Next: Trigonometric Interpolation Up: Spline Interpolation Previous: Piecewise Linear Case

Cubic Splines

A piece-wise technique which is very popular. Recall the philosophy in splining is to use low order polynomials to interpolate from grid point to grid point. This is ideally suited when one has control of the grid locations and the values of the data being interpolated (i.e. you have a way to produce them at any location). So if we have this control, as we saw above, we can control the relative accuracy by changing the overall spacing between the grid points.

Why cubic splines? Because cubic splines are the lowest order polynomial endowed with inflection points. Think about interpolating a set of data points using parabolic (quadratic) functions: without inflection points the interpolation can look rather strange....why not higher order? because it is more expensive and the eye cannot really discern an appreciable difference between the cubic and higher order interpolating splines.

Here we could have used the method presented in connection with the piece-wise linear splines to construct the spline interpolation of the data using cubics. However, we purposely show an alternative technique, which is less elegant but nevertheless revealing.

Consider functions $Sp(x_n)$ such that $S_n^3(x) \in Sp(x_n)$ then $S(x)$ satisfies these properties:

Cubic Spline $S_n^3(x)$

(18)
$$\begin{cases} S_n^3(x) \in C^2[a,b] \text{ i.e. } S_n^3(x), \partial_x S_n^3(x), \partial_x^2 S_n^3(x) \text{ continuous on } [a,b] \\ S_n^3(x_j) = f(x_j) \equiv f_j \quad 0 \le j \le n \Rightarrow S_n^3(x) \text{interpolates } f(x) \text{ on } [a,b] \\ S_n^3(x) \text{ is a cubic polynomial on each interval} \\ [x_j, x_{j+1}] \quad 0 \le j \le n-1. \end{cases}$$

Higher-order splines can be constructed similarly.

Can such functions be constructed?

So $S_n^3(x)$ is represented by the cubic polynomials S_0^3 on $[x_0, x_1]$, S_1^3 on $[x_1, x_2]$ and S_2^3 on $[x_2, x_3]$.

Since $S_j^3(x) \in \mathcal{P}_3$ it has 4 coefficients and we'd like it to satisfy (18) therefore we have 12 coefficients to pin down.

Since each $S_j^3(x) \in \mathcal{P}_3$, all its derivatives are continuous for any $x \in (x_j, x_{j+1})$ (open interval)

therefore need to worry about continuity at each node x_j only, where cubics ``patch together"with 2-order continuity.

Let $[a,b] \equiv [x_0, x_3]$. We need 12 equations for 12 unknowns.

6 equations are generated by:

$$\begin{cases} S_0^3(x_0) = f_0 & S_1^3(x_1) = f_1 & S_2^3(x_2) = f_2 & S_2^3(x_3) = f_3 \\ S_0^3(x_1) = S_1^3(x_1) & S_1^3(x_2) = S_2^3(x_2) & & \end{cases}$$

Since $S_n^3(x)$ and its first two derivatives are continuous at x_1 and x_2 we can get 4 more equations:

$$\begin{cases} \partial_x S_0^3(x_1) = \partial_x S_1(x_1) \quad , \quad \partial_x^2 S_0^3(x_1) = \partial_x^2 S_1^3(x_1) \\ \partial_x S_1^3(x_2) = \partial_x S_2^3(x_2) \quad , \quad \partial_x^2 S_1^3(x_2) = \partial_x^2 S_2^3(x_2) \end{cases}$$

All told, 10 equations thus we expect an infinite number - a family of 2-parameter solutions. Need 2 more equations to pin down the 12 coefficients. We use EITHER:

$$\begin{aligned} \partial_x^2 S^3(x_0) &= \partial_x^2 S^3(x_3) = 0 \\ &\quad \text{(“free” or “natural” boundary condition).} \\ \partial_x S^3(x_0) &= \partial_x f_0 \text{ and } \partial_x S^3(x_3) = \partial_x f_3 \\ &\quad \text{(“clamped” boundary condition).} \end{aligned}$$

So a complete list of properties is

(19) S_j^3 is cubic polynomial on $[x_j, x_{j+1}] \quad j = 0 \cdots n-1.$

(20) $S^3(x_j) = f_j \quad j = 0 \cdots n$

(21) $S_{j+1}^3(x_{j+1}) = S_j^3(x_{j+1}) \quad j = 0 \cdots n-2$

(22) $\partial_x S_{j+1}^3(x_{j+1}) = \partial_x S_j^3(x_{j+1}) \quad j = 0 \cdots n-2$

(23) $\partial_x^2 S_{j+1}^3(x_{j+1}) = \partial_x^2 S_j^3(x_{j+1}) \quad j = 0 \cdots n-2$

One of the following is satisfied:

(i)

$$\partial_x^2 S^3(x_0) = \partial_x^2 S^3(x_n) = 0 \text{ free, OR}$$

(ii)

$$\partial_x S^3(x_0) = \partial_x f_0 \text{ and } \partial_x S^3(x_n) = \partial_x f_n \text{ clamped.}$$

Note that for the clamped case we need derivative information: either from the physics or from some assumption.

Construction: apply above conditions.

$$S_j^3(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \quad j = 0 \cdots n-1$$

(24) $$S_j^3(x_j) = a_j = f(x_j) = f_j$$

$$a_{j+1} = S_{j+1} = S_j(x_{j+1}) =$$

(25) $$a_j + b_j(x_{j+1} - x_j) + c_j(x_{j+1} - x_j)^2 + d_j(x_{j+1} - x_j)^3 \quad j = 0 \cdots n-2$$

$$\text{let } \begin{cases} h_j = x_{j+1} - x_j & j = 0, \ldots, n-1 \\ a_n = f_n \end{cases}$$

(26) $$a_{j+1} = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3 \quad j = 0 \ldots n-1$$

and let $b_n = \partial_x S^3(x_n)$.

(27) $$\partial_x S_j^3(x) = b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2 \Rightarrow \partial_x S_j^3(x_j) = b_j$$

(28) $$b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2 \quad j = 0 \cdots n-1$$

(29) $$\text{let } c_n = \partial_x^2 S^3(x_n)/2$$

Solve for d_j in

(30) $$c_{j+1} = c_j + 3d_j h_j$$

and substitute into (28) and (26) to get

$$a_{j+1} = a_j + b_j h_j + \frac{h_j^2}{3}(2c_j + c_{j+1})$$

(31) $$j = 0 \cdots n-1$$

(32) $$b_{j+1} = b_j + h_j(c_j + c_{j+1})$$

Then solve for b_j in (31)

(33) $$b_j = \frac{1}{h_j}(a_{j+1} - a_j) - \frac{h_j}{3}(2c_j + c_{j+1})$$

$$\therefore \quad b_{j-1} = \frac{1}{h_{j-1}}(a_j - a_{j-1}) - \frac{h_{j-1}}{3}(2c_{j-1} + c_j)$$

Substitute b_j and b_{j-1} into (32): gives

(34) $$h_{j-1}c_{j-1} + 2(h_{j-1} + h_j)c_j + h_j c_{j+1} =$$
$$\frac{3}{h_j}(a_{j+1} - a_j) - \frac{3}{h_{j-1}}(a_j - a_{j-1})$$

$$j = 1, 2 \cdots n-1$$

Since we know h_j and $a_j = f_j \Rightarrow \{c_j\}_0^n$ are unknowns.

Once $\{c_j\}$ are known then $\{b_j\}$ solved by (33) and $\{d_j\}$ solved by (30)

Natural spline case: $\partial_x^2 S^3(a) = \partial_x^2 S^3(b) = 0$

$$\therefore c_n = \frac{\partial_x^2 S^3(x_n)}{2} = 0$$

$$0 = \partial_x^2 S^3(x_0) = 2c_0 + 6d_0(x_0 - x_0) \therefore c_0 = 0$$

$$A\mathbf{x} = \mathbf{b}$$

A is $(n+1) \times (n+1)$ matrix

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ \vdots \\ c_n \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \vdots \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 0 \end{bmatrix}$$

Clamped Case: $\partial_x S^3(a) = \partial_x f(a)$ and $\partial_x S^3(b) = \partial_x f(b)$

$$A\mathbf{x} = \mathbf{b}$$

$$A = \begin{bmatrix} 2h_0 & h_0 & 0 & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & 0 & h_{n-1} & 2h_{n-1} \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ \vdots \\ c_n \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} \frac{3}{h_0}(a_1 - a_0) - 3f'(a) \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \vdots \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 3f'(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1}) \end{bmatrix}$$

□

Algorithm Natural Cubic Spline

input $n; x_0 \cdots x_n, a_0 = f(x_0) \cdots a_n = f(x_n)$

output $a_j, b_j, c_j d_j$ for $j = 0, 1 \cdots n-1$

Step 1 for $i = 1 \cdots n-1 \quad h_i = x_{i+1} - x_1$ Step 2 for $i - 1 \cdots n-1$

$$a_i = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - a_{i-1})$$

Step 3 $l_0 = 1$

$\mu_0 = 0$

$z_0 = 0$

Step 4 for $i = 1 \cdots n-1$

$l_i = 2(x_{j+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$

$\mu_i = h_i / l_i$

$$z_i = (\alpha_i - h_{i-1} z_{i-1})/l_i$$

Step 5 $l_n = 1$

$x_n = 0$

$c_n = 0$

Step 6 for $j = n-1 \cdots 0$

$c_j = z_j - \mu_j c_{j+1}$

$b_j = (a_{j+1} - a_j)/h_j - h_j(c_{j+1} + zc_j)/3$

$d_j = (c_{j+1} - c_j)/(3h_j)$

Step 7 output $(a_j, b_j, c_j, d_j$ for $j = 0 - n - 1)$

STOP

Clamped Cubic Spline

input $n_j, x_0 \cdots x_n, a_0 = f(x_0) \cdots a_n = f(x_n); FPO = f'(x_0) FPN = f'(x_n)$

output (a_j, b_j, c_j, d_j for $j = 0 \cdots n-1$)

Step 1 for $i = 0 \cdots n-1 \quad h_i = x_{i_{+1}} - x_i$

Step 2 Set $\alpha_0 = 3(a_1 - a_0)/h_0 - 3$ FPO

$\alpha_n = 3$ FPN $-3(a_n - a_{n-1})/h_{n-1}$

Step 3 for $i = 1, 2 \cdots n-1$

$$\alpha_i = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - a_{i-1})$$

Step 4 Set $l_0 = 2h_0$

$\mu_0 = 0.5$

$z_0 = \alpha_0/l_0$

Step 5 For $i = 1, 2 \cdots n-1$

$$l_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$$

$$\mu_i = h_i / l_i$$

$$z_i = (\alpha_1 - h_{i-1} z_{i-1})/l_i$$

Step 6 $l_n = h_{n-1}(2 - \mu_{n-1})$

$$z_n = (\alpha_n - h_{n-1} z_{n-1})/l_n$$

$$c_n = z_n$$

Step 7 for $j = n-1 \cdots 0$

set $c_j = z_j - u_j c_{j+1}$

$$b_j = (a_{j+1} - a_j)/h_j - h_j(c_{j+1} + 2c_j)/3$$

$$d_j = (c_{j+1} - c_j)/(3hj)$$

Step 8 Output $(a_{j-g}, c_j, d_j j = 0 \cdots n-1)$

STOP

□

Figures 36 and 37 illustrate how cubic spline interpolation compares with a Lagrange polynomial interpolation. It also shows the effect of using different end conditions on the spline interpolation.

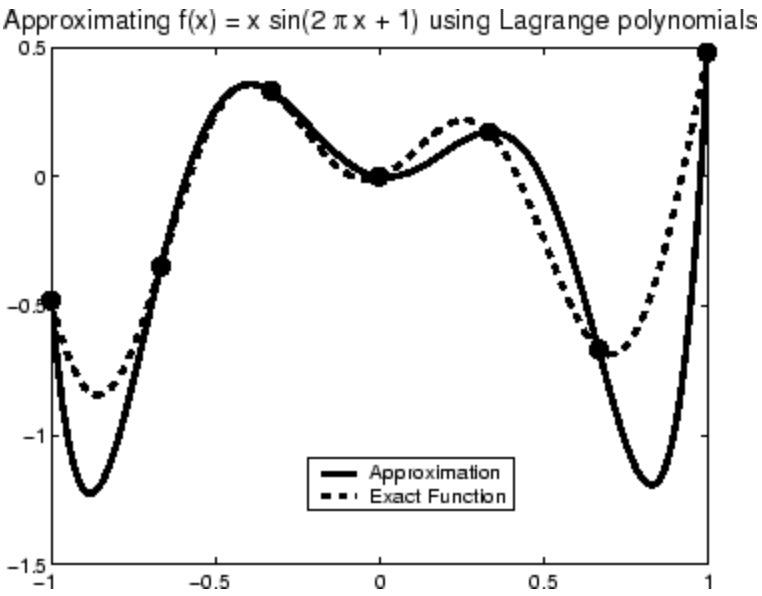
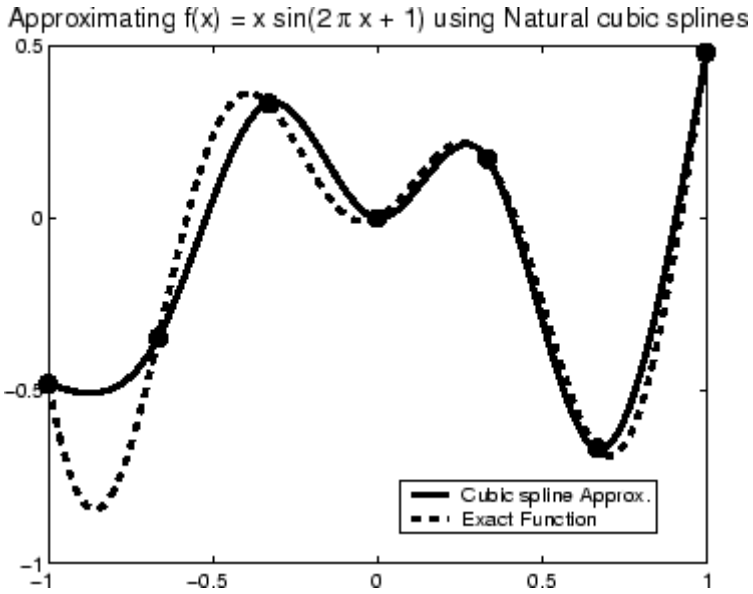


Figure: Approximation of $f(x) = x\sin(2\pi x + 1)$

using a sixth order Lagrange polynomial.

In both cases we are using the same knots. The derivative information for the clamped spline case was easy to derive from the function itself. However, we emphasize that this information is not always readily available. The result of the free conditions at the end points for the cubic spline clearly show an effect on the error near the end points.



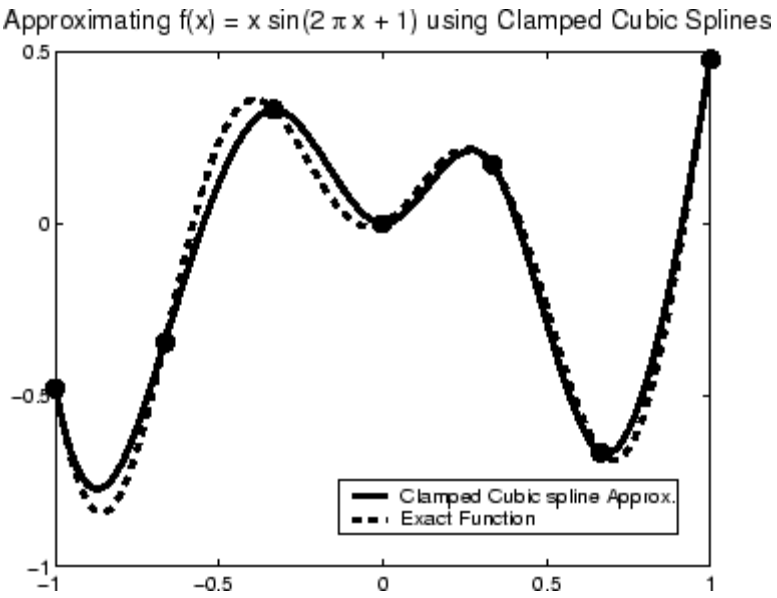


Figure: Approximation of $f(x) = x\sin(2\pi x + 1)$ using cubic splines. The first figure uses the natural cubic spline end conditions. The second figure uses the clamped cubic spline conditions.

---

previc

Next: Trigonometric Interpolation **Up:** Spline Interpolation **Previous:** Piecewise Linear Case
Juan Restrepo 2003-04-12