Knot insertion

Michael S. Floater

March 2, 2022

In these notes we consider how to refine a spline function by adding knots. To understand this we focus on what happens to a single B-spline when we add one knot.

1 Refining a spline by inserting a knot

Let us suppose that we start with a spline function

$$s(x) = \sum_{j=1}^{n} c_j B_{j,d,\mathbf{t}}(x), \qquad x \in [t_{d+1}, t_{n+1}], \tag{1}$$

where, as usual, $B_{j,d,\mathbf{t}} = B_{j,d}$ is the j-th B-spline and $d \geq 0$ and $n \geq 1$ and $\mathbf{t} = (t_1, t_2, \ldots, t_{n+d+1})$ is the corresponding non-decreasing knot vector. We use the notation $B_{j,d,\mathbf{t}}$ here to indicate that $B_{j,d}$ depends on the knot vector $\mathbf{t}$.

Suppose we now add a new knot z to some knot interval $[t_\mu, t_{\mu+1})$, where $d+1 \leq \mu \leq n$, to form the refined knot vector

$$\boldsymbol{\tau} = (t_1, t_2, \ldots, t_\mu, z, t_{\mu+1}, \ldots, t_{n+d+1}).$$

We then ask, how can we represent the same spline function s with respect to the refined knot vector $\boldsymbol{\tau}$? What we need to do is to find coefficients b_j such that

$$s(x) = \sum_{j=1}^{n+1} b_j B_{j,d,\boldsymbol{\tau}}(x). \tag{2}$$

To do this we need to express each B-spline $B_{j,d,\mathbf{t}}$ as a linear combination of the $B_{j,d,\boldsymbol{\tau}}$. We need only consider $B_{j,d,\mathbf{t}}$ where $j = \mu - d, \ldots, \mu$, since the remaining B-splines on $\mathbf{t}$ are not affected by the insertion of z.

Suppose then that $\mu - d \leq j \leq \mu$, and consider the support $[t_j, t_{j+d+1}]$ of $B_{j,d,\mathbf{t}}$. Only two of the B-splines over the knot vector $\boldsymbol{\tau}$ have support contained in $[t_j, t_{j+d+1}]$, namely, $B_{j,d,\boldsymbol{\tau}}$ and $B_{j+1,d,\boldsymbol{\tau}}$. Thus, we would expect to be able to express $B_{j,d,\mathbf{t}}$ as a linear combination of $B_{j,d,\boldsymbol{\tau}}$ and $B_{j+1,d,\boldsymbol{\tau}}$.

2 Refining a B-spline by inserting a knot

We will derive a formula for the refinement of a single B-spline when we add one knot to the interior of its support. To this end, we will first derive a corresponding refinement formula for divided differences.

Lemma 1 *Consider a divided difference $[x_0, x_1, \ldots, x_k]f$, in which x_0 and x_k are distinct, and let $z \in \mathbb{R}$. Then*

$$[x_0, \ldots, x_k]f = \frac{z - x_0}{x_k - x_0}[x_0, \ldots, x_{k-1}, z]f + \frac{x_k - z}{x_k - x_0}[x_1, \ldots, x_k, z]f. \tag{3}$$

We note here that we are implicitly assuming that if any of the points $x_0, \ldots, x_k$ and z are not distinct, f has sufficiently many derivatives at the multiple points for these divided differences to be well-defined.

Proof. We have

$$\begin{aligned}
&[x_1, \ldots, x_k]f - [x_0, \ldots, x_{k-1}]f = \\
&([x_1, \ldots, x_{k-1}, z]f - [x_0, \ldots, x_{k-1}]f) + ([x_1, \ldots, x_k]f - [x_1, \ldots, x_{k-1}, z]f) \\
&= (z - x_0)[x_0, \ldots, x_{k-1}, z]f + (x_k - z)[x_1, \ldots, x_k, z]f,
\end{aligned}$$

and dividing by $x_k - x_0$ gives (3). □

By this lemma we now obtain a formula for the refinement of a B-spline.

Theorem 1 *Suppose $t_j < t_{j+d+1}$, and let z be any point in $[t_j, t_{j+d+1}]$. Then*

$$B_{j,d,\mathbf{t}}(x) = \begin{cases} \dfrac{z - t_j}{t_{j+d} - t_j} B_{j,d,\boldsymbol{\tau}}(x) + B_{j+1,d,\boldsymbol{\tau}}(x), & z \leq t_{j+1}; \\ \dfrac{z - t_j}{t_{j+d} - t_j} B_{j,d,\boldsymbol{\tau}}(x) + \dfrac{t_{j+d+1} - z}{t_{j+d+1} - t_{j+1}} B_{j+1,d,\boldsymbol{\tau}}(x), & t_{j+1} < z < t_{j+d}; \\ B_{j,d,\boldsymbol{\tau}}(x) + \dfrac{t_{j+d+1} - z}{t_{j+d+1} - t_{j+1}} B_{j+1,d,\boldsymbol{\tau}}(x), & z \geq t_{j+d}. \end{cases} \tag{4}$$

Proof. By Lemma 1,

$$
\begin{aligned}
(t_{j+d+1}-t_j)[t_j,\ldots,t_{j+d+1}]f &= (z-t_j)[t_j,\ldots,t_{j+d},z]f \\
&\quad + (t_{j+d+1}-z)[t_{j+1},\ldots,t_{j+d+1},z]f,
\end{aligned}
$$

and applying this equation to the function $f(y)=(\cdot-x)_+^d$ leads to (4) □

3 Algorithm for refining a spline

We now return to the problem posed in Section 1. Using Theorem 1, we can find the coefficients b_j in (2). The algorithm for computing the b_j is known as Boehm's algorithm.

Theorem 2 *The coefficients b_j in (2) are*

$$
b_j = \begin{cases} c_j & \text{if } 1 \le j \le \mu-d; \\ \dfrac{t_{j+d}-z}{t_{j+d}-t_j}c_{j-1} + \dfrac{z-t_j}{t_{j+d}-t_j}c_j; & \text{if } \mu-d+1 \le j \le \mu, \\ c_{j-1} & \text{if } \mu+1 \le j \le n+1. \end{cases} \tag{5}
$$

Proof. We convert the sum in (1) into the form (2) by expressing each B-spline $B_{j,d,\mathbf{t}}$ as a linear combination of the B-splines $B_{j,d,\boldsymbol{\tau}}$. We have $B_{j,d,\mathbf{t}} = B_{j,d,\boldsymbol{\tau}}$ for $j=1,\ldots,\mu-d-1$, since z is not in the support of these B-splines, and so

$$
\sum_{j=1}^{\mu-d-1} c_j B_{j,d,\mathbf{t}} = \sum_{j=1}^{\mu-d-1} c_j B_{j,d,\boldsymbol{\tau}}.
$$

Similarly, $B_{j,d,\mathbf{t}} = B_{j+1,d,\boldsymbol{\tau}}$ for $j=\mu+1,\ldots,n$, and so

$$
\sum_{j=\mu+1}^{n} c_j B_{j,d,\mathbf{t}} = \sum_{j=\mu+1}^{n} c_j B_{j+1,d,\boldsymbol{\tau}} = \sum_{j=\mu+2}^{n+1} c_{j-1} B_{j,d,\boldsymbol{\tau}}.
$$

To treat the remaining part of the sum in (1) we use Theorem 2, and find

$$\begin{aligned}
\sum_{j=\mu-d}^{\mu} c_j B_{j,d,\mathbf{t}} &= c_{\mu-d}\left(B_{\mu-d,d,\boldsymbol{\tau}}(x) + \frac{t_{\mu+1}-z}{t_{\mu+1}-t_{\mu-d+1}} B_{j+1,d,\boldsymbol{\tau}}(x)\right) \\
&+ \sum_{j=\mu-d+1}^{\mu-1} c_j \left(\frac{z-t_j}{t_{j+d}-t_j} B_{j,d,\boldsymbol{\tau}}(x) + \frac{t_{j+d+1}-z}{t_{j+d+1}-t_{j+1}} B_{j+1,d,\boldsymbol{\tau}}(x)\right) \\
&+ c_\mu \left(\frac{z-t_\mu}{t_{\mu+d}-t_\mu} B_{\mu,d,\boldsymbol{\tau}}(x) + B_{\mu+1,d,\boldsymbol{\tau}}(x)\right) \\
&= c_{\mu-d} B_{\mu-d,d,\boldsymbol{\tau}}(x) \\
&+ \sum_{j=\mu-d+1}^{\mu} \left(\frac{t_{j+d}-z}{t_{j+d}-t_j} c_{j-1} + \frac{z-t_j}{t_{j+d}-t_j} c_j\right) B_{j,d,\boldsymbol{\tau}}(x) \\
&+ c_\mu B_{\mu+1,d,\boldsymbol{\tau}}(x),
\end{aligned}$$

which gives us (5). □