Module 6 – Spline Interpolation

Lesson goals

1. Understanding that splines minimize oscillations by fitting lower-order polynomials to data in a piecewise fashion.
2. Recognizing why cubic polynomials are preferable to quadratic and higher-order splines.
3. Knowing how to fit a spline to data with MATLAB's built-in functions.

Introduction

There are cases where the polynomial interpolating functions can lead to erroneous results because of round off error and oscillations. In fact, high-degree polynomials can oscillate erratically, that is, a minor fluctuation over a small portion of the interval can induce large fluctuations over the entire range.

An alternative approach is to apply lower-order polynomials in a piecewise fashion to subsets of data points. Such connecting polynomials are called ***spline functions***. For example, third-order curves employed to connect each pair of data points are called *cubic splines*. The key idea in constructing these functions is that the connections between adjacent cubic equations are **visually smooth**.

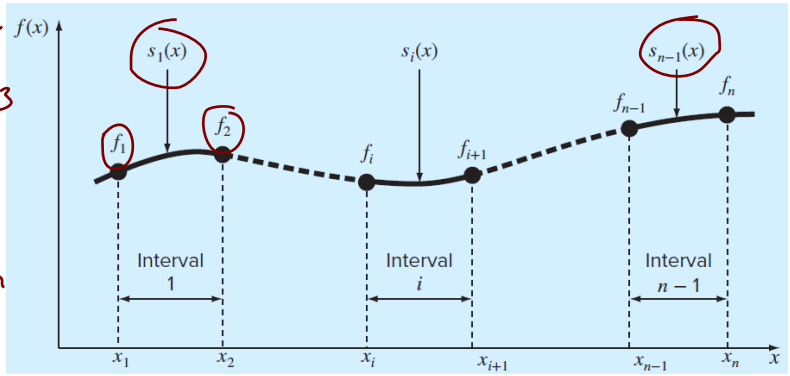
Linear Spline

For n data points,

$$(x_1, f_1), (x_2, f_2), \dots, (x_n, f_n)$$

there are $n-1$ subintervals $[x_i, x_{i+1}]$, for $i = 1,2, \dots, n$. Each i-th subinterval $[x_i, x_{i+1}]$ has its own spline function, $s_i(x)$.

$$S(x) = \begin{cases} S_1(x) \\ S_2(x) \\ \\ S_{n-1}(x) \end{cases} \qquad \begin{matrix} x_1 \le x \le x_2 \\ x_2 \le x \le x_3 \\ \vdots \\ x_{n-1} \le x \le x_n \end{matrix}$$



The simplest spline approximation is piecewise-linear interpolation, is merely the straight line connecting the two points at each end of the interval, which is formulated as

$$s_i(x) = a_i + b_i(x - x_i)$$

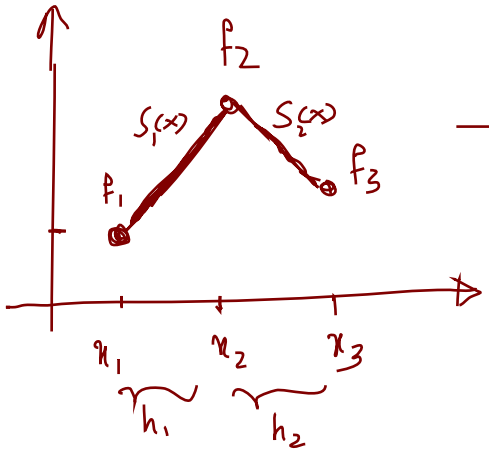
$$\begin{aligned} S_1 &= a_1 + b_1(x - x_1) \\ S_2 &= a_2 + b_2(x - x_2) \\ &\vdots \\ S_{n-1} &= a_{n-1} + b_{n-1}(x - x_{n-1}) \end{aligned}$$

where a_i is the y-intercept, which is defined as

$$a_i = f_i$$

and b_i is the slope of the straight line connecting the points:

$$b_i = \frac{f_{i+1} - f_i}{x_{i+1} - x_i}$$



$$\begin{cases} S_1(x) = a_1 + b_1(x - x_1) \Rightarrow \begin{cases} S_1(x_1) = f_1 \\ S_1(x_2) = f_2 \end{cases} \\ S_2(x) = a_2 + b_2(x - x_2) \Rightarrow \begin{cases} S_2(x_2) = f_2 \\ S_2(x_3) = f_3 \end{cases} \end{cases}$$

$$\begin{cases} a_1 = f_1 \\ a_1 + b_1(x_2 - x_1) = f_2 \\ a_2 = f_2 \\ a_2 + b_2(x_3 - x_2) = f_3 \end{cases} \longmapsto \begin{matrix} b_1 = \dfrac{f_2 - f_1}{x_2 - x_1} = f[x_2, x_1] \\ b_2 = \dfrac{f_3 - f_2}{x_3 - x_2} = f[x_3, x_2] \end{matrix} \longrightarrow S(x) = \begin{cases} f_1 + f[x_2, x_1](x - x_1) & \text{for } x_1 \le x \le x_2 \\ f_2 + f[x_3, x_2](x - x_2) & \text{for } x_2 \le x \le x_3 \end{cases}$$

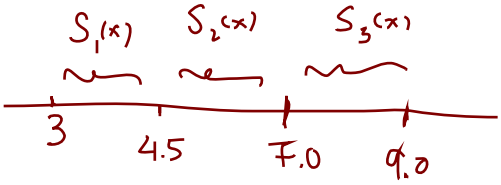
Example. Fit the data in the following table with first-order splines. Evaluate the Function at $x = 5$.

i	x_i	f_i
1	3.0	2.5
2	4.5	1.0
3	7.0	2.5
4	9.0	0.5

$S_1(x) = a_1 + b_1(x-3)$

$S_2(x) = a_2 + b_2(x-4.5)$

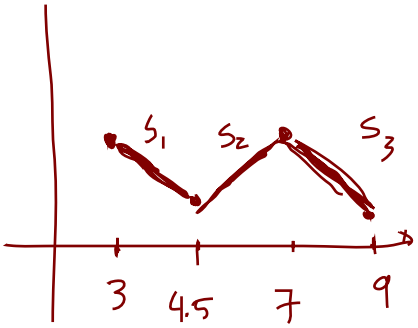
$S_3(x) = a_3 + b_3(x-7)$



$\begin{cases} S_1(3) = f_1 = 2.5 \mapsto a_1 = 2.5 \\ S_1(4.5) = f_2 = 1 \mapsto a_1 + b_1(4.5-3) = 1 \mapsto b_1 = -1 \end{cases}$ ✓

$\begin{cases} S_2(4.5) = 1 \mapsto a_2 = 1 \\ S_2(7) = 2.5 \mapsto a_2 + b_2(7-4.5) = 2.5 \mapsto b_2 = 0.6 \end{cases}$

$\begin{cases} S_3(7) = 2.5 \mapsto a_3 = 2.5 \\ S_3(9) = 0.5 \mapsto a_3 + b_3(9-7) = 0.5 \mapsto b_3 = -1 \end{cases}$



$$S(x) = \begin{cases} 2.5 - (x-3) & 3 \le x \le 4.5 \\ 1 + 0.6(x-4.5) & 4.5 \le x \le 7 \\ 2.5 - (x-7) & 7 \le x \le 9 \end{cases}$$

$f(5) = S(5) = 1 + 0.6 \times 0.5 = 1.3$

Observation. The primary disadvantage of first-order splines is that they are not *smooth*. In essence, at the data points where two splines meet (called a *knot*), the slope changes abruptly. In formal terms, the *first derivative of the function is discontinuous* at these points. This deficiency is overcome by using higher-order polynomial splines that ensure smoothness at the knots by equating derivatives at these points.

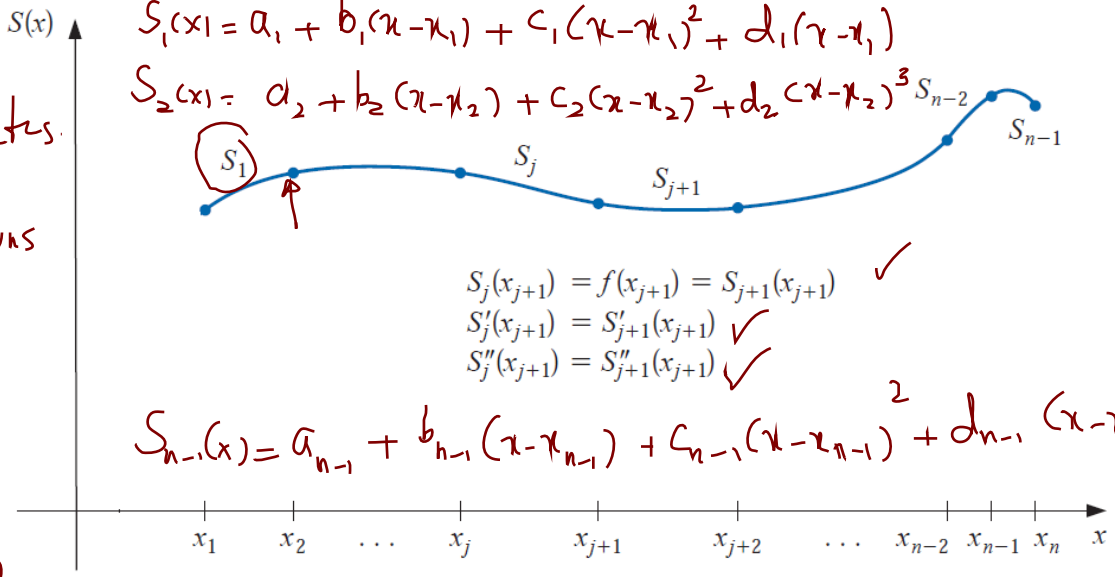
Cubic Splines

The most common piecewise-polynomial approximation uses cubic polynomials between each successive pair of nodes and is called cubic spline interpolation.

The objective in cubic splines is to derive a third-order polynomial for each interval between knots as represented generally by

$$s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

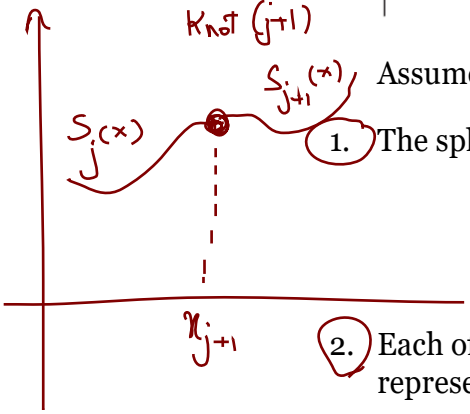
There are $n - 1$ intervals and $4(n - 1)$ unknown coefficients to evaluate. Consequently, the following $4(n - 1)$ conditions are required for their evaluation:



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

$S'_j(x_{j+1}) = S'_{j+1}(x_{j+1})$

$S''_j(x_{j+1}) = S''_{j+1}(x_{j+1})$



Assume that $h_i = x_{i+1} - x_i$, for $i = 1, 2, \ldots, n - 1$.

1. The spline must pass through all the data points:

$$s_i(x_i) = f_i \rightarrow a_i = f_i \quad i = 1,2, \ldots, n - 1$$

2. Each of the cubic functions must join at the knots. For knot $i + 1$, this can be represented as

$$s_i(x_{i+1}) = f_{i+1} \rightarrow a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 = f_{i+1} \quad i = 1,2, \ldots, n - 2$$

where $h_i = x_{i+1} - x_i$.

3. The first derivatives at the interior nodes must be equal. That is,

$$s_i'(x_{i+1}) = s_{i+1}'(x_{i+1}) \rightarrow b_i + 2c_i h_i + 3d_i h_i^2 = b_{i+1} \qquad i = 1,2,\dots,n-2$$

4. The second derivatives at the interior nodes must be equal. That is,

$$s_i''(x_{i+1}) = s_{i+1}''(x_{i+1}) \rightarrow c_i + 3d_i h_i = c_{i+1} \qquad i = 1,2,\dots,n-2$$

After doing multiple substitutions, we can come up with a system of equations in terms of c_i's (as below) which further enables us to compute the other coefficients by backward substitution.

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = 3(f[x_{i+1}, x_i] - f[x_i, x_{i-1}]) \qquad i = 2,3,\dots,n-2$$

This system has $n-3$ simultaneous tridiagonal equations with $n-1$ unknown coefficients, $c_1, c_2, \dots, c_{n-1}$. Therefore, if we have two additional conditions, we can solve for the unique c's.

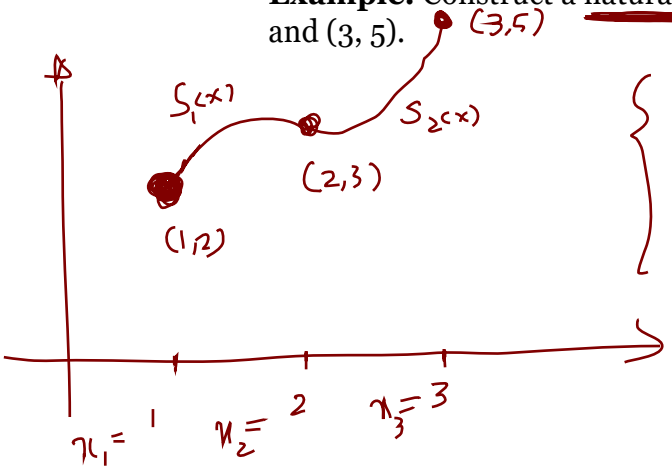
Boundary conditions: Usually, we assume that one of the following sets of boundary conditions is satisfied:

- $s''(x_1) = s''(x_n) = 0$ (**natural (or *free*) boundary**); When the free boundary conditions occur, the spline is called a ***natural spline***.
- $s'(x_1) = f_1'$ and $s'(x_n) = f_n'$ (**Clamped boundary**);

Now, using the natural boundary, the system of equations for unique solution c can be represented as below:

$$\begin{bmatrix} 1 & & & & \\ h_1 & 2(h_1+h_2) & h_2 & & \\ & \ddots & \ddots & \ddots & \\ & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} = \begin{bmatrix} 0 \\ 3(f[x_3, x_2] - f[x_2, x_1]) \\ \vdots \\ 3(f[x_n, x_{n-1}] - f[x_{n-1}, x_{n-2}]) \\ 0 \end{bmatrix}$$

Example. Construct a natural cubic spline that passes through the points (1, 2), (2, 3), and (3, 5).



$$S_1(x) = a_1 + b_1(x-1) + c_1(x-1)^2 + d_1(x-1)^3$$

$$S_2(x) = a_2 + b_2(x-2) + c_2(x-2)^2 + d_2(x-2)^3$$

$$S_1'(x) = b_1 + 2c_1(x-1) + 3d_1(x-1)^2$$

$$S_1''(x) = 2c_1 + 6d_1(x-1) \qquad S_2''(x) = 2c_2 + 6d_2(x-2)$$

(1) $S_1(1) = 2 \Rightarrow a_1 = 2$, $S_2(2) = 3 \Rightarrow a_2 = 3$, $S_2(3) = 5 \Rightarrow a_2 + b_2 + c_2 + d_2 = 5$

(2) $S_1(2) = S_2(2) \Rightarrow a_1 + b_1 + c_1 + d_1 = a_2 \Rightarrow b_1 + c_1 + d_1 = 1$

(3) $S_1'(2) = S_2'(2) \Rightarrow b_1 + 2c_1 + 3d_1 = b_2 \Rightarrow b_1 - b_2 + 2c_1 + 3d_1 = 0$

(4) $S_1''(2) = S_2''(2) \Rightarrow 2c_1 + 6d_1 = 2c_2 \Rightarrow 2c_1 + 6d_1 - 2c_2 = 0$

Natural Conditions

$S''(1) = 0 \Rightarrow 2c_1 = 0 \rightarrow c_1 = 0$

$S''(3) = 0 \Rightarrow 2c_2 + 6d_2 = 0$

$$\Rightarrow \begin{cases} b_2 + c_2 + d_2 = 2 \\ b_1 + d_1 = 1 \\ b_1 - b_2 + 3d_1 = 0 \\ 6d_1 - 2c_2 = 0 \\ 2c_2 + 6d_2 = 0 \end{cases}$$

$b_1 = 1 - \frac{1}{3}c_2$

$b_2 = 1 + \frac{2}{3}c_2$

$d_1 = \frac{1}{3}c_2$

$d_2 = -\frac{1}{3}c_2$

$b_2 + c_2 + d_2 = 2 \Rightarrow 1 + \frac{2}{3}c_2 + c_2 - \frac{1}{3}c_2 = 2$

$\Rightarrow c_2 = 3/4$

$\Rightarrow \begin{cases} b_1 = 3/4 & d_1 = 1/4 \\ b_2 = 3/2 & d_2 = -1/4 \end{cases}$

$$S(x) = \begin{cases} 2 + \frac{3}{4}(x-1) + \frac{1}{4}(x-1)^3 & 1 \le x \le 2 \\ 3 + \frac{3}{2}(x-2) + \frac{3}{4}(x-2)^2 - \frac{1}{4}(x-2)^3 & 2 \le x \le 3 \end{cases}$$

References

1. Chapra, Steven C. (2018). *Numerical Methods with* MATLAB *for Engineers and Scientists*, 4th Ed. McGraw Hill.
2. Burden, Richard L., Faires, J. Douglas (2011). *Numerical Analysis*, 9th Ed. Brooks/Cole Cengage Learning