Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 369 (2020) 113230
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Adaptive refinement with locally linearly independent LR B-splines: Theory and applications

Francesco Patrizi[a,b,*], Carla Manni[c], Francesca Pelosi[c], Hendrik Speleers[c]

[a] *Department of Applied Mathematics and Cybernetics, SINTEF, Oslo, Norway*
[b] *Department of Mathematics, University of Oslo, Oslo, Norway*
[c] *Department of Mathematics, University of Rome Tor Vergata, Rome, Italy*

Received 28 December 2019; received in revised form 21 May 2020; accepted 20 June 2020
Available online 2 July 2020

Abstract

In this paper, we describe an adaptive refinement strategy for LR B-splines. The presented strategy ensures, at each iteration, local linear independence of the obtained set of LR B-splines. This property is then exploited in two applications: the construction of efficient quasi-interpolation schemes and the numerical solution of elliptic problems using the isogeometric Galerkin method.

Keywords: Adaptive refinement; LR B-splines; Local linear independence; Quasi-interpolation; Isogeometric analysis

1. Introduction

Since the ’70s, curves and surfaces in engineering are usually expressed by means of computer aided design (CAD) technologies such as B-splines and non-uniform rational B-splines (NURBS). Thanks to properties like non-negativity, local support and partition of unity, they allow for an easy control and modification of the geometries they describe, and this motivates their undisputed success as modeling tools for objects with complex shapes in engineering; see, e.g., [1–3] and references therein. On the other hand, B-splines also provide a very efficient representation of smooth piecewise polynomial spaces, and so are a popular ingredient in the construction of approximation schemes; see, e.g., [4–6] and references therein.

More recently, the advent of isogeometric analysis (IgA) has integrated spline and CAD technologies into finite element analysis (FEA); see, e.g., [7–9]. IgA aims to unify the geometric description of the domain of the differential problem with its numerical resolution, in order to expedite the simulation process and gain in accuracy. In addition to the properties listed above, B-splines and NURBS feature other qualities, appreciated in this context, such as (local) linear independence and high global smoothness.

Tensor structures admit a simple but powerful multivariate extension of univariate splines and B-splines. Unfortunately, they lack adequate local refinement, while the constantly increasing demand for higher precision in

* Corresponding author at: Department of Applied Mathematics and Cybernetics, SINTEF, Oslo, Norway.
E-mail addresses: francesco.patrizi@sintef.no (F. Patrizi), manni@mat.uniroma2.it (C. Manni), pelosi@mat.uniroma2.it (F. Pelosi), speleers@mat.uniroma2.it (H. Speleers).

https://doi.org/10.1016/j.cma.2020.113230

simulations and reverse engineering processes begs for adaptive local refinement strategies, in order to reduce the approximation error at low computational costs. This request for adaptivity triggered the interest in new formulations of B-splines and NURBS, still based on local tensor structures [10–16]. All these new classes of functions are defined on locally refined meshes, in which T-vertices in the interior of the domain are allowed, the so-called T-meshes.

Locally refined B-splines, or in short LR B-splines [15], are one of these new formulations, and their definition is inspired by the knot insertion refinement process of tensor B-splines. These latter are defined on global knot vectors, one per direction. The insertion of a new knot in a knot vector corresponds to a line segment in the mesh crossing the entire domain. This refines all the B-splines whose supports are crossed. Instead, LR B-splines are defined on local knot vectors and the insertion of a new knot is always performed with respect to a particular LR B-spline. As a consequence, the LR B-spline definition is consistent with the tensor B-spline definition when the underlying mesh at the end of the process is a tensor mesh, and the formulation of LR B-splines remains broadly similar to classical tensor B-splines even though they enable local refinements. This makes them one of the most elegant extensions of univariate B-splines on local tensor structures.

LR B-splines possess almost all the properties of classical tensor B-splines. Unfortunately, they are not always linearly independent. Heretofore, it is not yet known what are the precise conditions on the locally refined mesh to ensure a linearly independent set of LR B-splines. Nevertheless, some progress has been made in this direction. In [15] an efficient algorithm to seek and destroy linear dependence relations has been introduced, but it does not handle every possible locally refined mesh. In [17] a first analysis on the necessary conditions for encountering a linear dependence relation has been presented. There, it has also been proved that, for any bidegree, a linear dependence relation in the LR B-spline set involves at least eight functions. In [18,19] a characterization of the local linear independence of LR B-splines has been provided. Such a strong property is guaranteed only on locally refined meshes with certain constraints on the lengths and positions of the line segments that yield particular arrangements of the LR B-spline supports. On the other hand, a practical adaptive refinement strategy to produce meshes with the local linear independence property is still missing in the literature. To the best of our knowledge, the only mesh construction that leads to a locally linearly independent set of LR B-splines can be found in [19]. Such a construction, however, cannot be considered as a practical strategy because the regions to be refined and the maximal resolution, i.e., the sizes of the smallest cells in the domain induced by the mesh, must be chosen a priori.

In this paper, we describe a practical refinement strategy ensuring the local linear independence of the corresponding LR B-splines; the resulting mesh is called N_2S_2 LR-mesh. With examples we illustrate that locally refined meshes can be achieved which are only slightly larger than standard refined (structured) LR-meshes without linear independence warranty; see Fig. 11. The strong theoretical property of local linear independence is appealing and admits, e.g., the construction of efficient local approximation (quasi-interpolation) schemes and the unisolvency of sparse linear systems in isogeometric discretizations of differential problems based on such LR B-splines.

As mentioned above, there are several alternative spline technologies based on local tensor structures that are suited for adaptive refinement. The most popular are T-splines [11,14], hierarchical and truncated hierarchical B-splines (HB-splines and THB-splines) [10,13], and polynomial splines over hierarchical T-meshes (PHT-splines) [12]. Numerical comparisons between LR B-splines and some of these alternatives have been presented in [20]. Our refinement strategy is endowed with interesting theoretical features: it ensures local linear independence of the basis functions and does not require any restriction on degree and smoothness. Analysis-suitable (and dual-compatible) T-splines also enjoy both properties [9, Section 7], but their structure entails a certain amount of non-local refinement propagation. Local linear independence is not available for HB/THB-splines, while PHT-splines assume reduced regularity. In this perspective, our refinement strategy strengthens LR B-splines as an attractive choice among adaptive spline methods based on local tensor structures.

The remainder of the paper is divided into 5 sections. Section 2 contains the definition of LR B-splines and a summary of their main properties. Section 3 describes the mesh refinement strategy and is the core of the paper. Sections 4 and 5 present applications of the refinement strategy in the context of quasi-interpolation and isogeometric Galerkin discretizations of elliptic problems. We end in Section 6 with some concluding remarks.

Throughout the paper, we assume the reader to be familiar with the definition and main properties of (univariate) B-splines, in particular with the knot insertion procedure. An introduction to this topic can be found, e.g., in the review papers [3,6] or in the classical books [4] and [5].

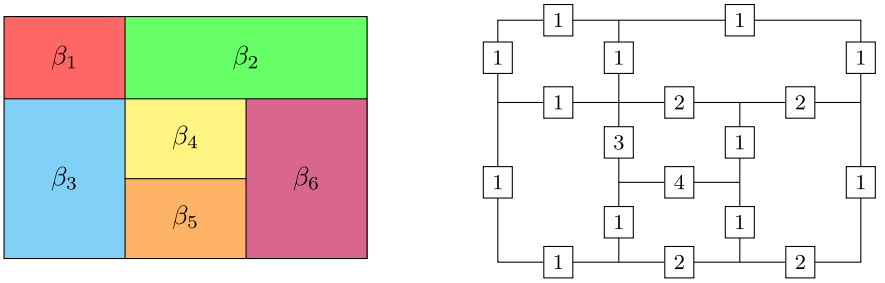


Fig. 1. Example of a box-partition $\mathcal{E}$ of a rectangle Ω in (a), and the mesh corresponding to $\mathcal{E}$ in (b). The meshlines are identified by squares showing the associated multiplicities.

2. Locally refined B-splines

In this section, we introduce locally refined B-splines, or in short LR B-splines, and discuss several of their properties, following the terminology from [17]. We denote by Π_p the space of univariate polynomials of degree less than or equal to p, and by $\Pi_{\boldsymbol{p}}$ the space of bivariate polynomials of degrees less than or equal to $\boldsymbol{p} = (p_1, p_2)$ component-wise. Furthermore, we denote by $B[\boldsymbol{x}, \boldsymbol{y}]$ the bivariate B-spline defined on the (local) knot vectors $\boldsymbol{x} = (x_1, \ldots, x_{p_1+2})$ and $\boldsymbol{y} = (y_1, \ldots, y_{p_2+2})$, where $x_i \leq x_{i+1}$ and $y_i \leq y_{i+1}$ for all i. The bidegree of $B[\boldsymbol{x}, \boldsymbol{y}]$ is $\boldsymbol{p} = (p_1, p_2)$ and is implicitly specified by the length of $\boldsymbol{x}$ and $\boldsymbol{y}$.

In order to define LR B-splines, we first introduce the concept of box-partition.

Definition 2.1. Given an axis-aligned rectangle $\Omega \subseteq \mathbb{R}^2$, a **box-partition** of Ω is a finite collection $\mathcal{E}$ of axis-aligned rectangles in Ω such that

1. $\mathring{\beta}_1 \cap \mathring{\beta}_2 = \emptyset$ for any $\beta_1, \beta_2 \in \mathcal{E}$, with $\beta_1 \neq \beta_2$;
2. $\bigcup_{\beta \in \mathcal{E}} \beta = \Omega$.

Given a box-partition $\mathcal{E}$, we define the **vertices** of $\mathcal{E}$ as the vertices of its elements. A **meshline** is an axis-aligned segment contained in an edge of an element of $\mathcal{E}$, connecting two and only two vertices of $\mathcal{E}$ located at its end-points. The collection of all the meshlines of the box-partition is called **mesh**, and denoted by $\mathcal{M}$. A meshline can be expressed as the Cartesian product of a point in $\mathbb{R}$ and a finite interval. Let $\alpha \in \mathbb{R}$ be the value of such a point and let $k \in \{1, 2\}$ be its position in the Cartesian product. If $k = 1$ the meshline is vertical and if $k = 2$ the meshline is horizontal. We sometimes write k-meshline to specify the direction of the meshline, and (k, α)-meshline to specify exactly on which axis-parallel line in $\mathbb{R}^2$ the meshline lies. A vertex of $\mathcal{E}$ is called **T-vertex** if it is the intersection of two collinear meshlines and another meshline, say γ, orthogonal to them. We call the T-vertex vertical if γ is vertical, and horizontal otherwise.

For defining splines of a certain bidegree $\boldsymbol{p} = (p_1, p_2)$ and smoothness across the meshlines, we also need the notion of **multiplicity** of a meshline. This is a positive integer associated with every meshline in $\mathcal{M}$. For a k-meshline this number is assumed to be maximally $p_k + 1$. A meshline in $\mathcal{M}$ has **full multiplicity** if its multiplicity is maximal, and we say that $\mathcal{M}$ is **open** if every boundary meshline has full multiplicity. If all the meshlines of the box-partition have the same multiplicity m we say that $\mathcal{M}$ has multiplicity m. When the T-vertices of $\mathcal{E}$ occur only on $\partial\Omega$ and all collinear meshlines have the same multiplicity, the corresponding mesh is called **tensor mesh**. Fig. 1 shows an example of a box-partition and its associated mesh.

Given a bivariate B-spline $B[\boldsymbol{x}, \boldsymbol{y}]$, let $x_{i_1}, \ldots, x_{i_r}$ and $y_{j_1}, \ldots, y_{j_s}$ be the distinct knots in $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. The mesh

$$\begin{aligned} \mathcal{M}(\boldsymbol{x}, \boldsymbol{y}) := &\{\{x_{i_\ell}\} \times [y_{j_n}, y_{j_{n+1}}] : \ \ell = 1, \ldots, r;\ n = 1, \ldots, s-1\} \\ &\cup \{[x_{i_n}, x_{i_{n+1}}] \times \{y_{j_\ell}\} : \ell = 1, \ldots, s;\ n = 1, \ldots, r-1\} \end{aligned} \tag{1}$$

is a tensor mesh in $\operatorname{supp} B[\boldsymbol{x}, \boldsymbol{y}]$. The multiplicities of the meshlines in $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ are given by the multiplicities of the knots of $B[\boldsymbol{x}, \boldsymbol{y}]$. For instance, the $(1, x_{i_\ell})$-meshlines $\{x_{i_\ell}\} \times [y_{j_n}, y_{j_{n+1}}]$ for $n = 1, \ldots, s-1$ have all the same multiplicity equal to the multiplicity of x_{i_ℓ} in $\boldsymbol{x}$.

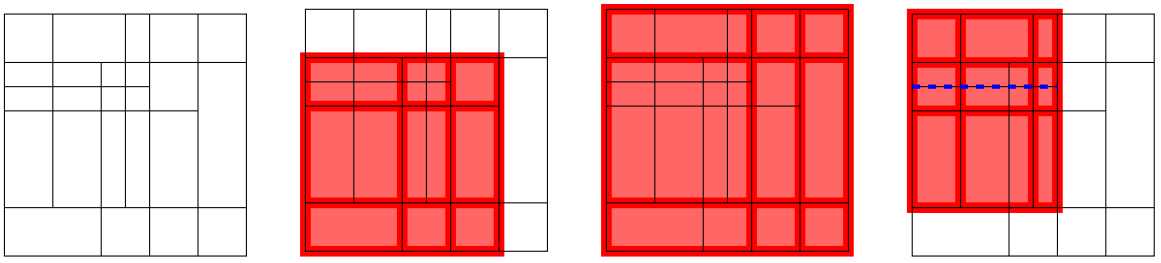


Fig. 2. Some supports of B-splines of bidegree $(2, 2)$ on a mesh $\mathcal{M}$ of multiplicity 1. The mesh is shown in (a). The B-splines whose supports are depicted in (b) and (c) have minimal support on $\mathcal{M}$. The tensor meshes defined by the B-spline's knots are highlighted with thicker lines. On the other hand, the B-spline in (d) does not have minimal support on $\mathcal{M}$: the collection of meshlines contained in the dashed line disconnects its support.

Definition 2.2. Given a mesh $\mathcal{M}$ and a B-spline $B[\boldsymbol{x}, \boldsymbol{y}]$, we say that $B[\boldsymbol{x}, \boldsymbol{y}]$ has **support** on $\mathcal{M}$ if

- the meshlines in $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ can be obtained as unions of meshlines in $\mathcal{M}$, and
- their multiplicities are less than or equal to the multiplicities of the corresponding meshlines in $\mathcal{M}$.

Furthermore, we say that $B[\boldsymbol{x}, \boldsymbol{y}]$ has **minimal support** on $\mathcal{M}$ if

- it has support on $\mathcal{M}$,
- the multiplicities of the interior meshlines in $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ are equal to the multiplicities of the corresponding meshlines in $\mathcal{M}$, and
- there is no collection γ of collinear meshlines in $\mathcal{M}\backslash\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ such that $\operatorname{supp} B[\boldsymbol{x}, \boldsymbol{y}]\backslash\gamma$ is not connected.

Fig. 2 shows examples of supports of B-splines of bidegree $(2, 2)$ on a mesh of multiplicity 1. In particular, the B-splines in (b)–(c) have minimal support, while the support of the B-spline in (d) can be disconnected by the collection of meshlines γ, visualized by dashed lines in the figure.

Given a mesh $\mathcal{M}$ and a B-spline $B[\boldsymbol{x}, \boldsymbol{y}]$ with support in $\mathcal{M}$, assume that it does not have minimal support on $\mathcal{M}$. Then, there exists a collection of (k, α)-meshlines γ such that $\operatorname{supp} B[\boldsymbol{x}, \boldsymbol{y}]\backslash\gamma$ is not connected and either γ is in $\mathcal{M}\backslash\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ or $\gamma \subseteq \mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$, i.e., α is an internal knot of $\boldsymbol{x}$ for $k = 1$ or $\boldsymbol{y}$ for $k = 2$ but its meshlines have lower multiplicities in $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ than in $\mathcal{M}$. Assume that the meshlines in γ have all the same multiplicity m in $\mathcal{M}$. Denoting by $\mu(\alpha) \geq 0$ the number of times α appears in the knot vector of $B[\boldsymbol{x}, \boldsymbol{y}]$ in the kth direction, then $m - \mu(\alpha)$ is strictly positive as $B[\boldsymbol{x}, \boldsymbol{y}]$ has support, but not minimal support, on $\mathcal{M}$. One could consider such α as an extra knot, of multiplicity $m - \mu(\alpha)$, with respect to the knot vector of $B[\boldsymbol{x}, \boldsymbol{y}]$ in the kth direction (in $\boldsymbol{x}$ if $k = 1$ and in $\boldsymbol{y}$ if $k = 2$), and perform knot insertion on $B[\boldsymbol{x}, \boldsymbol{y}]$. If α was already a knot of $B[\boldsymbol{x}, \boldsymbol{y}]$, so $\mu(\alpha) \geq 1$, this means raising its multiplicity by $m - \mu(\alpha)$. The resulting generated B-splines will still have support on $\mathcal{M}$ and eventually they will also have minimal support on $\mathcal{M}$. As an example, the collection γ highlighted with dashed lines in Fig. 2(d) is made of $(2, \alpha)$-meshlines, for some α, of multiplicity 1. Such α can be inserted as new knot of multiplicity 1 in the knot vector in the y-direction of the considered B-spline to refine it in two B-splines via knot insertion.

The LR B-splines are generated by means of the above procedure. We start by considering a coarse tensor mesh and we refine it by inserting collections of collinear meshlines, one at a time, of the same multiplicity. On the initial mesh we consider the standard tensor B-splines and whenever a B-spline in our collection has no longer minimal support during the mesh refinement process, we refine it by using the knot insertion procedure. The LR B-splines will be the final set of B-splines produced by this algorithm. In the following definitions we formalize this by describing the mesh refinement process in our framework.

Definition 2.3. Given a box-partition $\mathcal{E}$ and an axis-aligned segment γ, we say that γ **traverses** $\beta \in \mathcal{E}$ if $\gamma \subseteq \beta$ and the interior of β is divided into two parts by γ, i.e., $\beta\backslash\gamma$ is not connected. A **split** is a finite union of contiguous and collinear axis-aligned segments $\gamma = \cup_i \gamma_i$ such that every γ_i either is a meshline of the box-partition or traverses some $\beta \in \mathcal{E}$. A mesh $\mathcal{M}$ has **constant splits** if each split in it is made of meshlines of the same multiplicity.

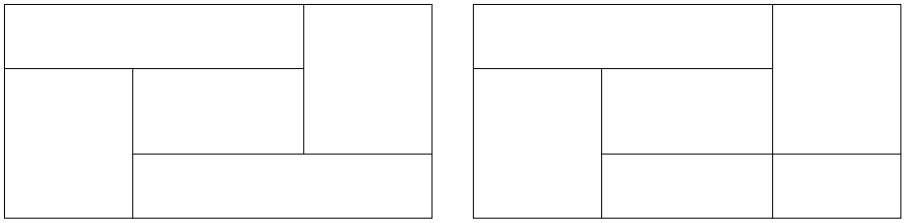


Fig. 3. Two meshes. Assume that the boundary has a multiplicity large enough so that it is possible to define a B-spline of bidegree $\boldsymbol{p}$ on it. Then, the mesh in (a) is not an LR-mesh because it cannot be built through a sequence of split insertions. Indeed, the initial coarse tensor mesh is formed by the boundary, and no other meshline traverses the only element of this initial box partition. The mesh in (b) is an LR-mesh similar to the one in (a).

Like for meshlines, we sometimes write k-split with $k \in \{1, 2\}$ to specify the direction of the split or (k, α)-split to specify on what axis-parallel line in $\mathbb{R}^2$ the split lies.

When a split γ is inserted in a box-partition $\mathcal{E}$, any traversed $\beta \in \mathcal{E}$ is replaced with the two subrectangles β_1, β_2 given by the closures of the connected components of $\beta \backslash \gamma$. The resulting new box-partition will be denoted by $\mathcal{E} + \gamma$ and its corresponding mesh by $\mathcal{M} + \gamma$. We also assume that a positive integer μ_γ has been assigned to any split γ. The multiplicities of the meshlines in $\mathcal{M} \cap (\mathcal{M} + \gamma)$ and not contained in γ are unchanged. Contrarily, the multiplicities of the meshlines in γ that were already in $\mathcal{M}$ are raised by μ_γ, and the new meshlines in γ have multiplicity equal to μ_γ on $\mathcal{M} + \gamma$.

The LR B-splines are defined on a class of meshes with constant splits, called LR-meshes. Thus, from now on, we restrict our attention to meshes that have constant splits. In particular, we note that when refining a mesh $\mathcal{M}$ by inserting a split γ, either γ is made solely of new meshlines or it is made solely of meshlines already on $\mathcal{M}$, in order for $\mathcal{M} + \gamma$ to have constant splits.

Definition 2.4. Given a mesh $\mathcal{M}$ with constant splits, a B-spline $B[\boldsymbol{x}, \boldsymbol{y}]$ with support on $\mathcal{M}$ and a split γ, we say that γ **traverses** $B[\boldsymbol{x}, \boldsymbol{y}]$ if the interior of $\operatorname{supp} B[\boldsymbol{x}, \boldsymbol{y}]$ is divided into two parts by γ, i.e., $\operatorname{supp} B[\boldsymbol{x}, \boldsymbol{y}] \backslash \gamma$ is not connected and either γ is in $\mathcal{M} \backslash \mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ or $\gamma \subseteq \mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ but the multiplicity of its meshlines is lower in $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ than in $\mathcal{M}$.

We are now ready to define the mesh refinement process and the LR B-splines. The meshes generated by this procedure will be called LR-meshes.

Definition 2.5. Given a bidegree $\boldsymbol{p} = (p_1, p_2)$, let $\mathcal{M}_1$ be a tensor mesh such that the set of standard tensor B-splines of bidegree $\boldsymbol{p}$ on $\mathcal{M}_1$ is non-empty, and denote it by $\mathcal{B}_1$. We then define a sequence of meshes $\mathcal{M}_2, \mathcal{M}_3, \ldots$ and corresponding function sets $\mathcal{B}_2, \mathcal{B}_3, \ldots$ as follows. For $i = 1, 2, \ldots$, let γ_i be a split such that $\mathcal{M}_{i+1} := \mathcal{M}_i + \gamma_i$ has constant splits and such that the support of at least one B-spline in $\mathcal{B}_i$ is traversed by a split in $\mathcal{M}_{i+1}$. On this refined mesh $\mathcal{M}_{i+1}$, the new set of B-splines $\mathcal{B}_{i+1}$ is constructed by the following procedure.

1. Initialize the set by $\mathcal{B}_{i+1} \leftarrow \mathcal{B}_i$.
2. As long as there exists $B[\boldsymbol{x}^j, \boldsymbol{y}^j] \in \mathcal{B}_{i+1}$ with no minimal support on $\mathcal{M}_{i+1}$:
   (a) Apply knot insertion: $\exists B[\boldsymbol{x}_1^j, \boldsymbol{y}_1^j], B[\boldsymbol{x}_2^j, \boldsymbol{y}_2^j] : B[\boldsymbol{x}^j, \boldsymbol{y}^j] = \alpha_1 B[\boldsymbol{x}_1^j, \boldsymbol{y}_1^j] + \alpha_2 B[\boldsymbol{x}_2^j, \boldsymbol{y}_2^j]$.
   (b) Update the set: $\mathcal{B}_{i+1} \leftarrow (\mathcal{B}_{i+1} \backslash \{B[\boldsymbol{x}^j, \boldsymbol{y}^j]\}) \cup \{B[\boldsymbol{x}_1^j, \boldsymbol{y}_1^j], B[\boldsymbol{x}_2^j, \boldsymbol{y}_2^j]\}$.

The mesh generated at each step is called **LR-mesh** and the corresponding function set is called **LR B-spline set**.

Not every mesh is an LR-mesh. For instance, one could consider meshes that do not have constant splits or meshes that cannot be built through a sequence of split insertions, as illustrated by the mesh depicted in Fig. 3(a). The LR-mesh in Fig. 3(b) is generated by four split insertions and the order of insertion is fixed in this case. In general, however, the mesh refinement process producing a given LR-mesh $\mathcal{M} = \mathcal{M}_N$ is not unique because the split insertion ordering can often be changed. Nevertheless, the LR B-spline set on $\mathcal{M}$ is well defined because it is independent of such insertion ordering, as proved in [15, Theorem 3.4].

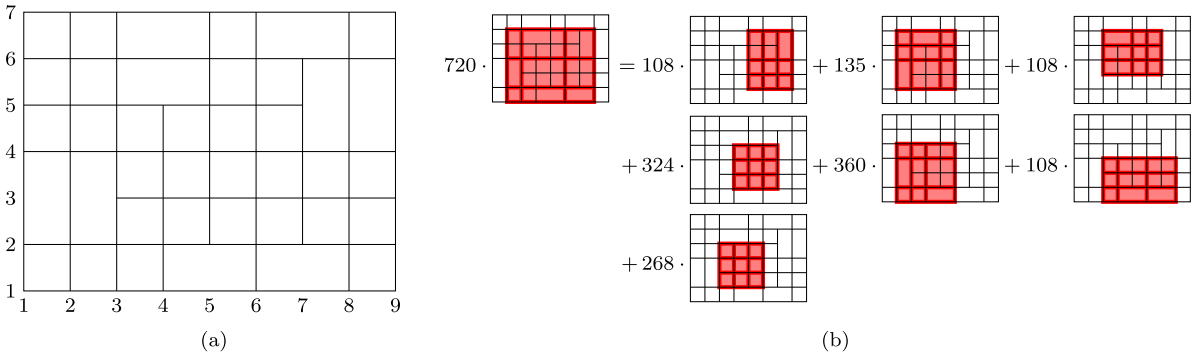


Fig. 4. Example of linear dependence in the LR B-spline set. The parameterization of an LR-mesh $\mathcal{M}$ of multiplicity 1 is considered in (a), and the linear dependence relation among some of the LR B-splines of bidegree $(2, 2)$ defined on $\mathcal{M}$ is illustrated in (b). The LR B-splines are represented by means of their supports on the mesh and the tensor meshes induced by their knots are highlighted with thicker meshlines.

Given an LR-mesh, the corresponding LR B-splines have several desirable properties for applications. By their definition, it is clear that

- they are non-negative,
- they have minimal support, and
- they can be expressed by the LR B-splines on finer LR-meshes using non-negative coefficients (provided by the knot insertion procedure).

Furthermore, it is possible to scale them by means of positive weights so that they also form a partition of unity; see [15, Section 7].

Unfortunately, they are not always linearly independent. Fig. 4 shows an example of linear dependence among the LR B-splines of bidegree $(2, 2)$ defined on an LR-mesh of multiplicity 1. Heretofore, it is not yet known what are the precise conditions on the LR-mesh to ensure a linearly independent set of LR B-splines.

In [19] a characterization of the local linear independence of LR B-splines has been provided. Such a strong property is guaranteed only on LR-meshes with certain constraints on the split lengths and positions that yield particular arrangements of the LR B-spline supports. This last statement is formalized in the following.

Definition 2.6. Given a mesh $\mathcal{M}$, let $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ and $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ be two different LR B-splines defined on $\mathcal{M}$. We say that $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ is **nested** in $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, and we write $B[\boldsymbol{x}^2, \boldsymbol{y}^2] \preceq B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, if

- $\operatorname{supp} B[\boldsymbol{x}^2, \boldsymbol{y}^2] \subseteq \operatorname{supp} B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, and
- any meshline γ of $\mathcal{M}$ in $\partial\operatorname{supp} B[\boldsymbol{x}^1, \boldsymbol{y}^1] \cap \partial\operatorname{supp} B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ has a higher (or equal) multiplicity when considered in $\mathcal{M}(\boldsymbol{x}^1, \boldsymbol{y}^1)$ than in $\mathcal{M}(\boldsymbol{x}^2, \boldsymbol{y}^2)$.

An open mesh where no LR B-spline is nested is said to have the **non-nested support property**, or in short the **N_2S-property**.

The definition of nested LR B-splines was formulated for the first time in [18]. Definition 2.6 is different but equivalent to it (see Appendix). Fig. 5 shows an example of an LR B-spline nested into another. The following result from [19, Theorem 4] relates the local linear independence of LR B-splines to the N_2S-property of the mesh.

Theorem 2.7. *Given a bidegree $\boldsymbol{p} = (p_1, p_2)$, let $\mathcal{M}$ be an open LR-mesh corresponding to a box-partition $\mathcal{E}$ and let $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ be the set of LR B-splines of bidegree $\boldsymbol{p}$ on $\mathcal{M}$. The following statements are equivalent.*

1. *The elements of $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ are locally linearly independent.*
2. *$\mathcal{M}$ has the N_2S-property.*
3. *For any element $\beta \in \mathcal{E}$, the number of non-zero LR B-splines over β satisfies*

$$\#\{B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M}) : \operatorname{supp} B \supseteq \beta\} = (p_1 + 1)(p_2 + 1).$$

4. *The LR B-splines form a partition of unity, without the use of scaling weights.*

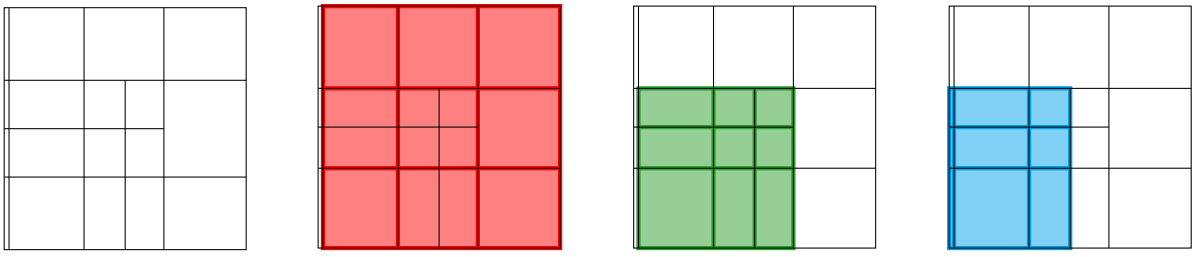


Fig. 5. Example of nested LR B-splines on the mesh $\mathcal{M}$ shown in (a). All the meshlines have multiplicity 1 except those in the left edge of $\mathcal{M}$, highlighted with a double line, which have multiplicity 2. In (b)–(d) three LR B-splines ($B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$, $B[\boldsymbol{x}^3, \boldsymbol{y}^3]$, respectively) of bidegree $(2, 2)$ with minimal support on $\mathcal{M}$ are represented by means of their supports and the tensor meshes induced by their knots. All the knots of these LR B-splines have multiplicity 1 except x_1^3 which has multiplicity 2. Therefore, $B[\boldsymbol{x}^2, \boldsymbol{y}^2] \preceq B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ but $B[\boldsymbol{x}^3, \boldsymbol{y}^3] \not\preceq B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ and $B[\boldsymbol{x}^3, \boldsymbol{y}^3] \not\preceq B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, despite that $\operatorname{supp} B[\boldsymbol{x}^3, \boldsymbol{y}^3] \subseteq \operatorname{supp} B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ and $\operatorname{supp} B[\boldsymbol{x}^3, \boldsymbol{y}^3] \subseteq \operatorname{supp} B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, because the shared meshlines in the left edge of $\operatorname{supp} B[\boldsymbol{x}^3, \boldsymbol{y}^3]$, $\operatorname{supp} B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ and $\operatorname{supp} B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ have multiplicity 2 in $\mathcal{M}(\boldsymbol{x}^3, \boldsymbol{y}^3)$ and multiplicity 1 in $\mathcal{M}(\boldsymbol{x}^2, \boldsymbol{y}^2)$ and $\mathcal{M}(\boldsymbol{x}^1, \boldsymbol{y}^1)$.

An element of $\mathcal{E}$ for which item 3 of Theorem 2.7 holds is said to be **non-overloaded**. Note that $(p_1+1)(p_2+1)$ is the dimension of the polynomial space over the element.

In [19] one can also find an algorithm to construct LR-meshes so that the N_2S-property is fulfilled. This approach, however, has a relevant drawback for practical purposes: the regions to be refined and the maximal resolution have to be chosen a priori. Moreover, the algorithm cannot be stopped prematurely, before having inserted all the splits determined initially. In practice, one rarely knows in advance where the error will be large and how fine the mesh has to be chosen to reduce it under a certain tolerance.

In the next section, we present an alternative way to generate LR-meshes so that the N_2S-property is guaranteed.

3. N_2S-structured mesh refinement strategy

In this section, we define a local refinement strategy that ensures the N_2S-property for the obtained meshes. It consists of two steps. First, we apply the so-called structured mesh refinement, defined in [21], to the LR B-splines whose contribution to the approximation error is larger than a given tolerance. Then, we slightly modify the obtained mesh by prolonging some splits, to recover the N_2S-property. The meshes generated by this refinement are open meshes with internal meshlines of multiplicity one.

As opposed to the classical finite element method, in which the refinement is applied to the box-partition elements, the **structured mesh refinement** is a refinement applied to the function space, i.e., we select for refinement the LR B-splines contributing more to the approximation error rather than the box-partition elements where a larger error occurs. This approach is justified by the fact that on an LR-mesh, any new split inserted must traverse at least the support of one LR B-spline. If we choose to select the elements where the error is larger, then the refinement has to be extended anyway to traverse the support of at least one LR B-spline containing the elements, resulting in a refinement of the LR B-spline basis. Moreover, since several LR B-splines contain such elements, those chosen for the refinement extension could be not those contributing more to the error, resulting in a suboptimal refinement, or we could refine more LR B-splines than necessary, wasting degrees of freedom.

Once the LR B-splines are selected, we refine them by halving the interval steps in their knot vectors. This corresponds to the insertion of a net of meshlines in the B-spline supports on the mesh. We therefore perform the LR B-spline generation algorithm described in Definition 2.5. Every selected LR B-spline is fragmented into LR B-splines of smaller support and replaced with them. The LR-mesh obtained in this way will be called a **structured LR-mesh**.

In summary, the structured mesh refinement consists of two steps:

1. LR B-splines are selected to be refined and not box-partition elements;
2. the interval steps of their knot vectors are halved to obtain the new LR-mesh.

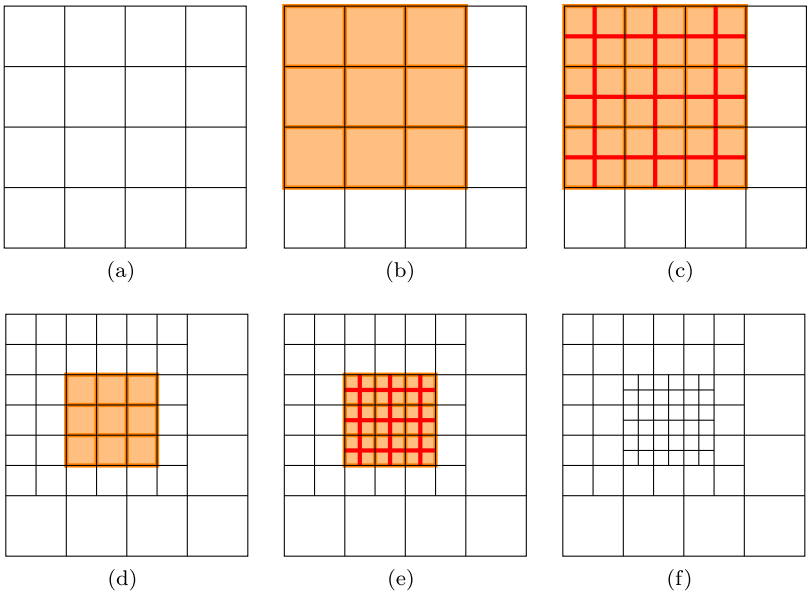


Fig. 6. Two iterations of the structured mesh refinement of bidegree $(2, 2)$. We consider the initial open tensor mesh with internal meshlines of multiplicity 1 in (a). Figure (b) shows the support of an LR B-spline selected for refinement. We refine it by halving the interval steps in its knot vectors. This results in the insertion of a net of meshlines in the LR-mesh as shown in (c). In (d) we select another LR B-spline in the new set of LR B-splines and we refine it as illustrated in (e). Figure (f) depicts the final mesh obtained.

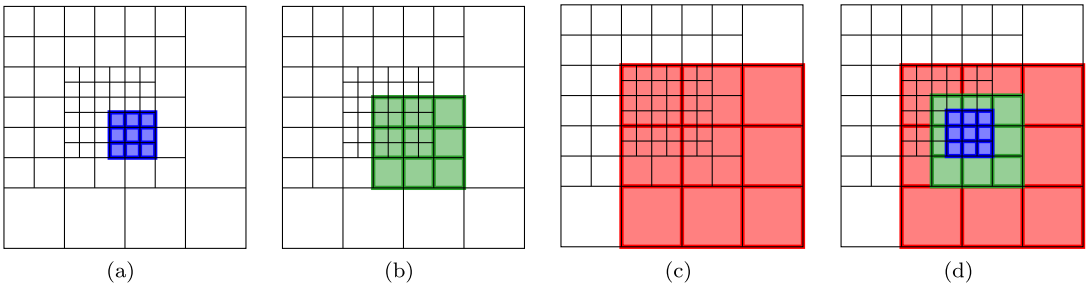


Fig. 7. An LR B-spline nested in another LR B-spline, which in turn is nested in another LR B-spline on a structured LR-mesh for bidegree $(2, 2)$. Consider again the mesh in Fig. 6(f). In (a)–(c) we depict the supports of three LR B-splines on this mesh. The support in (a) is contained in the interior of the support in (b) and (c), and the support in (b) is contained in the interior of the support in (c), as shown in (d). Therefore, the LR B-spline considered in (a) is nested both in the LR B-splines in (b) and (c), and the LR B-spline in (b) is nested in the LR B-spline in (c). Hence, the considered mesh does not have the N_2S-property.

Fig. 6 shows two iterations of such refinement. In general, the structured mesh refinement does not generate LR-meshes with the N_2S-property. The LR-mesh in Fig. 6(f) is an example as explained in Fig. 7. Furthermore, the structured mesh refinement may produce linearly dependent sets of LR B-splines. Fig. 8 shows an example for bidegree $(4, 4)$.

On the other hand, the standard B-splines defined on a plain tensor mesh are locally linearly independent, and the meshes generated by the structured mesh refinement are locally tensor meshes far from the boundary of the region where the structured mesh refinement is applied. The LR B-splines defined in these zones of the mesh behave like the standard B-splines, and therefore are locally linearly independent. On the boundary of the region where the refinement has been applied, LR B-splines of smaller support can be nested in LR B-splines of larger support. Hence, in such case the resulting LR-mesh does not have the N_2S-property.

The idea for our refinement strategy, which will be called **N_2S-structured mesh refinement**, is therefore to recover the N_2S-property in the mesh by slightly modifying it in these transition regions. When an LR B-spline $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ is nested into another LR B-spline $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, one could prolong the splits in $\mathcal{M}(\boldsymbol{x}^2, \boldsymbol{y}^2)$ in some direction

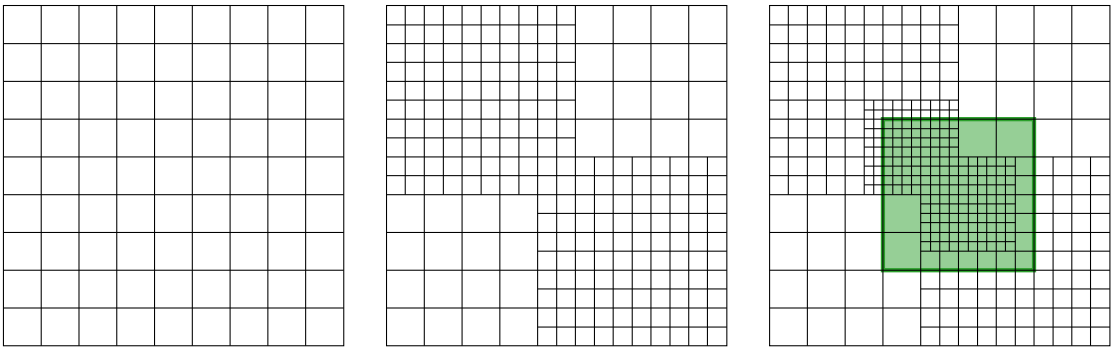


Fig. 8. A structured LR-mesh with a linear dependence relation among the LR B-splines of bidegree $(4, 4)$ defined on the highlighted region in (c). We start by considering an open tensor mesh with interior meshlines of multiplicity 1 as in (a). Then, we apply two iterations of structured mesh refinement as shown in (b)–(c). The LR B-splines with support in the region highlighted in (c) are linearly dependent. In particular, the region corresponds to the support of an LR B-spline that has many nested LR B-splines in it. One can prove the existence of the linear dependence relation by computing the spline space dimension and the number of LR B-splines defined on the mesh as explained in the examples of [17]. This configuration can be reproduced for any bidegree (p_1, p_2) with $p_k \geq 4$ for $k = 1, 2$.

to traverse entirely supp $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$. This, by Definition 2.5, would refine $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ in LR B-splines that turn out not to have nested LR B-splines in their supports anymore. This last statement is formalized in Corollary 3.3. To this end, we first prove the N_2S-property for LR-meshes with a particular structure.

Definition 3.1. An LR-mesh $\mathcal{M}$ on the domain Ω is said to be **tensorized in the kth direction**, for $k \in \{1, 2\}$, if all the internal k-meshlines in $\mathcal{M}$ are contained in k-splits crossing Ω entirely, i.e., there are no vertical, if $k = 1$, or horizontal, if $k = 2$, T-vertices in the interior of Ω.

Proposition 3.2. *Let $\mathcal{M}$ be an LR-mesh tensorized in the kth direction for some $k \in \{1, 2\}$. Then, the LR B-splines defined on $\mathcal{M}$ are all non-nested.*

Proof. Without loss of generality, we can assume that $\mathcal{M}$ is tensorized in the first direction, i.e., the vertical meshlines are all contained in vertical splits crossing the domain entirely. This means that in $\mathcal{M}$ no vertical meshline ends in the interior of the domain and therefore in the interior of the support of any LR B-splines defined on $\mathcal{M}$. We now proceed by contradiction and assume that there exists an LR B-spline in $\mathcal{M}$, say $B^2 = B[\boldsymbol{x}^2, \boldsymbol{y}^2]$, nested in another, say $B^1 = B[\boldsymbol{x}^1, \boldsymbol{y}^1]$; see Definition 2.6. Because of the tensorization in the first direction, this can only happen if they share the same knot vector in the x-direction, $\boldsymbol{x}^1 = \boldsymbol{x}^2$. In particular, their supports have the same extreme values in the x-direction. This implies that all the horizontal splits, counting the multiplicities, of $\mathcal{M}$ traversing supp B^2 must traverse supp B^1 as well. Since B^2 is nested in B^1 and B^1 has minimal support (it is an LR B-spline), it follows that $\boldsymbol{y}^1 = \boldsymbol{y}^2$, and as a consequence we have $B^1 = B^2$. This is a contradiction and concludes the proof. □

Corollary 3.3. *Given an LR-mesh $\mathcal{M}$, let $B = B[\boldsymbol{x}, \boldsymbol{y}]$ and $B^1 = B[\boldsymbol{x}^1, \boldsymbol{y}^1], \ldots, B^n = B[\boldsymbol{x}^n, \boldsymbol{y}^n]$ be LR B-splines defined on $\mathcal{M}$ such that $B^1, \ldots, B^n \preceq B$. Let $\mathcal{N}$ be the mesh defined by the restriction of $\mathcal{M}$ to the meshlines of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y}), \mathcal{M}(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, \mathcal{M}(\boldsymbol{x}^n, \boldsymbol{y}^n)$. Then,*

1. *there are at least one horizontal T-vertex and one vertical T-vertex of $\mathcal{N}$ in the interior of supp B;*
2. *by extending all the splits of $\mathcal{N}$ in some direction to cross supp B entirely, B is refined, by Definition 2.5, in LR B-splines that do not have any nested LR B-splines anymore.*

Proof. We prove the two statements separately as follows.

1. Assume that there are no vertical T-vertices of $\mathcal{N}$ in the interior of supp B. Then, $\mathcal{N}$ would be tensorized in the first direction. By Proposition 3.2, it would imply that all the LR B-splines defined on $\mathcal{N}$ are non-nested,

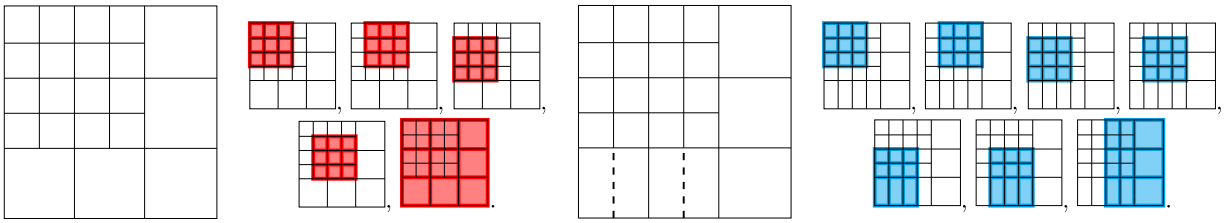


Fig. 9. Example of a vertical tensor expansion. We consider five LR B-splines of bidegree $(2, 2)$, namely B and $B^1, \dots, B^4$, with $\operatorname{supp} B^1, \dots, \operatorname{supp} B^4$ contained in the upper left corner of $\operatorname{supp} B$. The mesh $\mathcal{N}$, of multiplicity 1, generated by the meshlines of $B, B^1, \dots, B^4$ is depicted in (a), and the supports of the LR B-splines are shown in (b). In (c) we perform a vertical tensor expansion of $B^1, \dots, B^4$ in B, and in (d) the supports of the new set of LR B-splines are shown: none of them has a nested LR B-spline anymore. Here local linear independence is achieved by replacing the LR B-spline B with the three LR B-splines corresponding to the bottom row of (d).

which is a contradiction. Analogously, one can prove that at least one horizontal T-vertex of $\mathcal{N}$ must be in the interior of $\operatorname{supp} B$.

2. Since B contains the support of other B-splines, $\mathcal{N} \neq \mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ and in particular there exist at least one horizontal T-vertex and one vertical T-vertex by the previous item. We now focus on the vertical T-vertices, but of course the same argument can also be carried out for the horizontal T-vertices. We extend all the vertical splits in $\mathcal{N}$ to cross $\operatorname{supp} B$ entirely, and denote this new mesh as $\widetilde{\mathcal{N}}$. By Definition 2.5, the extensions trigger a refinement of B via knot insertions. $\widetilde{\mathcal{N}}$ is tensorized in the first direction and, by Proposition 3.2, no LR B-spline defined on $\widetilde{\mathcal{N}}$ is nested into another. □

The extension of the splits considered in item 2 of Corollary 3.3 will be called a **one-directional tensor expansion** of $B^1, \dots, B^n$ in B. An example of such an expansion is shown in Fig. 9.

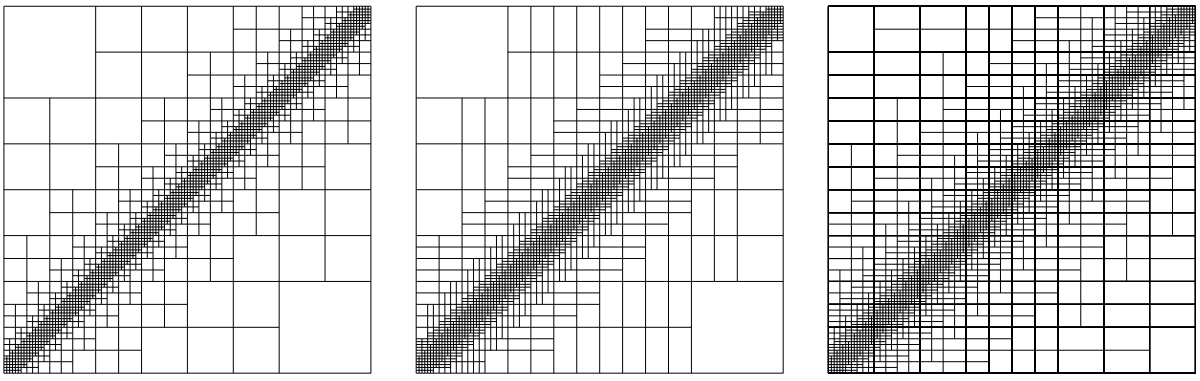
The N_2S-structured mesh refinement is defined algorithmically as follows. We start from a structured mesh refinement to obtain a new set of LR B-splines. We then collect in a set $\mathcal{B}$ all those LR B-splines that have nested LR B-splines in their supports. If $\mathcal{B}$ is non-empty, we select an LR B-spline B in $\mathcal{B}$ and we apply a one-directional tensor expansion to it. This triggers a refinement of the LR B-spline set, and therefore it changes also the set $\mathcal{B}$. We repeat this procedure till $\mathcal{B}$ becomes empty. In Theorem 3.5 we shall prove that this always happens in a finite number of steps. The complete algorithm is described by Algorithm 3.4.

Algorithm 3.4 (*N_2S-structured mesh refinement*)**.**

1. $\mathcal{B}_1$ is the B-spline set on the open tensor mesh equal to the domain's boundary;
2. **for** $i = 1, 2, \dots$ **do**
3. perform a structured mesh refinement of $\mathcal{B}_i$;
4. initialize $\mathcal{B}_{i+1}$ as the LR B-spline set defined on the new LR-mesh;
5. define $\mathcal{B} = \{B \in \mathcal{B}_{i+1} : \exists\, B' \in \mathcal{B}_{i+1} \text{ with } B' \preceq B\}$;
6. **while** $\mathcal{B} \neq \emptyset$ **do**
7. select $B \in \mathcal{B}$;
8. perform a one-directional tensor expansion of the LR B-splines nested in B;
9. update $\mathcal{B}_{i+1}$ as the LR B-spline set defined on the new LR-mesh;
10. update $\mathcal{B} = \{B \in \mathcal{B}_{i+1} : \exists\, B' \in \mathcal{B}_{i+1} \text{ with } B' \preceq B\}$;

The one-directional tensor expansions (line 8 of Algorithm 3.4) are performed by alternating the direction for i even and odd, respectively, in order to bound the thinning of the box-partition elements in a specific direction and preserve the uniformity of the mesh as much as possible. The LR-mesh obtained in this way will be called an **N_2S-structured LR-mesh**, or in short **N_2S_2 LR-mesh**.

Theorem 3.5. *Given an axis-aligned rectangular domain $\Omega \subseteq \mathbb{R}^2$, let $\mathcal{B}_1$ be the set of standard bivariate B-splines defined on the open tensor mesh whose meshlines are the edges of $\partial\Omega$. Then,*



(a) 1430 LR B-splines (b) 1894 LR B-splines (c) 2243 LR B-splines

Fig. 10. Visual comparison between meshes of bidegree (2, 2) obtained by performing seven iterations of different mesh refinement strategies along the diagonal: (a) the structured LR-mesh, (b) the N_2S_2 LR-mesh, and (c) the LR-mesh proposed in [19].

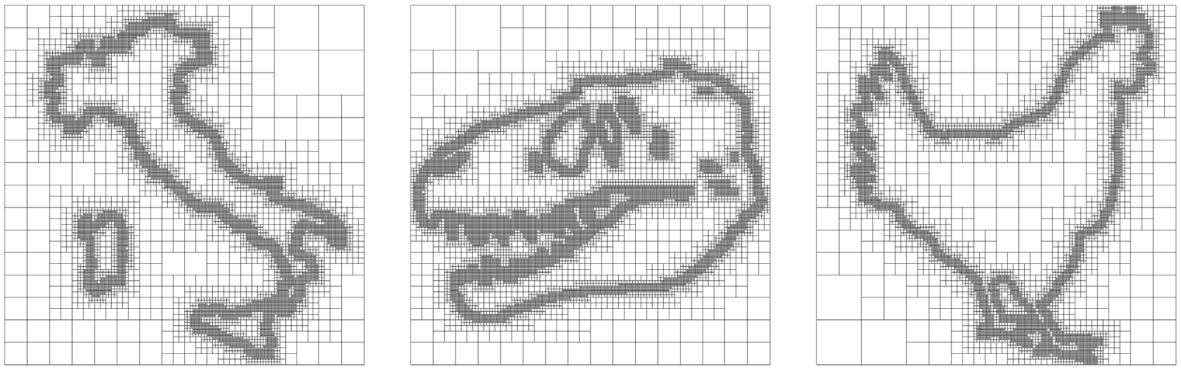
1. *the LR B-spline sets $\mathcal{B}_i$ provided by Algorithm 3.4 are well defined, i.e., the set $\mathcal{B}$ of the algorithm becomes empty in a finite number of iterations, for every index $i \geq 2$;*
2. *all the LR B-splines in $\mathcal{B}_i$ are non-nested, for every $i \geq 1$.*

Proof. Without loss of generality, we can assume that $\Omega = [0, 1] \times [0, 1]$. We proceed by induction on the index of the B-spline set. For $i = 1$, $\mathcal{B}_1$ is the set of standard B-splines on the open tensor mesh equal to the domain's boundary and we know they are locally linearly independent. By Theorem 2.7, this is equivalent to be all non-nested. Assume now that $\mathcal{B}_i$ for some $i \geq 1$ is well defined and that the functions in it are all non-nested. Let us then prove that also $\mathcal{B}_{i+1}$ is well defined and there is no LR B-spline nested into another LR B-spline of it. At every loop iteration in the algorithm, the LR B-splines that have a nested LR B-spline in their support are collected in the set $\mathcal{B}$. Therefore, whenever we can show that $\mathcal{B}$ becomes empty after a certain iteration of the loop, we can immediately conclude both statements in the theorem.

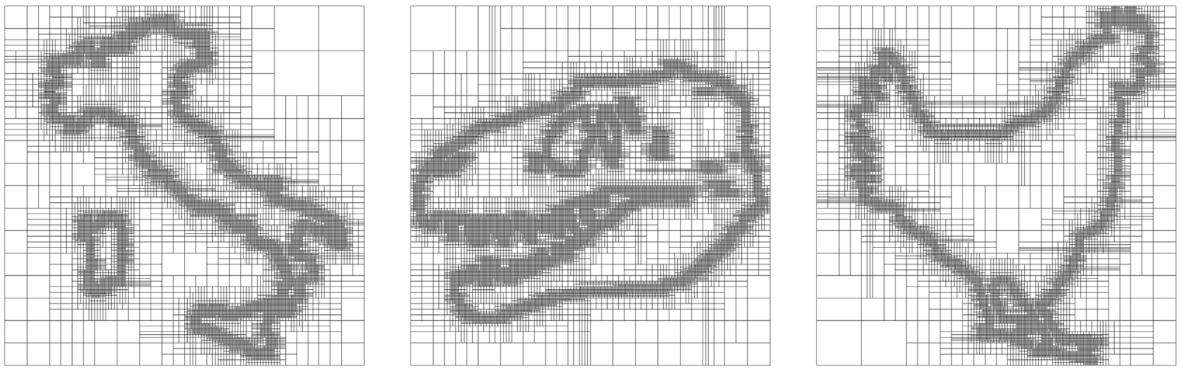
By Corollary 3.3, all the one-directional tensor expansions performed to define the set $\mathcal{B}_{i+1}$ can be done in the same direction $k \in \{1, 2\}$, which is therefore fixed once and for all by the index $i+1$. The length of the LR B-spline supports in the $(3-k)$th direction at any iteration of the loop cannot become shorter than $2^{-(i+1)}$ regardless of the number of one-directional tensor expansions applied until then. This is because the $(3-k)$-splits on the LR-meshes defined in the loop are fixed by the structured mesh refinement performed on $\mathcal{B}_i$ at the beginning of the process and the minimal length of the box-partition elements in the $(3-k)$th direction is $2^{-(i+1)}$. Therefore, the split extensions applied when performing a one-directional tensor expansion in the kth direction have lengths bounded from below by $2^{-(i+1)}$ in all the steps of the loop. This means that in a finite number of one-directional tensor expansions a k-split could be extended up to the domain's boundary, if needed, to remove nestedness issues, as these extensions cannot become arbitrarily small. In the worst case scenario, we must extend all the k-splits to cross entirely the domain. However, in this case, the resulting LR-mesh would be tensorized in the kth direction. By Proposition 3.2, there are only non-nested LR B-splines on this LR-mesh and thus $\mathcal{B}$ becomes empty in a finite number of loop iterations. □

In practice, the loop related to $\mathcal{B}$ stops quickly and the N_2S_2 LR-meshes are far from being entirely tensorized in one direction. In Fig. 10 we depict (a) the structured LR-mesh, (b) the corresponding N_2S_2 LR-mesh, and (c) the LR-mesh proposed in [19], obtained by performing seven iterations of diagonal refinement in $[0, 1]^2$, using bidegree (2, 2). For ease of comparison, we also indicate the number of LR B-splines defined on each of these meshes. We recall that the LR B-splines are not locally linearly independent on the structured LR-mesh, while they are on the N_2S_2 LR-mesh and the LR-mesh proposed in [19].

In Figs. 10(b) and 11 one can see how the refinement in the N_2S_2 LR-meshes propagates from the region where the structured mesh refinement has been applied. In all the considered cases, the refinement does not heavily spread out. It is important to highlight, however, that the prolongation of the splits needed to recover the N_2S-property



(a) 10281 LR B-splines (b) 13459 LR B-splines (c) 8608 LR B-splines



(d) 12438 LR B-splines (e) 15993 LR B-splines (f) 10841 LR B-splines

Fig. 11. Meshes of bidegree $(2, 2)$ obtained by performing eight iterations of mesh refinement on three different regions resulting in a structured LR-mesh (top row) and an N_2S_2 LR-mesh (bottom row).

is not unique. Indeed, when refining an LR B-spline to remove nestedness issues, the inserted split prolongations refine not only the considered LR B-spline but in general also other LR B-splines in the neighborhood. Then, some of the newly introduced neighboring LR B-splines might not need a one-directional tensor expansion anymore and the ordering used for removing nestedness has thus an effect on the resulting mesh.

One might consider to treat all the LR B-splines "in parallel", i.e., first collect all the split extensions needed to remove nestedness in all the LR B-splines requiring a treatment and then insert all of them at the same time to refine the function basis. This could result in a more uniform propagation of the refinement out of the region where the structured mesh has been applied. On the other hand, by doing this, some split extensions could be unnecessary for recovering the N_2S-property. Therefore, in general, also the number of LR B-splines on these meshes would be higher than the number obtained when treating one LR B-spline at a time. In the examples presented in this paper we do not remove nestedness "in parallel". Hence, the resulting N_2S_2 LR-meshes depend on the order used when the one-directional tensor expansions are applied. On the other hand, the number of LR B-splines will be closer to the number of LR B-splines obtained when performing only the structured mesh refinement, i.e., closer to the "optimal" number of LR B-splines needed to reduce the error while preserving the local linear independence.

We finally remark that one could also opt for full tensor expansions in the supports, instead of one-directional tensor expansions, to solve nestedness issues. The proof of Theorem 3.5 can be rephrased for the case of full tensor expansions. The key is that we only prolong splits provided by the structured mesh refinement performed at the beginning of the process. Therefore, if we do full tensor expansions, in the worst case scenario we would end up with a standard tensor mesh of size $h = 2^{-(i+1)}$ to define the set $\mathcal{B}^{i+1}$, instead of an LR-mesh tensorized in one direction. In such case, $\mathcal{B}$ would still become empty in a finite number of loop iterations. However, we decided to do the expansion of the splits only in one direction at a time because it results in fewer propagation. Our choice of

alternating the direction in each refinement step was motivated by the wish to preserve the uniformity of the mesh as much as possible. However, this choice could be improved and optimized towards the specific problem to be addressed (for instance, to achieve a certain amount of anisotropy).

4. Application I: Quasi-interpolation

A quasi-interpolation method is a procedure to compute the coefficients assigned to the basis elements of a prescribed function space, with the aim of approximating a given arbitrary function or data set. The resulting approximant is called a **quasi-interpolant (QI)**. The computation of any of such coefficients may depend only on the data/function restricted to the corresponding basis element's support (local method), and perhaps some neighboring other basis elements' supports, or it may depend on the data/function in the entire domain (global method), as in the least-squares method. Given a function f and an approximation space, whose basis is denoted by $\mathcal{B}$, we write a related QI in the form

$$\mathfrak{Q}f := \sum_{B\in\mathcal{B}} \lambda_B(f)B, \tag{2}$$

where $\lambda_B(f)$ is the coefficient of the basis element $B \in \mathcal{B}$ computed by the selected method.

Definition 4.1. A quasi-interpolation method such that $\mathfrak{Q}f = f$ for all f in a space V is said to **reproduce** the space V.

When using spline spaces of bidegree $\boldsymbol{p}$ as approximation spaces, a common requirement is that the polynomial space $\Pi_{\boldsymbol{p}}$ is reproduced by the quasi-interpolation method, in order to ensure good approximation properties. A general recipe for constructing local quasi-interpolation methods for tensor spline spaces, with the polynomial reproduction property, can be found in [22]. We summarize it for the convenience of the reader.

Recipe 4.2. *Let f be a given function defined on the rectangle Ω. Given a bidegree $\boldsymbol{p}$, let $\widetilde{\mathcal{M}}$ be an open tensor mesh on Ω, and let $\mathcal{B}(\widetilde{\mathcal{M}})$ be the set of tensor B-splines of bidegree $\boldsymbol{p}$ on $\widetilde{\mathcal{M}}$. For every $B = B[\boldsymbol{x}, \boldsymbol{y}] \in \mathcal{B}(\widetilde{\mathcal{M}})$, we compute the coefficient $\lambda_B(f)$ as follows.*

1. *Let $U \subseteq \mathbb{R}^2$ be an open set that intersects the interior of supp B (for instance, U can be a box-partition element of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$), and let $\mathcal{B}(U)$ be the subset of $\mathcal{B}(\widetilde{\mathcal{M}})$ consisting of all the tensor B-splines not identically zero on U.*
2. *Choose a local polynomial approximation method $\mathfrak{P}_U$ such that $\mathfrak{P}_U g = g$ for all $g \in \Pi_{\boldsymbol{p}}$ defined on U (typical choices are least-squares or interpolation methods). Letting $g_{|U}$ be the restriction of g to U, we can write*

$$(\mathfrak{P}_U f)_{|U} = \sum_{\widetilde{B}\in\mathcal{B}(U)} b_{\widetilde{B}}(f)\widetilde{B}_{|U},$$

   for some coefficients $b_{\widetilde{B}}(f)$ provided by the chosen local approximation method.
3. *Since $B \in \mathcal{B}(U)$, set $\lambda_B(f) := b_B(f)$.*

Then, according to (2), the quasi-interpolant to f in the space spanned by $\mathcal{B}(\widetilde{\mathcal{M}})$ is defined as

$$\mathfrak{Q}f := \sum_{B\in\mathcal{B}(\widetilde{\mathcal{M}})} \lambda_B(f)B.$$

Note that the choice of the local polynomial approximation method is an open ingredient in the above recipe for tensor splines (step 2). Actually, it is even allowed to take different polynomial approximation schemes for different B-splines $B \in \mathcal{B}(\widetilde{\mathcal{M}})$. In the same spirit and similar to the local quasi-interpolation strategy developed for THB-splines in [23,24], we can formulate a general recipe for constructing QIs in the space spanned by $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ on a given open LR-mesh $\mathcal{M}$ as follows: for every LR B-spline B in $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$, select any local tensor space containing B, and pick the coefficient corresponding to B in the expression of any QI in such a tensor space. We have here the flexibility of choosing both the local space and the local QI for determining the coefficient of an LR B-spline. In particular, when the smallest local tensor space is considered, we arrive at the following recipe.

Recipe 4.3. *Let f be a given function defined on the rectangle Ω. Given a bidegree $\boldsymbol{p}$, let $\mathcal{M}$ be an open LR-mesh on Ω, and let $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ be the set of LR B-splines of bidegree $\boldsymbol{p}$ on $\mathcal{M}$. For every $B = B[\boldsymbol{x}, \boldsymbol{y}] \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M})$, we compute the coefficient $\lambda_B(f)$ as follows.*

1. *Let $\widetilde{\mathcal{M}}_B$ be the open tensor mesh obtained by raising the boundary meshline multiplicities of $\mathcal{M}_B = \mathcal{M}(\boldsymbol{x}, \boldsymbol{y})$ to full multiplicity, and let $\mathcal{B}(\widetilde{\mathcal{M}}_B)$ be the set of tensor B-splines defined on $\widetilde{\mathcal{M}}_B$.*
2. *Consider a quasi-interpolation method in the space spanned by $\mathcal{B}(\widetilde{\mathcal{M}}_B)$,*

$$\mathfrak{Q}_B f = \sum_{\widetilde{B} \in \mathcal{B}(\widetilde{\mathcal{M}}_B)} b_{\widetilde{B}}(f) \widetilde{B},$$

   reproducing all $g \in \Pi_{\boldsymbol{p}}$ (for instance, use Recipe 4.2).
3. *Since $B \in \mathcal{B}(\widetilde{\mathcal{M}}_B)$, set $\lambda_B(f) := b_B(f)$.*

Then, according to (2), the quasi-interpolant to f in the space spanned by $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ is defined as

$$\mathfrak{Q} f := \sum_{B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M})} \lambda_B(f) B.$$

Since $B \in \mathcal{B}(\widetilde{\mathcal{M}}_B)$ for any $B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M})$, the function $\mathfrak{Q} f$ in Recipe 4.3 is well defined. Moreover, it will reproduce polynomials on the entire domain if the LR-mesh has the N_2S-property as stated in the following proposition.

Proposition 4.4. *Given a bidegree $\boldsymbol{p}$, let $\mathcal{M}$ be an open LR-mesh and let $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ be the set of LR B-splines of bidegree $\boldsymbol{p}$ on $\mathcal{M}$. Assume that $\mathcal{M}$ has the N_2S-property, then*

$$\mathfrak{Q} g = g, \quad \forall g \in \Pi_{\boldsymbol{p}},$$

where the quasi-interpolation operator $\mathfrak{Q}$ is defined in Recipe 4.3.

Proof. From [18, Theorem 4.6], if $\mathcal{M}$ has the N_2S-property, then for all $g \in \Pi_{\boldsymbol{p}}$ we have

$$g = \sum_{B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M})} g_B B, \quad g_B \in \mathbb{R},$$

where for all $B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M})$, the coefficient g_B only depends on g and on the knots defining the LR B-spline B. Therefore, g_B remains the same if we represent g in any set of tensor B-splines containing B. Since, according to Recipe 4.3, each $\mathfrak{Q}_B$ reproduces all polynomials in $\Pi_{\boldsymbol{p}}$ we have

$$g_B = \lambda_B(g), \quad \forall B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M}), \quad g \in \Pi_{\boldsymbol{p}},$$

which completes the proof. □

We have tested the quasi-interpolation strategy described in Recipe 4.3 on N_2S_2 LR-meshes to approximate polynomials and transcendental functions. Given an N_2S_2 LR-mesh $\mathcal{M}$, this recipe requires the construction of a QI based on $\mathcal{B}(\widetilde{\mathcal{M}}_B)$ for each of the LR B-splines $B \in \mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ of bidegree $\boldsymbol{p} = (p_1, p_2)$. Following Recipe 4.2, we have used interpolation as local approximation method in the computation of these QIs. More precisely, we have selected a unisolvent set of $(p_1+1)(p_2+1)$ interpolation points, organized in a tensor grid over a single box-partition element of $\widetilde{\mathcal{M}}_B$, and then we have set a linear system by evaluating f and the tensor B-splines in $\mathcal{B}(\widetilde{\mathcal{M}}_B)$ at these points. This guarantees polynomial reproduction (actually spline reproduction) of the quasi-interpolation method in the spaces spanned by $\mathcal{B}(\widetilde{\mathcal{M}}_B)$, for B varying in $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$. Therefore, also the resulting quasi-interpolation method in the space spanned by $\mathcal{B}^{\mathcal{LR}}(\mathcal{M})$ has the polynomial reproduction property thanks to Proposition 4.4. Indeed, in all the tests with polynomial functions of bidegree at most $\boldsymbol{p}$, the maximal error was in the order of the machine precision, regardless of the number of iterations performed to construct the N_2S_2 LR-mesh. The maximal error was computed on a uniform 150×150 grid.

As test with a transcendental function, we have considered

$$f(x, y) = \frac{2}{3} e^{-\sqrt{(10x-3)^2+(10y-3)^2}} + \frac{2}{3} e^{-\sqrt{(10x+3)^2+(10y+3)^2}} + \frac{2}{3} e^{-\sqrt{(10x)^2+(10y)^2}}, \tag{3}$$

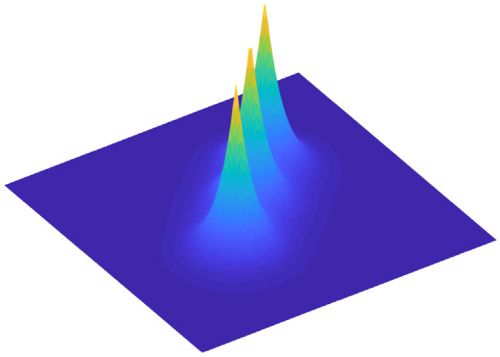


Fig. 12. The transcendental function (3) defined on $[-1, 1]^2$.

Table 1
Dimensions of LR B-spline spaces of bidegree $(2, 2)$ used to approximate the transcendental function (3) on tensor meshes and N_2S_2 LR-meshes for different levels of maximal resolution, and the corresponding optimal maximal errors.

	Dimension of LR B-spline space						
Tensor mesh	36	100	324	1156	4356	16 900	66 564
N_2S_2 LR-mesh	36	86	161	254	363	450	537
Max error	5.686e-01	4.645e-01	2.575e-01	1.472e-01	5.955e-02	2.156e-02	1.415e-02
Level	1	2	3	4	5	6	7

which is characterized by three steep peaks on the square $[-1, 1]^2$ located at $(-0.3, -0.3)$, $(0, 0)$ and $(0.3, 0.3)$; see Fig. 12. This function has also been used in [23] to investigate the approximation power of a similar quasi-interpolation method developed for THB-splines. In Table 1 we compare the dimensions of LR B-spline spaces of bidegree $(2, 2)$ on global tensor meshes and local N_2S_2 LR-meshes for different levels of maximal resolution. For level ℓ, the smallest box-partition elements on the mesh have length $2^{-\ell}$. Each N_2S_2 LR-mesh is obtained by refining the LR B-splines whose supports contain one of the three points where a peak occurs via structured mesh refinement and then by recovering the N_2S-property via one-directional tensor expansions; see Fig. 13. For a given maximal resolution level, the optimal maximal error, i.e., the maximal error when using the global tensor mesh, is preserved by the N_2S_2 LR-mesh. However, the corresponding spline dimensions (degrees of freedom) are significantly different and the discrepancy exponentially grows with the maximal resolution level.

5. Application II: Isogeometric analysis

Isogeometric analysis, or in short IgA [7], is a numerical technique to perform computational simulations on complex geometries. The numerical solutions are represented by means of the same functions as used for the domain modeling. Nowadays, complex geometries are expressed in terms of CAD technologies such as B-splines, NURBS, and their generalizations to address adaptive refinements.

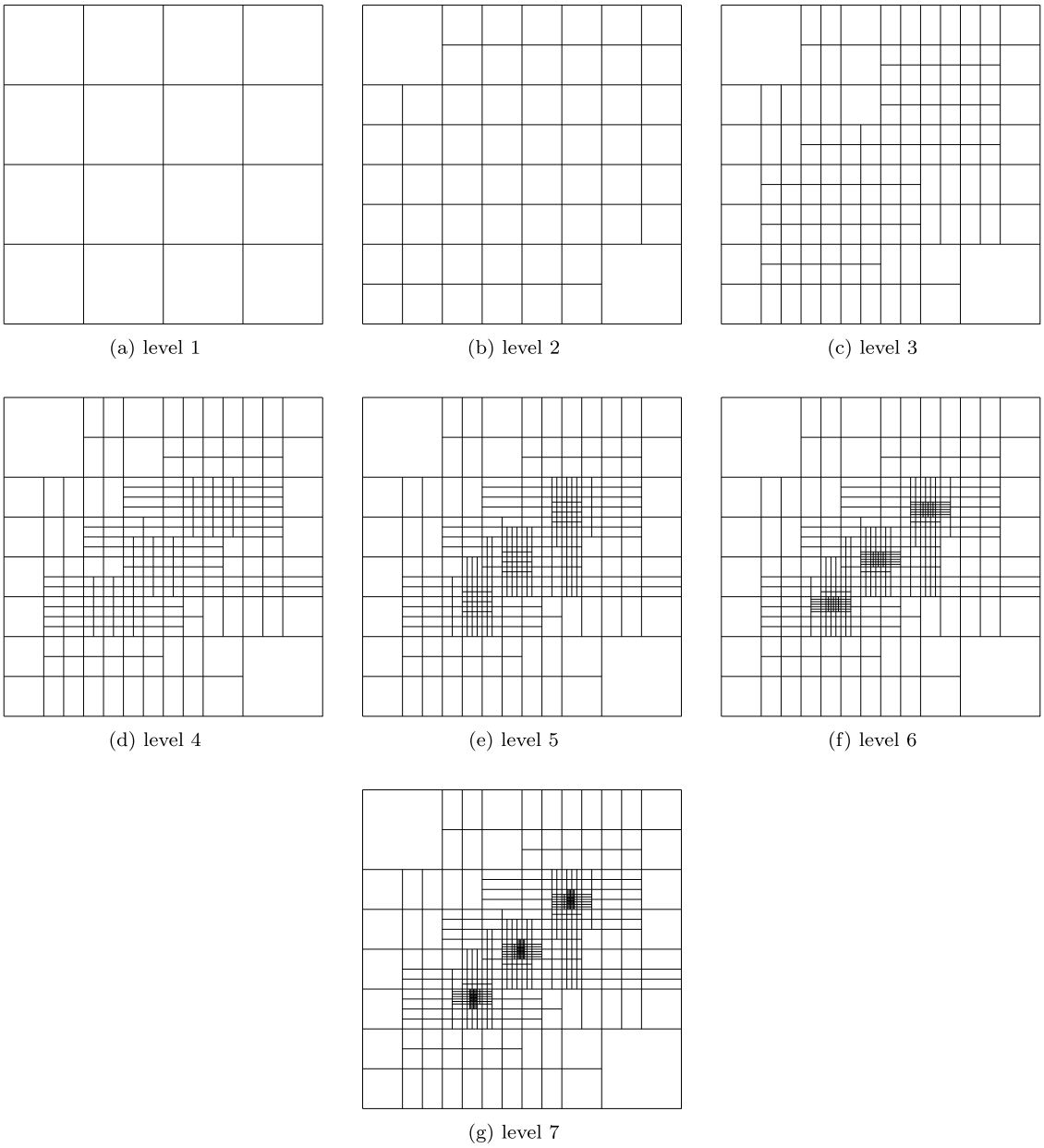
In this section, we adopt the IgA approach, using our refinement strategy for LR B-splines, to approximate the solution of the Poisson problem on $\Omega = [0, 1]^2$,

$$\begin{cases} -\Delta u = f, & \text{in } \mathring{\Omega}, \\ u = u_D, & \text{on } \partial\Omega, \end{cases} \tag{4}$$

whose exact solution is

$$u(x, y) = \arctan\left(100\left(\sqrt{(x - 1.25)^2 + (y + 0.25)^2} - \frac{\pi}{3}\right)\right);$$

see Fig. 14. This example is a good benchmark for numerical schemes, as the sharp interior layer of the exact solution highlights the approximation quality, and it has been used extensively in the literature; see, e.g., [21,25,26].



(a) level 1 (b) level 2 (c) level 3

(d) level 4 (e) level 5 (f) level 6

(g) level 7

Fig. 13. N_2S_2 LR-meshes produced to approximate the transcendental function (3) for different levels of maximal resolution.

In the context of Galerkin discretizations, two properties are desirable:

- (local) linear independence of the space generators,
- refinement adaptivity.

The linear independence of the functions used as building blocks of the numerical solution avoids the numerical complexity posed by the singularity of the matrices associated to the problem discretization. The refinement adaptivity is desired for balancing accuracy and computational cost as it allows for a higher precision, only in the regions where it is needed to reproduce large variations of the exact solution. LR B-splines on N_2S_2 LR-meshes are suitable candidates as both the (local) linear independence of the space generators and the adaptivity of the refinement are guaranteed.

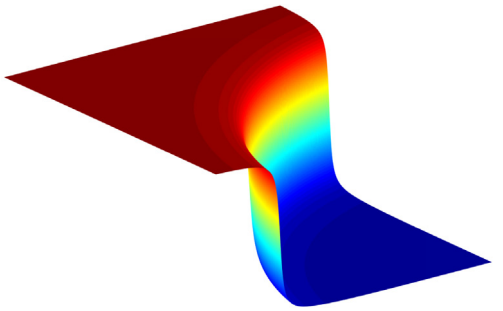


Fig. 14. Exact solution of the Poisson problem (4) defined on $[0, 1]^2$.

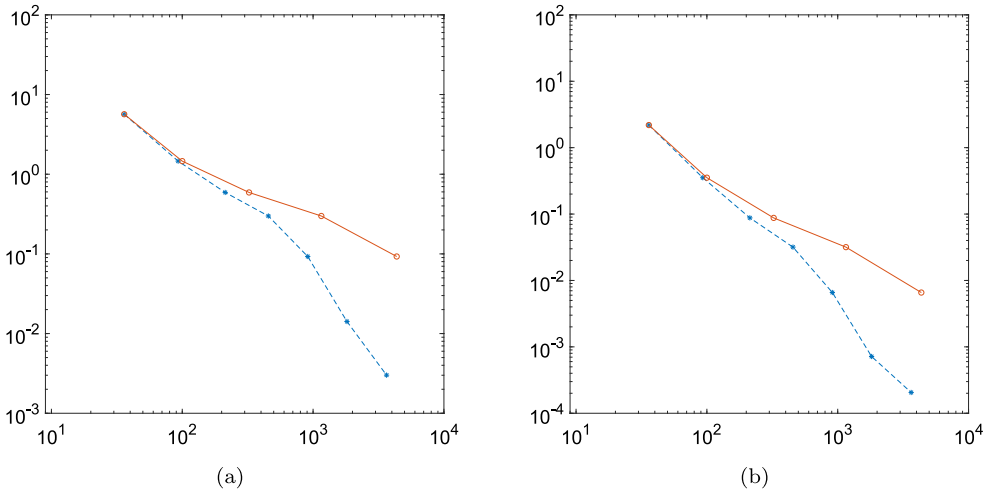
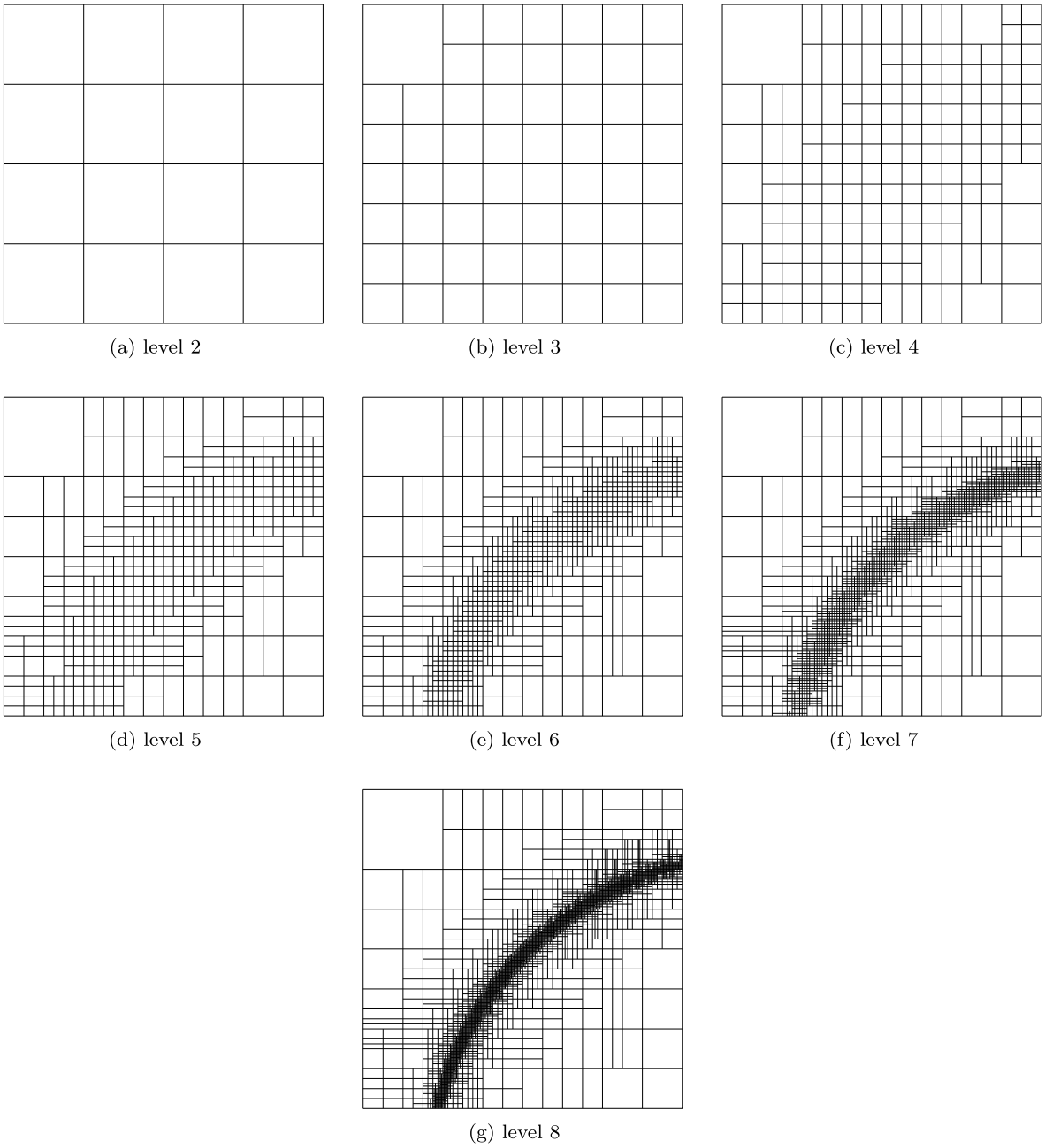


Fig. 15. Error versus spline dimension (degrees of freedom) when approximating the solution of the Poisson problem (4) with LR B-splines of bidegree $(2, 2)$ on tensor meshes (solid line) and N_2S_2 LR-meshes (dashed line) for different levels of maximal resolution. The L^∞-norm of the error is depicted in (a) and the L^2-norm of the error in (b).

In Fig. 15 we compare the L^∞-norm and the L^2-norm of the error (Fig. 15(a) and (b), respectively), using LR B-spline spaces of bidegree $(2, 2)$ on global tensor meshes and local N_2S_2 LR-meshes for different levels of maximal resolution to approximate the solution of the Poisson problem (4). Again, for level ℓ, the smallest box-partition elements on the mesh have length $2^{-\ell}$. Each N_2S_2 LR-mesh is computed by first applying the structured mesh to the LR B-splines whose supports intersect the curve where the sharp interior layer in the exact solution occurs, and then by performing one-directional tensor expansions to recover the N_2S-property; see Fig. 16. The error norms, which are computed discretely on a uniform grid of 1000×1000 points, are plotted in log–log scale with respect to the spline dimensions (degrees of freedom). The solid line with circular markers shows the decay when using global tensor meshes, while the dashed line with star markers shows the decay for the N_2S_2 LR-meshes. In the figures, the first marker corresponds to the 4×4 tensor mesh, for maximal resolution level $\ell = 2$, and it is the maximal level for which the LR B-spline and standard tensor B-spline sets coincide. When considering a comparable number of degrees of freedom, the N_2S_2 LR-mesh leads to a significant reduction of both the L^∞-norm and the L^2-norm of the error with respect to the tensor mesh, thanks to the adaptivity of the refinement. In Table 2 we show the 2-norm condition numbers and dimensions of the corresponding stiffness matrices. Note that the dimension of the stiffness matrix does not take into account the degrees of freedom related to the boundary condition.

6. Conclusion

LR B-splines are one of the most elegant extensions of univariate B-splines on local tensor structures that allow for local refinement. They possess almost all the properties of classical B-splines, but they are not always linearly



(a) level 2 (b) level 3 (c) level 4

(d) level 5 (e) level 6 (f) level 7

(g) level 8

Fig. 16. N_2S_2 LR-meshes produced to approximate the solution of the Poisson problem (4) for different levels of maximal resolution.

independent. Recently, a characterization of LR-meshes ensuring local linear independence of the corresponding LR B-splines has been presented in the literature [18,19]. However, a practical adaptive refinement strategy for LR-meshes that maintain such a property was missing. In this paper, we have filled this gap by describing an adaptive refinement strategy that generates LR-meshes where the corresponding LR B-splines are locally linearly independent. Subsequently, we have exploited the local linear independence of the LR B-splines to construct efficient quasi-interpolation schemes and to solve elliptic problems using the isogeometric Galerkin method.

Table 2
Condition numbers and dimensions of the stiffness matrices assembled to approximate the solution of the Poisson problem (4) with LR B-splines of bidegree $(2, 2)$ on tensor meshes and N_2S_2 LR-meshes for different levels of maximal resolution.

	Condition number and dimension of stiffness matrix						
Tensor mesh	4.00e00 16	5.22e00 64	1.98e01 256	7.81e01 1024	3.12e02 4096	1.24e03 16 384	4.98e03 65 536
N_2S_2 LR-mesh	4.00e00 16	5.79e00 59	3.02e01 163	9.44e01 390	2.73e02 835	9.86e02 1728	2.76e03 3515
Level	2	3	4	5	6	7	8

Besides the structural properties of LR B-splines, the refinement strategy we have proposed relies on the results in [18] which extend to any higher dimensional setting. It seems thus reasonable that our bivariate construction can be generalized to the multivariate case with similar definitions and proofs; see also [19].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We are very grateful to Tor Dokken (SINTEF, Oslo) and Kjetil André Johannessen (SINTEF, Trondheim) for sharing with us the example in Fig. 8 that shows that the standard structured mesh refinement may produce linearly dependent sets of LR B-splines. This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme (grant agreement No 675789) and from the MIUR Excellence Department Project awarded to the Department of Mathematics, University of Rome Tor Vergata (CUP E83C18000100006). The authors are members of Gruppo Nazionale per il Calcolo Scientifico, Istituto Nazionale di Alta Matematica.

Appendix. Nested LR B-splines

The purpose of this appendix is to show the equivalence of the definition of nestedness used in this paper (Definition 2.6) and the original definition provided in [18, Definition 2.4] for LR B-splines. The latter definition is formulated in terms of repeated knot insertion and, in view of [18, Proposition 2.5], it is equivalent to the following definition.

Definition A.1. Let $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ and $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ be two different tensor B-splines. Let $\mu_{\boldsymbol{x}^k}(z)$ and $\mu_{\boldsymbol{y}^k}(z)$, for $k = 1, 2$, denote the number of times $z \in \mathbb{R}$ occurs in the vectors $\boldsymbol{x}^k$ and $\boldsymbol{y}^k$, respectively. Then, we say that $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ is **nested** in $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, and we write $B[\boldsymbol{x}^2, \boldsymbol{y}^2] \preceq B[\boldsymbol{x}^1, \boldsymbol{y}^1]$, if

1. $\begin{cases} \mu_{\boldsymbol{x}^2}(z) \geq \mu_{\boldsymbol{x}^1}(z), & \forall\, z \in]x_1^2, x_{p_1+2}^2[, \\ \mu_{\boldsymbol{y}^2}(z) \geq \mu_{\boldsymbol{y}^1}(z), & \forall\, z \in]y_1^2, y_{p_2+2}^2[, \end{cases}$
2. $\begin{cases} \mu_{\boldsymbol{x}^2}(z) \leq \mu_{\boldsymbol{x}^1}(z), & \forall\, z \notin]x_1^1, x_{p_1+2}^1[, \\ \mu_{\boldsymbol{y}^2}(z) \leq \mu_{\boldsymbol{y}^1}(z), & \forall\, z \notin]y_1^1, y_{p_2+2}^1[. \end{cases}$

We now prove the equivalence of the two definitions for LR B-splines.

Proposition A.2. *Given a mesh $\mathcal{M}$, let $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ and $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ be two different LR B-splines defined on $\mathcal{M}$. For $B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ and $B[\boldsymbol{x}^2, \boldsymbol{y}^2]$, Definition 2.6 is equivalent to Definition A.1.*

Proof. Let $B^1 := B[\boldsymbol{x}^1, \boldsymbol{y}^1]$ and $B^2 := B[\boldsymbol{x}^2, \boldsymbol{y}^2]$ be two LR B-splines defined on the mesh $\mathcal{M}$. Assuming that they satisfy the conditions in Definition A.1, we prove that they also satisfy the conditions in Definition 2.6. Let us first show that $\operatorname{supp} B^2 \subseteq \operatorname{supp} B^1$. This means that $[x_1^2, x_{p_1+2}^2] \subseteq [x_1^1, x_{p_1+2}^1]$ and $[y_1^2, y_{p_2+2}^2] \subseteq [y_1^1, y_{p_2+2}^1]$. Suppose $x_1^2 < x_1^1$. Then, $\mu_{\boldsymbol{x}^1}(x_1^2) = 0$ and $\mu_{\boldsymbol{x}^2}(x_1^2) > 0$, but this contradicts item 2 in Definition A.1, and hence $x_1^2 \geq x_1^1$. The

other inequalities can be proved in a similar way to have the interval inclusions. Let now $\gamma \subseteq \partial\text{supp}\, B^1 \cap \partial\text{supp}\, B^2$. Assume without loss of generality that γ is a 1-meshline. Then, γ is a $(1, z)$-meshline for some $z \in \{x^1_1, x^1_{p_1+2}\}$. For any choice of such z, we have $\mu_{\boldsymbol{x}^2}(z) \leq \mu_{\boldsymbol{x}^1}(z)$, by item 2 of Definition A.1, and, by the definition of multiplicity of γ in $\mathcal{M}(\boldsymbol{x}^1, \boldsymbol{y}^1)$ and $\mathcal{M}(\boldsymbol{x}^2, \boldsymbol{y}^2)$, this implies that $\mu(\gamma)$ is higher (or equal) in $\mathcal{M}(\boldsymbol{x}^1, \boldsymbol{y}^1)$ than in $\mathcal{M}(\boldsymbol{x}^2, \boldsymbol{y}^2)$. Therefore, both the conditions in Definition 2.6 are satisfied if those in Definition A.1 are satisfied.

Let us now show the converse. Assuming that the conditions in Definition 2.6 are fulfilled, we prove that the conditions in Definition A.1 are satisfied as well. Let $z \in]x^2_1, x^2_{p_1+2}[$. Since $\text{supp}\, B^2 \subseteq \text{supp}\, B^1$, we have $]x^2_1, x^2_{p_1+2}[\subseteq]x^1_1, x^1_{p_1+2}[$. If $z \notin \boldsymbol{x}^1$, then $\mu_{\boldsymbol{x}^1}(z) = 0$ and therefore $\mu_{\boldsymbol{x}^2}(z) \geq \mu_{\boldsymbol{x}^1}(z)$ for any value of $\mu_{\boldsymbol{x}^2}(z)$. If $z \in \boldsymbol{x}^1$, then it must be also in $\boldsymbol{x}^2$, otherwise the $(1, z)$-split $\{z\} \times [y^1_1, y^1_{p_2+2}]$ would traverse B^2, which would not have minimal support. For the same reason, it must also hold that $\mu_{\boldsymbol{x}^2}(z) = \mu_{\boldsymbol{x}^1}(z)$. This proves item 1 of Definition A.1. Assume now $z \notin [x^1_1, x^1_{p_1+2}]$. Since $\text{supp}\, B^2 \subseteq \text{supp}\, B^1$, we have $x^1_1 \leq x^2_1$ and $x^2_{p_1+2} \leq x^1_{p_1+2}$. Therefore, $\mu_{\boldsymbol{x}^1}(z) = \mu_{\boldsymbol{x}^2}(z) = 0$. If $z \in \{x^1_1, x^1_{p_1+2}\}$ but $z \notin \boldsymbol{x}^2$, then trivially $\mu_{\boldsymbol{x}^2}(z) \leq \mu_{\boldsymbol{x}^1}(z)$. If $z \in \boldsymbol{x}^2$, then z corresponds to $(1, z)$-meshlines in $\partial\text{supp}\, B^1 \cap \partial\text{supp}\, B^2$. By assumption, these meshlines have a higher (or equal) multiplicity in $\mathcal{M}(\boldsymbol{x}^1, \boldsymbol{y}^1)$ than in $\mathcal{M}(\boldsymbol{x}^2, \boldsymbol{y}^2)$, which means $\mu_{\boldsymbol{x}^2}(z) \leq \mu_{\boldsymbol{x}^1}(z)$. The same line of arguments also applies to the knots in the second direction. This proves item 2 of Definition A.1. □

References

[1] L. Piegl, W. Tiller, The NURBS Book, second ed., Springer-Verlag, Berlin, 1997.
[2] E. Cohen, R.F. Riesenfeld, G. Elber, Geometric Modeling with Splines: An Introduction, A K Peters, Wellesley, 2001.
[3] C. Manni, H. Speleers, Standard and non-standard CAGD tools for isogeometric analysis: A tutorial, in: A. Buffa, G. Sangalli (Eds.), IsoGeometric Analysis: A New Paradigm in the Numerical Approximation of PDEs, in: Lecture Notes in Mathematics, vol. 2161, Springer International Publishing AG, 2016, pp. 1–69.
[4] C. de Boor, A Practical Guide to Splines, revised ed., Springer-Verlag, New York, 2001.
[5] L.L. Schumaker, Spline Functions: Basic Theory, third ed., Cambridge University Press, 2007.
[6] T. Lyche, C. Manni, H. Speleers, Foundations of spline theory: B-splines, spline approximation, and hierarchical refinement, in: T. Lyche, et al. (Eds.), Splines and PDEs: From Approximation Theory to Numerical Linear Algebra, in: Lecture Notes in Mathematics, vol. 2219, Springer International Publishing AG, 2018, pp. 1–76.
[7] T.J.R. Hughes, J.A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4135–4195.
[8] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs, Isogeometric Analysis: Toward Integration of CAD and FEA, John Wiley & Sons, Chichester, 2009.
[9] L. Beirão da Veiga, A. Buffa, G. Sangalli, R. Vázquez, Mathematical analysis of variational isogeometric methods, Acta Numer. 23 (2014) 157–287.
[10] D.R. Forsey, R.H. Bartels, Hierarchical B-spline refinement, ACM Siggraph Comput. Graph. 22 (1988) 205–212.
[11] T.W. Sederberg, J. Zheng, A. Bakenov, A. Nasri, T-splines and T-NURCCs, ACM Trans. Graph. 22 (2003) 477–484.
[12] J. Deng, F. Chen, X. Li, C. Hu, W. Tong, Z. Yang, Y. Feng, Polynomial splines over hierarchical T-meshes, Graph. Models 70 (2008) 76–86.
[13] C. Giannelli, B. Jüttler, H. Speleers, THB-splines: The truncated basis for hierarchical splines, Comput. Aided Geom. Design 29 (2012) 485–498.
[14] L. Beirão da Veiga, A. Buffa, G. Sangalli, R. Vázquez, Analysis-suitable T-splines of arbitrary degree: Definition, linear independence and approximation properties, Math. Models Methods Appl. Sci. 23 (2013) 1979–2003.
[15] T. Dokken, T. Lyche, K.F. Pettersen, Polynomial splines over locally refined box-partitions, Comput. Aided Geom. Design 30 (2013) 331–356.
[16] N. Engleitner, B. Jüttler, Patchwork B-spline refinement, Comput. Aided Des. 90 (2017) 168–179.
[17] F. Patrizi, T. Dokken, Linear dependence of bivariate minimal support and locally refined B-splines over LR-meshes, Comput. Aided Geom. Design 77 (2020) 101803.
[18] A. Bressan, Some properties of LR-splines, Comput. Aided Geom. Design 30 (2013) 778–794.
[19] A. Bressan, B. Jüttler, A hierarchical construction of LR meshes in 2D, Comput. Aided Geom. Design 37 (2015) 9–24.
[20] K.A. Johannessen, F. Remonato, T. Kvamsdal, On the similarities and differences between classical hierarchical, truncated hierarchical and LR B-splines, Comput. Methods Appl. Mech. Engrg. 291 (2015) 64–101.
[21] K.A. Johannessen, T. Kvamsdal, T. Dokken, Isogeometric analysis using LR B-splines, Comput. Methods Appl. Mech. Engrg. 269 (2014) 471–514.
[22] B.G. Lee, T. Lyche, K. Mørken, Some examples of quasi-interpolants constructed from local spline projectors, in: T. Lyche, L.L. Schumaker (Eds.), Mathematical Methods in CAGD: Oslo 2000, Vanderbilt University Press, Nashville, 2001, pp. 243–252.
[23] H. Speleers, C. Manni, Effortless quasi-interpolation in hierarchical spaces, Numer. Math. 132 (2016) 155–184.
[24] H. Speleers, Hierarchical spline spaces: Quasi-interpolants and local approximation estimates, Adv. Comput. Math. 43 (2017) 235–255.
[25] W.F. Mitchell, M.A. McClain, A survey of *hp*-adaptive strategies for elliptic partial differential equations, in: T.E. Simos (Ed.), Recent Advances in Computational and Applied Mathematics, Springer Dordrecht, 2011, pp. 227–258.
[26] J. Cao, Z. Chen, X. Wei, Y.J. Zhang, A finite element framework based on bivariate simplex splines on triangle configurations, Comput. Methods Appl. Mech. Engrg. 357 (2019) 112598.