AN EFFICIENT SOLVER TO APPROXIMATE CAD CURVES WITH SUPER-CONVERGENT RATES

Julia Docampo-Sánchez   Eloi Ruiz-Gironés   Xevi Roca

Barcelona Supercomputing Centre-Centro Nacional de Supercomputacin BSC-CNS, Spain.
julia.docampo@bsc.es, eloi.ruiz@bsc.es, xevi.roca@bsc.es

ABSTRACT

We present a specific-purpose solver to approximate curves with super-convergent rates. To obtain super-convergence, we minimize a disparity measure in terms of a piece-wise polynomial approximation and a curve re-parametrization. We have numerical evidence that the disparity converges with $2p$ order for planar curves and $\lfloor \frac{3}{2}(p-1) \rfloor + 2$ for 3D curves, p being the mesh polynomial degree. To meet these rates, we exploit the quadratic convergence of a globalized Newton's method with the help of three main ingredients. First, we employ a nonmonotone line search reducing the number of nonlinear iterations. The second ingredient is to introduce a log barrier function preventing element inversion in the curve re-parameterization. Third, we propose a constrained optimization of the disparity functional where the element interfaces are fixed, improving the computational efficiency whilst preserving super-convergence. We approximate analytic curves as well as CAD models with meshes of several polynomial degrees. We conclude that the solver is well-suited to obtain super-convergent approximations to curves at reasonable computational times.

Keywords: curve approximation, distance optimization, super-convergence, high-order meshes

1. INTRODUCTION AND MOTIVATION

Geometric accuracy plays a major role in the performance of unstructured high-order methods [1], requiring curved elements to meet the desired accuracy. The geometric accuracy is measured as the distance between the mesh and the target geometry. Traditionally, in computational geometry, this corresponded to the Fréchet and Hausdorff distances [2]. More recently, distance optimization techniques have been proposed. For example, in [3, 4] an area and Taylor based distance optimizer are used respectively for 2D and 3D geometry. The authors report significant mesh-CAD distance reductions at adequate computational times.

In addition, a disparity measure for generating optimal curved high-order meshes was proposed in [5]. The optimization combines a distortion measure for mesh quality and a geometric L^2-disparity measure for geometric error [6]. It produces optimal non-interpolative meshes and it has been observed that this disparity is $2p$ super-convergent [7]. This affords a straightforward advantage: one can obtain the desired geometric accuracy using smaller polynomial degrees than with standard interpolation approaches.

As in many optimization problems, the original disparity was solved with a Newton method combined with Armijo backtracking line search. The Armijo rule is monotonic: the iteration is valid if the objective function decreases. For highly nonlinear problems, monotonicity can trap the optimizer if it follows a narrow curved valley (making very short steps or zigzagging) and reduces the convergence rate [8]. This has motivated the development of nonmonotone line searches.

The first nonmonotone line search was based on a maximum principle [9]. A step is considered valid if it improves the objective function with respect to the maximum value of the objective functions corresponding to the N previous iterations. Later, a family of nonmonotone line searches were proposed where a tuning parameter shifts the search condition from maximum to an average [10]. The authors prove global convergence for this family of line search methods. In fact, nonmonotone line searches have demonstrated an improvement in computational efficiency, as well as likelihood of finding a global minimum [9, 11].

The work presented here focuses on improving the computational performance during the optimization of the disparity measure. Focusing on curves, we propose:

1. A reduction of the optimization dimension by fixing the element interfaces.
2. A logarithmic barrier preventing curve tangling.
3. An average based (nonmonotone) line search.

Our results show that switching to a nonmonotone line search consistently reduces the number of iterations. Furthermore, by fixing the element interfaces we reduce the dimension of the optimization problem. Although this constrained version is sub-optimal in terms of the error compared to the original disparity (free interfaces), we are able to preserve the same super-convergent property. We have performed numerical tests based on analytic curves as well as CAD geometry obtained through ESP [12].

This paper is organized as follows. In Section 2, we define the disparity measure for curves and show the super-convergent property through an example. In Section 3, we discuss the disparity and constrained optimization and the new solver features. Finally, in Sections 4 and 5, we compare the errors and iterations from the constrained and unconstrained problem studying several CAD geometries. The paper concludes with Section 6 discussing main results.

2. DISPARITY MEASURE: CURVES

We begin by introducing the disparity formulation for curves. We will discuss how it is defined, optimized and implemented as well as show how it attains $2p$ order. For more details, we refer the reader to [7].

2.1 Mathematical formulation

We define our mesh as a set of elements where for each physical element e^P there is a reference element e^R. The physical mesh $\mathcal{M}^P$ can be defined in terms of an element-wise parametrization ϕ^P:

$$\phi^P|_{e^R} : e^R \to e^P \subset \mathbb{R}^n \tag{1}$$

$$\xi \to \mathbf{x} = \sum_{i=1}^{p+1} \mathbf{x}_i N_i^p(\xi), \tag{2}$$

where $p+1$ are the number of nodes for the high-order element e^P, $\mathbf{x}_i$ the $\mathbb{R}^n$-physical coordinates of the i-th node and $\{N_i^p\}_{i=1}^{p+1}$ a Lagrangian basis of degree p.

Let $\mathcal{C} \subset \mathbb{R}^n$ be a curve parametrized by $\boldsymbol{\alpha} : [a, b] \subset \mathbb{R} \to \mathcal{C}$ and consider the family of mappings:

$$\Pi := \left\{ \pi \in \mathcal{H}^1(\mathcal{M}^P, \mathcal{C}),\ \pi \text{ diffeomorphism} \right\}.$$

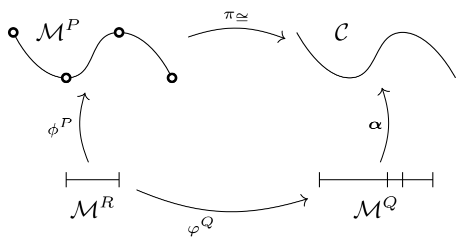


Figure 1: commutative diagram showing the reference mesh $\mathcal{M}^R$, the mappings to the physical meshes: ϕ^P and ϕ^Q respectively, the curve $\mathcal{C}$ and its parametrization $\boldsymbol{\alpha}$ and the projection π.

The diagram in Figure 1 shows the reference mesh $\mathcal{M}^R$, the physical mesh $\mathcal{M}^P$ consisting of three elements, the target curve $\mathcal{C}$ and the diffeomorphism π. The disparity measure between the high-order mesh and the curve is the minimal projection error

$$d(\mathcal{M}^P, \mathcal{C}) = \inf_{\pi \in \Pi} \left(\int_{\mathcal{M}^P} |\mathbf{x} - \pi(\mathbf{x})|^2 d\mathbf{x} \right)^{1/2}, \tag{3}$$

where $|\cdot|$ denotes the Euclidean norm of vectors. Defining the functional

$$E(\mathbf{x}, \pi) = \int_{\mathcal{M}^P} |\mathbf{x} - \pi(\mathbf{x})|^2 d\mathbf{x} \tag{4}$$

$$= \int_{\mathcal{M}^R} |\phi^P(\xi) - \pi \circ \phi^P(\xi)|^2 |\dot{\phi}^P(\xi)| d\xi \tag{5}$$

$$= ||\phi^P - \pi \circ \phi^P||_\sigma^2, \tag{6}$$

we establish the following relation:

$$d(\mathcal{M}^P, \mathcal{C})^2 = \inf_{\pi \in \Pi} E(\mathbf{x}, \pi). \tag{7}$$

Notice that we use the sub-index σ to denote that it is integral with weight $|\dot{\phi}^P|$.

Consider any possible curve reparametrization: $\boldsymbol{\alpha} \circ s$. As in (1), we define a 1D mesh through the mapping:

$$\varphi^Q|_{e^R} : e^R \to e^Q \subset \mathbb{R} \tag{8}$$

$$\xi \to s = \sum_{i=1}^{q+1} s_i \cdot N_i^q(\xi). \tag{9}$$

Note 1 *The mesh ϕ^P is in the physical space whereas φ^Q is a mesh in the parametric space. Modifying φ^Q results in different curve parametrizations.*

As shown in Figure 1, we have that $\pi \circ \phi^P = \boldsymbol{\alpha} \circ \varphi^Q$. Hence, we can reformulate the problem:

$$E(\mathbf{x}, \pi) = E(\mathbf{x}, s) = ||\mathbf{x} - \boldsymbol{\alpha} \circ s||^2_\sigma \quad (10)$$

$$= \int_{\mathcal{M}^R} |\phi^P(\xi) - \boldsymbol{\alpha} \circ \varphi^Q(\xi)|^2 |\dot{\phi^P}(\xi)| d\xi. \quad (11)$$

Therefore, the mapping $s : \mathcal{M}^R \to \mathcal{M}^Q$ needs to be a diffeomorphism, too. In Section 3, we show how to enforce this numerically.

Optimizing E with respect to s gives the disparity between the mesh and the curve. If we optimize E with respect to both $\mathbf{x}$ and s, we obtain the mesh with optimal geometric accuracy according to the disparity measure. We define the optimal approximation as:

$$\mathbf{x}^\star, s^\star = \arg\min_{\mathbf{x},s} ||\mathbf{x} - \boldsymbol{\alpha} \circ s||^2_{\boldsymbol{\sigma}} \quad (12)$$

$$= \arg\min_{\mathbf{x},s} \int_{\mathcal{M}^R} |\mathbf{x}(\xi) - \boldsymbol{\alpha} \circ s(\xi)|^2 |\dot{\mathbf{x}}(\xi)| d\xi. \quad (13)$$

Remark 1 *The optimal $s^\star$ (with respect to $\mathbf{x}^\star$) minimizes the error in the tangent direction. In Figure 2, we draw the point-wise errors starting with an interpolative mesh. After optimization, we see that the errors align with the curve normal direction.*

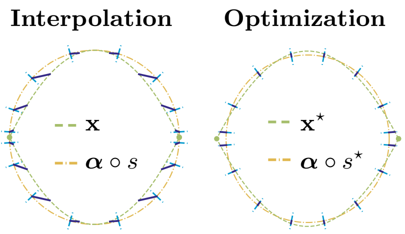


Figure 2: a circle meshed with two $p = 2$ elements showing the point-wise errors (solid deep blue lines). The curve normal direction is denoted by dashed light blue lines.

2.2 An example of super-convergence

Let us discuss the role played by the physical ($\mathbf{x}$) and parametric (s) meshes. Figure 3 shows several point-wise errors: first, we approximate the circle with interpolative meshes: $(\mathbf{x}_0, s_0)$. Then, we compute the disparity measure of this mesh optimizing the parametric mesh, $s^\star = \arg\min_s ||\mathbf{x} - \boldsymbol{\alpha} \circ s||^2_\sigma$. Finally, we optimize both: $\mathbf{x}^\star, s^\star = \arg\min_{\mathbf{x},s} ||\mathbf{x} - \boldsymbol{\alpha} \circ s||^2_\sigma$. Notice that the optimal mesh significantly improves the geometric error.

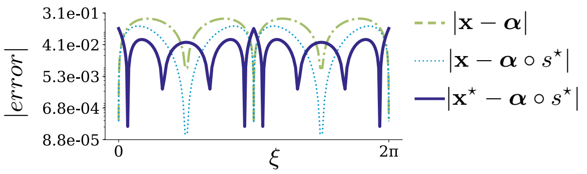


Figure 3: point-wise errors approximating a circle with two elements of degree $p = 2$ obtained with: direct interpolation $|\mathbf{x}(\xi) - \boldsymbol{\alpha}(\xi)|$ (dashed green), the disparity $|\mathbf{x}(\xi) - \boldsymbol{\alpha} \circ s^\star(\xi)|$ (dotted light blue) and the optimized disparity $|\mathbf{x}^\star(\xi) - \boldsymbol{\alpha} \circ s^\star(\xi)|$ (solid deep blue).

In Figure 4, we show convergence plots of the disparity measure when approximating the same circle using meshes of degree $p = 2, 3, 4$ and for several h-refinements. The initial disparity gives to $p + 1$ order. On the other hand, the slope of the optimal pair $(\mathbf{x}^\star, s^\star)$ shows a convergence order of $2p$.

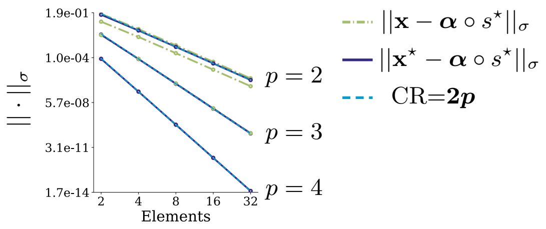


Figure 4: slopes (log-log) of the disparity approximating a circle for several mesh refinements showing the convergence rates (CR) before $(\mathbf{x}, s^\star)$ and after $(\mathbf{x}^\star, s^\star)$ optimizing the disparity measure.

2.3 Optimization challenges

The solution $(\mathbf{x}^\star, s^\star) = \arg\min_{\mathbf{x},s} E(\mathbf{x}, s)$ is found with a monotone backtracking line search:

$$(\mathbf{x}_{n+1}, s_{n+1}) = (\mathbf{x}_n, s_n) + \alpha \boldsymbol{\delta}_n, \quad (14)$$

where $\boldsymbol{\delta}_n$ is a Newton step and $\alpha \in (0, 1]$ is such that

$$E_{n+1} < E_n + \alpha 10^{-4} \boldsymbol{\delta}_n \cdot \nabla E_n \quad \text{(Armijo rule)}. \quad (15)$$

This rule ensures that the objective function (disparity) decreases in every step. However, it has an impact on the number of iterations. Furthermore, decreasing E_n does not assure valid solutions; since s is unconstrained, $\boldsymbol{\alpha} \circ s$ can tangle during minimization.

In Figure 5, we show the optimization results for the six edges of a simple CAD body obtained from ESP [12]. We study the number of iterations (top) as well as the point-wise errors (bottom). The point-wise error is computed as $|x - \boldsymbol{\alpha} \circ s|$ (initial) and $|x^\star - \boldsymbol{\alpha} \circ s^\star|$ (optimal), respectively. Observe that although the error improves, the number of iterations is large, reaching almost 4000 in several cases.

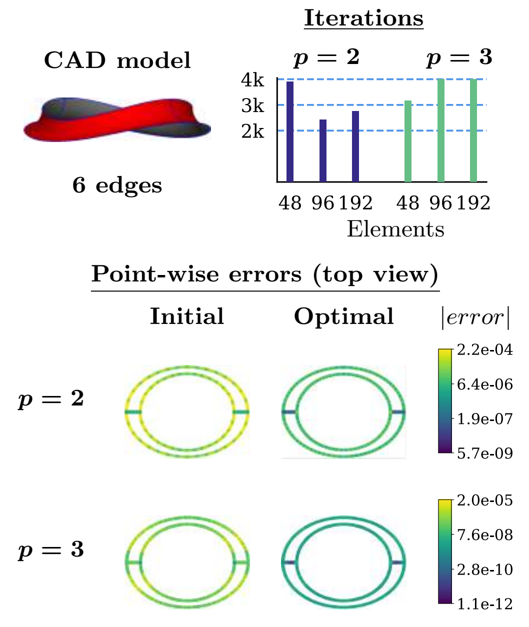


Figure 5: optimal disparity using the Armijo rule for the 6 edges of a CAD model and meshes consisting of 8 elements (degrees $p = 2, 3$). The plots show the iterations taken by the optimizer (top) and the initial and optimal point-wise errors respectively (bottom).

This paper addresses these computational issues. First, we propose reducing the dimension of the optimization problem by defining a constrained disparity functional with fixed element interfaces. Second, we insert a logarithmic barrier preventing $s^\star$ from tangling. Last, we introduce the average line search in [10] reducing the number of nonlinear iterations.

3. NEW SOLVER: CONSTRAINED OPTIMIZATION, LOG BARRIER AND AVERAGE LINE SEARCH

We now present a modified optimization approach for the disparity. We give an analogous mathematical formulation and highlight the differences with respect to the original method. Then, we focus on the solver details and introduce two major modifications: the log barrier function which effectively prevents curves from tangling and the Zhang-Hager [10] line search.

3.1 Optimizing the constrained disparity

To optimize the geometric accuracy of a high-order mesh, we solve the problem:

$$E(\mathbf{x}^\star, s^\star) = \min_{\mathbf{x},s} ||\mathbf{x} - \boldsymbol{\alpha} \circ s||^2_\sigma,$$

with $\mathbf{x}$ consisting of elements of degree p and s elements of degree q. Since we define our elements as all possible mappings from the reference to the physical element: $\phi^P : e^R \to e^P$, we are considering all possible partitions along the curve. Thus, the optimal mesh according to the disparity alters the initial element partition. This is illustrated in Figure 6 where we have optimized a spiral curve. The initial element configuration both in $\mathbf{x}$ and s changes after optimization, moving elements towards the spiral end.

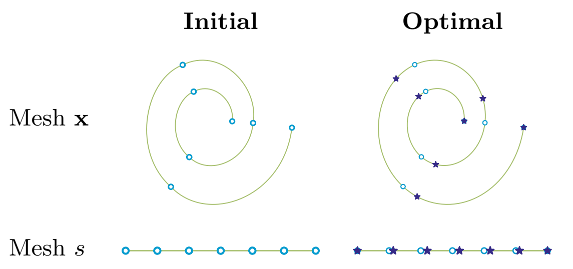


Figure 6: a spiral approximated with 6 elements (interfaces denoted by $\circ$, $\star$) showing the meshes $\mathbf{x}$ and s before and after optimizing the disparity.

Assume now a fixed element configuration and, at each element, define the meshes by

$$\tilde{\mathbf{x}}(\xi) = \underbrace{\mathbf{x}_0 N_0^p(\xi) + \mathbf{x}_p N_{p+1}^p(\xi)}_{\mathbf{x}_F(\xi)} + \sum_{i=2}^{p} \tilde{\mathbf{x}}_i \cdot N_i^p(\xi), \quad (16)$$

$$\tilde{s}(\xi) = \underbrace{s_0 N_0^q(\xi) + s_q N_{q+1}^q(\xi)}_{s_F(\xi)} + \sum_{i=2}^{q} \tilde{s}_i \cdot N_i^q(\xi), \quad (17)$$

where $\mathbf{x}_0$, $\mathbf{x}_{p+1}$, s_0, s_{q+1} are fixed throughout the optimization. Note that the indices run from 2 to p and 2 to q respectively, excluding the first and last nodes. We define the optimal *constrained* mesh as

$$\tilde{\mathbf{x}}^\star, \tilde{s}^\star = \arg\min_{\tilde{\mathbf{x}},\tilde{s}} ||\tilde{\mathbf{x}} - \boldsymbol{\alpha} \circ \tilde{s}||^2_\sigma. \quad (18)$$

Note that the overall accuracy is dictated by the initial element distribution. A uniform parametric partition on the CAD may lead to a poor element distribution. When a curvature-based mesher is not available, we propose a pre-processing stage: first, optimize the unconstrained disparity functional and obtain the linear meshes $(\mathbf{x}_1^\star,\ s_1^\star)$ with optimal (in the disparity sense) element distribution. Then, we p-refine using $s_1^\star$:

- For each element in the parametric mesh, we create a high-order element in the physical mesh.
- For each element of the parametric mesh, the nodes of the physical mesh are created as

  $$\mathbf{x}_i = \boldsymbol{\alpha} \circ s_1^\star(\xi_i), \quad i = 1, \ldots, p+1,$$

  where ξ_i is a distribution of high-order nodes in the master element.

Finally, we fix the high-order element interfaces and optimize the constrained disparity functional. In Section 4, we show an example of how the initial interpolative meshes benefit from this pre-processing stage.

3.2 Logarithmic barrier

The commutative diagram shown in Figure 1 holds if all mappings are diffeomorphisms. Since there are no constraints in formulation (12), in particular, diffeomorphism s is not actively enforced. The curve will tangle if elements in the parametric mesh s are inverted. A possible solution is to add a constraint on s' through the line search [13, 14]. Alternatively, we can avoid curve tangling by introducing a log barrier function.

log barrier [15, Ch.9]: consider the nonlinear programming problem:

$$\min f(x) \quad \text{subject to} \quad c_i \geq 0, \quad i = 1, \ldots, m.$$

The logarithmic barrier is a *penalty* term that moves the optimizer away from violating any of the constraints c_i. For a given μ, one can solve instead

$$\min_x P(x;\mu) = \min_x f(x) - \mu \sum_{i=1}^{m} \log(c_i(x)).$$

A suitable curve parametrization should follow the curve either forward or backwards. When we compose $\boldsymbol{\alpha} \circ s$, we can ensure that the reparametrization preserves direction by fixing the sign of s' (positive being forward and negative backwards) at the beginning of the optimization. For example, in Figure 7 we show a NACA curve and the initial parametrization $\boldsymbol{\alpha} \circ s = \boldsymbol{\alpha}$. Note that since s has not been optimized, it behaves as a linear mapping with a constant derivative.

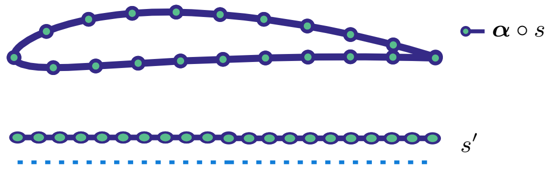


Figure 7: initial parametrization of a NACA curve and the s' profile with respect to the zero axis (...).

Regarding the disparity (either constrained or unconstrained), we want to solve:

$$\min_{\mathbf{x},s} E(\mathbf{x}, s) \quad \text{subject to} \quad s'(\xi) > 0 \quad \forall \xi.$$

Remark 2 *In this case, E can be used to obtain either the original $(\mathbf{x}^\star, s^\star)$ or the constrained $(\tilde{\mathbf{x}}^\star, \tilde{s}^\star)$ solutions. Both solutions benefit from this technique.*

We need a continuous barrier function so we introduce:

$$P(\mathbf{x}, s; \mu) = E(\mathbf{x}, s) - \mu \int_{\mathcal{M}^R} \log(s'(\xi)) d\xi. \qquad (19)$$

For each μ, we optimize instead the following:

$$(\mathbf{x}^\star, s^\star) = \arg\min_{\mathbf{x},s} P(\mathbf{x}, s; \mu), \qquad (20)$$

and use that if s is not tangled, then

$$\lim_{\mu \to 0} P(\mathbf{x}, s; \mu) = E(\mathbf{x}, s).$$

Note 2 *In practise, the* log*-barrier is activated only if at a particular Newton step, the solver detects a change in the sign of s'. This check is done oversampling s at each element. At that point, it retrieves the previous valid pair $(\mathbf{x}_n, s_n)$ and solves instead the penalized problem $P(\mathbf{x}, s; \mu_k)$, $k = 1, \ldots, M$.*

In Figure 8 (left), we show the optimized NACA curve $\boldsymbol{\alpha} \circ s$ without a log barrier where the curve develops artificial loops. Observe how the derivative profile s' crosses the zero axis in several locations. On the right, we show the same curve but optimized with the log-barrier, resulting in a valid curve.

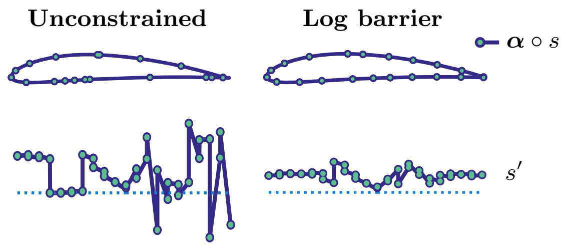


Figure 8: untangling the NACA curve. Optimizing with and without log barrier. (...) denotes $s' = 0$.

3.3 Zhang-Hager line search

We find the solution to our minimization problem combining Newton with backtracking line search:

$$(\mathbf{x}_{n+1}, s_{n+1}) = (\mathbf{x}_n, s_n) + \alpha_n \boldsymbol{d}_n. \qquad (21)$$

Let $\nabla(\cdot)$ and $H(\cdot)$ denote the gradient and Hessian operators respectively and $\boldsymbol{\delta}_n$ a Newton step:

$$H(E_n)\boldsymbol{\delta}_n = -\nabla(E_n). \qquad (22)$$

At each iteration, we ensure a descent direction using:

$$\boldsymbol{d}_n := \begin{cases} \boldsymbol{\delta}_n, & \text{if } \boldsymbol{\delta}_n \cdot \nabla E_n < 0, \\ -\Big[diag(H(E_n))\Big]^{-1} \nabla E_n, & \text{otherwise.} \end{cases} \qquad (23)$$

As mentioned before, the standard line search choice is the *Armijo* (monotone) rule: $\alpha \in (0, 1]$ satisfies that

$$E_{n+1} < E_n + \alpha 10^{-4} \boldsymbol{d}_n \cdot \nabla E_n. \quad (24)$$

Instead, we use a type of Zhang-Hager nonmonotone line search [10]. Let $C_0 = E_0,\ Q_0 = 1$ and define:

$$Q_{n+1} = \eta_n Q_n + 1 \quad C_{n+1} = \frac{\eta_n Q_n C_n + E_{n+1}}{Q_{n+1}}, \quad (25)$$

Notice that $\eta \equiv 0$ gives Armijo's monotone line search. On the other hand, $\eta \equiv 1$ gives an average based rule:

$$C_n = \frac{1}{n} \sum_{i=1}^{n} E_n, \quad (26)$$

which is the one that we will use in our experiments. Finally, α_n satisfies **Wolfe conditions:**

$$E_{n+1} \leq C_n + \sigma_1 \alpha_n \nabla E_n \cdot d_n \quad (27)$$

$$\nabla E_{n+1} \cdot d_n \geq \sigma_2 \nabla E_n \cdot d_n \quad (28)$$

with $\sigma_1 = 10^{-4}$ and $\sigma_2 = 0.9$ as suggested in [15].

3.4 Algorithm

We provide the implementation details of the proposed solver for the optimization of the constrained and unconstrained disparity measure, see Algorithm 1.

For the constrained problem, we solve the nonlinear optimization problem:

$$\tilde{E}(\tilde{\mathbf{x}}^\star, \tilde{s}^\star) = \min_{\tilde{\mathbf{x}}, \tilde{s}} ||\tilde{\mathbf{x}} - \boldsymbol{\alpha} \circ \tilde{s}||_\sigma, \quad (29)$$

where $\tilde{\mathbf{x}}$ and $\tilde{s}$ have the element interfaces fixed.

The main function is OPTIMIZE. It takes several arguments: $\boldsymbol{\alpha}$ gives information about the curve. Parameters p and q denote the polynomial degrees in $\mathbf{x}$ and s, respectively. M is the number of outer iterations corresponding to the log barrier term μ and N the maximum nonlinear iterations allowed. In our experiments, we use M=6 with μ decreasing by a factor of 10^{-2} at each step. The parameter optimizePartitions indicates whether or not the initial element partition should be optimized (using the original disparity with $p = q = 1$). Finally, freeInterfaces is a flag indicating if the element interfaces are fixed or not.

At the start (lines 2-5), we check if the CAD model has already an initial tessellation. Otherwise, we create an element partition. Then, if optimizePartitions is activated, we set $p = q = 1$ and call again the function optimize with the flag freeInterfaces activated (lines 6-7). In this case, the same routine follows but changes $\tilde{E}$ (constrained) to E (original disparity). Next (line 8), we generate the high-order interpolative meshes. At each element, we set:

Algorithm 1 constrained disparity minimization

```
 1: function OPTIMIZE($\boldsymbol{\alpha}$, $p$, $q$, M, N, optimizePartition, freeInterfaces)
 2:     if $\boldsymbol{\alpha}.tess = true$ then
 3:         $r = \boldsymbol{\alpha}.tess$
 4:     else
 5:         $r = divide(\boldsymbol{\alpha}, n)$
 6:     if optimizePartition $= true$ then
 7:         $r$, _ ←OPTIMIZE($\boldsymbol{\alpha}$, 1, 1, M, N, false, true)
 8:     $\tilde{\mathbf{x}}_0,\ \tilde{s}_0$ ← REFINE($\boldsymbol{\alpha}$, $r$, $p$, $q$)
 9:     $\mu = 0$; logBarrier $= false$;
10:     for $m = 1$ :M do
11:         if $m$ =M then
12:             $\mu = 0$
13:         else
14:             $\mu = \mu \cdot 10^{-2}$
15:         for n=1:N do
16:             $\nabla E_n, H(E_n)$ ←GRADHESS($\boldsymbol{\alpha}$, $\mathbf{x}_n$, $s_n$, freeInterfaces, $\mu_k$)
17:             if $|\nabla(E_n)| < tol$ then
18:                 break
19:             $\boldsymbol{\delta} = -H(E_n)^{-1} \nabla E_n$
20:             if $\boldsymbol{\delta} \cdot \nabla E_n < 0$ then
21:                 $\boldsymbol{d} = \boldsymbol{\delta}$
22:             else
23:                 $\boldsymbol{d} = -\Big[diag(H(E_n))\Big]^{-1} \nabla E_n$
24:             $\beta = 1$
25:             repeat
26:                 $(\tilde{x}_{n+1},\ \tilde{s}_{n+1}) = (\tilde{x}_n,\ \tilde{s}_n) + \beta \boldsymbol{d}$
27:                 $\beta = \beta/2$
28:             until ZHANGHAGER($\tilde{x}_{n+1}$, $\tilde{s}_{n+1}$) = true
29:             if any($sign(\tilde{s}'_{n+1}) \neq sign(\tilde{s}'_0)$) then
30:                 $(x_{n+1},\ s_{n+1}) = (x_n,\ s_n)$
31:                 $\mu = ||\tilde{\mathbf{x}}_n - \boldsymbol{\alpha} \circ \tilde{s}_n||^2_\sigma$
32:                 logBarrier $= true$
33:                 break
34:         if logBarrier $= false$ then
35:             break
36:     return $\tilde{\mathbf{x}}_n, \tilde{s}_n$
```

$$\tilde{\mathbf{x}}(\xi) = \mathbf{x}^0(\xi) + \sum_{i=2}^{p} \tilde{\mathbf{x}}_i N_i^p(\xi), \quad (30)$$

$$\tilde{s}(\xi) = s^0(\xi) + \sum_{i=2}^{q} \mathbf{x}_i N_i^q(\xi), \quad (31)$$

where $\mathbf{x}^0$, s^0 are fixed (free) when optimizing $\tilde{E}$ (E).

We then initialize the log barrier variable (line 9) and enter the optimization loops. The outer loop (starting at line 10) corresponds to the penalized problem (equation (19)). The inner loop (lines 15-34) is the backtracking Newton scheme minimizing $\tilde{E}(\tilde{\mathbf{x}}^\star, \tilde{s}^\star)$. We compute the gradient and Hessian (line 16) and check the stopping criteria (lines 17-19). In our experiments, it corresponds to $tol = 10^{-12}$. Then, a descent direction is chosen (lines 19-23) and the solver enters a loop until the line search condition is met (lines 25-28). Finally, we sample s' for any changes in its sign. If so, we activate the log barrier (lines 29-32). The main function returns the optimized pair $(\tilde{\mathbf{x}}_n,\ \tilde{s}_n)$.

4. EXPLOITING LOCAL HIGHER ORDER ACCURACY

4.1 Constrained versus unconstrained optimization

Here we compare the convergence of the solution to the target geometry for the constrained and unconstrained optimization of the disparity. Then, we study the error behaviour focusing on a single element and show that it is possible to attain super-convergence optimizing only the internal nodes.

In Figure 9, we show the point-wise errors of the original $(\mathbf{x}^\star, s^\star)$ and constrained $(\tilde{\mathbf{x}}^\star, \tilde{s}^\star)$ optimization using the spiral from Section 3 as the target geometry (Figure 6). The point-wise error $|\tilde{\mathbf{x}}^\star - \boldsymbol{\alpha} \circ \tilde{s}^\star|$ is larger than $|\mathbf{x}^\star - \boldsymbol{\alpha} \circ s^\star|$. Also, unlike $\mathbf{x}^\star$, $\tilde{\mathbf{x}}^\star$ interpolates at the element interfaces. We will see that despite having larger errors, the constrained problem preserves the super-convergent behavior from the original disparity.

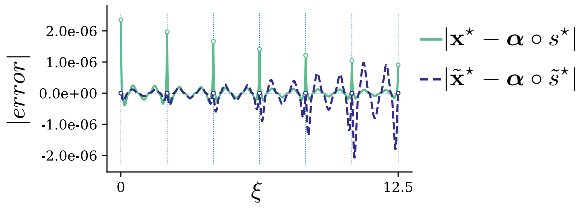


Figure 9: spatial error distribution showing element interfaces (light blue) for the minimization arguments $(\mathbf{x}^\star, s^\star)$ (solid green) and $(\tilde{\mathbf{x}}^\star, \tilde{s}^\star)$ (dashed deep blue) of the original and constrained disparity, respectively.

Fixing element interfaces transforms the optimization problem in R (total elements) independent copies. This is illustrated in Figure 10 for a semi-circle successively split into 1,2 and 4 elements and the corresponding errors after optimizing the constrained disparity. Note that only the internal nodes are considered during optimization, reducing the problem dimension.

In Figure 11, we show convergence plots for a circle and a sphere arc for $p = 2, 3, 4$ and five mesh refinements. For the 2D case, as for the original disparity, we attain $2p$ order. For the 3D case, we obtain $\lfloor \frac{3}{2}(p-1) \rfloor + 2$ order. This means that the $p = 2$ leads

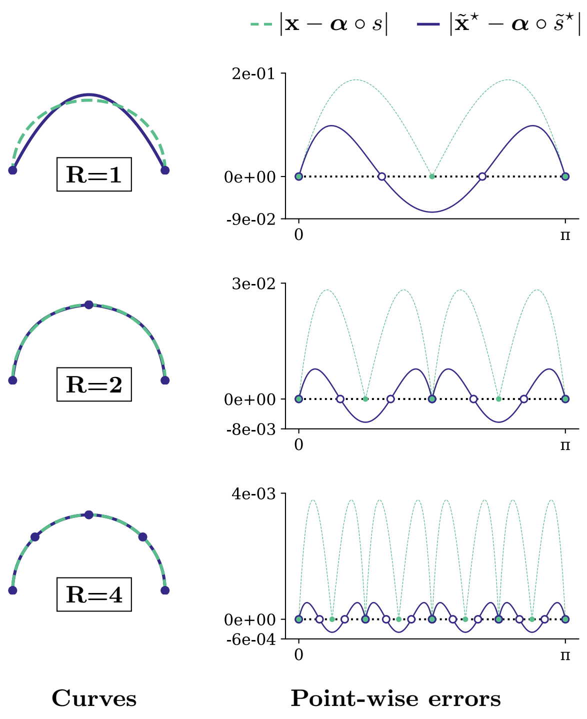


Figure 10: h-refinement (R=1,2, 4 elements) for $p = 2$ meshes. Left: optimized curves $\tilde{\mathbf{x}}^\star$ (solid deep blue) and $\boldsymbol{\alpha} \circ \tilde{s}^\star$ (dashed green). Right: error curves featuring roots (dots) and interface points (solid dots).

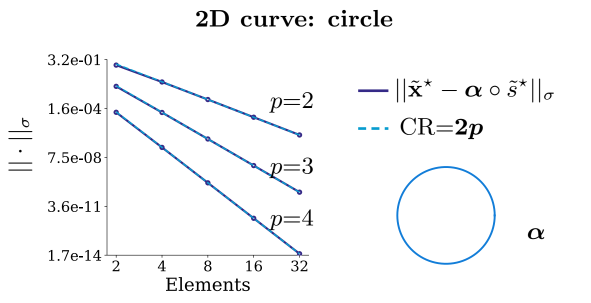
to the usual third order. However, the errors are lower than those resulting from direct interpolation. Later, when we look at the error over a single element, we discuss why we attain respectively, $2p$ and $\lfloor \frac{3}{2}(p-1) \rfloor + 2$.

Now, we use as the target geometry the edges of a CAD model. Figure 12 shows convergence plots for the top edge. Although the disparity values with fixed interfaces are slightly larger, the order of accuracy is the same as freeing interfaces and both cases significantly improve the initial approximation. In Figure 13, we show the point-wise errors after optimizing all edges for $p = 2, 3$ and 20 elements per edge. Concerning the initial approximation, both the constrained and original optimization decrease the errors with the same magnitude.

4.2 Planar curves: error profile for a single element.

Here, we study the local behaviour of the optimizer focusing on a single element. We use a semi-circle as the target geometry to make the plots clearer.

In Figure 14, we show the point-wise error plots when approximating a semi-circle with a single element for



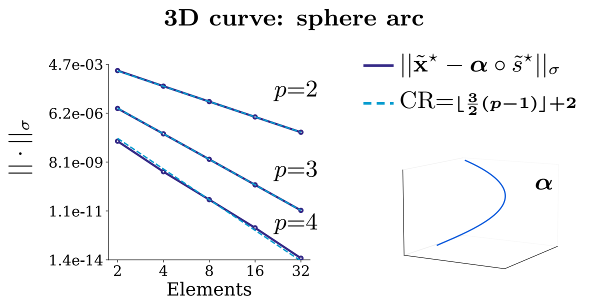


Figure 11: slopes (log-log) of the $||\cdot||_\sigma$ norm for several mesh refinements showing the convergence rates (CR) for a 2D (top) and 3D (bottom) after optimizing the constrained disparity $(\tilde{\mathbf{x}}^\star, \tilde{s}^\star)$.

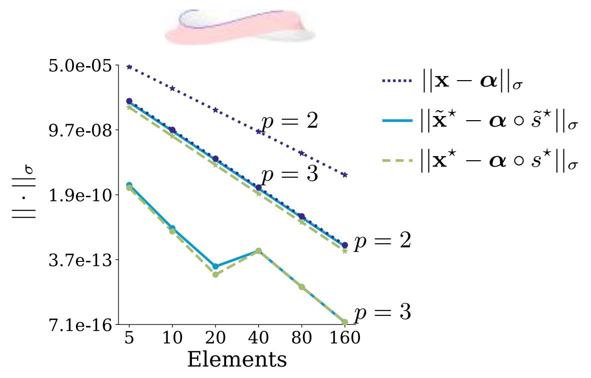


Figure 12: slopes (log-log) of the $||\cdot||_\sigma$ norm for several mesh refinements for the top curve of the CAD model (marked in blue) using direct interpolation (dotted dark blue) vs. optimizing the constrained (solid light blue) and the original (dashed green) disparities.

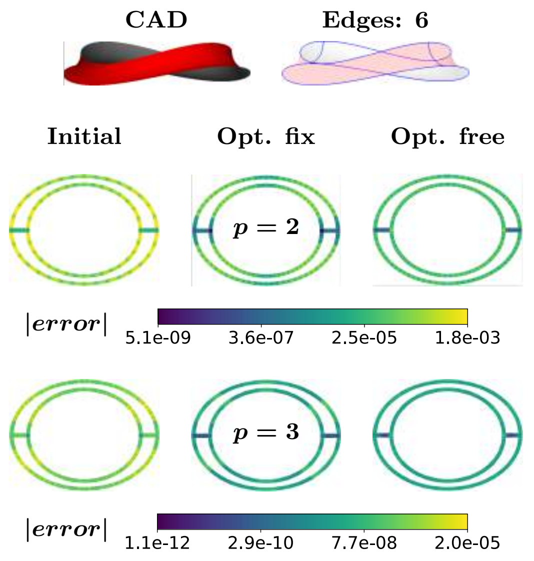


Figure 13: point-wise errors $|\mathbf{x} - \boldsymbol{\alpha} \circ s|$ (top view) approximating the edges of a CAD model with meshes made of 20 elements and $p = 2, 3$, respectively. From left to right: direct interpolation, optimizing the constrained (fix) and original disparities .

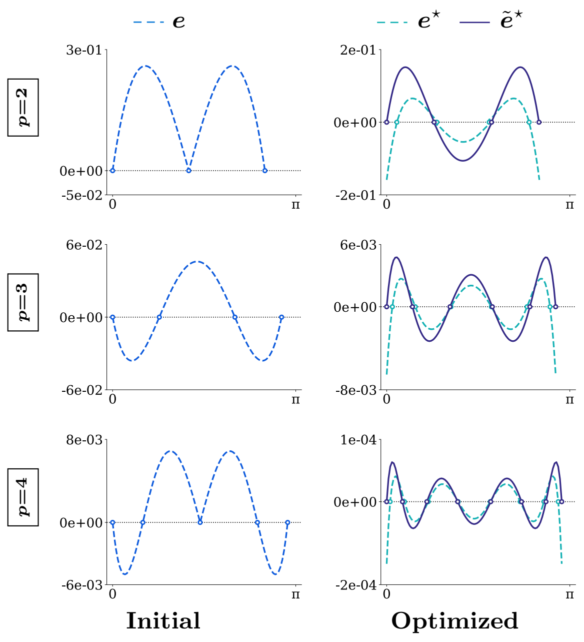


Figure 14: point-wise error plots $\boldsymbol{e} = |\mathbf{x} - \boldsymbol{\alpha} \circ s|$ approximating a semi-circle with a single element before (left) and after optimizing (right) the constrained (solid deep blue, $(\tilde{\mathbf{x}}^\star,\ \tilde{s}^\star)$) and the original disparity (dashed light blue, $(\mathbf{x}^\star,\ s^\star)$).

several polynomial degrees. The y-axis denotes the magnitude of the error $\boldsymbol{e} = \boldsymbol{e}(\xi) = |\mathbf{x}(\xi) - \boldsymbol{\alpha} \circ s(\xi)|$. The initial approximation (interpolation) is a polynomial of degree p and the error curve has the expected behaviour: $p + 1$ roots. The right plots show the results from both optimizations: fixed and free element interfaces. Notice that although the fixing interfaces produces slight larger errors, both solutions behave similarly: the curves have $2p$ roots (instead of $p + 1$).

In Section 2 (Figure 2) we showed with a circle that the point-wise errors align with the curve normal direction after optimization. Now we will discuss how this can be related to the disparity super-convergent property. Denote $\{\boldsymbol{t}, \boldsymbol{n}\}$ the curve tangent and normal vectors respectively. For planar curves, we can decompose the error $\boldsymbol{e} = \mathbf{x} - \boldsymbol{\alpha} \circ s$ along these directions:

$$\boldsymbol{e} = (\boldsymbol{e} \cdot \boldsymbol{t})\boldsymbol{t} + (\boldsymbol{e} \cdot \boldsymbol{n})\boldsymbol{n}.$$

The parametric mesh s uses polynomials of degree q so at each element, we have a total of $q+1$ degrees of freedom. On the other hand, since our physical mesh uses polynomials of degree p in $\mathbb{R}^2$, it has $2(p+1)$ degrees of freedom per element. Since our problem is constrained (fixed interfaces) we have a total of $q+1-2+2(p+1-2)$ degrees of freedom (per element). Hence, we can have $(2p-2)+(q-1)$ equations that will be optimizing the disparity.

During optimization, we impose zero tangent error (weakly) in $q-1$ equations. If we assume that solving the nonlinear equations behaves similar to interpolation, we would expect at least $q+1$ roots in the error function. Recall that the end-points are fixed, hence why we go from $q-1$ to $q+1$. This is shown in Figure 12 for the $q = 2p-1$ case: the optimized tangent error has 5 and 7 roots for $p = 2, 3$, respectively.

The $2p-2$ remaining equations are used to impose the total error equal to zero. Assume we can make the tangent error as small as desired by increasing q. At the optimum, we can think of these $2p-2$ equations essentially imposing zero normal error (weakly). With the same reasoning as for s, we expect $2p$ roots along the normal component. The $+2$ corresponds to the interfaces which are interpolation points. This can be appreciated in Figure 15 looking at the plots from the optimized case. Also, provided $q > 2p-1$, the normal error dictates the overall accuracy. Note that in both $q = 2p-1$ and $q = 10$, the normal error is larger.

4.3 Discussion for 3D curves

We have just discussed the 2D case and how the optimal error behaves in terms of the tangent and normal component. We will now extend our results to the 3D case. In this case, the error decomposition becomes:

$$\boldsymbol{e} = (\boldsymbol{e} \cdot \boldsymbol{t})\boldsymbol{t} + (\boldsymbol{e} \cdot \boldsymbol{n})\boldsymbol{n} + (\boldsymbol{e} \cdot \boldsymbol{b})\boldsymbol{b}, \qquad (32)$$

where $\boldsymbol{e} = \mathbf{x} - \boldsymbol{\alpha} \circ s$ and $\{\boldsymbol{t}, \boldsymbol{n}, \boldsymbol{b}\}$ are the curve tangent, normal and binormal vectors, respectively. Our physical mesh uses polynomials of degree p in 3D space with fixed end-points. So, at each element, we have $3(p-1)$ degrees of freedom. As for the 2D case, we impose in $q-1$ equations zero tangent error (weakly). The rest $3(p-1)$ equations impose total zero error (weakly).

As before, we assume that the tangent error decreases as q increases. At the optimum, the combined solution

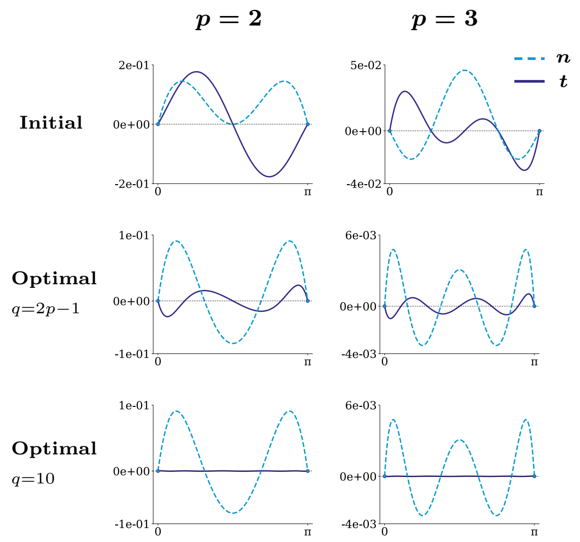


Figure 15: tangent ($\boldsymbol{t}$) and normal ($\boldsymbol{n}$) error components approximating a semi-circle with one element before and after optimizing the internal nodes.

implies that we have $3(p-1)$ equations imposing zero along both the normal and binormal components. In analogy with the 2D discussion, we now expect at least $\lfloor \frac{3}{2}(p-1) \rfloor$ interpolation points along each component: $\{\boldsymbol{n}, \boldsymbol{b}\}$. Since end-points interpolate the curve, it gives $(\lfloor \frac{3}{2}(p-1) \rfloor + 2)$ roots per component.

In Figure 16, we show the error plots before and after optimizing the constrained disparity approximating a sphere arc with a single element. As for the 2D case, as $\tilde{\mathbf{x}}$ follows the image of $\boldsymbol{\alpha} \circ \tilde{s}$, $\tilde{s}$ minimizes the tangent error. Notice that when s is of degree $q = 10$, the tangent error is negligible compared to the other two components. Also, observe how we obtain both along the normal and binormal directions: $5 = \lfloor \frac{3}{2}(3-1) \rfloor + 2$ roots for $p = 3$ and at least $6 = \lfloor \frac{3}{2}(4-1) \rfloor + 2$ for $p = 4$.

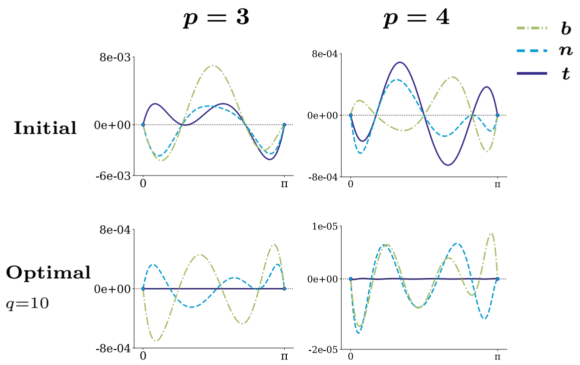


Figure 16: tangent, normal and binormal $\{\boldsymbol{t}, \boldsymbol{n}, \boldsymbol{b}\}$ errors approximating a sphere arc with one element before and after minimizing the constrained disparity.

4.4 Mesh initialization

Here, we focus on the element partition. In Figure 17 (top), we show a $p = 2$ mesh approximating a spline-based NACA curve with ESP [12] using direct interpolation. In this case, the elements are equi-distributed along the curve parametric space. Notice that the leading edge is poorly resolved. We can improve the element partition optimizing the original disparity using $p = q = 1$ meshes. Then, we save the partition in $s^\star$ and p-refine both meshes: s and $\mathbf{x}$. The result is shown at the bottom images from Figure 17. Notice that now the leading edge is well approximated. Alternatively, we could have obtained the initial partition performing an arc-length based optimization [16].

In Figure 18, we study the accuracy of the initial meshes (before optimization) for $p = 1, 2, 3$ when approximating the NACA curve. We compare sampling directly along the parametric space (s_1 equi-par) with the pre-processing step: optimizing the linear meshes (s_1 opt. all). Notice that the errors significantly improve for the latter one. We also show the case where only the coarser mesh is optimized (s_1 opt. first). Then, the finer meshes are obtained splitting directly each element in two. In this case, the errors are comparable to optimizing at every refinement. This approach can be used to save computational time.

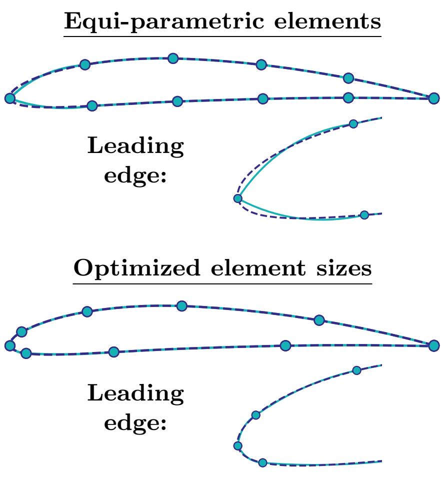


Figure 17: interpolating a NACA curve with 10 elements ($p = 2$) using an equi-parametric distribution (top) vs. the proposed pre-processing step (bottom).

5. NUMERICAL RESULTS

Here, we perform several numerical experiments for the solver performance in terms of the number of iterations. We start studying the impact of the line search choice. Finally, we discuss the trade-offs between errors and iterations comparing the constrained disparity to the original formulation.

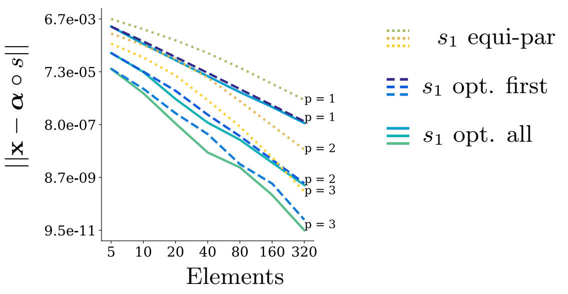


Figure 18: log-log error plots of the initial meshes approximating a NACA curve for several refinements comparing equi-parametric elements (dotted lines) to optimizing only the coarser mesh (dashed lines) and optimizing at every h-refinement (solid lines).

5.1 Zhang-Hager vs. Armijo line search

Here, we compare the performance between Armijo and the average line search. In Figure 19, we show the error contours for a CAD body to highlight that both Armijo and Zhang search produce exactly the same solution. In Figure 20, we compare iterations for three different curves and polynomial degrees. Looking at the number of nonlinear iterations, Zhang-Hager line search is systematically faster than Armijo. On average, it is 76% faster (in terms of iterations).

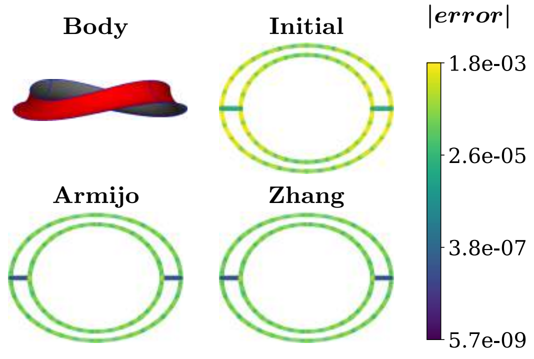


Figure 19: point-wise errors (top view) at the six edges of a CAD body before and after optimizing the original disparity using Armijo vs. Zhang-Hager rule. Each edge consists of eight $p = 2$ elements.

5.2 Constrained optimization: errors vs. iterations

Here, we focus on more complex bodies made of several surfaces and study the trade-offs from fixing the interfaces. We omit straight edges since they can be represented exactly with linear elements and set a stop criterion of $|\nabla(E)| < 10^{-12}$.

In Figure 21, we show a CAD model made of 54 edges out of which 36 are curved. We use meshes consist-

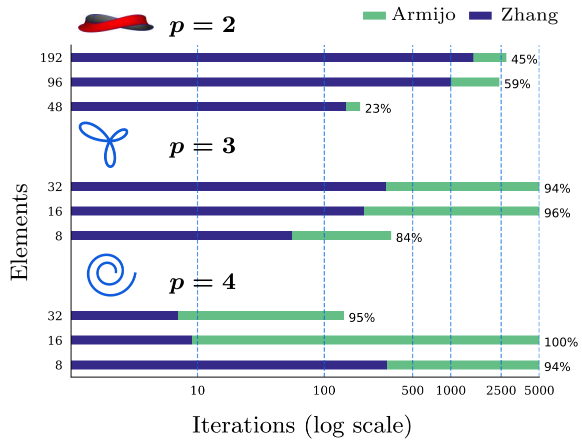


Figure 20: Armijo vs. Zhang line searches optimizing curves (analytic and CAD) for degrees $p = 2, 3, 4$. The % in the bars indicate relative lesser iterations.

ing of $p = 3$, $q = 9$ elements and compare the results from computing the constrained and original disparities. The optimal pair $(\mathbf{x}^\star, s^\star)$ (free interfaces) produces lower errors, although in some cases, the constrained solution $(\tilde{\mathbf{x}}^\star, \tilde{s}^\star)$ leads to a lower disparity value. This is because, in that case, the optimization of the original disparity converged to a local minimum with a higher disparity value. Optimizing the disparity with fixed interfaces took a maximum of 23 iterations whereas the unconstrained problem took a maximum of 808. On average, the constrained problem converges in 9 iterations, and the original in 167.

Finally, in Figure 22 we present an aircraft model consisting of 102 faces and 238 edges. Since the model is symmetric, we study only its left half. This gives 51 curves that we approximate with $p = 2$ meshes. In this case, optimizing the constrained disparity took a maximum of 43 iterations whereas optimizing the original disparity went, in many cases, beyond 500 iterations. In both cases, we have used the Zhang-Hager line search. On average, the constrained problem takes 4 iterations to converge compared to 387 taken by the unconstrained problem, becoming 87% faster.

6. CONCLUSIONS

We have developed a robust solver designed to minimize the disparity measure. We have introduced a log barrier penalty term to avoid curve tangling. The Zhang-Hager average line search is less restrictive, producing the same results as the Armijo rule in significantly fewer iterations. On average, it reduces the number of iterations by 76%.

The original disparity (free interfaces) gives optimal errors. On the other hand, the constrained disparity (fixed interfaces) is sub-optimal in terms of the error but still yields super-convergence. We have numerically shown how both disparities are $2p$ super-convergent for 2D curves and $\lfloor \frac{3}{2}(p-1) \rfloor + 2$ for curves in 3D space.

Initially, solving the original disparity with the Armijo rule took, on average, around 2000 nonlinear iterations. Our experiments for fixed element interfaces show that optimizing the disparity with the Zhang-Hager line-search, produces a residual less than 10^{-12} in less than 10 iterations. This corresponds to a reduction factor of 100 when compared to the original optimization of the problem. In the future, we will extend this methodology to surface mesh generation.

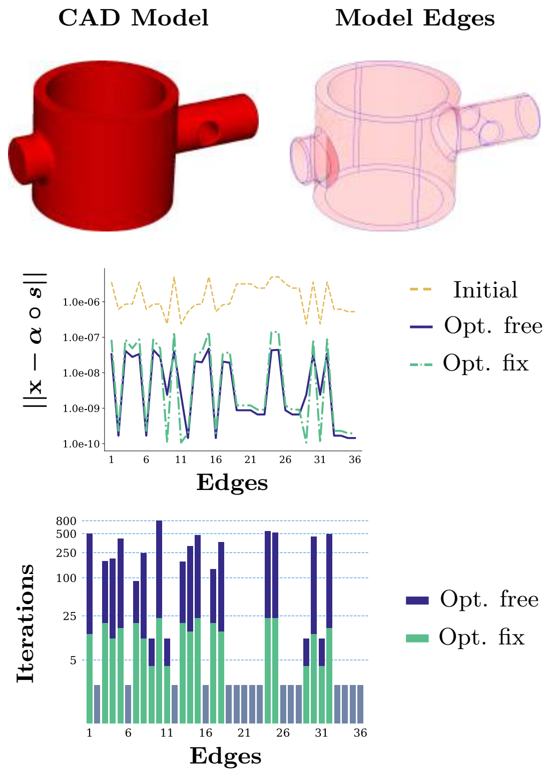


Figure 21: fix vs. free interfaces optimization for $p = 3$ meshes approximating all the curved edges of a CAD model. Total elements: 560.

7. ACKNOWLEDGEMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 893378 as well as the European Research Council (ERC) grant agreement No 715546.

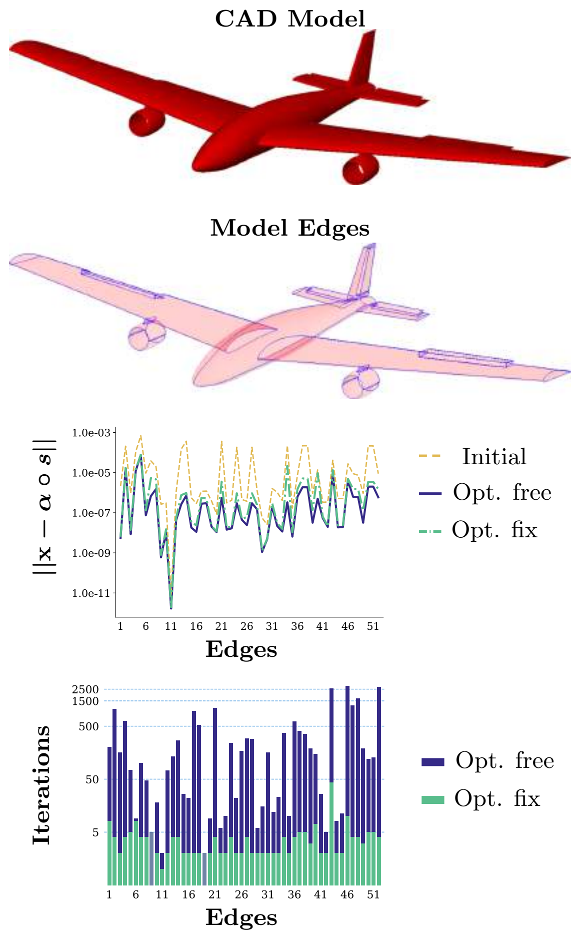


Figure 22: fix vs. free element interfaces optimization for $p = 2$ meshes approximating all the curved edges of an aircraft model. Total elements: 1375.

References

[1] Slotnick J.P., Khodadoust A., Alonso J., Darmofal D., Gropp W., Lurie E., Mavriplis D.J. "CFD vision 2030 study: a path to revolutionary computational aerosciences." Tech. Rep. NASA/CR-2014-218178, 2014

[2] Alt H., Godau M. "Computing the Fréchet distance between two polygonal curves." *Int. J. Comput. Geom. Appl.*, vol. 5, 75–91, 1995

[3] Remacle J., Lambrechts J., Geuzaine C., Toulorge T. "Optimizing the geometrical accuracy of 2D curvilinear meshes." *Procedia Eng.*, vol. 82, 228–239, 2014. 23rd International Meshing Roundtable

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

[5] Ruiz-Girons E., Sarrate J., Roca X. "Defining an $\mathcal{L}_2$-disparity measure to check and improve the geometric accuracy of non-interpolating curved high-order meshes." *Procedia Eng.*, vol. 124, 122–134, 2015. 24th International Meshing Roundtable

[6] Ruiz-Girons E., Sarrate J., Roca X. "Generation of curved high-order meshes with optimal quality and geometric accuracy." *Procedia Eng.*, vol. 163, 315–327, 2016. 25th International Meshing Roundtable

[7] Ruiz-Girons E., Sarrate J., Roca X. "Measuring and improving the geometric accuracy of piece-wise polynomial boundary meshes." *J. Comput. Phys.on*, vol. 443, 110500, 2021

[8] Dai Y.H. "On the nonmonotone line search." *Journal of Optimization Theory and Applications*, vol. 112, no. 2, 315–330, 2002

[9] Grippo L., Lampariello F., Lucidi S. "A nonmonotone line search technique for Newtons method." *SIAM Journal on Numerical Analysis*, vol. 23, no. 4, 707–716, 1986

[10] Zhang H., Hager W.W. "A nonmonotone line search technique and its application to unconstrained optimization." *SIAM Journal on Optimization*, vol. 14, no. 4, 1043–1056, 2004

[11] Toint P.L. "An Assessment of nonmonotone linesearch techniques for unconstrained optimization." *SIAM Journal on Scientific Computing*, vol. 17, no. 3, 725–739, 1996

[12] Haimes R., Dannenhoffer J. "The engineering sketch pad: A solid-modeling, feature-based, web-enabled system for building parametric geometry." Sep. 2013. 21st AIAA Computational Fluid Dynamics Conference

[13] Garimella R.V., Shashkov M.J., Knupp P.M. "Triangular and quadrilateral surface mesh quality optimization using local parametrization." *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 9, 913–928, 2004

[14] Dobrev V., Knupp P., Kolev T., Mittal K., Tomov V. "The Target-Matrix Optimization Paradigm for High-Order Meshes." *SIAM Journal on Scientific Computing*, vol. 41, no. 1, B50–B68, 2019

[15] Nocedal J., Wright S. *Numerical Optimization.* Springer Science & Business Media, 2006

[16] McLaurin D., Shontz S.M. "Automated edge grid generation based on arc-length optimization." *Proceedings of the 22nd International Meshing Roundtable*, pp. 385–403. Springer International Publishing, 2014