1 Piecewise Cubic Interpolation

Typically the problem with piecewise linear interpolation is the interpolant is not differentiable as the interpolation points (it has a kinks at every interpolation point). A commonly used approach to obtain a smoother interpolation, is to instead replace the linear functions with cubic ones. This means that we will have an interpolation function of the form

$$s(x) = \begin{cases} s_0(x) & \text{if } x_0 \leq x < x_1 \\ s_1(x) & \text{if } x_1 \leq x < x_2 \\ \vdots & \vdots \\ s_{n-1}(x) & \text{if } x_{n-1} \leq x \leq x_n \end{cases}, \tag{1}$$

where $s_i(x)$ is a cubic polynomial of the form

$$s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3. \tag{2}$$

Since there are n polynomial functions with 4 coefficients each, we have $4n$ unknowns to solve for. To satisfy the interpolation condition $s(x_i) = y_i$ requires that

$$s_i(x_i) = y_i, \quad i = 0, \ldots, n-1, \quad \text{and} \quad s_{n-1}(x_n) = y_n \tag{3}$$

Note that since $s_i(x_i) = a_i$, the first n conditions above immediately imply that $a_i = y_i$ for $i = 0, \ldots n-1$. Additionally we will impose that $s(x)$ is twice continuously differentiable at the points $x_1, \ldots, x_{n-1}$. This means

$$\begin{aligned} s_{i-1}(x_i) = s_i(x_i) = y_i, &\quad i = 1, \ldots, n-1 \\ s'_{i-1}(x_i) = s'_i(x_i), &\quad i = 1, \ldots, n-1 \\ s''_{i-1}(x_i) = s''_i(x_i), &\quad i = 1, \ldots, n-1. \end{aligned} \tag{4}$$

The conditions (3) and (4) are the fundamental conditions that define a cubic spline. Note that in addition to $s_i(x_i) = a_i$ we also have $s'_i(x_i) = b_i$ and $s'\prime_i(x_i) = 2c_i$. If we count up all the constraints, we see that the interpolation condition (3) imposes $n+1$ constraints, while twice continuous differentiability (4) imposes $3(n-1)$ constraints. This amounts to

$$4n - 2 = (n+1) + 3(n-1)$$

total constraints, which is obviously 2 less than the $4n$ coefficients to be determined. Typically this is corrected by imposing two conditions at the end points of the interval. Some of the most common examples are the following

1. **Natural Spline**: $s''_0(x_1) = s''_{n-1}(x_n) = 0$
   This choice produces a spline with the natural property that it minimizes the total curvature of the the approximating spline (this will be discussed in more detail later).

2. **Clamped Spline**: $s_0'(x_0) = y_0'$ and $s_{n-1}'(x_n) = y_n'$
   Here y_0' and y_n' are either approximations of the derivatives of the function at that point or arbitrarily chosen to pin the slopes at a particular angle.

3. **Not-a-Knot Spline**: $s_0'''(x_1) = s_1'''(x_1)$ and $s_{n-2}'''(x_{n-1}) = s_{n-1}'''(x_{n-1})$

Of course, in order to determine the value of $s(x)$ for any one of these choices, one must solve a complicated linear system for the coefficients a_i, b_i, c_i, d_i, for $i = 0, \dots, n-1$. As it turns out, such a linear system can be written as a tridiagonal linear system and therefore solved in $\mathcal{O}(n)$ floating point operations.

Example: Lets consider an example. Suppose we want to find the natural cubic spline that interpolates the points $(0, 1), (1/2, -1)$ and $(1, 2)$. The spline take the form

$$s(x) = \begin{cases} s_0(x) & \text{if} \quad 0 \le x \le 1/2 \\ s_1(x) & \text{if} \quad 1/2 \le x \le 1, \end{cases} \tag{5}$$

where

$$s_0(x) = a_0 + b_0 x + c_0 x^2 + d_0 x^3$$

and

$$s_1(x) = a_1 + b_0(x - 1/2) + c_0(x - 1/2)^2 + d_0(x - 1/2)^3$$

From the interpolation conditions (3) and the continuity condition in (4) we have

$$\begin{aligned} s_0(0) = 1: &\qquad a_0 = 1, \\ s_1(1/2) = -1: &\qquad a_1 = -1, \\ s_1(1/2) = 2: &\qquad a_1 + \tfrac{1}{2}b_1 + \tfrac{1}{4}c_1 + \tfrac{1}{8}d_1 = 2, \\ s_0(1/2) = -1: &\qquad a_0 + \tfrac{1}{2}b_0 + \tfrac{1}{4}c_0 + \tfrac{1}{8}d_0 = -1. \end{aligned}$$

Also the derivative conditions from (4) imply that

$$\begin{aligned} s_0'(1/2) = s_1'(1/2): &\qquad b_0 + c_0 + \tfrac{3}{4}d_0 = b_1, \\ s_0''(1/2) = s_1''(1/2): &\qquad 2c_0 + 3d_0 = 2c_1. \end{aligned}$$

Finally the condition of being a natural spline implies that

$$\begin{aligned} s_0''(0) = 0: &\qquad c_0 = 0 \\ s_1''(0) = 0: &\qquad 2c_1 + 3d_1 = 0. \end{aligned}$$

Now one simply need to solve the above linear system for the coefficients. After doing this, one obtains

$$s_0(x) = 1 - \frac{13}{2}x + 10x^3$$

and

$$s_1(x) = -1 + (x - 1/2) + 15(x - 1/2)^2 - 10(x - 1/2)^3.$$

Lets see how this approach works in general. We will assume in what follows that the interpolation points $x_0 < x_1 < \ldots < x_n$ (knots) are equally spaced with $h = x_{i+1} - x_i$ being the spacing between points. As remarked already, the first n conditions in (3) imply that $a_i = y_i$. We still need to determine the remaining $3n$ coefficients b_i, c_i and d_i for $i = 0, \ldots, n-1$. If we include the last condition of (3) with the continuity condition of (4), then we have

$$s_{i-1}(x_i) = y_i, \quad i = 1, \ldots n.$$

Using the fact that $a_i = y_i$, this can be written as

$$y_i + b_i h + c_i h^2 + d_i h^3 = y_{i+1}, \quad i = 0, \ldots n-1. \tag{6}$$

When considering the derivative conditions in (4) we may use the fact that $s_i'(x_i) = b_i$ and $s_i''(x_i) = 2c_i$ to conclude that the first derivative conditions $s_{i-1}'(x_i) = s_i'(x_i), \quad i = 1, \ldots, n-1$ can be written as

$$b_i + 2c_i h + 3d_i h^2 = b_{i+1} \quad i = 0, \ldots n-2 \tag{7}$$

while the second derivative conditions $s_{i-1}''(x_i) = s_i''(x_i), \quad i = 1, \ldots, n-1$ can be written as

$$c_i + 3d_i h = c_{i+1} \quad i = 0, \ldots n-2. \tag{8}$$

If we consider the natural cubic spline condition, then we also have $c_0 = \frac{1}{2}s_0''(x_0) = 0$ and $c_{n-1} = \frac{1}{2}s_{n-1}''(x_n) = 0$. In this case it will make the most sense to attempt to reduce solving the linear systems (6), (7), (8) to solving for the coefficients c_i, $i = 1, \ldots n-2$, first (This is because the natural cubic spline condition are on c_0 and c_{n-1}. If we were to consider clamped splines, then it would make more sense to solve for b_i first.).

To begin, we solve for d_i in equation (8)

$$d_i = \frac{c_{i+1} - c_i}{3h}. \tag{9}$$

We may also solve for b_i in (6), which after using the above expression for d_i gives

$$b_i = f[x_i, x_{i+1}] - c_i h - d_i h^2 = f[x_i, x_{i+1}] - \frac{2}{3}c_i h - \frac{1}{3}c_{i+1} h \tag{10}$$

where we have used the divided difference notation $f[x_i, x_{i+1}] = (y_{i+1} - y_i)/h$. Finally we can substitute the expression for b_i and the expression for d_i into equation (7) to obtain

$$f[x_i, x_{i+1}] - \frac{2}{3}c_i h - \frac{1}{3}c_{i+1} h + 2c_i h + (c_{i+1} - c_i)h = f[x_{i+1}, x_{i+2}] - \frac{2}{3}c_{i+1} h - \frac{1}{3}c_{i+2} h$$

Rearranging this and collecting terms gives an equation for c_i, $i = 1, \ldots n-2$

$$\frac{1}{3}c_i + \frac{4}{3}c_{i+1} + \frac{1}{3}c_{i+2} = \frac{f[x_{i+1}, x_{i+2}] - f[x_i, x_{i+1}]}{h} = f[x_i, x_{i+1}, x_{i+2}].$$

Finally, using the fact that for the natural spline $c_0 = 0$ and $c_{n-1} = 0$ gives the following linear system

$$\frac{1}{3}\begin{bmatrix} 4 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 4 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 4 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ \vdots \\ c_{n-3} \\ c_{n-2} \end{bmatrix} = \begin{bmatrix} f[x_0, x_1, x_2] \\ f[x_1, x_2, x_3] \\ \vdots \\ \vdots \\ f[x_{n-3}, x_{n-2}, x_{n-1}] \\ f[x_{n-2}, x_{n-1}, x_n] \end{bmatrix} \tag{11}$$

Note that this is a tridiagonal linear system and is strictly diagonally dominant since $4 > 1+1$. Therefore the matrix is positive definite and non-singular and can be solved quickly using the Thomas algorithm with $\mathcal{O}(n)$ flops. Once you solve this system, for c_i, $i = 1, \ldots n-1$ the other coefficients b_i and d_i can be obtained from equations (10) and (9), each costs $\mathcal{O}(n)$ flops.

1.1 Cubic B-Splines

As another approach to the computation of cubic splines, we will find useful follow an approach similar to piecewise linear interpolation or Lagrange interpolation approach, introducin a family of basis functions. Our goal is to write the spline as

$$s(x) = \sum_i a_i B_i(x)$$

where $B_i(x)$ is a certain piecewise cubic spline. The functions $B_i(x)$ are called *cubic B-Splines*. Although similar to the formulas for piecewise linear interpolation and Lagrange interpolation formulas, there is an important difference. The coefficients a_i are not necessarily equal to the function output values y_i.

The main benefit of this approach is that the interpolation procedure can be broken into two pieces. First construct the B-splines $B_i(x)$ and then find the a_i's. The has the benefit that, since the construction of the B-splines will not in any way depend on the y_i data, meaning, once you have computed the B-spines you do not need to compute them again for different sets of y_i values.

Computing the B-splines

We will assume that each $B_i(x)$ is piecewise cubic function of the following form

$$B_i(x) = \begin{cases} 0 & \text{if } x \leq x_{i-2}, \\ q_{i-2}(x) & \text{if } x_{i-2} \leq x_{i-1}, \\ q_{i-1}(x) & \text{if } x_{i-1} \leq x_i, \\ q_{i+1}(x) & \text{if } x_i \leq x_{i+1}, \\ q_{i+2}(x) & \text{if } x_{i+1} \leq x_{i+2}, \\ 0 & \text{if } x_{i+2} \leq x, \end{cases}$$

where $q_i(x)$, $i = 0, \ldots n$ are the usual cubic polynomials of the form

$$q_i(x) = A_i + B_i(x - x_i) + C_i(x - x_i)^2 + D_i(x - x_i)^3.$$

The functions q_i will be determined by the requirement that

$$B_i(x_{i-2}) = B_i'(x_{i-2}) = B_i''(x_{i-2}) = 0 \tag{12}$$

and

$$B_i(x_{i+2}) = B_i'(x_{i+2}) = B_i''(x_{i+2}) = 0$$

and that $B_i(x)$ is twice continuously differentiable at x_{i-1}, x_i, x_{i+1}. Since we are assuming the interpolation points are equally spaced, this will imply that $B_i(x)$ is symmetric about $x = x_i$. Note that the two continuous derivatives and the symmetry automatically imply that $B_i'(x_i) = 0$. Using symmmetry, it follows that we only need to ensure equation (12) holds,that the two continuous derivatives at x_{i-1} match and that $B_i'(x_i) = 0$. This is accomplished by requiring the following

$$q_{i-2}(x_{i-2}) = q_{i-2}'(x_{i-2}) = q_{i-2}''(x_{i-2}) = 0$$

$$q_{i-2}(x_{i-1}) = q_{i-1}(x_{i-1}), \quad q_{i-2}'(x_{i-1}) = q_{i-1}'(x_{i-1}), \quad q_{i-2}''(x_{i-1}) = q_{i-1}''(x_{i-1})$$

as well as

$$q_{i-1}'(x_i) = 0.$$

From the conditions at x_{i-2} we can easily conclude (by integration) that

$$q_{i-2} = D_{i-2}(x - x_{i-2})^3.$$

Additionally, from the remaining smoothness conditions at x_{i-1} and the zero derivative condition at x_i one finds that

$$q_{i-1}(x) = D_{i-2}\left(h^3 + 3h^2(x - x_{i-1}) + 3h(x - x_{i-1})^2 - 3(x - x_{i-1})^3\right)$$

Of course, since we needed to determine 8 coefficients for q_{i-2} and q_{i-1} with only 7 constraints, we have an undetermined constant D_{i-1}. Typically the convention is to take $B_i(x_i) = 2/3$ which sets $D_i = 1/6h^3$. Using the symmetry of $B_i(x)$ about $x = x_i$ we automatically obtain $q_{i+1}(x)$ and $q_{i+2}(x)$ by the formulas

$$q_{i+1}(x) = q_{i-1}(2x_i - x), \quad q_{i+2}(x) = q_{i-2}(2x_i - x).$$

After doing this, we find that $B_i(x)$ is completely determined by

$$\begin{aligned}
q_{i-2} &= \frac{1}{6h^3}(x - x_{i-2})^3 \\
q_{i-1} &= \frac{1}{6} + \frac{1}{2h}(x - x_{i-1}) + \frac{1}{2h^2}(x - x_{i-1})^2 - \frac{1}{2h^3}(x - x_{i-1})^3 \\
q_{i+1} &= \frac{1}{6} - \frac{1}{2h}(x - x_{i+1}) + \frac{1}{2h^2}(x - x_{i+1})^2 - \frac{1}{2h^3}(x - x_{i+1})^3 \\
q_{i+2} &= -\frac{1}{6h^3}(x - x_{i+2})^3
\end{aligned}$$

By factoring the polynomials in the above formula, it is possible to show that

$$B_i(x) = B\left(\frac{x - x_i}{h}\right)$$

where

$$B(x) = \begin{cases} \frac{2}{3} - x^2\left(1 - \frac{1}{2}|x|\right) & \text{if } |x| < 1, \\ \frac{1}{6}(2 - |x|)^3 & \text{if } 1 \leq |x| \leq 2, \\ 0 & \text{if } 2 \leq |x|. \end{cases}$$

Finding the coefficients

Now we write our spline as

$$s(x) = \sum_{i=-1}^{n+1} a_i B_i(x)$$

Note that we have taken the sum over $i = -1, \ldots, n+1$ even though we do not have points $x_{-3}, x_{-2}, x_{-1}, x_{n+1}, x_{n+2}, x_{n+3}$. This is because we want to sum over all B_i which are non-zero on the interval $[x_0, x_n]$. When a piece of $B_i(x)$ lies outside of the interval $[x_0, x_n]$, we simply ignore that piece. Any piece of $B_i(x)$ that lies inside $[x_0, x_n]$ will, by the piecewise polynomial nature of $B_i(x)$, necessarily only depend on the points x_i that belong to $x_0, \ldots, x_n$.

The task of determining the coefficients a_i is now simple once you realize that at any point x_i only $B_{i-1}(x_i)$, $B_i(x_i)$ and $B_{i+1}(x_i)$ are non-zero (i.e. they are the only functions 'turned on' at x_i). Using the fact that for each $i = 1, \ldots n-1$ we have the exact values

$$B_i(x_{i-1}) = \frac{1}{6}, \quad B_i(x_i) = \frac{2}{3}, \quad B_i(x_{i+1}) = \frac{1}{6}$$

We see that

$$s(x_i) = \frac{1}{6}\left(a_{i-1} + 4a_i + a_{i+1}\right)$$

and therefore, the interpolation requirement $s(x_i) = y_i$ for $i = 0, \ldots n$ implies that

$$a_{i-1} + 4a_i + a_{i+1} = 6y_i \quad \text{for} \quad i = 1, 2, \ldots, n.$$

In order to solve for a_i we need to know the values of a_{-1} and a_{n+1}. Again, this is where the additional endpoint conditions are used. It is not difficult to show that for the natural spline, we have

$$s''(x_i) = a_{i-1}B_i''(x_i) + a_i B_i''(x_i) + a_{i-1}B_{i+1}''(x_i) = \frac{1}{h^2}(a_{i-1} - 2a_i + a_{i+1}).$$

Solving the natural spline conditions $s''(x_0) = 0$ and $s''(x_n) = 0$ using the above formula give

$$a_{-1} = 2a_0 - a_1, \quad a_{n+1} = 2a_n - a_{n-1}.$$

This means when $i = 0$, we get $a_0 = y_0$ and when $i = n$, we get $a_n = y_n$. Now the remaining linear system for a_i, $i = 1, \ldots, n-1$ can be solved by a tridiagonal system

$$\begin{bmatrix} 4 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 4 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 4 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_{n-2} \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} 6y_1 - y_0 \\ 6y_2 \\ \vdots \\ \vdots \\ 6y_{n-2} \\ 6y_{n-1} - y_n \end{bmatrix}.$$

Again we find that the remaining coefficients can be computed by inverting a tridiagonal, positive definite system. The main difference between this system and the tridiagonal system obtained (11) lies in the right-hand side. In (11) the right-hand side involves computing $n-2$ 2nd order divided differences, while the right-hand side of the above linear system only involves the values y_i, $i = 0, \ldots, n$. In some sense we have already done this computation in computing the B-splines $B_i(x)$.