Matrix representations for multi-degree B-splines

Matrix representations for multi-degree B-splines

Carolina Vittoria Beccari, Giulio Casciola

Department of Mathematics, University of Bologna, Bologna, Italy

The final published version is available online at:

https://doi.org/10.1016/j.cam.2020.113007

Matrix representations for multi-degree B-splines

Carolina Vittoria Beccari[a,*], Giulio Casciola[a]

[a]*Department of Mathematics, University of Bologna, Piazza di Porta San Donato 5, 40126 Bologna, Italy*

Abstract

The paper is concerned with computing the B-spline basis of a multi-degree spline space, namely a space of piecewise functions comprised of polynomial segments of different degrees. To this aim, we provide a general method to work out a matrix representation relating the sought basis with another one easier to compute. This will allow us, for example, to calculate a multi degree B-spline basis starting from local Bernstein bases of different degrees or from the B-spline basis of a spline space where all sections have the same degree. This change of basis can be translated into a conceptually simple and computationally efficient algorithm for the evaluation of multi-degree B-splines.

Keywords: Multi-Degree spline, B-spline basis, matrix representation, evaluation, B-spline derivative
2010 MSC: 65D07, 65D17, 41A15, 68W25

1. Introduction

Spline functions are classical tools in approximation theory, with application in various fields ranging from geometric modeling and computer-aided design, up to solving partial differential equations within the framework of isogeometric analysis [1]. Classically, a spline is a piecewise function whose pieces belong to a polynomial space of fixed degree d. Removing the constraint that each piece have the same degree yields a more ample family of splines, refereed to as multi-degree splines (MD-splines for short) or also changeable-degree splines [2]. Multi-degree splines are thus piecewise polynomial functions, where each piece may belong to a different space of algebraic polynomials. In this way, the structure and the dimension of an MD-spline space can be chosen so as to involve the least degrees of freedom necessary to approximate an assigned dataset, producing a more efficient representation.

Multi-degree splines were initially investigated in the context of approximation theory with a view to studying their zeros and their potential in the solution of Hermite interpolation problems [3, 4] and curve modeling [5]. Recently they have enjoyed revived interest, not only in the context of geometric modeling [6–8], but also for the resolution of differential problems on multivariate domains within the framework of isogeometric analysis [9, 10]. The usual polynomial splines can also be generalized by allowing each piece to be drawn from an Extended Chebyshev space (see e.g. [11] and references therein, or [12] for a more computationally-oriented approach). In this respect, recent studies [13, 14] are concerned with extending the multi-degree framework to Chebyshevian splines with section spaces of different dimensions.

For practical applications and integration into CAD systems, spline functions are usually expanded in the B-spline basis, for which efficient and stable evaluation algorithms have long been known (see e.g. [15]). Akin to classical splines, MD-spline spaces possess a basis of normalized, compactly supported functions, that we will refer to as the *MDB-spline basis*. Early approaches for the derivation of this basis [2, 16–18] are subject to specific continuity restrictions at the join of pieces of different degrees and rely on integral recurrence relations, which require a symbolic implementation and therefore become overly inefficient as soon as the dimension of the space or the degrees are relatively high. Two recent methods [6, 9], instead, are capable of producing splines that have arbitrary continuity, meaning that, between two pieces of degrees d_i and d_{i+1}, they can be C^k continuous for any $k \leqslant \min\{d_i, d_{i+1}\}$. This behavior properly generalizes the continuity properties of conventional splines to the multi-degree framework.

*Corresponding author.
Email addresses: carolina.beccari2@unibo.it (Carolina Vittoria Beccari), giulio.casciola@unibo.it (Giulio Casciola)

The paper [6] exploits an integral definition of the MDB-spline basis to show that all the classical properties of B-spline basis functions carry over to multi-degree spaces. To efficiently calculate a spline, it is then suggested to resort to another basis of an MD-spline space formed by *transition functions*. The transition functions are somewhat locally supported functions, each of which can easily be computed by solving a system of linear equation of small size. Thus this approach has the advantage of locality, despite involving the resolution of as many linear systems as the dimension of the space.

A purely algebraic approach is instead pursued in [9]. The method starts from a vector containing a sequence of conventional B-spline bases of different degrees and works out a linear operator, called the *H-operator*, which represents the conversion matrix into the MDB-spline basis. In particular, matrix H has as many rows as the dimension of the MD-spline space and as many columns as the sum of the dimensions of the classical spline spaces to be connected. Its construction is a global procedure which, however, turns out to be quite efficient thanks to the sparsity of the involved matrices, see [7] for implementation details. It is noteworthy that the procedure itself does not guarantee that a basis of a proper MDB-spline space be generated. Indeed, a rather elaborate proof of its correctness was provided a posteriori [19].

We finally recall that multi-degree splines, in general, do not possess evaluation formulae akin to Cox-de Boor recurrence relation [20, 21], which is the main tool for computing conventional splines. More precisely, such recurrence relations have only been found restraining the continuity between pieces of different degrees to be (at most) C^1 [8, 22], whereas it was inferred in [6] that they may not exist otherwise.

In this paper we propose an innovative procedure to derive a matrix representation relating the MDB-spline basis with another convenient basis. This representation has the form $\mathbf{N} = \mathrm{M}\,\mathbf{N}_0$, where $\mathbf{N}$ is the sought MDB-spline basis and $\mathbf{N}_0$ should be chosen as a basis for which stable and efficient evaluation algorithms are available. In this way, we can for example choose to express an MDB-spline basis $\mathbf{N}$ in terms of local Bernstein bases, or also of a conventional B-spline basis of maximum degree. The method has a similar flavor as the H-operator, but encompasses some major differences and strengths. In particular, our algorithm relies on iterated knot insertions and degree elevations, familiar tools to spline and geometric-modeling practitioners. From a conceptual point of view, this means that the impact of a round of knot insertion and degree elevation is readily predictable and understandable, whereas from a computational standpoint one is guaranteed to perform efficient operations, both of which, in the multi-degree context, are of local nature. Immediate advantages are that the method generates by construction a basis of a proper MD-spline space and that computations can be optimized by exploiting the sparsity of the involved linear operators. Moreover, one may retrieve the H-operator as a special case of this matrix representation, by simply setting the initial basis $\mathbf{N}_0$ to be the same as in [19]. Thus, a by-product of our approach is to provide an alternative proof of correctness of the H-operator. At the same time, the method in this paper is more general, since it allows us to select the initial basis $\mathbf{N}_0$ from a vast class of spaces so as to minimize computational cost and improve numerical accuracy. To illustrate this potential, we will consider spaces restrained to having C^0 continuity between pieces of different degrees, therefore called C^0 MD-spline spaces. After observing that their MDB-spline basis can efficiently be computed by a proper generalization of Cox-de Boor recurrence scheme, we will discuss how the matrix representation relative to C^0 MDB-splines is often the most compact and entails the lowest computational complexity.

From the numerical point of view, previous works have focused on the efficiency of the evaluation algorithm in terms of number of operations, which is optimized by exploiting the compact support of the B-spline basis functions to either solve a local problem [6] or a global problem characterized by sparsity properties [9]. Instead, a study on the accuracy of computation, similar to those existing for conventional splines [20], has never been undertaken. As regards efficiency, it will become clear that the strength of our approach is the synergetic application of subsequent knot insertion and degree elevation steps, which allows us to further reduce the number of performed operations. In addition, unlike previous papers, we will also consider the problem of validating the accuracy of the procedure by experimentally showing its good behavior in a variety of challenging situations.

The remainder of the paper is organized as follows. Section 2 recalls the necessary background on multi-degree spline spaces and B-spline bases and the results on knot insertion and degree elevation that will serve as building blocks for the matrix representation. In addition, Section 2.2 illustrates how the basis of a C^0 multi-degree spline space can efficiently be evaluated by a simple generalization of Cox-de Boor recurrence scheme, which, to the best of our knowledge, was never discussed elsewhere. Section 3 contains the actual algorithm to convert an assigned B-spline basis into that of a target MD-spline space and discusses how this procedure yields matrix representations in terms of easy-to-compute bases. Section 4 presents a numerical study on the computational aspects of the proposed

method. Conclusions are drawn in Section 5.

2. Multi-degree spline spaces and B-spline bases

In the following we merely recall notions and results that will be necessary for the purpose of this paper, all of which are taken from [6]. For further acquaintance on multi-degree splines, we refer readers to [6] itself and [2, 5, 7, 9, 16–19].

A multi-degree spline (MD-spline, for short) is a piecewise polynomial function where each segment may have different degree. To define a space of such functions on a real interval $[a, b]$, we shall consider a breakpoint sequence $\mathcal{X} := \{x_i\}_{i=1}^{q}$, $a \equiv x_0 < x_1 < \ldots < x_q < x_{q+1} \equiv b$, and a vector of positive integers $\mathbf{d} := \left(d_0, \ldots, d_q\right)$, where d_i represents the degree on the interval $[x_i, x_{i+1}]$. In addition we shall take a vector $\mathcal{K} := \left(k_1, \ldots, k_q\right)$ of non negative integers, where k_i represents the continuity at the breakpoint x_i, such that

$$0 \leqslant k_i \leqslant \min(d_{i-1}, d_i), \quad i = 1, \ldots, q.$$

A multi-degree spline space corresponding to the assigned configuration of breakpoint intervals, degrees and continuities is hence defined as follows.

Definition 1 (Multi-degree spline space)**.** Given a partition $\mathcal{X} = \{x_i\}_{i=0}^{q+1}$ on the bounded and closed interval $[a, b]$, an associated vector of polynomial degrees $\mathbf{d} := \left(d_0, \ldots, d_q\right)$, and a vector $\mathcal{K} := \left(k_1, \ldots, k_q\right)$ of degrees of smoothness, the corresponding space of *multi-degree splines* $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ is the set

$$\begin{aligned}\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K}) := \Big\{ f \,\Big|\, & \text{there exist } p_i \in \mathcal{P}_{d_i}, i = 0, \ldots, q, \text{ such that:} \\ & \text{i) } f(x) = p_i(x) \text{ for } x \in [x_i, x_{i+1}],\ i = 0, \ldots, q; \\ & \text{ii) } D^{\ell} p_{i-1}(x_i) = D^{\ell} p_i(x_i) \text{ for } \ell = 0, \ldots, k_i,\ \ i = 1, \ldots, q \Big\}.\end{aligned}$$

In the above definition, $\mathcal{P}_{d_i}$ is intended as the space of algebraic polynomials of degree at most d_i and $\mathcal{P}_{\mathbf{d}}$ is the piecewise polynomial space on $\mathcal{X}$ whose restriction to $[x_i, x_{i+1}]$ coincides with $\mathcal{P}_{d_i}$, for $i = 0, \ldots, q$.

It follows from standard arguments that a multi-degree spline space $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ has dimension K, where

$$K = d_0 + 1 + \sum_{i=1}^{q} (d_i - k_i) \quad \text{or, equivalently,} \quad K = \sum_{i=1}^{q} (d_{i-1} - k_i) + d_q + 1.$$

Remark 1. *We remark that when $d_{i-1} = d_i = k_i$, then a multi-degree spline is a polynomial of degree d_i in the entire interval $[x_{i-1}, x_{i+1}]$.*

Remark 2 (Conventional splines are multi-degree splines)**.** *When all degrees are the same, namely $d_i = d$, for all $i = 0, \ldots, q$, then $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ reduces to the conventional spline space $\mathcal{S}(\mathcal{P}_d, \mathcal{X}, \mathcal{K})$, intended as a classical polynomial spline space of degree d. Therefore, conventional splines are a special instance of multi-degree splines.*

In order to generalize classical spline theory to the multi-degree context, it is appropriate to associate with a spline space two different extended partitions, as firstly proposed in [4] and later used for the definition and computation of MDB-spline basis functions in [6]. When dealing with conventional spline spaces, these two partitions are reduced to one.

Definition 2 (Extended partitions)**.** The set of knots $\mathbf{s} := \left\{s_j\right\}_{j=1}^{K}$ is called the *left extended partition* associated with a multi-degree spline space $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ if and only if:

i) $s_1 \leqslant s_2 \leqslant \ldots \leqslant s_K$;

ii) $s_{d_0+1} \equiv a$;

iii) $\{s_{d_0+2}, \ldots, s_K\} \equiv \{\underbrace{x_1, \ldots, x_1}_{d_1-k_1 \text{ times}}, \ldots, \underbrace{x_q, \ldots, x_q}_{d_q-k_q \text{ times}}\}$.

Similarly, the set of knots $\mathbf{t} := \left\{t_j\right\}_{j=1}^{K}$ is called the *right extended partition* associated with $\mathcal{S}(\mathcal{P}_\mathbf{d}, \mathcal{X}, \mathcal{K})$ if and only if:

i) $t_1 \leqslant t_2 \leqslant \ldots \leqslant t_K$;

ii) $t_{K-d_q} \equiv b$;

iii) $\left\{t_1, \ldots, t_{K-d_q-1}\right\} \equiv \{\underbrace{x_1, \ldots, x_1}_{d_0-k_1 \text{ times}}, \ldots, \underbrace{x_q, \ldots, x_q}_{d_{q-1}-k_q \text{ times}}\}$.

Remark 3. *Hereinafter we will confine ourselves to considering* clamped *partitions, i.e. extended partitions where* $s_1 = \ldots = s_{d_0+1} = x_0$ *and* $t_{K-d_q} = \ldots = t_K = x_{q+1}$. *Generalizing the results in this paper to periodic partitions is straightforward, relying on the classical approach of wrapping breakpoint intervals (see e.g. [15, Chapter 8.1]).*

In multi-degree spline spaces it is possible to identify a set of basis functions with properties similar to those of conventional B-splines. By virtue of this analogy, this basis is called MDB-spline basis and denoted by $N_{i,m}$, $i = 1 \ldots, K$, where $m := \max_j\{d_j\}$ [6]. Each element of the basis can be determined through an integral recurrence relation which warrants that it be nonnegative, have compact support and enjoy the properties listed below. A graphical illustration of such bases is presented later on (see Figure 2).

Proposition 1 (Properties of MDB-splines)**.** *The MDB-spline functions* $\{N_{i,m}\}_{i=1}^{K}$ *of* $\mathcal{S}(\mathcal{P}_\mathbf{d}, \mathcal{X}, \mathcal{K})$ *enjoy the following properties:*

i) Local Support*:* $N_{i,m}(x) = 0$ *for* $x \notin [s_i, t_i]$*;*

ii) Positivity*:* $N_{i,m}(x) > 0$ *for* $x \in (s_i, t_i)$*;*

iii) End Point*:* $N_{i,m}$ *vanishes exactly*

- $d_{ps_i} - \max\{j \geqslant 0 \mid s_i = s_{i+j}\}$ *times at* s_i*,*
- $d_{pt_i-1} - \max\{j \geqslant 0 \mid t_{i-j} = t_i\}$ *times at* t_i*,*

where ps_i *is the index of the break-point associated with the knot* s_i *and* pt_i *is the index of the break-point associated with the knot* t_i*;*

iv) Partition of unity*:* $\sum_i N_{i,m}(x) = 1$, $\forall x \in [a, b]$.

Remark 4. *Note that an MDB-spline* $N_{i,m}$ *cannot have continuity higher than* C^{k_j} *at a breakpoint* $x_j \in (s_i, t_i)$*. This can easily be proved using the fact that any function in a multi-degree spline space* $\mathcal{S}(\mathcal{P}_\mathbf{d}, \mathcal{X}, \mathcal{K})$ *can have at most as many zeros as* $\dim \mathcal{S}(\mathcal{P}_\mathbf{d}, \mathcal{X}, \mathcal{K}) - 1$ *(see [4]).*

We conclude by observing that a function $f \in \mathcal{S}(\mathcal{P}_\mathbf{d}, \mathcal{X}, \mathcal{K})$, expanded in the MDB-spline basis as

$$f(x) = \sum_{i=1}^{K} c_i \, N_{i,m}(x), \quad x \in [a, b],$$

can be represented on a single breakpoint interval $[x_j, x_{j+1}]$ by only involving the nonvanishing MDB-splines. This yields

$$f(x) = \sum_{i=\ell-d_j}^{\ell} c_i \, N_{i,m}(x), \quad x \in [x_j, x_{j+1}), \quad s_\ell \leqslant x_j < \min\left(s_{\ell+1}, b\right),$$

where we shall take $\min\left(s_{\ell+1}, b\right)$ to be b if $s_{\ell+1}$ is not defined or equivalently if $\ell + 1 > K$.

2.1. Knot insertion and degree elevation for multi-degree splines

Throughout the paper we will strongly rely on the two classical spline-related tools of knot insertion and degree elevation, that have a natural counterpart in the context of multi-degree splines. In particular, the following Propositions 2 and 3 can be proved by standard arguments, by merely using the compact support and partition of unity properties of the MDB-spline basis functions.

Proposition 2. *(Knot insertion at a break point) Let $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ and $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \widehat{\mathcal{K}})$ be MD-spline spaces with same breakpoint sequence and degrees. Let also the associated continuity vectors be $\mathcal{K} = (k_1, \dots, k_j, \dots, k_q)$ and $\widehat{\mathcal{K}} = (k_1, \dots, k_j - 1, \dots, k_q)$, for $j \in \{1, \dots, q\}$. Then, the corresponding MDB-spline bases $\{N_{i,m}\}_{i=1}^{K}$ and $\{\hat{N}_{i,m}\}_{i=1}^{K+1}$ are related through*

$$N_{i,m} = \alpha_i \hat{N}_{i,m} + (1 - \alpha_{i+1})\hat{N}_{i+1,m}, \quad i = 1, \dots, K, \tag{1}$$

with coefficients

$$\alpha_i = \begin{cases} 1, & i = 1, \dots, \ell - d_j, \\ \beta_i, & i = \ell - d_j + 1, \dots, \ell - d_j + k_j, \\ 0, & i = \ell - d_j + k_j + 1, \dots, K + 1, \end{cases} \tag{2}$$

where ℓ is such that $s_\ell \leqslant x_j \leq \min(s_{\ell+1}, b)$, and $0 < \beta_i < 1$.

Proposition 3. *(Degree elevation on one interval) Let $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ and $\mathcal{S}(\mathcal{P}_{\widehat{\mathbf{d}}}, \mathcal{X}, \mathcal{K})$ be MD-spline spaces with same breakpoint sequence and continuities. Let also the associated degree vectors be $\mathbf{d} = (d_0, \dots, d_j, \dots, d_q)$ and $\widehat{\mathbf{d}} = (d_0, \dots, d_j + 1, \dots, d_q)$, for $j \in \{0, \dots, q\}$. Then, the corresponding MDB-spline bases $\{N_{i,m}\}_{i=1}^{K}$ and $\{\hat{N}_{i,\hat{m}}\}_{i=1}^{K+1}$ satisfy the identity*

$$N_{i,m} = \alpha_i \hat{N}_{i,\hat{m}} + (1 - \alpha_{i+1})\hat{N}_{i+1,\hat{m}}, \quad i = 1, \dots, K, \tag{3}$$

with coefficients

$$\alpha_i = \begin{cases} 1, & i = 1, \dots, \ell - d_j, \\ \gamma_i, & i = \ell - d_j + 1, \dots, \ell, \\ 0, & i = \ell + 1, \dots, K + 1, \end{cases} \tag{4}$$

where ℓ is such that $s_\ell \leqslant x_j \leq \min(s_{\ell+1}, b)$,and $0 < \gamma_i < 1$.

Remark 5. *Within an MD-spline space knot insertion can be performed in two different ways: one consists in diminishing the continuity at a breakpoint by one order, as in Proposition 2; the other corresponds to placing an additional breakpoint in an existing interval (x_j, x_{j+1}), thus generating two subintervals. Proposition 2 is deliberately concerned with the former circumstance, since it is the only one of interest in this paper. However, also knot insertion along a breakpoint interval is featured by a relation akin to (1), which similarly follows from the compact support and partition of unity properties of MDB-splines.*

Remark 6. *Degree elevating a multi-degree spline space means raising the degree d_j on a given interval $[x_j, x_{j+1}]$ is raised, while leaving all other degrees d_i, $i \neq j$ unchanged. Therefore, like knot insertion, also degree elevation is a local operation, meaning that it only locally affects the structure of a space and its MDB-spline basis functions. Obviously, the same is not true in the context of conventional splines, where the degree must simultaneously be raised on all intervals.*

Classically, that is for conventional splines, knot insertion and degree elevation formulae akin to (1) and (3) are viewed as tools to compute the unknown coefficients β_i and γ_i in (2)–(4), using the fact that the bases of both spaces can efficiently be calculated by well-established algorithms. These coefficients are typically used to update the representation of a spline function (or often the control polygon of a parametric spline curve) from the "starting" space to the "arrival" space. Due to the formal analogy of relations (1) and (3) to their conventional counterpart, the same holds for MD-spline spaces. In particular, knowing the two bases, the expressions of the coefficients β_i and γ_i were derived in [6, Propositions 2 and 3]. It is worth noticing that, in this paper, we will instead deviate from the classical view of knot insertion and degree elevation, since these tools will be used to compute an unknown MDB-spline basis from a known one.

2.2. Recurrence relations for C^0 multi-degree splines

The purpose of this section is to show that a particular class of MD-splines admits simple and efficient evaluation algorithms, which represent a direct generalization of classical B-splines recurrence relations. More precisely, these splines are identified as follows.

Definition 3 (C^0 multi-degree (MD-) spline space)**.** A space $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ as in Definition 1 is said to be a C^0 multi-degree spline space if its functions are C^0 continuous at the join of pieces of different degrees. Therefore it is defined by

$$\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K}), \quad \text{with} \quad \mathcal{K} = (k_0, k_1, \dots, k_q) \quad \text{and} \quad k_i = 0 \quad \text{if} \quad d_{i-1} \neq d_i, \quad i = 1, \dots, q.$$

The following result represents an adaptation to the MD-spline setting of the classical Cox-de Boor recurrence formula [20, 21], accomplished by using the left and right extended partitions and a local index of recursion related to the specific degree associated with the evaluation interval. The proof is straightforward and therefore omitted.

Proposition 4 (Cox-de Boor recurrence relation for C^0 MD-splines)**.** *Let $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ be a C^0 MD-spline space with left extended partition $\mathbf{s} = \{s_j\}_{j=1}^K$, right extended partition $\mathbf{t} = \{t_j\}_{j=1}^K$ and $m := \max_j\{d_j\}_{j=0}^q$. Then an MDB-spline $N_{i,m}$, for $i = 1, \dots, K$, can be computed on the break-point interval $[x_j, x_{j+1})$ contained in its support $[s_i, t_i]$ by the following recurrence scheme:*

$$\begin{aligned} N_{\ell,m-d_j}(x) &= \begin{cases} 1 & x_j \leqslant x < x_{j+1} \quad \text{and} \quad \ell \quad \text{such that} \quad s_\ell \leqslant x_j < \min(s_{\ell+1}, b), \\ 0 & \text{otherwise}, \end{cases} \\ N_{i,n}(x) &= \frac{x - s_i}{t_{i-m+n-1} - s_i} N_{i,n-1}(x) + \frac{t_{i-m+n} - x}{t_{i-m+n} - s_{i+1}} N_{i+1,n-1}(x), \quad n = m - d_j + 1, \dots, m. \end{aligned} \tag{5}$$

Note that each undefined $N_{i,n}$ function in (5) shall be regarded as the zero function and $0/0$ shall be intended as 0.

Moreover, derivatives of C^0 MDB-splines can efficiently be evaluated by the following recurrence process.

Proposition 5 (Recurrence relation for derivatives of C^0 MD-splines)**.** *The setting being the same of Proposition 4, for $x \in [x_j, x_{j+1})$ and ℓ such that $s_\ell \leqslant x_j < \min(s_{\ell+1}, b)$, the right derivatives of MDB-splines can be computed by the following recurrence scheme:*

$$D_+ N_{i,n}(x) = \begin{cases} 0 & n \leqslant m - d_j, \\ (d_j - m + n)\left(\dfrac{N_{i,n-1}(x)}{t_{i-m+n-1} - s_i} - \dfrac{N_{i+1,n-1}(x)}{t_{i-m+n} - s_{i+1}} \right), & n = m - d_j + 1, \dots, m, \quad i = \ell - n + m - d_j, \dots, \ell. \end{cases} \tag{6}$$

If the evaluation point is instead chosen so that $x \in (x_j, x_{j+1}]$, the above relation holds with the left-hand side of (6) replaced by $D_- N_{i,n}(x)$.

We remark that, not only Propositions 4 and 5 are an immediate generalization of standard evaluation methods, but are also reduced to their classical counterpart when applied within conventional spline spaces. This should be no wonder in that the B-spline basis of a C^0 MDB-spline space is locally nothing else than either a Bernstein-type basis or a conventional B-spline-type basis. An illustration such a basis is given in Figure 2(a) for the C^0 MD-spline space on $[a, b] = [0, 4]$ featured by $\mathcal{X} = \{1, 2, 3\}$, $\mathbf{d} = (3, 2, 2, 2)$ and $\mathcal{K} = (0, 1, 1)$, whose MDB-splines are the C^0 join of the the cubic Bernstein basis relative to $[0, 1]$ and the (conventional) quadratic B-spline basis relative to $[1, 4]$.

3. Evaluation and matrix representations of multi-degree B-splines

In this section we develop an algorithmic way to construct the B-spline basis of a multi degree spline space. The proposed approach yields a matrix representation of the basis functions in terms of another suitable basis and is based on iterated application of the following Propositions 6 and 7, which are closely related to the concepts of knot insertion and degree elevation recalled in the previous section.

Since no ambiguity may arise, **in the reminder of the paper we will drop the subscript m and use the shorter notation N_i to indicate $N_{i,m}$.**

Proposition 6 (Reverse Knot Insertion (RKI))**.** *Let $\mathcal{S} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ and $\widehat{\mathcal{S}} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \widehat{\mathcal{K}})$ be multi-degree spline spaces having same breakpoint sequence $\mathcal{X}$ and same vector of degrees $\mathbf{d}$. Moreover let the associated continuity vectors $\mathcal{K} = \left(k_1, \dots, k_q\right)$ and $\widehat{\mathcal{K}} = \left(\hat{k}_1, \dots, \hat{k}_q\right)$ be such that $\hat{k}_j = k_j - 1$, for a given $j \in \{1, \dots, q\}$, and $\hat{k}_i = k_i$, for all $i = 1, \dots, q$, $i \neq j$. Denoted by $\{\hat{N}_i\}_{i=1}^{K+1}$ the B-spline basis of $\widehat{\mathcal{S}}$, the coefficients $\alpha_i = \beta_i$, $i = \ell - d_j + 1, \dots, \ell - d_j + k_j$, in (2) can be computed through the following formula:*

$$1 - \alpha_i = -\alpha_{i-1} \frac{D_-^{k_j} \hat{N}_{i-1}|_{x_j} - D_+^{k_j} \hat{N}_{i-1}|_{x_j}}{D_-^{k_j} \hat{N}_i|_{x_j} - D_+^{k_j} \hat{N}_i|_{x_j}}, \tag{7}$$

where $\alpha_{\ell - d_j} = 1$.

Proof. Since the MDB-spline function $N_i \in \mathcal{S}$ must have continuity k_j at x_j, we shall require that

$$D_-^{k_j} N_i|_{x_j} = D_+^{k_j} N_i|_{x_j}.$$

Substituting (1) in the above equality yields

$$D_-^{k_j} \left(\alpha_i \hat{N}_i + (1 - \alpha_{i+1}) \hat{N}_{i+1}\right)|_{x_j} = D_+^{k_j} \left(\alpha_i \hat{N}_i + (1 - \alpha_{i+1}) \hat{N}_{i+1}\right)|_{x_j},$$

from which (7) follows straightforwardly. □

Proposition 7 (Reverse Degree Elevation (RDE))**.** *Let $\mathcal{S} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ and $\widehat{\mathcal{S}} \equiv \mathcal{S}(\mathcal{P}_{\hat{\mathbf{d}}}, \mathcal{X}, \mathcal{K})$ be multi-degree spline spaces having same breakpoint sequence $\mathcal{X}$ and same vector of continuities $\mathcal{K}$. Moreover let the associated degree vectors $\mathbf{d} = \left(d_0, \dots, d_q\right)$ and $\widehat{\mathbf{d}} = \left(\hat{d}_0, \dots, \hat{d}_q\right)$ be such that $\hat{d}_j = d_j + 1$, for a given $j \in \{0, \dots, q\}$, and $\hat{d}_i = d_i$, for all $i = 0, \dots, q$, $i \neq j$. Denoted by $\{\hat{N}_i\}_{i=1}^{K+1}$ the B-spline basis of $\widehat{\mathcal{S}}$, the coefficients $\alpha_i = \gamma_i$, $i = \ell - d_j + 1, \dots, \ell$, in (4) can be computed through the following formula:*

$$1 - \alpha_i = -\alpha_{i-1} \frac{D_+^{d_j+1} \hat{N}_{i-1}|_{\bar{x}}}{D_+^{d_j+1} \hat{N}_i|_{\bar{x}}}, \tag{8}$$

where $\alpha_{\ell - d_j} = 1$ and $\bar{x}$ is any point in $[x_j, x_{j+1})$.

Proof. On $[x_j, x_{j+1})$ all the nonvanishing MDB-splines $N_i \in \mathcal{S}$ have degree d_j and hence

$$D_+^{d_j+1} N_i|_{\bar{x}} = 0, \text{ for any } \bar{x} \in [x_j, x_{j+1}).$$

Substituting equation (3) in the above identity yields

$$D_+^{d_j+1} \left(\alpha_i \hat{N}_i + (1 - \alpha_{i+1}) \hat{N}_{i+1}\right)|_{\bar{x}} = 0, \text{ for any } \bar{x} \in [x_j, x_{j+1}),$$

from which (8) directly follows. □

Remark 7. *Note that the denominator in (7) does never vanish in virtue of Remark 4, whereas the denominator in (8) does never vanish since $\hat{N}_i$ is a polynomial of degree exactly equal to $d_j + 1$ on $[x_j, x_{j+1}]$.*

From the above Propositions 6 and 7 one may see that, provided $\mathcal{S} \subset \widehat{\mathcal{S}}$, the coefficients of knot insertion, resp. degree elevation, only depend on the MDB-splines of the space having lower (local) continuity, resp. higher (local) degree. Once these coefficients are known, the MDB-splines N_i can in turn be generated by (1) or (3). Due to the locality of reverse knot insertion and degree elevation, the procedure turns out to be rather efficient, since *i)* when passing from $\widehat{\mathcal{S}}$ to $\mathcal{S}$ only a limited number of basis functions need to be updated and *ii)* the computation of each MDB-spline N_i is extremely local, involving only two MDB-splines $\hat{N}_i$.

One can hence start from an *initial* MD-spline space $\mathcal{S}_0 \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}_0}, \mathcal{X}, \mathcal{K}_0)$, such that $\mathcal{S} \subset \mathcal{S}_0$, and perform successive steps of reverse knot insertion (RKI) and/or reverse degree elevation (RDE) to generate the MDB-splines of a *target*

space $\mathcal{S} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$. Each round of reverse degree elevation is aimed at diminishing by one the degree in an interval, until each interval $[x_j, x_{j+1}]$ reaches the target degree d_j, whereas each round of reverse knot insertion is used to raise by one the continuity at a breakpoint until each break point x_j reaches the target continuity k_j. Overall, the number of steps G required to pass from $\mathcal{S}_0$ to $\mathcal{S}$ amounts to the total number of RDE and RKI steps to be performed, that is

$$G = \sum_{j=0}^{q} (d_j^0 - d_j) + \sum_{j=1}^{q} (k_j - k_j^0),$$

where d_j^0 and k_j^0 are the degrees and continuities associated with $\mathcal{S}_0$.

The process must be accomplished in such a way to generate a sequence of MD-spline spaces $\mathcal{S}_r \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}_r}, \mathcal{X}, \mathcal{K}_r)$, $r = 0, \dots, G$, such that

$$\mathcal{S} \coloneqq \mathcal{S}_G \subset \mathcal{S}_{G-1} \subset \cdots \subset \mathcal{S}_1 \subset \mathcal{S}_0,$$

where each space $\mathcal{S}_r$ is defined on $[a, b]$, has same breakpoints $\mathcal{X}$ and has dimension $K_r \coloneqq K + (G - r)$, being K the dimension of the target space $\mathcal{S}$. In general, there may be more than one sequence of nested spaces leading from $\mathcal{S}_0$ to $\mathcal{S}_G$ and therefore, while $\mathcal{S}_0$ and $\mathcal{S}_G$ are fixed, the intermediate spaces $\mathcal{S}_1, \dots, \mathcal{S}_{G-1}$ will depend on the specific ordering of RKI and RDE steps performed. In addition, there may be several ways to choose the initial space $\mathcal{S}_0$. For example, a natural criterion would be to select $\mathcal{S}_0$ so as to minimize the total number of necessary steps G – see Section 4 for a more thorough discussion on this regard.

Despite there is no other constraint on the sequence of RKI and RDE steps, to formalize the method it is convenient to establish an ordering. We will therefore alternately consider an interval and a breakpoint, starting from the leftmost interval and proceeding from left to right, applying how many RDE or RKI steps necessary, respectively, to reach the target degree or continuity. The procedure is illustrated in Algorithm 1. It shall be noted that the algorithm is just intended to schematize the fundamental idea discussed above, whereas the details of the method will be provided later on in Algorithm 2.

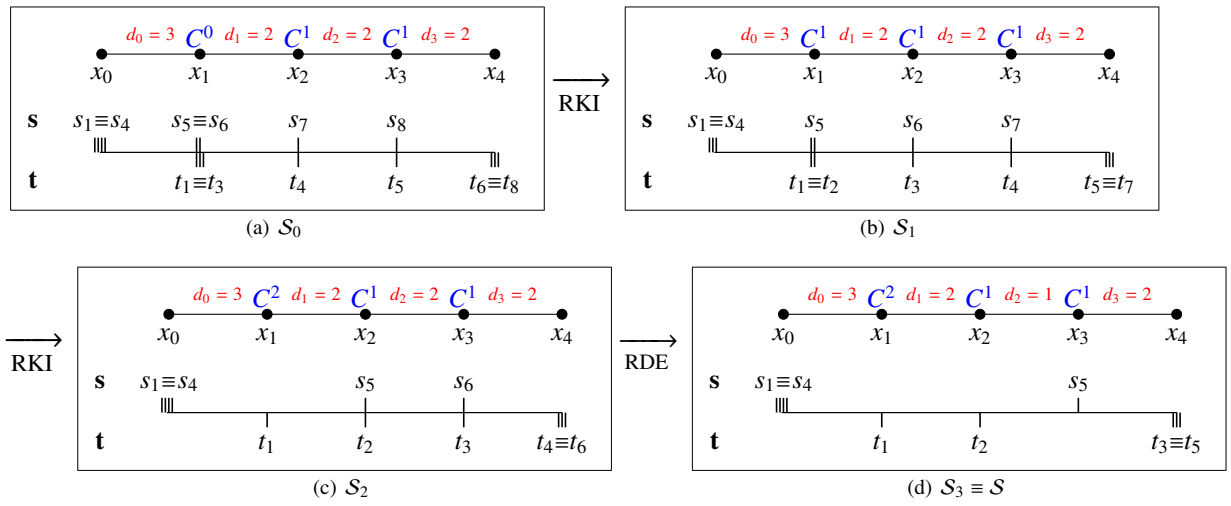
Algorithm 1: Basic outline for the computation of the MDB-spline basis

Data: The MDB-spline basis of an initial space $\mathcal{S}_0$.
Result: The MDB-spline basis of a target space $\mathcal{S}$.

$r \leftarrow 0$;
for $j \leftarrow 0$ **to** q **do**
  for $h \leftarrow d_j^0 - 1$ **to** d_j **do**
    perform RDE to generate the MDB-spline basis of $\mathcal{S}_{r+1}$ from the MDB-spline basis of $\mathcal{S}_r$ with $\mathbf{d}_{r+1} = (d_0, \dots, d_{j-1}, h, d_{j+1}^0, \dots, d_q^0)$ and $\mathcal{K}_{r+1} = (k_1, \dots, k_j, k_{j+1}^0, \dots, k_q^0)$;
    $r \leftarrow r + 1$;
  end
  if $j < q$ **then**
    for $h \leftarrow k_{j+1}^0 + 1$ **to** k_{j+1} **do**
      perform RKI to generate the MDB-spline basis of $\mathcal{S}_{r+1}$ from the MDB-spline basis of $\mathcal{S}_r$ with $\mathcal{K}_{r+1} = (k_1, \dots, k_j, h, k_{j+2}^0, \dots, k_q^0)$ and $\mathbf{d}_{r+1} = (d_0, \dots, d_j, d_{j+1}^0, \dots, d_q^0)$;
      $r \leftarrow r + 1$;
    end
  end
end

The following example illustrates the process of generating the MDB-spline basis of a given target space $\mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ by following the outline of the discussed algorithm.

Example 1. (Evaluation of the MDB-spline basis) In the interval $[0, 4]$, let us consider the MD-spline space $\mathcal{S} = \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ with $\mathcal{X} = \{1, 2, 3\}$, $\mathbf{d} = (3, 2, 1, 2)$ and $\mathcal{K} = (2, 1, 1)$. Let us also consider the spaces $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{S}_2$, defined



(a) $\mathcal{S}_0$ (b) $\mathcal{S}_1$ (c) $\mathcal{S}_2$ (d) $\mathcal{S}_3 \equiv \mathcal{S}$

Figure 1: Defining parameters for the spline spaces in Example 1 and corresponding left and right extended partitions.

on the same interval and having same breakpoint sequence as $\mathcal{S}$, such that

$$\mathcal{S} := \mathcal{S}_3 \subset \mathcal{S}_2 \subset \mathcal{S}_1 \subset \mathcal{S}_0.$$

A possible choice is to take $\mathcal{S}_0$ to be the MD-spline space featured by $\mathbf{d}_0 = (3,2,2,2)$ and $\mathcal{K}_0 = (0,1,1)$; $\mathcal{S}_1$ with $\mathbf{d}_1 = (3,2,2,2)$ and $\mathcal{K}_1 = (1,1,1)$ and $\mathcal{S}_2$ with $\mathbf{d}_2 = (3,2,2,2)$ and $\mathcal{K}_2 = (2,1,1)$. Each space $\mathcal{S}_j$, $j = 0,\dots,3$, has dimension $8-j$. The defining parameters for these spaces are represented in Figure 1. Note that $\mathcal{S}_0$ is a space of C^0 MD-splines and hence its MDB-spline basis can efficiently be computed by the generalized Cox-de Boor type recurrence relation discussed in Section 2.2. Denoted by $\{N_i^j\}$ the MDB-spline basis of $\mathcal{S}_j$, one can pass from the basis $\{N_i^0\}$ to $\{N_i^1\}$ and from $\{N_i^1\}$ to $\{N_i^2\}$ through two successive rounds of RKI and from $\mathcal{S}_2$ to $\mathcal{S}_3$ through one round of RDE.

More precisely, the RKI step to compute $\{N_i^1\}$ from $\{N_i^0\}$ reads as follows. According to equation (2), $\alpha_i^1 = 1$, for $i \leqslant 3$, and $\alpha_i^1 = 0$ for $i \geqslant 5$. Moreover, from (7),

$$(1-\alpha_4^1) = -\frac{D'_-\hat{N}_3^0|_{x_1} - D'_+\hat{N}_3^0|_{x_1}}{D'_-\hat{N}_4^0|_{x_1} - D'_+\hat{N}_4^0|_{x_1}},$$

which yields $\alpha_4^1 = \frac{2}{5}$. The basis $\{N_i^1\}$ is hence given by (1), where $N_i^1 = N_i^0$, for $i = 1,2$, $N_3^1 = N_3^0 + (1-\alpha_4^1)N_4^0$, $N_4^1 = \alpha_4^1 N_4^0 + N_5^0$ and $N_i^1 = N_{i+1}^0$, for $i = 5,6,7$.

Knowing $\{N_i^1\}$, we can now obtain $\{N_i^2\}$ by applying a round of RKI. In this case, from (2) we have $\alpha_i^2 = 1$, for $i \leqslant 2$, and $\alpha_i^2 = 0$ for $i \geqslant 6$. Moreover, from (7), we obtain

$$(1-\alpha_3^2) = -\frac{D''_-\hat{N}_2^1|_{x_1} - D''_+\hat{N}_2^1|_{x_1}}{D''_-\hat{N}_3^1|_{x_1} - D''_+\hat{N}_3^1|_{x_1}} \qquad \text{and} \qquad (1-\alpha_4^2) = -\alpha_3^2\frac{D''_-\hat{N}_3^1|_{x_1} - D''_+\hat{N}_3^1|_{x_1}}{D''_-\hat{N}_4^1|_{x_1} - D''_+\hat{N}_4^1|_{x_1}},$$

which results in $\alpha_3^2 = \frac{3}{8}$ and $\alpha_4^2 = \frac{5}{23}$. Having determined all the necessary coefficients α_i^2, we can compute the MDB-splines $\{N_i^2\}$ from (1). In particular, $N_1^2 = N_1^1$, $N_2^2 = N_2^1 + (1-\alpha_3^2)N_3^1$, $N_3^2 = \alpha_3^2 N_3^1 + (1-\alpha_4^2)N_4^1$, $N_4^2 = \alpha_4^2 N_4^1 + N_5^1$, and $N_i^1 = N_{i+1}^0$, for $i = 5,6$.

Finally, generating $\{N_i^3\}$ from $\{N_i^2\}$ requires an RDE step as follows. From (4) we have $\alpha_i^3 = 1$, for $i \leqslant 3$, and

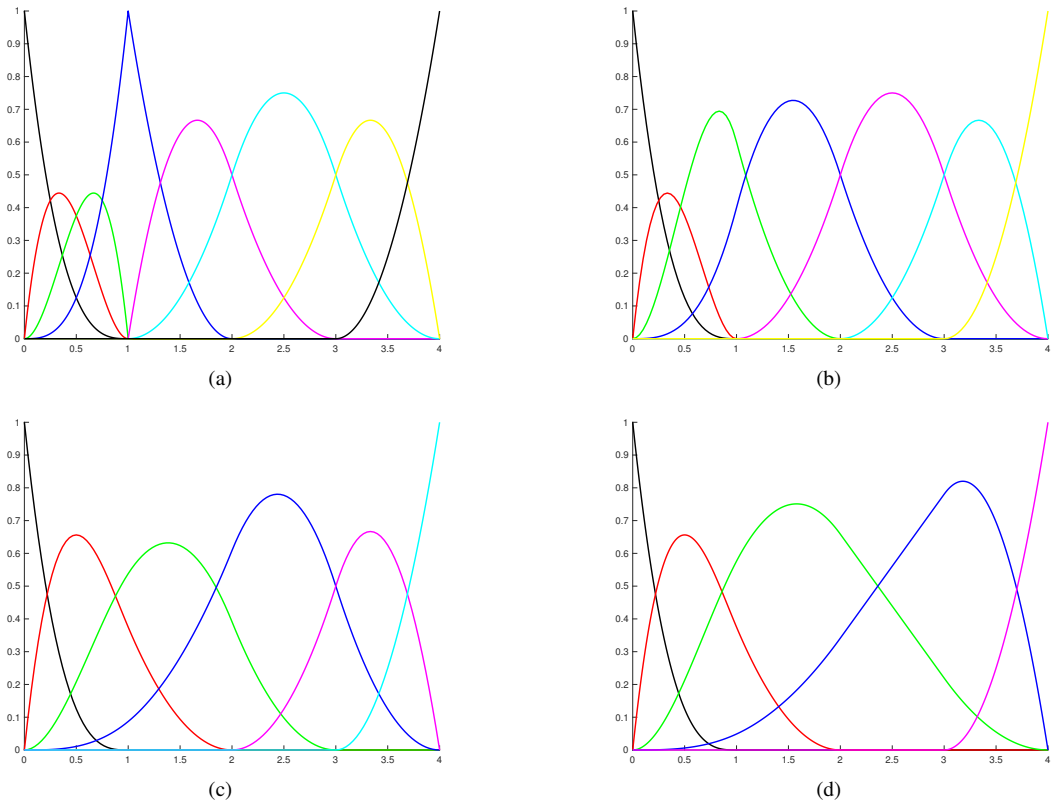


Figure 2: MDB-spline bases of the spaces in Example 1, defined on $[a, b] = [0, 4]$, with $\mathcal{X} = \{1, 2, 3\}$ and (a) $\mathbf{d} = (3, 2, 2, 2)$ and $\mathcal{K} = (0, 1, 1)$ (initial C^0 MD-spline space $\mathcal{S}_0$); (b) $\mathbf{d} = (3, 2, 2, 2)$ and $\mathcal{K} = (1, 1, 1)$ (space $\mathcal{S}_1$); (c) $\mathbf{d} = (3, 2, 2, 2)$ and $\mathcal{K} = (2, 1, 1)$ (space $\mathcal{S}_2$); (d) $\mathbf{d} = (3, 2, 1, 2)$ and $\mathcal{K} = (2, 1, 1)$ (target space $\mathcal{S} = \mathcal{S}_3$).

$\alpha_i^3 = 0$ for $i \geqslant 5$. Moreover, from (8), we obtain

$$(1 - \alpha_4^3) = -\frac{D_+'' \hat{N}_3^2|_{x_1}}{D_+'' \hat{N}_4^2|_{x_1}},$$

from where $\alpha_4^3 = \frac{23}{41}$. The computed RDE coefficients together with (3) allow us to derive the MDB-spline basis of the target space $\mathcal{S} \equiv \mathcal{S}_3$ as $N_i^3 = N_i^2$, for $i = 1, 2$, $N_3^3 = N_3^2 + (1 - \alpha_4^3)N_4^2$, $N_4^3 = \alpha_4^3 N_4^2 + N_5^2$ and $N_5^3 = N_6^2$. The basis functions of spaces $\mathcal{S}_0, \ldots, \mathcal{S}_3$ are illustrated in Figure 2.

In the reminder of the section we will show how iterated steps of RDE and RKI type lead to an efficient algorithm for evaluating the MDB-spline basis of a target space $\mathcal{S}$ and to a matrix representation relating the MDB-spline basis of $\mathcal{S}$ with that of the initial space $\mathcal{S}_0$. To this aim, we observe that a step of reverse knot insertion or reverse degree elevation between two spaces $\mathcal{S}$ and $\widehat{\mathcal{S}}$ can be represented in matrix form as

$$\mathbf{N} = \Gamma\, \hat{\mathbf{N}},$$

where $\mathbf{N} \coloneqq (N_1, \ldots, N_K)^T$ and $\widehat{\mathbf{N}} \coloneqq (\widehat{N}_1, \ldots, \widehat{N}_{\hat{K}})^T$ are the corresponding MDB-spline bases. If K and $\widehat{K} = K + 1$ are the dimension of $\mathcal{S}$ and $\widehat{\mathcal{S}}$, respectively, then Γ is a matrix of size $K \times (K + 1)$. In particular, for one step of reverse

knot insertion Γ has the following bidiagonal structure

$$\Gamma = \begin{pmatrix} 1 & 0 & & & & & & & & & & \\ & \ddots & 0 & & & & & & & & & \\ & & 1 & 1-\alpha_{\ell-d_j+1} & & & & & & & & \\ & & & \alpha_{\ell-d_j+1} & 1-\alpha_{\ell-d_j+2} & & & & & & & \\ & & & & \alpha_{\ell-d_j+2} & & & & & & & \\ & & & & & \ddots & 1-\alpha_{\ell-d_j+k_j} & & & & & \\ & & & & & & \alpha_{\ell-d_j+k_j} & 1 & & & & \\ & & & & & & & 0 & \ddots & & & \\ & & & & & & & & 0 & 1 & & \\ & & & & & & & & & & \ddots & \end{pmatrix}, \tag{9}$$

whereas for one step of reverse degree elevation Γ is a bidiagonal matrix having the following form

$$\Gamma = \begin{pmatrix} 1 & 0 & & & & & & & & & \\ & \ddots & 0 & & & & & & & & \\ & & 1 & 1-\alpha_{\ell-d_j+1} & & & & & & & \\ & & & \alpha_{\ell-d_j+1} & 1-\alpha_{\ell-d_j+2} & & & & & & \\ & & & & \alpha_{\ell-d_j+2} & & & & & & \\ & & & & & \ddots & 1-\alpha_{\ell} & & & & \\ & & & & & & \alpha_{\ell} & 1 & & & \\ & & & & & & & 0 & \ddots & & \\ & & & & & & & & 0 & 1 & \\ & & & & & & & & & & \ddots \end{pmatrix}. \tag{10}$$

Let us now denote by $\{N_i^r\}$ the MDB-spline basis of the K_r-dimensional spline space $\mathcal{S}_r$ and define the vector $\mathbf{N}_r \coloneqq \left(N_1^r, \ldots, N_{K_r}^r\right)^T$. On account of the above discussion, there exist a sequence of matrices Γ_r, $r = 1 \ldots G$, each one having size $K_r \times K_{r-1}$ and either one of the forms (9) or (10), such that

$$\mathbf{N}_1 = \Gamma_1 \mathbf{N}_0, \quad \mathbf{N}_2 = \Gamma_2 \mathbf{N}_1, \ \ldots, \ \mathbf{N}_G = \Gamma_G \mathbf{N}_{G-1}.$$

If we now indicate by I_n the identity matrix of size n and set

$$\mathrm{M}_0 = I_{K_0} \quad \text{and} \quad \mathrm{M}_r = \Gamma_r \mathrm{M}_{r-1}, \ i = 1, \ldots, G,$$

the MDB-spline bases of $\mathcal{S}$ and $\widehat{\mathcal{S}}$ turn out to be related via the matrix representation

$$\mathbf{N}_G = \mathrm{M}\,\mathbf{N}_0, \quad \text{with} \quad \mathrm{M} \coloneqq \mathrm{M}_G, \tag{11}$$

where M has size $K_G \times K_0$.

Algorithm 2, which we call the *RDE-RKI Algorithm*, contains the steps for computing the representation matrix M. The algorithm starts from the $K_0 \times K_0$ identity matrix M_0 and produces, by successive transformations of M_0, the matrix M in (11), that has size $K_G \times K_0$.

Algorithm 2: RDE-RKI Algorithm

Data: A vector $\mathbf{N}_0$, whose entries are the elements of the MDB-spline basis of $\mathcal{S}_0$.

Result: A matrix M of size $K_G \times K_0$ such that $\mathbf{N}_G = \mathrm{M}\mathbf{N}_0$.

1 $\mathrm{M}_0 \leftarrow I_{K_0}$;
2 $r \leftarrow 0$;
3 **for** $j \leftarrow 0$ **to** q **do**
4 **for** $h \leftarrow d_j^0 - 1$ **to** d_j **do**
5 compute $D_+^{h+1}\mathbf{N}_0$ at x_j;
6 $D_+^{h+1}\mathbf{N}_r \leftarrow \mathrm{M}_r\, D_+^{h+1}\mathbf{N}_0$;
7 compute α_i^{r+1} from (8), $i = \ell - h + 1, \dots, \ell$, with ℓ such that $s_\ell \leqslant x_j \leqslant \min(s_{\ell+1}, b)$;
8 $\mathrm{M}_{r+1} \leftarrow \Gamma_{r+1}\mathrm{M}_r$;
9 $r \leftarrow r + 1$;
10 **end**
11 **if** $j < q$ **then**
12 **for** $h \leftarrow k_{j+1}^0 + 1$ **to** k_{j+1} **do**
13 compute $D_-^h\mathbf{N}_0$ and $D_+^h\mathbf{N}_0$ at x_{j+1};
14 $D_-^h\mathbf{N}_r - D_+^h\mathbf{N}_r \leftarrow \mathrm{M}_r\,(D_-^h\mathbf{N}_0 - D_+^h\mathbf{N}_0)$;
15 compute α_i^{r+1} from (7), $i = \ell - d_{j+1} + 1, \dots, \ell - d_{j+1} + h$, with ℓ such that $s_\ell \leqslant x_j \leqslant \min(s_{\ell+1}, b)$;
16 $\mathrm{M}_{r+1} \leftarrow \Gamma_{r+1}\mathrm{M}_r$;
17 $r \leftarrow r + 1$;
18 **end**
19 **end**
20 **end**
21 $\mathrm{M} \leftarrow \mathrm{M}_r$.

Remark 8 (Computational complexity of the RDE-RKI Algorithm)**.** *The total number of coefficients α_i^{r+1} to be determined by the entire algorithm, that is iterating over all the intervals (for RDE steps) and all breakpoints (for RKI steps), is*

$$Comp_{RDE\text{-}RKI} = \frac{1}{2}\left(\sum_{j=0}^{q} d_j^0(d_j^0 - 1) - d_j(d_j - 1)\right) + \frac{1}{2}\left(\sum_{j=1}^{q} k_j(k_j + 1) - k_j^0(k_j^0 + 1)\right). \tag{12}$$

The above formula takes into account that no computation is required in case an interval has target degree d_j s.t. $d_j = d_j^0$, or a breakpoint has target continuity k_j s.t. $k_j = k_j^0$. Note that $Comp_{RDE\text{-}RKI}$ depends on the choice of the initial space $\mathcal{S}_0$ and may thus vary accordingly.

We will use $Comp_{RDE\text{-}RKI}$ as an index of computational complexity, since the number of operations necessary for the computation of the coefficients α_i^{r+1} and the subsequent updating of matrices M_r can be bounded above by $Comp_{RDE\text{-}RKI}$ times a suitable constant, as will be explained by the following two remarks.

Remark 9 (Cost of the computation of each RDE or RKI coefficient)**.** *In our implementation of Algorithm 2, the derivatives of the C^0 MDB-splines in $\mathcal{S}_0$ at Lines 5 and 13 are computed in preprocessing at each break point x_j up to the necessary order. The computational cost of each coefficient α_i^{r+1} in an RDE or RKI step is thus given by number of floating point operations for the computation of the derivatives at Line 6, resp. 14, which involves the product of the $(i-1)$th row of M_r and the precalculated derivatives in $\mathcal{S}_0$, plus the number of elementary operations in (8), resp. (7) (Lines 7 and 15).*

Remark 10 (Computational cost of a matrix update)**.** *The products $\Gamma_{r+1}M_r$ at Lines 8 and 16 of Algorithm 2 can efficiently be computed exploiting the structure and sparsity of the involved matrices. In particular any such product can be obtained by replacing each row of index i of M_r with a combination of the two rows of indices i and $i + 1$,*

namely

$$\textit{ith row of } M_{r+1} \quad \longleftarrow \quad \alpha_i^{r+1}\ (\textit{ith row of } M_r) + (1-\alpha_{i+1}^{r+1})\ ((i+1)\textit{th row of} M_r)\,, \tag{13}$$

where α_i^{r+1} are the nontrivial elements of the bidiagonal matrix Γ_{r+1} and correspond to rows $i = \ell - h, \dots, \ell$ for the product at Line 8, resp. $i = \ell - d_{j+1}, \dots, \ell - d_{j+1} + h$ for the product at Line 16. In other words, for each pair α_i^{r+1} and α_{i+1}^{r+1}, such that α_{i+1}^{r+1} has been calculated on the previous line of the algorithm, it is necessary to perform the update (13), the computational cost of which is the combination of two rows of M_r. The cost of (13) can be further optimized taking into account that each M_r is a band matrix, in that it represents MDB-spline functions in terms of (compactly supported) C^0 MDB-splines, and so most of its row entries will be zero.

In our implementation of Algorithm 2 matrices M_r are stored in a single matrix, which is overwritten at each iteration, so no other information needs to be stored while the algorithm is being executed. In particular, the algorithm starts from the identity matrix M of size $K_0 \times K_0$ and updates it until obtaining the final matrix M of size $K_G \times K_0$ (where K_0 is the dimension of the initial space $\mathcal{S}_0$ and $K_G < K_0$ is the dimension of the target space $\mathcal{S} = \mathcal{S}_G$).

Continuing Example 1, we illustrate in the following the successive matrices M_r generated by the RDE-RKI Algorithm and the resulting matrix representation relating the MDB-spline basis of the target space $\mathcal{S}$ with that of the C^0 MD-spline space $\mathcal{S}_0$ considered in the example.

Example 1 (Continued)**.** In the considered setting, the starting data are $\mathrm{M}_0 = I_8$ and a vector $\mathbf{N}_0 = (N_1^0, \dots, N_8^0)^T$ containing the MDB-spline basis of the C^0 MD-spline space $\mathcal{S}_0$. The first step of the algorithm, which is of RKI type, leads to a matrix M_1 of size 7×8

$$\mathrm{M}_1 = \Gamma_1 \mathrm{M}_0 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 3/5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2/5 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

such that $(N_1^1, \dots, N_7^1)^T = \mathrm{M}_1\ (N_1^0, \dots, N_8^0)^T$. The second RKI-type step produces a matrix M_2 of size 6×8

$$\mathrm{M}_2 = \Gamma_2 \mathrm{M}_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 5/8 & 3/8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3/8 & 99/184 & 18/23 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2/23 & 5/23 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

such that $(N_1^2, \dots, N_6^2)^T = \mathrm{M}_2\ (N_1^0, \dots, N_8^0)^T$. Finally, the third and last step, which is of RDE type, yields the following matrix $\mathrm{M} \coloneqq \mathrm{M}_3$ of size 5×8

$$\mathrm{M} \coloneqq \mathrm{M}_3 = \Gamma_3 \mathrm{M}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 5/8 & 3/8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3/8 & 189/328 & 36/41 & 18/41 & 0 & 0 \\ 0 & 0 & 0 & 2/41 & 5/41 & 23/41 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Matrix M represents the MDB-splines of $\mathcal{S} \equiv \mathcal{S}_3$ in terms of the MDB-splines of $\mathcal{S}_0$ in such a way that

$$(N_1^3, \dots, N_5^3)^T = \mathrm{M}\ (N_1^0, \dots, N_8^0)^T.$$

The RDE-RKI Algorithm enables us to compute the MDB-splines of a target space $\mathcal{S}$ from those of any space $\mathcal{S}_0$ such that $\mathcal{S} \subset \mathcal{S}_0$. The choice of $\mathcal{S}_0$ may thus be driven by specific application requirements, simple computational

efficiency considerations, or potentially any other criterion. In Section 3.4, we will further delve on this topic and we will see that a generally good choice is to trigger the algorithm form a space of C^0 MD-splines, whose basis can efficiently evaluated by the generalized Cox-de Boor recurrence relation in Section 2.2. Alternatively, we may compute the MDB-spline basis starting from the Bernstein bases relative to each interval, or from a "global" conventional B-spline basis of maximum degree $m := \max_j\{d_j\}_{j=0}^q$. We can as well choose the initial basis in such a way that the representation matrix M coincides with the H-operator in [7]. The following Sections 3.1, 3.2 and 3.3 are devoted to analyzing these special instances of the proposed algorithm.

3.1. Matrix representation in terms of Bernstein bases

In this section, we seek a matrix representation in terms of the Bernstein bases of degree d_j relative to the break-point intervals $[x_j, x_{j+1}]$, $j = 0, \ldots, q$. For any given target space $\mathcal{S} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$, such a representation is obtained by taking an initial space $\mathcal{S}_0$ with continuity vector $\mathcal{K}_0 = (0, \ldots, 0)$ and same degree vector as $\mathcal{S}$ and by applying RKI-type steps to generate the MDB-spline basis of $\mathcal{S}$. Each round of reverse knot insertion raises by one the continuity at a break point until each break point x_j reaches the target continuity k_j, in such a way that the total number of steps necessary to pass from $\mathcal{S}_0$ to $\mathcal{S}$ is $G = \sum_{j=1}^q k_j$ and $\mathcal{S} \equiv \mathcal{S}_G$. Denoted by $\{B_{r,d_j}\}$, $r = 0, \ldots, d_j$ the Bernstein basis relative to $[x_j, x_{j+1}]$ and set $\mathbf{B} = (B_{0,d_0}, \ldots, B_{d_0,d_0}, B_{0,d_1}, \ldots, B_{d_q,d_q})^T$, we shall hence take $\mathbf{N}_0$ in (11) to be

$$\mathbf{N}_0 = A\mathbf{B}, \text{ with } A = \begin{pmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{bmatrix} & & & \\ & \begin{bmatrix} 1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{bmatrix} & & \\ & & \ddots & \\ & & & \begin{bmatrix} 1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{pmatrix}, \tag{14}$$

where the matrix blocks have sizes $(d_0 + 1) \times (d_0 + 1)$, $(d_1 + 1) \times (d_1 + 1)$, …, $(d_q + 1) \times (d_q + 1)$. This yields the following matrix representation in terms of Bernstein bases:

$$\mathbf{N}_G = \overline{\mathrm{M}}\,\mathbf{B}, \quad \text{with} \quad \overline{\mathrm{M}} := \mathrm{M}\,A.$$

In this particular situation, Algorithm 2 is reduced to a simplified version, which we refer to as the *RKI Algorithm*, consisting of rows 11–19 only and where the continuity vector of $\mathcal{S}_0$ has entries $k_{j+1}^0 = 0$, for all $j = 0, \ldots, q-1$.

The computational complexity of the RKI Algorithm, measured as the total number of coefficients α_i^{r+1} to be calculated is

$$\mathrm{Comp}_{\mathrm{RKI}} = \frac{1}{2}\sum_{j=1}^{q} k_j\,(k_j + 1), \tag{15}$$

which corresponds to the second summand at the right-hand side of (12).

3.2. Matrix representation in terms of the conventional B-spline basis of maximum degree

We now address the special case in which we want to represent the MDB-spline basis in matrix form, via multiplication by the B-spline basis of a conventional spline space where all sections have the same degree $m := \max_j\{d_j\}_{j=0}^q$.

In this situation, for a given target space $\mathcal{S} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$, we shall choose an initial space $\mathcal{S}_0$ having continuity vector $\mathcal{K}_0 = \mathcal{K}$ and a constant degree vector $\mathbf{d}_0 = (m, \ldots, m)$. We will hence generate the MDB-spline basis of $\mathcal{S}$ by performing successive reverse degree elevation steps only. Each round of degree elevation diminishes by one the degree on an interval, until each interval $[x_j, x_{j+1}]$ reaches the target degree d_j, in such a way that the total number of steps G necessary to pass from $\mathcal{S}_0$ to $\mathcal{S}$ is $G = \sum_{j=0}^q (m - d_j)$ and $\mathcal{S} \equiv \mathcal{S}_G$. In this case, the entries of $\mathbf{N}_0$ in (11)

are the elements of the conventional B-spline basis of degree m and Algorithm 2 consists in processing each interval $[x_j, x_{j+1}]$, $j = 0, \dots, q$, and perform $m - d_j$ RDE steps to diminish the degree up to the target one. This leads to a simplified version of the algorithm, which we refer to as the *RDE Algorithm*, consisting of lines 4–10 only and where the degree vector associated with $\mathcal{S}_0$ has entries $d_j^0 = m$, for all $j = 0, \dots, q$. The computational complexity of the RDE Algorithm, intended as the total number of computed coefficients α_i^{r+1}, is

$$\mathrm{Comp}_{\mathrm{RDE}} = \frac{1}{2}\left((q+1)\,m\,(m-1) - \sum_{j=0}^{q} d_j\,(d_j-1)\right), \tag{16}$$

which corresponds to the first summand at the right-hand side of (12).

3.3. Matrix representation in terms of local conventional B-spline bases

We are interested here in the special setting of the RDE-RKI Algorithm enabling us to reproduce the H-operator [7]. This occurs when one starts from an MDB-spline basis which is a conventional B-spline basis piecewisely. More precisely, for a given target space $\mathcal{S} \equiv \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$, we shall define the vector $J = (j_1, \dots, j_p)$, $j_r < j_{r+1}$, whose entries are the indices $j \in \{0, \dots, q\}$ such that $d_{j-1} \neq d_j$. The restriction of $\mathcal{S}$ to $[x_0, x_{j_1}]$ is a conventional spline space of degree d_0 and we denote its B-spline basis $\{N_{r,d_0}\}$. Analogously, for $i = 1, \dots, p-1$, the restriction of $\mathcal{S}$ to $[x_{j_i}, x_{j_{i+1}}]$ is a conventional spline space of degree d_{j_i} with B-spline basis $\{N_{r,d_{j_i}}\}$ and finally the restriction to $[x_{j_p}, x_{q+1}]$ is a degree-d_q spline space with B-spline basis $\{N_{r,d_q}\}$. Altogether these basis functions describe a piecewise conventional spline space, with C^{-1} continuity at each knot x_{j_i}.

Taking the vector $\mathbf{N}_{-1} = (N_{1,d_0}, N_{2,d_0}, \dots, N_{1,d_{j_1}}, N_{2,d_{j_1}}, \dots, N_{1,d_q}, N_{2,d_q} \dots)^T$ one may obtain a basis $\mathbf{N}_0$ of a C^0 MD-spline space by the transformation

$$\mathbf{N}_0 = A\mathbf{N}_{-1},$$

where matrix A has $p+1$ blocks and a similar structure as in (14), the only difference being that the size of each block is now equal to the number of corresponding conventional B-spline functions. In particular $\mathbf{N}_0$ is the basis of an MD-spline space having degree vector $\mathbf{d}_0 = \mathbf{d}$ and continuity vector $\mathcal{K}_0$ such that, for $i = 0, \dots, q$,

$$k_i^0 = \begin{cases} 0 & \text{if } i \text{ is an element of } J, \\ k_i & \text{otherwise,} \end{cases}$$

which we will take to be the initial space $\mathcal{S}_0$. The MDB–spline basis of $\mathcal{S}$ is hence generated through the RKI Algorithm, where each round of reverse knot-insertion raises by one the continuity at a breakpoint x_j up to the target continuity k_j. The total number of steps necessary to pass from $\mathcal{S}_0$ to $\mathcal{S} \equiv \mathcal{S}_G$ is thus $G = \sum_{j \in J} k_j$. This results in the following matrix representation in terms of local conventional B-spline bases:

$$\mathbf{N}_G = \overline{\mathrm{M}}\,\mathbf{N}_{-1}, \quad \text{with} \quad \overline{\mathrm{M}} \coloneqq \mathrm{M}\,A,$$

where $\overline{\mathrm{M}}$ coincides with the H-operator in [7].

3.4. Which initial space?

Given the possibility of selecting $\mathcal{S}_0$ within a vast family of spaces, it is natural to examine whether there can be an "optimal" choice or at least a default good one. As usual, specific application requirements, if existing, represent the most important criterion of choice. For example, IGA practitioners will probably privilege a representation through local Bernstein bases, like the one we can get from the RKI Algorithm described in Section 3.1. For geometric modeling purposes, the customary B-spline basis representation will make us more inclined to define $\mathcal{S}_0$ as in Sections 3.2 and 3.3. If instead no application-related requirement is dictated, it is reasonable to aim at containing as much as possible the size of the representation matrix M and the overall computational cost of the algorithm. Recalling that M has size $K_G \times K_0$, we will therefore be naturally oriented towards choosing the space of minimum dimension among those containing $\mathcal{S}$. In most cases, this will also entail the least computational complexity, estimated as the total number of coefficients α_i^{k+1} to be calculated (see equations (12), (15), (16)). As a consequence, we may conclude that

minimizing the dimension of $\mathcal{S}_0$ provides a good default criterion, which guarantees the efficiency of the algorithm in general circumstances.

It is interesting to notice, however, that there are special cases where the space of smallest dimension does not correspond to the least computational complexity, nor the best performance of the algorithm. This will be illustrated by a counterexample later on (see Table 4 and related discussion in Section 4), showing that these aspects can be improved by an ad-hoc choice of $\mathcal{S}_0$.

We conclude the discussion by providing an illustrative example of how many and which initial spaces can be identified in correspondence of an assigned target space. In the interval $[0, 3]$, consider the MD-spline space $\mathcal{S} = \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ with $\mathcal{X} = \{1, 2\}$, $\mathbf{d} = (4, 3, 5)$ and $\mathcal{K} = (3, 1)$, having dimension $K = 9$. In this setting:

- If we seek a representation in terms of local Bernstein bases as in Section 3.1, then $\mathcal{S}_0$ should have same degree vector as $\mathcal{S}$, continuities $\mathcal{K} = (0, 0)$ and thus $K_0 = 13$.
- If instead we aim at a representation in terms of the conventional B-spline basis of maximum degree, as in Section 3.2, then $\mathcal{S}_0$ and $\mathcal{S}$ should have the same vector of continuities, but $\mathcal{S}_0$ will be a conventional spline space of degree 5 and dimension $K_0 = 12$.
- A further analysis, shows that there are candidate spaces $\mathcal{S}_0$ of even lower dimension, such as the C^0 MD-spline space having degrees $\mathbf{d}_0 = (4, 4, 5)$ and continuities $\mathcal{K}_0 = (3, 0)$, having dimension $K_0 = 11$ and resulting in $\mathrm{Comp}_{\text{RDE-RKI}} = 4$.
- Though the above are effective and reasonable choices, there are indeed many more possibilities. We might as well take, for example, $\mathbf{d}_0 = (4, 5, 5)$ and $\mathcal{K}_0 = (0, 1)$. Clearly this space has worst performance than the previous one, since $K_0 = 14$ and $\mathrm{Comp}_{\text{RDE-RKI}} = 13$.
- Modifying slightly our example, we may now consider the MD-spline space $\mathcal{S} = \mathcal{S}(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ on $[0, 3]$, having $\mathcal{X} = \{1, 2\}$, $\mathbf{d} = (4, 3, 5)$, $\mathcal{K} = (3, 2)$ and dimension $K = 8$. There are now two spaces of minimum dimension, corresponding to $\mathbf{d}_0 = (4, 4, 5)$, $\mathcal{K}_0 = (3, 0)$ and $\mathbf{d}_0 = (5, 5, 5)$, $\mathcal{K}_0 = (3, 2)$, for both of which $K_0 = 11$. In this circumstance, an analysis of the computational complexity will orient our choice towards the former space, for which (12) yields $\mathrm{Comp}_{\text{RDE-RKI}} = 6$ against 11 for the latter space.

Summarizing, we have identified different spaces $\mathcal{S}_0$ to choose from, based on the sought type of representation, the size of matrix M and the computational complexity of the algorithm. We have observed that, in general, starting from a C^0 MD-spline space allows us to contain the number of steps of the algorithm (fewer matrices M_r to be calculated) and the size of M, and that this criterion of choice can effectively be complemented by an analysis of computational complexity.

4. Numerical considerations

The RDE-RKI Algorithm yields a representation matrix M whose entries are nonnegative, smaller than or equal to one, and where the sum of each column is equal to one. Therefore the change of basis $\mathbf{N} = \mathrm{M}\mathbf{N}_0$ is well-suited for numerical computations. However the calculation of the entries of M according to (7)–(8) involves a few passages that may raise some concern from a numerical point of view, such as the evaluation of the derivatives of MDB-spline functions in $\mathcal{S}_0$ and a few arithmetic operations that may be subject to numerical cancellation errors. This makes it worthy of investigating the accuracy achieved in the computation of the elements of M.

In the absence of round-off error on the data, a common technique for estimating the accuracy of a numerical method consists in recomputing at a higher precision and test how many digits of the original and more accurate answer agree (see e.g. [23]). For this purpose, we implemented the RDE-RKI Algorithm and its variants in double and quadruple precision (16 and 32 decimal digits of precision, respectively) using MATLAB variable-precision arithmetic capability. We hence performed a thorough experimentation to verify that, in extreme circumstances, the aforementioned criticality do not destructively affect the computation.

In the following sections, we analyze passages which deserve special attention, discussing how to ensure the accuracy of computations, and we present an illustrative sample of our numerical experiments. This analysis will confirm that the RDE-RKI Algorithm yields accurate results for target spaces of practical interest and even beyond.

4.1. On the computation of the MDB-spline basis functions and of their derivatives

Akin to previous methods [6, 9], the RDE-RKI Algorithm relies on the derivatives of basis functions in the initial space $\mathcal{S}_0$. When such a basis is formed piecewisely by Bernstein or B-spline type bases, the derivatives can be computed by classical recurrence schemes [21], which are known to be stable for any order of differentiation [24]. As discussed in Section 2.2, these recurrence schemes can be adapted for the case where $\mathcal{S}_0$ is a space of C^0 MD-splines, in such a way that the same stability properties can be expected. This expectation has been experimentally confirmed by comparing results obtained in double and quadruple precision for several initial data and derivatives of any order, leading us to conclude that the computed derivatives of a C^0 MDB-spline basis are accurate whatever the initial space $\mathcal{S}_0$.

In spite of this, the derivatives of basis functions may become large for high degrees and/or orders of differentiation and may have different signs at a breakpoint. For example, at the breakpoints of a uniform partition, B-splines of degree 20 have 16th derivatives of magnitude $\approx 10^{21}$. Thus to generate a matrix M whose entries are non-negative and less than or equal to 1, we may have to use much larger intermediate values. This contradicts well-known good practice, which, in order to prevent accuracy problems, would require, when possible, to avoid working with intermediate values whose magnitude is significantly higher than the final solution (see e.g. [23], pp. 27).

The above discussion prompts us to use additional care in order to minimize potential loss of significant digits when dealing with multi-degree spline spaces of locally high degree – namely when derivatives of high order are involved. In this respect, numerical instability issues may be overcome resorting to multi-precision software libraries, paying the price for a reduction in the calculation speed, or to the more efficient EFT (Error-Free Transformation) approaches, also called *compensated algorithms* (see e.g. [25, 26] and the monograph [23]). A compensated algorithm is one where only a subset of operations, those potentially subject to loss of information, are performed with higher precision and without affecting computational efficiency. In our case, we have realized a compensated version of the RDE-RKI Algorithm in which dangerous calculations are performed with 32 digits of precision, whereas 16 digits of precision are used elsewhere. This strategy has produced highly accurate results in all our numerical experiments, a sample of which will be discussed in Section 4.3.

4.2. Analysis of the RDE-RKI Algorithm and identification of steps that need to be compensated

To identify potential sources of errors and passages that need to be implemented in a compensated manner, we shall analyze step by step the operations performed by the RDE-RKI Algorithm. We refer below to Algorithm 2, but for obvious reasons the same applies to its variants in Sections 3.1, 3.2, 3.3.

i) *Evaluation of derivatives*. Derivatives of the basis functions in $\mathcal{S}_0$ must be calculated at each breakpoint up to a specific order (Lines 5 and 13 in Algorithm 2). More precisely, an RDE-type step (Line 5), for an interval with target degree d_j, requires computing the derivatives of basis functions of degree d_j^0 for all orders of differentiation from d_j to d_j^0; An RKI-type step (Line 13), at a breakpoint x_j, entails evaluating derivatives of basis functions of degrees d_{j-1}^0 and d_j^0 up to the target order k_j. In virtue of the previous discussion (see Section 4.1), the derivatives $D^h\mathbf{N}_0$ involved in RDE-type steps and $D_+^h\mathbf{N}_0$ and $D_-^h\mathbf{N}_0$ involved in RKI-type steps are to be deemed accurate.

ii) *Matrix-vector products*. As soon as $h \geqslant 1$, successive entries of the vectors $D_+^h\mathbf{N}_0$ and $D_-^h\mathbf{N}_0 - D_+^h\mathbf{N}_0$ may have different signs and large values. As a consequence, the matrix-vector products $\mathrm{M}_r D_+^h\mathbf{N}_0$ and $\mathrm{M}_r(D_-^h\mathbf{N}_0 - D_+^h\mathbf{N}_0)$ (Lines 6 and 14) are potentially subject to numerical cancellation and should be handled with care.

iii) *Calculation of the coefficients α_i*. The calculation of the coefficients α_i (Lines 7 and 15) according to (7)–(8) may be subject to numerical errors in case higher order derivatives are large.

iv) *Update of matrix M_r*. The computation of M_{r+1} (Lines 8 and 16) from M_r involves linear combinations of positive entries of M_r with coefficients $\alpha_i \in [0, 1]$ and thus is numerically safe.

On account of this analysis, in the compensated version of the RDE-RKI Algorithm only the delicate operations at the above items ii) and iii) are performed using 32 decimal digits. More precisely, at each iteration, though derivatives are computed with 16 digits of precision, the matrix-vector products at ii) are then performed and stored with 32 digits; the same precision is also used to calculate the coefficients α_i at iii), which are finally converted to 16 digits of precision prior to the matrix update in iv).

4.3. Numerical experiments

The error on the result was quantified as

$$\mathrm{Err} = \|\mathrm{M}_{16\,\mathrm{digits}} - \mathrm{M}_{32\,\mathrm{digits}}\|_1,$$

where $\mathrm{M}_{16\,\mathrm{digits}}$ and $\mathrm{M}_{32\,\mathrm{digits}}$ are the matrix representations of the target basis $\mathbf{N}_G$ in the initial basis $\mathbf{N}_0$ obtained with 16 and 32 decimal digits of precision. Measuring the error in 1-norm appears as a natural choice in that: *i)* regardless of the initial space $\mathcal{S}_0$, matrix M has the same number of elements in each column, since the size of M is $K_G \times K_0$ and *ii)* matrix M has column-sum equal to one, therefore producing the same relative and absolute error in 1-norm.

Our experimentation showed that in practical situations, that is for reasonable degrees and moderately non-uniform breakpoint sequences, the RDE-RKI Algorithm and its variants (merely RKI or merely RDE) always yield accurate results, with errors laying within working precision. The examples presented here focus instead on more challenging circumstances, featuring very high degrees and highly non-uniform partitions.

For a target space defined on two breakpoint intervals, the evaluation procedure will reduce to either one of the simplified versions in Sections 3.1 and 3.2, which consist in performing either RKI or RDE steps. Tables 1 and 2 present a comparison of the RKI and RDE Algorithms and their compensated versions in this circumstance. At first glance, for algorithms that do not use compensated operations, the error (column Err) seems to grow coherently with the increase in computational complexity (reported in column $\mathrm{Comp}_{\text{RDE-RKI}}$ and quantified as the total number of coefficients α_i to be determined). A more in-depth analysis, however, shows that in some cases the loss of accuracy increases at a higher rate that the number of performed arithmetic operations. This undesired behavior is corrected by the compensated steps (CS) of the algorithms (see column Err CS).

A similar experiment is presented in Table 3 for a target space with several breakpoint intervals. Here numerical results also suggest that the purely RDE Algorithm may be more prone to loss of accuracy, as one can see from the central row in the table. This can be explained by the fact that the RDE Algorithm generally works with higher order derivatives that, though accurate, have larger values. On the other hand, the RDE Algorithm can have much better performance than the RKI Algorithm for certain target spaces (see e.g. the last two rows), which suggests that, in more general situations, an adequate combination of RDE and RKI steps should be the best option.

Our last example further reinforces this view. In Section 3.4 good choices for the initial space $\mathcal{S}_0$ were identified to be the C^0 MD-spline space of minimum dimension among those containing $\mathcal{S}_G$, or the space yielding the lowest computational complexity. For an assigned target space $\mathcal{S}_G$, Table 4 presents an overview of the performance of the algorithm triggered from the more efficient initial spaces, along with an analysis on the dimension of $\mathcal{S}_0$ (or, equivalently, the size of M, that is $K_G \times K_0$) and the computational complexity. When using the non compensated algorithm, the best overall performance is not achieved by the space $\mathcal{S}_0$ of minimum dimension, nor by that implying the least number of RDE steps. It is instead obtained by the synergetic use of RDE and RKI steps corresponding to the space $\mathcal{S}_0$ of lowest computational complexity. Out of curiosity, we remark that in this example there exist as many as 16 different admissible spaces $\mathcal{S}_0$, whose dimension ranges from 20 to 33.

In view of the above discussion and of the obtained numerical results, we shall conclude that triggering the algorithm from the space $\mathcal{S}_0$ of minimum dimension significantly reduces the number of steps and, when coupled with the compensation strategy described above, produces extremely accurate results even for very high degrees (up to 30). At the same time, one should take into account how many steps of RKI and RDE type an initial space $\mathcal{S}_0$ will involve, avoiding spaces that require a very large number of RDE steps.

4.4. Further considerations

We gather below some additional considerations of various nature arising from our analysis of the algorithm.

- The MDB-splines $\mathbf{N}_G$ are evaluated by the matrix product $\mathrm{M}\mathbf{N}_0$, where M and $\mathbf{N}_0$ have nonnegative entries in $[0, 1]$ and $\mathbf{N}_0$ is computed within machine precision accuracy according to our numerical assessment (see Section 4.1). Therefore, it is reasonable to expect that the error in the evaluation of $\mathbf{N}_G$ be in line with that on M. This expectation was confirmed by our numerical tests.

- The matrix representation simultaneously yields the derivatives $D_+^h\mathbf{N}_G$ of the MDB-spline basis in the target space. In particular, $D_+^h\mathbf{N}_G = \mathrm{M}D_+^h\mathbf{N}_0$, where, as observed earlier, the entries of $D_+\mathbf{N}_0$ may have different signs

d	K	K_0		$\text{Comp}_{\text{RDE-RKI}}$		Err		Err CS	
		RKI	RDE	RKI	RDE	RKI	RDE	RKI	RDE
(10,5)	11	16	16	15	35	5.4×10^{-15}	1.3×10^{-14}	1.1×10^{-16}	1.1×10^{-16}
(10,7)	13	18	16	15	24	2.2×10^{-15}	2.7×10^{-15}	1.4×10^{-16}	8.2×10^{-17}
(10,9)	15	20	16	15	9	7.6×10^{-16}	2.9×10^{-39}	5.0×10^{-17}	2.9×10^{-39}
(10,11)	17	22	18	15	10	2.0×10^{-16}	7.1×10^{-14}	8.1×10^{-17}	2.5×10^{-37}
(10,13)	19	24	22	15	33	2.2×10^{-15}	2.1×10^{-12}	1.5×10^{-16}	8.5×10^{-17}
(10,15)	21	26	26	15	60	1.1×10^{-15}	9.2×10^{-12}	9.3×10^{-17}	1.7×10^{-16}
(10,17)	23	28	30	15	91	6.9×10^{-16}	5.2×10^{-10}	1.2×10^{-16}	2.0×10^{-16}
(10,19)	25	30	34	15	126	4.5×10^{-16}	3.9×10^{-8}	1.3×10^{-16}	1.2×10^{-16}

Table 1: Target space $S_G(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ with $[a, b] = [0, 2]$, $\mathcal{X} = (1)$ and $\mathcal{K} = (5)$.

k_1	K	K_0		$\text{Comp}_{\text{RDE-RKI}}$		Err		Err CS	
		RKI	RDE	RKI	RDE	RKI	RDE	RKI	RDE
5	17	40	36	15	19	2.0×10^{-16}	3.3×10^{-11}	1.0×10^{-16}	2.8×10^{-17}
7	19	40	34	28	19	6.7×10^{-15}	1.4×10^{-10}	1.4×10^{-16}	5.6×10^{-17}
9	21	40	32	45	19	4.6×10^{-14}	1.4×10^{-11}	1.7×10^{-16}	5.0×10^{-17}
11	23	40	30	66	19	1.7×10^{-13}	1.0×10^{-09}	2.2×10^{-16}	5.6×10^{-17}
13	25	40	28	91	19	1.4×10^{-12}	1.8×10^{-09}	2.1×10^{-16}	5.6×10^{-17}
15	27	40	26	120	19	2.4×10^{-11}	2.3×10^{-09}	2.5×10^{-16}	5.6×10^{-17}
17	29	40	24	153	19	2.2×10^{-10}	1.5×10^{-10}	8.7×10^{-12}	5.6×10^{-17}
19	31	40	22	190	19	1.3×10^{-9}	2.1×10^{-11}	5.6×10^{-11}	5.6×10^{-17}

Table 2: Target space $S_G(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ with $[a, b] = [0, 2]$, $\mathcal{X} = (1)$ and $\mathbf{d} = (19, 20)$.

h	K	K_0		$\text{Comp}_{\text{RDE-RKI}}$		Err	
		RKI	RDE	RKI	RDE	RKI	RDE
3	20	26	25	12	25	3.1×10^{-39}	1.1×10^{-13}
4	19	27	23	20	22	4.7×10^{-15}	1.0×10^{-13}
5	18	28	21	30	18	4.1×10^{-15}	1.4×10^{-13}
6	17	29	19	42	13	5.9×10^{-14}	1.3×10^{-15}
7	16	30	17	56	7	9.3×10^{-13}	1.1×10^{-39}

Table 3: Target space $S_G(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$, with $[a, b] = [0, 9]$, $\mathcal{X} = (1, 2, 3, 4, 5, 6, 7, 8)$, $\mathbf{d} = (8, 8, 8, 8, h, 8, 8, 8, 8)$, $\mathcal{K} = (7, 7, 7, h, h, 7, 7, 7)$. The initial space S_0 for the RKI algorithm has $\mathcal{K}_0 = (7, 7, 7, 0, 0, 7, 7, 7)$. The initial space S_0 for the RDE algorithm has $\mathbf{d}_0 = (8, 8, 8, 8, 8, 8, 8, 8, 8)$.

$S_0(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$	K_0	Comp_{RKI}+Comp_{RDE}	Err	Err CS
$\mathbf{d}$ =(7,7,7,7,7) $\mathcal{K}$ =(3,6,2,4)	21	39+0	2.6×10^{-14}	5.7×10^{-17}
$\mathbf{d}$ =(7,7,7,5,5) $\mathcal{K}$ =(3,6,0,4)	19	17+3	2.6×10^{-14}	5.7×10^{-17}
$\mathbf{d}$ =(5,6,7,5,5) $\mathcal{K}$ =(0,0,0,4)	25	0+30	5.6×10^{-16}	1.5×10^{-16}
$\mathbf{d}$ =(5,7,7,5,5) $\mathcal{K}$ =(0,6,0,4)	20	6+9	8.6×10^{-17}	7.8×10^{-17}

Table 4: Target space $S_G(\mathcal{P}_{\mathbf{d}}, \mathcal{X}, \mathcal{K})$ with $[a, b] = [0, 5]$, $\mathcal{X} = (1, 2, 3, 4)$, $\mathbf{d} = (5, 6, 7, 5, 5)$ and $\mathcal{K} = (3, 6, 2, 4)$, K_G=14.

and large values. We were hence concerned with verifying that the computation of derivatives $D_+^h \mathbf{N}_G$ is not affected by inaccuracy. Comparison of the results obtained with 16 and 32 digits of precision has shown that derivatives are computed within the same accuracy as the elements of M.

- For RDE-type steps, the calculation of the coefficients α_i in (8) involves ratio of derivatives of same order and degree. In particular, the hth derivatives of B-spline basis functions of (local) degree d_j will produce the factor $d_j \cdot \ldots \cdot d_{j-h+1}$ at both the numerator and the denominator of (8) and thus that factor can be omitted. This implementation measure helps in containing the magnitude of derivatives of high order in RDE type steps.

- Fixed the number and location of RDE and RKI steps, the sequence of these steps is not compulsory. In particular, as observed in Section 3, the RDE-RKI Algorithm implements just one of the possible sequences of RKI and RDE steps leading to a target space. In this respect, numerical experiments show that different admissible orderings of the steps produce analogous results in terms of accuracy. We can therefore infer that the specific order in which RDE and RKI steps are performed does not significantly affect the overall accuracy of the procedure.

- For nonuniform breakpoint sequences, the lengths of breakpoint intervals act as scaling factors on the derivatives of MDB-splines. It can thus be expected that the behavior of the algorithm for very nonuniform breakpoint sequences be similar to the case, illustrated in the tables, of a very uneven sequence of degrees. Our numerical experiments have confirmed an analogous behavior.

- C^1 MD-splines may be retrieved from Definition 3, replacing the condition "$k_i = 0$ if $d_{i-1} \neq d_i$" with "$k_i \leqslant 1$ if $d_{i-1} \neq d_i$", $i = 1, \ldots, q$. These splines and their derivatives can be evaluated by recurrence relations of Cox-de Boor type [8]. The RDE-RKI Algorithm can of course be triggered starting from a C^1 MD-spline space $\mathcal{S}_0$ as well. In this case, the evaluation of the C^1 MDB-spline basis of $\mathcal{S}_0$ has a slightly higher complexity, which is nevertheless compensated by the savings of some RKI-type steps.

- In Section 3.3 we have already shown that the H-operator can be retrieved as a special case of the more general matrix representation in this paper. It is also reasonable to expect an analogous behavior of the two algorithms (H-operator and RKI Algorithm) in terms of accuracy on the elements of the representation matrix. In fact both methods have the same criticality, due to the fact that, in some cases, it may be necessary to perform arithmetic operations involving high-order derivatives (and thus, possibly, large numbers).

5. Conclusions

We have developed a procedure to compute the MDB-spline basis of a multi-degree spline space, exploiting a matrix representation in terms of another basis which can be chosen according to different criteria. For example, one may want to express the MDB-spline basis starting from local Bernstein bases relative to each breakpoint interval, or from a conventional B-spline basis of highest degree m, or again from a basis of a C^0 MD-spline space. More generally, the initial basis can be drawn from an arbitrary space containing the target one. We have shown that a generalization of the classical Cox-de Boor recurrence relation exists in the particular case where the initial space is featured by C^0 MD-splines and that this setting is a good choice to optimize the performance of the algorithm. An experimental analysis of accuracy for increasing local degrees and continuities has emphasized an excellent numerical behavior of the algorithm for both spaces of practical relevance and in severely critical situations of more academic interest.

Acknowledgements

The authors gratefully acknowledge support from INdAM-GNCS Gruppo Nazionale per il Calcolo Scientifico.

References

[1] Hughes TJR, Cottrell JA, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Engrg 2005;194(39-41):4135–95. doi:10.1016/j.cma.2004.10.008.

[2] Shen W, Wang G. Changeable degree spline basis functions. J Comput Appl Math 2010;234(8):2516–29. doi:10.1016/j.cam.2010.03.015.

[3] Nürnberger G, Schumaker LL, Sommer M, Strauss H. Generalized chebyshevian splines. SIAM J Math Anal 1984;15(4):790–804. doi:10.1137/0515061.

[4] Buchwald B, Mühlbach G. Construction of B-splines for generalized spline spaces generated from local ECT-systems. J Comput Appl Math 2003;159(2):249–67. doi:10.1016/S0377-0427(03)00533-8.

[5] Sederberg TW, Zheng J, Song X. Knot intervals and multi-degree splines. Comput Aided Geom Design 2003;20(7):455–68. doi:10.1016/S0167-8396(03)00096-7.

[6] Beccari C, Casciola G, Morigi S. On multi-degree splines. Comput Aided Geom Design 2017;58:8–23. doi:10.1016/j.cagd.2017.10.003.

[7] Speleers H. Algorithm 999: Computation of multi-degree B-splines. ACM Transactions on Mathematical Software 2019;45(4):1–15.

[8] Beccari CV, Casciola G. A Cox-de Boor-type recurrence relation for C^1 multi-degree splines. Comput Aided Geom Design 2019;75:101784–. doi:https://doi.org/10.1016/j.cagd.2019.101784.

[9] Toshniwal D, Speleers H, Hiemstra RR, Hughes TJ. Multi-degree smooth polar splines: A framework for geometric modeling and isogeometric analysis. Comput Methods Appl Mech Engrg 2017;316:1005–61. doi:10.1016/j.cma.2016.11.009.

[10] Thomas DC, Engvall L, Schmidt SK, Tewa K, Scott MA. U-splines: Splines over unstructured meshes; 2018. Coreform report.

[11] Mazure ML. How to build all Chebyshevian spline spaces good for geometric design? Numer Math 2011;119(3):517–56. doi:10.1007/s00211-011-0390-3.

[12] Beccari CV, Casciola G, Mazure ML. Design or not design? A numerical characterisation for piecewise Chebyshevian splines. Numerical Algorithms 2019;81(1):1–31. doi:10.1007/s11075-018-0533-z.

[13] Beccari CV, Casciola G, Romani L. Computation and modeling in piecewise Chebyshevian spline spaces; 2017. ArXiv:1611.02068.

[14] Hiemstra RR, Hughes TJ, Manni C, Speleers H, Toshniwal D. A Tchebycheffian extension of multi-degree B-splines: Algorithmic computation and properties; 2019. ICES REPORT 19-08, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, May 2019.

[15] Schumaker LL. Spline Functions: Basic Theory. Cambridge, UK: Cambridge University Press; third ed.; 2007.

[16] Shen W, Wang G. A basis of multi-degree splines. Comput Aided Geom Design 2010;27(1):23–35. doi:10.1016/j.cagd.2009.08.005.

[17] Shen W, Wang G, Yin P. Explicit representations of changeable degree spline basis functions. J Comput Appl Math 2013;238(1):39–50. doi:10.1016/j.cam.2012.08.017.

[18] Shen W, Yin P, Tan C. Degree elevation of changeable degree spline. Journal of Computational and Applied Mathematics 2016;300:56 – 67. doi:http://dx.doi.org/10.1016/j.cam.2015.11.030.

[19] Toshniwal D, Speleers H, Hiemstra RR, Manni C, Hughes TJ. Multi-degree B-splines: Algorithmic computation and properties. Comput Aided Geom Design 2020;76:101792–. doi:https://doi.org/10.1016/j.cagd.2019.101792.

[20] Cox M. The numerical evaluation of B-splines. J Inst Maths Applics 1972;10:134–49.

[21] de Boor C. On calculating with B-splines. J Approx Theory 1972;6(1):50–62. doi:10.1016/0021-9045(72)90080-9.

[22] Li X, Huang ZJ, Liu Z. A geometric approach for multi-degree spline. Journal of Computer Science and Technology 2012;27(4):841–50. doi:10.1007/s11390-012-1268-2.

[23] Higham NJ. Accuracy and Stability of Numerical Algorithms. Philadelphia, USA: SIAM Society for Industrial and Applied Mathematics; second ed.; 2002.

[24] Butterfield KR. The computation of all derivatives of a B-spline basis. J Inst Maths Applics 1976;17:15–25.

[25] Ogita T, Rump SM, Oishi S. Accurate sum and dot product. SIAM J Sci and Stat Comput 1983;38:144–66.

[26] Graillat S, Langlois P, Louvet N. Algorithms for accurate, validated and fast polynomial evaluation. Japan J Indust Appl Math 2009;26:191–214.