2. Data fitting schemes with hierarchical splines

Constructing continuous and flexible geometric models that are easy to shape and manipulate is a fundamental requirement in many applications. For example, they can serve for visualization and direct measurements within a design phase or a manufacturing process, as well as for simulations, i. e. performing numerical computations directly on the model or its components.

In this Chapter, we consider the problem of constructing a parametric (spline) model $\boldsymbol{s}\colon \Omega \to \mathbb{R}^N$ on a domain $\Omega \subseteq \mathbb{R}^D$ in any dimension $D \in \mathbb{N}_{>0}$ from pointwise data and parametric values

$$\mathcal{U} \times \mathcal{P} \coloneqq \left\{(\boldsymbol{u}_i, \boldsymbol{p}_i) \in \mathbb{R}^D \times \mathbb{R}^N \,|\, i = 1, \dots, m\right\} \tag{23}$$

where,

$$\mathcal{P} \coloneqq \left\{\boldsymbol{p}_i \in \mathbb{R}^N \,|\, i = 1, \dots, m\right\}$$

for $N \in \mathbb{N}_{>0}$ are the point observations as in (1), at the parametric sites

$$\mathcal{U} \coloneqq \left\{\boldsymbol{u}_i \in \Omega \subset \mathbb{R}^D \,|\, i = 1, \dots, m\right\},$$

as in (3). In CAGD applications, if $N = 2$ the data points lay on a plane, whereas if $N = 3$ the data points belong to the physical space. Moreover, we require the spline model $\boldsymbol{s}$ to approximate the data $\mathcal{P}$ within a certain tolerance $\epsilon \in \mathbb{R}_{>0}$, in the sense that

$$\operatorname{dist}(\boldsymbol{s}_i, \boldsymbol{p}_i) \leq \epsilon \text{ for each } i = 1, \dots, m,$$

where $\operatorname{dist}(\cdot, \cdot)$ is a suitable distance metric.

By virtue of their properties, such as locality, non-negativity and partition of unity and their suitability for adaptive refinement, THB-splines are a desirable tool for building flexible geometric representations, see Section 1. Therefore, the THB-spline fitting problem can be stated in the following way, see also (2). Given a point cloud as in (23) and an error tolerance $\epsilon > 0$, find a THB-spline model $\boldsymbol{s} : \Omega \subseteq \mathbb{R}^D \to \mathbb{R}^N$, so that

$$\operatorname{dist}(\boldsymbol{s}(\boldsymbol{u}_i), \boldsymbol{p}_i) \leq \epsilon, \quad \text{with } \boldsymbol{u}_i \in \Omega, \quad \text{for each} \quad i = 1, \dots, m. \tag{24}$$

The design of a THB-spline approximation depends on the identification of the hierarchical spline space $\mathcal{H}$, defined in Definition 4, the multivariate

Sofia Imperatore, Eindhoven University of Technology, the Netherlands, s.imperatore@tue.nl, 0009-0003-9116-9978

Referee List (DOI 10.36253/fup_referee_list)
FUP Best Practice in Scholarly Publishing (DOI 10.36253/fup_best_practice)

Book References DOI 10.36253/979-12-215-1002-7.references

degree $\boldsymbol{d} \in \mathbb{N}^D_{>0}$, the hierarchical mesh, which defines a tessellation $T(\Omega)$, see also Remark 8, and finally also the computation of the control points.

In this Chapter, we exploit two main approximation methods to define the THB-spline model, i. e. the *reWeighted Least Squares (rWLS)* approximation and the *Quasi-Interpolation (QI)* schemes. In particular, we first focus in Section1 on two very well established classical methods: interpolation and weighted least squares, and extend the latter as an adaptive rWLS fitting scheme. Subsequently, in Section 2, we present a hierarchical QI scheme with THB-splines. The present Chapter is mostly based on [57, 17].

2.1. Interpolation and weighted least squares

We here focus on two established classical methods: interpolation and weighted least squares; see e. g., [31]. In particular, weighted least squares approximations can be considered a more general instance of ordinary least squares methods, where fixed real positive values are associated with the observations to incorporate different weightings in the least squares scheme; see e. g., [152].

Although interpolation and least squares methods are often regarded as complementary techniques, they share a strong connection. In this Section, we propose a general formulation of the weighted least squares approximant as a convex combination of suitable interpolants, for any finite-dimensional function space consisting of real-valued functions defined on a domain $\Omega \subseteq \mathbb{R}^D$ and address its consequences. As a special case, our formulation includes the vector space of polynomials up to a certain degree, for which a relation between weighted polynomial least squares and interpolation has been discussed in [22].

In addition, we also focus on the choice of the weights. Our main aim is to update these weights in the spirit of the Iterative Reweighted Least Squares (IRLS) method [125, 7], usually considered to smoothen the reconstruction and limit the influence of outliers in the input data. However, the main difference between our approach to the IRLS method is that our updates allow us to preserve the sharp features of the final model and are also suitable for adaptive approximations. Our numerical experiments show the performance of the proposed method within the spline framework, from curve to surface fitting, including adaptive spline constructions.

Interpolation

Let $\Omega \subseteq \mathbb{R}^D$ and consider a set of given observations as in (23). Moreover, let V be a vector space of real-valued functions defined on Ω, with finite dimension $n+1 = \dim V$ and generated by a basis $\mathcal{B} = \{\beta_0, \dots, \beta_n\}$, with basis functions $\beta_j : \Omega \to \mathbb{R}$, i. e. $V = \operatorname{span}\{\mathcal{B}\}$.

The *interpolation problem* aims to find an element $\boldsymbol{s} \in V$ such that $\boldsymbol{s}(\boldsymbol{u}_i) = \boldsymbol{p}_i$ holds for each $i = 1, \dots, m$. If a solution $\boldsymbol{s}$ exists, it can be expressed as

$$\boldsymbol{s}(\boldsymbol{x}) = \sum_{j=0}^{n} \boldsymbol{c}_j \beta_j, \text{ for } \boldsymbol{x} \in \Omega, \tag{25}$$

where the coefficients $\boldsymbol{c}_j = (c_j^1, \ldots, c_j^N) \in \mathbb{R}^N$ can be determined by solving a linear system for each component $k = 1, \ldots, N$, of the form $B\boldsymbol{c}^k = \boldsymbol{p}^k$, where

$$B = B(\mathcal{U}) = \begin{pmatrix} \beta_0(\boldsymbol{u}_1) & \ldots & \beta_n(\boldsymbol{u}_1) \\ \vdots & \ddots & \vdots \\ \beta_0(\boldsymbol{u}_m) & \ldots & \beta_n(\boldsymbol{u}_m) \end{pmatrix} \in \mathbb{R}^{m\times(n+1)} \tag{26}$$

is the $m \times (n+1)$ collocation matrix defined by the basis $\mathcal{B}$ and the parametric sites $\mathcal{U}$,

$$\boldsymbol{c}^k = (c_0^k, \ldots, c_n^k)^\top \in \mathbb{R}^{n+1} \text{ and } \boldsymbol{p}^k = (p_1^k, \ldots, p_m^k)^\top \in \mathbb{R}^m.$$

The solution $\boldsymbol{s} \in V$ is unique, if and only if B is invertible, hence $n+1 = m$ is a necessary condition. However, even if the interpolant $\boldsymbol{s} \in V$ exists, it is well known that it can be affected by poor approximation quality. Several factors can contribute to this, such as the nature of the data (e. g., their quantity or distribution) and the intrinsic properties of the function space V (e. g., the presence of oscillatory behaviour [139]).

Weighted least squares

As an alternative to interpolation, a common approach is to define the approximant $\boldsymbol{s} \in V$ as the solution to a *weighted least squares problem* when $n+1 \leq m$. In this context, we assign a strictly positive *weight* value $\omega_i \in \mathbb{R}_{>0}$ to each $\boldsymbol{u}_i \in \Omega$ for $i = 1, \ldots, m$. Let $W \in \mathbb{R}^{m\times m}$ be the associated diagonal matrix with the i-th diagonal entry given by ω_i. Then, the weighted least squares solution $\boldsymbol{s} \in V$ is obtained by solving the minimization problem

$$\min_{\boldsymbol{v}\in V} \frac{1}{2} \sum_{i=1}^{m} \omega_i \left\| \boldsymbol{v}(\boldsymbol{u}_i) - \boldsymbol{p}_i \right\|_2^2 . \tag{27}$$

By expanding the solution $\boldsymbol{s} \in V$ of (27) as (25) and letting $\boldsymbol{c} = (\boldsymbol{c}_0, \ldots, \boldsymbol{c}_n)^\top \in \mathbb{R}^{(n+1)\times D}$ be the coefficient matrix, we can re-write problem (27) as

$$\min_{\boldsymbol{c}\in\mathbb{R}^{(n+1)\times N}} \| W^{\frac{1}{2}} B\boldsymbol{c} - W^{\frac{1}{2}} \boldsymbol{p} \|_2^2, \tag{28}$$

where, $\boldsymbol{p} = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m)^\top \in \mathbb{R}^{m\times N}$,

$$W = \operatorname{diag}\{\omega_1, \ldots, \omega_m\} = \begin{pmatrix} \omega_1 & 0 & \ldots & 0 \\ 0 & \omega_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \omega_m \end{pmatrix} \in \mathbb{R}^{m\times m}$$

and $W^{\frac{1}{2}}$ denotes the element-wise evaluation of the square root of W. The minimization in (28) entails solving a system of linear equations for each column $\boldsymbol{c}^k = (c_0^k, \dots, c_n^k)$, $k = 1, \dots, N$, of $\boldsymbol{c}$. More precisely, for each $k = 1, \dots, N$, $\boldsymbol{c}^k$ is obtained from the normal equation

$$B^\top W B \boldsymbol{c}^k = B^\top W \boldsymbol{f}^k. \tag{29}$$

Weighted least squares via interpolation

In this part, to alleviate the notation, we will assume that $N = 1$. Therefore, we avoid the superscript k introduced above and reduce the bold writing according to the dimension. Nevertheless, all the results apply in the more general case of $N \geq 1$. Consider $\Omega \subseteq \mathbb{R}^D$,

$$\mathcal{U} \times \mathcal{P} = \left\{ (\boldsymbol{u}_i, p_i) \in \mathbb{R}^D \times \mathbb{R} \,|\, i = 1, \dots, m \right\}$$

a set of given observations with $\boldsymbol{u}_i \in \Omega$ and $p_i \in \mathbb{R}$ and finally $V = \operatorname{span}\{\mathcal{B}\}$ a vector space of dimension $n+1$, with basis $\mathcal{B} = \{\beta_0, \dots, \beta_n\}$ and functions $\beta_j : \Omega \to \mathbb{R}$, for $j = 0, \dots, n$.

We define

$$\mathcal{K}_{n+1} = \{K \subseteq \{1, \dots, m\} \colon \#(K) = n + 1\},$$

the set of all subsets of $\{1, \dots, m\}$ with cardinality $n + 1$, equal to the dimension of the vector space V. For each $K \in \mathcal{K}_{n+1}$, there exists $k_i \in \{1, \dots, m\}$ for $i = 0, \dots, n$, such that $K = \{k_0, \dots, k_n\}$. We denote by $s_K \in V$ the interpolant of p_{k_i} at points $\boldsymbol{u}_{k_i}$ for $i = 0, \dots, n$. For such K, we define

$$B_K = \begin{pmatrix} \beta_0(\boldsymbol{u}_{k_0}) & \dots & \beta_n(\boldsymbol{u}_{k_0}) \\ \vdots & \ddots & \vdots \\ \beta_0(\boldsymbol{u}_{k_n}) & \dots & \beta_n(\boldsymbol{u}_{k_n}) \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)} \tag{30}$$

which is called the *collocation matrix* at the points $\boldsymbol{u}_{k_i}$ with respect to the whole basis $\mathcal{B}$ of V. For $K \in \mathcal{K}_{n+1}$ and positive weights $\omega_1, \dots, \omega_m \in \mathbb{R}$, we write

$$\omega_K = \prod_{i \in K} \omega_i. \tag{31}$$

Following the notation in [22], we define $\lambda_K = \omega_K \left|B_K\right|^2$, where $\left|B_K\right|$ denotes the determinant of B_K. Finally, we define $\mathcal{K}^\star_{n+1} \subset \mathcal{K}_{n+1}$ as the set of all subsets of $\{1, \dots, m\}$ of cardinality $n + 1$ with $|B_K| \neq 0$. In other words,

$$\mathcal{K}^\star_{n+1} = \{K \subseteq \{1, \dots, m\} \colon \#(K) = n + 1, \text{ and } |B_K| \neq 0\}.$$

Theorem 1. *The weighted least squares approximant $s \in V$ of the set of points $\left\{(\boldsymbol{u}_i, p_i)\right\}_{i=1}^m$ is the weighted sum of the interpolants $s_K \in V$ for $K \in \mathcal{K}_{n+1}^\star$ which interpolate the points $(\boldsymbol{u}_i, p_i)$ for $i \in K$, i. e.*

$$s(\boldsymbol{x}) = \frac{\sum_{K \in \mathcal{K}_{n+1}^\star} \lambda_K s_K(\boldsymbol{x})}{\sum_{K \in \mathcal{K}_{n+1}^\star} \lambda_K}, \quad \forall \boldsymbol{x} \in \mathbb{R}^D. \tag{32}$$

Dimostrazione. By hypothesis, $s \in V$ minimizes (27). Let $\mathcal{B}$ a basis for V and write s as in (25), for some coefficients $c_j \in \mathbb{R}$, $j = 0, \ldots, n$. Hence, we set

$$\boldsymbol{c} = (c_0, \ldots, c_n)^\top \quad \text{and} \quad \boldsymbol{p} = (p_1, \ldots, p_m)^\top,$$

and write the normal equation (29) as $A\boldsymbol{c} = \boldsymbol{b}$ with

$$A = B^\top W B = \begin{pmatrix} \omega_1 \beta_0(\boldsymbol{u}_1) & \ldots & \omega_m \beta_0(\boldsymbol{u}_m) \\ \vdots & \ddots & \vdots \\ \omega_1 \beta_n(\boldsymbol{u}_1) & \ldots & \omega_m \beta_n(\boldsymbol{u}_m) \end{pmatrix} \begin{pmatrix} \beta_0(\boldsymbol{u}_1) & \ldots & \beta_n(\boldsymbol{u}_1) \\ \vdots & \ddots & \vdots \\ \beta_0(\boldsymbol{u}_m) & \ldots & \beta_n(\boldsymbol{u}_m) \end{pmatrix},$$

$$\boldsymbol{b} = B^\top W \boldsymbol{p} = \left(\sum_{i=1}^m \omega_i \beta_0(\boldsymbol{u}_i) p_i, \quad \ldots, \quad \sum_{i=1}^m \omega_i \beta_n(\boldsymbol{u}_i) p_i \right)^\top.$$

By the Cauchy-Binet theorem [18, Section 4.6],

$$|A| = \sum_{K \in \mathcal{K}_{n+1}} \omega_K \left| B_K \right|^2$$

and by Cramer's rule, we obtain

$$c_j = |A_j| \Big/ |A|,$$

for each $j = 0, \ldots, n$, where A_j is obtained from A replacing its j-th column with $\boldsymbol{b}$ for $j = 0, \ldots, n$, namely, $A_j = B^\top W B_j$ with

$$B_j = \begin{pmatrix} \beta_0(\boldsymbol{u}_1) & \ldots & \beta_{j-1}(\boldsymbol{u}_1) & p_1 & \beta_{j+1}(\boldsymbol{u}_1) & \ldots & \beta_n(\boldsymbol{u}_1) \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \beta_0(\boldsymbol{u}_m) & \ldots & \beta_{j-1}(\boldsymbol{u}_m) & p_m & \beta_{j+1}(\boldsymbol{u}_m) & \ldots & \beta_n(\boldsymbol{u}_m) \end{pmatrix} \in \mathbb{R}^{m \times (n+1)}.$$

Again from the Cauchy-Binet theorem, we obtain

$$|A_j| = \sum_{K \in \mathcal{K}_{n+1}} \omega_K \left| B_K \right| \left| B_{j,K} \right|$$

for $j = 0, \ldots, n$, where $B_{j,K}$ is obtained from B_K by replacing its j-th column with $(p_1, \ldots, p_m)^\top$ for $j = 0, \ldots, n$. If $K \in \mathcal{K}_{n+1}^\star$, the interpolation conditions satisfy

$$B_K \boldsymbol{c}_K = \boldsymbol{p}_K,$$

where $\boldsymbol{p}_K = (p_{k_1}, \ldots, p_{k_n})^\top$ with $K = \{k_0, \ldots, k_n\}$. The solution of the linear problem $\boldsymbol{c}_K = (c_{0,K}, \ldots, c_{n,K})^\top$ is unique if $|B_K| \neq 0$, and can be obtained by Cramer's rule as follows,

$$c_{j,K} = |B_{j,K}| \Big/ |B_K| .$$

The associated interpolant $s_K(\boldsymbol{x})$ for $K \in \mathcal{K}^\star_{n+1}$ can be written as

$$s_K(\boldsymbol{x}) = \sum_{j=0}^{n} c_{j,K} \beta_j(\boldsymbol{x}),$$

hence by letting $\lambda_K = \omega_K |B_K|^2$, it holds

$$\begin{aligned} s(\boldsymbol{x}) = \sum_{j=0}^{n} c_j \beta_j(\boldsymbol{x}) = \sum_{j=0}^{n} \frac{|A_j| \, \beta_j(\boldsymbol{x})}{|A|} = \sum_{j=0}^{n} \frac{\sum_{K \in \mathcal{K}_{n+1}} \omega_K |B_K| \, |B_{j,K}| \, \beta_j(\boldsymbol{x})}{\sum_{K \in \mathcal{K}_{n+1}} \omega_K |B_K|^2} \\ = \frac{\sum_{K \in \mathcal{P}^\star_n} \omega_K |B_K|^2 \sum_{j=1}^{n} c_{j,K} \beta_j(\boldsymbol{x})}{\sum_{K \in \mathcal{P}^\star_n} \omega_K |B_K|^2} = \frac{\sum_{K \in \mathcal{P}^\star_n} \lambda_K s_K(\boldsymbol{x})}{\sum_{K \in \mathcal{P}^\star_n} \lambda_K}. \end{aligned}$$

□

Remark 11. *Note that Theorem 1 is applicable to any finite-dimensional (multivariate) vector space. This encompasses various function spaces, such as the space of polynomials up to a specific degree, spline spaces with fixed degree and order, spline spaces with varying locations of knot lines, or any other real function space of finite dimension equipped with a basis.*

Remark 12. *In general, we have* $\#\left(\mathcal{K}^\star_{n+1}\right) \leq \#\left(\mathcal{K}_{n+1}\right) = \binom{m}{n+1}$*, due to the non-nullity condition on the determinants* $|B_K|$*, which guarantees the existence of the interpolants. In particular, in the case of splines, the condition* $|B_K| \neq 0$ *is equivalent to the Schoenberg-Whitney nesting condition [11]. In addition, the value of* λ_K *depends on the location of the knot lines.*

Remarks on weighted least squares approximation

For the univariate polynomial case, results on the upper and lower pointwise error bounds of the approximant derivatives up to a certain order as well as on the influence of the weights have been presented in [22, Section 3] and [22, Section 4], respectively. We extend these results to any vector space V of finite dimension $n+1$. This generalization extends the derived conclusions of [22] in a broader setting, beyond the polynomial scenario.

Let $r \in \mathbb{N}$ be given, if the r-th order derivative of $s \in V$ exists, it can be expressed as the weighted average of the r-th order derivatives of the interpolants, i. e.

$$\partial^\alpha s(\boldsymbol{x}) = \frac{\sum_{K \in \mathcal{K}^\star_{n+1}} \lambda_K \partial^\alpha s_K(\boldsymbol{x})}{\sum_{K \in \mathcal{K}^\star_{n+1}} \lambda_K}, \tag{33}$$

where $\alpha = (\alpha_1, \ldots, \alpha_D)$ is a suitable multi-index, with $\sum_{j=1}^{D} \alpha_j = r \geq 0$.

In addition, pointwise upper and lower bounds for the value of the r-th order derivative of $s(\boldsymbol{x})$ can be obtained from (33), i. e.

$$\min_{K \in \mathcal{K}^{\star}_{n+1}} \partial^{\alpha} s_K(\boldsymbol{x}) \leq \partial^{\alpha} s(\boldsymbol{x}) \leq \max_{K \in \mathcal{K}^{\star}_{n+1}} \partial^{\alpha} s_K(\boldsymbol{x}).$$

Likewise, the pointwise approximation error shares the same weighted average, given by

$$f(\boldsymbol{x}) - s(\boldsymbol{x}) = \frac{\sum_{K \in \mathcal{K}^{\star}_{n+1}} \lambda_K \left(f(\boldsymbol{x}) - s_K(\boldsymbol{x}) \right)}{\sum_{K \in \mathcal{K}^{\star}_{n+1}} \lambda_K}.$$

Furthermore, the following consequences on the influence of the weights ω_i for $i = 1, \ldots, m$ and derivative estimations can be inferred.

Remark 13. *Let $I \subseteq \{1, \ldots, m\}$ be of cardinality $\#(I) = r$, with $1 \leq r \leq n+1$. Define*

$$s_I(\boldsymbol{x}) = \lim_{\substack{\omega_i \to +\infty \\ \forall i \in I}} s(\boldsymbol{x})$$

and

$$\mathcal{K}^{\star}_{n+1} \setminus I = \{ K \subseteq \{1, \ldots, m\} \setminus I \mid \#(K) = n+1, \text{ and } |B_K| \neq 0 \}.$$

If $r = n+1$, then $s_I(\boldsymbol{x}) = s_I(\boldsymbol{x})$, represents the interpolant of the data points indexed by I. If $1 \leq r < n+1$, then

$$s_I(\boldsymbol{x}) = \frac{\sum_{K \in \mathcal{K}^{\star}_{n+1-r} \setminus \{I\}} \lambda_{I,K} s_{I \cup K}(\boldsymbol{x})}{\sum_{K \in \mathcal{K}^{\star}_{n+1-r} \setminus \{I\}} \lambda_{I,K}}, \quad \text{with} \quad \lambda_{I,K} = \omega_K \left| B_{I \cup K} \right|^2.$$

The proof can be obtained by considering $V = \operatorname{span}\{\mathcal{B}\}$, where $\mathcal{B} = \{\beta_0, \ldots, \beta_n\}$ with $\beta_j : \Omega \to \mathbb{R}$ for $j = 0, \ldots, n$, and substituting the Vandermonde matrices in [22] with the corresponding collocation matrices B in (26) and B_K in (30).

Another important implication of Theorem 1 is the possibility of rewriting ℓ^p–approximation problems as suitable convex combination of interpolants as outlined in Remark 14 where we exploit the IRLS method, see e. g., [7, Section 4.5].

Remark 14. *Let us consider the problem*

$$\min_{v \in V} \frac{1}{2} \sum_{i=1}^{m} \| v(\boldsymbol{x}_i) - f_i \|_p^p \tag{34}$$

with $1 < p < 2$, where $V = \operatorname{span}\{\mathcal{B}\}$ is a vector space of dimension $n+1$, and $\mathcal{B} = \{\beta_0, \ldots, \beta_n\}$ its basis. The IRLS method approximates the exact solution of problem (34) through the iterative computation of weighted least

squares problems. In particular, the weights are recursively updated for a maximum number iterations $K_{\max}$ *by*

$$\omega_i^{k+1} = \left| s^k(\boldsymbol{x}_i) - f_i \right|^{\frac{(p-2)}{2}},$$

with weights ω_i^k *and solution* s^k *of problem* (27)*, for iterations* $k = 1, \ldots, K_{\max}$*. By direct application of* (32) *in Theorem* 1*, the outcome of each iteration can be rewritten as a convex combination of interpolants. Note that the interpolants and the determinant of the matrices* B_K *do not need to be recomputed. In other words, it is enough to update the weights* ω_K *according to* (31)*, to compute the solution of problem* (34)*.*

Weighted least squares with splines

We present the numerical verification of Theorem 1 for two different vector spaces V: the space of polynomials and the space of univariate splines with a specific degree and regularity. We consider a set of $m = 7$ observations in the form (u_i, p_i), for $i = 1, \ldots, m$, given by

$$\{(-4.5, -2), (-3.5, 0), (-2.2, -1), (-1.2, 2.8), (0.8, 2.9), (2.2, 0.5), (4.0, -2)\}.$$

We define the weights ω_i for $i = 1, \ldots, m$ as uniformly distributed on $(0, 1)$. In the polynomial case, we choose the polynomial degree $d = 2$, which implies

$$\#\left(\mathcal{K}_{d+1}\right) = \binom{m}{d+1} = \binom{7}{3} = 35.$$

Thus, we have a total of 35 interpolation problems to be solved. Moving on to polynomial spline spaces, in addition to the degree, we consider the spline order $k = d + 1 = 3$ and set the knot vector $\boldsymbol{t} = [-5, -5, -5, -5/3, 5/3, 5, 5, 5]$ of length 8, implying that the spline space has dimension $n + 1 = 8 - 3 = 5 \leq m$. Since $n \leq m$, there are at most

$$\#\left(\mathcal{K}_{n+1}\right) = \binom{m}{n+1} = \binom{7}{5} = 21$$

interpolation problems required to fully reconstruct the spline least squares approximant. In the case of spline spaces, there is no guarantee that each point subsequence satisfies the Schoenberg-Whitney nesting conditions. To illustrate this, consider the data set identified by $K = \{1, 2, 3, 4, 5\} \subset \{1, \ldots, m\}$, which does not satisfy these conditions. This is due to the absence of data points within the support of the last basis function, specifically $u_i \notin [5/3, 5]$ for $i = 1, \ldots, 5$. Consequently, the interpolant s_K does not exist, and no interpolation problem needs to be solved for this particular subset. To fully reconstruct the final global least squares approximation in this example, we need to compute 20 interpolation problems instead of the original 21.

Figure 18 illustrates the interpolants and the global least squares approximation involved in (32) for the polynomial and spline spaces introduced

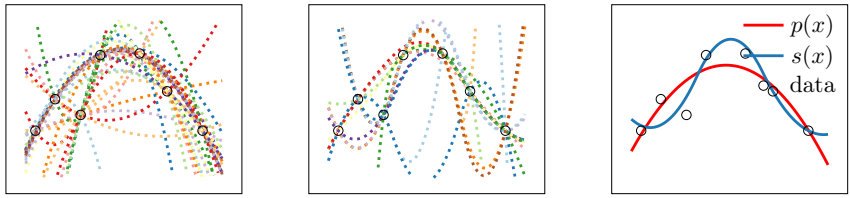


Figura 18. Numerical verification of (32), Theorem 1: $s_K(\boldsymbol{x})$ interpolants for polynomials (left) and splines (center) and the final weighted least squares approximations as sum of interpolants (right) for polynomials (red) and splines (blue).

for this numerical example. Note that we choose to show an example related to the univariate polynomial and spline space, but our results hold in general for any (multivariate) vector space of finite dimension.

Weighted least squares with hierarchical splines

In the following, we perform a numerical investigation of the role of the weights addressed in Corollary 13, for a weighted least squares problem with hierarchical splines in the bivariate case.

We consider a point cloud of 64×64, i.e. $m = 4096$, uniformly gridded data, obtained by sampling the function

$$f(\boldsymbol{x}) = f(x,y) = \frac{2}{3\exp\left(\sqrt{(10x-3)^2+(10y-3)^2}\right)} + \frac{2}{3\exp\left(\sqrt{(10x+3)^2+(10y+)^2}\right)} + \frac{2}{3\exp\left(\sqrt{(10x)^2+(10y)^2}\right)}, \tag{35}$$

for $\boldsymbol{x} = (x,y) \in [-1,1]^2$ and we associate to each item of the point cloud a weight $\omega_i > 0$ for $i = 1,\dots,m$. For different weight values we compute the weighted least squares approximation $s(\boldsymbol{x})$ of the input data in terms of $521C^2$ quartic box splines [12], in their hierarchical extension [79].

In order to investigate the role of the weights in the approximation process, we execute the following two steps for fixed choices of ω_0 and $\omega_\gamma \in \mathbb{R}_{>0}$,

1. we perform a standard least squares approximation and individuate the data sites whose approximation error is above a certain threshold ϵ, thus define $K \coloneqq \{i \in \{1,\dots,m\} : |s(\boldsymbol{x}_i) - f(\boldsymbol{x}_i)| > \epsilon\}$;

2. or $i \in \{1,\dots,m\} \setminus K$, we set the corresponding weights values $\omega_i = \omega_0$ and similarly for $i \in K$ we set $\omega_i = \omega_\gamma$.

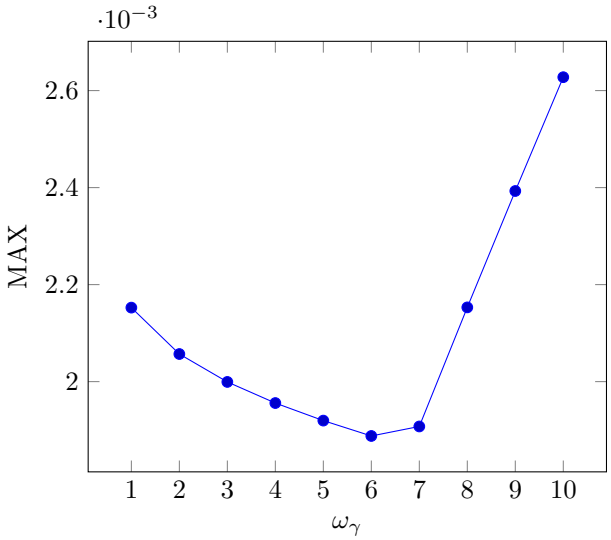


Figura 19. MAX trend with respect to the weights ω_γ for the weighted least suqares approximation of (35) with Hierarchical box splines.

We then analyse the accuracy behaviour of the resulting approximant $s(\boldsymbol{x})$ for different choices of $(\omega_0, \omega_\gamma)$ in terms of MAXimum error (MAX). Setting $\epsilon = 5\text{e}{-4}$, then $\#(K) = 124$ and for $\omega_0 = \omega_\gamma = 1$, the resulting ordinary least squares approximation is characterized by MAX$= 2.15\text{e} - 3$. We then keep $\omega_0 = 1$ and vary ω_γ between 1 and 100. The behaviour of MAX is depicted in Figure 19, for $\omega_\gamma = 1, \ldots, 10$. In particular, we note that increasing the value of the weights ω_γ may help the final accuracy of the approximant. More specifically, we obtain MAX$= 2.15\text{e}{-3}, 2.06\text{e}{-3}, 2.00\text{e}{-3}$ for $\omega_\gamma = 2, 3, 4$ and MAX decreases further when increasing ω_γ until achieving its minimum among the sample values, with MAX$= 1.89\text{e} - 3$, for $\omega_\gamma = 6$. However, if the weight values ω_γ are too large, the final approximant deteriorates. For instance, for $\omega_\gamma = 7$, MAX$= 1.91\text{e} - 3$, for $\omega_\gamma = 10$, MAX$= 2.63\text{e} - 3$. If ω_γ would be further increased, the approximation quality would also deteriorate, e. g., for $\omega_\gamma = 50$, MAX$= 0.92\text{e} - 2$ and for $\omega_\gamma = 100$, MAX$= 1.36\text{e} - 2$. Finally, the choice of weights can lead to either better or worse spline fitting results in terms of the MAX error compared to ordinary least squares, where all the weights are set to ones.

These results can also be observed in Figure 20, which depicts the hierarchical box spline mesh (left) together with two weighted least squares approximations resulting from two different choices of weights, namely $\omega_\gamma = 6$ (center) and $\omega_\gamma = 100$ (right). In particular, the scaled pointwise error color map indicates a worse approximation power for the latter weight choice.

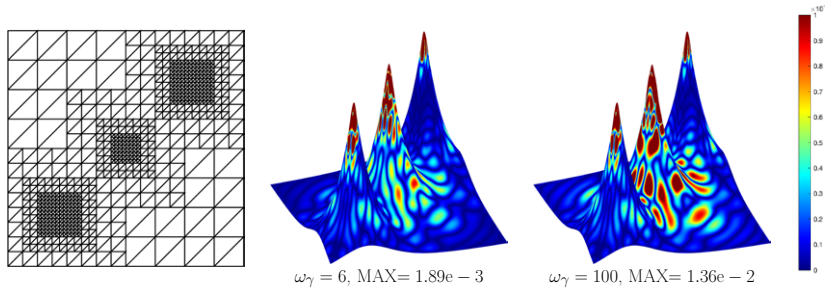


Figura 20. Hierarchical box spline mesh (left); error plots on the WLS hierarchical box spline approximations for $\omega_\gamma = 6$ (center) and 100 (right).

Reweighted least squares spline fitting

In this section, we propose a strategy to take effective advantage of the weights associated with each input data in the context of fitting problems. This approach is inspired by the notion of landmarks used in shape analysis [10], but we introduce a more general concept of *markers* and use them within the framework of curve and surface fitting. Landmarks can be understood as a set of labeled points that represent some physical identifiable parts of an object, as well as important features of the input data, which need to be encoded and reproduced in the final continuous approximation model. Figure 21 shows four point clouds (in blue) and the chosen landmarks (in black), which represent the features desired to be preserved by the fitted curve. However, depending on the acquisition process, data can be affected by noise and outliers, while the final approximation models should avoid the reproduction of corrupted data.

For the given set of observations as in (1), we generalize the concept of landmarks to *markers* of two types. More precisely, we define the index set

$$K_I \subseteq \{1, \dots, m\}$$

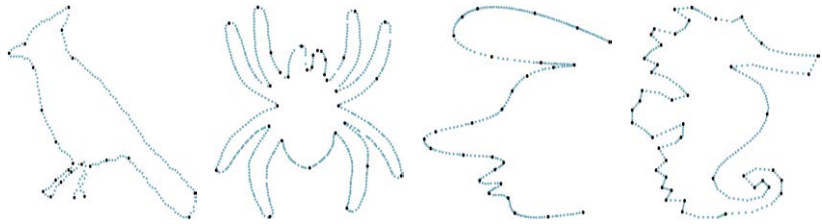
as *markers of type I* if the associated points $\{\boldsymbol{u}_i, \boldsymbol{p}_i\}$ for $i \in K_I$ represent data features to be preserved, while we define the index set

$$K_{II} \subseteq \{1, \dots, m\}$$

as *markers of type II* if the index indicates noisy data or outliers which should not be reproduced. In particular, we have

$$K_I \cap K_{II} = \emptyset \text{ and } K_I \cup K_{II} \subseteq \{1, \dots, m\}.$$

Note that the choice of type I and type II markers and their identification depends on the problem at hand, and it is still an open research topic; see, see e. g., for automatic feature selection [171, 68] or for outlier recognition [122, 42, 43].



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

Figura 21. Point cloud (blue dots) and set of type I markers (black dots)

The markers identification is out of the scope of the present Section and we assume the markers (of both types) to be known a priori. Nevertheless, for synthetic data, we devise an error-driven detection of markers through point cloud pre-processing. Finally, we provide fitting schemes that can address both types of markers simultaneously by leveraging the weight values associated with them depending on the approximation error, also within an adaptive approximation framework.

The fitting problem with markers consist of finding an element $\boldsymbol{s} \in V$ which approximate the points $\{\boldsymbol{u}_i, \boldsymbol{p}_i\}_{i=1}^m$, i. e. $\boldsymbol{s}(\boldsymbol{x}_i) \approx \boldsymbol{p}_i$, with

$$\|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2 < \mathrm{tol}_I, \text{ for } i \in K_I \text{ and } \|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2 > \mathrm{tol}_{II}, \text{ for } i \in K_{II},$$

for two different input tolerances tol_I and tol_{II}.

Remark 15. *Note that usual formulation of a fitting problem can be interpreted as a special instance of the present one. In particular, it is equivalent to setting* $K_I = \{1, \dots, m\}$, $K_{II} = \emptyset$ *and choosing* $\mathrm{tol}_I = \mathrm{tol}_{II} = \epsilon$.

The reweighted least squares algorithm with markers is described in Algorithm 1 and it consists of the following steps. For an initial choice of the weights

1. Solve the weighted least squares approximation problem (27).
2. Update the point-wise error $e_i = \|\boldsymbol{s}(\boldsymbol{x}_i) - \boldsymbol{f}_i\|_2$ for each $i = 1, \dots, m$.
3. Check whether the current fitting $\boldsymbol{s} \in V$ meets the requirements prescribed by the markers K_I and K_{II} and the respective error tolerances. If $e_i < \mathrm{tol}_I$ for $i \in K_I$ and $e_i > \mathrm{tol}_{II}$ for $i \in K_{II}$, stop and return the current approximant. Otherwise, if $e_i > \mathrm{tol}_I$ and $i \in K_I$, set $\omega_i = \omega_i \cdot \alpha$ with $\alpha > 1$; if $e_i < \mathrm{tol}_{II}$ and $i \in K_{II}$, set $\omega_i = \omega_i \cdot \alpha$ with $\alpha < 1$.
4. Start again from step 1 with the new weight values.

Note that the update choice of the weights in step 3 increases the values of the weights related to markers of type I, whereas it decreases them for

Algorithm 1: General formulation of the reweighted least squares fitting.

Input: Point cloud $\{\boldsymbol{u}_i, \boldsymbol{p}_i\}_{i=1}^m$, the set of markers $K_I, K_{II} \subseteq \{1, \dots, m\}$, the tolerances tol_I, tol_{II}, a fixed vector space $V = \text{span}\{\beta_0, \dots, \beta_n\}$ and a maximum number of iterations $\text{M}_{\max}$;

Initialize the weights $\omega_i = 1$ and the point-wise errors $e_i = 1$ for each $i = 1, \dots, m$ and $\text{loop} = 0$

while $\max_{i \in K_I} e_i > \text{tol}_I$ *and* $\max_{i \in I \setminus K_{II}} e_i > \text{tol}_{II}$ *and* $\text{loop} < \text{M}_{\max}$ **do**

Solve the weighted least squares problem

$$\boldsymbol{s}(x) = \arg\min_{\boldsymbol{v} \in V} \frac{1}{2} \sum_{i=1}^{m} \omega_i \|\boldsymbol{v}(\boldsymbol{x}_i) - \boldsymbol{p}_i\|_2^2.$$

Compute the errors

$$e_i = \|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2, \text{ for } i = 1, \dots, m.$$

If $e_i > \text{tol}_I$ and $i \in K_I$, or $e_i < \text{tol}_{II}$ and $i \in K_{II}$, update the weights associated to the landmaks K_I and K_{II}, namely

$$\omega_i = \omega_i \cdot \alpha, \text{ with } \alpha \text{ as in (36).}$$

Set $\text{loop} = \text{loop} + 1$.

end

Output: $\boldsymbol{s} \in V$ the reweighted least squares approximant.

markers of type II, for points not satisfying the tolerance conditions. In particular, we suggest an error-driven update of the weights

$$\alpha = \begin{cases} (1 + e_i), & \text{if } i \in K_I \\ \frac{1}{(1+e_i)}, & \text{if } i \in K_{II}. \end{cases} \tag{36}$$

The reweighted fitting scheme from Algorithm 1 is presented in its more general formulation, namely, the approximant is sought in any *fixed* finite-dimensional vector space V.

Finally, it is worth highlighting that the IRLS procedure [125, 7] falls within this general fitting scheme for the special choice of $\alpha = 1/\max\{\delta, e_i\}$ for each $i \in K_{II}$, with $\delta > 0$ necessary for the stability of the IRLS method.

Reweighted least squares adaptive spline fitting

In this section we show how to suitably combine the update of the weights with adaptive spline approximation schemes. More precisely, we will extend our scheme to advanced THB-spline constructions discussed in Section 1.

Once an initial configuration is chosen, any adaptive approximation procedure is characterised by four main steps which are successively repeated, to suitably identify the adaptive mesh to be used in the next iteration of the adaptive loop. In particular, any adaptive scheme as in (4), is characterized by 1. SOLVE, i. e. computation of the approximation on the current mesh; 2. ESTIMATE, i. e. error estimation; 3. MARK, i. e. mesh marking strategy; REFINE, i. e. mesh refinement strategy. We revisit the adaptive global least squares method proposed in [85], by suitably assigning weights values to the data observation within the adaptive routine. The main idea which drives an adaptive fitting algorithm consists in adding iteratively degrees of freedom in regions of the domain Ω where the approximation error exceeds a certain input tolerance ϵ.

Remark 16. *In the presence of landmarks, the algorithm will be then characterized by three input threshold, i. e.* ϵ*, which guides the adaptive refinement and* tol_I *and* tol_{II} *which guide the weights update.*

Similarly to Algorithm 1, if the points with a too high error belong to K_I, then their weights will be augmented; otherwise, if they have a too low error and they belong to K_{II}, their weights will be diminished. In addition, at each iteration of the adaptive loop, not only the updates of the weight values but also of the sets K_I and K_{II} take place. This strategy is effective since, thanks to the adaptive refinement, in some regions of the domain the accuracy requirements $\mathrm{tol}_I, \mathrm{tol}_{II}$ are already locally achieved, and it is useless or even armful to keep on modifying the weight values.

Once the initial configuration is chosen, i. e. for a fixed THB-spline space and for an initial choice of the weights and fixed tolerances the first step of the adaptive fitting loop, i. e. SOLVE, consists in computing the control points $\boldsymbol{c}_j^\ell$ for each $j \in A_{\boldsymbol{k}}^\ell$ and $\ell = 0, \dots, L-1$, to define the geometric model in (19). As concerns the weighted least squared method, this is achieved by solving the penalized least squares problem

$$\min_{\substack{\boldsymbol{c}_j^\ell, j \in A_{\boldsymbol{k}}^\ell, \\ \ell=0,\dots,L-1}} \frac{1}{2} \sum_{i=1}^{m} \omega_i \left\| \boldsymbol{s}\left(\boldsymbol{u}_i\right) - \boldsymbol{p}_i \right\|_2^2 + \lambda J\left(\boldsymbol{s}\right), \tag{37}$$

where the penalization term J is the thin-plate energy functional, whose influence is controlled by a weight $\lambda \geq 0$, i. e. for $\boldsymbol{x} = (x, y) \in \Omega$,

$$J\left(\boldsymbol{s}\right) = \int_\Omega \left(\left\| \frac{\partial^2 \boldsymbol{s}}{\partial x \partial x} \right\|_2^2 + 2 \left\| \frac{\partial^2 \boldsymbol{s}}{\partial x \partial y} \right\|_2^2 + \left\| \frac{\partial^2 \boldsymbol{s}}{\partial y \partial y} \right\|_2^2 \right) \mathrm{d}x\mathrm{d}y. \tag{38}$$

The second step of the adaptive fitting loop, i. e. ESTIMATE, consists of evaluating the THB-spline approximant on the data sites $\boldsymbol{u}_i \in \Omega$ related to the data points $\boldsymbol{p}_i$ to compute a suitable error indicator. In particular, we choose the point-wise error distance

$$\left\| \boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i \right\|_2$$

for each $i = 1, \dots, m$, among others. The error indicator indicates the region of the domain Ω where additional degrees of freedom are needed to meet the prescribed surface accuracy, by individuating the sites $\boldsymbol{u}_i \in \Omega$ where it exceeds a certain input threshold, i. e.

$$\|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2 \geq \epsilon.$$

For the making strategy, i. e. MARK, we select the cells of the current hierarchical level ℓ that contain the parameters $\boldsymbol{u}_i$ identified by the error indicator and mark them for refinement, together with two surrounding rings of cells in the hierarchical mesh. Subsequently, the marked cells are dyadically refined to effectively enlarge the hierarchical spline space, i. e. REFINE.

Finally, if

$$\|\boldsymbol{s}(\boldsymbol{x}_i) - \boldsymbol{p}_i\|_2 < \epsilon$$

for each $i = 1, \dots, m$, we accept the approximation computed in (37), otherwise before performing another loop of the iterative procedure, starting again from SOLVE, we update the weights values and the marker sets K_I and K_{II}. In particular, for each $i \in K_I$, if $e_i < \text{tol}_I$, then the i-th data satisfies the accuracy requirements and should not belong to the markers of type I any more, namely $K_I = K_I \setminus \{i\}$. On the contrary, if $e_i > \text{tol}_I$, then its corresponding weight needs to be enlarged, i. e. $\omega_i = \alpha \cdot \omega_i$, with $\alpha > 1$. Similar considerations hold for $i \in K_{II}$, with inverted inequalities and $\alpha < 1$. A reasonable error-driven choice for α can be again the one suggested in (36).

For more details about the marking and refinement strategies, we refer the reader to [85], whereas the pseudo-code of the reweighted least squares adaptive spline algorithm is reported in Algorithm 2.

Numerical results for rWLS spline models

In this Section, we present a selection of numerical experiments to show the performance of the proposed reweighted least squares fitting schemes. In particular, we start b showing the effectiveness of the proposed spline fitting method to recover the sharp features of the four different point sequences with type I markers, depicted in Figure 21; subsequently we consider the extension of the proposed method to adaptive spline spaces for surface reconstruction and suggests an automatic recognition of sharp features and type I markers.

Reweighted spline curve fitting with type I markers

To show the performance of Algorith 1, we consider the points clouds and markers of type I depicted in Figure 21. The point clouds are similar to the ones presented in the following works: for Figure 21 (a) [124], (b) [89], (c) and (d) [108]. Our main goal is to improve the reconstruction while keeping the spline space intact. Each considered point sequence $\mathcal{P}$ is equipped with suitable uniform parameter values $\mathcal{U}$, and we compute the approximations

Algorithm 2: Reweighted adaptive least squares spline fitting.

Input: Point cloud $\{\boldsymbol{u}_i, \boldsymbol{p}_i\}_{i=1}^m$, the set of markers $K_I, K_{II} \subseteq \{1, \ldots, m\}$, the tolerances tol_I, tol_{II} and $\epsilon > 0$, a tensor product spline space V^0 and a maximum number hierarchical level L.

Initialize the weights $\omega_i = 1$ and the point-wise errors $e_i = 1$ for each $i = 1, \ldots, m$ and loop $= 0$

while $\max_i e_i > \epsilon$ *and* loop $<$ L **do**

  Solve the penalized weighted least squares problem

$$\min_{\substack{\boldsymbol{c}_j^\ell, j \in A_{\boldsymbol{k}}^\ell, \\ \ell = 0, \ldots, L-1}} \frac{1}{2} \sum_{i=1}^{m} \omega_i \left\| \boldsymbol{s}\left(\boldsymbol{u}_i\right) - \boldsymbol{p}_i \right\|_2^2 + \lambda J\left(\boldsymbol{s}\right).$$

  Estimate the pointwise errors

$$e_i = \|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2, \text{ for } i = 1, \ldots, m.$$

  For each $i \in K_I$, **if** $e_i > \text{tol}_I$ **then**

    update $\omega_i = \omega_i \cdot \alpha$, with $\alpha > 1$

  else

    $K_I = K_I \setminus \{i\}$.

  end

  For each $i \in K_{II}$, **if** $e_i < \text{tol}_{II}$ **then**

    update $\omega_i = \omega_i \cdot \alpha$, with $\alpha < 1$

  else

    $K_{II} = K_{II} \setminus \{i\}$.

  end

  Mark the domain elements where $e_i > \epsilon$ and two surrounding rings of cells in the hierarchical mesh.

  Perform dyadic refinment of the marked cells.

  Update the hierarchical mesh $\mathcal{M}$ and the hierarchical space V.

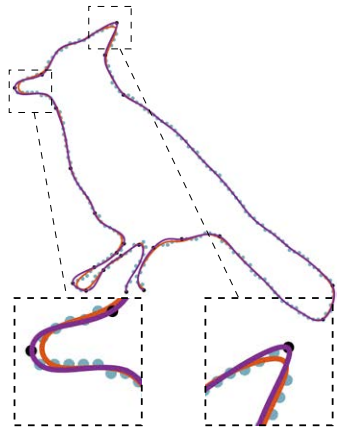
  Set loop = loop+1.

end

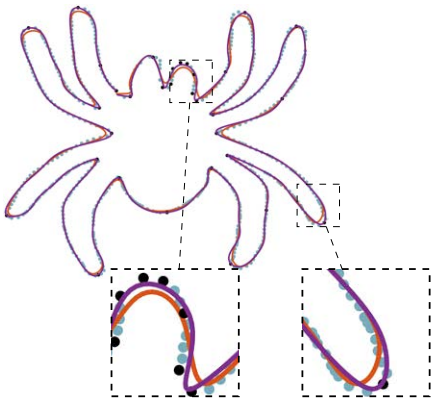
Output: $\boldsymbol{s} \in V$ the reweighted adaptive least squares approximant.

of each dataset for the same spline space, i. e. with the same polynomial degree and the same amount of interior nodes in the *uniform* knot vector.

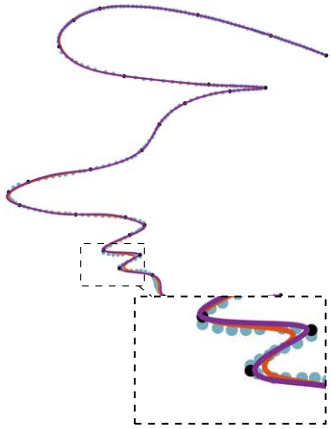
We set the tolerance of the proposed method as $\text{tol}_I = 10^{-3}$ and compare the solution of the rWLS problem with the solution of the ordinary LS problem, i. e. all the weights are equal to one. The solutions are depicted in Figure 22 (a), (b), (c), and (d), for the input data of Figure 21 (a), (b), (c), and (d), respectively. In particular, we can clearly see that the approximation using the reweighted least squares method (depicted in purple in Figure 22) shows better accuracy with respect to the solution of the ordinary least squares problem (shown in red). In particular, we can



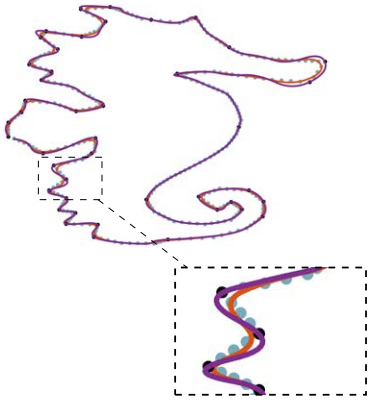
(a) Degree 3 and 50 interior knots



(b) Degree 3 and 60 interior knots



(c) Degree 3 and 40 interior knots



(d) Degree 2 and 84 interior knots

Figura 22. Curve fitting experiments described in Section 19 using ordinary LS (red) and rWLS (purple) for data with markers of type I.

see that the marked data (illustrated in black) are better approximated by the fitted curves. Regarding the approximation error for each experiment, we report that the total Mean Squared Error (MSE) and global MAX are comparable with the ordinary least squares. However, if we compare only how well we can preserve the sharp features, the rWLS method considerably outperforms the ordinary LS. More precisely, in Table 1, we report the MSE and MAX errors measured only for the set of type I marks, respectively for rWLS and LS.

Tabella 1. MSE and MAX errors for type I markers of LS and rWLS solutions in Section 19. For rWLS also the number of iterations (Iterations) is reported.

	Method	Iterations	MSE	MAX
(a)	LS	–	3.93×10^{-5}	1.29×10^{-2}
	rWLS	1442	6.90×10^{-7}	8.94×10^{-4}
(b)	LS	–	1.50×10^{-6}	2.13×10^{-3}
	rWLS	1250	2.28×10^{-7}	9.98×10^{-4}
(c)	LS	–	2.14×10^{-5}	1.21×10^{-2}
	rWLS	30	2.15×10^{-6}	3.53×10^{-3}
(d)	LS	–	3.52×10^{-5}	1.25×10^{-2}
	rWLS	81	1.42×10^{-7}	8.81×10^{-4}

2.2. Hierarchical quasi-interpolation with adaptive spline constructions

In this Section, we consider another fitting routine, which leads to the definition of a hierarchical QI model with THB-splines, particularly suitable for the reconstruction of unstructured data sets. The proposed scheme facilitates the computation of high-quality approximations with an increased level of resolution only in strictly localized areas. More specifically, the computation of the QI control net consists of implementing a two-stage data fitting algorithm with THB-splines, by combining *local* LS B-spline approximations (first stage) with the assembly of the hierarchical quasi-interpolant based on THB-splines (second stage). Finally, to construct the adaptive spline model approximating the whole data set, a suitable strategy to guide the adaptive refinement is developed.

Local least squares spline fitting

By focusing on two-stage spline approximation schemes for THB-splines, a QI operator Q is defined, so that

$$Q\left(\mathcal{U}, \mathcal{P}\right)\left(\boldsymbol{x}\right) = \boldsymbol{s}\left(\boldsymbol{x}\right) \tag{39}$$

where $\boldsymbol{s} : \Omega \to \mathbb{R}^N$ is as in (19). In particular, in this case every coefficient $\boldsymbol{c}_j \in \mathbb{R}^N$ is computed in the first stage of the scheme by using a certain *local* subset of data $\mathcal{U}_j \times \mathcal{P}_j \subset \mathcal{U} \times \mathcal{P}$ for each $j \in \mathcal{A}_{\boldsymbol{k}}^{\ell}$, with $\mathcal{A}_{\boldsymbol{k}}^{\ell}$ as defined in (14), and for each $\ell = 0, \ldots, L-1$, so that

$$\boldsymbol{s}\left(\boldsymbol{u}_i\right) \approx \boldsymbol{p}_i, \text{ for } i = 1, \ldots, m.$$

Note that by exploiting THB-spline constructions, it is possible to define hierarchical QI schemes without any additional efforts with respect to their tensor-product formulations, see [149].

More precisely, for each $j \in \mathcal{A}_{\boldsymbol{k}}^{\ell}$ and $\ell = 0, \ldots, L-1$, let β_j^{ℓ} be the *mother* tensor-product B-spline of the truncated τ_j^{ℓ}, i. e. $\tau_j^{\ell} = \mathrm{Trunc}^{\ell+1}\left(\beta_j^{\ell}\right)$, see Definition 5 in Section 1. In addition, let

$$\Omega_j \subset \Omega, \text{ so that } \Omega_j \cap \operatorname{supp}\left(\beta_j^{\ell}\right) \neq \emptyset,$$

namely, a suitably chosen local subdomain which has a non-empty intersection with the support of β_j^{ℓ}. Subsequently, the indices

$$I_j = \{i \,|\, \boldsymbol{u}_i \in \mathcal{U} \cap \Omega_j\} \subset \{1, \ldots, m\}, \text{ so that } n_{min} \leq |I_j| \ll m$$

have to be selected, together with the corresponding data

$$\mathcal{P}_j = \{\boldsymbol{p}_i \,|\, i \in I_j\}\,, \; \mathcal{U}_j = \{\boldsymbol{u}_i \mid i \in I_j\}\,. \tag{40}$$

Finally, denote with

$$\mathcal{B}_j \coloneqq \left\{\beta_r^{\ell} \,|\, r \in \Lambda_{\boldsymbol{k}}^{\ell,j} \subset \Gamma_{\boldsymbol{k}}^{\ell}\right\}$$

the subset of tensor-product B-splines which do not vanish on Ω_j (by definition β_j^{ℓ} belongs to it). Therefore, the coefficient $\boldsymbol{c}_j^{\ell} \in \mathbb{R}^N$ associated to τ_j^{ℓ}, correspond to the coefficient associated with β_j^{ℓ}, so that $\tau_j^{\ell} = \mathrm{Trunc}^{\ell+1}\left(\beta_j^{\ell}\right)$, in a *local* spline approximation $\boldsymbol{s}_j \in V_j$, with $\boldsymbol{s}_j : \Omega_j \to \mathbb{R}^N$ and $V_j \coloneqq \operatorname{span}\{\mathcal{B}_j\}$.

In the case of surface approximation, hence for $D = 2$ and $N = 3$, the local spline space V_j has a reasonable approximation power and it includes the restriction to Ω_j of any linear polynomial, as we prove in the following Proposition.

Proposition 1. *The following space inclusion holds true,*

$$\Pi_{d|\Omega_j} \subseteq V_j = \operatorname{span}\{\mathcal{B}_j\},$$

where $\Pi_{d|\Omega_j}$ denotes the restriction to Ω_j of the tensor-product space of bi-variate polynomials of bi-degree $\boldsymbol{d} = (d_1, d_2)$.

Dimostrazione. Let q be a cell of the tensor-product mesh G^{ℓ} associated to V^{ℓ}, such that $q \cap \Omega_j \neq$. Then the definition of $\Lambda_{\boldsymbol{k}}^{\ell,j}$, implies that

$$\text{if } q \subseteq \operatorname{supp}\left(\beta_r^{\ell}\right), \text{ then } r \in \Lambda_{\boldsymbol{k}}^{\ell,j}.$$

Thus,

$$\operatorname{span}\{\beta_r^{\ell} \mid q \subseteq \operatorname{supp}\left(\beta_r^{\ell}\right)\} \subset V_j.$$

Since

$$\operatorname{span}\{\beta_r^{\ell} \mid q \subseteq \operatorname{supp}\left(\beta_r^{\ell}\right)\} = \Pi_{d|q},$$

the proof is completed considering that q is any cells of G^{ℓ} with non-vanishing intersection with Ω_j. □

Consequently, if the sites in U_j are not collinear, we then approximate $\mathcal{P}_j$ with

$$\boldsymbol{s}_j(\boldsymbol{u}) = \sum_{r \in \Lambda_{\boldsymbol{k}}^{\ell,j}} \boldsymbol{\alpha}_r^\ell \beta_r^\ell(\boldsymbol{u}), \tag{41}$$

with control points $\boldsymbol{\alpha}_r^\ell \in \mathbb{R}^N$ estimated by solving the following penalized tensor-product B-spline *local least squares* approximation,

$$\min_{\boldsymbol{\alpha}_r^\ell, r \in \Lambda_{\boldsymbol{k}}^{\ell,j}} \sum_{i \in I_j} \|\boldsymbol{s}_j(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2^2 + \lambda J(\boldsymbol{s}_j), \tag{42}$$

where the penalization term J has been chosen again as the thin-plate energy in (38). Hence, we define the j-th control point $\boldsymbol{c}_j^\ell$ associated to τ_j^ℓ as $\boldsymbol{\alpha}_j^\ell$. Finally, If the sites in U_j are collinear, we define the j-th control point $\boldsymbol{c}_j^\ell \in \mathbb{R}^N$ associated to τ_j^ℓ as

$$\boldsymbol{c}_j^\ell = \frac{1}{|I_j|} \sum_{j \in I_j} \boldsymbol{p}_j.$$

The existence and uniqueness of the local approximant (42) in the case of surface approximation, i. e. for $D = 2$ and $N = 3$, is guaranteed by the following Theorem.

Theorem 2. *Let $\boldsymbol{d} = (d_1, d_2)$ be the polynomial bi-degree. If the sites $\boldsymbol{u}_j \in \Omega_j$, $j \in I_j$ are not collinear, there exists a unique local splines $\boldsymbol{s}_j \in V_j$ minimizing the following objective function,*

$$\sum_{i \in I_j} \|\boldsymbol{s}_j(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2^2 + \lambda J(\boldsymbol{s}_j),$$

where the penalization term J has been chosen again as the thin-plate energy in (38). If such points in Ω_j are collinear, then such minimizer does not exists or is not unique.

Dimostrazione. The objective function of Theorem 2 can be split in the sum in the sum of three analogous objective functions, one for each component $\boldsymbol{s}_j^{(k)}$, for $k = 1, 2, 3$. Therefore, to alleviate the analysis and the notation, we develop the proof in the scalar case, which can be extended to $\mathbb{R}^3$ in a straightforward way.

Let $s_j : \Omega_j \to \mathbb{R}$ an element of V_j, hence

$$s_j(\boldsymbol{x}) = \sum_{r \in \Lambda_{\boldsymbol{k}}^{\ell,j}} c_j \beta_r^\ell.$$

Moreover, let $\dim V_j = (n_j + 1)$. Therefore, the functional $J(s_j)$ can be computed as

$$J(s_j) = \boldsymbol{c}^\top G \boldsymbol{c},$$

where $\boldsymbol{c} = \left(c_0, \ldots, c_{n_j}\right)^\top \in \mathbb{R}^{(n_j+1)}$ and $G \in \mathbb{R}^{(n_j+1)\times(n_j+1)}$, so that

$$G_{i,r} = \int_{\Omega_j} \left(\frac{\partial^2 \beta_i^\ell}{\partial x \partial x} \frac{\partial^2 \beta_r^\ell}{\partial x \partial x} + 2 \frac{\partial^2 \beta_i^\ell}{\partial x \partial y} \frac{\partial^2 \beta_r^\ell}{\partial x \partial y} + \frac{\partial^2 \beta_i^\ell}{\partial x \partial x} \frac{\partial^2 \beta_r^\ell}{\partial y \partial y} \right) \mathrm{d}x\mathrm{d}y,$$

where we are assuming that, in the adopted ordering of the tensor-product B-spline basis $\mathcal{B}_j$ of V_j, β_i^ℓ and β_r^ℓ are the i-th and r-th basis functions.

In addition,

$$\sum_{i \in I_j} \left(s_j\left(\boldsymbol{u}_i\right) - p_i\right)^2 = \|B\boldsymbol{c} - \boldsymbol{p}\|_2^2 = \boldsymbol{c}^\top B^\top B \boldsymbol{c} - 2\boldsymbol{p}^\top B \boldsymbol{c} + \boldsymbol{p}^\top \boldsymbol{p},$$

where

$$\boldsymbol{p} = \left(p_1, \ldots, p_{|I_j|}\right)^\top \in \mathbb{R}^{|I_j|}$$

denotes the vector collecting all the p_i, for $i \in I_j$, and $B \in \mathbb{R}^{|I_j|\times(n_j+1)}$ is the collocation matrix defined on $\mathcal{U}_j$ as in (26), for the basis $\mathcal{B}_j$ generating V_j.

The objective function can then be written also in the following quadratic form,

$$\boldsymbol{c}^\top \left(B^\top B + \lambda G\right) \boldsymbol{c} - 2\boldsymbol{p}^\top B \boldsymbol{c} + \boldsymbol{p}^\top \boldsymbol{p}.$$

As well known, a quadratic function admits a global unique minimizer if and only if the symmetric matrix defining its homogeneous quadratic terms is positive definite and in such case the minimizer is given by its unique stationary point. In our case, such a matrix is $B^\top B + \lambda G$ and the stationary points are the solutions of the following linear system of $n_j + 1$ normal equations in as many unknowns,

$$\left(B^\top B + \lambda G\right) \boldsymbol{c} = B^\top \boldsymbol{p}.$$

For all positive λ, the matrix $B^\top B + \lambda G$ is symmetric and positive semidefinite, since for any vector $\boldsymbol{\zeta} \in \mathbb{R}^{(n_j+1)}, \boldsymbol{\zeta} \neq \boldsymbol{0}$, it holds

$$\boldsymbol{\eta}^\top B^\top B \boldsymbol{\zeta} \geq 0, \text{ and } \boldsymbol{\zeta}^\top G \boldsymbol{\zeta} \geq 0,$$

where the right inequality descends from the fact that

$$\boldsymbol{\zeta}^\top G \boldsymbol{\zeta} = J\left(s_{\boldsymbol{\zeta}}\right),$$

with

$$s_{\boldsymbol{\zeta}}\left(\boldsymbol{x}\right) = \sum_{r \in \Lambda_{\boldsymbol{k}}^{\ell,j}} \zeta_r \beta_{r\cdot}^\ell\left(\boldsymbol{x}\right).$$

If the points $\boldsymbol{u}_i$, $i \in I_j$ are distributed on Ω_j along a straight line $ax + by + c = 0$, Proposition 1 implies that it is possible to find $\boldsymbol{\zeta} \in \mathbb{R}^{(n_j+1)}$, $\boldsymbol{\zeta} \neq \boldsymbol{0}$, such that $s_{\boldsymbol{\zeta}}\left(\boldsymbol{x}\right) \equiv ax + by + c$, for all $\boldsymbol{x} \in \Omega_j$. This implies that $s_{\boldsymbol{\zeta}}\left(\boldsymbol{u}_i\right) = 0$, for each $i \in I_j$, namely the vector $B\boldsymbol{\zeta} \in \mathbb{R}^{|I_j|}$ vanishes.

Moreover, also $0 = J(s_{\boldsymbol{\zeta}}) = \boldsymbol{\zeta}^\top G \boldsymbol{\zeta}$, since $s_{\boldsymbol{\zeta}|\Omega_j}$ is a linear polynomial. This proves that the symmetric positive semidefinite matrix $\left(B^\top B + \lambda M\right)$ is not positive definite when all $\boldsymbol{u}_i$, for $i \in I_j$ are collinear. This is also the only possible data distribution associated to a non positive definite matrix. If the points $\boldsymbol{u}_i$, for $i \in I_j$ are not collinear, if $\boldsymbol{\zeta} \in \mathbb{R}^{(n_j+1)}$, $\boldsymbol{\zeta} \neq \mathbf{0}$ is associated to a non-vanishing linear polynomial, it holds $\boldsymbol{\zeta}^\top G \boldsymbol{\zeta} = 0$, but $B\boldsymbol{\zeta} \neq \mathbf{0}$, hence $\boldsymbol{\zeta}^\top B^\top B \boldsymbol{\zeta} > 0$. On the other hand, if $\boldsymbol{\zeta} \in \mathbb{R}^{(n_j+1)}$, $\boldsymbol{\zeta} \neq \mathbf{0}$ is not associated to a linear polynomial, then $\boldsymbol{\zeta}^\top G \boldsymbol{\zeta} > 0$.

□

The computation of all the local approximations (41) for each $j \in \mathcal{A}_{\boldsymbol{k}}^\ell$ and $\ell = 0, \ldots, L-1$ terminates the first stage. The pseudo-code related to the computation of the first stage is illustrated in Algorithm 3.

Remark 17. *Note that each control point $\boldsymbol{c}_j^\ell$ depends on a local subset of the input data, i. e. $\mathcal{U}_j \subset \mathcal{U}$ and $\mathcal{P}_j \subset \mathcal{P}$; hence, differently from the scheme presented in Section 7, the resolution of a global LS linear system is avoided.*

Remark 18. *Solving local LS* tensor-product spline *approximations allow to directly exploit the local tensor-product spline spaces, hence it significantly simplifies the algorithm originally proposed for the first stage in [14, 15], where a variable-degree local* polynomial *approximation was considered. More precisely, the scheme here proposed does not require the selection of a suitable degree for the computation of any coefficient and eliminates the conversion of the computed approximant from the polynomial to the local tensor-product B-spline basis.*

Since the scheme is locally applied, an automatic and eventually data-dependent selection of the parameter λ in (42) could be considered. For example, the choice may take into account the cardinality of $\mathcal{U}_j$ of the local sample, or the area of Ω_j, which influence the value of the first and second addend in (42), respectively. In view of this influence, we can also observe that a constant value of λ implies that the balancing between the fitting and the smoothing term in the objective function usually increases when the size of $\mathcal{U}_j$ or the area of Ω_j increases. This is true in the second case because second derivatives are involved in the smoothing term. Both of these choices seem reasonable and are confirmed by the quality of the results obtained in our experiments, where a constant value for λ is suitably chosen. For completeness, adaptive and local choice of λ for least squares data fitting by tensor-product B-spline surfaces has been proposed in [119, 97].

Differently from [14, 15], in order to better avoid overfitting, a lower bound $n_{\min}$ for the cardinality of $\mathcal{U}_j$ is now required, being $n_{\min} \geq 3$ the only extra input parameter added to the algorithm, besides λ, see Algorithm 3. To fulfill this condition, $\mathcal{U}_j$ is initialized as $\mathrm{supp}(\beta_j^\ell)$ and enlarged until $|\mathcal{U}_j| \geq n_{\min}$. Moreover, to guarantee the locality of the method, the choice of $n_{\min}$ should be kept reasonably small with respect to the amount of input data m. On the other hand, it is not necessary to set a maximum

Algorithm 3: Local least squares spline fitting.

Input: Point cloud $\{\boldsymbol{u}_i, \boldsymbol{p}_i\}_{i=1}^m$, a tensor-product mother B-spline β_j^ℓ with basis index $j \in \mathcal{A}_{\boldsymbol{k}}^\ell$ for some $\ell = 0, \dots, L-1$, the underlying tensor-product mesh M^ℓ of level ℓ, the smoothing parameter $\lambda \geq 0$ and a minimum required number of local data $3 \leq n_{\min} \ll m$;

Initialize $\Omega_j = \operatorname{supp}(\beta_j^\ell)$ and

$$I_j = \{i \,|\, \boldsymbol{u}_i \in \mathcal{U} \cap \Omega_j\}\,,\ \mathcal{P}_j = \{\boldsymbol{p}_i \,|\, i \in I_j\}\,,\ \mathcal{U}_j = \{\boldsymbol{u}_i \mid i \in I_j\}\,.$$

while $|\Omega_j| < n_{\min}$, **do**
  Enlarge Ω_j with the first surrouding ring of cells of M^ℓ
  Update I_j, $\mathcal{P}_j$ and $\mathcal{U}_j$ accordingly.
end
if *The sites in* $\mathcal{U}_j$ *are not collinear,* **then**
  Compute the tensor-product B-spline

$$\boldsymbol{s}_j(\boldsymbol{u}) = \sum_{r \in \Lambda_{\boldsymbol{k}}^{\ell,j}} \boldsymbol{\alpha}_r^\ell \beta_r^\ell(\boldsymbol{u}),$$

  by minimizing (42), and set $\boldsymbol{c}_j^\ell = \boldsymbol{\alpha}_j^\ell$,
else

$$\boldsymbol{c}_j^\ell = \frac{1}{|I_j|} \sum_{i \in I_j} \boldsymbol{p}_i.$$

end

Output: The coefficient $\boldsymbol{c}_j^\ell \in \mathbb{R}^N$ associated to β_j^ℓ.

value for controlling the enlargement of the local data set due to the choice of the refinement strategy that takes this aspect into account, as detailed in the Section 10 (Remark 19), where the operator Q is extended to the hierarchical spline spaces.

Quasi-interpolantion with THB-splines

The second stage consists of assembling the global approximation defined in (19). In particular, due to the THB-spline properties and by following the general approach proposed in [149], the hierarchical QI in (39) can be constructed as

$$\boldsymbol{s}\,(\boldsymbol{x}) = \sum_{\ell=0}^{L-1} \sum_{j \in \mathcal{A}_{\boldsymbol{k}}^\ell} \boldsymbol{c}_j^\ell(\mathcal{P}_j, \mathcal{U}_j) \tau_j^\ell(\boldsymbol{x}),\ \text{for } \boldsymbol{x} \in \Omega, \tag{43}$$

where each coefficient $\boldsymbol{c}_j^\ell(\mathcal{P}_j, \mathcal{U}_j)$ directly corresponds to the coefficient $\boldsymbol{\alpha}_j^\ell$ associated with each mother B-spline β_j^ℓ in (41), as described in Section 2 and further detailed in Algorithm 3. The realization of the first and second stage concludes the first step of the adaptive fitting loop, i. e. SOLVE. Thereby, the iterative design of the hierarchical mesh $\mathcal{M}$ and space V follows.

As for the adaptive fitting procedure presented in Section 7, we choose as error indicator to drive the refinement the point-wise error distance. This concludes the second step of the adaptive loop, i. e. ESTIMATE. More precisely, given an input threshold $\epsilon > 0$, we evaluate the THB-spline approximant on the data sites $\boldsymbol{u}_i \in \Omega$ related to the data points $\boldsymbol{p}_i$ and compute the Euclidean distance $\|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2$ for each $i = 1, \dots, m$. This allows us to individuate the sites $\boldsymbol{u}_i \in \Omega$ where $\|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2 \geq \epsilon$.

The MARK step of the adaptive loop considers a a marking strategy based on basis functions. In particular, the basis function of level ℓ which are active on the data sites characterized by $\|\boldsymbol{s}\,(\boldsymbol{u}_i) - \boldsymbol{p}_i\|_2 \geq \epsilon$ are marked for refinement and replaced by basis functions of the successive hierarchical level $\ell + 1$.

Finally, the refinement strategy, i. e. REFINE, of the THB-spline fitting scheme has to be considered. For some data distributions, e. g., for unevenly scattered data, the parameter values corresponding to the local data set $\mathcal{U}_j$ can be concentrated in a small part of the support of a marked function, thus splitting its support may affect the quality of the final approximation. To better handle these configurations and exploit adaptivity, the refinement procedure is ruled by the parameter n_{loc}, with $n_{\text{loc}} \leq n_{\min}$. Thereby, consider a basis function $\tau_{\boldsymbol{k}}^\ell$ marked to be refined and its mother $\beta_{\boldsymbol{k}}^\ell$. Subsequently, we consider a splitting of the two sides of $\operatorname{supp}\left(\beta_j^\ell\right)$ in n_1 and n_2 uniform segments, respectively, and subdivide $\operatorname{supp}\left(\beta_j^\ell\right)$ in the resulting $n_1 n_2$ subregions, where we then check the presence of at least $\lceil n_{\text{loc}}/(n_1 n_2) \rceil$ data points. The pseudocode for the adaptive hierarchical QI spline fitting is illustrated in Algorithm 4.

Remark 19. *The requirement $n_{\text{loc}} \geq n_{\min}$ guarantees that the points needed to compute the coefficients associated with the new functions in the first stage of the next iteration can be found not too far from the support of the functions themselves. Indeed, in Algorithm 3, after a few enlargements, Ω_j will surely include the support of a refined function of the previous level intersecting* $\operatorname{supp}\left(\beta_j^\ell\right)$.

As a consequence, analogously to $n_{\min}$, a high value of n_{loc} contributes to the reduction of oscillations deriving from overfitting, but this value should also be low enough to guarantee that the refinement strategy can generate a hierarchical spline space with enough degrees of freedom for satisfying the given tolerance ϵ. For this reason, some tuning is necessary for a good selection of n_{loc}.

The proposed adaptive approximation method also extends to the case of surfaces closed in one (or even two) parametric directions, not addressed in previous works. Note that the local nature of the considered

Algorithm 4: Hierarchical QI spline fitting

Input: Point cloud $\{\boldsymbol{u}_i, \boldsymbol{p}_i\}_{i=1}^m$, a tensor-product mesh M^0 and the corresponding tensor-product spline space V^0, the maximum number of hierarchical levels $L \geq 1$, the smoothing parameter $\lambda \geq 0$ and a minimum required number of local data $3 \leq n_{\min} \ll m$, the error threshold $\epsilon > 0$ and the percentage of data points required to satisfy the error tolerance ν, minimum number of local data required for refinement $n_{\min} \leq n_{\text{loc}} \ll m$ and the number of support splits n_1 and n_2.

Initialize the hierarchical mesh $\mathcal{M} = M^0$, the hierarchical spline space $V = V^0$, the point-wise errors $e_i = 1$ for each $i = 1, \ldots, m$ and set loop $= 0$

while $|\{i \mid e_i > \epsilon\}| > \nu \cdot m$ *and* loop $< L$ **do**

  Compute the coefficients of the tensor-product QI with Algorithm 3 and define $\boldsymbol{s} \in V$ as in (43).

  Estimate the pointwise errors $e_i = \|\boldsymbol{s}(\boldsymbol{u}_i) - \boldsymbol{p}_i\|$, for $i = 1, \ldots, m$.

  Mark the basis functions τ_j^ℓ active on $\boldsymbol{u}_i$ such that $e_i > \epsilon$.

  Refine by dyadic split of the marked cells which contains at least $\lceil n_{\text{loc}}/(n_1 n_2) \rceil$ in any of the $n_1 n_2$ sub-rectangles which uniformly split supp $\left(\beta_j^\ell\right)$, mother of τ_j^ℓ.

  Update accordingly the hierarchical mesh $\mathcal{M}$ and the hierarchical space V.

  Set loop = loop+1.

end

Output: The hierarchical QI as defined in (43).

approximation approach makes the implementation especially easy, since coefficients associated with a THB-spline present at successive steps of the adaptive refinement procedure (even if possibly further truncated) do not need to be recomputed.

2.3. Industrial applications for hierarchical QI spline fitting

In this Section, we consider the reconstruction of scattered data of industrial complexity, obtained by an optical scanning process of four different aircraft engine parts, represented in terms of bivariate THB-spline models with bi-degree $\boldsymbol{d} = (d, d)$. For each of these surfaces, as a characterizing dimension, we report the length R of the diagonal of the minimal axis-aligned bounding box associated to the given point cloud. Note that, in industrial applications it is common to require a certain percentage ν of data points to satisfy the error tolerance ϵ.

The results highlight the effects of considering a minimum number of local data points $n_{\min}$ (also) in the first stage of the method, as well as the improvements obtained by introducing a regularized B-spline approximation

for each local fitting with respect to the scattered data fitting scheme considered in [15]. In order to try to produce a very detailed reconstruction, a quite small value has been chosen for $n_{\min}$, more precisely, we suggest selecting it among the integers in the range $[d^2, (d+1)^2]$. By combining these two changes, the two-stage approximation algorithm is more stable and unwanted oscillations are further reduced.

Tensile

In this example, we consider THB-spline approximations to reconstruct a part of a tensile from the set of 9281 scattered data shown in Figure 23, which has reference dimension $R = 2.5 \cdot 10^{-2}$m. We compare the new local scheme based on local B-spline approximations with the algorithm based on local polynomial approximations of variable degree presented in [15], where this test was originally considered. Note that for this example we have never dealt with local sets of collinear points in our experiments.

As a first test, we ran both algorithms with the same setting considered in [15], namely, by starting with a 4×4 tensor-product mesh with $\boldsymbol{d} = (2, 2)$, threshold $\epsilon = 5 \cdot 10^{-5}$m, percentage bound $\nu = 95\%$ and $n_{\text{loc}} = 20$. The algorithm with local polynomial approximations with the parameter choice considered in [15] led to an approximation with 2501 DOFs, 96.25% of points below the tolerance and MAX error of $1.23 \cdot 10^{-4}$m. The new scheme based on local B-spline approximations with $n_{\min} = 6$, $\lambda = 10^{-6}$, and $n_1 = n_2 = 1$ generated a THB-spline surface with 1855 degrees of freedom that satisfies the required tolerance in 98.88% of points with a MAX error of $8.06 \cdot 10^{-5}$m. The number of levels used is 5, but all the cells of the first two levels are refined.

As a second test, we ran both algorithms by starting with a 16×4 tensor-product mesh with $\boldsymbol{d} = (2, 2)$, percentage $\nu = 95\%$, threshold $\epsilon = 5 \cdot 10^{-5}$m, and $n_{\text{loc}} = 15$. The algorithm with local polynomial approximations led to an approximation with 5922 degrees of freedom, 98.18% of points below the tolerance, and a MAX error of $1.44 \cdot 10^{-4}$m. The surface and the corresponding hierarchical mesh are shown in Figure 23 (center). This approximation clearly shows strong oscillations on the boundary of the reconstructed surface due to a lack of available data points for the local fitting in correspondence with high refinement levels. The scheme based on local B-spline approximations, with $n_{\min} = 7$, $\lambda = 10^{-6}$, and $n_1 = n_2 = 1$, produced a THB-spline surface with 1960 degrees of freedom that satisfies the required tolerance in the 99.36% of the data points with a MAX error of $8.11 \cdot 10^{-5}$m. The surface, free of unwanted oscillations along the boundary, and the corresponding hierarchical mesh are shown in Figure 23 (bottom). The number of levels used is 4, with the cells of level 0 completely refined.

Blade

In this example, we consider the second test discussed in [15] on the set of 27191 scattered data representing a scanned part of a blade shown in Figure 24 (top), whose reference dimension is $R = 5 \cdot 10^{-2}$m. Again, to compare the new local scheme with the algorithm based on local polynomial

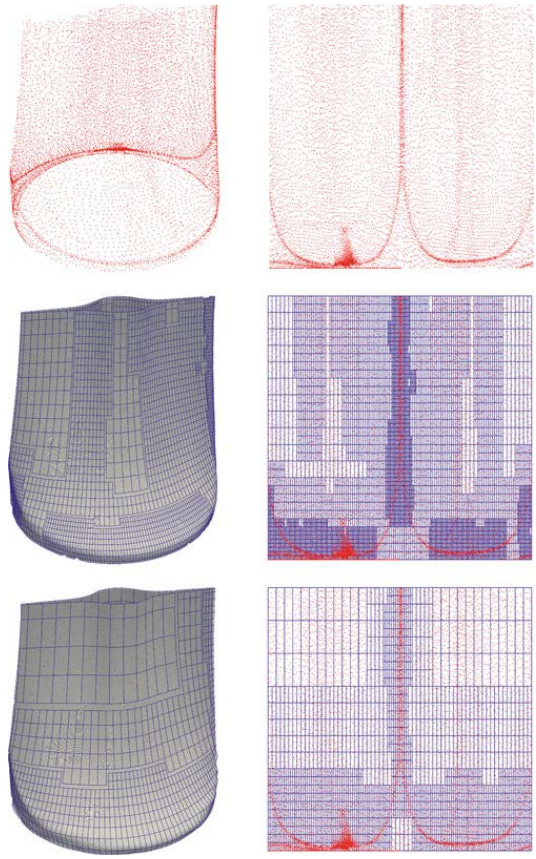


Figura 23. Scattered data set corresponding to a critical part of a tensile (top), the reconstructed surfaces and the corresponding hierarchical meshes obtained with the algorithm in [15] (center) and the new scheme (bottom) in Section 3.

approximations, we ran both algorithms with the same setting of [15], namely, by starting with a 4×4 tensor-product mesh with $\boldsymbol{d} = (3, 3)$, threshold $\epsilon = 2 \cdot 10^{-5}$m, percentage bound $\nu = 95\%$ and $n_{\text{loc}} = 60$. The algorithm with local polynomial approximations with the parameter choice as in [15] led to an approximation with 12721 DOFs, 97.06% of points below the tolerance and MAX error of $1.08 \cdot 10^{-4}$m. The new scheme based on local B-spline approximations with $n_{\min} = 6$, $\lambda = 10^{-8}$ and $n_1 = n_2 = 1$ generated a THB-spline surface with 8314 DOFs that satisfies the required tolerance in 99.94% of points with a maximum error of $1.33 \cdot 10^{-4}$m. The surface and the corresponding hierarchical mesh are shown in Figure 24 (bottom). The number of levels used with the new scheme is 7, with all the cells of the first two levels completely refined, one less than with the old approach. Finally, for completeness, we precise that the local collinearity check in Algorithm 3 is active for 5 coefficients.

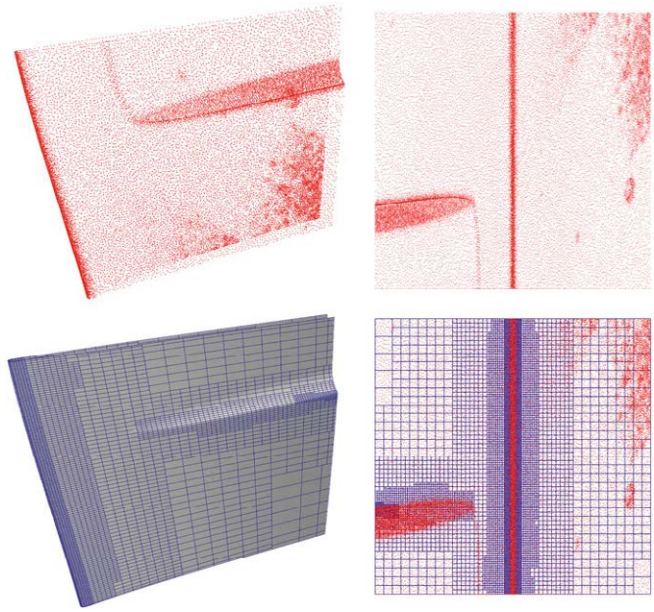


Figura 24. Scattered data set corresponding to a critical part of a blade (top), the reconstructed surface and the corresponding hierarchical mesh obtained with the new scheme(bottom) in Section 3

Endwall

In this example, we illustrate the behaviour of the adaptive algorithm on data sets with voids by considering the reconstruction of an endwall part from the scattered data set of 43.869 points shown in Figure 25 (top), whose reference dimension is $R = 5 \cdot 10^{-1}$m. The figure shows that in this case, the data set represents a model with three different holes where no input data are available. The aim of this reconstruction is to avoid artifacts due to a lack of points and obtain a sufficiently regular surface, i. e. by avoiding self-intersections, that can be post-processed with standard geometric software tools to obtain a suitably trimmed model. Consequently, not only the number of points in the local data sets is important to reach this aim, but also their distribution. To properly address this issue, we consider a real density parameter with a value between 0 and 1, which determines whether the distribution of the points in the local set is reliable or not for the fitting. The distribution of the local data points is computed as the number of mesh cells of level ℓ inside the support of of the marked function β_j^ℓ or its enlargement, which contain at least one point, over the total number of mesh cells, either in the support of β_j^ℓ or its enlargement. If this ratio is below a density threshold, then more data points are required and the function support is enlarged for the computation of the local approximation in the first stage of the method. The approximation is

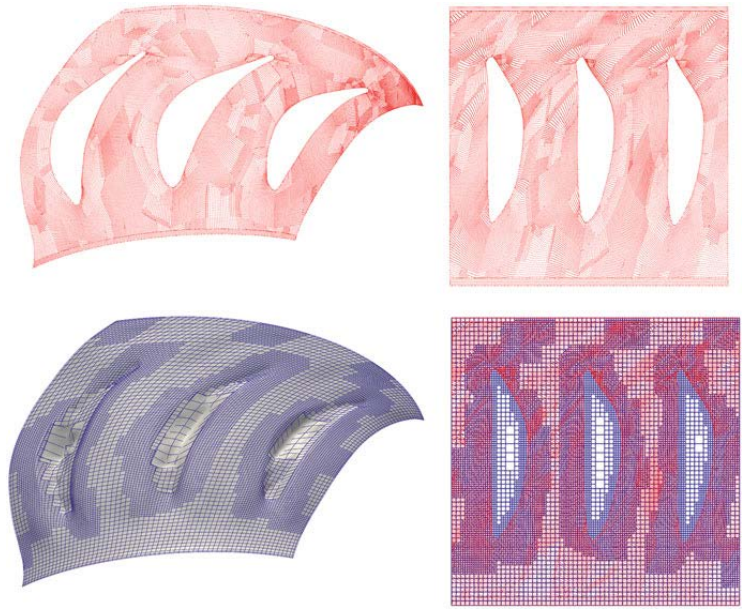


Figura 25. Scattered data set corresponding to a critical part of a endwall (top), the reconstructed surface and the corresponding hierarchical mesh obtained with the new scheme(bottom) in Section 3

developed by starting from a 32×32 tensor-product mesh, with $\boldsymbol{d} = (3, 3)$, $n_{\text{loc}} = 15$, $n_{\min} = 11$, $\lambda = 10^{-6}$ and $n_1 = n_2 = 2$. A choice of the density parameter to 0.3 permits to take care of the difficult distribution of data points in the construction of the approximation. By considering a threshold $\epsilon = 5 \cdot 10^{-5}$m and a percentage bound $\nu = 95\%$, the refinement generated a THB-spline approximation with 3 hierarchical levels, 11211 DOFs, 98.70% of points below the threshold and MAX error of $5.69 \cdot 10^{-4}$m. The surface and the corresponding hierarchical mesh are shown in Figure 25. In this case there are 18 coefficients of the last level and 15 of the last but one (all associated with B-splines whose support intersects a void) such that the related $\mathcal{U}_j$ is made up of all collinear points.

Airfoil

In this last example, we illustrate the behaviour of the new adaptive fitting algorithm with local B-spline approximations for surfaces closed in one parametric direction. In particular, we test the scheme to reconstruct a blade airfoil from a set of 19669 scattered data, shown in Figure 26, whose reference dimension is $R = 10^{-1}$m. We ran the method by starting from a 32×4 tensor-product mesh with $\boldsymbol{d} = (3, 3)$, setting the percentage bound $\nu = 0.95$, the threshold $\epsilon = 5 \cdot 10^{-5}$m, $n_{\text{loc}} = 30$, $n_{\min} = 12$, $\lambda = 10^{-6}$, and $n_1 = n_2 = 1$. The adaptive scheme produces a THB-spline approximation with 3 hierarchical levels and 1865 DOFs distributed in the last two levels,

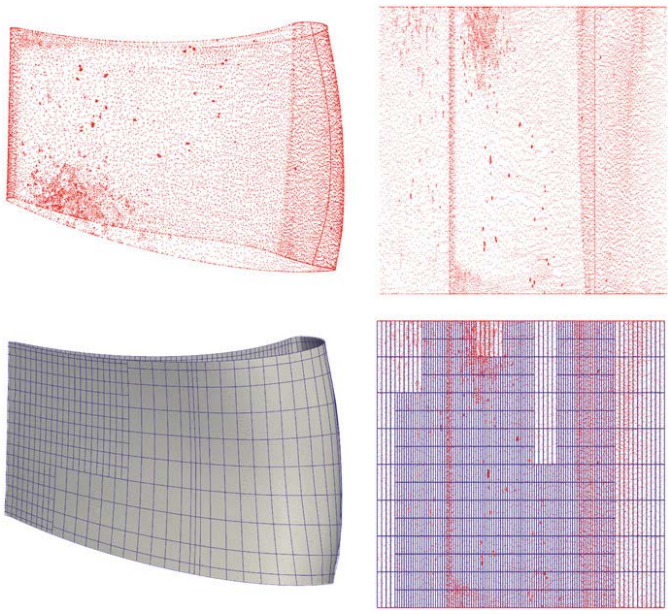


Figura 26. Scattered data set corresponding to a critical part of an airfoil (top), the reconstructed surface and the corresponding hierarchical mesh obtained with the new scheme(bottom) in Section 3

and it satisfies the tolerance in 95.06% of the data points with a MAX error of $1.88 \cdot 10^{-4}$m. The surface and the corresponding hierarchical mesh are shown in Figure 26 (bottom). By trying to force additional refinement, some oscillations appear. In this case, they are consistent with the data distribution since there are clusters of high-density points due to scan noise. Consequently, they do not represent artifacts caused by regions with very low density of data and cannot be prevented by exploiting the bound for cardinality of the local data sample governed by n_{loc}.