbf{x})$ using `mmg2d` [19], discard-

ing the inner vertices and keeping only the boundary vertices. These vertices form the initial front. New vertices are chosen from among the four potential neighbours of a vertex of the front. These neighbours lie at unit distance (measured with $\mathcal{M}^e$) from the vertex along the four directions given by moving forward or backward along the eigenvectors of $\mathcal{M}^I$, Algorithm 1 with $L = 1$. The principal sizes h_1^e, h_2^e of $\mathcal{M}^e$, as well as the angles θ^I, θ^e formed by the horizontal and the first eigenvector of $\mathcal{M}^I$ and $\mathcal{M}^e$, are used to compute the size along the eigenvector of $\mathcal{M}^I$ in the error metric. To avoid abrupt variations in the direction field, the next direction is taken as the closest to the previous one.

Input: Initial position $\mathbf{x}_0$, direction $j \in [1, 2, 3, 4]$, target length L, number of uniform steps N.
Result: Neighbouring vertex $\mathbf{x}$.
$\mathbf{x} = \mathbf{x}_0$
for $i = 1 \to N$ **do**
  $\mathbf{v}_1, \mathbf{v}_2 = \text{eigenvectors}(\mathcal{M}^I(\mathbf{x}))$
  if $i > 1$ **then**
    $\mathbf{v} = \arg\max_{\pm\mathbf{v}_1, \pm\mathbf{v}_2} \mathbf{v} \cdot \mathbf{v}_{\text{prev}}$
  else
    $\mathbf{V} = [\mathbf{v}_1, -\mathbf{v}_1, \mathbf{v}_2, -\mathbf{v}_2]$
    $\mathbf{v} = \mathbf{V}_j$
  end
  $h = \dfrac{h_1^e h_2^e}{\sqrt{h_1^e \sin^2(\theta^I - \theta^e) + h_2^e \cos^2(\theta^I - \theta^e)}}$
  $\mathbf{x} = \mathbf{x} + \dfrac{hL\mathbf{v}}{N}$
  $\mathbf{v}_{\text{prev}} = \mathbf{v}$
end
Algorithm 1: Compute neighbour to vertex $\mathbf{x}_0$.

The neighbour $\mathbf{x}_j, j = 1, 2, 3, 4$ is added to the front if it is not too close to another vertex of the front. To avoid computing distances to every existing vertex, an RTree data structure [20], consisting of a list of vertices along with their exclusion zone of characteristic size $1/\sqrt{2}$, is used. The exclusion zone of a vertex consists of its four neighbours at distance $L = 1/\sqrt{2}$, approximating the deformed unit ball of $\mathcal{M}^e(\mathbf{x})$ centered at the vertex and considering a varying metric (the true unit ball is not an ellipse anymore). A new vertex is added if it lies outside of the convex hull of the neighbouring vertices forming the exclusion zone.

The set of accepted vertices is triangulated using isotropic Delaunay triangulation, then edge swaps are performed to enhance element quality based on $\mathcal{M}^e$, yielding an anisotropic straight mesh.

3.3 Curving the edges

The edges are then curved by moving the midpoint. To curve the mesh in a single pass and not iteratively, the edges are curved to best approach the geodesics of the error metric $\mathcal{M}^e$. For each parabolic edge $\mathbf{x}(t) = \mathbf{x}(t, \boldsymbol{\alpha})$, we seek the displacement vector $\boldsymbol{\alpha}^* \in \mathbb{R}^2$ such that the error-weighted edge length:

$$\begin{aligned} \text{length}(\mathbf{x}) &= \int_0^1 \|\mathbf{x}'(t)\|_{\mathcal{M}^e}\, dt \\ &= \int_0^1 \sqrt{(\mathbf{x}'(t), \mathbf{x}'(t))_{\mathcal{M}^e}}\, dt \\ &= \int_0^1 \sqrt{(\boldsymbol{\gamma} + \boldsymbol{\alpha}\dot{L}, \boldsymbol{\gamma} + \boldsymbol{\alpha}\dot{L})_{\mathcal{M}^e}}\, dt \end{aligned} \tag{38}$$

is minimized. The minimization problem

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^2} \text{length}(\mathbf{x}) \tag{39}$$

is solved with a quasi-Newton method. In [6], the edges are curved by restricting the movement of the midpoint along the orthogonal bisector: we compare the influence of this choice in the results section. More precisely, three strategies are compared: (i) moving the midpoint along the bisector, (ii) moving the midpoint anywhere in $\mathbb{R}^2$ and (iii) moving the midpoint in $\mathbb{R}^2$, then relocating it at half of the curved edge's length, such that $\mathbf{x}(t = 1/2) = \mathbf{X}_{12}$.

This step is critical as several results show that curving the elements, i.e. using a non-affine transformation between the reference and the physical triangle, can have dramatic consequences on the interpolation quality, see e.g. [10, 21]. In particular, [10] shows that an asymptotic relation of the form:

$$\|u - \Pi^2 u\|_{L^2} = \mathcal{O}(h^3) \tag{40}$$

can be achieved on P^2 isoparametric triangles with Lagrange basis functions if the displacement vector satisfies:

$$\|\mathbf{X}_{12} - \mathbf{X}_{12}^{\text{straight}}\| = \mathcal{O}(h^2), \tag{41}$$

where $\mathbf{X}_{12}$ is the position of the P^2 midpoint and $\mathbf{X}_{12}^{\text{straight}} = (\mathbf{X}_1 + \mathbf{X}_2)/2$. This result is valid on regular families of element, that is, elements for which there exists a constant a_0 such that

$$0 < a_0 \leq \frac{\rho_h}{h}, \quad \forall h, \tag{42}$$

where h is the diameter of the element and ρ_h is the diameter of the inscribed sphere. This result was observed on a sequence of six regular meshes, such as the ones shown on Fig. 2. For each of these meshes, the inner edges are curved by moving the midpoint along the unit orthogonal bisector $\boldsymbol{\gamma}^\perp$:

$$\boldsymbol{\alpha} = C \left(\frac{1}{\sqrt{N}}\right)^m \boldsymbol{\gamma}^\perp \sim C h^m \boldsymbol{\gamma}^\perp, \tag{43}$$

with C a constant, N the number of vertices of the mesh and m an integer exponent.

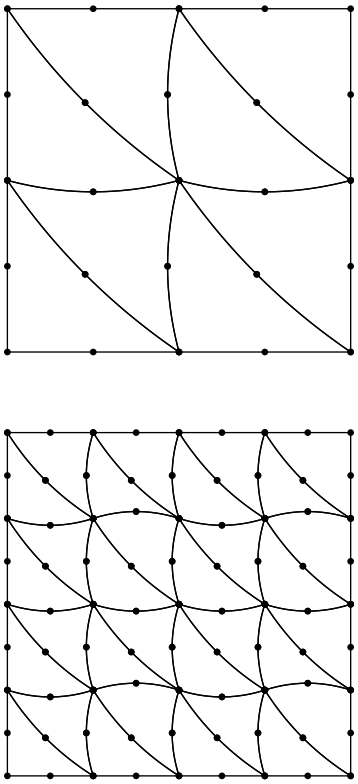


Figure 2: Structured meshes with edges curved along the orthogonal bisector.

The observed convergence, Fig. 3, follows the results from [10]: curving the elements with

$$\|\boldsymbol{\alpha}\| = \|\mathbf{X}_{12} - \mathbf{X}_{12}^{\text{straight}}\| = \mathcal{O}(h) \tag{44}$$

lowers the convergence rate to 2, whereas the optimal rate is maintained for a higher k.

However, we have observed in our numerical tests (Section 4) that curved meshes with the midpoint moved according to (39) can exhibit an $\mathcal{O}(h)$ evolution (or even lower) and still maintain the optimal convergence rate. This would suggest that the bounds from [10] are somewhat too conservative, and that curving the mesh along privileged directions can prevent this loss of convergence rate.

3.4 Making the mesh valid

The curved mesh is not necessarily valid, that is, we do not have $J_{\min} = \min_{\boldsymbol{\xi}} J(\boldsymbol{\xi}) > 0$ for each curved element, where $J(\boldsymbol{\xi}) = |\partial \mathbf{x}/\partial \boldsymbol{\xi}|$ is the determinant of the reference-to-physical transformation $\mathbf{x}(\boldsymbol{\xi})$. To make the mesh valid, we compute a lower bound on $J_{\min}$ on each element using the Bézier-based sufficient condition in [22]. If the element is invalid ($J_{\min} \leq 0$), we

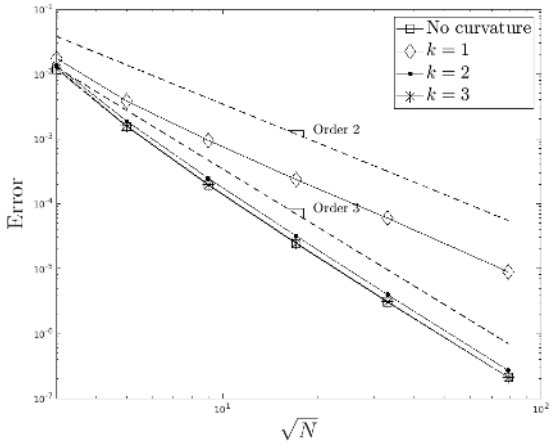


Figure 3: Interpolation error in L^2-norm on curved structured meshes for the function $u(x, y) = r^4(x, y) = x^4 + y^4 + 2x^2y^2$ for different curvature amplitudes.

backtrack on all three edges and simultaneously reduce the displacements $\boldsymbol{\alpha}_i, i = 1, 2, 3$ until $J_{\min}$ is positive. As the initial mesh is a valid straight mesh, the limit case is always valid.

3.5 Edge swaps

Finally, mesh quality is enhanced by performing edge swaps. As the vertices are supposed to be ideally placed, operations such as position smoothing, vertex insertions or edge collapses are not performed. The curvilinear quality indicator on a triangle T used is:

$$q_{\mathcal{M}} = 4\sqrt{3}\,\frac{\displaystyle\int_T \sqrt{\det \mathcal{M}}\, dx}{\sum_{i=1}^{3} \mathcal{L}_{\mathcal{M}}(e_i)}, \tag{45}$$

with $\mathcal{L}_{\mathcal{M}}(e_i)$ the length of the edge i. We select the error metric $\mathcal{M}^e$ to compute the quality.

4. NUMERICAL RESULTS

We test our methodology on two simple test cases:

- $u_1(x, y) = r^4(x, y) = (x^2 + y^2)^2$,
- $u_2(x, y) = \operatorname{atan}\left(10\left[\sin\left(\frac{3\pi y}{2}\right) - 2x\right]\right)$.

To focus on metric computation and mesh generation only, we use analytic derivatives $H_{ij}(\mathbf{x})$ and $C_{ijk}(\mathbf{x})$ of u. The log-simplex optimization problem (25) is solved using the SoPlex library [23] and the minimization problem (39) is solved using the Ceres library [24]. For each test case, we study the convergence of the Lagrange P^2 interpolation on

isoparametric elements by increasing the desired number of vertices N in each mesh. The study is performed on sets of meshes with target complexity $N = [50, 100, 200, 400, 600, 800, 1000, 1200, 1600]$. To generate a mesh of target complexity N_i, we proceed iteratively and start from a coarse structured background mesh. We compute the metric field on the background mesh, generate a straight anisotropic mesh then use this mesh as a background mesh to have a more accurate representation of the metric field. We iterate this way five times before curving the mesh. The generated meshes feature curved elements where necessary and typically contain about 1.15 times the requested number of vertices (Fig. 4 and 6).

Tests were performed sequentially on a laptop with an Intel Core i7 8750h CPU at 2.2 Ghz and 16Gb of memory. Non-optimized timings are presented in Table 1 for the second test case and for target complexities $N = 200$ and 1600. Despite some optimizations, the overall execution time remains very high. As expected, solving the minimization problem for $\mathcal{Q}$ is the costliest part of the metrics computation. Due to the high number of metric evaluations when computing length integrals, most of the meshing step is currently spent interpolating the metric and its derivatives from the background mesh.

	$N = 200$	1600
Metric computations (5 passes):		
Solve $\bar{s} = e^{-1}(1)$	1.47	10.27
Minimize $\det \mathcal{Q}$	8.20	51.89
Others (metric scaling, etc.)	0.81	8.04
Total metrics	$10.48s$	$70.2s$
Mesh generation:		
Generate nodes	3.22	28.24
Initial edges curving	0.82	6.9
Edge swaps	5.86	77.31
Others	0.42	3.49
Total mesh	10.32	115.94
incl. Metric interpolation	9.92	107.76
Total	$20.80s$	$186.57s$

Table 1: Timings for the second test case for target complexity $N = 200$ and 1600: computation of the metric tensor fields and mesh generation. All times are given in seconds.

As it is standard in anisotropic mesh adaptation, the error is reported as a function of the number of mesh vertices $N^{1/n}$ with n the space dimension, here 2. For straight meshes with interpolation functions of degree k in two dimensions, the continuous mesh theory [8, 15, 16] predicts an evolution of the error in the L^p norm of the form

$$\|e\|_{L^p} \sim CN^{-\frac{k+1}{2}} \sim C(\sqrt{N})^{-(k+1)}, \qquad (46)$$

thus an asymptotic convergence rate of $k+1 = 3$. The optimal convergence rate in L^2 norm is observed for both test cases, Fig. 5 and 7, as well as an order of 2 in H^1 norm. The graphs in Fig. 5 and 7 show the influence of the curving strategy on the interpolation error, and are associated to the three approaches discussed in Section 3.3. For both test cases, curving the edges along the bisector or relocating the midnode after curving without constraint yield very similar results, and both their error levels are slightly under those of the second curving strategy, which lets the midpoint move freely in $\mathbb{R}^2$. This suggests that curving along the orthogonal bisector is sufficient to generate optimally adapted meshes, in addition to being slightly faster (1 degree of freedom instead of 2). Finally, the evolution of the norm of the displacement $\boldsymbol{\alpha}$ as a function of $\sqrt{N}$ is shown on Fig. 5 and 7. Notably, the evolution is linear or sublinear, while maintaining optimal convergence rates for the interpolation error.

5. CONCLUSION AND FUTURE WORK

We have presented a methodology for two dimensional curvilinear mesh generation: interpolation error estimator for curved trajectories, generalization of the existing log-simplex algorithm for high-order metric tensor computation and frontal curved mesh generation. Adapted meshes exhibit mild to more pronounced curvature and reach the optimal third order convergence rate for P^2 Lagrange elements in L^2 norm. In particular, it was observed that edge curvature, represented by the displacement vector $\boldsymbol{\alpha}$, does not necessarily need to decrease faster than the edge length to maintain optimal convergence rates.

To keep the computation of the metric field tractable and to reduce the computational cost, the non-homogeneous character of the error polynomial was set aside to use existing techniques mostly as is. More tests are still necessary to assess the impact of this choice. Future work will be focused on this topic, as well as handling curved boundaries and tackling a three-dimensional extension of this framework.

6. ACKNOWLEDGMENTS

This work was funded by FRIA grant FC29571 (FRS-FNRS). Financial support from the Simulation-based Engineering Science program funded through the CREATE program from the Natural Sciences and Engineering Research Council of Canada is also gratefully acknowledged.

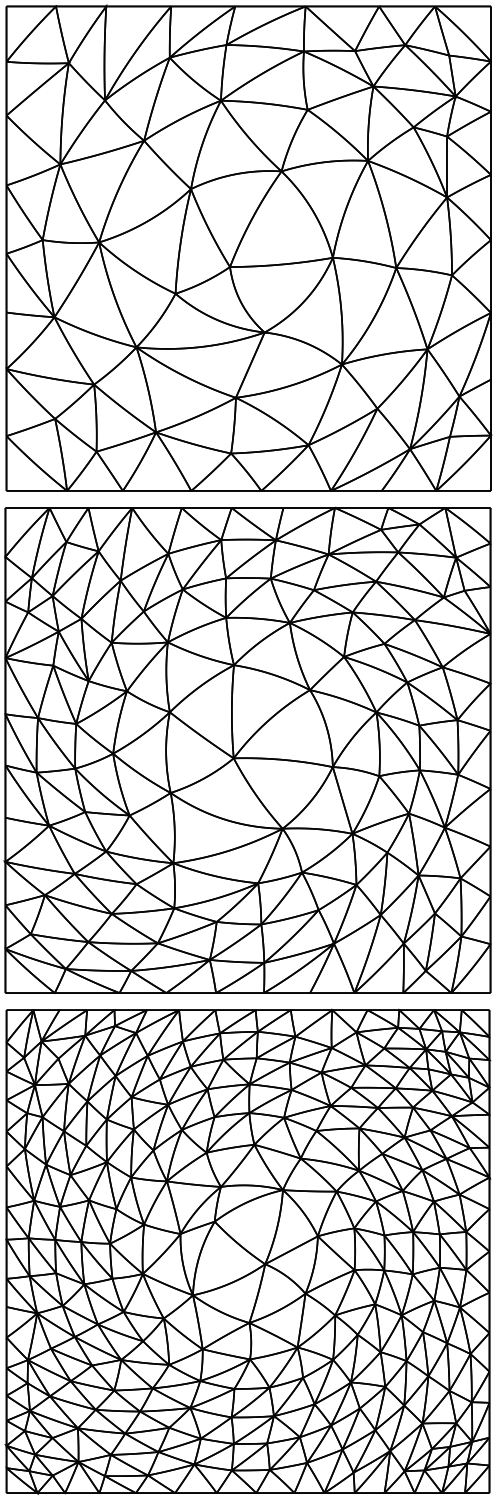


Figure 4: Adapted meshes for $u(x,y) = (x^2+y^2)^2$ and target complexity $N = 50, 100, 200$. Vertices are generated along the eigenvectors of the induced metric $\mathcal{M}^I$, which are the directions of the gradient and level curves of u. The size along these curves is determined by the error metric $\mathcal{M}^e$. The edges are curved by moving the midpoint freely to minimize the edge length in the error metric.

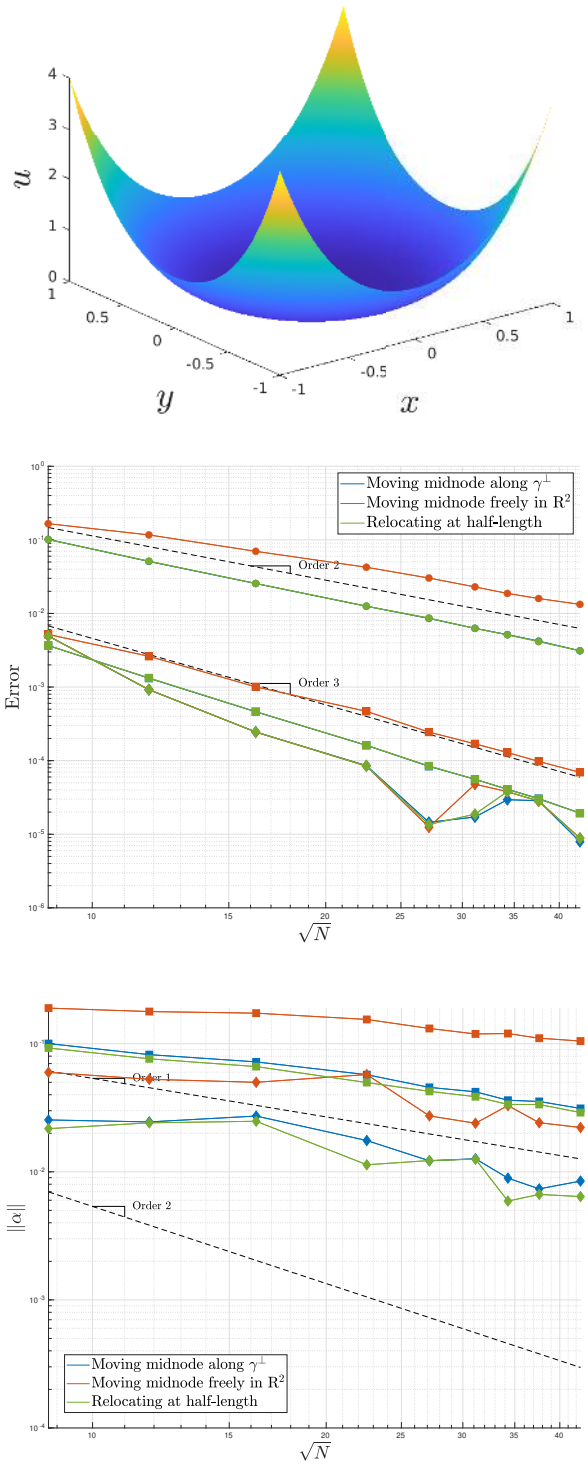


Figure 5: Top: surface plot for u_1. Middle: interpolation error for u_1 in L^2 (squares), L^∞ (diamonds) and H^1 (dots) norms for three approaches to edges curving. The error curves obtained by moving the midnode along the orthogonal bisector (blue) and by relocating the optimal midnode at half-length (green) are mostly identical. Bottom: L^2 (squares) and L^∞ (diamonds) norm of the displacement vector $\|\boldsymbol{\alpha}\| = \|\mathbf{X}_{12} - \mathbf{X}_{12}^{\text{straight}}\|$ as a function of $\sqrt{N}$ ($\sim 1/h$).

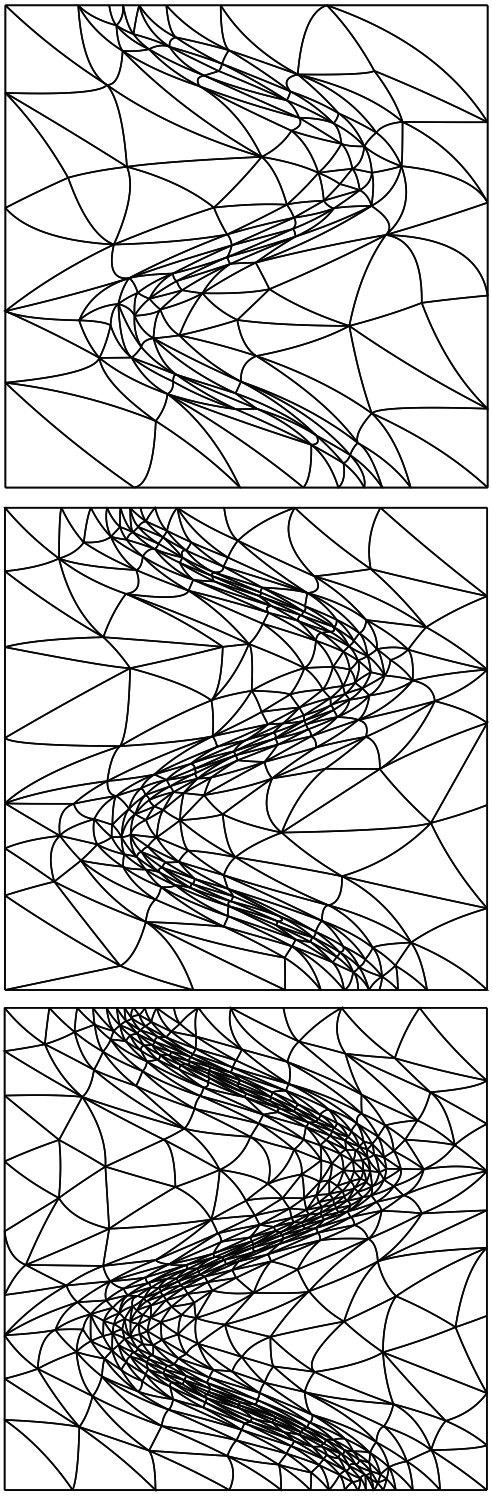
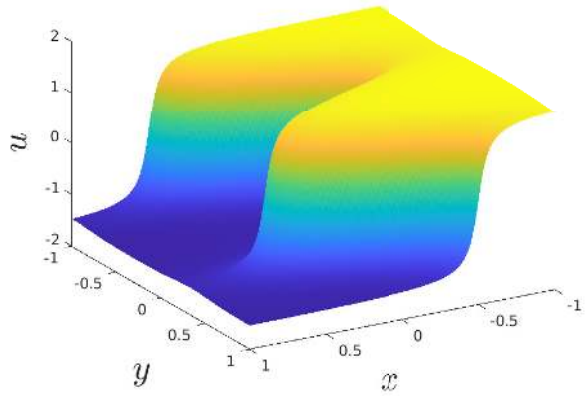
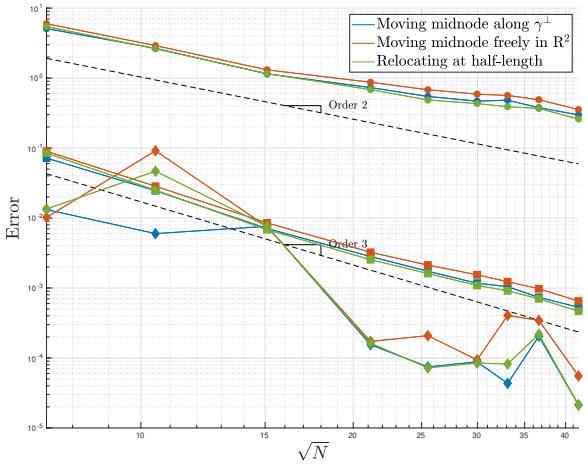


Figure 6: Adapted meshes for $u_2(x, y)$ and target complexity $N = 100, 200, 400$.





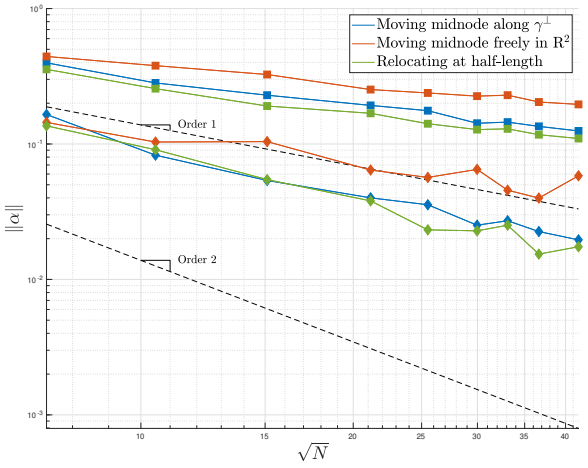


Figure 7: Top: surface plot for u_2. Middle: interpolation error for u_2 in L^2 (squares), L^∞ (diamonds) and H^1 (dots) norms for three approaches to edges curving. Bottom: L^2 (squares) and L^∞ (diamonds) norm of the displacement vector $\|\boldsymbol{\alpha}\| = \|\mathbf{X}_{12} - \mathbf{X}_{12}^{\text{straight}}\|$ as a function of $\sqrt{N}$ ($\sim 1/h$).

7. APPENDIX

Defining $a_i \triangleq \kappa v_i^{\perp}$ and

$$\bar{C}_{112} \triangleq C_{112} + C_{121} + C_{211}$$
$$\bar{C}_{122} \triangleq C_{122} + C_{212} + C_{221},$$

the coefficients of the error polynomial (14) are:

$$c_1 = \frac{1}{120}\Big(C_{111}a_1^3 + \bar{C}_{112}a_1^2a_2 + \bar{C}_{122}a_1a_2^2 + C_{222}a_2^3\Big),$$

$$c_2 = \frac{1}{20}\Big(C_{111}a_1^2v_1 + \bar{C}_{112}(a_1^2v_2 + 2a_1a_2v_1) + \bar{C}_{122}(a_2^2v_1 + 2a_1a_2v_2) + C_{222}a_2^2v_2\Big),$$

$$c_3 = \frac{1}{8}\Big(C_{111}a_1v_1^2 + \bar{C}_{112}(a_2v_1^2 + 2a_1v_1v_2) + \bar{C}_{122}(a_1v_2^2 + 2a_2v_1v_2) + C_{222}a_2v_2^2 + H_{11}a_1^2 + (H_{12} + H_{21})a_1a_2 + H_{22}a_2^2\Big),$$

$$c_4 = \frac{1}{6}\Big(C_{111}v_1^3 + 3\bar{C}_{112}v_1^2v_2 + 3\bar{C}_{122}v_1v_2^2 + C_{222}v_2^3\Big) + \frac{1}{2}\left(H_{11}a_1v_1 + H_{12}a_2v_1 + H_{21}a_1v_2 + H_{22}a_2v_2\right).$$

References

[1] Toulorge T., Geuzaine C., Remacle J.F., Lambrechts J. "Robust untangling of curvilinear meshes." *Journal of Computational Physics*, vol. 254, 8–26, 2013

[2] Fortunato M., Persson P.O. "High-order unstructured curved mesh generation using the Winslow equations." *Journal of Computational Physics*, vol. 307, 1–14, 2016

[3] Zhang R., Johnen A., Remacle J.F. "Curvilinear mesh adaptation." *International Meshing Roundtable*, pp. 57–69. Springer, 2018

[4] Aparicio-Estrems G., Gargallo-Peiró A., Roca X. "Defining a stretching and alignment aware quality measure for linear and curved 2D meshes." *International Meshing Roundtable*, pp. 37–55. Springer, 2018

[5] Aparicio-Estrems G., Gargallo-Peiró A., Roca X. "High-Order Metric Interpolation for Curved $r-$Adaptation by Distortion Minimization." *Proceedings of the 2022 SIAM International Meshing Roundtable*, pp. 11–22. 2022

[6] Zhang R., Johnen A., Remacle J.F., Henrotte F., Bawin A. "The generation of unit P^2 meshes: error estimation and mesh adaptation." *International Meshing Roundtable (virtual)*, pp. 1–13. 2021

[7] Rochery L., Loseille A. "P^2 Cavity Operator with Metric-Based Volume and Surface Curvature." *Proceedings of the 29th International Meshing Roundtable*, pp. 193–210. 2021

[8] Alauzet F., Loseille A., Dervieux A., Frey P. "Multi-dimensional continuous metric for mesh adaptation." *Proceedings of the 15th international meshing roundtable*, pp. 191–214. Springer, 2006

[9] Loseille A. *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.* Ph.D. thesis, Université Pierre et Marie Curie-Paris VI, 2008

[10] Ciarlet P.G., Raviart P.A. "Interpolation theory over curved elements, with applications to finite element methods." *Computer Methods in Applied Mechanics and Engineering*, vol. 1, no. 2, 217–249, 1972

[11] Mbinky E.C. *Adaptation de maillages pour des schémas numériques d'ordre très élevé.* Ph.D. thesis, Université Pierre et Marie Curie-Paris VI, 2013

[12] Hecht F., Kuate R. "An approximation of anisotropic metrics from higher order interpolation error for triangular mesh adaptation." *Journal of computational and applied mathematics*, vol. 258, 99–115, 2014

[13] Coulaud O., Loseille A. "Very high order anisotropic metric-based mesh adaptation in 3D." *Procedia engineering*, vol. 163, 353–365, 2016

[14] Shifrin T. "Differential geometry: a first course in curves and surfaces." *University of Georgia*, 2015

[15] Loseille A., Alauzet F. "Continuous mesh framework part I: well-posed continuous interpolation error." *SIAM Journal on Numerical Analysis*, vol. 49, no. 1, 38–60, 2011

[16] Loseille A., Alauzet F. "Continuous mesh framework part II: validations and applications." *SIAM Journal on Numerical Analysis*, vol. 49, no. 1, 61–86, 2011

[17] Frey P.J., George P.L. *Mesh generation: application to finite elements.* Iste, 2007

[18] Loseille A. "Metric-orthogonal anisotropic mesh generation." *Procedia Engineering*, vol. 82, 403–415, 2014

[19] Dobrzynski C. *MMG3D: User guide*, 2012

[20] Beckmann N., Kriegel H.P., Schneider R., Seeger B. "The R*-tree: An efficient and robust access method for points and rectangles." *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, pp. 322–331. 1990

[21] Botti L. "Influence of reference-to-physical frame mappings on approximation properties of discontinuous piecewise polynomial spaces." *Journal of Scientific Computing*, vol. 52, no. 3, 675–703, 2012

[22] Johnen A., Remacle J.F., Geuzaine C. "Geometrical validity of curvilinear finite elements." *Journal of Computational Physics*, vol. 233, 359–372, 2013

[23] Gamrath G., Anderson D., Bestuzheva K., et al. "The SCIP Optimization Suite 7.0." Tech. Rep. 20-10, ZIB, Takustr. 7, 14195 Berlin, 2020

[24] Agarwal S., Mierle K., Team T.C.S. "Ceres Solver.", 3 2022. URL `https://github.com/ceres-solver/ceres-solver`