ICES REPORT 18-22

November 2018

Multi-degree B-splines: Algorithmic computation and properties

by

Deepesh Toshniwal, Hendrik Speleers, Rene■ R. Hiemstra, Carla Manni, Thomas J. R. Hughes

THE UNIVERSITY OF TEXAS AT AUSTIN
Institute for Computational Engineering and Science
ICES

The Institute for Computational Engineering and Sciences
The University of Texas at Austin
Austin, Texas 78712

Reference: Deepesh Toshniwal, Hendrik Speleers, Rene■ R. Hiemstra, Carla Manni, Thomas J. R. Hughes, "Multi-degree B-splines: Algorithmic computation and properties," ICES REPORT 18-22, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, November 2018.

Multi-degree B-splines: Algorithmic computation and properties

Deepesh Toshniwal[a,*], Hendrik Speleers[b], René R. Hiemstra[a], Carla Manni[b], Thomas J. R. Hughes[a]

[a] *Institute for Computational Engineering and Sciences, University of Texas at Austin, USA*
[b] *Department of Mathematics, University of Rome Tor Vergata, Italy*

Abstract

This paper addresses theoretical considerations behind the algorithmic computation of polynomial multi-degree spline basis functions as presented in [23]. The approach in [23] breaks from the reliance on computation of integrals recursively for building B-spline-like basis functions that span a given multi-degree spline space. The gains in efficiency are indisputable; however, the theoretical robustness needs to be examined. In this paper, we show that the construction of [23] yields linearly independent functions with the minimal support property that span the entire multi-degree spline space and form a convex partition of unity.

Keywords: Non-uniform degrees, Smooth splines, Linear independence, Algorithmic computation

1. Introduction

Polynomial splines are well-established tools in the areas of geometric modeling and computational engineering analysis. Classically, given $p \in \mathbb{Z}_{\geq 0}$, a univariate spline is built by gluing together finitely many pieces drawn from $\mathcal{P}_p$, the vector space of polynomials of degree $\leq p$, such that the pieces join with specified orders of smoothness. The vector space of such splines is well-understood and a special set of basis functions called B(asis)-splines can be built for the same [3, 15, 11]. B-splines can be evaluated in a numerically stable manner and possess several properties (e.g., non-negativity, local support, partition of unity) that make them indispensable for the tasks of geometric modeling and design. On the other hand, B-splines also possess optimal approximation properties and are rapidly becoming the tool of choice for performing engineering analysis within the framework of isogeometric analysis [9]. Several multivariate generalizations of univariate B-splines have been formulated (e.g., tensor-product B-splines/NURBS [13, 14], T-splines [17, 2], THB-splines [7] and LR-splines [6]).

All spline constructions mentioned above assume a constant polynomial degree (in each parametric dimension for multivariate splines) and focus only on local mesh-size adaptivity. This precludes local polynomial-degree adaptive splines and is an inherently restrictive assumption. Indeed, relaxing the requirement for a spline to have polynomial pieces of the same degree would be immensely powerful. Such a flexible notion of splines would allow modeling complex geometries with fewer control points, and the same would lead to more efficient engineering analysis. The introduction of univariate multi-degree splines in the context of geometric modeling [18] and isogeometric analysis [23] was motivated by this observation. However, the use of splines consisting of different degrees was already explored earlier by [12] as an approximation tool.

A univariate multi-degree spline, as the name suggests, is built by gluing together finitely many polynomial pieces of (possibly) different degrees with some specified orders of smoothness. This notion clearly generalizes the aforementioned concept of constant degree splines. Furthermore, it can be shown that there exists a set of basis functions that span the vector space of multi-degree splines and possess the same properties that make B-splines useful for geometric modeling and engineering analysis. These basis functions are called multi-degree B-splines or MDB-splines.

Mathematically, MDB-splines can be recursively defined by using integral relations [20, 19]. This definition's complexity makes it overly inefficient for practical purposes and several alternative evaluation schemes have been proposed [18, 10, 23, 1]. However, most of these alternatives either restrict

*Corresponding author

Email addresses: `deepesh@ices.utexas.edu` (Deepesh Toshniwal), `speleers@mat.uniroma2.it` (Hendrik Speleers), `rene@ices.utexas.edu` (René R. Hiemstra), `manni@mat.uniroma2.it` (Carla Manni)

the choice of degrees [18] and orders of smoothness [10], or they need to solve several (small) linear systems [1]. The exception is the approach proposed in [23] — the focus of this paper.

An efficient algorithmic evaluation of MDB-splines was proposed in [23]; a small MATLAB toolbox incorporating the same has been provided by [22]. Representing the smoothness (or continuity) constraints satisfied by the polynomial pieces of a multi-degree spline in the form of a matrix, [23] proposed an algorithm to explicitly build its nullspace (i.e., without solving any linear systems). Each element of the nullspace is a multi-degree spline by construction, and [23] conjectured that the output of their algorithm is, in fact, exactly the entire set of MDB-splines that span the multi-degree spline space. The purpose of our paper is to prove this conjecture. A forthcoming paper [8] will focus on extending the theoretical and algorithmic approach presented herein to the setting of generalized Tchebycheffian splines.

In Section 2, the concept of multi-degree splines is rigorously defined and the similarities between the properties of B-splines and MDB-splines is underscored. In particular, Section 2.3 collects properties of MDB-splines that can be found in published literature. Using these known properties, Section 3 derives some additional properties of MDB-splines. Section 4 starts off by giving an overview of the algorithmic evaluation scheme proposed by [23] and, thereafter, uses the properties derived in Section 3 to conclusively demonstrate the correctness of the proposed algorithm. Specifically, Theorem 4.3 is the main result of this paper and shows that the multi-degree splines built by the algorithm form a convex partition of unity, are linearly independent, and span the entire multi-degree spline space. After presenting an example (many more can be found in [23, 22]) and a brief note on efficient implementation, we conclude the paper in Section 5.

2. Polynomial splines: B-splines and their generalization

In this section we present preliminary concepts about smooth polynomial splines defined on a partition of an interval, $\Omega := [a, b] \subset \mathbb{R}$. In particular, we will allow the spline pieces to have different polynomial degrees, thereby introducing the concept of multi-degree spline spaces. We also present a set of basis functions for such spaces called multi-degree B-splines (or MDB-splines) and list some of their properties. Classical B-splines are a special case of MDB-splines.

2.1. Polynomial splines

We start by partitioning the interval Ω into a finite number of points (called breakpoints) and subintervals (called elements); the space of polynomial splines on Ω will be defined with respect to this partition. Thereafter, we define degree and smoothness distributions on the partition.

Definition 2.1 (Breakpoints and elements)**.** The $m+1$ strictly increasing real numbers x_i, such that $a =: x_0 < x_1 < \cdots < x_m := b$, will be called breakpoints that partition Ω. The breakpoints define the intervals $[x_{i-1}, x_i)$ for $i = 1, \dots, m$, which will be called elements.

Definition 2.2 (Degree distribution)**.** The map $\boldsymbol{p} : \{1, \dots, m\} \mapsto \mathbb{Z}_{\geq 0}$ will be called a degree distribution.

Definition 2.3 (Smoothness distribution)**.** The map $\boldsymbol{r} : \{0, 1, \dots, m-1, m\} \mapsto \mathbb{Z}_{\geq -1}$ such that $\boldsymbol{r}(0) = \boldsymbol{r}(m) = -1$ will be called a smoothness distribution.

Using the above notation, we can define a space of polynomial splines on (a subdomain of) Ω. The non-negative integer $p_i := \boldsymbol{p}(i)$ will be used to specify the polynomial degree on the i^{th} element while the integer $r_i := \boldsymbol{r}(i)$ will be used to specify the order of smoothness across the i^{th} breakpoint. Let $\mathcal{P}_p$ be the vector space of polynomials of degree $\leq p$.

Definition 2.4 (Multi-degree spline space)**.** Given degree and smoothness distributions, we define a polynomial spline space on $[x_i, x_j)$, $0 \leq i < j \leq m$, as

$$\mathcal{S}(i,j) := \mathcal{S}^{\boldsymbol{r}}_{\boldsymbol{p}}(i,j) := \Big\{ [x_i, x_j) \xrightarrow{f} \mathbb{R} \ : \ f\big|_{[x_{k-1},x_k)} \in \mathcal{P}_{p_k} \ , \ i < k \leq j \ , \text{ and } \\ D^r_- f(x_k) = D^r_+ f(x_k) \ , \ i < k < j \ , \ 0 \leq r \leq r_k \Big\} \ . \tag{1}$$

The space $\mathcal{S} := \mathcal{S}(0, m)$ will be called a multi-degree spline space.

The spline space $\mathcal{S}$ is our main object of study and the spaces $\mathcal{S}(i,j)$ will be useful for the same. Note that all spaces in (1) are defined on a half-open interval. Since the domain of interest, Ω, is a closed interval, we assume that the extension $\mathcal{S}$ to Ω is defined by taking the limit from the left at the right end point b.

Before proceeding, we place the following mild compatibility assumption on the degree and smoothness distributions; this assumption will be in effect for the rest of the paper.

Assumption 2.5 (Degree-smoothness compatibility)**.** For all $1 \leq i \leq m-1$, $r_i \leq \min\{p_i, p_{i+1}\}$.

Given Assumption 2.5, the dimension of $\mathcal{S}(i,j)$ can be cleanly determined using classical arguments; see [5] for a proof, for example. A non-classical proof using the rank-nullity theorem is provided in Appendix A for the sake of completeness. The following result presents the associated dimension formula and uses the definitions

$$\boldsymbol{\theta}(i,j) := \sum_{k=i+1}^{j} (p_k+1)\ , \qquad \boldsymbol{\phi}(i,j) := \sum_{k=i+1}^{j-1} (r_k+1)\ , \qquad \boldsymbol{\nu}(i,j) := \boldsymbol{\theta}(i,j) - \boldsymbol{\phi}(i,j)\ . \tag{2}$$

Lemma 2.6.

$$\dim\left(\mathcal{S}(i,j)\right) = \boldsymbol{\nu}(i,j)\ .$$

When considering the space $\mathcal{S}$, we will denote $n := \boldsymbol{\nu}(0,m)$. It should be observed that the above result signifies that all continuity constraints satisfied by $\mathcal{S}(i,j)$ are linearly independent and thus decrease the dimension of a fully discontinuous piecewise-polynomial space as fast as possible.

Remark 2.7. In Assumption 2.5, the choice of $r_i = \min\{p_i, p_{i+1}\}$ is especially interesting. Assuming $p_i \leq p_{i+1}$, this choice implies that the polynomial $f|_{[x_{i-1},x_i)}$ is completely specified by the polynomial $f|_{[x_i,x_{i+1})}$. Specifically, the value and first p_i derivatives of the latter at x_i completely determine the former. In turn, this implies that when $r_i = p_i = p_{i+1}$, the restriction $f|_{[x_{i-1},x_{i+1})}$ is a polynomial of degree p_i.

2.2. B-splines

Before we examine properties of $\mathcal{S}$ in full generality, we consider the special choice of a constant degree distribution such that the space reduces to the well-understood space of B-splines.

Let $m \in \mathbb{N}$, $\boldsymbol{p} := p \in \mathbb{Z}_{\geq 0}$, and let $\boldsymbol{r}$ be a smoothness distribution. Then, $\mathcal{S}$ is the space of B-splines of polynomial degree p, i.e.,

$$\mathcal{S} = \Big\{ [a,b) \xrightarrow{f} \mathbb{R}\ :\ f\big|_{[x_{k-1},x_k)} \in \mathcal{P}_p\ ,\ 1 \leq k \leq m\ ,\ \text{and} \\ D_-^r f(x_k) = D_+^r f(x_k)\ ,\ 0 < k < m\ ,\ 0 \leq r \leq r_k \Big\}\ .$$

The evaluation of the B(asis)-splines that span $\mathcal{S}$ can be done in a rather efficient manner by using the well-known Cox–de Boor recursion formula. First, we define the B-spline knot vector $\boldsymbol{t}$ as

$$\boldsymbol{t} := [t_1,\ t_2,\ \dots,\ t_{n+p+1}] := [\underbrace{x_0,\ \dots,\ x_0}_{p-r_0 \text{ times}},\ \underbrace{x_1,\ \dots,\ x_1}_{p-r_1 \text{ times}},\ \dots,\ \underbrace{x_{m-1},\ \dots,\ x_{m-1}}_{p-r_{m-1} \text{ times}},\ \underbrace{x_m,\ \dots,\ x_m}_{p-r_m \text{ times}}]\ ,$$

and initiate the recursion by defining

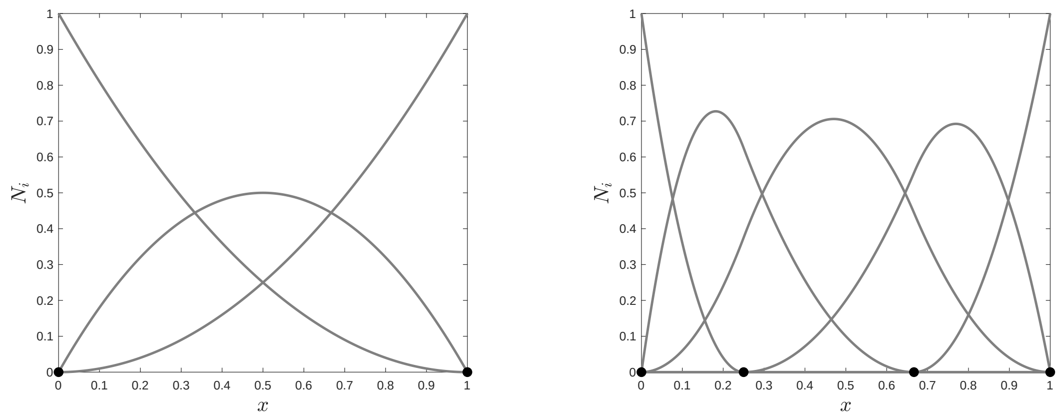
$$N_{i,0}(x) := \begin{cases} 1\ , & \text{if } t_i \leq x < t_{i+1}\ , \\ 0\ , & \text{otherwise}\ . \end{cases}$$

Thereafter, the recursion proceeds for $0 < q \leq p$,

$$N_{i,q}(x) := \frac{x - t_i}{t_{i+q} - t_i} N_{i,q-1}(x) + \frac{t_{i+q+1} - x}{t_{i+q+1} - t_{i+1}} N_{i+1,q-1}(x)\ .$$

These relations can be used to build the B-splines $N_i := N_{i,p}$ that span $\mathcal{S}$. Observe that only convex combinations of lower-degree B-spline evaluations are needed for evaluating higher-degree B-splines, thus making the recursion numerically stable.

B-splines are a special basis for the constant degree spline space. They possess a myriad of properties that make them useful for geometric modeling and engineering analysis. Some of theses properties are listed below and their proof can be found in, e.g., [3, 15, 11].



(a) Bernstein–Bézier polynomials (b) B-splines

Figure 1: On $\Omega = [0, 1]$, the figures (a) and (b) show the B-splines corresponding to the data chosen in Examples 2.9 and 2.10, respectively. Breakpoints in both figures are displayed as filled disks. In the special case where $m = 1$, as in (a), the B-splines are polynomials and are called the Bernstein–Bézier polynomials.

Proposition 2.8 (B-spline properties)**.**

(a) Local support: $N_i(x) = 0$ *for* $x \notin [t_i, t_{i+p+1}]$.

(b) Non-negativity: $N_i(x) > 0$ *for* $x \in (t_i, t_{i+p+1})$.

(c) End-point smoothness: N_i *is exactly* C^{α_i} *smooth at* t_i *and exactly* C^{β_i} *smooth at* t_{i+p+1}*, where* $\alpha_i := p - \max\{j \geq 0 : t_i = t_{i+j}\} - 1$ *and* $\beta_i := p - \max\{j \geq 0 : t_{i+p+1} = t_{i-j+p+1}\} - 1$.

(d) Partition of unity: $\sum_{i=1}^{n} N_i(x) = 1$ *for all* $x \in [a, b)$.

(e) Basis: $\{N_i : i = 1, \dots, n\}$ *are linearly independent and span the space* $\mathcal{S}$.

Example 2.9 (Bernstein–Bézier polynomials)*.* Let $\Omega = [0, 1]$, $m = 1$, $\boldsymbol{p} = 2$. Then, the spline space $\mathcal{S}$ is simply the space of quadratic polynomials on $[0, 1]$ and the corresponding B-splines are simply the Bernstein–Bézier polynomials of degree 2; see Figure 1(a) where the B-splines have been plotted.

Example 2.10 (B-splines and extraction matrix)*.* Let $\Omega = [0, 1]$, $m = 3$, $x_1 = 1/4$, $x_2 = 2/3$, $\boldsymbol{p} = 2$ and $\boldsymbol{r} = 1$. Then, the spline space $\mathcal{S}$ is spanned by $n = 5$ quadratic B-splines; see Figure 1(b) where the B-splines have been plotted. For a B-spline N_i shown in Figure 1(b), the restriction of N_i to any element $[x_{j-1}, x_j)$ in the domain is a quadratic polynomial and therefore can be expanded in terms of the quadratic Bernstein–Bézier basis on $[x_{j-1}, x_j)$. Then, denoting the Bernstein–Bézier basis on element $[x_{j-1}, x_j)$ with $B_{k,j}$, $1 \leq k \leq 3$, we can express the B-splines N_i as

$$\begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \\ N_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \frac{5}{8} & \frac{5}{8} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{3}{8} & \frac{3}{8} & 1 & \frac{4}{9} & \frac{4}{9} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{5}{9} & \frac{5}{9} & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} B_{1,1} \\ B_{2,1} \\ B_{3,1} \\ B_{1,2} \\ \vdots \\ B_{3,3} \end{bmatrix}.$$

The 5×9 matrix on the right hand side is called an extraction operator following terminology introduced in [4, 16]. Intuitively, this matrix extracts smooth B-splines from fully discontinuous Bernstein–Bézier polynomials.

2.3. Multi-degree B-splines

It turns out that, in general, a basis for the spline space $\mathcal{S}$ in the setting of non-constant degree distributions cannot be computed using a simple formula akin to the Cox–de Boor recursion. Nonetheless, there does exist a more complex approach for recursively building a basis for the space $\mathcal{S}$. While this approach is not suitable for practical implementations, it can be rigorously established that the basis functions it produces are B-spline-like, i.e., they possess the same properties that make B-splines useful for a myriad of applications. This section presents a discussion focused on these basis functions and collects results from published literature [20, 1].

Consider the space $\mathcal{S}$ when $\boldsymbol{p}$ is not forced to be a constant degree distribution. As before, we will denote with $N_1, \ldots, N_n$ the multi-degree B(asis)-splines or MDB-splines that span $\mathcal{S}$. We first need to define some additional notation. Following [5, 1], the two MDB-spline knot vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ are defined as

$$\begin{aligned}
\boldsymbol{u} &:= [u_1,\ u_2,\ \ldots,\ u_n] := [\underbrace{x_0,\ \ldots,\ x_0}_{p_1-r_0 \text{ times}},\ \underbrace{x_1,\ \ldots,\ x_1}_{p_2-r_1 \text{ times}},\ \ldots,\ \underbrace{x_{m-1},\ \ldots,\ x_{m-1}}_{p_m-r_{m-1} \text{ times}}]\ , \\
\boldsymbol{v} &:= [v_1,\ v_2,\ \ldots,\ v_n] := [\underbrace{x_1,\ \ldots,\ x_1}_{p_1-r_1 \text{ times}},\ \ldots,\ \underbrace{x_{m-1},\ \ldots,\ x_{m-1}}_{p_{m-1}-r_{m-1} \text{ times}},\ \underbrace{x_m,\ \ldots,\ x_m}_{p_m-r_m \text{ times}}]\ .
\end{aligned} \tag{3}$$

It can be verified that $u_i < v_i$ and $u_i \leq v_{i-1}$ for all i. Note that in the special case of a constant degree distribution $\boldsymbol{p} = p$, the two MDB-spline knot vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ relate to the B-spline knot vector $\boldsymbol{t}$ as follows:

$$u_i = t_i, \quad v_i = t_{i+p+1}, \quad i = 1, \ldots, n.$$

We will refer to $\boldsymbol{u}$ as the left MDB-spline knot vector and to $\boldsymbol{v}$ as the right MDB-spline knot vector.

With $p := \max_i p_i$, the MDB-splines $N_i := N_{i,p}$ are recursively defined. The MDB-spline $N_{i,q}$, $0 \leq q \leq p$ is supported on the interval $[u_i, v_{i-p+q}]$ and can be evaluated at $x \in [x_j, x_{j+1}) \subset [u_i, v_{i-p+q}]$ as follows:

$$N_{i,q}(x) := \begin{cases} 1\ , & x_j \leq x < x_{j+1} \text{ and } q = p - p_{j+1}\ , \\ \displaystyle\int_{-\infty}^{x} \left[\frac{N_{i,q-1}(y)}{d_{i,q-1}} - \frac{N_{i+1,q-1}(y)}{d_{i+1,q-1}} \right] dy\ , & q > p - p_{j+1}\ , \\ 0\ , & \text{otherwise}\ , \end{cases}$$

where

$$d_{i,q} := \int_{-\infty}^{\infty} N_{i,q}(y)\ dy\ ,$$

and we used the convention that if $d_{i,q} = 0$ then

$$\int_{-\infty}^{x} \frac{N_{i,q}(y)}{d_{i,q}}\ dy := \begin{cases} 1\ , & x \geq u_i\ , \\ 0\ , & \text{otherwise}\ . \end{cases}$$

The significant complexity of the above recursion compared to the Cox–de Boor recursion can be readily appreciated.

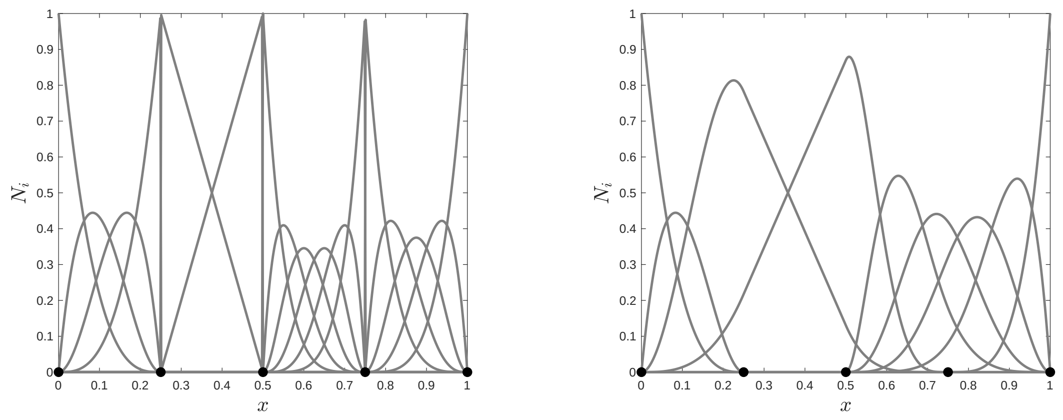
As mentioned earlier and as suggested by the name "MDB-splines", even in the general setting of non-constant polynomial degree distributions, the functions N_i possess the same properties as B-splines do. These are presented in Proposition 2.11 below and use the knot vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, as well as the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ defined next. The meaning of the vectors $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is elucidated by Proposition 2.11. Let $\iota_{\boldsymbol{u}}, \iota_{\boldsymbol{v}} : \{1, \ldots, n\} \to \{0, \ldots, m\}$ be maps such that

$$u_i = x_{\iota_{\boldsymbol{u}}(i)}\ , \qquad v_i = x_{\iota_{\boldsymbol{v}}(i)}\ ,$$

and define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as

$$\begin{aligned}
\boldsymbol{\alpha} &:= [\alpha_1,\ \alpha_2,\ \ldots,\ \alpha_n]\ , & \alpha_i &:= p_{\iota_{\boldsymbol{u}}(i)} - \max\{j \geq 0 : u_i = u_{i+j}\} - 1\ , \\
\boldsymbol{\beta} &:= [\beta_1,\ \beta_2,\ \ldots,\ \beta_n]\ , & \beta_i &:= p_{\iota_{\boldsymbol{v}}(i)-1} - \max\{j \geq 0 : v_i = v_{i-j}\} - 1\ .
\end{aligned} \tag{4}$$

It should be observed that, by definition, α_i and β_i are bounded from below by $r_{\iota_{\boldsymbol{u}}(i)}$ and $r_{\iota_{\boldsymbol{v}}(i)}$, respectively, and from above by $p_{\iota_{\boldsymbol{u}}(i)} - 1$ and $p_{\iota_{\boldsymbol{v}}(i)-1} - 1$, respectively. The following properties of MDB-splines can be deduced from the recursive evaluation scheme above; see, e.g., [1].



(a) (b)

Figure 2: On $\Omega = [0, 1]$, the figures (a) and (b) show the MDB-splines corresponding to the data chosen in Example 2.12. Breakpoints in both figures are displayed as filled disks.

Proposition 2.11 (MDB-spline properties)**.**

(a) Local support: $N_i(x) = 0$ *for* $x \notin [u_i, v_i]$.

(b) Non-negativity: $N_i(x) > 0$ *for* $x \in (u_i, v_i)$.

(c) End-point smoothness: N_i *is exactly* C^{α_i} *smooth at* u_i *and exactly* C^{β_i} *smooth at* v_i.

(d) Partition of unity: $\sum_{i=1}^{n} N_i(x) = 1$ *for all* $x \in [a, b)$.

(e) Basis: $\{N_i : i = 1, \dots, n\}$ *are linearly independent and span the space* $\mathcal{S}$.

Note that item (c) of Proposition 2.11 implies that the MDB-spline N_i does not present any super-smoothness at the end points, i.e., it is not C^{α_i+1} smooth at u_i and not C^{β_i+1} smooth at v_i. When comparing Proposition 2.11 with Proposition 2.8, it is clear that MDB-splines are a natural extension of classical B-splines.

Example 2.12 (MDB-splines). Let $\Omega = [0, 1]$, $m = 4$, $(x_1, x_2, x_3) = (1/4, 1/2, 3/4)$, $(p_1, p_2, p_3, p_4) = (3, 1, 5, 4)$. If we choose $(r_1, r_2, r_3) = (-1, -1, -1)$, then the above recursive definition of MDB-splines yields the splines in Figure 2(a). Alternatively, if we choose $(r_1, r_2, r_3) = (1, 1, 3)$, then MDB-splines shown in Figure 2(b) are obtained. Note that the MDB-splines in Figure 2(a) are equal to Bernstein–Bézier basis functions on each element.

Sums of subsets of MDB-splines can be used to define the notion of transition functions in the following manner,

$$T_i := \sum_{j=i}^{n} N_j \ . \tag{5}$$

Transition functions are elements of $\mathcal{S}$ and, following Proposition 2.11, are non-trivial only on $[u_i, v_{i-1})$, i.e., T_i is identically 0 on $[a, u_i)$ and 1 on $[v_{i-1}, b)$. Furthermore, it follows from the above definition that T_i is C^{α_i} smooth at u_i and $C^{\beta_{i-1}}$ smooth at v_{i-1}. The following result from [1] will be useful for proving additional properties of MDB-splines in Section 3.

Lemma 2.13 (End-point conditions uniquely determine transition functions)**.** *For* $u_i < v_{i-1}$*, let* $T := T_i\big|_{[u_i, v_{i-1})}$ *and let* $j := \iota_{\boldsymbol{u}}(i)$ *and* $k := \iota_{\boldsymbol{v}}(i-1)$*. Then,* $T \in \mathcal{S}(j, k)$ *is uniquely determined by the following end-point smoothness conditions:*

- $D_+^r T(u_i) = 0 \ , \ 0 \le r \le \alpha_i$*, and*
- $D_-^r T(v_{i-1}) = \delta(r, 0) \ , \ 0 \le r \le \beta_{i-1}$,

where $\delta(r,0)$ is the Kronecker delta. The above end-point smoothness conditions are linearly independent. Furthermore,

$$D_+^{\alpha_i+1}T(u_i) \neq 0 \neq D_-^{\beta_{i-1}+1}T(v_{i-1}) \ .$$

Proof. We reproduce the proof of T's uniqueness from [1] for completeness and because similar ideas will be used later in the proof of Lemma 3.2. From Lemma 2.6 and by definition of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, it can be verified that

$$\dim\left(\mathcal{S}(j,k)\right) = \boldsymbol{\nu}(j,k) = \alpha_i + \beta_{i-1} + 2 \ . \tag{6}$$

Thus, the dimension of the space is equal to the number of interpolation conditions. Furthermore, the following choice of interpolation nodes properly interlaces with the breakpoints of $\mathcal{S}(j,k)$ [5],

$$\big[\underbrace{x_j,\ \dots,\ x_j}_{\alpha_i+1 \text{ times}},\ \underbrace{x_k,\ \dots,\ x_k}_{\beta_{i-1}+1 \text{ times}}\big] \ ,$$

thus making the interpolation problem well posed. That is, the α_i+1 interpolation conditions satisfied by T at u_i are linearly independent of the $\beta_{i-1}+1$ interpolation conditions it satisfies at v_{i-1}, and T is uniquely determined by them.

The statement of non-vanishing $(\alpha_i+1)^{th}$ right derivative (resp., $(\beta_{i-1}+1)^{th}$ left derivative) of T at u_i (resp., v_{i-1}) follows from Equation (5) and Proposition 2.11, item (c); see [1]. ■

Finally, we reproduce without proof a special case of the knot insertion result from [1]. For the same breakpoints and degree distribution, the following result relates a smoother set of MDB-splines to one that is of lower regularity.

Lemma 2.14 (Knot insertion)**.** *Given breakpoints x_i and a degree distribution $\boldsymbol{p}$, let $\boldsymbol{r}$ and $\hat{\boldsymbol{r}}$ be smoothness distributions, both compatible with $\boldsymbol{p}$ (see Assumption 2.5), such that for some $1 \le i \le m-1$,*

$$\hat{r}_j = \begin{cases} r_j \ , & j \neq i \ , \\ r_j + 1 \ , & j = i \ . \end{cases}$$

With $\mathcal{S} := \mathcal{S}_{\boldsymbol{p}}^{\boldsymbol{r}}(0,m)$ and $\hat{\mathcal{S}} := \mathcal{S}_{\boldsymbol{p}}^{\hat{\boldsymbol{r}}}(0,m)$, denote the respective MDB-splines with N_i, $i = 1,\dots,n$, and $\hat{N}_i$, $i = 1,\dots,n-1$. Then, there exist constants $0 \le \gamma_{j,1}, \gamma_{j,2} \le 1$, $j = 1,\dots,n-1$, such that

$$\hat{N}_j = \gamma_{j,1}N_j + \gamma_{j,2}N_{j+1} \ .$$

Furthermore, with $\boldsymbol{u}$, $\boldsymbol{v}$, $\hat{\boldsymbol{u}}$ and $\hat{\boldsymbol{v}}$ defined in the way described before, and $\hat{\imath} := \max\{l : \iota_{\hat{\boldsymbol{v}}}(l) = i\}$, the following hold:

(a) $\gamma_{j,1} = 1$ *for* $1 \le j \le \hat{\imath}$,

(b) $\gamma_{j,2} = 1$ *for* $\hat{\imath} + \hat{r}_i + 2 \le j \le n-1$, *and*

(c) $\gamma_{j,2} + \gamma_{j+1,1} = 1$ *for* $1 \le j \le n-2$.

3. Some more properties of MDB-splines

In Section 2.3, we presented a specific collection of results on MDB-splines that can be found in published literature. In the current section, we improve upon this by using the results from Section 2.3 to prove certain additional properties of MDB-splines. In particular, the following will help provide an alternative characterization of MDB-splines and underscore their uniqueness. These characteristics will be used later in Section 4 to develop a rather efficient evaluation algorithm for them. We start by presenting a direct implication of Lemma 2.13.

Corollary 3.1 (Transition functions are not super-smooth)**.** *For $u_i < v_{i-1}$, let $T := T_i\big|_{[u_i,v_{i-1})}$ and let $j := \iota_{\boldsymbol{u}}(i)$ and $k := \iota_{\boldsymbol{v}}(i-1)$. Consider a breakpoint $x_l \in (x_j, x_k) = (u_i, v_{i-1})$ and let $r_l < \min\{p_l, p_{l+1}\}$. Then, T is C^{r_l} smooth at x_l but not C^{r_l+1} smooth.*

Proof. We proceed by contradiction. Let T be C^{r_l+1} smooth at x_l. Then, T is an element of the spline space $\hat{\mathcal{S}}(j,k) := \mathcal{S}_{\boldsymbol{p}}^{\hat{\boldsymbol{r}}}(j,k)$ where $\hat{\boldsymbol{r}}$ is obtained from $\boldsymbol{r}$ by adding 1 to the l^{th} entry. However, using (6) we have $\dim\left(\hat{\mathcal{S}}(j,k)\right) = \dim(\mathcal{S}(j,k)) - 1 = \alpha_i + \beta_{i-1} + 1$, thereby implying an over-constrained interpolation problem for the end-point smoothness conditions that T must satisfy. This contradicts Lemma 2.13 and therefore implies the claim. ■

Lemma 3.2 (Minimal support property). *For $0 \le i \le n$, let $j := \iota_{\boldsymbol{u}}(i)$ and $k := \iota_{\boldsymbol{v}}(i)$. Let $f \in \mathcal{S}(j,k)$ vanish $\alpha_i + 1$ times at u_i and $\beta_i + 1$ times at v_i. Then, $f = cN_i$ for some $c \in \mathbb{R}$.*

Proof. The proof is similar to that of Lemma 2.13. From Lemma 2.6 and by definition of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, it can be readily verified that

$$\dim(\mathcal{S}(j,k)) = \boldsymbol{\nu}(j,k) = \alpha_i + \beta_i + 3 .$$

We can impose $\boldsymbol{\nu}(j,k)$ linearly independent interpolation conditions on this space and get a unique solution. We choose the points

$$[\underbrace{x_j,\ \ldots,\ x_j}_{\alpha_i+1 \text{ times}},\ (x_j + x_k)/2,\ \underbrace{x_k,\ \ldots,\ x_k}_{\beta_i+1 \text{ times}}]$$

as the interpolation nodes, and it can be verified that they properly interlace with the breakpoints of $\mathcal{S}(j,k)$ [5], thus proving that the $\alpha_i + 1$ interpolation conditions satisfied by f at u_i are linearly independent of the $\beta_i + 1$ interpolation conditions it satisfies at v_i. Therefore, the dimension of the space spanned by elements of $\mathcal{S}(j,k)$ that satisfy the end-point interpolation conditions is equal to 1. The claim follows immediately since, from Proposition 2.11, $N_i \in \mathcal{S}(j,k)$ satisfies the end-point interpolation conditions. ■

Corollary 3.3 (MDB-splines are not super-smooth).

(a) Let $j := \iota_{\boldsymbol{u}}(i)$ and $k := \iota_{\boldsymbol{v}}(i)$. Consider a breakpoint $x_l \in (x_j, x_k)$ and let $r_l < \min\{p_l, p_{l+1}\}$. Then, N_i is C^{r_l} smooth at x_l but not C^{r_l+1} smooth.

(b) Consider a breakpoint x_i such that $r_i < \min\{p_i, p_{i+1}\}$, and define $i' := \max\{l : \iota_{\boldsymbol{v}}(l) = i\}$. Then, $N_{i'}, N_{i'+1}, \ldots, N_{i'+r_i+1}, N_{i'+r_i+2}$ are the only MDB-splines that are C^{r_i} smooth at x_i but not C^{r_i+1} smooth.

Proof.

(a) The proof is analogous to that of Corollary 3.1 and shows that $f \equiv 0$ is the unique spline in $\mathcal{S}(j,k)$ which satisfies the same end-point smoothness conditions as N_i and is super-smooth across x_l.

(b) The MDB-splines $N_{i'+1}, \ldots, N_{i'+r_i+1}$ all contain x_i in their interior and so are not super-smooth there from part (a). The claim for $N_{i'}$ and $N_{i'+r_i+2}$ follows from the exact end-point smoothness conditions satisfied by them; see Proposition 2.11.

■

The following are the three main results of this section that will help us prove mathematical soundness of the evaluation algorithm from [23].

Lemma 3.4. *Consider a breakpoint x_i such that $r_i < \min\{p_i, p_{i+1}\}$, and define $i' := \max\{l : \iota_{\boldsymbol{v}}(l) = i\}$. Then, the following equality holds for $i' \le k \le i' + r_i + 2$ only if $k = i' + r_i + 2$,*

$$D_-^{r_i+1}\left(\sum_{j=i'}^{k} N_j\right)(x_i) = D_+^{r_i+1}\left(\sum_{j=i'}^{k} N_j\right)(x_i) .$$

Proof. Since $N_1, \ldots, N_{i'-1}$ and $N_{i'+r_i+3}, \ldots, N_n$ are all C^{r_i+1} smooth at x_i, by the partition of unity and local support properties we obtain

$$D^{r_i+1}\left(\sum_{j=i'}^{i'+r_i+2} N_j\right)(x_i) = \begin{cases} 1 \,, & r_i = -1 \,, \\ 0 \,, & r_i \geq 0 \,. \end{cases} \tag{7}$$

Let us now assume that the equality in the claim holds for some $i' \leq k \leq i' + r_i + 2$. Then, from Equation (7) we obtain,

$$\begin{aligned} & D_-^{r_i+1}\left(\sum_{j=k+1}^{i'+r_i+2} N_j\right)(x_i) = D_+^{r_i+1}\left(\sum_{j=k+1}^{i'+r_i+2} N_j\right)(x_i) \,, \\ \Leftrightarrow \quad & D_-^{r_i+1}\left(\sum_{j=k+1}^{n} N_j\right)(x_i) = D_+^{r_i+1}\left(\sum_{j=k+1}^{n} N_j\right)(x_i) \,, \\ \Leftrightarrow \quad & D_-^{r_i+1} T_{k+1}(x_i) = D_+^{r_i+1} T_{k+1}(x_i) \,, \end{aligned}$$

where we have again used the super-smoothness of $N_{i'+r_i+3}, \ldots, N_n$ at x_i. The above implies that the transition function T_{k+1} is C^{r_i+1} smooth at x_i. From Corollary 3.1, however, T_{k+1} must be $T_{i'+r_i+3}$ as none of $T_{i'+1}, \ldots, T_{i'+r_i+2}$ are super-smooth at x_i. ■

Corollary 3.5 (Super-smooth linear combinations)**.** *Consider the setup in Lemma 3.4, and define the vector* $\hat{\mathbf{c}} := [\hat{c}_1, \ldots, \hat{c}_{r_i+3}]^T$ *by*

$$\hat{c}_j := D_-^{r_i+1} N_{i'+j-1}(x_i) - D_+^{r_i+1} N_{i'+j-1}(x_i) \,.$$

Then, none of the $\hat{c}_j$ *are zero and, moreover,* $\sum_{j=1}^{q} \hat{c}_j$ *is zero for* $1 \leq q \leq r_i + 3$ *only if* $q = r_i + 3$*. In particular, if there exist constants* f_j *such that* $\sum_{j=q_1}^{q_2} f_j \hat{c}_j = 0$*,* $1 \leq q_1 < q_2 \leq r_i + 3$*, then the linear combination* $\sum_{j=q_1}^{q_2} f_j N_{i'+j-1}$ *is* C^{r_i+1} *smooth at* x_i*.*

Proof. This is a direct consequence of Lemma 3.4. ■

Corollary 3.6 (Knot insertion)**.** *Consider the setup in Lemma 2.14. Let* $\eta_{j,1}, \eta_{j,2}$*,* $1 \leq j \leq n-1$*, be constants such that*

(a) $\eta_{j,1} = 1$ *for* $1 \leq j \leq \hat{i}$*,*

(b) $\eta_{j,2} = 1$ *for* $\hat{i} + \hat{r}_i + 2 \leq j \leq n-1$*, and*

(c) $\eta_{j,2} + \eta_{j+1,1} = 1$ *for* $1 \leq j \leq n-2$*,*

and define functions $\tilde{N}_i$*,* $i = 1, \ldots, n-1$*, using the relations*

$$\tilde{N}_j := \eta_{j,1} N_j + \eta_{j,2} N_{j+1} \,.$$

If $\hat{\mathcal{S}} \ni \tilde{N}_i$*,* $i = 1, \ldots, n-1$*, then* $\tilde{N}_i = \hat{N}_i$ *for all* i*, and* $0 \leq \eta_{j,1} = \gamma_{j,1}, \eta_{j,2} = \gamma_{j,2} \leq 1$*.*

Proof. From the premise, each $\tilde{N}_j$ is supported on $[\hat{u}_j, \hat{v}_j]$ and satisfies the same end-point smoothness conditions as $\hat{N}_j = \gamma_{j,1} N_j + \gamma_{j,2} N_{j+1}$. Indeed, the end-point smoothness of both $\hat{N}_j$ and $\tilde{N}_j$ is dictated by that of N_j at the left end and N_{j+1} at the right end. Thus, Lemma 3.2 implies that the $\tilde{N}_i$ must be scalar multiples of the $\hat{N}_i$. In particular, since the $\hat{N}_i$ are linearly independent, so are the $\tilde{N}_i$.

The above, therefore, implies that the $\tilde{N}_i$ must be exactly equal to the $\hat{N}_i$ since the $\hat{N}_i$ form a partition of unity and, by definition of the constants $\eta_{j,1}, \eta_{j,2}$, so do the splines $\tilde{N}_i$. ■

4. Algorithmic evaluation of MDB-splines

In this section we provide a short overview of the construction presented in [23] (see also [22]) for building multi-degree spline functions, M_i, that span, at least, a subspace of $\mathcal{S}$. Spoiler: actually, they are completely equivalent to the basis functions N_i from the last section and, therefore, span the full space and are linearly independent; this will be shown in Theorem 4.3.

Suppose we are looking to construct spline functions that lie in $\mathcal{S}$. For each of the elements $[x_{i-1}, x_i)$, $i = 1, \ldots, m$, we can build the $p_i + 1$ unique Bernstein–Bézier basis functions $B_{j,i}$, $j = 1, \ldots, p_i + 1$, that span $\mathcal{P}_{p_i}$ on $[x_{i-1}, x_i)$. For each i, extending the functions $B_{j,i}$ outside $[x_{i-1}, x_i)$ by 0, let us relabel them as

$$B_{\boldsymbol{\theta}(0,i-1)+j} := B_{j,i} \ , \qquad j = 1, \ldots, p_i + 1 \ . \tag{8}$$

Next, arrange these relabeled basis functions in a single vector $\boldsymbol{B}$ of length $\boldsymbol{\theta}(0, m)$. It can be easily verified that the basis functions $\{B_j : j = 1, \ldots, \boldsymbol{\theta}(0, m)\}$ coincide with the MDB-splines that span $\mathcal{S}_{\boldsymbol{p}}^{-\boldsymbol{1}}(0, m)$. Then, our objective is to construct a matrix $\boldsymbol{H}$ — in the spirit of the extraction matrix in Example 2.10 — of size $n \times \boldsymbol{\theta}(0, m)$ such that $\mathcal{S} \supseteq \operatorname{span}\{M_1, \ldots, M_n\}$ where,

$$\boldsymbol{M} := \boldsymbol{H}\boldsymbol{B} \ . \tag{9}$$

As mentioned above, if $\boldsymbol{H}$ is taken to be an identity matrix then $\boldsymbol{M} = \boldsymbol{B}$ and they will span the space $\mathcal{S}_{\boldsymbol{p}}^{-\boldsymbol{1}}(0, m)$. In order to create a set of basis functions that span $\mathcal{S}$, we

(a) build continuity constraints at all interior breakpoints according to the chosen smoothness distribution $\boldsymbol{r}$, and

(b) construct rows of $\boldsymbol{H}$ as suitable elements in the nullspace of those constraints.

Section 4.1 outlines the construction of continuity constraints while Section 4.2 presents a sparse nullspace construction. Finally, Section 4.3 proves the equivalence of M_i and N_i.

4.1. Continuity constraints at the i^{th} breakpoint

For $1 \leq i \leq m - 1$ let $\boldsymbol{K}_{i,-}$ be a matrix of size $(p_i + 1) \times (r_i + 1)$, whose j^{th} column is given by,

$$\begin{bmatrix} 0 & \cdots & 0 & D_-^{j-1} B_{\boldsymbol{\theta}(0,i)-j+1}(x_i) & \cdots & D_-^{j-1} B_{\boldsymbol{\theta}(0,i)}(x_i) \end{bmatrix}^T \ , \tag{10}$$

and let $\boldsymbol{K}_{i,+}$ be a matrix of size $(p_{i+1} + 1) \times (r_i + 1)$, whose j^{th} column is given by,

$$\begin{bmatrix} -D_+^{j-1} B_{\boldsymbol{\theta}(0,i)+1}(x_i) & \cdots & -D_+^{j-1} B_{\boldsymbol{\theta}(0,i)+j}(x_i) & 0 & \cdots & 0 \end{bmatrix}^T \ . \tag{11}$$

Using these matrices, we can build the matrix $\boldsymbol{K}_i$ of size $\boldsymbol{\theta}(0, m) \times (r_i + 1)$ which contains all constraints required to enforce C^{r_i} smoothness at x_i. This matrix is defined row-wise in the following manner:

1. the $(\boldsymbol{\theta}(0, i - 1) + k)^{th}$ row of $\boldsymbol{K}_i$ is equal to the k^{th} row of $\boldsymbol{K}_{i,-}$,
2. the $(\boldsymbol{\theta}(0, i) + k)^{th}$ row of $\boldsymbol{K}_i$ is equal to the k^{th} row of $\boldsymbol{K}_{i,+}$, and,
3. all other rows of $\boldsymbol{K}_i$ are identically zero.

It can be easily verified that for a row vector $\boldsymbol{f}$ such that $\boldsymbol{f}\boldsymbol{K}_i = \boldsymbol{0}$, the spline defined by $\boldsymbol{f}\boldsymbol{B}$ is going to be C^{r_i} smooth across x_i. Therefore, once all the matrices $\boldsymbol{K}_i$ have been assembled, the only remaining step is the construction of $\boldsymbol{H}$ such that its rows span their collective left-nullspace. The matrix $\boldsymbol{H}$ is a multi-degree spline extraction or an MDB-spline extraction, and we employ Algorithm 1 below for its construction. Algorithm 1 is a more efficient implementation of the one proposed in [23] and has been reproduced from [22]. Aside from efficiency, both algorithms produce exactly equivalent functions.

Algorithm 1 Computation of $\boldsymbol{H}$ (Section 4.1)

$\boldsymbol{H} \leftarrow$ identity matrix (size : $\boldsymbol{\theta}(0,m) \times \boldsymbol{\theta}(0,m)$)
for $i = 1 : m-1$ **do**
  $\boldsymbol{L} \leftarrow \boldsymbol{H}\boldsymbol{K}_i$
  for $j = 1 : r_i + 1$ **do**
    $\overline{\boldsymbol{H}} \leftarrow$ sparse nullspace of j^{th} column of $\boldsymbol{L}$
    $\boldsymbol{H} \leftarrow \overline{\boldsymbol{H}}\boldsymbol{H}$
    $\boldsymbol{L} \leftarrow \overline{\boldsymbol{H}}\boldsymbol{L}$
return $\boldsymbol{H}$

Algorithm 2 nullspace of $\hat{\boldsymbol{c}}$ (Section 4.2)

$\hat{\boldsymbol{H}} \leftarrow \boldsymbol{0}$ (size: $(q-1) \times q$)
$\hat{\boldsymbol{H}}(1,1) \leftarrow 1$
for $i = 1 : q-2$ **do**
  $\hat{\boldsymbol{H}}(i, i+1) \leftarrow -\dfrac{\hat{c}_i \hat{\boldsymbol{H}}_1(i,i)}{\hat{c}_{i+1}}$
  $\hat{\boldsymbol{H}}(i+1, i+1) \leftarrow 1 - \hat{\boldsymbol{H}}(i, i+1)$
$\hat{\boldsymbol{H}}(q-1, q) = 1$
return $\hat{\boldsymbol{H}}$

4.2. Sparse nullspace construction

Algorithm 1 utilizes Algorithm 2 for building a sparse left-nullspace of a given column from the continuity constraints matrix. The latter algorithm's functioning, motivated by the minimal support property of MDB-splines (Lemma 3.2; also see [3, 23]), can be explained as follows for a given column vector $\hat{\boldsymbol{c}}$.

Let $\hat{\boldsymbol{c}} = [\hat{c}_1, \ldots, \hat{c}_q]^T$, $\hat{c}_i \neq 0$, $2 \leq q \in \mathbb{N}$; we will call $\hat{\boldsymbol{c}}$ the constraint vector. The matrix $\hat{\boldsymbol{H}}$ built by Algorithm 2 can be explicitly represented as

$$\hat{\boldsymbol{H}} = \begin{bmatrix} 1 & h_1 & & & \\ & \overline{h}_1 & \ddots & & \\ & & \ddots & h_{q-2} & \\ & & & \overline{h}_{q-2} & h_{q-1} \end{bmatrix},$$

with $h_i := -\sum_{j=1}^{i} \hat{c}_j / \hat{c}_{i+1}$ and $\overline{h}_i := 1 - h_i = \sum_{j=1}^{i+1} \hat{c}_j / \hat{c}_{i+1} = -(\hat{c}_{i+2}/\hat{c}_{i+1}) h_{i+1}$. The next two lemmas outline some properties that the above definition endows upon the matrix $\hat{\boldsymbol{H}}$ depending on the properties of the constraint vector $\hat{\boldsymbol{c}}$. These properties will be useful for proving the equivalence between spline functions M_i and multi-degree basis functions N_i in Theorem 4.3.

Lemma 4.1. *The following statements hold.*

(a) The rows of $\hat{\boldsymbol{H}}$ are in the left-nullspace of $\hat{\boldsymbol{c}}$, i.e., $\hat{\boldsymbol{H}}\hat{\boldsymbol{c}} = \boldsymbol{0}$.

(b) If $\sum_{j=1}^{q} \hat{c}_j = 0$, then $h_{q-1} = 1$.

(c) If $\sum_{j=1}^{i} \hat{c}_j = 0 \Rightarrow i = q$, then $\hat{\boldsymbol{H}}$ has full rank, its rows span the left-nullspace of $\hat{\boldsymbol{c}}$, and all $h_i, \overline{h}_i \neq 0$.

Proof. Properties (a) and (b) hold by construction and can be easily verified. For Property (c), note that if $\sum_{j=1}^{i} \hat{c}_j = 0 \Rightarrow i = q$, then either $\sum_{j=1}^{i} \hat{c}_j \neq 0$ for all i, or only $\sum_{j=1}^{q} \hat{c}_j = 0$. In both cases, none of the $h_i, \overline{h}_i$ are zero and the full rank of $\hat{\boldsymbol{H}}$ follows. ■

Next, consider a column vector $\overline{\boldsymbol{c}} = [\boldsymbol{0},\ \hat{\boldsymbol{c}},\ \boldsymbol{0}]^T$ obtained by padding the vector $\hat{\boldsymbol{c}}$ with q_A and q_B zeros above and below, respectively. Define the block diagonal matrix $\overline{\boldsymbol{H}}$ as,

$$\overline{\boldsymbol{H}} = \begin{bmatrix} \boldsymbol{I}_A & & \\ & \hat{\boldsymbol{H}} & \\ & & \boldsymbol{I}_B \end{bmatrix},$$

where $\boldsymbol{I}_A$ and $\boldsymbol{I}_B$ are identity matrices of size q_A and q_B, respectively, and $\hat{\boldsymbol{H}}$ is as defined above.

Lemma 4.2. *The following statements hold by construction.*

(a) The rows of $\overline{\boldsymbol{H}}$ are in the left-nullspace of $\overline{\mathbf{c}}$, i.e., $\overline{\boldsymbol{H}}\overline{\mathbf{c}} = \mathbf{0}$.

(b) If $\hat{\boldsymbol{H}}$ has full rank, then $\overline{\boldsymbol{H}}$ has full rank and its rows span the left-nullspace of $\overline{\mathbf{c}}$.

Proof. The proof follows directly from Lemma 4.1. ■

4.3. Equivalence of N_i and M_i

The algorithmic construction summarized in this section thus far is highly intuitive and holds a big advantage over others in [20, 19, 21, 1]. Indeed, it does not rely on indirect or expensive methodologies (such as solutions of linear systems, or recursive computations of global integrals, for example) for building splines in $\mathcal{S}$. Instead, a sparse nullspace of the continuity constraints is explicitly and efficiently built; each row of this nullspace represents a smooth multi-degree spline by construction.

That said, [23] did not conclusively pin down properties of the splines M_i in the multi-degree setting. In particular, their linear independence and non-negativity were not established. Since violation of these properties may make M_i unsuitable for applications in geometric modeling and/or engineering analysis, we examine them in the following and show that the spline functions M_i are exactly the same as the MDB-splines N_i.

Theorem 4.3.

(a) With the continuity constraints built as in Section 4.1, all entries of the extraction matrix $\boldsymbol{H}$ output by Algorithm 1 lie in $[0, 1]$.

(b) With $\boldsymbol{M} := \boldsymbol{H}\boldsymbol{B}$, the identity $M_i = N_i$ holds for all $i = 1, \dots, n$. In particular, $\{M_i : i = 1, \dots, n\}$ are linearly independent, span the space $\mathcal{S}$, and form a convex partition of unity.

Proof. Notice that the matrix $\overline{\boldsymbol{H}}$ computed as per Algorithm 1 always has dimensions $(q-1) \times q$ for some $2 \leq q \in \mathbb{N}$. Let us index all the rectangular $\overline{\boldsymbol{H}}$ built during the course of Algorithm 1 according to their order of appearance; this coincides with the order in which continuity constraints at breakpoints x_j are resolved. Doing so, we can express the output $\boldsymbol{H}$ as,

$$\boldsymbol{H} = \overline{\boldsymbol{H}}_{\boldsymbol{\phi}(0,m)} \overline{\boldsymbol{H}}_{\boldsymbol{\phi}(0,m)-1} \cdots \overline{\boldsymbol{H}}_1 \ ,$$

where $\overline{\boldsymbol{H}}_1$ has size $(\boldsymbol{\theta}(0,m) - 1) \times \boldsymbol{\theta}(0,m)$, and $\boldsymbol{\phi}(0,m)$ is the total number of continuity constraints. From Lemma 2.6, the dimension of $\boldsymbol{H}$ is $n \times \boldsymbol{\theta}(0,m)$. Then, to prove the claim we need to show that all $\overline{\boldsymbol{H}}_k$, $k = 1, \dots, \boldsymbol{\phi}(0,m)$, have full rank; we proceed by induction.

Let us consider a sequence of smoothness vectors,

$$-\mathbf{1} =: \boldsymbol{r}^0 < \boldsymbol{r}^1 < \boldsymbol{r}^2 < \cdots < \boldsymbol{r}^{\boldsymbol{\phi}(0,m)} := \boldsymbol{r} \ ,$$

where $\boldsymbol{r}^{k_1} < \boldsymbol{r}^{k_2}$ if for all $j = 1, \dots, m-1$, $r_j^{k_1} \leq r_j^{k_2}$, with strict inequality holding for just one j. If the k^{th} continuity constraint resolved during Algorithm 1 corresponds to imposition of $C^{r_l^k}$ smoothness at the l^{th} breakpoint, $1 \leq l \leq m-1$, we define the entries of $\boldsymbol{r}^k$ using $\boldsymbol{r}^{k-1}$ as

$$r_j^k = r_j^{k-1} + \delta(j, l) \ , \quad j = 1, \dots, m-1 \ ,$$

with $\delta(j, l)$ denoting the Kronecker delta.

Induction hypotheses. For $1 \leq k \leq \boldsymbol{\phi}(0,m)$,

(H.a) The matrix $\overline{\boldsymbol{H}}_k$ has full rank.

(H.b.1) All entries of $\boldsymbol{H}_k := \overline{\boldsymbol{H}}_k \overline{\boldsymbol{H}}_{k-1} \cdots \overline{\boldsymbol{H}}_1$ lie in $[0, 1]$.

(H.b.2) The spline functions $M_{i,k}$ defined using the extraction $\boldsymbol{H}_k := \overline{\boldsymbol{H}}_k \overline{\boldsymbol{H}}_{k-1} \cdots \overline{\boldsymbol{H}}_1$ are equal to the MDB-splines spanning $\mathcal{S}_{\boldsymbol{p}}^{\boldsymbol{r}^k}(0, m)$.

Base step. When $k = 1$, a C^0 smoothness constraint is resolved at some breakpoint. The corresponding constraint vector is simply $[\mathbf{0},\ \hat{\boldsymbol{c}},\ \mathbf{0}]^T$ with $\hat{c} = [1,\ -1]^T$. It is easy to see that $\hat{\boldsymbol{H}}$ computed using Algorithm 2 takes the form $\hat{\boldsymbol{H}} = [1,\ 1]$. Therefore, by Lemma 4.2, $\overline{\boldsymbol{H}}_1$, defined using this $\hat{\boldsymbol{H}}$, has full rank and hypothesis (H.a) is satisfied. Moreover, from the structure of $\overline{\boldsymbol{H}}_1$ and Corollary 3.6, the $M_{i,1}$ defined by it are equal to the MDB-splines spanning $\mathcal{S}_{\boldsymbol{p}}^{\boldsymbol{r}^1}(0, m)$ and hypotheses (H.b.1) and (H.b.2) are satisfied.

Induction step. Assume that both the induction hypotheses are satisfied for some $k > 1$. This implies that, from Corollary 3.5, the smoothness constraint to be resolved at step $k + 1$, $[\mathbf{0},\ \hat{\boldsymbol{c}},\ \mathbf{0}]^T$, is such that $\hat{\boldsymbol{c}}$ satisfies the last condition of Lemma 4.1. This implies, as above, that the corresponding matrix $\hat{\boldsymbol{H}}$ computed using Algorithm 2 is of full rank. Again, from the structure of $\overline{\boldsymbol{H}}_{k+1}$ and Corollary 3.6, the $M_{i,k+1}$ are equal to the MDB-splines spanning $\mathcal{S}_{\boldsymbol{p}}^{\boldsymbol{r}^{k+1}}(0, m)$ and hypotheses (H.b.1) and (H.b.2) are satisfied. ∎

Example 4.4. Consider again the setup in Example 2.12. Then, for the choice of $(r_1, r_2, r_3) = (-1, -1, -1)$, there are no continuity constraints imposed at any of the interior breakpoints, and therefore the MDB-splines are simply equivalent to the Bernstein–Bézier basis on each element. Alternatively, for the choice of $(r_1, r_2, r_3) = (1, 1, 3)$, we can compute the constraint matrices to be,

$$\left[\begin{array}{c}\boldsymbol{K}_{1,-}\\ \hline \boldsymbol{K}_{1,+}\end{array}\right] = \left[\begin{array}{rr}0 & 0\\ 0 & 0\\ 0 & -12\\ 1 & 12\\ \hline -1 & 4\\ 0 & -4\end{array}\right], \quad \left[\begin{array}{c}\boldsymbol{K}_{2,-}\\ \hline \boldsymbol{K}_{2,+}\end{array}\right] = \left[\begin{array}{rr}0 & -4\\ 1 & 4\\ \hline -1 & 20\\ 0 & -20\\ 0 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 0\end{array}\right], \quad \left[\begin{array}{c}\boldsymbol{K}_{3,-}\\ \hline \boldsymbol{K}_{3,+}\end{array}\right] = \left[\begin{array}{rrrr}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & -3840\\ 0 & 0 & 320 & 11520\\ 0 & -20 & -640 & -11520\\ 1 & 20 & 320 & 3840\\ \hline -1 & 16 & -192 & 1536\\ 0 & -16 & 384 & -4608\\ 0 & 0 & -192 & 4608\\ 0 & 0 & 0 & -1536\\ 0 & 0 & 0 & 0\end{array}\right].$$

Using these constraint matrices as the inputs, the matrix $\boldsymbol{H}$ computed using Algorithm 1 is given below,

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & \frac{18}{23} & \frac{18}{23} & \frac{3}{23} & \frac{3}{23} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{5}{23} & \frac{5}{23} & \frac{20}{23} & \frac{20}{23} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \frac{4}{7} & \frac{9}{28} & \frac{5}{28} & \frac{5}{28} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{7} & \frac{159}{322} & \frac{135}{322} & \frac{135}{322} & \frac{15}{46} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{17}{92} & \frac{1445}{4508} & \frac{1445}{4508} & \frac{1105}{2254} & \frac{85}{147} & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{4}{49} & \frac{4}{49} & \frac{9}{49} & \frac{62}{147} & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

It can be verified that the rows of $\boldsymbol{H}$ are in the nullspace of the constraints, and multiplying the above matrix with the Bernstein–Bézier basis functions in Figure 2(a) yields the MDB-splines in Figure 2(b).

Therefore, instead of using the complicated recursive definition for evaluating the MDB-splines N_i, we can simply evaluate the Bernstein–Bézier basis functions on each element (using Cox–de Boor recursion) and linearly combine them using the matrix $\boldsymbol{H}$.

4.4. A brief note on efficient implementation

The previous sections demonstrate how Algorithm 1 can be used to explicitly build an extraction operator that specifies how Bernstein–Bézier basis functions of different polynomial degrees can be linearly combined to yield MDB-splines. This piecewise-polynomial approach is quite general and in most practical cases may be an overkill. Unless one wishes to work with a "checkerboard" degree distribution, p_i will not always be different from p_{i+1}. However, if $p_i = p_{i+1}$, then instead of adopting the algorithmic construction we can:

(a) simply build a B-spline basis of degree p_i on the segment $[x_{i-1}, x_{i+1})$ using Cox–de Boor recursion (see Section 2.2), and

(b) combine the B-spline basis on $[x_{i-1}, x_{i+1})$ with the Bernstein–Bézier basis on elements $[x_{i-2}, x_{i-1})$ and $[x_{i+1}, x_{i+2})$ using Algorithm 1.

The above suggests an alternate formulation and implementation of Algorithm 1 by shifting the perspective from a "piecewise polynomial" approach to a "piecewise B-spline" approach. Such a reformulation would proceed in the following manner.

(a) Partition $\{1, \ldots, m\}$ into maximal sets $J_i := \{m_{i-1} + 1, m_{i-1} + 2, \ldots, m_i\}$, $1 \leq i \leq e$, such that
   - $m_0 := 0$, $m_e := m$,
   - $j, k \in J_i \Rightarrow p_j = p_k$.

(b) Build constant degree B-splines of degree p_j, $j \in J_i$, on $\sigma_i := \bigcup_{j \in J_i} [x_{j-1}, x_j)$ using Cox–de Boor recursion.

(c) Build continuity constraints at the breakpoint x_{m_i} where B-splines on σ_i and σ_{i+1} meet.

(d) Explicitly build the nullspace of the constraints using Algorithm 1.

The above reformulation leaves the function of Algorithm 1 essentially invariant, but allows one to reuse existing efficient implementations of constant degree B-splines for the purpose of building MDB-splines. This point of view was adopted in [23, 22] and we refer the reader to those papers for the corresponding details. In particular, [22] also provides a Matlab toolbox incorporating this reformulated implementation of MDB-splines.

5. Conclusion

We have analyzed the algorithmic evaluation of univariate MDB-splines proposed by [23] and have rigorously proved its correctness. That is, we have shown that the evaluation scheme from [23] produces linearly independent spline functions that span the entire multi-degree spline space $\mathcal{S}$, form a convex partition of unity, and have the minimal support property. The evaluation scheme also allows for an efficient implementation of MDB-splines by leveraging existing B-spline implementations. A small Matlab toolbox demonstrating this feature can be found in the material accompanying [22].

Acknowledgments

D. Toshniwal and T.J.R. Hughes were partially supported by the Office of Naval Research (Grant Nos. N00014-17-1-2119, N00014-17-1-2039, and N00014-13-1-0500), and by the Army Research Office (Grant No. W911NF-13-1-0220). C. Manni and H. Speleers were partially supported by the Mission Sustainability Programme of the University of Rome Tor Vergata through the project IDEAS (CUP E81I18000060005) and by the MIUR Excellence Department Project awarded to the Department of Mathematics, University of Rome Tor Vergata (CUP E83C18000100006); they are members of Gruppo Nazionale per il Calcolo Scientifico – Istituto Nazionale di Alta Matematica.

A. Dimension of $\mathcal{S}(i, j)$

Proof of Lemma 2.6. Recall that $\mathcal{P}_p$ is the vector space of polynomials in variable x of degree $\leq p$, with $\mathcal{P}_p := 0$ for $p < 0$. For breakpoint x_k let $\hat{p}_k := \max\{p_k,\ p_{k+1}\}$ and define

$$\mathcal{J}_k := \begin{cases} (x - x_k)^{r_k+1} \mathcal{P}_{\hat{p}_k - r_k - 1}\ , & i < k < j\ , \\ \mathcal{P}_{\hat{p}_k}\ , & \text{otherwise}\ . \end{cases}$$

The vector space $\mathcal{J}_k$ contains multiples of the polynomial $(x - x_k)^{r_k+1}$ in $\mathcal{P}_{\hat{p}_k}$ when $i < k < j$, and its dimension is

$$\dim(\mathcal{J}_k) = \begin{cases} \hat{p}_k - r_k \ , & i < k < j \ , \\ \hat{p}_k + 1 \ , & \text{otherwise} \ . \end{cases}$$

Then, using Taylor's formula, the smoothness of a spline $f \in \mathcal{S}(i,j)$ at the breakpoint x_k, $i < k < j$, can be interpreted as the condition

$$f|_{[x_{k-1},\ x_k)} - f|_{[x_k,\ x_{k+1})} \in \mathcal{J}_k \ .$$

Therefore, the spline space $\mathcal{S}(i,j)$ can be interpreted as the kernel of the linear map $\overline{\partial}$,

$$\bigoplus_{k=i+1}^{j} \mathcal{P}_{p_k} \xrightarrow{\overline{\partial}} \bigoplus_{k=i}^{j} \mathcal{P}_{\hat{p}_k}/\mathcal{J}_k \ ,$$

where $\mathcal{P}_{\hat{p}_i}/\mathcal{J}_i = \mathcal{P}_{\hat{p}_j}/\mathcal{J}_j = 0$, and $\overline{\partial}$ is obtained by composing the map ∂ (specified below) with the natural quotient map,

$$\partial = \begin{bmatrix} -1 & & & \\ 1 & -1 & & \\ & 1 & \ddots & \\ & & \ddots & -1 \\ & & & 1 \end{bmatrix} \ .$$

We will prove the dimension formula for $\mathcal{S}(i,j)$ using the above setup along with the rank-nullity theorem. Specifically, we will show that $\overline{\partial}$ is a surjection, thereby implying that

$$\begin{aligned} \dim(\mathcal{S}(i,j)) &= \dim\left(\ker \overline{\partial}\right) \ , \\ &= \dim\left(\bigoplus_{k=i+1}^{j} \mathcal{P}_{p_k}\right) - \dim\left(\operatorname{im} \overline{\partial}\right) \ , \\ &= \dim\left(\bigoplus_{k=i+1}^{j} \mathcal{P}_{p_k}\right) - \dim\left(\bigoplus_{k=i}^{j} \mathcal{P}_{\hat{p}_k}/\mathcal{J}_k\right) \ , \\ &= \boldsymbol{\theta}(i,j) - \boldsymbol{\phi}(i,j) = \boldsymbol{\nu}(i,j) \ . \end{aligned}$$

To see that $\overline{\partial}$ is a surjection, consider the following element of $\oplus_{k=i}^{j} \mathcal{P}_{\hat{p}_k}/\mathcal{J}_k$ for some $i < l < j$,

$$\Big(0,\ \cdots,\ 0,\ f_l,\ 0,\ \cdots,\ 0\Big) \ . \tag{12}$$

We will show that the above is the image of an element of $\oplus_{k=i+1}^{j} \mathcal{P}_{p_k}$. Denoting the degree of a polynomial h with $\deg h$, for any polynomials h and g we can find $\tilde{h}$ and $\tilde{g}$ such that $h = \tilde{h}g + \tilde{g}$ and $\deg \tilde{g} \le \min\{\deg g - 1, \deg h\}$. Therefore, we can express the polynomial f_l in the form

$$f_l = \tilde{f}_l (x - x_l)^{r_l+1} - g_{l+1} \ , \tag{13}$$

where $\deg g_{l+1} \le \min\{p_l, p_{l+1}\}$ from Assumption 2.5. Therefore, in the quotient $\mathcal{P}_{\hat{p}_l}/\mathcal{J}_l$, the polynomial f_l is equivalent to $-g_{l+1} \in \mathcal{P}_{p_{l+1}}$. Furthermore, it is clear that, with $f_{l+1} := -g_{l+1}$, we have

$$\Big(0,\ \cdots,\ 0,\ g_{l+1},\ 0,\ \cdots,\ 0\Big) \overset{\overline{\partial}}{\mapsto} \Big(0,\ \cdots,\ 0,\ f_l,\ -f_{l+1},\ \cdots,\ 0\Big) \ . \tag{14}$$

Then, we repeat the process laid out in Equations (12)–(14) for the index $l+1$ instead of l, and so on. The resulting recursion will necessarily terminate at the following element (if not earlier),

$$\Big(0,\ \cdots,\ 0,\ g_j\Big) \overset{\overline{\partial}}{\mapsto} \Big(0,\ \cdots,\ 0,\ f_{j-1},\ 0\Big)$$

because, by definition, $\mathcal{J}_j = \mathcal{P}_{\hat{p}_j}$ and therefore $f_j := -g_j$ is zero in the quotient $\mathcal{P}_{\hat{p}_j}/\mathcal{J}_j$. The above shows that

$$\Big(0,\ \cdots,\ 0,\ g_{l+1},\ g_{l+2},\ \cdots,\ g_j\Big) \overset{\overline{\partial}}{\mapsto} \Big(0,\ \cdots,\ 0,\ f_l,\ 0,\ \cdots,\ 0\Big) \ .$$

The claim of surjectivity follows, thus completing the proof. ■

References

[1] C. V. Beccari, G. Casciola, and S. Morigi. On multi-degree splines. *Computer Aided Geometric Design*, 58:8–23, 2017.

[2] L. Beirão da Veiga, A. Buffa, D. Cho, and G. Sangalli. Analysis-suitable T-splines are dual-compatible. *Computer Methods in Applied Mechanics and Engineering*, 249–252:42–51, 2012.

[3] C. de Boor. *A Practical Guide to Splines, Revised Edition*. Springer-Verlag, 2001.

[4] M. J. Borden, M. A. Scott, J. A. Evans, and T. J. R. Hughes. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87:15–47, 2011.

[5] B. Buchwald and G. Mühlbach. Construction of B-splines for generalized spline spaces generated from local ECT-systems. *Journal of Computational and Applied Mathematics*, 159:249–267, 2003.

[6] T. Dokken, T. Lyche, and K. F. Pettersen. Polynomial splines over locally refined box-partitions. *Computer Aided Geometric Design*, 30:331–356, 2013.

[7] C. Giannelli, B. Jüttler, and H. Speleers. THB-splines: The truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29:485–498, 2012.

[8] R. R. Hiemstra, T. J. R. Hughes, C. Manni, H. Speleers, and D. Toshniwal. A Tchebycheffian extension of multi-degree B-splines: Algorithmic computation and properties. *In preparation*, 2018.

[9] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[10] X. Li, Z. J. Huang, and Z. Liu. A geometric approach for multi-degree splines. *Journal of Computer Science and Technology*, 27:841–850, 2012.

[11] T. Lyche, C. Manni, and H. Speleers. Foundations of spline theory: B-splines, spline approximation, and hierarchical refinement. In T. Lyche et al., editors, *Splines and PDEs: From Approximation Theory to Numerical Linear Algebra*, volume 2219 of *Lecture Notes in Mathematics*, pages 1–76. Springer International Publishing AG, 2018.

[12] G. Nürnberger, L. L. Schumaker, M. Sommer, and H. Strauss. Generalized Chebyshevian splines. *SIAM Journal on Mathematical Analysis*, 15:790–804, 1984.

[13] L. Piegl and W. Tiller. *The NURBS Book, Second Edition*. Springer-Verlag, 1997.

[14] D. F. Rogers. *An Introduction to NURBS: With Historical Perspective*. Morgan Kaufmann, 2001.

[15] L. L. Schumaker. *Spline Functions: Basic Theory, Third Edition*. Cambridge University Press, 2007.

[16] M. A. Scott, M. J. Borden, C. V. Verhoosel, T. W. Sederberg, and T. J. R. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 88:126–156, 2011.

[17] T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22:477–484, 2003.

[18] T. W. Sederberg, J. Zheng, and X. Song. Knot intervals and multi-degree splines. *Computer Aided Geometric Design*, 20:455–468, 2003.

[19] W. Shen and G. Wang. A basis of multi-degree splines. *Computer Aided Geometric Design*, 27:23–35, 2010.

[20] W. Shen and G. Wang. Changeable degree spline basis functions. *Journal of Computational and Applied Mathematics*, 234:2516–2529, 2010.

[21] W. Shen, G. Wang, and P. Yin. Explicit representations of changeable degree spline basis functions. *Journal of Computational and Applied Mathematics*, 238:39–50, 2013.

[22] H. Speleers. Computation of multi-degree B-splines. *Preprint*, arXiv:1809.01598, 2018.

[23] D. Toshniwal, H. Speleers, R. R. Hiemstra, and T. J. R. Hughes. Multi-degree smooth polar splines: A framework for geometric modeling and isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 316:1005–1061, 2017.