CHAPTER 2

Basic properties of splines and B-splines

In Chapter 1 we introduced splines through a geometric construction of curves based on repeated averaging, and it turned out that a natural representation of spline curves was as linear combinations of B-splines. In this chapter we start with a detailed study of the most basic properties of B-splines, illustrated by examples and figures in Section 2.1, and in Section 2.2 we formally define spline functions and spline curves. In Section 2.3 we give a matrix representation of splines and B-splines, and this representation is the basis for our development of much of the theory in later chapters.

2.1 Some simple consequences of the recurrence relation

We saw in Theorem 1.5 that a degree d spline curve $\boldsymbol{f}$ can be constructed from n control points $(\boldsymbol{c}_i)_{i=1}^n$ and $n+d+1$ knots $(t_i)_{i=1}^{n+d+1}$ and written as

$$\boldsymbol{f} = \sum_{i=1}^{n} \boldsymbol{c}_i B_{i,d},$$

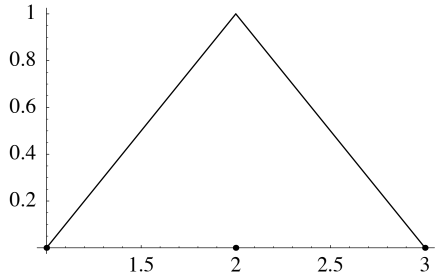
where $\{B_{i,d}\}_{i=1}^n$ are B-splines. In this section we will explore B-splines by considering a number of examples, and deducing some of their most basic properties. For easy reference we start by recording the definition of B-splines. Since we will mainly be working with functions in this chapter, we use x as the independent variable.

Definition 2.1. *Let d be a nonnegative integer and let $\boldsymbol{t} = (t_j)$, the* knot vector *or* knot sequence, *be a nondecreasing sequence of real numbers of length at least $d+2$. The jth B-spline of degree d with knots $\boldsymbol{t}$ is defined by*

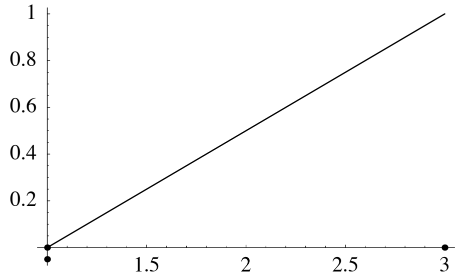
$$B_{j,d,\boldsymbol{t}}(x) = \frac{x - t_j}{t_{j+d} - t_j} B_{j,d-1,\boldsymbol{t}}(x) + \frac{t_{j+1+d} - x}{t_{j+1+d} - t_{j+1}} B_{j+1,d-1,\boldsymbol{t}}(x), \tag{2.1}$$

for all real numbers x, with

$$B_{j,0,\boldsymbol{t}}(x) = \begin{cases} 1, & \text{if } t_j \le x < t_{j+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$



(a)



(b)

Figure 2.1. A linear B-spline with simple knots (a) and double knots (b).

Here, the convention is assumed that '$0/0 = 0$'. When there is no chance of ambiguity, some of the subscripts will be dropped and the B-spline written as either $B_{j,d}$, $B_{j,\boldsymbol{t}}$, or simply B_j.

We say that a knot has *multiplicity* m if it appears m times in the knot sequence. Knots of multiplicity one, two and three are also called *simple, double* and *triple* knots.

Many properties of B-splines can be deduced directly from the definition. One of the most basic properties is that

$$B_{j,d}(x) = 0 \quad \text{for all } x \text{ when } t_j = t_{j+d+1}$$

which we made use of in Chapter 1. This is true by definition for $d = 0$. If it is true for B-splines of degree $d-1$, the zero convention means that if $t_j = t_{j+d+1}$ then both $B_{j,d-1}(x)/(t_{j+d} - t_j)$ and $B_{j+1,d-1}(x)/(t_{j+1+d} - t_{j+1})$ on the right in (2.1) are zero, and hence $B_{j,d}(x)$ is zero. The recurrence relation can therefore be expressed more explicitly as

$$B_{j,d}(x) = \begin{cases} 0, & \text{if } t_j = t_{j+1+d}; \\ s_1(x), & \text{if } t_j < t_{j+d} \text{ and } t_{j+1} = t_{j+1+d}; \\ s_2(x), & \text{if } t_j = t_{j+d} \text{ and } t_{j+1} < t_{j+1+d}; \\ s_1(x) + s_2(x), & \text{otherwise}; \end{cases} \tag{2.3}$$

where

$$s_1(x) = \frac{x - t_j}{t_{j+d} - t_j} B_{j,d-1}(x) \qquad \text{and} \qquad s_2(x) = \frac{t_{j+1+d} - x}{t_{j+1+d} - t_{j+1}} B_{j+1,d-1}(x)$$

for all x.

The following example shows that linear B-splines are quite simple.

Example 2.2 (B-splines of degree 1)**.** One application of the recurrence relation gives

$$B_{j,1}(x) = \frac{x - t_j}{t_{j+1} - t_j} B_{j,0}(x) + \frac{t_{j+2} - x}{t_{j+2} - t_{j+1}} B_{j+1,0}(x) = \begin{cases} (x - t_j)/(t_{j+1} - t_j), & \text{if } t_j \le x < t_{j+1}; \\ (t_{j+2} - x)/(t_{j+2} - t_{j+1}), & \text{if } t_{j+1} \le x < t_{j+2}; \\ 0, & \text{otherwise} \end{cases}.$$

A plot of this *hat function* is shown in Figure 2.1 (a) in a typical case where $t_j < t_{j+1} < t_{j+2}$. The figure shows clearly that $B_{j,1}$ consists of linear polynomial pieces, with breaks at the knots. In Figure 2.1 (b), the two knots t_j and t_{j+1} are identical; then the first linear piece is identically zero since $B_{j,0} = 0$, and $B_{j,1}$ is discontinuous. This provides an illustration of the smoothness properties of B-splines: a linear B-spline is discontinuous at a double knot, but continuous at simple knots.

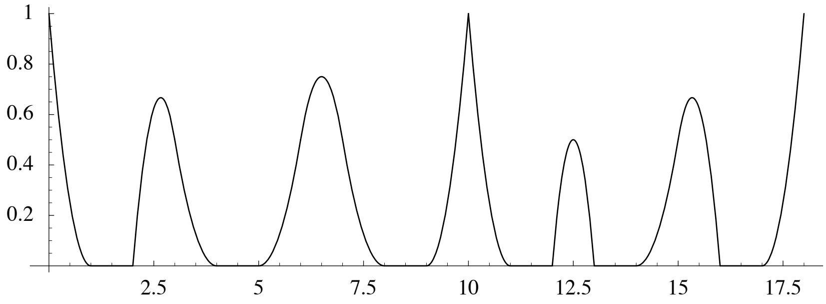


Figure 2.2. From left to right we see the quadratic B-splines $B(x \mid 0,0,0,1)$, $B(x \mid 2,2,3,4)$, $B(x \mid 5,6,7,8)$, $B(x \mid 9,10,10,11)$, $B(x \mid 12,12,13,13)$, $B(x \mid 14,15,16,16)$, and $B(x \mid 17,18,18,18)$.

The B-spline $B_{j,d}$ depends only on the knots $(t_k)_{k=j}^{j+d+1}$. For B-splines of degree 0 this is clear from equation (2.2), and Example 2.2 shows that it is also true for B-splines of degree 1. To show that it is true in general we use induction and assume that $B_{j,d-1}$ only depends on $(t_k)_{k=j}^{j+d}$ and $B_{j+1,d-1}$ only depends on $(t_k)_{k=j+1}^{j+d+1}$. By examining the recurrence relation (2.1) we see that then $B_{j,d}$ can only depend on the knots $(t_k)_{k=j}^{j+d+1}$, as we claimed.

The notation $B_{j,d}(x) = B(x \mid t_j, \ldots, t_{j+d+1})$ will sometimes be used to emphasise the dependence of a B-spline on the individual knots. For example, if $d \geq 2$ and if we set $(t_j, t_{j+1}, \ldots, t_{j+d}, t_{j+d+1}) = (a, b, \ldots, c, d)$, then (2.1) can be written

$$B(x \mid a, b, \ldots, c, d)(x) = \frac{x-a}{c-a} B(x \mid a, b, \ldots, c) + \frac{d-x}{d-b} B(x \mid b, \ldots, c, d). \tag{2.4}$$

Example 2.3 (Quadratic B-splines)**.** Using the zero convention and (2.4) we find

1. $B(x \mid 0,0,0,1) = (1-x)B(x \mid 0,0,1) = (1-x)^2 B(x \mid 0,1)$.
2. $B(x \mid 0,0,1,2) = x(2 - \frac{3}{2}x)B(x \mid 0,1) + \frac{1}{2}(2-x)^2 B(x \mid 1,2)$.
3. $B(x \mid 0,1,2,3) = \frac{x^2}{2} B(x \mid 0,1) + \Big(\frac{3}{4} - (x - \frac{3}{2})^2\Big) B(x \mid 1,2) + \frac{(3-x)^2}{2} B(x \mid 2,3)$.
4. $B(x \mid 0,1,1,2) = x^2 B(x \mid 0,1) + (2-x)^2 B(x \mid 1,2)$.
5. $B(x \mid 0,0,1,1) = 2x(1-x)B(x \mid 0,1)$.
6. $B(x \mid 0,1,2,2) = \frac{1}{2}x^2 B(x \mid 0,1) + (2-x)(\frac{3}{2}x - 1)B(x \mid 1,2)$.
7. $B(x \mid 0,1,1,1) = x^2 B(x \mid 0,1)$.

Translates (see (2.6)) of these functions are shown in Figure 2.2. Note that the B-spline $B(x \mid 0,1,2,3)$ consists of three nonzero polynomial pieces, but that in general the number of nonzero pieces depends on the multiplicity of the knots. For example, the functions $B(x \mid 0,0,0,1)$ and $B(x \mid 0,0,1,1)$ consist of only one nonzero piece. Figure 2.2 illustrates these smoothness properties of B-splines: At a single knot a quadratic B-spline is continuous and has a continuous derivative, at a double knot it is continuous, while at a triple knot it is discontinuous.

Figure 2.3 shows the quadratic B-spline $B(x \mid 0,1,2,3)$ together with its constituent polynomial pieces. Note how the three parabolas join together smoothly to make the B-spline have continuous first derivative at every point.

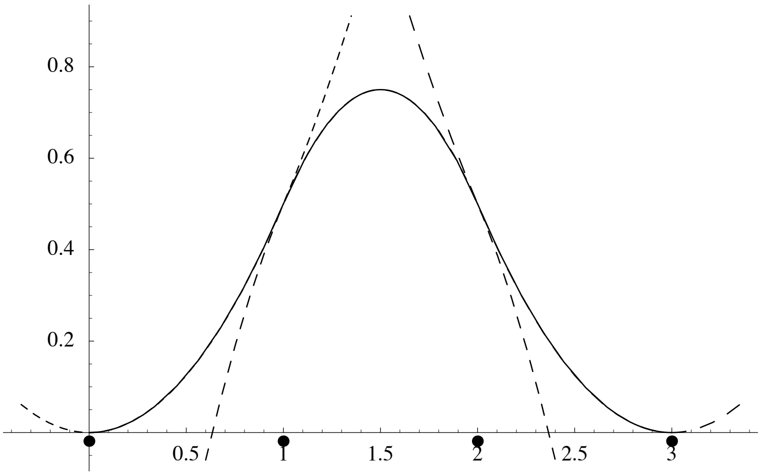


Figure 2.3. The different polynomial pieces of a quadratic B-spline.

By applying the recurrence relation (2.1) twice we obtain an explicit expression for a generic quadratic B-spline,

$$
\begin{aligned}
B_{j,2}(x) &= \frac{x-t_j}{t_{j+2}-t_j}\Big[\frac{x-t_j}{t_{j+1}-t_j}B_{j,0}(x) + \frac{t_{j+2}-x}{t_{j+2}-t_{j+1}}B_{j+1,0}(x)\Big] \\
&\quad + \frac{t_{j+3}-x}{t_{j+3}-t_{j+1}}\Big[\frac{x-t_{j+1}}{t_{j+2}-t_{j+1}}B_{j+1,0}(x) + \frac{t_{j+3}-x}{t_{j+3}-t_{j+2}}B_{j+2,0}(x)\Big] \\
&= \frac{(x-t_j)^2}{(t_{j+2}-t_j)(t_{j+1}-t_j)}B_{j,0}(x) + \frac{(t_{j+3}-x)^2}{(t_{j+3}-t_{j+1})(t_{j+3}-t_{j+2})}B_{j+2,0}(x) \\
&\quad + \Big(\frac{(x-t_j)(t_{j+2}-x)}{(t_{j+2}-t_j)(t_{j+2}-t_{j+1})} + \frac{(t_{j+3}-x)(x-t_{j+1})}{(t_{j+3}-t_{j+1})(t_{j+2}-t_{j+1})}\Big)B_{j+1,0}(x).
\end{aligned}
\tag{2.5}
$$

The complexity of this expression gives us a good reason to work with B-splines through other means than explicit formulas.

Figure 2.4 shows some cubic B-splines. The middle B-spline, $B(x \mid 9, 10, 11, 12, 13)$, has simple knots and its second derivative is therefore continuous for all real numbers x, including the knots. In general a cubic B-spline has $3 - m$ continuous derivatives at a knot of multiplicity m for $m = 1, 2, 3$. A cubic B-spline with a knot of multiplicity 4 is discontinuous at the knot.

Before considering the next example we show that B-splines possess a property called *translation invariance*. Mathematically this is expressed by the formula

$$B(x+y \mid t_j+y, \ldots, t_{j+d+1}+y) = B(x \mid t_j, \ldots, t_{j+d+1}) \quad x, y \in \mathbb{R}. \tag{2.6}$$

We argue by induction, and start by checking the case $d = 0$. We have

$$B(x+y \mid t_j+y, t_{j+1}+y) = \begin{cases} 1, & \text{if } t_j+y \le x+y < t_{j+1}+y; \\ 0, & \text{otherwise} \end{cases} = \begin{cases} 1, & \text{if } t_j \le x < t_{j+1}; \\ 0, & \text{otherwise,} \end{cases}$$

so equation (2.6) holds for $d = 0$. Suppose that the translation invariance holds for B-splines of degree $d-1$. In the recurrence (2.1) for the left-hand-side of (2.6) the first coefficient $(x-t_j)/(t_{j+d}-t_j)$ can be written

$$\frac{(x+y)-(t_j+y)}{(t_{j+d}+y)-(t_j+y)} = \frac{x-t_j}{t_{j+d}-t_j},$$

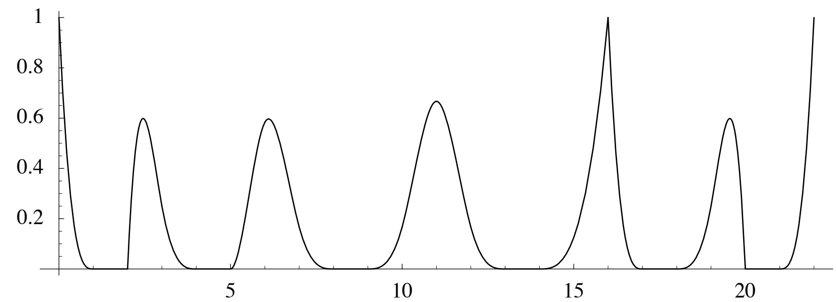


Figure 2.4. From left to right we see the cubic B-splines $B(x \mid 0,0,0,0,1)$, $B(x \mid 2,2,2,3,4)$, $B(x \mid 5,5,6,7,8)$, $B(x \mid 9,10,11,12,13)$, $B(x \mid 14,16,16,16,17)$, $B(x \mid 18,19,20,20,20)$, and $B(x \mid 21,22,22,22,22)$.

i.e., the same as before translation. This also holds for the other coefficient $(t_{j+d+1} - x)/(t_{j+d+1} - t_{j+1})$ in (2.1). Since the two B-splines of degree $d-1$ are translation invariant by the induction hypothesis, we conclude that (2.6) holds for all polynomial degrees.

Example 2.4 (Uniform B-splines)**.** The B-splines on a uniform knot vector are of special interest. Let the knots be the set $\mathbb{Z}$ of all integers. We index this knot sequence by letting $t_j = j$ for all integers j. We denote the uniform B-spline of degree $d \geq 0$ by

$$M_d(x) = B_{0,d}(x) = B(x \mid 0, 1, \cdots, d+1), \quad x \in \mathbb{R}. \tag{2.7}$$

The functions M_d are also called *cardinal B-splines*. On this knot vector all B-splines can be written as translates of the function M_d. Using (2.6) we have

$$B_{j,d}(x) = B(x \mid j, j+1, \dots, j+d+1) = B(x-j \mid 0, 1, \dots, d+1) = M_d(x-j) \quad \text{for all } j.$$

In particular, $B_{1,d-1}(x) = B(x \mid 1, \dots, d+1) = M_{d-1}(x-1)$ and the recurrence relation implies that for $d \geq 1$

$$M_d(x) = \frac{x}{d} M_{d-1}(x) + \frac{d+1-x}{d} M_{d-1}(x-1). \tag{2.8}$$

Using this recurrence we can compute the first few uniform B-splines

$$\begin{aligned}
M_1(x) &= x M_0(x) + (2-x) M_0(x-1) \\
M_2(x) &= \frac{x^2}{2} M_0(x) + \Big(\frac{3}{4} - (x - \frac{3}{2})^2\Big) M_0(x-1) + \frac{(3-x)^2}{2} M_0(x-2) \\
M_3(x) &= \frac{x^3}{6} M_0(x) + \Big(\frac{2}{3} - \frac{1}{2} x (x-2)^2\Big) M_0(x-1) \\
&\quad + \Big(\frac{2}{3} - \frac{1}{2}(4-x)(x-2)^2\Big) M_0(x-2) + \frac{(4-x)^3}{6} M_0(x-3)
\end{aligned} \tag{2.9}$$

(compare with Examples 2.2 and 2.3). As we shall see in Chapter 3, the B-spline M_d has $d-1$ continuous derivatives at the knots. The quadratic cardinal B-spline M_2 is shown in Figure 2.2, translated to the interval $[5, 8]$, while M_3 is shown in Figure 2.4, translated to $[9, 13]$.

Example 2.5 (Bernstein polynomials)**.** The Bernstein polynomials that appeared in the representation of Bézier curves in Section 1.4 are special cases of B-splines. In fact it turns out that the jth Bernstein polynomial on the interval $[a, b]$ is (almost) given by

$$B_j^d(x) = B(x \mid \overbrace{a, \dots, a}^{d+1-j}, \overbrace{b, \dots, b}^{j+1}), \quad \text{for } j = 0, \dots, d.$$

The recurrence relation (2.4) now takes the form

$$\begin{aligned}
B_j^d(x) &= \frac{x-a}{b-a} B(x \mid \overbrace{a, \dots, a}^{d+1-j}, \overbrace{b, \dots, b}^{j}) + \frac{b-x}{b-a} B(x \mid \overbrace{a, \dots, a}^{d-j}, \overbrace{b, \dots, b}^{j+1}) \\
&= \frac{x-a}{b-a} B_{j-1}^{d-1}(x) + \frac{b-x}{b-a} B_j^{d-1}(x).
\end{aligned} \tag{2.10}$$

This is also valid for $j = 0$ and $j = d$ if we define $B_j^{d-1} = 0$ for $j < 0$ and $j \geq d$. Using induction on d one can show the explicit formula

$$B_j^d(x) = \binom{d}{j} \left(\frac{x-a}{b-a}\right)^j \left(\frac{b-x}{b-a}\right)^{d-j} B(x \mid a,b), \quad \text{for } j = 0, 1, \ldots, d, \tag{2.11}$$

see exercise 5. These are essentially the Bernstein polynomials for the interval $[a, b]$, except that the factor $B(x \mid a, b)$ causes B_j^d to be zero outside $[a, b]$. To represent Bézier curves, it is most common to use the Bernstein polynomials on the interval $[0, 1]$ as in Section 1.4, i.e., with $a = 0$ and $b = 1$,

$$B_j^d(x) = \binom{d}{j} x^j (1-x)^{d-j} B(x \mid 0,1) = b_{j,d}(x) B(x \mid 0,1), \quad \text{for } j = 0, 1, \ldots, d; \tag{2.12}$$

here b_j^d is the jth Bernstein polynomial of degree d. For example, the quadratic Bernstein basis polynomials are given by

$$b_{0,2}(x) = (1-x)^2, \quad b_{1,2}(x) = 2x(1-x), \quad b_{2,2}(x) = x^2$$

which agrees with what we found in Chapter 1. These functions can also be recognised as the polynomial part of the special quadratic B-splines in (1), (5) and (7) in Example 2.3. For Bernstein polynomials on $[0, 1]$ the recurrence relation (2.10) takes the form

$$b_{j,d}(x) = x b_{j-1,d-1}(x) + (1-x) b_{j,d-1}(x), \quad j = 0, 1, \ldots, d. \tag{2.13}$$

We have now seen a number of examples of B-splines and some characteristic features are evident. The following lemma sums up the most basic properties.

Lemma 2.6. *Let d be a nonnegative polynomial degree and let $\boldsymbol{t} = (t_j)$ be a knot sequence. The B-splines on $\boldsymbol{t}$ have the following properties:*

1. Local knots. *The jth B-spline $B_{j,d}$ depends only on the knots $t_j, t_{j+1}, \ldots, t_{j+d+1}$.*

2. Local support.

   (a) *If x is outside the interval $[t_j, t_{j+d+1})$ then $B_{j,d}(x) = 0$. In particular, if $t_j = t_{j+d+1}$ then $B_{j,d}$ is identically zero.*

   (b) *If x lies in the interval $[t_\mu, t_{\mu+1})$ then $B_{j,d}(x) = 0$ if $j < \mu - d$ or $j > \mu$.*

3. Positivity. *If $x \in (t_j, t_{j+d+1})$ then $B_{j,d}(x) > 0$. The closed interval $[t_j, t_{j+d+1}]$ is called* the support *of $B_{j,d}$.*

4. Piecewise polynomial. *The B-spline $B_{j,d}$ can be written*

$$B_{j,d}(x) = \sum_{k=j}^{j+d} B_{j,d}^k(x) B_{k,0}(x) \tag{2.14}$$

   where each $B_{j,d}^k(x)$ is a polynomial of degree d.

5. Special values. *If $z = t_{j+1} = \cdots = t_{j+d} < t_{j+d+1}$ then $B_{j,d}(z) = 1$ and $B_{i,d}(z) = 0$ for $i \neq j$.*

6. Smoothness. *If the number z occurs m times among $t_j, \ldots, t_{j+d+1}$ then the derivatives of $B_{j,d}$ of order $0, 1, \ldots, d-m$ are all continuous at z.*

Proof. Properties 1–3 follow directly, by induction, from the recurrence relation, see exercise 3. In Section 1.5 in Chapter 1 we saw that the construction of splines produced piecewise polynomials, so this explains property 4. Property 5 is proved in exercise 6 and property 6 will be proved in Chapter 3. ∎

2.2 Linear combinations of B-splines

In Theorem 1.5 we saw that B-splines play a central role in the representation of spline curves. The purpose of this section is to define precisely what we mean by spline functions and spline curves and related concepts like the control polygon.

2.2.1 Spline functions

The B-spline $B_{j,d}$ depends on the knots $t_j, \ldots, t_{j+1+d}$. This means that if the knot vector is given by $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ for some positive integer n, we can form n B-splines $\{B_{j,d}\}_{j=1}^{n}$ of degree d associated with this knot vector. A linear combination of B-splines, or a spline function, is a combination of B-splines on the form

$$f = \sum_{j=1}^{n} c_j B_{j,d}, \tag{2.15}$$

where $\boldsymbol{c} = (c_j)_{j=1}^{n}$ are n real numbers. We formalise this in a definition.

Definition 2.7 (Spline functions)**.** *Let $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ be a nondecreasing sequence of real numbers, i.e., a knot vector for a total of n B-splines. The linear space of all linear combinations of these B-splines is the* spline space $\mathbb{S}_{d,\boldsymbol{t}}$ *defined by*

$$\mathbb{S}_{d,\boldsymbol{t}} = \operatorname{span}\{B_{1,d}, \ldots, B_{n,d}\} = \Big\{ \sum_{j=1}^{n} c_j B_{j,d} \mid c_j \in \mathbb{R} \text{ for } 1 \le j \le n \Big\}.$$

An element $f = \sum_{j=1}^{n} c_j B_{j,d}$ of $\mathbb{S}_{d,\boldsymbol{t}}$ is called a spline function, or just a spline, of degree d with knots $\boldsymbol{t}$, and $(c_j)_{j=1}^{n}$ are called the B-spline coefficients *of f.*

As we shall see later, B-splines are linearly independent so $\mathbb{S}_{d,\boldsymbol{t}}$ is a linear space of dimension n.

It will often be the case that the exact length of the knot vector is of little interest. Then we may write a spline as $\sum_j c_j B_{j,d}$ without specifying the upper and lower bounds on j.

Example 2.8 (A linear spline)**.** Let $(x_i, y_i)_{i=1}^{m}$ be a set of data points with $x_i < x_{i+1}$ for $i = 1, 2, \ldots, m-1$. On the knot vector

$$\boldsymbol{t} = (t_j)_{j=1}^{m+2} = (x_1, x_1, x_2, x_3, \ldots, x_{m-1}, x_m, x_m)$$

we consider the linear ($d = 1$) spline function

$$s(x) = \sum_{j=1}^{m} y_j B_{j,1}(x), \quad \text{for } x \in [x_1, x_m].$$

From Example 2.2 we see that s satisfies the interpolatory conditions

$$s(x_i) = \sum_{j=1}^{m} y_j B_{j,1}(x_i) = y_i, \quad i = 1, \ldots, m-1 \tag{2.16}$$

since $B_{i,1}(x_i) = 1$ and all other B-splines are zero at x_i. At $x = x_m$ all the B-splines are zero according to Definition 2.1. But the limit of $B_m(x)$ when x tends to x_m from the left is 1. Equation (2.16) therefore

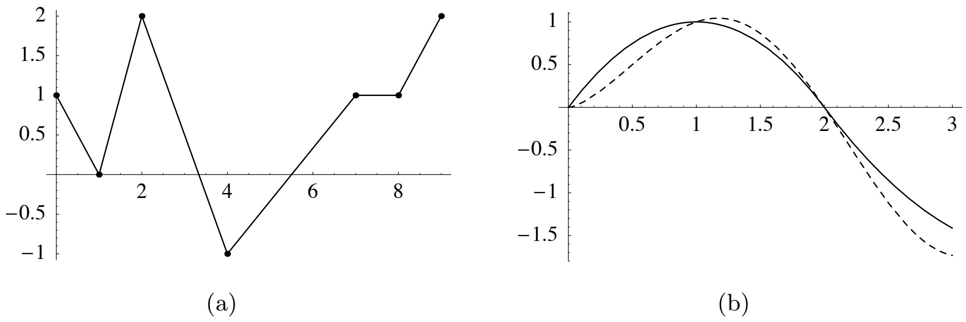


Figure 2.5. A linear spline interpolating data (a), and a quadratic spline (solid) that approximates $\sin(\pi x/2)$ (dashed).

also holds for $i = m$ if we take limits from the left at $x = x_m$. In addition s is linear on each subinterval $[t_\mu, t_{\mu+1})$ since

$$\begin{aligned} s(x) &= y_{\mu-1}B_{\mu-1,1}(x) + y_\mu B_{\mu,1}(x) \\ &= \frac{t_{\mu+1}-x}{t_{\mu+1}-t_\mu}y_{\mu-1} + \frac{x-t_\mu}{t_{\mu+1}-t_\mu}y_\mu. \end{aligned} \tag{2.17}$$

when x is in $[t_\mu, t_{\mu+1})$. It follows that s is the piecewise linear interpolant to the data. An example is shown in Figure 2.5 (a).

Example 2.9 (A quadratic spline)**.** Let $f : [a, b] \to \mathbb{R}$ be a given function defined on some interval $[a, b]$, and let n be an integer greater than 2. On $[a, b]$ we assume that we have a knot vector $\boldsymbol{t} = (t_j)_{j=1}^{n+3}$, where

$$a = t_1 = t_2 = t_3 < t_4 < \cdots < t_n < t_{n+1} = t_{n+2} = t_{n+3}.$$

We can then define the quadratic spline function

$$s(x) = Qf(x) = \sum_{j=1}^{n} f(t_j^*)B_{j,2}(x),$$

where

$$t_j^* = (t_{j+1} + t_{j+2})/2, \quad j = 1, \ldots, n.$$

We note that

$$a = t_1^* < t_2^* < \cdots < t_n^* = b.$$

The function Qf is called the *Variation Diminishing Spline Approximation* to f of degree 2. As a particular instance of this approximation we approximate the function $f(x) = \sqrt{2}\sin\left(\frac{\pi}{2}x\right)$ on the interval $[0, 3]$. With

$$\boldsymbol{t} = (t_j)_{j=1}^{8} = (0, 0, 0, 1, 2, 3, 3, 3),$$

we obtain $(t_j^*)_{j=1}^5 = (0, 1/2, 3/2, 5/2, 3)$ and

$$s(x) = B_{2,2}(x) + B_{3,2}(x) - B_{4,2}(x) - \sqrt{2}B_{5,2}(x).$$

A plot of this function together with $f(x)$ is shown in Figure 2.5 (b).

Example 2.10 (A cubic polynomial in Bernstein form)**.** On the knot vector

$$\boldsymbol{t} = (t_j)_{j=1}^{8} = (0, 0, 0, 0, 1, 1, 1, 1)$$

we consider the cubic spline function

$$s(x) = -B_{1,3}(x) + 5B_{2,3}(x) - 5B_{3,3}(x) + B_{4,3}(x).$$

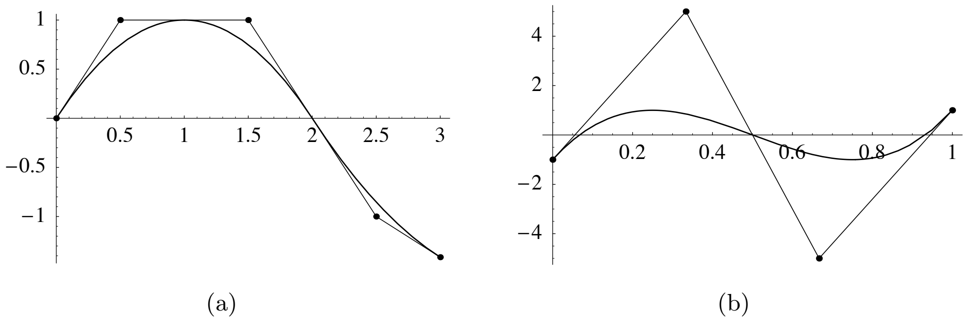


Figure 2.6. The quadratic spline from Example 2.9 with its control polygon (a) and the cubic Chebyshev polynomial with its control polygon (b).

In terms of the cubic Bernstein basis we have

$$s(x) = -b_{0,3}(x) + 5b_{1,3}(x) - 5b_{2,3} + b_{3,3}, \quad 0 \le x \le 1.$$

This polynomial is shown in Figure 2.6 (b). It is the cubic *Chebyshev polynomial* with respect to the interval $[0, 1]$.

Note that the knot vectors in the above examples all have knots of multiplicity $d + 1$ at both ends. If in addition no knot occurs with multiplicity higher than $d + 1$ (as in the examples), the knot vector is said to be $d + 1$-*regular*.

When we introduced spline curves in Chapter 1, we saw that a curve mimicked the shape of its control polygon in an intuitive way. The control polygon of a spline function is not quite as simple as for curves since the B-spline coefficients of a spline function is a number. What is needed is an abscissa to associate with each coefficient.

Definition 2.11 (Control polygon for spline functions)**.** *Let $f = \sum_{j=1}^{n} c_j B_{j,d}$ be a spline in $\mathbb{S}_{d,\boldsymbol{t}}$. The* control points *of f are the points with coordinates (t_j^*, c_j) for $j = 1, \ldots, n$, where*

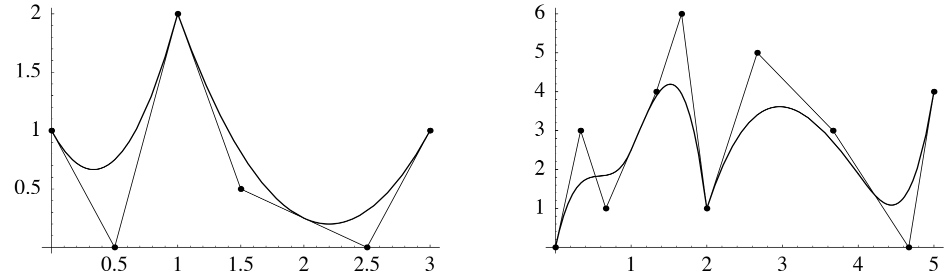
$$\boldsymbol{t}_j^* = \frac{t_{j+1} + \cdots + t_{j+d}}{d}$$

are the knot averages *of $\boldsymbol{t}$. The* control polygon *of f is the piecewise linear function obtained by connecting neighbouring control points by straight lines.*

Some spline functions are shown with their control polygons in Figures 2.6–2.7. It is quite striking how the spline is a smoothed out version of the control polygon. In particular we notice that at a knot with multiplicity at least d, the spline and its control polygon agree. This happens at the beginning and end of all the splines since we have used $d + 1$-regular knot vectors, and also at some points in the interior for the splines in Figure 2.7. We also note that the control polygon is tangent to the spline function at a knot of multiplicity d or $d + 1$. This close relationship between a spline and its control polygon is a geometric instance of one of the many nice properties possessed by splines represented in terms of B-splines.

From our knowledge of B-splines we immediately obtain some basic properties of splines.

Lemma 2.12. *Let $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ be a knot vector for splines of degree d with $n \ge d + 1$, and let $f = \sum_{j=1}^{n} c_j B_{j,d}$ be a spline in $\mathbb{S}_{d,\boldsymbol{t}}$. Then f has the following properties:*



(a) (b)

Figure 2.7. Two splines with corresponding control polygons. The spline in (a) is quadratic with knots $\boldsymbol{t} = (0,0,0,1,1,2,3,3,3)$ and B-spline coefficients $\boldsymbol{c} = (1,0,2,1/2,0,1)$, while the spline in (b) is cubic with knots $\boldsymbol{t} = (0,0,0,0,1,1,2,2,2,4,5,5,5,5)$ and B-spline coefficients $0,3,1,4,6,1,5,3,0,4$).

1. *If x is in the interval $[t_\mu, t_{\mu+1})$ for some μ in the range $d+1 \le \mu \le n$ then*

$$f(x) = \sum_{j=\mu-d}^{\mu} c_j B_{j,d}(x).$$

2. *If $z = t_{j+1} = \cdots = t_{j+d} < t_{j+d+1}$ for some j in the range $1 \le j \le n$ then $f(z) = c_j$.*
3. *If z occurs m times in $\boldsymbol{t}$ then f has continuous derivatives of order $0, \ldots, d-m$ at z.*

Proof. This follows directly from Lemma 2.6. ∎

2.2.2 Spline curves

For later reference we give a precise definition of spline curves, although we have already made extensive use of them in Chapter 1.

In many situations spline functions will be the right tool to represent a set of data or some desired shape. But as we saw in Section 1.2, functions have some inherent restrictions in that, for a given x, a function can only take on one function value. We saw that one way to overcome this restriction was by representing the x- and y-components by two different functions,

$$\boldsymbol{f}(u) = \bigl(f_1(u), f_2(u)\bigr).$$

Vector functions in higher dimensions are obtained by adding more components. We will be particularly interested in the special case where all the components are spline functions on a common knot vector.

Definition 2.13 (Spline curves). *Let $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ be a nondecreasing sequence of real numbers, and let $q \ge 2$ be an integer. The space of all spline curves in $\mathbb{R}^q$ of degree d and with knots $\boldsymbol{t}$ is defined as*

$$\mathbb{S}_{d,\boldsymbol{t}}^q = \Bigl\{\sum_{j=1}^{n} \boldsymbol{c}_j B_{j,d} \mid \boldsymbol{c}_j \in \mathbb{R}^q \text{ for } 1 \le j \le n\Bigr\}.$$

More precisely, an element $f = \sum_{j=1}^{n} c_j B_{j,d}$ of $\mathbb{S}_{d,\boldsymbol{t}}^q$ is called a spline vector function *or* a parametric spline curve *of degree d with knots $\boldsymbol{t}$, and $(\boldsymbol{c}_j)_{j=1}^{n}$ are called the* B-spline coefficients *or* control points *of f.*

We have already defined what we mean by the control polygon of a spline curve, but for reference we repeat the definition here.

Definition 2.14 (Control polygon for spline curves)**.** *Let $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ be a knot vector for splines of degree d, and let $\boldsymbol{f} = \sum_{j=1}^{n} \boldsymbol{c}_j B_{j,d}$ be a spline curve in $\mathbb{S}_{d,\boldsymbol{t}}^q$ for $q \geq 2$. The* control polygon *of f is the piecewise linear function obtained by connecting neighbouring control points by straight lines.*

Some examples of spline curves with their control polygons can be found in Section 1.5.

Spline curves may be thought of as spline functions with B-spline coefficients that are vectors. This means that virtually all the algorithms that we develop for spline functions can be generalised to spline curves by simply applying the functional version of the algorithm to each component of the curve in turn.

2.3 A matrix representation of B-splines

Mathematical objects defined by recurrence relations can become very complex even if the recurrence relation is simple. This is certainly the case for B-splines. The structure of the recurrence relation (2.1) is relatively simple, but if we try to determine the symbolic expressions for the individual pieces of a B-spline in terms of the knots and the variable x, for degree five or six, the algebraic complexity of the expressions is perhaps the most striking feature. It turns out that these rather complex formulas can be represented in terms of products of simple matrices, and this is the theme of this section. This representation will be used in Section 3.1 to show how polynomials can be represented in terms of B-splines and to prove that B-splines are linearly independent. In Section 2.4 we will make use of the matrix notation to develop algorithms for computing function values and derivatives of splines. The matrix representation will also be useful in the theory of knot insertion in Chapter 4.

We start by introducing the matrix representation for linear, quadratic and cubic splines in three examples.

Example 2.15 (Vector representation of linear B-splines)**.** Consider the case of linear B-splines with knots $\boldsymbol{t}$, and focus on one nonempty knot interval $[t_\mu, t_{\mu+1})$. We have already seen in previous sections that in this case the B-splines are quite simple. From the support properties of B-splines we know that the only linear B-splines that are nonzero on this interval are $B_{\mu-1,1}$ and $B_{\mu,1}$ and their restriction to the interval can be given in vector form as

$$\begin{pmatrix} B_{\mu-1,1} & B_{\mu,1} \end{pmatrix} = \begin{pmatrix} \dfrac{t_{\mu+1}-x}{t_{\mu+1}-t_\mu} & \dfrac{x-t_\mu}{t_{\mu+1}-t_\mu} \end{pmatrix}. \tag{2.18}$$

Example 2.16 (Matrix representation of quadratic B-splines)**.** The matrices appear when we come to quadratic splines. We consider the same nonempty knot interval $[t_\mu, t_{\mu+1})$; the only nonzero quadratic B-splines on this interval are $\{B_{j,2}\}_{j=\mu-2}^{\mu}$. By checking with Definition 2.1 we see that for x in $[t_\mu, t_{\mu+1})$, the row vector of these B-splines may be written as the product of two simple matrices,

$$\begin{aligned}
\begin{pmatrix} B_{\mu-2,2} & B_{\mu-1,2} & B_{\mu,2} \end{pmatrix} &= \begin{pmatrix} B_{\mu-1,1} & B_{\mu,1} \end{pmatrix} \begin{pmatrix} \dfrac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu-1}} & \dfrac{x-t_{\mu-1}}{t_{\mu+1}-t_{\mu-1}} & 0 \\ 0 & \dfrac{t_{\mu+2}-x}{t_{\mu+2}-t_\mu} & \dfrac{x-t_\mu}{t_{\mu+2}-t_\mu} \end{pmatrix} \\
&= \begin{pmatrix} \dfrac{t_{\mu+1}-x}{t_{\mu+1}-t_\mu} & \dfrac{x-t_\mu}{t_{\mu+1}-t_\mu} \end{pmatrix} \begin{pmatrix} \dfrac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu-1}} & \dfrac{x-t_{\mu-1}}{t_{\mu+1}-t_{\mu-1}} & 0 \\ 0 & \dfrac{t_{\mu+2}-x}{t_{\mu+2}-t_\mu} & \dfrac{x-t_\mu}{t_{\mu+2}-t_\mu} \end{pmatrix}.
\end{aligned} \tag{2.19}$$

If these matrices are multiplied together the result would of course agree with that in Example 2.3. However, the power of the matrix representation lies in the factorisation itself, as we will see in the next section. To obtain the value of the B-splines we can multiply the matrices together, but this should be done numerically, after values have been assigned to the variables. In practise this is only done implicitly, see the algorithms in Section 2.4.

Example 2.17 (Matrix representation of cubic B-splines)**.** In the cubic case the only nonzero B-splines on $[t_\mu, t_{\mu+1})$ are $\{B_{j,3}\}_{j=\mu-3}^{\mu}$. Again it can be checked with Definition 2.1 that for x in this interval these B-splines may be written

$$
\begin{aligned}
\big(B_{\mu-3,3} \;\; B_{\mu-2,3} \;\; B_{\mu-1,3} \;\; B_{\mu,3}\big) &= \big(B_{\mu-2,2} \;\; B_{\mu-1,2} \;\; B_{\mu,2}\big)
\begin{pmatrix}
\frac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu-2}} & \frac{x-t_{\mu-2}}{t_{\mu+1}-t_{\mu-2}} & 0 & 0 \\
0 & \frac{t_{\mu+2}-x}{t_{\mu+2}-t_{\mu-1}} & \frac{x-t_{\mu-1}}{t_{\mu+2}-t_{\mu-1}} & 0 \\
0 & 0 & \frac{t_{\mu+3}-x}{t_{\mu+3}-t_{\mu}} & \frac{x-t_{\mu}}{t_{\mu+3}-t_{\mu}}
\end{pmatrix} \\
&= \begin{pmatrix} \frac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu}} & \frac{x-t_{\mu}}{t_{\mu+1}-t_{\mu}} \end{pmatrix}
\begin{pmatrix}
\frac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu-1}} & \frac{x-t_{\mu-1}}{t_{\mu+1}-t_{\mu-1}} & 0 \\
0 & \frac{t_{\mu+2}-x}{t_{\mu+2}-t_{\mu}} & \frac{x-t_{\mu}}{t_{\mu+2}-t_{\mu}}
\end{pmatrix} \\
&\quad \begin{pmatrix}
\frac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu-2}} & \frac{x-t_{\mu-2}}{t_{\mu+1}-t_{\mu-2}} & 0 & 0 \\
0 & \frac{t_{\mu+2}-x}{t_{\mu+2}-t_{\mu-1}} & \frac{x-t_{\mu-1}}{t_{\mu+2}-t_{\mu-1}} & 0 \\
0 & 0 & \frac{t_{\mu+3}-x}{t_{\mu+3}-t_{\mu}} & \frac{x-t_{\mu}}{t_{\mu+3}-t_{\mu}}
\end{pmatrix}.
\end{aligned}
$$

The matrix notation generalises to B-splines of arbitrary degree in the obvious way.

Theorem 2.18. *Let $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ be a knot vector for B-splines of degree d, and let μ be an integer such that $t_\mu < t_{\mu+1}$ and $d+1 \le \mu \le n$. For each positive integer k with $k \le d$ define the matrix $\boldsymbol{R}_k^\mu(x) = \boldsymbol{R}_k(x)$ by*

$$
\boldsymbol{R}_k(x) = \begin{pmatrix}
\frac{t_{\mu+1}-x}{t_{\mu+1}-t_{\mu+1-k}} & \frac{x-t_{\mu+1-k}}{t_{\mu+1}-t_{\mu+1-k}} & 0 & \cdots & 0 \\
0 & \frac{t_{\mu+2}-x}{t_{\mu+2}-t_{\mu+2-k}} & \frac{x-t_{\mu+2-k}}{t_{\mu+2}-t_{\mu+2-k}} & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & \frac{t_{\mu+k}-x}{t_{\mu+k}-t_{\mu}} & \frac{x-t_{\mu}}{t_{\mu+k}-t_{\mu}}
\end{pmatrix}. \tag{2.20}
$$

Then, for x in the interval $[t_\mu, t_{\mu+1})$, the $d+1$ B-splines $\{B_{j,d}\}_{j=\mu-d}^{\mu}$ of degree d that are nonzero on this interval can be written

$$
\boldsymbol{B}_d^T = \big(B_{\mu-d,d} \;\; B_{\mu-d+1,d} \;\; \ldots \;\; B_{\mu,d}\big) = \boldsymbol{R}_1(x)\boldsymbol{R}_2(x)\cdots\boldsymbol{R}_d(x). \tag{2.21}
$$

If $f = \sum_j c_j B_{j,d}$ is a spline in $\mathbb{S}_{d,\boldsymbol{t}}$, and x is restricted to the interval $[t_\mu, t_{\mu+1})$, then $f(x)$ is given by

$$
f(x) = \boldsymbol{R}_1(x)\boldsymbol{R}_2(x)\cdots\boldsymbol{R}_d(x)\boldsymbol{c}_d, \tag{2.22}
$$

where the vector $\boldsymbol{c}_d$ is given by $\boldsymbol{c}_d = (c_{\mu-d}, c_{\mu-d+1}, \ldots, c_\mu)^T$. The matrix $\boldsymbol{R}_k$ is called a B-spline matrix.

For $d = 0$ the usual convention of interpreting an empty product as 1 is assumed in equations (2.21) and (2.22).

Theorem 2.18 shows how one polynomial piece of splines and B-splines are built up, by multiplying and adding together (via matrix multiplications) certain linear polynomials. This representation is only an alternative way to write the recurrence relation (2.1), but the advantage is that all the recursive steps are captured in one equation. This will be convenient for developing the theory of splines in Section 3.1.2. The factorisation (2.22) will also be helpful for designing algorithms for computing $f(x)$. This is the theme of Section 2.4.

It should be emphasised that equation (2.21) is a representation of $d+1$ polynomials, namely the $d+1$ polynomials that make up the $d+1$ B-splines on the interval $[t_\mu, t_{\mu+1})$. This equation can therefore be written

$$\begin{pmatrix} B^\mu_{\mu-d,d}(x) & B^\mu_{\mu-d+1,d}(x) & \dots & B^\mu_{\mu,d}(x) \end{pmatrix} = \boldsymbol{R}^\mu_1(x)\boldsymbol{R}^\mu_2(x)\cdots\boldsymbol{R}^\mu_d(x),$$

see Lemma 2.6.

Likewise, equation (2.22) gives a representation of the polynomial f^μ that agrees with the spline f on the interval $[t_\mu, t_{\mu+1})$,

$$f^\mu(x) = \boldsymbol{R}_1(x)\boldsymbol{R}_2(x)\cdots\boldsymbol{R}_d(x)\boldsymbol{c}_d.$$

Once μ has been fixed we may let x take values outside the interval $[t_\mu, t_{\mu+1})$ in both these equations. In this way the B-spline pieces and the polynomial f^μ can be evaluated at any real number x. Figure 2.3 was produced in this way.

Example 2.19 (Matrix representation of a quadratic spline)**.** In Example 2.9 we considered the spline

$$s(x) = B_{2,2}(x) + B_{3,2}(x) - B_{4,2}(x) - \sqrt{2}B_{5,2}(x)$$

on the knot vector

$$\boldsymbol{t} = (t_j)_{j=1}^8 = (0, 0, 0, 1, 2, 3, 3, 3).$$

Let us use the matrix representation to determine this spline explicitly on each of the subintervals $[0, 1]$, $[1, 2]$, and $[2, 3]$. If $x \in [0, 1)$ then $t_3 \le x < t_4$ so $s(x)$ is determined by (2.22) with $\mu = 3$ and $d = 2$. To determine the matrices $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ we use the knots

$$(t_{\mu-1}, t_\mu, t_{\mu+1}, t_{\mu+2}) = (0, 0, 1, 2)$$

and the coefficients

$$(c_{\mu-2}, c_{\mu-1}, c_\mu) = (0, 1, 1).$$

Then equation (2.22) becomes

$$s(x) = \begin{pmatrix} 1-x, & x \end{pmatrix} \begin{pmatrix} 1-x & x & 0 \\ 0 & (2-x)/2 & x/2 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = x(2-x)$$

If $x \in [1, 2)$ then $t_4 \le x < t_5$ so $s(x)$ is determined by (2.22) with $\mu = 4$ and $d = 2$. To determine the matrices $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ in this case we use the knots

$$(t_{\mu-1}, t_\mu, t_{\mu+1}, t_{\mu+2}) = (0, 1, 2, 3)$$

and the coefficients

$$(c_{\mu-2}, c_{\mu-1}, c_\mu) = (1, 1, -1).$$

From this we find

$$s(x) = \frac{1}{2}\begin{pmatrix}2-x, & x-1\end{pmatrix}\begin{pmatrix}2-x & x & 0\\ 0 & 3-x & x-1\end{pmatrix}\begin{pmatrix}1\\1\\-1\end{pmatrix} = 2x - x^2.$$

For $x \in [2,3)$ we use $\mu = 5$, and on this interval $s(x)$ is given by

$$s(x) = \begin{pmatrix}3-x, & x-2\end{pmatrix}\begin{pmatrix}(3-x)/2 & (x-1)/2 & 0\\ 0 & 3-x & x-2\end{pmatrix}\begin{pmatrix}1\\-1\\-\sqrt{2}\end{pmatrix} = \big(2-x\big)\big(6-2\sqrt{2}-(2-\sqrt{2})x\big).$$

2.4 Algorithms for evaluating a spline

We originally introduced spline curves as the result of the geometric construction given in Algorithm 1.3 in Chapter 1. In this section we will relate this algorithm to the matrix representation of B-splines and develop an alternative algorithm for computing splines.

2.4.1 High level description

Recall from Theorem 2.18 that a spline f of degree d with knots $\boldsymbol{t}$ and B-spline coefficients $\boldsymbol{c}$ can be expressed as

$$f(x) = \boldsymbol{R}_1(x)\cdots\boldsymbol{R}_d(x)\boldsymbol{c}_d \tag{2.23}$$

for any x in the interval $[t_\mu, t_{\mu+1})$. Here $\boldsymbol{c}_d = (c_{\mu-d}, \ldots, c_\mu)$ denotes the B-spline coefficients that are active on this interval. To compute $f(x)$ from this representation we have two options: We can accumulate the matrix products from left to right or from right to left.

If we start from the right, the computations are

$$\boldsymbol{c}_{k-1} = \boldsymbol{R}_k\boldsymbol{c}_k, \qquad \text{for } k = d,\, d-1,\, \ldots,\, 1. \tag{2.24}$$

Upon completion of this we have $f(x) = \boldsymbol{c}_0$ (note that $\boldsymbol{c}_0$ is a vector of dimension 1, i.e., a scalar). We see that this algorithm amounts to post-multiplying each matrix $\boldsymbol{R}_k$ by a vector which in component form becomes

$$(\boldsymbol{R}_k(x)\boldsymbol{c}_k)_j = \frac{t_{j+k}-x}{t_{j+k}-t_j}c_{j-1,k} + \frac{x-t_j}{t_{j+k}-t_j}c_{j,k} \tag{2.25}$$

for $j = \mu-k+1, \ldots, \mu$. This we immediately recognise as Algorithm 1.3.

The alternative algorithm accumulates the matrix products in (2.23) from left to right. This is equivalent to building up the nonzero B-splines at x degree by degree until we have all the nonzero B-splines of degree d, then multiplying with the corresponding B-spline coefficients and summing. Computing the B-splines is accomplished by starting with $\boldsymbol{B}_0(x)^T = 1$ and then performing the multiplications

$$\boldsymbol{B}_k(x)^T = \boldsymbol{B}_{k-1}(x)^T\boldsymbol{R}_k(x), \qquad k = 1, \ldots, d.$$

The vector $\boldsymbol{B}_d(x)$ will contain the value of the nonzero B-splines of degree d at x,

$$\boldsymbol{B}_d(x) = \big(B_{\mu-d,d}(x), \ldots, B_{\mu,d}(x)\big)^T.$$

We can then multiply with the B-spline coefficients and add up.

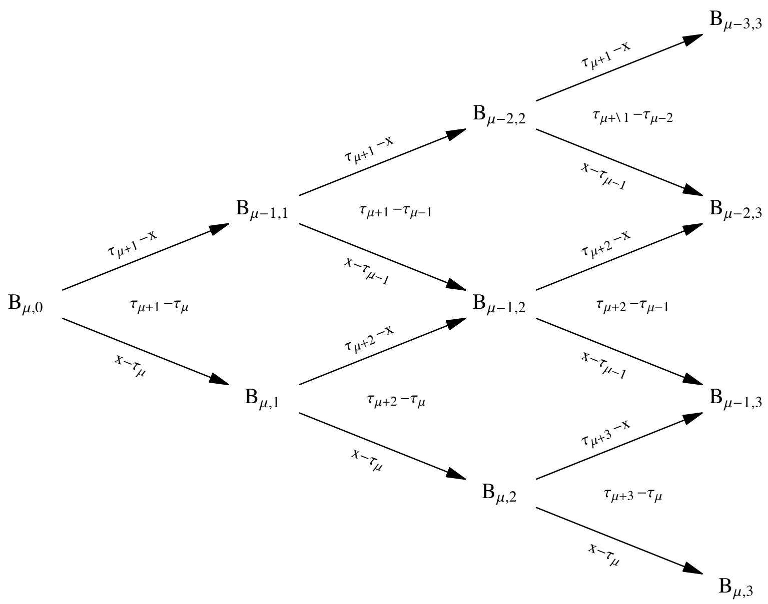


Figure 2.8. A triangular algorithm for computation of all the nonzero cubic B-splines at x.

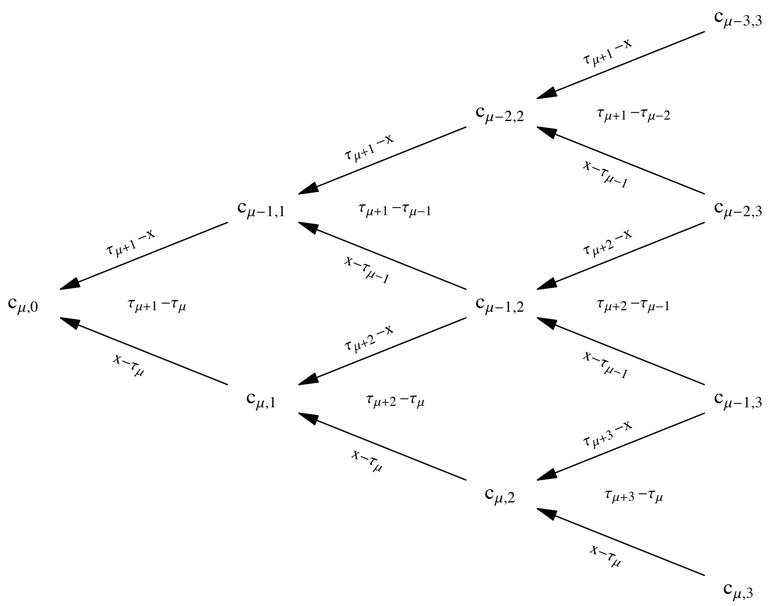


Figure 2.9. A triangular algorithm for computing the value of a cubic spline with B-spline coefficients $\boldsymbol{c}$ at $x \in [t_\mu, t_{\mu+1})$.

Algorithm 2.20 (L). *Let the polynomial degree d, the $2d$ knots $t_{\mu-d+1} \le t_\mu < t_{\mu+1} \le t_{\mu+d}$, the B-spline coefficients $\boldsymbol{c}_d^{(0)} = \boldsymbol{c}_d = (c_j)_{j=\mu-d}^{\mu}$ of a spline f, and a number x in $[t_\mu, t_{\mu+1})$ be given. After evaluation of the products*

$$\boldsymbol{c}_{k-1} = \boldsymbol{R}_k(x)\boldsymbol{c}_k, \qquad k = d, \ldots, 1,$$

the function value $f(x)$ is given by

$$f(x) = \boldsymbol{c}_0.$$

Algorithm 2.21 (R). *Let the polynomial degree d, the knots $t_{\mu-d+1} \le t_\mu < t_{\mu+1} \le t_{\mu+d}$ and a number x in $[t_\mu, t_{\mu+1})$ be given and set $\boldsymbol{B}_0 = 1$. After evaluation of the products*

$$\boldsymbol{B}_k(x)^T = \boldsymbol{B}_{k-1}(x)^T \boldsymbol{R}_k(x), \qquad k = 1, \ldots, d,$$

the vector $\boldsymbol{B}_d(x)$ will contain the value of the $d+1$ B-splines at x,

$$\boldsymbol{B}_d(x) = \big(B_{\mu-d,d}(x), \ldots, B_{\mu,d}(x)\big)^T.$$

These algorithms have a simple triangular structure, just like Algorithm 1.3, see figures 2.9–2.8. Figure 2.9 shows how the value of a cubic spline can be computed at a point x, while Figure 2.8 shows the computation of all the nonzero B-splines at a point.

In Algorithms 2.20 and 2.21 it is assumed that there are $2d$ knots to the left and right of x. This may not always be the case, especially near the ends of the knot vector, unless it is $d+1$-regular. Exercise 19 discusses evaluation in such a case.

2.4.2 More detailed algorithms

Algorithms 2.20 and 2.21 are high level algorithms. Although they may be implemented directly by forming the matrices $\{\boldsymbol{R}_k\}_{k=1}^d$, it is usually better to polish the algorithms a bit more. In this section we will discuss Algorithm 2.21 in more detail. For more details on Algorithm 2.20, we refer to Algorithm 1.3 in Chapter 1 and exercise 18 below.

Algorithm 2.21 corresponds to pre-multiplying each matrix $\boldsymbol{R}_k$ by a row vector. In component form this can be written

$$(\boldsymbol{B}_{k-1}(x))^T \boldsymbol{R}_k(x))_j = \frac{x - t_j}{t_{j+k} - t_j} B_{j,k-1}(x) + \frac{t_{j+1+k} - x}{t_{j+1+k} - t_{j+1}} B_{j+1,k-1}(x) \tag{2.26}$$

for $j = \mu - k, \ldots, \mu$. This is of course just the recurrence relation for B-splines. Here it should be noted that $B_{\mu-k,k-1}(x) = B_{\mu+1,k-1}(x) = 0$ when $x \in [t_\mu, t_{\mu+1})$. For $j = \mu - k$, the first term on the right in (2.26) is therefore zero, and similarly, for $j = \mu$, the last term in (2.26) is zero.

We are going to give two more detailed versions of Algorithm 2.21. In the first one, we make use of vector operations. This version would be suitable for a language like Matlab or Mathematica where for-loops are relatively slow, but the built-in vector operations are fast.

We assume that the elementary arithmetic operations may be applied to vectors of the same size. For example, the vector operation $\boldsymbol{a}/\boldsymbol{b}$ would produce a vector of the same length as $\boldsymbol{a}$ and $\boldsymbol{b}$, with entry i equal to a_i/b_i. We can also combine a scalar and a vector as in $x + \boldsymbol{a}$; then the first operand is converted to a vector of the same length as $\boldsymbol{a}$ by duplicating x the correct number of times.

We will need two more vector operations which we denote $\boldsymbol{a}_{+l}$ and $\boldsymbol{a}_{+f}$. The first denotes the vector obtained by appending a zero to the end of $\boldsymbol{a}$, while $\boldsymbol{a}_{+f}$ denotes the result of prepending a zero element at the beginning of $\boldsymbol{a}$. In Matlab syntax this would be written as $\boldsymbol{a}_{+l} = [\boldsymbol{a}, 0]$ and $\boldsymbol{a}_{+f} = [0, \boldsymbol{a}]$. We leave it to the reader to verify that Algorithm 2.21 can then be written in the following more explicit form. A vector version of Algorithm 2.20 can be found in exercise 18.

Algorithm 2.22 (R—vector version)**.** *Let the polynomial degree d, the knots $t_{\mu-d+1} \le t_\mu < t_{\mu+1} \le t_{\mu+d}$ and a number x in $[t_\mu, t_{\mu+1})$ be given. After evaluation of*

1. $\boldsymbol{b} = 1$;
2. For $r = 1, 2, \ldots, d$
    1. $\boldsymbol{t}1 = (t_{\mu-r+1}, \ldots, t_\mu)$;
    2. $\boldsymbol{t}2 = (t_{\mu+1}, \ldots, t_{\mu+r})$;
    3. $\boldsymbol{\omega} = (x - \boldsymbol{t}1)/(\boldsymbol{t}2 - \boldsymbol{t}1)$;
    4. $\boldsymbol{b} = \big((1-\boldsymbol{\omega}) * \boldsymbol{b}\big)_{+l} + \big(\boldsymbol{\omega} * \boldsymbol{b}\big)_{+f}$;

the vector $\boldsymbol{b}$ will contain the value of the $d+1$ B-splines at x,

$$\boldsymbol{b} = \big(B_{\mu-d,d}(x), \ldots, B_{\mu,d}(x)\big)^T.$$

When programming in a traditional procedural programming language, the vector operations will usually have to be replaced by for-loops. This can be accomplished as follows.

Algorithm 2.23 (R—scalar version)**.** *Let the polynomial degree d, the knots $t_{\mu-d+1} \le t_\mu < t_{\mu+1} \le t_{\mu+d}$ and a number x in $[t_\mu, t_{\mu+1})$ be given. After evaluation of*

1. $b_{d+1} = 1$; $b_i = 0$, $i = 1, \ldots, d$;
2. For $r = 1, 2, \ldots, d$
    1. $k = \mu - r + 1$;
    2. $\omega_2 = (t_{k+r} - x)/(t_{k+r} - t_k)$;
    3. $b_{d-r} = \omega_2\, b_{d-r+1}$;
    4. For $i = d-r+1, d-r+2, \ldots, d-1$
        1. $k = k+1$;
        2. $\omega_1 = \omega_2$;
        3. $\omega_2 = (t_{k+r} - x)/(t_{k+r} - t_k)$;
        4. $b_i = (1-\omega_1)\, b_i + \omega_2\, b_{i+1}$;
    5. $b_d = (1-\omega_2)\, b_d$

the vector $\boldsymbol{b}$ will contain the value of the $d+1$ B-splines at x,

$$\boldsymbol{b} = \big(B_{\mu-d,d}(x), \ldots, B_{\mu,d}(x)\big)^T.$$

Exercises for Chapter 2

2.1 Show that

$$B(x \mid 0,3,4,6) = \frac{1}{12}x^2 B(x \mid 0,3) + \frac{1}{12}(-7x^2 + 48x - 72)B(x \mid 3,4) + \frac{1}{6}(6-x)^2 B(x \mid 4,6).$$

2.2 Find the individual polynomial pieces of the following cubic B-splines and discuss smoothness properties at knots

a) $B(x \mid 0,0,0,0,1)$ and $B(x \mid 0,1,1,1,1)$

b) $B(x \mid 0,1,1,1,2)$

2.3 Show that the B-spline $B_{j,d}$ satisfies properties 1–3 of Lemma 2.6.

2.4 Show that $B_{j,d}$ is a piecewise polynomial by establishing equation 2.14. Use induction on the degree d.

2.5 In this exercise we are going to establish some properties of the Bernstein polynomials.

a) Prove the differentiation formula

$$Db_{j,d}(x) = d(b_{j-1,d-1}(x) - B_{j,d-1}(x)).$$

b) Show that the Bernstein basis function $b_{j,d}(x)$ has a maximum at $x = j/d$, and that this is the only maximum.

c) Show that

$$\int_0^1 B_{j,d}(x)dx = 1/(d+1).$$

2.6 a) When a B-spline is evaluated at one of its knots can be simplified according to the formula

$$B(t_i \mid t_j, \dots, t_{j+1+d}) = B(t_i \mid t_j, \dots, t_{i-1}, t_{i+1}, \dots, t_{j+1+d}) \tag{2.27}$$

which is valid for $i = j, j+1, \dots, j+1+d$. Prove this by induction on the degree d.

b) Use the formula in (2.27) to compute the following values of a quadratic B-spline at the interior knots:

$$B_{j,2}(t_{j+1}) = \frac{t_{j+1} - t_j}{t_{j+2} - t_j}, \quad B_{j,2}(t_{j+2}) = \frac{t_{j+3} - t_{j+2}}{t_{j+3} - t_{j+1}}. \tag{2.28}$$

c) Prove property (5) of Lemma 2.6.

2.7 Prove the following formula using (2.4) and (2.11)

$$B(x \mid a, \overbrace{b, \dots, b}^{d}, c) = \frac{(x-a)^d}{(b-a)^d} B(x \mid a,b) + \frac{(c-x)^d}{(c-b)^d} B(x \mid b,c).$$

Show that this function is continuous at all real numbers.

2.8 Prove the following formulas by induction on d.

$$\begin{aligned}
B(x \mid \overbrace{a,\ldots,a}^{d}, b, c) &= \frac{x-a}{b-a}\sum_{i=0}^{d-1}\frac{(c-x)^i(b-x)^{d-1-i}}{(c-a)^i(b-a)^{d-1-i}}B(x \mid a,b) \\
&\quad + \frac{(c-x)^d}{(c-a)^{d-1}(c-b)}B(x \mid b,c), \\
B(x \mid a, b, \overbrace{c,\ldots,c}^{d}) &= \frac{(x-a)^d}{(c-a)^{d-1}(b-a)}B(x \mid a,b) \\
&\quad + \frac{c-x}{c-b}\sum_{i=0}^{d-1}\frac{(x-a)^i(x-b)^{d-1-i}}{(c-a)^i(c-b)^{d-i}}B(x \mid b,c).
\end{aligned}$$

2.9 When the knots are simple we can give explicit formulas for the B-splines.

a) Show by induction that if $t_j < \cdots < t_{j+1+d}$ then

$$B_{j,d}(x) = (t_{j+1+d} - t_j)\sum_{i=j}^{j+1+d}\frac{(x-t_i)_+^d}{\prod_{\substack{k=j\\k\neq i}}^{j+1+d}(t_k - t_i)}$$

where

$$(x-t_i)_+^d = \begin{cases}(x-t_i)^d, & \text{if } x \geq t_i;\\ 0, & \text{otherwise.}\end{cases}$$

b) Show that $B_{j,d}$ can also be written

$$B_{j,d}(x) = (t_{j+1+d} - t_j)\sum_{i=j}^{j+1+d}\frac{(t_i-x)_+^d}{\prod_{\substack{k=j\\k\neq i}}^{j+1+d}(t_i - t_k)}$$

but now the $(\cdot)_+$-function must be defined by

$$(t_i-x)_+^d = \begin{cases}(t_i-x)^d, & \text{if } t_i > x;\\ 0, & \text{otherwise.}\end{cases}$$

2.10 Write down the matrix $\boldsymbol{R}_3(x)$ for $\mu = 4$ in the case of uniform splines ($t_j = j$ for all j). Do the same for the Bernstein basis ($\boldsymbol{t} = (0,0,0,0,1,1,1,1)$).

2.11 Given a knot vector $\boldsymbol{t} = (t_j)_{j=1}^{n+d+1}$ and a real number x with $x \in [t_1, t_{n+d+1})$, write a procedure for determining the index μ such that $t_\mu \leq x < t_{\mu+1}$. A call to this routine is always needed before Algorithms 2.20 and 2.21 are run. By letting μ be an input parameter as well as an output parameter you can minimise the searching for example during plotting when the routine is called with many values of x in the same knot interval.

2.12 Implement Algorithm 2.21 in your favourite programming language.

2.13 Implement Algorithm 2.20 in your favourite programming language.

2.14 Count the number of operations (additions, multiplications, divisions) involved in Algorithm 2.20.

2.15 Count the number of operations (additions, multiplications, divisions) involved in Algorithm 2.21.

2.16 Write a program that plots the cubic B-spline $B(x \mid 0, 1, 3, 5, 6)$ and its polynomial pieces. Present the results as in Figure 2.3.

2.17 a) What is computed by Algorithm 2.20 if x does not belong to the interval $[t_\mu, t_{\mu+1})$?

b) Repeat (b) for Algorithm 2.21.

2.18 Algorithm 2.22 gives a vector version of Algorithm 2.21 for computing the nonzero B-splines at a point x. Below is a similar vector version of Algorithm 2.20 for computing the value of a spline at x. Verify that the algorithm is correct and compare it with Algorithm 2.22.

Let $f = \sum_i c_i B_{i,d,\boldsymbol{t}}$ be a spline in $\mathbb{S}_{d,\boldsymbol{t}}$, and let x be a real number in the interval $[t_\mu, t_{\mu+1})$. Then $f(x)$ can be computed as follows:

1. $\boldsymbol{c} = (c_{\mu-k+1}, \ldots, c_\mu)$;
2. For $r = k-1,\ k-2,\ \ldots,\ 1$
   1. $\boldsymbol{t}1 = (t_{\mu-r+1}, \ldots, t_\mu)$;
   2. $\boldsymbol{t}2 = (t_{\mu+1}, \ldots, t_{\mu+r})$;
   3. $\boldsymbol{\omega} = (x - \boldsymbol{t}1)/(\boldsymbol{t}2 - \boldsymbol{t}1)$;
   4. $\boldsymbol{c} = (1 - \boldsymbol{\omega}) * \boldsymbol{c}_{-l} + \boldsymbol{\omega} * \boldsymbol{c}_{-f}$;

After these statements $\boldsymbol{c}$ will be a vector of length 1 that contains the number $f(x)$. Here the notation $\boldsymbol{c}_{-l}$ and $\boldsymbol{c}_{-f}$ denote the vectors obtained by dropping the last, respectively the first, entry from $\boldsymbol{c}$.

2.19 Suppose that $d = 3$ and that the knot vector is given by

$$\hat{\boldsymbol{t}} = (t_j)_{j=1}^5 = (0, 1, 2, 3, 4).$$

With this knot vector we can only associate one cubic B-spline, namely $B_{1,3}$. Therefore, if we are to compute $B_{1,3}(x)$ for some x in $(0, 4)$, none of the algorithms of this section apply. Define the augmented knot vector $\boldsymbol{t}$ by

$$\boldsymbol{t} = (-1, -1, -1, -1, 0, 1, 2, 3, 4, 5, 5, 5, 5).$$

Explain how this knot vector can be exploited to compute the B-spline $B_{1,3}(x)$ by Algorithms 2.20 or 2.21.