1 The spline interpolation problem

Same as in polynomial interpolation, we will assume that our data corresponds to function evaluations $y_j = f(x_j)$ at interpolation nodes x_j. For simplicity's sake, we will assume that $x_0 = a$ and $x_n = b$, and that the nodes are ordered: $a = x_0 \leq x_1 \leq \cdots \leq x_n = b$.

A spline of degree k is a function $s(x)$ such that its restriction to each subinterval $I_i = [x_{i-1}, x_i]$ for $i = 1, \ldots, n$ is a polynomial of degree $\leq q$, that is, $s(x)|_{I_i} \in \mathcal{P}_k$. Linear splines are thus piece-wise linear, quadratic splines are piece-wise quadratic, and so-on.

2 Linear splines

Linear splines (piece-wise linear) are the most straight-forward to work with, and intuitively we know this because the interpolant is essentially just "connect-the-dots" for the interpolation data with straight lines. Further: we can use what we know from polynomial interpolation to find the formula:

$$s(x)|_{I_i} = y_{i-1}\left(\frac{x - x_i}{x_{i-1} - x_i}\right) + y_i\left(\frac{x - x_{i-1}}{x_i - x_{i-1}}\right)$$

That is, we have used Lagrange interpolation (linear) to define each piece in terms of the two available data points. We can also readily see from this that the linear spline is unique (corresponds to the unique solution to n linear interpolation problems).

We can also, if we wish, define a spline basis that mimics the Lagrange basis; this is related to so-called B-splines ("base splines"). If we define our basis terms such that:

$$B_j(x) = \left(\frac{x - x_{j-1}}{x_j - x_{j-1}}\right) \quad x \in I_j$$

$$B_j(x) = \left(\frac{x - x_{j+1}}{x_j - x_{j+1}}\right) \quad x \in I_{j+1}$$

$$B_j(x) = 0 \quad x \in [a, b] \setminus (I_j \cup I_{j+1})$$

These "linear tent" spline basis is such that

$$s(x) = \sum_{j=0}^{n} y_j B_j(x)$$

Is the unique spline interpolant that solves the problem for data y_j.

2.1 Error analysis

Assume that the $n + 1$ interpolation nodes for a linear spline in $[a, b]$ are equispaced, with spacing $h = (b - a)/n$. First, this simplifies the formulas above; for instance:

$$s(x)|_{I_i} = y_{i-1}\left(\frac{x - x_i}{h}\right) + y_i\left(\frac{x - x_{i-1}}{h}\right)$$

Second: let's recall our polynomial interpolation error estimates. Assuming $f \in C^2([a, b])$, for each linear interpolation problem, we have:

$$|E(t)| = |f(t) - s(t)| \leq \frac{\max_{x \in I_i} |f''(x)|}{2} |(t - x_{i-1})(t - x_i)| \quad x \in I_i \tag{2.1}$$

Now, because our points are equispaced, we can simply find the maximum value for $\Psi(x)$ for the interval $[0, h]$, and use that to find a uniform bound for $|(t - x_{i-1})(t - x_i)|$.

$$q(x) = x(x - h) = x^2 - hx$$
$$q'(x) = 2x - h = 0 \quad \rightarrow x = h/2$$
$$q(h/2) = -h^2/4$$

So, the maximum value for $|\Psi(x)|$ is $h^2/4$. As a consequence:

Theorem 2.1 (Linear spline interpolation error) *Let $x_j = jh$, $h = (b - a)/n$ equispaced interpolation nodes in $[a, b]$, and $f \in C^2([a, b])$. Let $E(t) = f(t) - s(t)$, where $s(x)$ is the unique linear spline interpolant. Then,*

$$|E(t)| \leq \frac{\max_{x \in [a,b]} |f''(x)|}{8} h^2 \tag{2.2}$$

3   Cubic splines

There are two reasons we might want to increase the degree of each polynomial piece in our splines: for one, we might want convergence of the error to go to zero faster than $O(h^2)$. More importantly: the spline $s(x)$ is a very jagged function: it is not generally differentiable at the interpolation nodes (and is piece-wise constant), and all derivatives of higher order are zero (except at the interpolation nodes, where they are undefined).

While it is possible to discuss splines of any order (or even mix polynomial orders for different sub-intervals), for many practical applications cubic splines are a good compromise between smoothness and practicality / cost efficiency. We will thus focus our discussion on splines which, restricted to each sub-interval, are cubic polynomials.

Something is immediately apparent: each cubic polynomial has 4 constants to determine, and yet, on each subinterval we have only two conditions (interpolation data at each end-point). This means we have two degrees of freedom to play with for each polynomial piece! We want our cubic spline to be as smooth as possible, and it turns out we can "use" these degrees of freedom to ask that $s'(x)$ and $s''(x)$ be continuous in the whole interval. In other words, we could try to define the space of cubic splines as the set of functions that satisfy:

$$s(x)|_{I_i} \in \mathcal{P}_3 \quad s \in C^2([a, b]) \tag{3.1}$$

For notational purposes, we will denote $s|_{I_i} = s_i$. These added continuity requirements correspond to asking that

$$s_i'(x_i) = s_{i+1}'(x_i)$$
$$s_i''(x_i) = s_{i+1}''(x_i)$$

this adds two extra equations per polynomial piece, *except* for the edge pieces $s_0(x)$ and $s_n(x)$. That means we have two *additional* conditions we can impose (and that finally determine a unique cubic spline that satisfies all of this). What we choose for these two conditions determines the kind of cubic spline we are using:

- **Normal splines:** We set $s''(a) = 0$ and $s''_n(b) = 0$. This condition has to do with minimizing overall curvature of the resulting cubic spline.

- **Clampled splines:** Given derivative data at endpoints, set $s'_0(a) = f'(a)$ and $s'_n(b) = f'(b)$. Intuitively, this fixes rates of change in and out of the interval.

- **Periodic splines:** Assuming our function $f(x)$ is periodic, it is natural to ask that $s'_0(a) = s'_n(b)$ and $s''_0(a) = s''_n(b)$.

- **Not-a-knot splines:** This matches the third derivatives for the pairs of pieces nearest to the end-points. That is: $s'''_0(x_1) = s'''_1(x_1)$ and $s'''_{n-1}(x_{n-1}) = s'''_n(x_{n-1})$.

Each of these families of splines is uniquely determined by $4n$ interpolation conditions: Asking that the spline, its derivative and its second derivative be continuous across the n sub-intervals ($3(n-1)$ conditions), asking that the spline interpolates ($n+1$ conditions) and two extra conditions at the end points.

3.1 Constructing a cubic spline: natural spline example

There are a number of ways to determine formulas for each cubic $s_i(x)$, or to find a spline basis to write the solution in terms of the interpolation data (which again, amounts to solving the interpolation problem for data $y_j = \delta_{i,j}$, much like for Lagrange, Hermite-Lagrange, etc).

Setting the second derivative condition

One way to construct them is to start with the second derivative. That is,

$$s''_i(x) = a_{i-1}\frac{x - x_i}{x_{i-1} - x_i} + a_i\frac{x - x_{i-1}}{x_i - x_{i-1}} = \frac{1}{h_i}\left(-a_{i-1}(x - x_i) + a_i(x - x_{i-1})\right) \tag{3.2}$$

where $h_i = x_i - x_{i-1}$. If our nodes are equispaced, $h_i = h = (b - a)/n$.

What we have achieved here is, in one fell-swoop, define $s(x)$ such that $s''(x)$ is continuous, with $s''(x_i) = a_i$. Further, imposing that s be a natural spline is as simple as asking that $a_0 = a_n = 0$ in this representation.

Antidifferentiate and set interpolation conditions

We antidifferentiate (twice) and look at the resulting formulas for each cubic piece. By doing that, the antiderivative is defined up to a linear factor, which we write in terms of Lagrange polynomials for convenience. This yields:

$$s_i(x) = \frac{1}{h_i}\left(-a_{i-1}\frac{(x - x_i)^3}{6} + a_i\frac{(x - x_{i-1})^3}{6} - b_i(x - x_i) + c_i(x - x_{i-1})\right) \tag{3.3}$$

Setting the interpolation condition at the left end-point $x = x_{i-1}$ removes two terms, and gives us:

$$y_{i-1} = s_i(x_{i-1}) = \frac{1}{h_i}\left(a_{i-1}\frac{h_i^3}{6} + b_i h_i\right)$$

$$y_{i-1} - a_{i-1}\frac{h_i^2}{6} = b_i$$

And setting conditions at the right end-point $x = x_i$ removes the other two terms, and gives us:

$$y_i = s_i(x_i) = \frac{1}{h_i}\left(a_i\frac{h_i^3}{6} + c_i h_i\right)$$

$$y_i - a_i\frac{h_i^2}{6} = c_i$$

That is: the coefficients for the linear term are a function of a_i and the data y_i. In this, remember that $a_0 = a_n = 0$.

Setting continuity of the derivative

The last and most important step is to set the continuity conditions for the derivative. Given all of our previous work, this will give us a set of $n-1$ linear equations on the coefficients $a_1, \ldots, a_{n-1}$. Once we solve this linear system, we can recover b_i and c_i, and we now have formulas for each cubic spline piece.

We compute the derivative:

$$s_i'(x) = \frac{1}{h_i}\left(-a_{i-1}\frac{(x-x_i)^2}{2} + a_i\frac{(x-x_{i-1})^2}{2} - b_i + c_i\right)$$

$$= \frac{1}{h_i}\left(-a_{i-1}\frac{(x-x_i)^2}{2} + a_i\frac{(x-x_{i-1})^2}{2} + (y_i - y_{i-1}) - (a_i - a_{i-1})\frac{h_i^2}{6}\right)$$

after plugging in the formulas for b_i and c_i. The continuity conditions for the derivative at subinterval endpoints $s_i'(x_i) = s_{i+1}'(x_i)$ for $i = 1, \ldots, n-1$ yield:

$$a_i\frac{h_i}{2} + \frac{(y_i - y_{i-1})}{h_i} - (a_i - a_{i-1})\frac{h_i}{6} = -a_i\frac{h_{i+1}}{2} + \frac{(y_{i+1} - y_i)}{h_{i+1}} - (a_{i+1} - a_i)\frac{h_{i+1}}{6}$$

$$a_{i-1}\frac{h_i}{6} + a_i\frac{2(h_i + h_{i+1})}{6} + a_{i+1}\frac{h_{i+1}}{6} = \frac{(y_{i+1} - y_i)}{h_{i+1}} - \frac{(y_i - y_{i-1})}{h_i} = f[x_i, x_{i+1}] - f[x_{i-1}, x_i]$$

$$a_{i-1}\frac{h_i}{6(h_i + h_{i+1})} + a_i\frac{2}{6} + a_{i+1}\frac{h_{i+1}}{6(h_i + h_{i+1})} = \frac{\frac{(y_{i+1}-y_i)}{h_{i+1}} - \frac{(y_i - y_{i-1})}{h_i}}{h_i + h_{i+1}} = f[x_{i-1}, x_i, x_{i+1}]$$

If the nodes are equispaced, this simplifies to:

$$a_{i-1}\frac{1}{12} + a_i\frac{4}{12} + a_{i+1}\frac{1}{12} = \frac{\frac{(y_{i+1}-y_i)}{h} - \frac{(y_i - y_{i-1})}{h}}{2h} = \frac{y_{i+1} - 2y_i + y_{i-1}}{2h^2} = f[x_{i-1}, x_i, x_{i+1}] \qquad (3.4)$$

In either case, what we have here is a tri-diagonal system of $n-1$ linear equations, where the right-hand-side is a second difference of values of $f(x)$ (and so, approximately equal to a second derivative). In matrix form, the systems for the general case and the equispaced case are:

$$\frac{1}{12}\begin{bmatrix} 4 & \frac{2h_2}{(h_1+h_2)} & 0 & \cdots & 0 & 0 \\ \frac{2h_2}{(h_2+h_3)} & 4 & \frac{2h_3}{(h_2+h_3)} & \cdots & 0 & 0 \\ 0 & \frac{2h_3}{(h_3+h_4)} & 4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 4 & \frac{2h_{n-1}}{(h_{n-2}+h_{n-2})} \\ 0 & 0 & 0 & \cdots & \frac{2h_{n-1}}{(h_{n-1}+h_n)} & 4 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-2} \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} f[x_0,x_1,x_2] \\ f[x_1,x_2,x_3] \\ f[x_2,x_3,x_4] \\ \vdots \\ f[x_{n-3},x_{n-2},x_{n-1}] \\ f[x_{n-2},x_{n-1},x_n] \end{bmatrix}$$
$$(3.5)$$

$$\frac{1}{12}\begin{bmatrix} 4 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 4 & 1 & \cdots & 0 & 0 \\ 0 & 1 & 4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 4 & 1 \\ 0 & 0 & 0 & \cdots & 1 & 4 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-2} \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} f[x_0,x_1,x_2] \\ f[x_1,x_2,x_3] \\ f[x_2,x_3,x_4] \\ \vdots \\ f[x_{n-3},x_{n-2},x_{n-1}] \\ f[x_{n-2},x_{n-1},x_n] \end{bmatrix}$$
$$(3.6)$$

In either case, the matrix for this system could not be nicer: It is tri-diagonal, strictly diagonally dominant, and symmetric positive definite. We can do Gaussian Elimination *without pivoting*, solving this system in about $3n$ operations. Once we have a solution for this system, we can find b_i and c_i using the formulas we found in the previous step.

3.2 Error Estimates

The error analysis for cubic splines is a bit more involved than for polynomial or piecewise linear interpolation, and it can of course depend on what kind of cubic spline we choose (natural, periodic, not-a-knot, clamped) and the assumptions we can make on the underlying function $f(x)$.

We can state this general theorem (without proof) for cubic spline error estimates:

Theorem 3.1 (Cubic spline error estimates) *Let $f \in C^4([a,b])$, and let $s(x)$ be a cubic spline as defined in these notes, with $h = max_i\{h_i\}$. There exist positive constants C_m for $m = 0,1,2,3$ such that:*

$$|f^m(t) - s^m(t)| \le C_m\|f^4\|_\infty h^{4-m}$$

The optimal constants for equispaced points are: $C_0 = \frac{5}{384}$, $C_1 = \frac{1}{24}$, $C_2 = \frac{3}{8}$ and $C_4 = 1$.

This general result, which considers largest error bound for all possible cubic splines, can be found by showing that cubic splines are close to a piecewise Hermite polynomial interpolant, and that in turn is close to the true function. In other words, given $f(x)$, a cubic spline $s(x)$ and a piece-wise hermite interpolant $h(x)$ are found. We apply the triangle inequality to the error:

$$|f(t) - s(t)| \le |f(t) - h(t)| + |h(t) + s(t)| \tag{3.7}$$

It is then shown that both terms in that sum can be bounded by expressions of the form $C\|f^4\|_\infty h^4$.

What this theorem tells us is that the error between our function and a cubic spline for n equispaced points with spacing h goes uniformly to zero and is $O(h^4)$ (for non-uniform spacing, we can either uniformly bound with the largest spacing, or look at the local error which is $O(h_i^4)$). Not only that: the difference between the first derivatives is $O(h^3)$, second derivatives is $O(h^2)$ and third derivatives is $O(h)$.

If we plan to use a cubic spline as a model of a smooth function, this gives us an idea of how close the derivatives of our interpolant will be to the true derivatives. This interpolation error is then a crucial part in any error analysis we do using it as a model or discretization of the underlying function.