*axioms*

MDPI

*Article*

# Refinement Algorithms for Adaptive Isogeometric Methods with Hierarchical Splines

**Cesare Bracco [1], Carlotta Giannelli [1,*] and Rafael Vázquez [2,3]**

[1] Dipartimento di Matematica e Informatica "U. Dini", Università degli Studi di Firenze, Viale Morgagni 67/A, 50134 Florence, Italy; cesare.bracco@unifi.it

[2] Institute of Mathematics, Ecole Polytechnique Fédérale de Lausanne, Station 8, 1015 Lausanne, Switzerland; rafael.vazquez@epfl.ch

[3] Istituto di Matematica Applicata e Tecnologie Informatiche "E. Magenes" del CNR, via Ferrata 5, 27100 Pavia, Italy

* Correspondence: carlotta.giannelli@unifi.it; Tel.: +39-055-275-1407

Received: 11 May 2018; Accepted: 18 June 2018; Published: 21 June 2018


**Abstract:** The construction of suitable mesh configurations for spline models that provide local refinement capabilities is one of the fundamental components for the analysis and development of adaptive isogeometric methods. We investigate the design and implementation of refinement algorithms for hierarchical B-spline spaces that enable the construction of locally graded meshes. The refinement rules properly control the interaction of basis functions at different refinement levels. This guarantees a bounded number of nonvanishing (truncated) hierarchical B-splines on any mesh element. The performances of the algorithms are validated with standard benchmark problems.

**Keywords:** isogeometric analysis; adaptive methods; hierarchical splines; THB-splines; local refinement

## 1. Introduction

The design of isogeometric methods for the numerical solution of partial differential equations extends classical finite element techniques by taking into account computer aided design (CAD) methods and standards [1,2]. For this reason, the isogeometric approach naturally encourages a tighter connection between computer aided engineering and design software libraries. While the potential of exploiting a common representation model, as well as the enhanced smoothness of higher order spline schemes, opened the possibility of developing highly accurate methods, the backward CAD compatibility also poses some limits and challenges. One of the most important restrictions is the tensor-product structure of standard multivariate B-spline models, which necessarily prevents the possibility of local mesh refinement. This motivates several authors to advance the study of adaptive spline constructions.

Hierarchical B-splines constitute one of the most promising solutions to easily define adaptive spline basis which preserve the nonnegativity of standard B-splines and facilitate the design of fully automatic schemes [3–7]. When the truncated basis of hierarchical B-spline spaces is considered the overlap of truncated basis functions at different hierarchical levels is reduced and the partition of unity property recovered [8]. Regarding the implementation of hierarchical B-splines, different data structures and algorithms have been already proposed in the literature (see e.g., [9–11] and [12,13]). Particular attention was also devoted to the efficient Bernstein–Bézier evaluation in the hierarchical setting [14,15]. In this work, we employ the structures introduced in [11], which have an implementation in GeoPDEs [16].

A key ingredient for the analysis of adaptive isogeometric methods is the possibility to consider certain class of *admissible* meshes which automatically guarantee a bounded number of non-zero

basis functions on each mesh element. In the hierarchical spline setting, admissible meshes also guarantee that the level of the mesh element and the level of any basis function that does not vanish on the element differ at most a fixed value. This kind of restricted hierarchies naturally reduces the overlapping of basis functions introduced at different levels and influences the sparsity patterns of the discretization matrices. Being connected to the number of elements influenced by the support of a function in the hierarchical basis, the development of a refine module for the adaptive isogeometric method based on these observations can properly profit of (truncated) basis functions with reduced support. This paper is devoted to the study of design and implementation aspects for the development of hierarchical refinement strategies that guarantee the control of different classes of admissibility. Our analysis allows the definition and construction of suitable admissible meshes for standard and truncated hierarchical B-splines by considering a unified framework. The theoretical properties of the refinement algorithms and the resulting meshes are thoroughly analyzed and presented together with extensive numerical testing.

The structure of the paper is as follows. Section 2 introduces the hierarchical B-spline model by focusing on the basis construction and the main data structures needed for the implementation. Section 3 presents the algorithms for the construction and refinement of admissible hierarchical meshes. In Section 4, we briefly recall the adaptive isogeometric setting, while Section 5 shows the results obtained by integrating the refinement procedures in an adaptive isogeometric scheme. Finally, Section 6 concludes the paper.

## 2. The Hierarchical B-Spline Model

We briefly review in this section the construction of (truncated) hierarchical B-splines and introduce the data structures and basic functionalities considered for the implementation of the spline hierarchy. The key components for the design of a software architecture devoted to hierarchical spline structures rely on the storage of the hierarchical mesh as well as on the information related to the adaptive spline space.

### *2.1. Basis Construction*

Let $V^0 \subset V^1 \subset \ldots \subset V^{N-1}$ be a nested sequence of $N$ tensor-product $d$-variate spline spaces defined on a closed hyper-rectangle $D$ in $\mathbb{R}^d$. For each level $\ell$, $\ell = 0, \ldots, N-1$, we consider the tensor-product B-spline basis $\mathcal{B}^\ell$ of degree $\mathbf{p} = (p_1, \ldots, p_d)$ defined on the rectilinear grid $G^\ell$. B-splines have local support and satisfy the following properties: local linear independence, non-negativity, partition of unity (see, e.g., [17,18]).

We also consider a nested sequence of closed subsets of $D$ to define the domain hierarchy:

$$\Omega = \Omega^0 \supseteq \Omega^1 \supseteq \ldots \supseteq \Omega^{N-1} \supseteq \Omega^N = \emptyset \quad \text{with} \qquad \Omega^\ell = \bigcup_{Q \in \mathcal{R}^{\ell-1}} \overline{Q},$$

where $\mathcal{R}^{\ell-1} \subset G^{\ell-1}$ are the refined elements of level $\ell - 1$. The hierarchical mesh is defined as the collection of the open active elements at different levels, namely

$$\mathcal{Q} := \left\{ Q \in \mathcal{G}^\ell, \, \ell = 0, \ldots, N-1 \right\} \quad \text{with} \quad \mathcal{G}^\ell := \left\{ Q \in G^\ell : Q \subset \Omega^\ell \wedge Q \not\subset \Omega^{\ell+1} \right\}. \tag{1}$$

By following [3,4], a subset of B-splines at different hierarchical levels can be properly selected to construct the hierarchical basis according to the following definition.

**Definition 1.** *The hierarchical B-spline (HB-spline) basis $\mathcal{H}$ with respect to the mesh $\mathcal{Q}$ is defined as*

$$\mathcal{H}(\mathcal{Q}) := \left\{ \beta^\ell \in \mathcal{B}^\ell : \operatorname{supp} \beta^\ell \subseteq \Omega^\ell \wedge \operatorname{supp} \beta^\ell \not\subseteq \Omega^{\ell+1}, \, \ell = 0, \ldots, N-1 \right\}.$$

For any level $\ell$, the selection mechanism identifies the set of B-splines of this level whose support is contained in the domain $\Omega^\ell$ but not fully contained in the next domain of the hierarchy, $\Omega^{\ell+1}$. This part of the domain will be covered by selecting B-splines of levels greater than $\ell$. The construction preserves the non-negativity and linear independence of one-level B-splines [3,4]. Obviously, coarser B-splines will interact with elements and refined B-splines of subsequent hierarchical levels. In view of this overlap between basis functions at different levels of detail, the partition of unity property is lost in the hierarchical construction. The truncated basis for hierarchical splines [8] recovers this property by removing the contribution of finer B-splines in the hierarchical basis from coarser ones. This is possible thanks to the two-scale relation between any B-spline of level $\ell$ and refined B-splines of level $\ell+1$. More precisely, any $s \in V^\ell \subset V^{\ell+1}$ can be expressed as

$$s = \sum_{\beta \in \mathcal{B}^{\ell+1}} c_{\beta^{\ell+1}}(s)\beta^{\ell+1}, \tag{2}$$

in terms of the coefficients $c_{\beta^{\ell+1}}$. The truncation of $s$ with respect to level $\ell+1$ simply removes, in the above sum, the B-splines of level $\ell+1$ having supports fully contained in $\Omega^{\ell+1}$, and that will be included in the basis $\mathcal{H}(\mathcal{Q})$. It is then defined as follows:

$$\text{trunc}^{\ell+1} s := \sum_{\beta^{\ell+1} \in \mathcal{B}^{\ell+1},\, \text{supp}\, \beta^{\ell+1} \not\subseteq \Omega^{\ell+1}} c_{\beta^{\ell+1}}(s)\beta^{\ell+1}. \tag{3}$$

**Definition 2.** *The truncated hierarchical B-spline (THB-spline) basis $\mathcal{T}$ with respect to the mesh $\mathcal{Q}$ is defined as*

$$\mathcal{T}(\mathcal{Q}) := \left\{ \text{Trunc}^{\ell+1}\, \beta^\ell : \beta^\ell \in \mathcal{B}^\ell \cap \mathcal{H}(\mathcal{Q}),\, \ell = 0, \dots, N-2 \right\} \bigcup \left\{ \beta^{N-1} : \beta^{N-1} \in \mathcal{B}^{N-1} \cap \mathcal{H}(\mathcal{Q}) \right\},$$

*where* $\text{Trunc}^{\ell+1}\, \beta^\ell := \text{trunc}^{N-1}(\text{trunc}^{N-2}(\dots(\text{trunc}^{\ell+1}(\beta^\ell))\dots))$, *for any* $\beta^\ell \in \mathcal{B}^\ell \cap \mathcal{H}(\mathcal{Q})$.

THB-splines are non-negative, linearly independent, form a partition of unity, and span the same space of HB-splines [8]. The properties of non-negativity and partition of unity imply the convex hull property, a fundamental concept for geometric modelling applications.

### *2.2. Data Structures for the Implementation*

The implementation of numerical methods based on hierarchical B-splines requires the definition of suitable data structures, which contain all the necessary information for the computation of the basis $\mathcal{H}(\mathcal{Q})$ (or $\mathcal{T}(\mathcal{Q})$). In this work, we employ the data structures introduced in [11], and, for the sake of completeness, we recall their main fields and functionalities, before using them to develop the refinement algorithms. We will need four different data structures: two for tensor-product B-splines, and two for hierarchical B-splines.

The first two-structures regard the computation of tensor-product B-splines. We assume that, for each level $\ell$, we have a *mesh structure* with all the information of the rectilinear grid $G^\ell$, and a *space structure* with all the required information to define the basis functions of the tensor-product space $V^\ell$. In particular, these data structures contain the following methods:

- **get_basis_functions**: given an element $Q \in G^\ell$, compute the indices of the basis functions in $\mathcal{B}^\ell$ that do not vanish in $Q$;
- **get_cells**: given a function $\beta^\ell \in \mathcal{B}^\ell$, compute the elements $Q$ in the support of $\beta$;
- **get_support_extension**: for a given element $Q \in G^\ell$, compute its support extension $\widetilde{Q}$, defined as

$$\widetilde{Q} := \left\{ Q' \in G^\ell : \exists\, \beta^\ell \in \mathcal{B}^\ell,\, \text{supp}\, \beta^\ell \cap Q' \neq \emptyset \wedge \text{supp}\, \beta^\ell \cap Q \neq \emptyset \right\}. \tag{4}$$

This last function can be implemented as a sequential call of the previous two methods.

The next two structures contain the information about the hierarchical B-splines. The first one is the *hierarchical mesh structure*, denoted by `MESH` in the algorithms of Section 3, which contains all the information about the hierarchical mesh $\mathcal{Q}$. In particular, it includes the following fields:

- the number of levels, $N$;
- for each level $\ell$, a structure for the rectilinear grid $G^\ell$;
- for each level $\ell$, the list of active elements $\mathcal{G}^\ell$, denoted by $\mathtt{E}^{\mathtt{A}}_\ell$ in the algorithms of Section 3;
- the kind of refinement (dyadic, triadic...) between levels.

It also contains two methods, necessary for the development of refinement. They can be briefly described as follows:

- **get_parent_of_cell**: given a cell $Q \in G^\ell$ (or a list of cells), compute the index of its parent, that is, the unique cell $Q' \in G^{\ell-1}$ such that $Q \subset Q'$;
- **get_ancestor_of_cell**: given a cell $Q \in G^\ell$ (or a list of cells) of level $\ell$, and given $0 \leq k < \ell$, return the unique index of the ancestor of $Q$ of level $k$, that is, the unique cell $Q' \in G^k$ such that $Q \subset Q'$.

The first method was already presented in [11], while the second one is detailed in the recursive Algorithm 1.

**Algorithm 1**: **get_ancestor_of_cell.**
**Description**: get ancestor of level $k$ for an element $Q$ (or a list of elements) of level $\ell > k$.

**Input:** `MESH`, $Q, \ell, k$
1: `ancestors` $\leftarrow$ **get_parent_of_cell** (`MESH`, $\ell$, $Q$)
2: **if** $(k < \ell - 1)$ **then**
3: `ancestors` $\leftarrow$ **get_ancestor_of_cell** (`MESH`, `ancestors`, $\ell - 1, k$)
4: **end if**
**Output:** `ancestors`

The last data structure is the *hierarchical space structure*, which will be denoted by `SPACE` in the algorithms of Section 3, and which contains the necessary information regarding the basis functions $\mathcal{H}(\mathcal{Q})$, or $\mathcal{T}(\mathcal{Q})$. In particular, it contains the following fields:

- the number of levels, $N$;
- for each level $\ell$, a space structure for the tensor-product space $V^\ell$;
- for each level $\ell$, the set of active basis functions in $\mathcal{B}^\ell \cap \mathcal{H}(\mathcal{Q})$;
- the coefficients of the two-scale relation (2) between levels $\ell$ and $\ell + 1$.

This is all the functionality required to implement the refinement algorithms of the following section. We refer the interested reader to [11] for a more detailed description of the data structures, and their use in isogeometric analysis.

## 3. Admissible Refinement Algorithms

In order to develop the theory for adaptive isogeometric methods, exploiting the reduced support of THB-splines with respect to standard HB-splines, Buffa and Giannelli introduced in [5] the concept of admissible meshes, for which the basis functions acting in one element come from a limited number of levels. The same concept was introduced for HB-splines in [7], limiting the number of levels to two. The two types of admissibility are enclosed in the following definition.

**Definition 3.** *A mesh $\mathcal{Q}$ is $\mathcal{H}$-admissible (respectively, $\mathcal{T}$-admissible) of class $m$, with $m \geq 2$, if the basis functions in $\mathcal{H}(\mathcal{Q})$ (resp. $\mathcal{T}(\mathcal{Q})$) which take non-zero values over any element $Q \in \mathcal{Q}$ belong to at most $m$ successive levels.*

Let $Q$ be an active element of level $\ell$ where at least one hierarchical basis function of the same level is non-zero. When an admissible mesh of class 2 is considered, the active basis functions which are non-zero on $Q$ are only the ones of level $\ell-1$ and $\ell$. They are of levels $\ell-2, \ell-1, \ell$ when $m=3$, and in general the basis functions non-vanishing on $Q \in \mathcal{G}^\ell$ belong to levels $\ell-m+1, \ell-m+2, \ldots, \ell$. References [5] and [7] provide refinement algorithms that generate $\mathcal{T}$-admissible and $\mathcal{H}$-admissible meshes, respectively, limited to class 2 in the second case. Both algorithms follow the same idea: given a set of marked elements, coarse elements in their neighborhood are also refined to enforce the admissibility of the mesh. Before properly defining the neighborhood, we follow [5] and extend the definition of *support extension* of a mesh element, introduced in Equation (4), to the hierarchical setting. We note that the definition is independent on whether we work with HB-splines or THB-splines.

**Definition 4.** *The multilevel support extension $S(Q,k)$ of an element $Q \in G^\ell$ with respect to level $k$, with $0 \leq k \leq \ell$, is defined as*

$$S(Q,k) := \left\{ Q' \in G^k : \exists\, \beta^k \in \mathcal{B}^k, \operatorname{supp} \beta^k \cap Q' \neq \emptyset \wedge \operatorname{supp} \beta^k \cap Q \neq \emptyset \right\}.$$

To keep the notation as simple as possible, we will also denote by $S(Q,k)$ the region occupied by the closure of elements in $S(Q,k)$. Algorithm 2 details the computation of the multilevel support extension using the data structures and the methods introduced in Section 2.2. We first compute, in case $k < \ell$, the ancestor of level $k$ of the given element, and then compute its corresponding support extension in the tensor-product setting. Notice that the support extension depends on the degree and the regularity of the basis functions, so the hierarchical space structure, which also includes data structures for the tensor-product spaces $V^\ell$, must be given as an input.

**Algorithm 2**: **get_multilevel_support_extension**.
**Description**: Multilevel support extension of an element (or list of elements) $Q$ of level $\ell$, with respect to level $k < \ell$, for a hierarchical space.

**Input:** `MESH`, `SPACE`, $Q, \ell, k$
**if** $(k = \ell)$ **then**
  `extension` $\leftarrow$ **get_support_extension** $(Q, G^\ell, V^\ell)$
**else**
  `ancestors` $\leftarrow$ **get_ancestor_of_cell** $(\texttt{MESH}, Q, \ell, k)$
  `extension` $\leftarrow$ **get_support_extension** $(\texttt{ancestors}, G^k, V^k)$
**end if**
**Output:** `extension`

We can now rigorously define the neighborhood, which will depend on the chosen basis. The definition for the THB-splines case follows [5], while the definition for the standard hierarchical B-splines generalizes the definition in [7] to a general $m$.

**Definition 5.** *Given an element $Q \in \mathcal{Q} \cap \mathcal{G}^\ell$, its $\mathcal{H}$-neighborhood and its $\mathcal{T}$-neighborhood with respect to $m$ are respectively defined as*

$$\begin{aligned}
\mathcal{N}_\mathcal{H}(\mathcal{Q}, Q, m) &:= \left\{ Q' \in \mathcal{G}^{\ell-m+1} : Q' \in S(Q, \ell-m+1) \right\}, \\
\mathcal{N}_\mathcal{T}(\mathcal{Q}, Q, m) &:= \left\{ Q' \in \mathcal{G}^{\ell-m+1} : \exists\, Q'' \in S(Q, \ell-m+2), Q'' \subseteq Q' \right\},
\end{aligned}$$

*when $\ell - m + 1 \geq 0$, and $\mathcal{N}_\mathcal{H}(\mathcal{Q}, Q, m) = \mathcal{N}_\mathcal{T}(\mathcal{Q}, Q, m) = \emptyset$ for $\ell - m + 1 < 0$.*

Notice that, given the level $\ell$ and the admissibility class $m$, the elements of the neighborhood (either $\mathcal{T}$- or $\mathcal{H}$-) have level $\ell - m + 1$. However, since the multilevel support extensions satisfy $S(Q, \ell-m+2) \subset S(Q, \ell-m+1)$, the $\mathcal{T}$-neighborhood is always contained in the $\mathcal{H}$-neighborhood.

Moreover, we note that all the elements in the neighborhood are contained in $\mathcal{Q}$, that is, they are all active. We take advantage of this fact in Algorithms 3 and 4, which detail the computation of the $\mathcal{H}$-neighborhood and the $\mathcal{T}$-neighborhood, respectively.

**Algorithm 3**: **get_H-neighborhood.**
**Description**: $\mathcal{H}$-neighborhood of an element $Q$, of level $\ell$, with respect to the admissibility class $m$.

**Input:** MESH, SPACE, $Q, \ell, m$
$k \leftarrow \ell - m + 1$
**if** $(k < 0)$ **then**
  neighborhood $\leftarrow \emptyset$
**else**
  extension $\leftarrow$ **get_multilevel_support_extension** (MESH, SPACE, $Q, \ell, k$)
  neighborhood $\leftarrow$ extension $\cap \mathbb{E}_k^{\mathbb{A}}$
**end if**
**Output:** neighborhood

**Algorithm 4**: **get_T-neighborhood.**
**Description**: $\mathcal{T}$-neighborhood of an element $Q$, of level $\ell$, with respect to the admissibility class $m$.

**Input:** MESH, SPACE, $Q, \ell, m$
$k \leftarrow \ell - m + 2$
**if** $(k - 1 < 0)$ **then**
  neighborhood $\leftarrow \emptyset$
**else**
  extension $\leftarrow$ **get_multilevel_support_extension** (MESH, SPACE, $Q, \ell, k$)
  parents $\leftarrow$ **get_parent_of_cell** (MESH, extension, $k, k-1$)
  neighborhood $\leftarrow$ parents $\cap \mathbb{E}_{k-1}^{\mathbb{A}}$
**end if**
**Output:** neighborhood

To develop the algorithm for refinement with guaranteed admissible meshes, we also need to define, for $\ell = 0, \ldots, N-1$, the auxiliary subdomains

$$\omega_{\mathcal{H}}^{\ell} := \bigcup \left\{ \overline{Q} : Q \in G^{\ell} \wedge S(Q, \ell - 1) \subseteq \Omega^{\ell} \right\},$$
$$\omega_{\mathcal{T}}^{\ell} := \bigcup \left\{ \overline{Q} : Q \in G^{\ell} \wedge S(Q, \ell) \subseteq \Omega^{\ell} \right\},$$

and it clearly holds that $\omega_{\mathcal{H}}^{\ell} \subseteq \omega_{\mathcal{T}}^{\ell}$. Then, we also need to introduce a different concept of admissibility, which is based on these auxiliary subdomains.

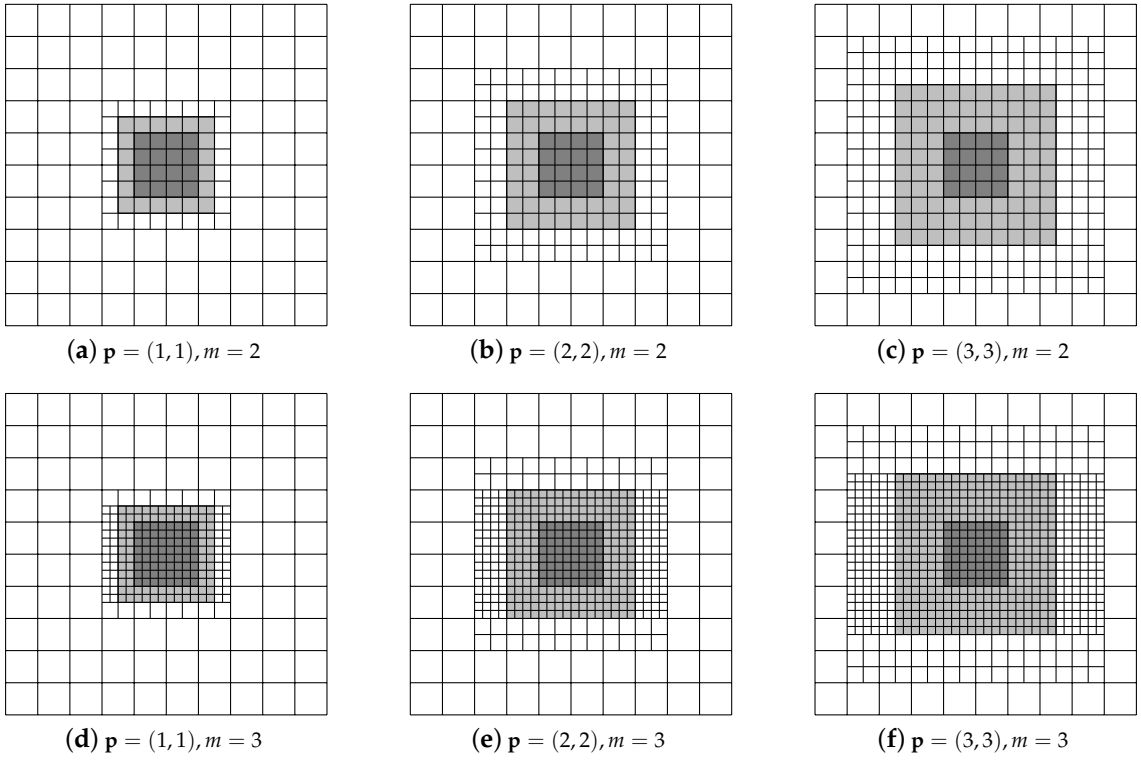
**Definition 6.** *A mesh $\mathcal{Q}$ is strictly $\mathcal{H}$-admissible (respectively, strictly $\mathcal{T}$-admissible) of class m if it holds that*

$$\Omega^{\ell} \subseteq \omega_{\mathcal{H}}^{\ell-m+1}, \qquad (\textit{resp. } \Omega^{\ell} \subseteq \omega_{\mathcal{T}}^{\ell-m+1}),$$

*for $\ell = m, m+1, \ldots, N-1$.*

The definition of the auxiliary subdomains, and their role in strict admissibility, is better understood with the help of Figure 1. We represent $\omega_{\mathcal{H}}^1$ and $\omega_{\mathcal{T}}^1$ for different mesh configurations and degrees, and note that, in all cases $\omega_{\mathcal{H}}^1 \subseteq \omega_{\mathcal{T}}^1$, and the difference becomes bigger with higher degree. Moreover, in the examples of the figure, the subdomains $\Omega^0$ and $\Omega^1$ do not change between the top and bottom row, so also the auxiliary subdomains $\omega_{\mathcal{H}}^1$ and $\omega_{\mathcal{T}}^1$ do not change. Finally, for the degree and the class in the caption of each subfigure, we can refine the highlighted region $\omega_{\mathcal{H}}^1$ (respectively,

$\omega_{\mathcal{T}}^1$), adding one more level and maintaining the strict $\mathcal{H}$-admissibility (resp. strict $\mathcal{T}$-admissibility) of the mesh.



**(a)** $\mathbf{p} = (1,1), m = 2$ **(b)** $\mathbf{p} = (2,2), m = 2$ **(c)** $\mathbf{p} = (3,3), m = 2$

**(d)** $\mathbf{p} = (1,1), m = 3$ **(e)** $\mathbf{p} = (2,2), m = 3$ **(f)** $\mathbf{p} = (3,3), m = 3$

**Figure 1.** The domains $\omega_{\mathcal{H}}^1$ and $\omega_{\mathcal{T}}^1$ are highlighted in dark gray and light gray, respectively, for different mesh configurations.

The relation between the different admissibility classes is also stated in Proposition 1. One of the results of this proposition requires the following assumption:

$$\text{if } Q \in \mathcal{G}^\ell, \text{ then } \exists \beta \in \mathcal{B}^\ell \cap \mathcal{H}(Q) : \operatorname{supp} \beta \cap Q \neq \emptyset. \tag{5}$$

This means that, for any mesh element $Q$, at least one B-spline of the same level of $Q$, whose support overlaps this element, is active in the hierarchical B-spline basis. The assumption may not be satisfied when refining few adjacent elements, as in the example of Figure 2 below.

**Proposition 1.** *Let $\mathcal{Q}$ be a hierarchical mesh. We have the following results:*

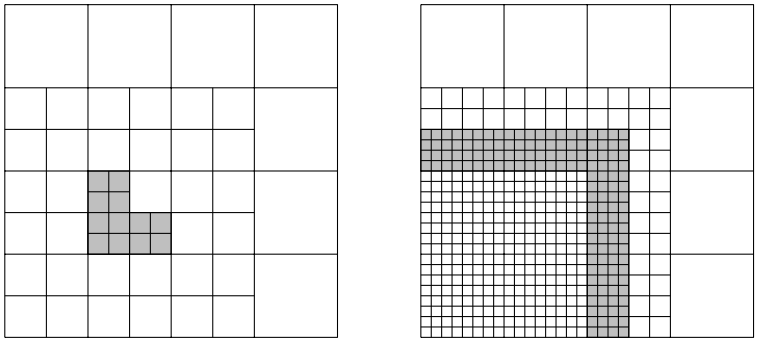
1. *if $\mathcal{Q}$ is strictly $\mathcal{T}$-admissible of class $m$, then it is $\mathcal{T}$-admissible of the class $m$.*
2. *if $\mathcal{Q}$ is strictly $\mathcal{H}$-admissible of class $m$, then it is $\mathcal{H}$-admissible of the class $m$.*
3. *if $\mathcal{Q}$ is (strictly) $\mathcal{H}$-admissible of class $m$, then it is (strictly) $\mathcal{T}$-admissible of the class $m$.*
4. *if $\mathcal{Q}$ is $\mathcal{H}$-admissible of class $m$ and satisfies assumption (5), then it is strictly $\mathcal{H}$-admissible of the class $m$.*

**Proof.** Point 1 was already proved in [5] (Prop. 9). Point 3 is a trivial consequence of Definition 3 and the reduced support of THB-splines for admissible meshes, and a consequence of Definition 6 and the fact that $\omega_{\mathcal{H}}^\ell \subseteq \omega_{\mathcal{T}}^\ell$ for strictly admissible meshes. It remains to prove Points 2 and 4.

Given an active element $Q \in \mathcal{G}^\ell$, by definition of hierarchical B-splines, we know that any finer function in $\mathcal{H}(\mathcal{Q}) \cap \mathcal{B}^k$, with $k > \ell$ vanishing on $Q$. We need to prove that the same is true for any $k \leq \ell - m$. Since $Q \subseteq \Omega^\ell \subseteq \omega_{\mathcal{H}}^{\ell-m+1}$, by definition of $\omega_{\mathcal{H}}^{\ell-m+1}$, all the functions of level $\ell - m$ that do not vanish on $Q$ have support contained in $\Omega^{\ell-m+1}$, and are not active. Noting that $\Omega^\ell \subseteq \Omega^{\ell-j}$, for $0 \leq j \leq \ell$, we can use the same argument for coarser levels. This proves Point 2.

To prove Point 4, let the mesh element $Q \in \mathcal{G}^\ell$ for $\ell = m, m+1, \dots, N-1$. We observe that, under assumption (5), since the mesh is $\mathcal{H}$-admissible, the B-splines in $\mathcal{H}(\mathcal{Q})$ whose support overlaps $Q$ belong to levels $\ell, \ell-1, \dots, \ell-m+1$. Consequently, it does not exist an active B-spline in $\mathcal{H}(\mathcal{Q})$ of level $\ell-m$ whose support overlaps $Q$. The support extension of $Q$ with respect to level $\ell-m$ is then completely contained in $\Omega^{\ell-m+1}$, and since this holds for any active element of level $\ell$, we have $\Omega^\ell \subseteq \omega_{\mathcal{H}}^{\ell-m+1}$, namely the mesh is strictly $\mathcal{H}$-admissible. □

Points 2 and 4 of Proposition 1 prove the equivalence of $\mathcal{H}$-admissibility and strictly $\mathcal{H}$-admissibility under assumption (5). In general, however, if the hierarchical mesh $\mathcal{Q}$ is $\mathcal{H}$-admissible ($\mathcal{T}$-admissible) of class $m$, then it is not necessarily strictly $\mathcal{H}$-admissible (respectively, $\mathcal{T}$-admissible). Counterexamples for both cases are shown in Figure 2, where we highlight the elements in $\mathcal{G}^\ell$, and thus, in $\Omega^\ell$, such that are not contained in $\omega_{\mathcal{H}}^{\ell-m+1}$. Notice that, in the $\mathcal{H}$-admissible mesh, the highlighted elements are precisely those that do not satisfy (5). In the $\mathcal{T}$-admissible case instead, the highlighted elements of level $\ell = 3$ do not belong to $\omega_{\mathcal{T}}^{\ell-m+1} = \omega_{\mathcal{T}}^1$ for $m = 3$.



(**a**) $\mathcal{H}$-admissible mesh for $m = 2$ (**b**) $\mathcal{T}$-admissible mesh for $m = 3$

**Figure 2.** Examples of degree $\mathbf{p} = (2, 2)$ of meshes that are admissible, but not strictly admissible.

The last two algorithms of this section adapt to our setting the recursive refinement algorithm in [5], that, given an admissible hierarchical mesh and a set of marked elements, generates a refined $\mathcal{T}$-admissible mesh. Similar to the usage of cell-arrays in Octave/Matlab, we write $\{\texttt{marked}\}$ when referring to the marked elements of all levels, and $\texttt{marked}_\ell$ when referring to a single level.

The refinement algorithm is presented in Algorithm 5. Given a set of marked (active) elements, in lines 1 to 3, we first enrich this set by a recursive algorithm, explained in detail below, where all active elements in the neighborhood of the marked ones are also marked. Then, in line 4, we refine the hierarchical mesh by refining the updated set of marked elements, and replacing them with their children. This second step can be done as in [11] (Algorithm 2), and is not limited to dyadic refinement. After refining the hierarchical mesh, it is necessary to refine the hierarchical space, that is, to update the list of active basis functions. This last step, which we omit, has been already explained in [11].

**Algorithm 5**: **admissible_refinement**.
**Description**: given a hierarchical mesh and a set of marked elements, generate a refined admissible mesh of class $m$.

**Input:** MESH, SPACE, $\{\texttt{marked}\}$, $m$
1: **for** $\ell = 0, \dots, N-1$ **do**
2: $\quad \{\texttt{marked}\} \leftarrow$ **mark_recursive** (MESH, SPACE, $\{\texttt{marked}\}$, $\ell$, $m$)
3: **end for**
4: MESH $\leftarrow$ **refine_hierarchical_mesh** (MESH, $\{\texttt{marked}\}$) ▷ see Algorithm 2 in [11]
**Output:** MESH

Finally, Algorithm 6 is a recursive algorithm that, given a list of marked elements, adds the elements in their neighborhood (either $\mathcal{T}$ or $\mathcal{H}$) to the list. The difference between the two kinds of admissibility only affects the algorithm in the computation of the neighborhood (see Definition 5), in lines 1 to 5. Recursivity is necessary because, to guarantee admissibility, we must also mark the elements in the neighborhood of the newly marked ones.

As we mentioned above, our algorithm is a simple adaption of the one in [5] for $\mathcal{T}$-admissible meshes, while for $\mathcal{H}$-admissible meshes it generalizes to arbitrary $m \geq 2$ the refinement algorithm in [7] (Algorithm 3.1), which only considers the case $m = 2$. Note that $\mathcal{H}$-admissible refinements for $m \geq 2$ were also considered in [19].

**Algorithm 6**: **mark_recursive**.
**Description**: recursive algorithm to mark the elements in the neighborhood of marked ones.

**Input:** MESH, SPACE, $\{\texttt{marked}\}, \ell, m$
1: **if** $\mathcal{T}$-admissibility **then**
2: neighbors $\leftarrow$ **get_T-neighborhood** $(\texttt{MESH}, \texttt{SPACE}, \texttt{marked}_\ell, \ell, m)$
3: **else if** $\mathcal{H}$-admissibility **then**
4: neighbors $\leftarrow$ **get_H-neighborhood** $(\texttt{MESH}, \texttt{SPACE}, \texttt{marked}_\ell, \ell, m)$
5: **end if**
6: **if** $(\texttt{neighbors} \neq \emptyset)$ **then**
7: $k \leftarrow \ell - m + 1$
8: $\texttt{marked}_k \leftarrow \texttt{marked}_k \cup \texttt{neighbors}$
9: $\{\texttt{marked}\} \leftarrow$ **mark_recursive** $(\texttt{MESH}, \texttt{SPACE}, \{\texttt{marked}\}, k, m)$
10: **end if**
**Output:** $\{\texttt{marked}\}$

The properties of these algorithms were analyzed in [5] and [7,19], for (strictly) $\mathcal{T}$- and $\mathcal{H}$-admissibility, respectively. We report in Lemma 1 and Proposition 2 the proofs of the key results for Algorithm 5, which include both cases in a unified setting.

**Lemma 1.** *Let $\mathcal{Q}$ be a strictly $\mathcal{H}$-admissible (respectively, strictly $\mathcal{T}$-admissible) hierarchical mesh of class $m$, associated with a hierarchical space $V$, let $\mathcal{M} \subseteq \mathcal{Q}$ a set of marked active elements, and let the integer $m \geq 2$. The call to Algorithm 5 in the form*

$$\mathcal{Q}_* = \textbf{admissible\_refinement}(\mathcal{Q}, V, \mathcal{M}, m)$$

*terminates and returns a refined mesh $\mathcal{Q}_*$, such that all elements in $\mathcal{Q}_*$ were already active, or are obtained by a single refinement of an element of $\mathcal{Q}$.*

**Proof.** The algorithm terminates because the recursive algorithm ends when the neighborhood is empty, and this condition is reached in a finite number of steps (see lines 1 to 3 in Algorithms 3 and 4). The elements in the input set $\mathcal{M}$ are all active in $\mathcal{Q}$, and the neighborhood only consists of active elements in $\mathcal{Q}$, hence the output set of **mark_recursive** (Algorithm 6) is a list of active elements in $\mathcal{Q}$. Since no further marking is performed, all elements in $\mathcal{Q}_*$ were already active, or are obtained by a single refinement of an element in $\mathcal{Q}$. □

**Proposition 2.** *Let $\mathcal{Q}$, $V$, $\mathcal{M}$ and $m$ the input arguments of Algorithm 5, as in Lemma 1, where $\mathcal{Q}$ is strictly $\mathcal{H}$-admissible (respectively, strictly $\mathcal{T}$-admissible) of class $m$. Then, the algorithm returns a refined hierarchical mesh $\mathcal{Q}_*$, which is strictly $\mathcal{H}$-admissible (resp. strictly $\mathcal{T}$-admissible) of class $m$.*

**Proof.** The proof follows the same steps for strictly $\mathcal{H}$-admissible and strictly $\mathcal{T}$-admissible meshes, so we will prove both at once, introducing the notation

$$\omega^\ell \equiv \begin{cases} \omega^\ell_{\mathcal{H}} & \text{for strictly } \mathcal{H}\text{-admissible meshes,} \\ \omega^\ell_{\mathcal{T}} & \text{for strictly } \mathcal{T}\text{-admissible meshes,} \end{cases}$$

and remarking any other important difference whenever needed.

Let us denote by $\Omega^\ell_* \supseteq \Omega^\ell$, for $\ell = 0, \dots, N$, the subdomains after refinement. We use the same subindex $(*)$ to identify whatever depends on the subdomains, such as the active elements at each level $\mathcal{G}^\ell_*$ as in Equation (1), and the auxiliary subdomains $\omega^\ell_*$.

Let $Q_* \in \mathcal{G}^\ell_*$, and we have by definition $Q_* \subseteq \Omega^\ell_* \setminus \Omega^{\ell+1}_*$. We need to prove that $Q_* \subseteq \omega^{\ell-m+1}_*$ (with the notation introduced above). There exist two possibilities, depending on whether $Q_*$ was already active or not.

- If $Q_* \in \mathcal{G}^\ell$, then $Q_* \subseteq \Omega^\ell \subseteq \omega^{\ell-m+1}$. Obviously, since $\Omega^k_* \subseteq \Omega^k$ for every $k$, it holds $\omega^{\ell-m+1} \subseteq \omega^{\ell-m+1}_*$ (both for $\mathcal{H}$- and $\mathcal{T}$-admissible), and we obtain the desired result.
- If $Q_* \notin \mathcal{G}^\ell$, by Lemma 1, it is obtained by a single refinement of an element $Q_r \in \mathcal{G}^{\ell-1}$, thus $Q_* \subseteq Q_r$. The definition of multilevel support extension immediately gives, for any $0 \leq j \leq \ell - 1$,

  $$Q_* \subseteq S(Q_*, \ell - j) \subseteq S(Q_r, \ell - j) \subseteq S(Q_r, \ell - j - 1).$$

  Moreover, since $\mathcal{Q}$ is strictly admissible we know that $Q_r \subseteq \omega^{\ell-m}$, which combined with the previous equation and the definition of $\omega^{\ell-m}$ yields

  $$\begin{array}{ll} S(Q_*, \ell - m) \subseteq S(Q_r, \ell - m - 1) \subseteq \Omega^{\ell-m} & \text{if } \mathcal{Q} \text{ is strictly } \mathcal{H}\text{-admissible,} \\ S(Q_*, \ell - m + 1) \subseteq S(Q_r, \ell - m) \subseteq \Omega^{\ell-m} & \text{if } \mathcal{Q} \text{ is strictly } \mathcal{T}\text{-admissible.} \end{array}$$

  According to this, let us introduce $k = m$ for strictly $\mathcal{H}$-admissible meshes, and $k = m - 1$ for strictly $\mathcal{T}$-admissible meshes. Since $Q_r$ was a marked element (either $Q_r \in \mathcal{M}$, or it was marked during the recursive marking), it has been used as an input for **mark_recursive** (Algorithm 6), and as a consequence all the active elements in $\mathcal{G}^{\ell-k-1} \cap S(Q_r, \ell - k - 1)$ have been marked. Hence, $Q_* \subseteq S(Q_*, \ell - k) \subseteq \Omega^{\ell-m+1}_*$, and by definition of $\omega^{\ell-m+1}_*$ the result is proved.

□

Proposition 2 guarantees that the strictly admissible nature of the meshes is preserved by Algorithm 5 during the iterative marking and refinement processes of the adaptive loop. Several examples of strictly $\mathcal{H}$- and $\mathcal{T}$-admissible meshes obtained with this algorithm will be shown in Section 5.

**Remark 1.** *It is important to note that, since the two bases span the same hierarchical space, one could apply the strictly $\mathcal{T}$-admissible refinement with the standard HB-splines (or the strictly $\mathcal{H}$-admissible refinement with THB-splines). In general, the admissibility property is not valid for HB-splines on $\mathcal{T}$-admissible meshes, while it is always valid for THB-splines on $\mathcal{H}$-admissible meshes, as already stated in Proposition 1. The lack of the admissibility property has a negative impact on the sparsity of the matrix, as we will see in the numerical tests of the following section.*

**Remark 2.** *The linear complexity of Algorithm 6 was proved in [20] for strictly $\mathcal{T}$-admissible meshes, and, subsequently, in [7] and [19] for strictly $\mathcal{H}$-admissible meshes of class $m = 2$ and $m \geq 2$, respectively. The complexity estimates provide an upper bound for the number of elements generated by the adaptive strategy with respect to the number of elements marked for refinement. These results are in line with the estimates obtained in the context of adaptive finite element methods [21,22]. Note that the complexity analysis of the refinement algorithms is a fundamental ingredient to prove the optimality of hierarchical isogeometric methods [6,7].*

## 4. Adaptive Isogeometric Methods

The hierarchical spaces are a natural choice for the definition of adaptive isogeometric methods [5–7], which are usually based on the adaptive loop

$$\text{SOLVE} \longrightarrow \text{ESTIMATE} \longrightarrow \text{MARK} \longrightarrow \text{REFINE}.$$

In order to illustrate the framework of such kind of methods, let us consider the Poisson model problem with Dirichlet boundary conditions

$$\begin{cases} -\Delta u = f & \text{in } \widehat{\Omega}, \\ u = g & \text{on } \partial\widehat{\Omega}, \end{cases} \tag{6}$$

where $\widehat{\Omega} = \mathbf{F}(\Omega)$ is the physical domain, parametrized by the mapping $\mathbf{F} : \Omega \to \widehat{\Omega}$.

To *solve* the model problem, we consider the hierarchical spline space $V := \text{span}\{\mathcal{H}(\mathcal{Q})\} = \text{span}\{\mathcal{T}(\mathcal{Q})\}$, which gives the discretization space $\widehat{V} := \text{span}\{\beta \circ \mathbf{F}^{-1} : \beta \in V\}$. Then, we determine the solution $u_h$ of the discretized problem (in its variational form)

$$\int_{\widehat{\Omega}} \nabla u_h \cdot \nabla \widehat{v}_h = \int_{\widehat{\Omega}} f\, \widehat{v}_h, \qquad \forall \widehat{v}_h \in \widehat{V}_0 := \{\widehat{w}_h \in \widehat{V} : \widehat{w}_h|_{\partial\widehat{\Omega}} = 0\}, \tag{7}$$

where $u_h = u_0 + u_g$, with $u_0 \in \widehat{V}_0$ and $u_g \in \widehat{V}$ a lifting of an approximation of the boundary function, such that $u_g|_{\partial\widehat{\Omega}} \approx g$, that we obtain by an $L^2$-projection (see [16]). Note that (7) leads to a linear system whose structure depends on the choice of $\mathcal{H}(\mathcal{Q})$ or $\mathcal{T}(\mathcal{Q})$ as basis of $V$.

To *estimate* the error, and assuming that the basis functions are at least $C^1$ continuous, we employ the following element-based residual error estimator [5]. For any $Q \in \mathcal{G}$, the estimator is defined as

$$\varepsilon_Q(u_h) = h_Q \left( \int_{\mathbf{F}(Q)} |f + \Delta u_h|^2 \right)^{1/2}, \tag{8}$$

where $h_Q := \text{diam}\,(\mathbf{F}(Q))$. We note that other estimators, such as the function based residual estimator in [23], or recovery-based error estimators [24,25] could also be used. At each iteration of the adaptive loop, we compute the initial set of *marked* elements by applying Dörfler's marking strategy [26] with the indicator (8). Finally, in order to get admissible meshes, we *refine* applying the Algorithms of Section 3.

Note that this framework can be easily extended to multipatch configurations with $C^0$ continuity. Let us assume that the physical domain is composed of several patches $\widehat{\Omega} = \cup_i \widehat{\Omega}_i$, each with its own parametrization $\mathbf{F}_i : \Omega \to \widehat{\Omega}_i$, which overall give a $C^0$ parametrization. Defining a tensor-product space on each patch with the restrictions of having coinciding knot vectors and control points at the interfaces is enough to get a $C^0$ spline space [1].

Then, multipatch $C^0$ hierarchical spline spaces are obtained with the same construction presented in Section 2.1, simply by considering multipatch $C^0$ spline spaces as elements of the sequence $V^0 \subset V^1 \subset \ldots \subset V^{N-1}$, that is, we have a $C^0$ multipatch space for each level. The refinement algorithms described in Section 3 do not need any significant modification: only the support extensions, which determine the neighborhoods of the cells, are modified according to the different supports of the multipatch $C^0$ basis functions (see [11] (Section 3.4) and [27] for more details).

Since the basis functions are only $C^0$ across the interfaces, the element-based estimator must take into account the jump of the normal derivative, that is, for any $Q \in \mathcal{G}$, the estimator is

$$\varepsilon_Q(u_h) = \left( h_Q^2 \int_{\mathbf{F}(Q)} |f + \Delta u_h|^2 + h_Q \sum_{i \neq j} \int_{\Gamma_{ij} \cap \mathbf{F}(Q)} |\nabla u_h \cdot \mathbf{n}|^2 \right)^{1/2}, \tag{9}$$

where $\Gamma_{ij} := \partial\widehat{\Omega}_i \cap \partial\widehat{\Omega}_j$ denotes the interface between two patches, and $\mathbf{n}$ is the unit normal vector to $\Gamma_{ij}$.

**Remark 3.** *The admissible refinement algorithm can be easily extended to a posteriori estimators based on functions. In this case, one would first mark the elements in the support of the marked functions (see Algorithms 1 and 10 in [11]), and then apply the recursive marking of Algorithm 5 before refining the mesh.*

## 5. Numerical Results

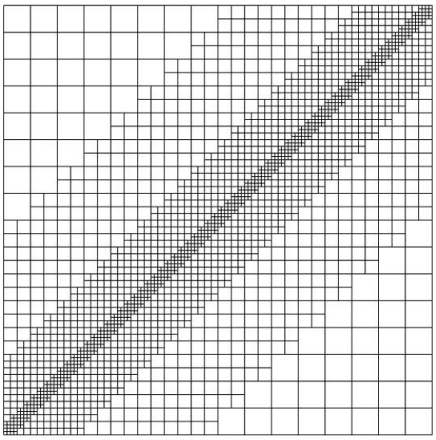
In this section, we present some numerical examples to show the effectivity of the proposed algorithms, and to compare the different admissibility classes presented in the previous sections. The first numerical test consists of an ad hoc refinement of the unit square, while, in the second numerical test, we perform automatic adaptivity for a Poisson problem with singular solution.

### *5.1. Diagonal Refinement of the Unit Square*

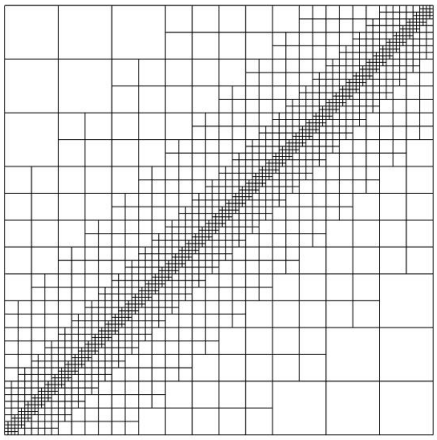
In the first numerical test, we study how the choice of the basis and the admissibility class may affect the matrix of the linear system arising in the isogeometric method. In particular, we are interested in the sparsity pattern and the condition numbers of the matrices.

For our numerical tests, we have chosen a diagonal refinement of the unit square, similar to the one used in [13], as it gives a good compromise between locality of the refinement and an increasing number of basis functions at each step. More precisely, we start from a $4 \times 4$ mesh, and at each step refine a strip of $2\left\lceil \frac{p+1}{2} \right\rceil - 1$ cells centered at the diagonal (compared to $2p+1$ cells in [13]), which ensures that at each step we add functions of the finest level. The meshes obtained after six levels of refinement are shown in Figures 3–5 for degrees two, three and four, respectively. The degrees of freedom (DOFs) associated to the different meshes are also indicated.

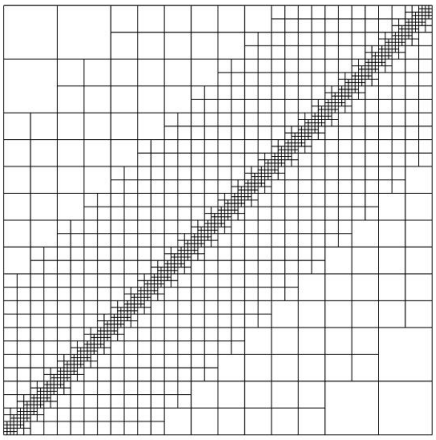
In Tables 1–3, we show the number of nonzeros of the stiffness matrix, and its percentage with respect to the global size of the matrix, after ten refinement steps, for HB-splines and THB-splines considering degrees $\mathbf{p} \equiv (p,p) = (2,2),(3,3),(4,4)$, and for strictly $\mathcal{H}$-admissible and strictly $\mathcal{T}$-admissible hierarchical meshes of classes $m = 2,3,4,\infty$, where $m = \infty$ corresponds to refining only the marked elements of the finest level. The reduced support of THB-splines always reduces the number of nonzero entries compared to HB-splines, and this reduction is more evident for $\mathcal{T}$-admissible meshes and for high values of $m$. For $\mathcal{H}$-admissible meshes, instead, the reduction is not so significant, as the number of functions acting on one element is bounded for HB-splines. We also point out that higher values of $m$ increase the number of nonzero entries with respect to the global size of the matrix, especially for high degree.



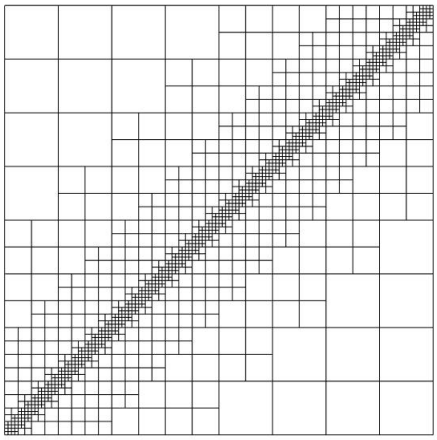
(**a**) $m = 2$, $\mathcal{H}$-admissible (2030 DOFs)



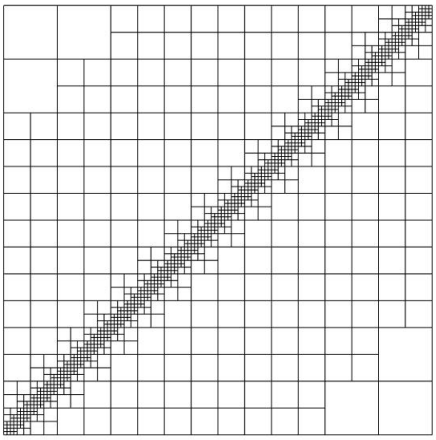
(**b**) $m = 2$, $\mathcal{T}$-admissible (1420 DOFs)



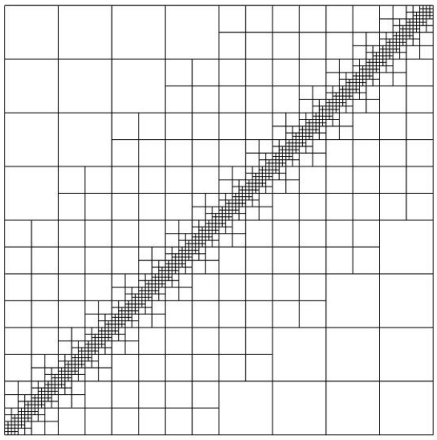
(**c**) $m = 3$, $\mathcal{H}$-admissible (1120 DOFs)



(**d**) $m = 3$, $\mathcal{T}$-admissible (908 DOFs)

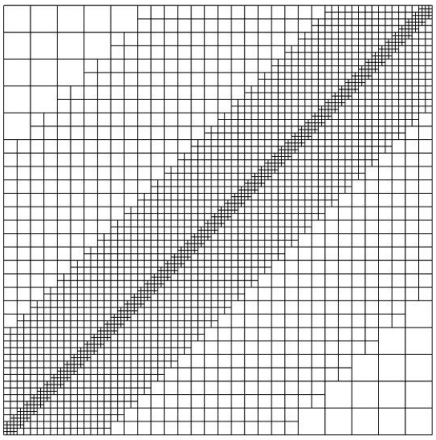


(**e**) $m = 4$, $\mathcal{H}$-admissible (774 DOFs)

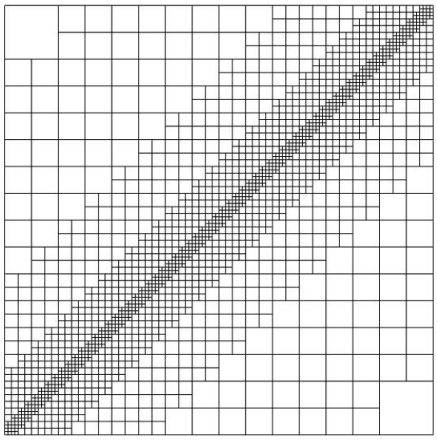


(**f**) $m = 4$, $\mathcal{T}$-admissible (716 DOFs)

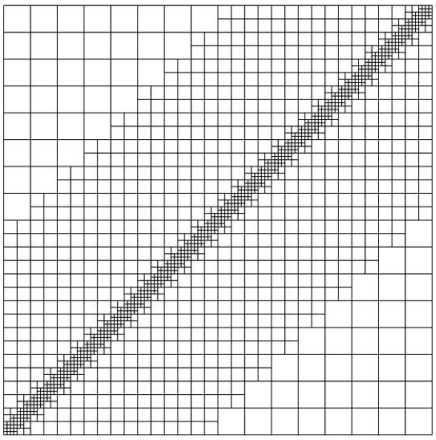
**Figure 3.** Hierarchical meshes obtained with $\mathbf{p} = (2, 2)$ and $m = 2, 3, 4$ (from **top** to **bottom**), by using $\mathcal{H}$-admissible (**left**) and $\mathcal{T}$-admissible (**right**) meshes.



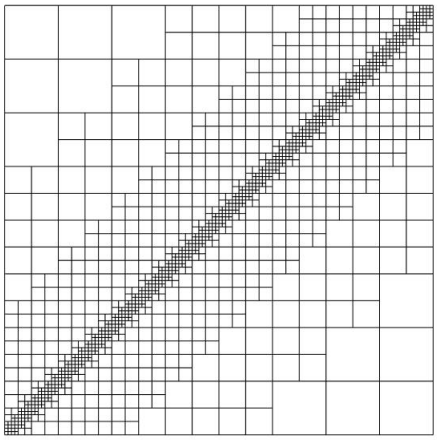
(**a**) $m = 2$, $\mathcal{H}$-admissible (2336 DOFs)



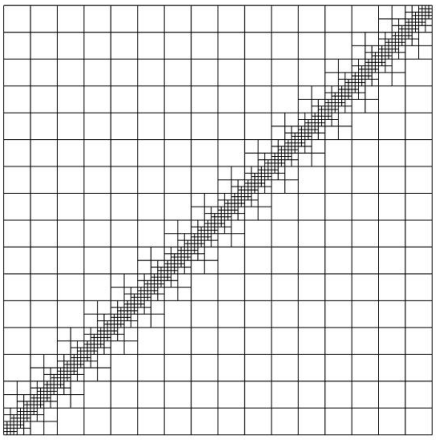
(**b**) $m = 2$, $\mathcal{T}$-admissible (1514 DOFs)



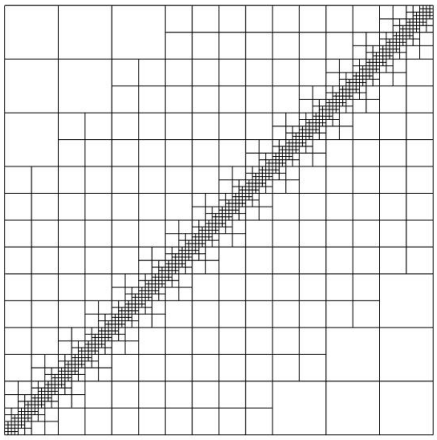
(**c**) $m = 3$, $\mathcal{H}$-admissible (1051 DOFs)



(**d**) $m = 3$, $\mathcal{T}$-admissible (763 DOFs)

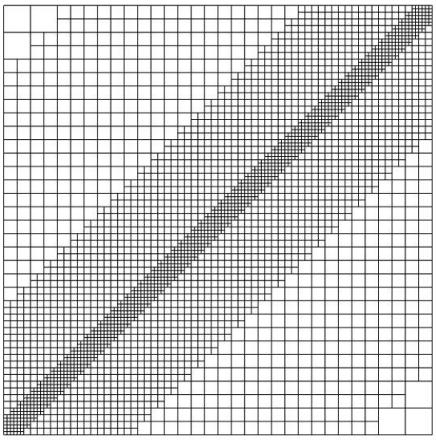


(**e**) $m = 4$, $\mathcal{H}$-admissible (536 DOFs)

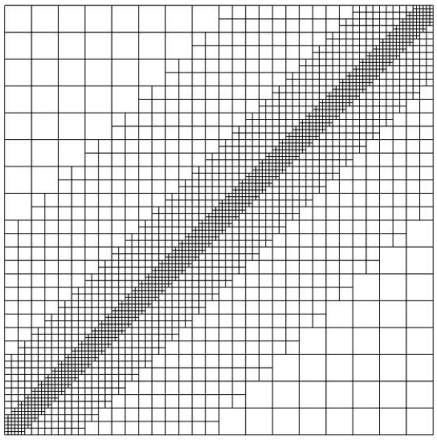


(**f**) $m = 4$, $\mathcal{T}$-admissible (464 DOFs)

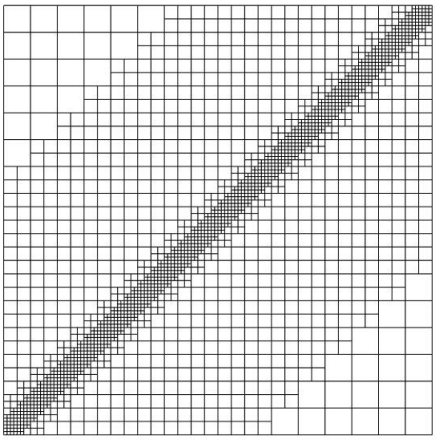
**Figure 4.** Hierarchical meshes obtained with $\mathbf{p} = (3, 3)$ and $m = 2, 3, 4$ (from **top** to **bottom**), by using $\mathcal{H}$-admissible (**left**) and $\mathcal{T}$-admissible (**right**) meshes.



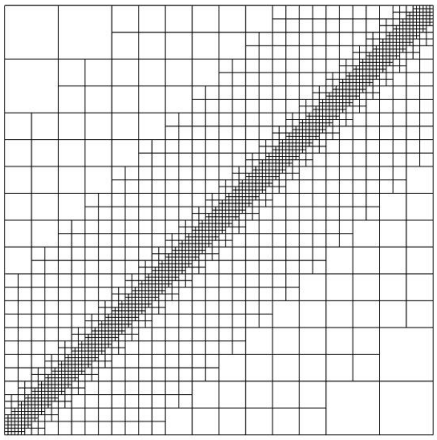
(**a**) $m = 2$, $\mathcal{H}$-admissible (2998 DOFs)



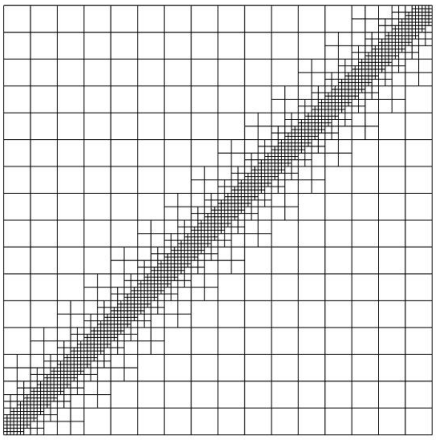
(**b**) $m = 2$, $\mathcal{T}$-admissible (2052 DOFs)



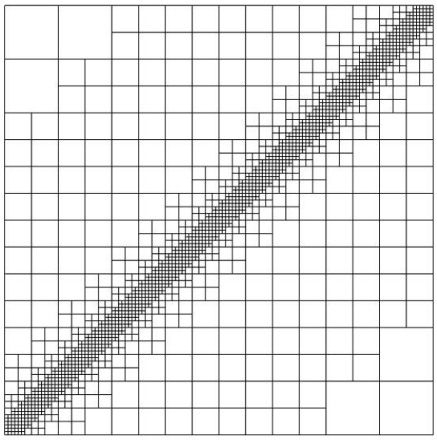
(**c**) $m = 3$, $\mathcal{H}$-admissible (1534 DOFs)



(**d**) $m = 3$, $\mathcal{T}$-admissible (1216 DOFs)



(**e**) $m = 4$, $\mathcal{H}$-admissible (962 DOFs)



(**f**) $m = 4$, $\mathcal{T}$-admissible (912 DOFs)

**Figure 5.** Hierarchical meshes obtained with $\mathbf{p} = (4, 4)$ and $m = 2, 3, 4$ (from **top** to **bottom**), by using $\mathcal{H}$-admissible (**left**) and $\mathcal{T}$-admissible (**right**) meshes.

**Table 1.** Number of nonzeros of the stiffness matrix for $\mathbf{p} = (2,2)$.

| | | DOFs | HB | (%) | THB | (%) | THB/HB |
|---|---|---|---|---|---|---|---|
| | $m = \infty$ | 8228 | 808,628 | (1.19) | 542,548 | (0.80) | 0.67 |
| $\mathcal{H}$-admissible | $m = 2$ | 40,058 | 1,248,786 | (0.08) | 1,099,583 | (0.07) | 0.88 |
| $\mathcal{H}$-admissible | $m = 3$ | 21,028 | 749,616 | (0.17) | 627,864 | (0.14) | 0.84 |
| $\mathcal{H}$-admissible | $m = 4$ | 14,106 | 589,834 | (0.30) | 476,115 | (0.24) | 0.81 |
| $\mathcal{T}$-admissible | $m = 2$ | 24,200 | 990,728 | (0.17) | 706,113 | (0.12) | 0.71 |
| $\mathcal{T}$-admissible | $m = 3$ | 14,664 | 898,652 | (0.42) | 478,963 | (0.22) | 0.53 |
| $\mathcal{T}$-admissible | $m = 4$ | 11,360 | 714,736 | (0.55) | 412,453 | (0.32) | 0.58 |

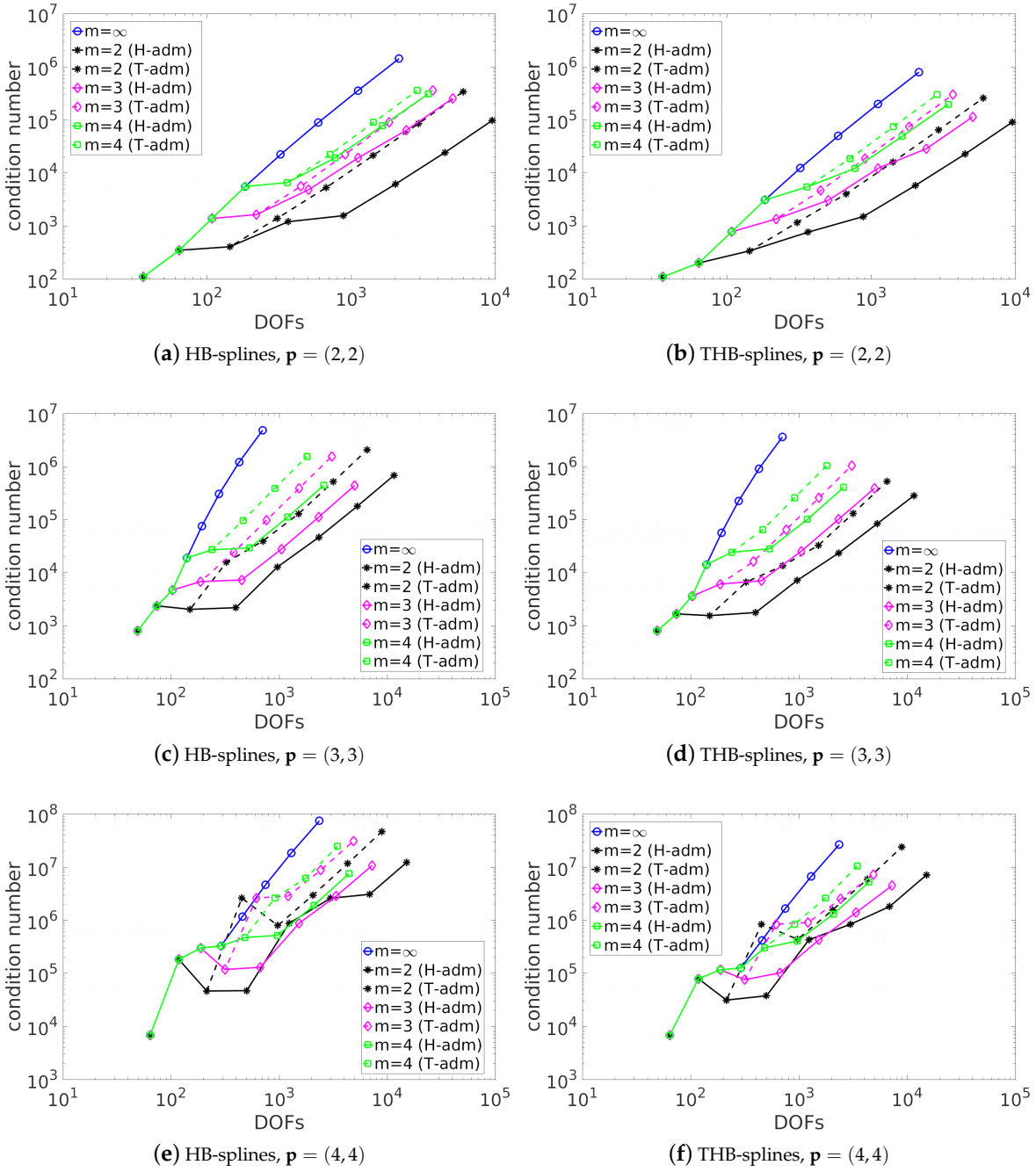
**Table 2.** Number of nonzeros of the stiffness matrix for $\mathbf{p} = (3,3)$.

| | | DOFs | HB | (%) | THB | (%) | THB/HB |
|---|---|---|---|---|---|---|---|
| | $m = \infty$ | 2186 | 156,764 | (3.28) | 122,728 | (2.57) | 0.78 |
| $\mathcal{H}$-admissible | $m = 2$ | 49,940 | 2,941,926 | (0.12) | 2,620,770 | (0.11) | 0.89 |
| $\mathcal{H}$-admissible | $m = 3$ | 21,227 | 1,318,125 | (0.29) | 1,118,981 | (0.25) | 0.85 |
| $\mathcal{H}$-admissible | $m = 4$ | 11,064 | 674,020 | (0.55) | 571,544 | (0.47) | 0.85 |
| $\mathcal{T}$-admissible | $m = 2$ | 26,554 | 2,087,894 | (0.30) | 1,486,588 | (0.21) | 0.71 |
| $\mathcal{T}$-admissible | $m = 3$ | 12,107 | 1,466,741 | (1.00) | 709,261 | (0.48) | 0.48 |
| $\mathcal{T}$-admissible | $m = 4$ | 7020 | 746,362 | (1.51) | 392,128 | (0.80) | 0.53 |

**Table 3.** Number of nonzeros of the stiffness matrix for $\mathbf{p} = (4,4)$.

| | | DOFs | HB | (%) | THB | (%) | THB/HB |
|---|---|---|---|---|---|---|---|
| | $m = \infty$ | 8446 | 1,819,856 | (2.55) | 1,410,796 | (1.98) | 0.78 |
| $\mathcal{H}$-admissible | $m = 2$ | 66,390 | 6,548,354 | (0.15) | 5,885,286 | (0.13) | 0.90 |
| $\mathcal{H}$-admissible | $m = 3$ | 31,112 | 3,299,540 | (0.34) | 2,861,116 | (0.30) | 0.87 |
| $\mathcal{H}$-admissible | $m = 4$ | 18,778 | 2,075,442 | (0.59) | 1,805,250 | (0.51) | 0.87 |
| $\mathcal{T}$-admissible | $m = 2$ | 36,516 | 4,819,354 | (0.36) | 3,499,152 | (0.26) | 0.73 |
| $\mathcal{T}$-admissible | $m = 3$ | 19,412 | 3,613,896 | (0.96) | 2,002,780 | (0.53) | 0.55 |
| $\mathcal{T}$-admissible | $m = 4$ | 13,456 | 2,773,686 | (1.53) | 1,437,096 | (0.79) | 0.52 |

In Figures 6 and 7, we show the computations of the condition number for the mass and the stiffness matrix, respectively, the latter computed after applying Dirichlet homogeneous boundary conditions. The results in Figure 6 show that, for the mass matrix, THB-splines get a lower condition number than HB-splines. Moreover, $\mathcal{H}$-admissible meshes give lower condition numbers than the corresponding $\mathcal{T}$-admissible ones, and lower values of $m$ also produce lower condition numbers, which suggests that limiting the interaction between levels reduces the condition number of the mass matrix. These conclusions cannot be applied to the stiffness matrix. Indeed, the results of Figure 7 do not show any clear advantage of any option, neither for the chosen basis, nor for the admissibility class. We remark that for this particular refinement HB-splines in non-admissible meshes surprisingly provide the lowest condition numbers, which is completely opposite to the behavior for the mass matrix. It should be mentioned that a similar study for THB-splines defined on general (non admissible) hierarchical meshes and strictly admissible meshes of class 2 was already presented in [28]. In that case, there were some advantages on admissible meshes also for the condition number of the stiffness matrix. A better understanding of the problem would require a deeper investigation, which is behind the scope of the present paper.

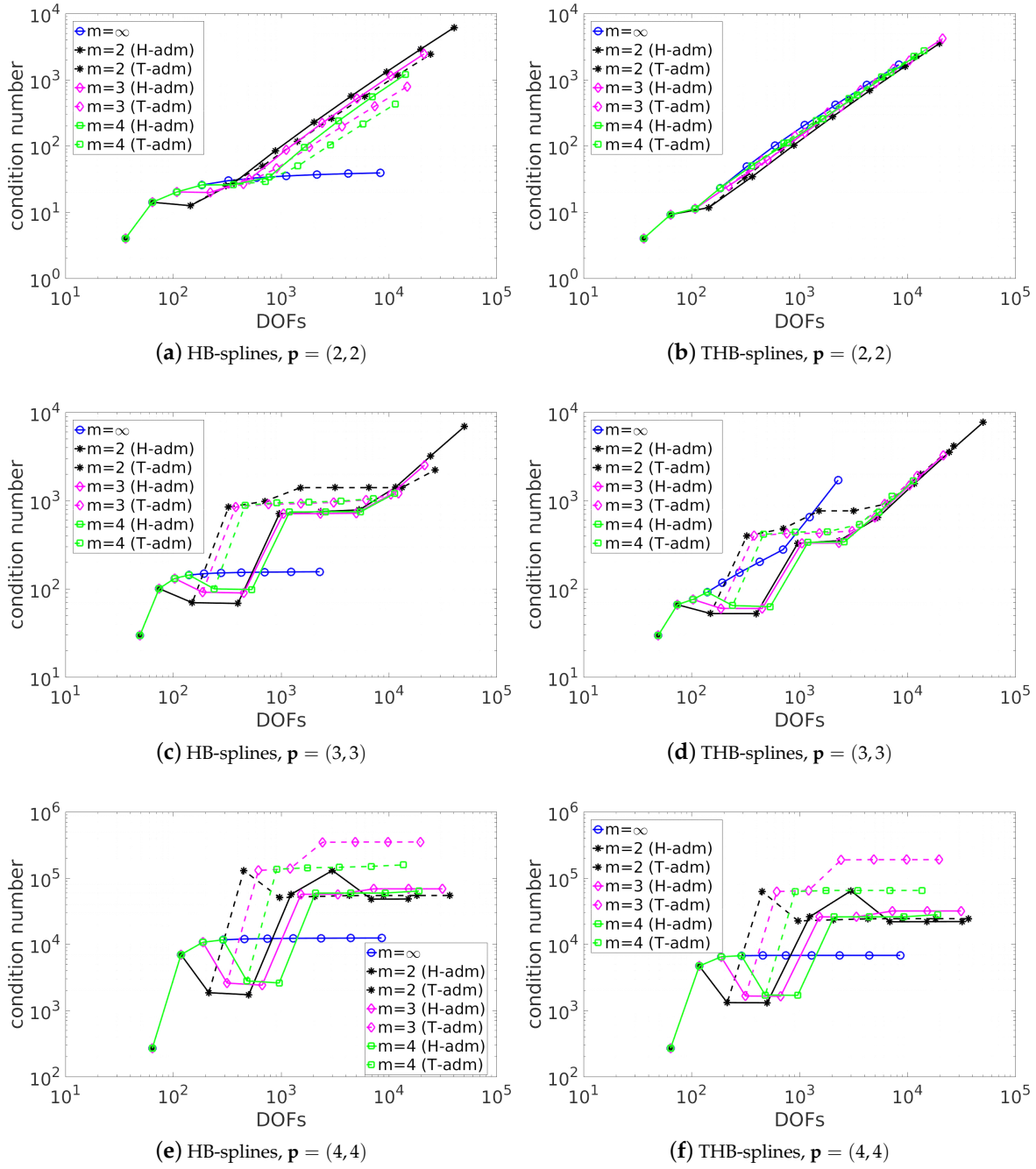



(**a**) HB-splines, $\mathbf{p} = (2,2)$ (**b**) THB-splines, $\mathbf{p} = (2,2)$

(**c**) HB-splines, $\mathbf{p} = (3,3)$ (**d**) THB-splines, $\mathbf{p} = (3,3)$

(**e**) HB-splines, $\mathbf{p} = (4,4)$ (**f**) THB-splines, $\mathbf{p} = (4,4)$

**Figure 6.** Condition numbers of the mass matrix for HB-splines and THB-splines, for different admissibility classes and different values of the degree.




**(a)** HB-splines, $\mathbf{p} = (2,2)$ **(b)** THB-splines, $\mathbf{p} = (2,2)$

**(c)** HB-splines, $\mathbf{p} = (3,3)$ **(d)** THB-splines, $\mathbf{p} = (3,3)$

**(e)** HB-splines, $\mathbf{p} = (4,4)$ **(f)** THB-splines, $\mathbf{p} = (4,4)$

**Figure 7.** Condition numbers of the stiffness matrix for HB-splines and THB-splines, for different admissibility classes and different values of the degree.

### *5.2. Adaptive Method*

For our second numerical test, we present the results obtained by applying the adaptive isogeometric methods described in Section 4 to the model problem (6) where $f = 0$ and $g$ is the restriction to $\partial\widehat{\Omega}$ of the exact solution

$$u(\rho, \phi) = \rho^{2/3} \sin(2\phi/3),$$

defined in polar coordinates on the curved L-shaped domain shown in Figure 8, which is formed by three patches.

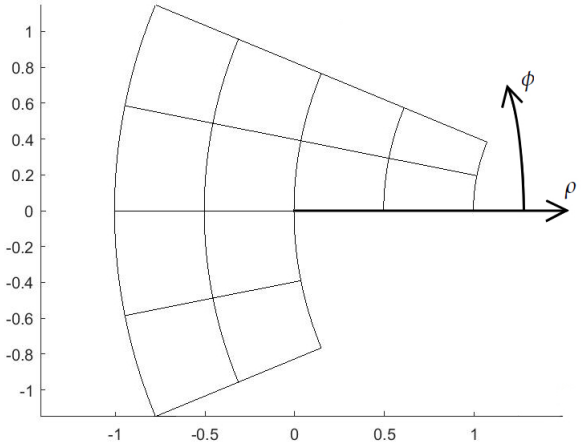
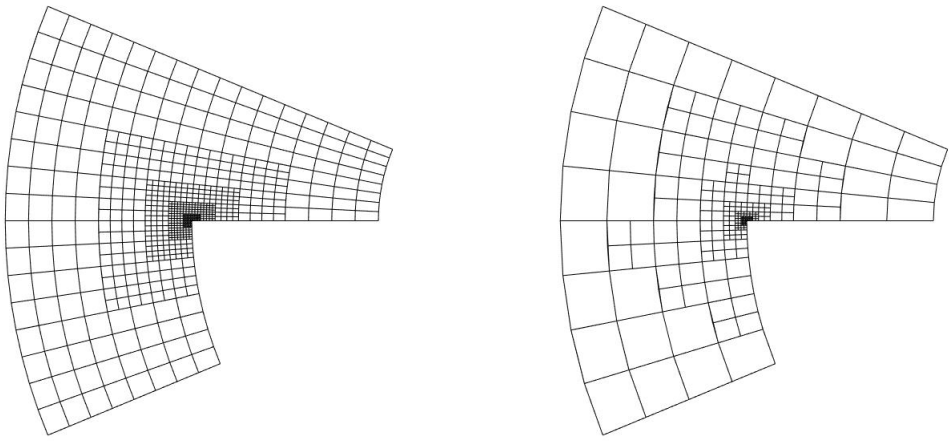


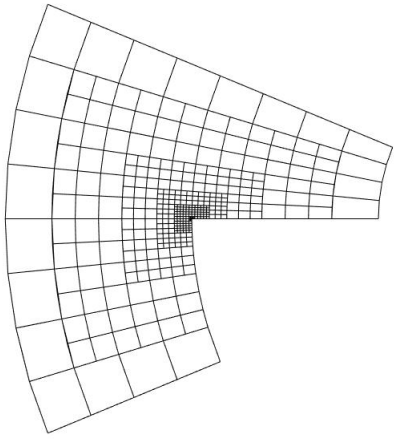

**Figure 8.** Curved L-shaped domain with the initial mesh mapped on it.

We apply the method with degrees $\mathbf{p} \equiv (p, p) = (2,2), (3,3), (4,4)$ and continuity $C^{p-1}$ inside each patch, and $C^0$ continuity across the interfaces, using classes of $\mathcal{H}$-admissibility and $\mathcal{T}$-admissibility $m = 2, 3, 4, \infty$, where $m = \infty$ corresponds to pure Dörfler's marking, without later applying the recursive marking of Section 3. In all of the cases, the inital mesh is a 3-patch mesh with a $2 \times 2$ mesh on each patch, and we set a limit of maximum $n = 8$ hierarchical levels. For the Dörfler's marking strategy, we set the value of the parameter $\theta = 0.90$. Figures 9–11 show, for each degree and for each class of admissibility, the differences between $\mathcal{H}$-admissibility and $\mathcal{T}$-admissibility. These figures clearly show how the choice of the parameter $m$ influences the grading of the mesh. As expected, $\mathcal{H}$-admissibility produces more refined meshes than $\mathcal{T}$-admissibility because the $\mathcal{H}$-neighborhood always contains the $\mathcal{T}$-neighborhood.

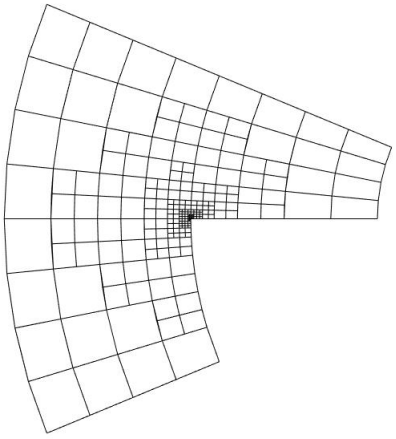


(**a**) $m = 2$, $\mathcal{H}$-admissible (770 DOFs) (**b**) $m = 2$, $\mathcal{T}$-admissible (278 DOFs)

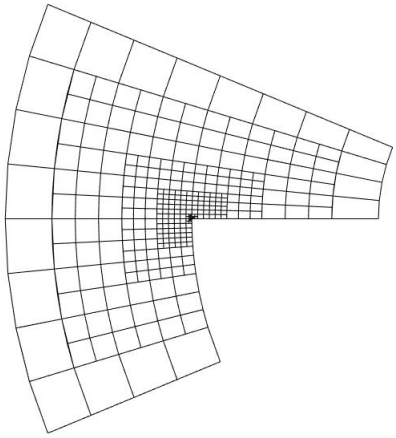
**Figure 9.** *Cont.*



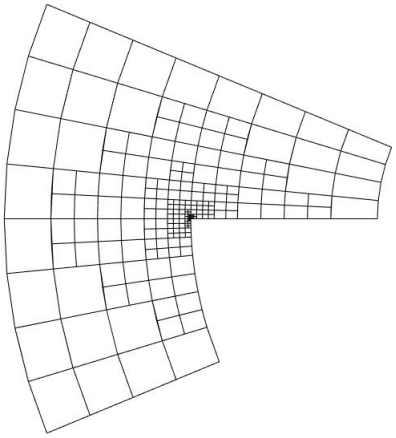
(**c**) $m = 3$, $\mathcal{H}$-admissible (410 DOFs)



(**d**) $m = 3$, $\mathcal{T}$-admissible (258 DOFs)

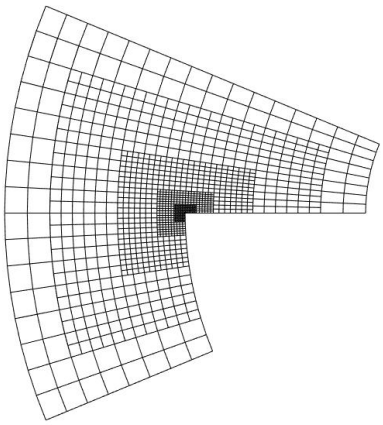


(**e**) $m = 4$, $\mathcal{H}$-admissible (344 DOFs)

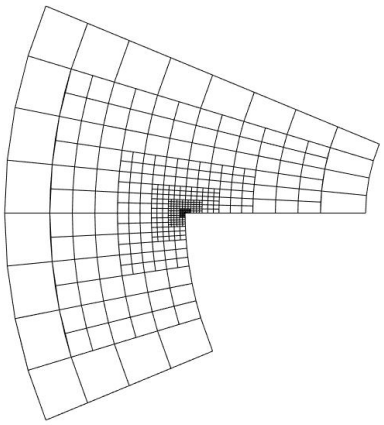


(**f**) $m = 4$, $\mathcal{T}$-admissible (241 DOFs)

**Figure 9.** Hierarchical meshes obtained with $\mathbf{p} = (2, 2)$ and $m = 2, 3, 4$ (from **top** to **bottom**), by using $\mathcal{H}$-admissible (**left**) and $\mathcal{T}$-admissible (**right**) meshes.

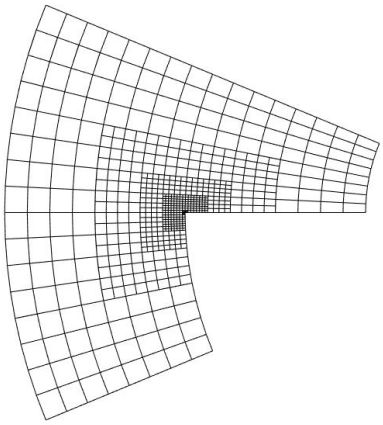


(**a**) $m = 2$, $\mathcal{H}$-admissible (1323 DOFs)

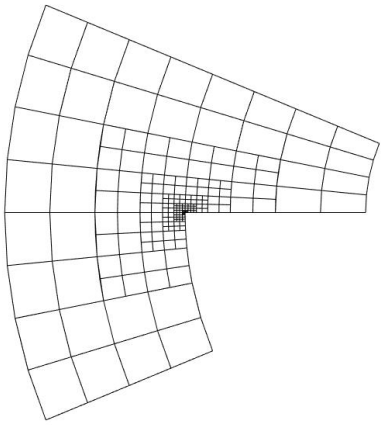


(**b**) $m = 2$, $\mathcal{T}$-admissible (472 DOFs)

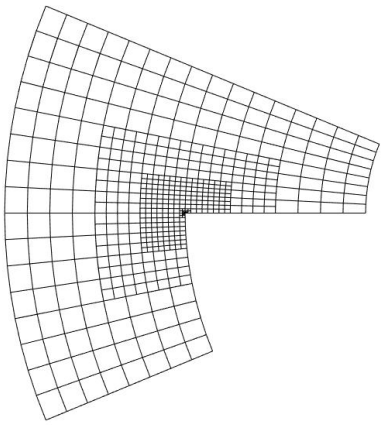
**Figure 10.** *Cont.*



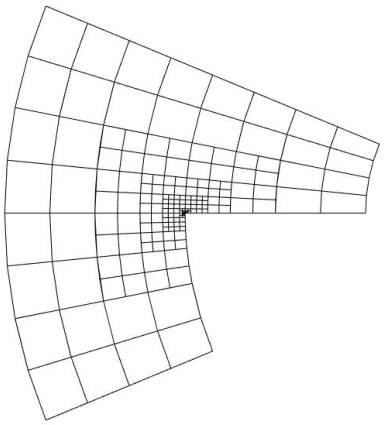
(**c**) $m = 3$, $\mathcal{H}$-admissible (763 DOFs)



(**d**) $m = 3$, $\mathcal{T}$-admissible (275 DOFs)

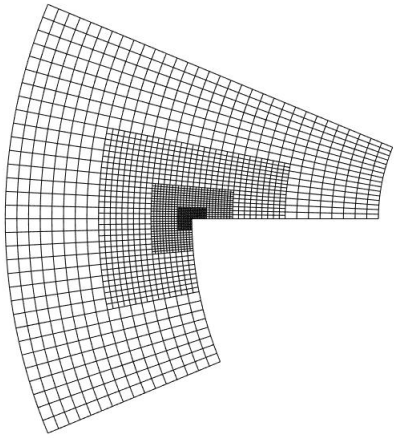


(**e**) $m = 4$, $\mathcal{H}$-admissible (634 DOFs)

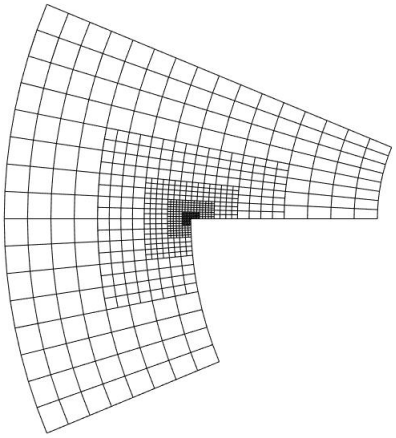


(**f**) $m = 4$, $\mathcal{T}$-admissible (250 DOFs)

**Figure 10.** Hierarchical meshes obtained with $\mathbf{p} = (3, 3)$ and $m = 2, 3, 4$ (from **top** to **bottom**), by using $\mathcal{H}$-admissible (**left**) and $\mathcal{T}$-admissible (**right**) meshes.

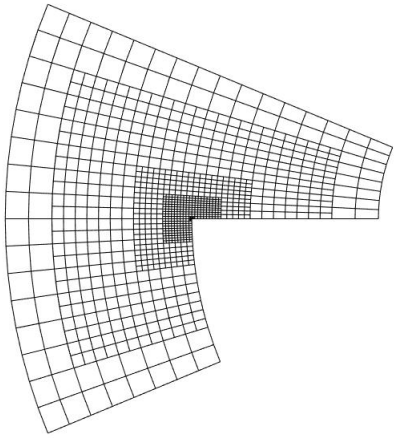


(**a**) $m = 2$, $\mathcal{H}$-admissible (2369 DOFs)

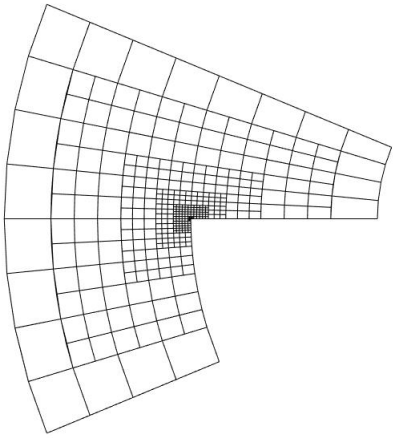


(**b**) $m = 2$, $\mathcal{T}$-admissible (898 DOFs)

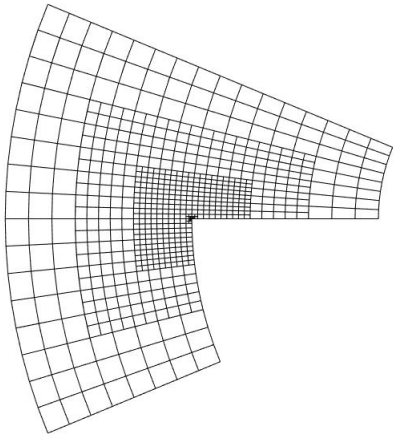
**Figure 11.** *Cont.*



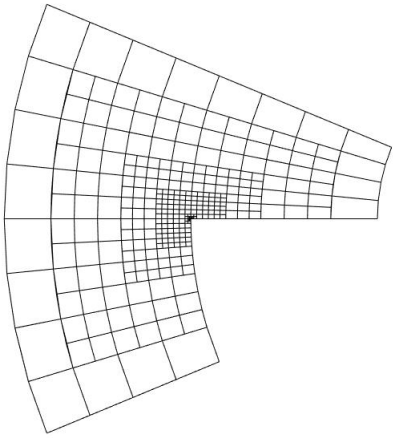
**(c)** $m = 3$, $\mathcal{H}$-admissible (1164 DOFs)



**(d)** $m = 3$, $\mathcal{T}$-admissible (490 DOFs)



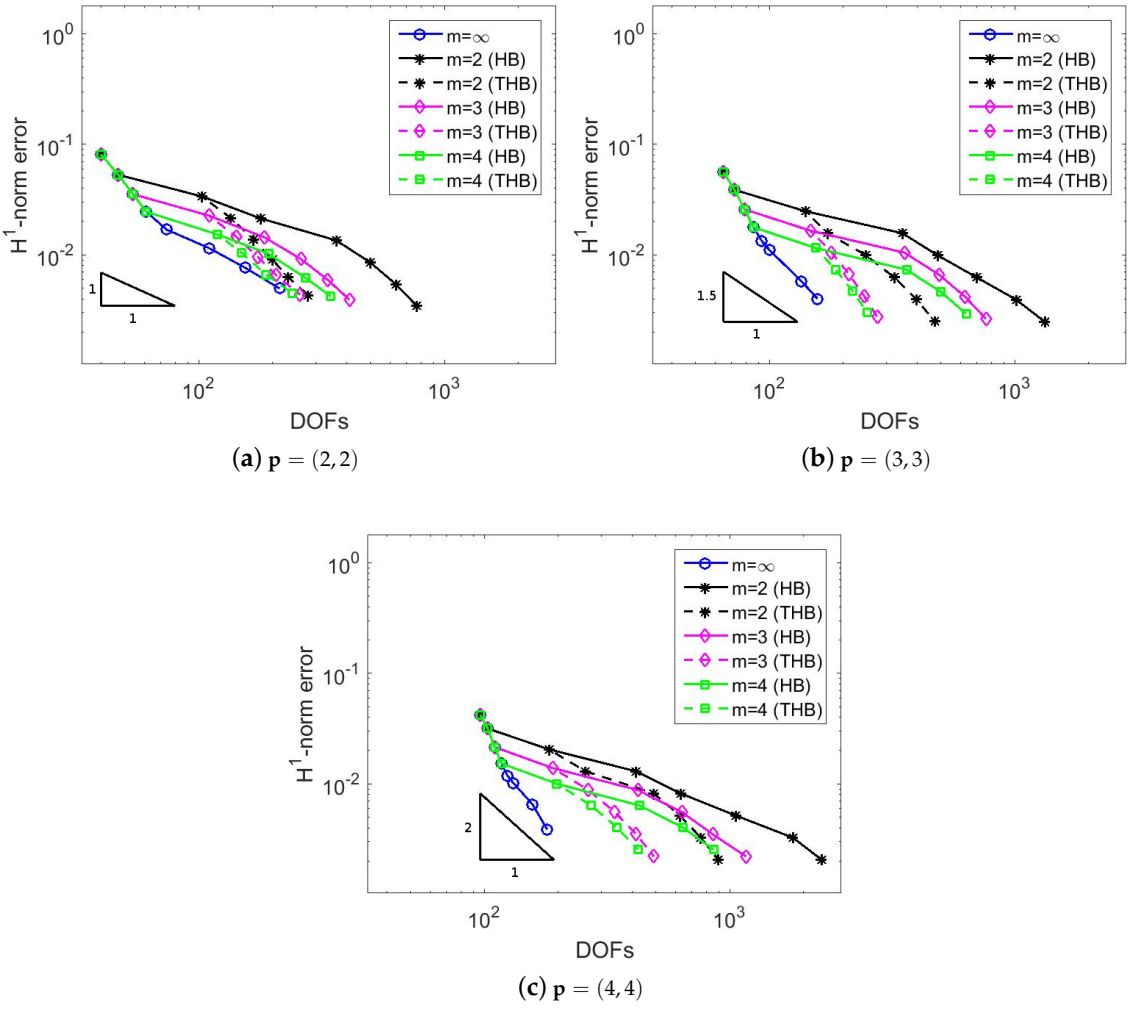
**(e)** $m = 4$, $\mathcal{H}$-admissible (859 DOFs)



**(f)** $m = 4$, $\mathcal{T}$-admissible (422 DOFs)

**Figure 11.** Hierarchical meshes obtained with $\mathbf{p} = (4, 4)$ and $m = 2, 3, 4$ (from **top** to **bottom**), by using $\mathcal{H}$-admissible (**left**) and $\mathcal{T}$-admissible (**right**) meshes.

The importance of the admissibility class is more evident in Figure 12, where we show the convergence of the error in $H^1$-norm with respect to the number of degrees of freedom. While both $\mathcal{H}$-admissible and $\mathcal{T}$-admissible meshes provide optimal convergence rates, the $\mathcal{T}$-admissible ones require a lower number of degrees of freedom to obtain the same error. This difference between the two classes is particularly evident for higher degree of the basis functions. Obviously, when no additional refinement is considered ($m = \infty$) the refinement is even more localized, but the assumptions of the current theory of adaptive isogeometric methods with hierarchical splines (see [5–7]) are not satisfied.




**(a)** $\mathbf{p} = (2,2)$ **(b)** $\mathbf{p} = (3,3)$

**(c)** $\mathbf{p} = (4,4)$

**Figure 12.** Comparison of the convergence of the $H^1$-norm error versus degrees of freedom (DOFs) for $\mathcal{H}$-admissible (HB) and $\mathcal{T}$-admissible (THB) meshes, with $\mathbf{p} = (2,2), (3,3), (4,4)$ (from **top** to **bottom**).

## 6. Conclusions

We presented a general framework for the design and implementation of refinement algorithms with (truncated) hierarchical B-splines. The properties of the admissible mesh configurations obtained with the iterative application of these algorithms were thoroughly analyzed. Note that the structure of hierarchical meshes with a certain class of admissibility can be naturally connected with a corresponding mesh grading, as confirmed by the numerical examples. The truncation mechanism behind the construction of THB-splines influences the strictly $\mathcal{T}$-admissible property leading to more localized refinement possibilities (and in turn to a reduced number of degrees of freedom) than the $\mathcal{H}$-admissible counterpart, that is, for the same admissibility class $m$. On the other hand, $\mathcal{H}$-admissible meshes guarantee a bounded number of hierarchical B-splines without the need of considering the truncated basis. The numerical examples also confirm the advantages of THB-splines with respect to the sparsity of the discretizations matrices and the condition number of the mass matrix. Concerning the condition number of the stiffness matrix, the situation is more unclear and a deeper study would be required. The comparison between $\mathcal{H}$- and $\mathcal{T}$-admissible refinements was never presented before and opens the path to additional studies on the optimal configuration for the development of hierarchical isogeometric methods.

The refinement algorithms here presented can be properly combined with coarsening algorithms that preserve the admissible nature of the mesh. This is an important aspect for controlling the effect of successive refinement and coarsening of hierarchical meshes in adaptive isogeometric methods (see e.g., [29,30]) and will be the subject matter of a future study.

**Author Contributions:** Conceptualization, C.B., C.G. and R.V.; Investigation, C.B., C.G. and R.V.; Methodology, C.B., C.G. and R.V.; Software, C.B., C.G. and R.V.; Writing—original draft, C.B., C.G. and R.V.; Writing—Review & Editing, C.B., C.G. and R.V.

**Funding:** This work was partially supported by the MIUR "Futuro in Ricerca" programme through the project "DREAMS", grant number RBFR13FBI3. Rafael Vázquez has been partially supported by the ERC Advanced Grant "CHANGE", grant number 694515, 2016-2020.

**Acknowledgments:** The authors are members of the INdAM Research group GNCS. The INdAM support through GNCS and Finanziamenti Premiali SUNRISE is gratefully acknowledged.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Cottrell, J.A.; Hughes, T.J.R.; Bazilevs, Y. *Isogeometric Analysis: Toward Integration of CAD and FEA*; John Wiley & Sons: Hoboken, NJ, USA, 2009.
2. Hughes, T.J.R.; Cottrell, J.A.; Bazilevs, Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comput. Methods Appl. Mech. Eng.* **2005**, *194*, 4135–4195. [CrossRef]
3. Kraft, R. Adaptive and Linearly Independent Multilevel B-Splines. In *Surface Fitting and Multiresolution Methods*; Le Méhauté, A., Rabut, C., Schumaker, L.L., Eds.; Vanderbilt University Press: Nashville, TN, USA, 1997; pp. 209–218.
4. Vuong, A.V.; Giannelli, C.; Jüttler, B.; Simeon, B. A hierarchical approach to adaptive local refinement in isogeometric analysis. *Comput. Methods Appl. Mech. Eng.* **2011**, *200*, 3554–3567. [CrossRef]
5. Buffa, A.; Giannelli, C. Adaptive isogeometric methods with hierarchical splines: Error estimator and convergence. *Math. Models Methods Appl. Sci.* **2016**, *26*, 1–25. [CrossRef]
6. Buffa, A.; Giannelli, C. Adaptive isogeometric methods with hierarchical splines: Optimality and convergence rates. *Math. Models Methods Appl. Sci.* **2017**, *27*, 2781–2802. [CrossRef]
7. Gantner, G.; Haberlik, D.; Praetorius, D. Adaptive IGAFEM with optimal convergence rates: Hierarchical B-splines. *Math. Models Methods Appl. Sci.* **2017**, *27*, 2631–2674. [CrossRef]
8. Giannelli, C.; Jüttler, B.; Speleers, H. THB-Splines: The truncated basis for hierarchical splines. *Comput. Aided Geom. Des.* **2012**, *29*, 485–498. [CrossRef]
9. Bornemann, P.; Cirak, F. A subdivision-based implementation of the hierarchical B-spline finite element method. *Comput. Methods Appl. Mech. Eng.* **2013**, *253*, 584–598. [CrossRef]
10. Scott, M.A.; Thomas, D.C.; Evans, E.J. Isogeometric spline forests. *Comput. Methods Appl. Mech. Eng.* **2014**, *269*, 222–264. [CrossRef]
11. Garau, E.; Vázquez, R. Algorithms for the implementation of adaptive isogeometric methods using hierarchical B-splines. *Appl. Numer. Math.* **2018**, *123*, 58–87. [CrossRef]
12. Kiss, G.; Giannelli, C.; Jüttler, B. Algorithms and data structures for truncated hierarchical B-splines. In *Mathematical Methods for Curves and Surfaces*; Floater, M., Lyche, T., Mazure, M.-L., Moerken, K., Schumaker, L.L., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8177, pp. 304–323.
13. Giannelli, C.; Jüttler, B.; Kleiss, S.K.; Mantzaflaris, A.; Simeon, B.; Špeh, J. THB-splines: An effective mathematical technology for adaptive refinement in geometric design and isogeometric analysis. *Comput. Methods Appl. Mech. Eng.* **2016**, *299*, 337–365. [CrossRef]
14. Hennig, P.; Müller, S.; Kästner, M. Bézier extraction and adaptive refinement of truncated hierarchical NURBS. *Comput. Methods Appl. Mech. Eng.* **2016**, *305*, 316–339. [CrossRef]
15. D'Angella, D.; Kollmannsberger, S.; Rank, E.; Reali, A. Multi-level Bézier extraction for hierarchical local refinement of Isogeometric Analysis. *Comput. Methods Appl. Mech. Eng.* **2018**, *328*, 147–174. [CrossRef]
16. Vázquez, R. A new design for the implementation of isogeometric analysis in Octave and Matlab: GeoPDEs 3.0. *Comput. Math. Appl.* **2016**, *72*, 523–554. [CrossRef]
17. de Boor, C. *A Practical Guide to Splines*, revised ed.; Springer: Berlin/Heidelberg, Germany, 2001.

18. Schumaker, L.L. *Spline Functions: Basic Theory*, 3rd ed.; Cambridge University Press: Cambridge, UK, 2007.
19. Morgenstern, P. Mesh Refinement Strategies for the Adaptive Isogeometric Method. Ph.D. Thesis, Institut für Numerische Simulation, Rheinische Friedrich-Wilhelms-Universität Bonn, Bonn, Germany, 2017.
20. Buffa, A.; Giannelli, C.; Morgenstern, P.; Peterseim, D. Complexity of hierarchical refinement for a class of admissible mesh configurations. *Comput. Aided Geom. Des.* **2016**, *47*, 83–92. [CrossRef]
21. Binev, P.; Dahmen, W.; DeVore, R. Adaptive finite element methods with convergence rates. *Numer. Math.* **2004**, *97*, 219–268. [CrossRef]
22. Stevenson, R. Optimality of a standard adaptive finite element method. *Found. Comput. Math.* **2007**, *7*, 245–269. [CrossRef]
23. Buffa, A.; Garau, E.M. A posteriori error estimators for hierarchical B-spline discretizations. *Math. Models Methods Appl. Sci.* **2016**. [CrossRef]
24. Kumar, M.; Kvamsdal, T.; Johannessen, K.A. Superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis. *Comput. Methods Appl. Mech. Eng.* **2017**, *316*, 1086–1156. [CrossRef]
25. Anitescu, C.; Hossain, M.N.; Rabczuk, T. Recovery-based error estimation and adaptivity using high-order splines over hierarchical T-meshes. *Comput. Methods Appl. Mech. Eng.* **2018**, *328*, 638–662. [CrossRef]
26. Dörfler, W. A convergent algorithm for Poisson's equation. *SIAM J. Numer. Anal.* **1996**, *33*, 1106–1124. [CrossRef]
27. Buchegger, F.; Jüttler, B.; Mantzaflaris, A. Adaptively refined multi-patch B-splines with enhanced smoothness. *Appl. Math. Comput.* **2016**, *272*, 159–172.
28. Hennig, P.; Kästner, M.; Morgenstern, P.; Peterseim, D. Adaptive mesh refinement strategies in isogeometric analysis—A computational comparison. *Comput. Methods Appl. Mech. Eng.* **2017**, *316*, 424–448. [CrossRef]
29. Lorenzo, G.; Scott, M.A.; Tew, K.; Hughes, T.J.R.; Gomez, H. Hierarchically refined and coarsened splines for moving interface problems, with particular application to phase-field models of prostate tumor growth. *Comput. Methods Appl. Mech. Eng.* **2017**, *319*, 515–548. [CrossRef]
30. Hennig, P.; Ambati, M.; De Lorenzis, L.; Kästner, M. Projection and transfer operators in adaptive isogeometric analysis with hierarchical B-splines. *Comput. Methods Appl. Mech. Eng.* **2018**, *334*, 313–336. [CrossRef]