

Lecture 13: Splines

Spline fitting, our next topic in interpolation theory, plays an essential role in engineering design. As in the last lecture, we strive to interpolate data using low-degree polynomials between consecutive grid points. The piecewise linear functions of Section 2.7.1 were simple, but suffered from unsightly kinks at each interpolation point, reflecting a discontinuity in the first derivative. By increasing the degree of the polynomial used to model f on each subinterval, we can obtain smoother functions.

2.7.2. Piecewise Cubic Hermite Interpolation.

To remove the discontinuities in the first derivative of the piecewise linear interpolant, we begin by modeling our data with cubic polynomials over each interval $[x_j, x_{j+1}]$. Each such cubic has four free parameters (since \mathcal{P}_3 is a vector space of dimension 4); we require these polynomials to interpolate both f and its first derivative:

$$\begin{aligned} s_j(x_{j-1}) &= f(x_{j-1}), & j &= 1, \dots, n; \\ s_j(x_j) &= f(x_j), & j &= 1, \dots, n; \\ s'_j(x_{j-1}) &= f'(x_{j-1}), & j &= 1, \dots, n; \\ s'_j(x_j) &= f'(x_j), & j &= 1, \dots, n. \end{aligned}$$

To satisfy these conditions, take s_j to be the Hermite interpolant to the data $(x_{j-1}, f(x_{j-1}), f'(x_{j-1}))$ and $(x_j, f(x_j), f'(x_j))$. The resulting function, $S(x) := s_j(x)$ for $x \in [x_{j-1}, x_j]$, will always have a continuous derivative, $S \in C^1[x_0, x_n]$, but generally $S \notin C^2[x_0, x_n]$ due to discontinuities in the second derivative at the interpolation points.

In many applications, we don't have specific values for $S'(x_j) = f'(x_j)$ in mind; we just want the function $S(x)$ to be as *smooth* as possible. That motivates the main topic of this lecture, splines.

2.8. Splines.

Rather than setting $S'(x_j)$ to a particular value, suppose we simply require S' to be continuous throughout $[x_0, x_n]$. This added freedom allows us to impose a further condition: require S'' to be continuous on $[x_0, x_n]$, too. The polynomials we construct are called *cubic splines*. In spline parlance, the interpolation points $\{x_j\}_{j=0}^n$ are called *knots*.[†]

These cubic spline requirements can be written as:

$$\begin{aligned} s_j(x_{j-1}) &= f(x_{j-1}), & j &= 1, \dots, n; \\ s_j(x_j) &= f(x_j), & j &= 1, \dots, n; \\ s'_j(x_j) &= s'_{j+1}(x_j), & j &= 1, \dots, n-1; \\ s''_j(x_j) &= s''_{j+1}(x_j), & j &= 1, \dots, n-1. \end{aligned}$$

Compare these requirements to those imposed by piecewise cubic Hermite interpolation. Add up all these new requirements, and notice that we impose $2n + 2(n-1) = 4n - 2$ conditions on a total

[†]Apparently in the shipbuilding and aircraft industry where wooden splines were used in the pre-computer era, these knots went by the more colorful names of *rats*, *dogs*, or *ducks*. See the brief "History of Splines" note by James Epperson in the 19 July 1998 NA Digest, linked from the class website. For a beautiful derivation of cubic splines from Euler's beam equation—that is, from the original physical situation where splines were thin pieces of wood running through 'rats,' see Gilbert Strang, *Introduction to Applied Mathematics*, Wellesley Cambridge Press, 1986.

of n cubic polynomials, which together have $4n$ degrees of freedom. Thus, there will be infinitely many choices for the function $S(x)$. There are several canonical choices for the two extra conditions that make S uniquely defined:

- *complete* splines specify values for $S'(x_0)$ and $S'(x_n)$.
- *natural* splines require $S''(x_0) = S''(x_n) = 0$.
- *not-a-knot* splines require S''' to be continuous at x_1 and x_{n-1} .[‡]

Natural cubic splines are the most common choice, for they can be shown, in a precise sense, to minimize curvature over all the other possible splines.[§] They also model the physical origin of splines, where beams of wood extend straight (i.e., zero second derivative) beyond the first and final ‘rats.’

Whatever we decide for the two additional conditions, we arrive at a system of $4n$ equations (the various constraints) in $4n$ unknowns (the cubic polynomial coefficients). These equations can be set up as a system involving a tridiagonal coefficient matrix (zero everywhere except for the main diagonal and the first super- and sub-diagonals); we will see later in the semester that systems with this structure can be efficiently solved via Gaussian elimination. We could derive this linear system by directly enforcing the continuity conditions on the cubic polynomial that we have just described. (Try it!) However, we will prefer a more general approach that expresses the spline function $S(x)$ as the linear combination of special basis functions, which themselves are splines.

2.8.1. B-Splines.

Throughout our discussion of standard polynomial interpolation, we viewed \mathcal{P}_n as a linear space of dimension $n + 1$, and then expressed the unique interpolating polynomial in several different bases (monomial, Newton, Lagrange). The most elegant way to develop spline functions uses the same approach. A set of *basis splines*, depending only on the location of the knots and the degree of the approximating piecewise polynomials (cubics, discussed above, are a special case), can be developed in a convenient, numerically stable manner. For example, each cubic basis spline, or *B-spline*, is a continuous piecewise-cubic function with continuous first and second derivatives. Thus any linear combination of such B-splines will inherit the same continuity properties. The coefficients in the linear combination are chosen to obey the specified interpolation conditions.

B-splines are built up recursively from constant B-splines. Though we are interpolating data at $n + 1$ knots x_0, \dots, x_n , to derive B-splines we need extra nodes outside $[x_0, x_n]$ as scaffolding upon which to build our basis. Thus, add knots on either end of x_0 and x_n :

$$\cdots < x_{-2} < x_{-1} < x_0 < x_1 < \cdots < x_n < x_{n+1} < \cdots.$$

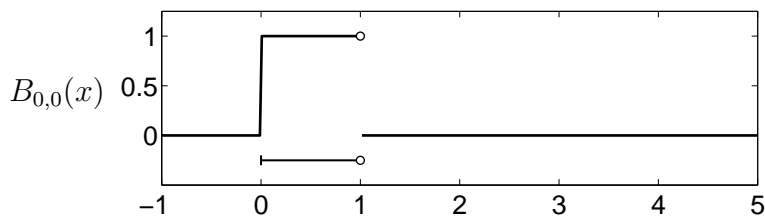
Given these knots, we can define the constant B-splines:

$$B_{j,0}(x) = \begin{cases} 1 & x \in [x_j, x_{j+1}); \\ 0 & \text{otherwise.} \end{cases}$$

The following plot shows the basis function $B_{0,0}$ for the knots $x_j = j$. Note, in particular, that $B_{j,0}(x_{j+1}) = 0$. The line drawn beneath the spline marks the *support* of the spline, that is, the values of x for which $B_{0,0}(x) \neq 0$.

[‡]Since the third derivative of a cubic is a constant, this requirement amounts to forcing $s_1 = s_2$ and $s_{n-1} = s_n$. Hence, while $S(x)$ interpolates the data at x_2 and x_{n-1} , the derivative continuity requirements are automatic at those knots; hence the name “not-a-knot”.

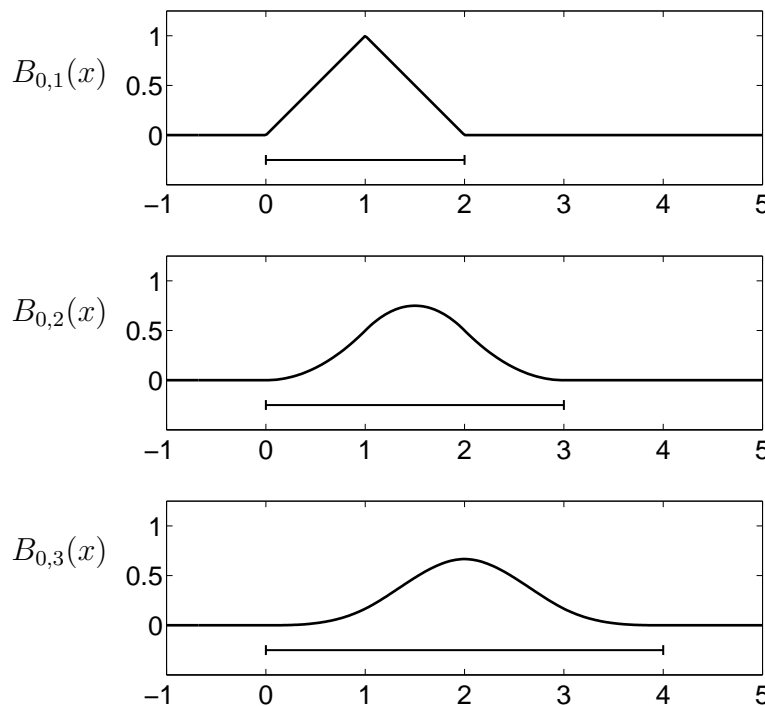
[§]See Süli and Mayers, *An Introduction to Numerical Analysis*, p. 300.



From these degree-0 B-splines, we can manufacture B-splines of higher degree via the recurrence

$$B_{j,k}(x) = \left(\frac{x - x_j}{x_{j+k} - x_j} \right) B_{j,k-1}(x) + \left(\frac{x_{j+k+1} - x}{x_{j+k+1} - x_{j+1}} \right) B_{j+1,k-1}(x). \quad (13.1)$$

While not immediately obvious from the formula, this construction ensures that $B_{j,k}$ will have one more continuous derivative than $B_{j,k-1}$ does. Thus, while $B_{j,0}$ is discontinuous (see previous plot), $B_{j,1}$ is continuous, $B_{j,2} \in C^1(\mathbb{R})$, and $B_{j,3} \in C^2(\mathbb{R})$. One can see this in the three plots below, where again $x_j = j$. As the degree increases, the B-spline $B_{j,k}$ becomes increasingly smooth. Smooth is good, but it has a consequence: the *support* of $B_{j,k}$ gets larger as we increase k . This, as we will see, has implications on the number of nonzero entries in the linear system we must ultimately solve to find the expansion of the desired spline in the B-spline basis.



From these plots and the recurrence defining $B_{j,k}$, one can deduce several important properties:

- $B_{j,k} \in C^{k-1}(\mathbb{R})$ (continuity);
- $B_{j,k}(x) = 0$ if $x \notin (x_j, x_{j+k+1})$ (compact support);
- $B_{j,k}(x) > 0$ for $x \in (x_j, x_{j+k+1})$ (positivity).

Finally, we are prepared to write down a formula for the spline that interpolates $\{(x_j, f_j)\}_{j=0}^n$ as a linear combination of the basis splines we have just constructed. Let $S_k(x)$ denote the spline

consisting of piecewise polynomials in \mathcal{P}_k . In particular, S_k must obey the following properties:

- $S_k(x_j) = f_j$ for $j = 0, \dots, n$.
- $S_k \in C^{k-1}[x_0, x_n]$ for $k \geq 1$.

The beauty B-splines is that the second of these properties is automatically inherited from the B-splines themselves. (Any linear combination of $C^{k-1}(\mathbb{R})$ functions must itself be a $C^{k-1}(\mathbb{R})$ function.) The interpolation conditions give $n+1$ equations that constrain the unknown coefficients $c_{j,k}$ in the expansion of S_k :

$$S_k(x_j) = \sum_{j=-\infty}^{\infty} c_{j,k} B_{j,k}(x_j).$$

The compact support of the B-splines immediately suggest that we set most of the $c_{j,k}$ coefficients to zero, giving S_k as a finite sum. For $k \geq 1$, we are left with $n+k$ nontrivial $c_{j,k}$ variables to be determined from $n+1$ interpolation conditions. Thus for quadratic and higher degree splines, we require extra conditions to get a unique S_k . To illustrate how the spline S_k is ultimately arrived at, we walk through several cases slowly.

Constant splines, $k = 0$. In this simple case, the basis functions are piecewise constants, and so the spline $S_0(x)$ will itself be piecewise constant (hence discontinuous, in general). The coefficients $c_{j,0}$ in the expansion

$$S_0(x) = \sum_{j=-\infty}^{\infty} c_{j,0} B_{j,0}(x)$$

are completely determined by the interpolation requirement: $S_0(x_j) = f_j$ for $j = 0, \dots, n$. Since $B_{j,0}(x_\ell) = 0$ if $j \neq \ell$, and $B_{\ell,0}(x_\ell) = 1$ (recall the plot of $B_{0,0}(x)$ shown earlier),

$$S_0(x_\ell) = c_{\ell,0} B_{\ell,0}(x_\ell) = c_{\ell,0}.$$

The interpolation condition $S_0(x_\ell) = f_\ell$ implies $c_{\ell,0} = f_\ell$. Since we do not care what value the spline takes outside $[x_0, x_n]$, we set $c_{j,0} = 0$ for $j < 0$ and $j > n$:

$$S_0(x) = \sum_{j=0}^n f_j B_{j,0}(x).$$

Linear splines, $k = 1$. Linear splines are similarly simple to construct. The support of $B_{j,1}(x)$ is (x_j, x_{j+2}) . That is, $B_{j,1}(x_\ell) = 0$ if $\ell \neq j+1$. Substituting x_j into the recursion (13.1) for $B_{j,1}(x)$ gives $B_{j,1}(x_{j+1}) = 1$, as apparent from the previous plot of $B_{0,1}(x)$. Thus

$$f_\ell = S_1(x_\ell) = \sum_{j=-\infty}^{\infty} c_{j,1} B_{j,1}(x_\ell) = c_{\ell-1,1}.$$

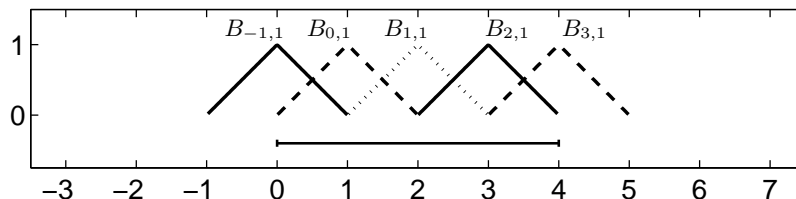
Again ignoring all splines that are zero throughout $[x_0, x_n]$, we have

$$S_1(x) = \sum_{j=-1}^{n-1} f_{j+1} B_{j,1}(x).$$

It is instructive now to look at an example.

j	0	1	2	3	4
x_j	0	1	2	3	4
f_j	1	3	2	-1	1

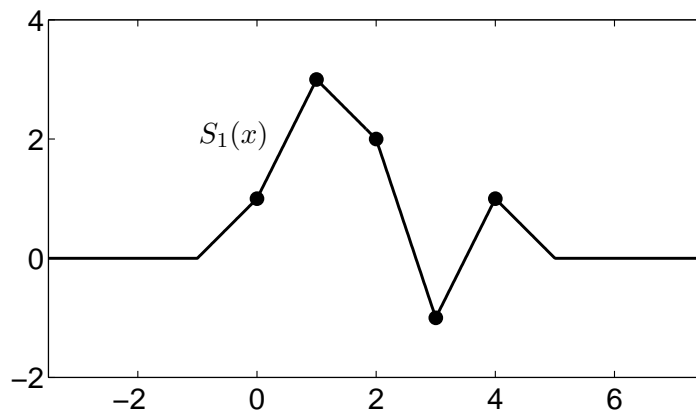
The desired linear spline S_1 is a linear combination of the five B-splines $B_{-1,1}$, $B_{0,1}$, $B_{1,1}$, $B_{2,1}$, and $B_{3,1}$, shown in the plot below.



More explicitly:

$$\begin{aligned}
 S_1(x) &= f_0 B_{-1,1}(x) + f_1 B_{0,1}(x) + f_2 B_{1,1}(x) + f_3 B_{2,1}(x) + f_4 B_{3,1}(x) \\
 &= B_{-1,1}(x) + 3B_{0,1}(x) + 2B_{1,1}(x) - B_{2,1}(x) + B_{3,1}(x).
 \end{aligned}$$

The spline S_1 is shown in the plot below. Note that linear splines are simply C^0 functions that interpolate a given data set—between the knots, they are identical to the piecewise linear functions constructed in §2.7.1. Note that $S_1(x) = 0$ for all $x \notin (x_{-1}, x_{n+1})$. This is a general feature of splines: Outside the range of interpolation, $S_k(x)$ goes to zero as quickly as possible for a given set of knots while still maintaining the specified continuity.

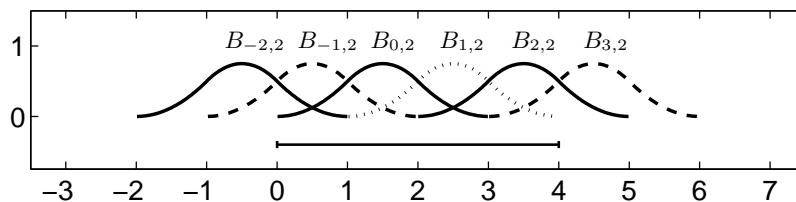


So far, you might be skeptical about the use of splines: all we have done is construct an alternative basis for piecewise constant and piecewise linear interpolants. The advantage of splines will be evident as the degree of the approximating polynomials increases.

Quadratic splines, $k = 2$. The construction of quadratic B-splines from the linear splines via the recurrence (13.1) forces the functions $B_{j,2}$ to have a continuous derivative, and also to be supported over three intervals per spline, as seen in the plot of $B_{0,2}$ shown earlier.

To understand the implications for determining the coefficients $c_{j,2}$, it is best to return to our concrete example. Suppose we are given data at the five points x_0, x_1, \dots, x_4 , as defined in the

previous table. Now the six splines $B_{-2,2}, B_{-1,2}, \dots, B_{3,2}$ all have nontrivial values on $[x_0, x_4]$, as shown in the following figure.



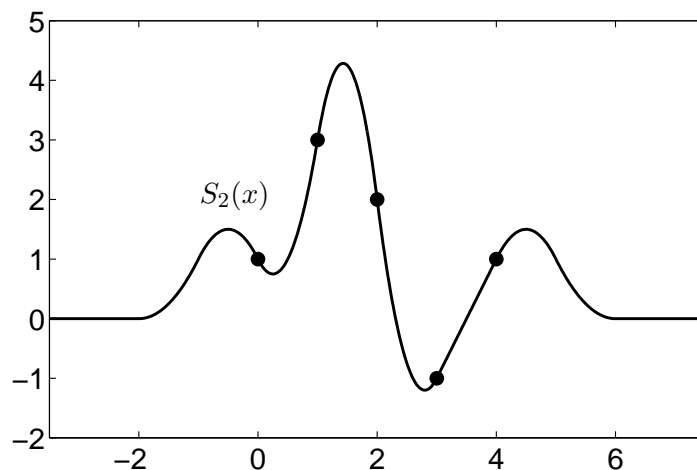
In general, S_2 will have the form

$$S_2(x) = \sum_{j=-2}^{n-1} c_{j,2} B_{j,2}(x).$$

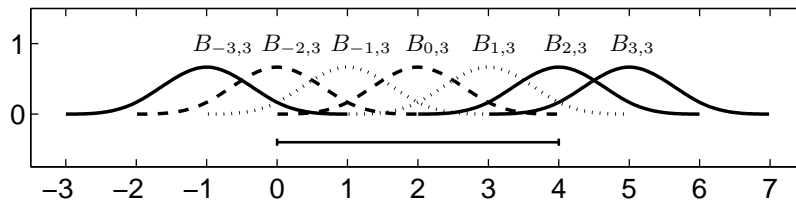
Since $n = 4$ in our specific example,

$$S_2(x) = \sum_{j=-2}^3 c_{j,2} B_{j,2}(x).$$

This equation has $n + 2 = 6$ nontrivial coefficients, but only five requirements: $S_2(x_j) = f_j$ for $j = 0, \dots, 4$. Thus, there are infinitely many quadratic splines that satisfy the interpolation conditions. How to choose one among them? Impose some extra condition, such as $S_2'(x_0) = 0$, or simply demand that $c_{-2,2} = 0$. In the figures below, we've arbitrarily set $c_{-2,2} = c_{n-1,2}$. Note that S_2 is supported on (x_{-2}, x_{n+2}) .



Cubic splines, $k = 3$. Cubic splines are the most famous of all splines. We began this section by discussing cubic splines as an alternative to piecewise cubic Hermite interpolation. Now we will show how to derive the same cubic splines from the cubic B-splines. The resulting S_3 will be a C^2 function that interpolates specified data. For the previous example with knots x_0, \dots, x_4 , the spline will be a linear combination of the $7 = n + k$ B-splines shown below.



To determine the coefficients of S_3 in the linear combination

$$S_3(x) = \sum_{j=-3}^{n-1} c_{j,3} B_{j,3}(x),$$

use the $n + 1$ conditions imposed by the interpolation requirement: $S_3(x_j) = f_j$. Infinitely many cubic splines satisfy these interpolation conditions; two independent requirements must be imposed to determine a unique spline. Recall the three alternatives discussed earlier: complete splines (specify a value for S'_3 at x_0 and x_n), natural splines (force $S''_3(x_0) = S''_3(x_n) = 0$), or not-a-knot splines. One can show that imposing natural spline conditions on S_3 introduces the following equations:

$$(x_2 - x_{-1})c_{-3,3} - (x_2 + x_1 - x_{-1} - x_{-2})c_{-2,3} + (x_1 - x_{-2})c_{-1,3} = 0$$

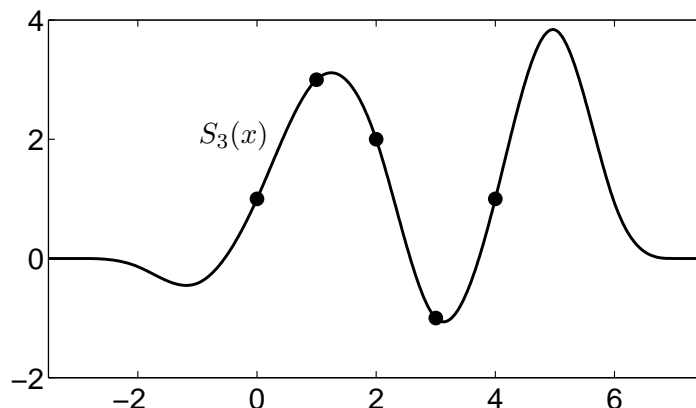
$$(x_{n+2} - x_{n-1})c_{n-3,3} - (x_{n+2} + x_{n+1} - x_{n-1} - x_{n-2})c_{n-2,3} + (x_{n+1} - x_{n-2})c_{n-1,3} = 0.$$

If the knots are equally spaced, $x_j = x_0 + jh$ for some fixed $h > 0$, these natural boundary conditions simplify:

$$3hc_{-3,3} - 6hc_{-2,3} + 3hc_{-1,3} = 0$$

$$3hc_{n-3,3} - 6hc_{n-2,3} + 3hc_{n-1,3} = 0.$$

With these in hand, the coefficients for S_3 are completely determined. In the notes for the next lecture, we will give details about construction of the associated linear system of equations. The plot below shows the cubic spline with natural boundary conditions, based on the data used in our previous example. Clearly this spline satisfies the interpolation conditions, but now there seems to be an artificial peak near $x = 5$ that you might not have anticipated from the data values. This is a feature of the natural boundary conditions: by forcing the second derivative to be zero, we ensure that the spline S_3 has constant slope at x_0 and x_n . Eventually this slope must be reversed, as our B-splines force $S_3(x)$ to be zero outside (x_{-3}, x_{n+3}) .



Of course, one can go to higher degree splines if greater continuity is required at the knots, or if there are more than two boundary conditions to impose (e.g., if one wants both first and second derivatives to be zero at the boundary).

Some omissions. The great utility of B-splines in engineering has led to the development of the subject far beyond these meager notes. Among the omissions are: interpolation imposed at points distinct from the knots, convergence of splines to the function they are approximating as the number of knots increases, integration and differentiation of splines, tension splines, etc. Splines in higher dimensions ('thin-plate splines') are used, for example, to design the panels of a car body.