# Interpolation by Splines

**KEY WORDS.** interpolation, polynomial interpolation, spline.

## GOAL.

- Understand what splines are
- Why the spline is introduced
- Approximating functions by splines

We have seen in previous lecture that a function $f(x)$ can be interpolated at $n+1$ points in an interval $[a,b]$ using a single polynomial $p_n(x)$ defined over the entire interval. The error estimate of polynomial interpolation gives

$$|f(x) - p_n(x)| \leq \frac{|\omega_{n+1}(x)|}{(n+1)!}|f^{(n+1)}(\xi_x)|,$$

where $\xi_x$ is a point depending on $x$ and the interpolation points $\{x_i\}_{i=0}^n$. Since the term $\left|\omega_{n+1}(x)\right|$ in the bound is the product of $n+1$ linear factors $|x - x_i|$, each being the distance between two points that both lie in $[a,b]$, we have $|x - x_i| \leq |b - a|$ and so

$$\max_{x\in[a,b]} |f(x) - p_n(x)| \leq |b-a|^{n+1} \frac{\max_{x\in[a,b]} |f^{(n+1)}(x)|}{(n+1)!}. \tag{1}$$

This bound suggests that we can make the error as small as we wish by freezing the value of $n$ and then reducing the size of $|b - a|$. But we still need an approximation over the original interval $[a,b]$, so we use a *piecewise polynomial*, or *spline*, approximation, in which the original interval is divided into non-overlapping subintervals and a different polynomial fit of the data is used on each subinterval.

## 1 Splines

To draw smooth curves through data points, drafters once used thin flexible strips of wood, hard rubber, metal or plastic called mechanical splines. To use a mechanical spline, pins were placed at a judicious selection of points along a curve in a design, then the spline was bent so that it touched each of these pins. Clearly, with this construction, the spline *interpolates* the curve at these pins and could be used to reproduce the curve in other drawings. The location of the pins are called *knots*. We can change the shape of the curve defined by the spline by adjusting the location of the knots. For example, to interpolate the data $\{(x_i, f_i)\}$ we can place knots at each of the nodes $x_i$.

Mathematically, a **spline function** consists of polynomial pieces on subintervals joined together with certain continuity conditions. Formally, suppose that $n+1$ points $x_0, x_1, \cdots, x_n$ have been specified and satisfy $x_0 < x_1 < \cdots < x_n$. These points are called **knots**. Suppose also that an integer $k \geq 0$ has been prescribed. A **spline function of degree** $k$ having knots $x_0, x_1, \cdots, x_n$ is a function $S$ such that:

1. on each interval $[x_{i-1}, x_i]$, $S$ is a polynomial of degree $\leq k$;
2. $S$ has a continuous $(k-1)^{st}$ derivative on $[x_0, x_n]$.

Hence, $S$ is a **piecewise polynomial** of degree at most $k$ having continuous derivatives of all orders up to $k-1$.

A spline of degree 1 is piecewise linear and has the form

$$S(x) = \begin{cases} p_1(x) = a_1 + b_1 x, & x \in [x_0, x_1), \\ p_2(x) = a_2 + b_2 x, & x \in [x_1, x_2), \\ \vdots \\ p_n(x) = a_n + b_n x, & x \in [x_{n-1}, x_n], \end{cases}$$

In this lecture, we will only consider spline interpolation using linear splines (splines of degree 1), quadratic splines (splines of degree 2), and cubic splines (splines of degree 3). Generalization to splines of general order is relatively straightforward.

# 2 Linear Interpolating Splines

A simple piecewise polynomial fit is the continuous linear interpolating spline. For the data set

| x | $x_0$ | $x_1$ | $\cdots$ | $x_n$ |
|---|---|---|---|---|
| y | $f_0$ | $f_1$ | $\cdots$ | $f_n$ |

where

$$a = x_0 < x_1 < \ldots < x_n = b,$$

the linear spline $S_{1,n}(x)$ is a **continuous function** that interpolates the data and is constructed from linear functions that are two–point interpolating polynomials:

$$S_{1,n}(x) = \begin{cases} p_1(x) = f_0 \frac{x-x_1}{x_0-x_1} + f_1 \frac{x-x_0}{x_1-x_0}, & x \in [x_0, x_1], \\ p_2(x) = f_1 \frac{x-x_2}{x_1-x_2} + f_2 \frac{x-x_1}{x_2-x_1}, & x \in [x_1, x_2], \\ \vdots \\ p_n(x) = f_{n-1} \frac{x-x_n}{x_{n-1}-x_n} + f_n \frac{x-x_{n-1}}{x_n-x_{n-1}}, & x \in [x_{n-1}, x_n]. \end{cases} \tag{2}$$

Note that

$$p_1(x_0) = f_0, p_1(x_1) = p_2(x_1) = f_1, \ldots, p_{n-1}(x_{n-1}) = p_n(x_{n-1}) = f_{n-1}, p_n(x_n) = f_n.$$

From the bound on the error for polynomial interpolation in the case of an interpolating polynomial of degree one, we have

$$\max_{\bar{x}\in[x_{i-1},x_i]} |f(\bar{x})-S_{1,n}(\bar{x})| \le \frac{|x_i - x_{i-1}|^2}{2!} \cdot \max_{x\in[x_{i-1},x_i]} |f^{(2)}(x)| \le \frac{h^2}{2} \max_{x\in[a,b]} |f^{(2)}(x)|,$$

where $h = \max_{1\le i\le n} |x_i - x_{i-1}|$. That is, as the maximum interval length $h \to 0$, the bound on the maximum absolute error behaves like $Ch^2$, where $C$ is a positive constant independent of $h$.

Suppose that the nodes are chosen to be equally spaced in $[a, b]$, so that $x_i = a + ih,\ \ i = 0, 1, \ldots, n$, where $h \equiv (b - a)/n$. As the number of points $n$ increases, the error in using $S_{1,n}(\bar{x})$ as an approximation to $f(\bar{x})$ tends to zero like $1/n^2$.

**Example:** We construct the linear spline interpolating the data

| x | $-1$ | 0 | 1 |
|---|---|---|---|
| y | 0 | 1 | 3 |

from (2), as follows:

$$S_{1,2}(x) = \begin{cases} 0 \cdot \frac{x-0}{(-1)-0} + 1 \cdot \frac{x-(-1)}{0-(-1)}, & x \in [-1, 0], \\ \\ 1 \cdot \frac{x-1}{0-1} + 3 \cdot \frac{x-0}{1-0}, & x \in [0, 1], \end{cases} \tag{3}$$

$$\tag{4}$$

$$\tag{5}$$

$$= \begin{cases} x + 1, & x \in [-1, 0], \\ \\ 2x + 1, & x \in [0, 1]. \end{cases} \tag{6}$$

An alternative way of representing a linear spline uses a **linear spline basis**, $\ell_i(x),\ \ i = 0, 1, \ldots, n$, chosen so that

$$\ell_i(x_j) = \begin{cases} 0, & j \ne i \\ 1, & j = i. \end{cases} \tag{7}$$

Here, each $\ell_i(x)$ is a "roof" shaped function with the apex of the roof at the point $(x_i, 1)$ and the span on the interval $[x_{i-1}, x_{i+1}]$, and with $\ell_i(x) \equiv 0$ outside $[x_{i-1}, x_{i+1}]$. That is,

$$\ell_i(x) = \begin{cases} \frac{x-x_{i-1}}{x_i-x_{i-1}}, & x \in [x_{i-1}, x_i], \\ \\ \frac{x_{i+1}-x}{x_{i+1}-x_i}, & x \in [x_i, x_{i+1}], \\ \\ 0, & \text{for all other } x. \end{cases} \tag{8}$$

In terms of the linear spline basis, we can write

$$S_{1,n}(x) = \sum_{i=0}^{n} f_i \ell_i(x). \tag{9}$$

Then, using (7), we have

$$S_{1,n}(x_j) = \sum_{i=0}^{n} f_i \ell_i(x_j) = f_j, \quad j = 0, \ldots, n.$$

**Example:** To construct the linear spline interpolating the data

| x | $-1$ | 0 | 1 |
|---|---|---|---|
| y | 0 | 1 | 3 |

using the linear spline basis, we first construct the basis functions as follows:

$$\ell_0(x) = \begin{cases} \frac{0-x}{0-(-1)}, & x \in [-1,0], \\ 0, & x \in [0,1], \end{cases} = \begin{cases} -x, & x \in [-1,0], \\ 0, & x \in [0,1], \end{cases}$$

$$\ell_1(x) = \begin{cases} \frac{x-(-1)}{0-(-1)}, & x \in [-1,0], \\ \frac{1-x}{1-0}, & x \in [0,1], \end{cases} = \begin{cases} x+1, & x \in [-1,0], \\ 1-x, & x \in [0,1], \end{cases}$$

$$\ell_2(x) = \begin{cases} 0, & x \in [-1,0], \\ \frac{x-0}{1-0}, & x \in [0,1], \end{cases} = \begin{cases} 0, & x \in [-1,0], \\ x, & x \in [0,1]. \end{cases}$$

Then we obtain our linear spline interpolant,

$$p_2(x) = 0 \cdot \ell_0(x) + 1 \cdot \ell_1(x) + 3 \cdot \ell_2(x).$$

Linear splines suffer from a major limitation: the derivative of a linear spline is generally discontinuous at each interior node $x_i$. To derive a piecewise polynomial approximation with a continuous derivative requires that we use piecewise polynomial pieces of higher degree and constrain the pieces to make the curve smoother.

# 3 Quadratic Interpolating Splines

To derive a mathematical model of a quadratic spline, suppose the data are $\{(x_i, f_i)\}_{i=0}^{n}$, where, as for linear splines,

$$a = x_0 < x_1 < \ldots < x_n = b, \quad h \equiv \max_i |x_i - x_{i-1}|.$$

A quadratic spline $S_{2,n}(x)$ is a $C^1$ piecewise quadratic polynomial. This means that:

- $S_{2,n}(x)$ is piecewise quadratic; that is, between consecutive knots $x_i$,

$$S_{2,n}(x) = \begin{cases} p_1(x) & = a_1 + b_1 x + c_1 x^2, \quad x \in [x_0, x_1], \\ p_2(x) & = a_2 + b_2 x + c_2 x^2, \quad x \in [x_1, x_2], \\ & \vdots \\ p_n(x) & = a_n + b_n x + c_n x^2, \quad x \in [x_{n-1}, x_n]; \end{cases}$$

- $S_{2,n}(x)$ is $C^1$; that is, $S_{2,n}(x)$ *is continuous and has continuous first derivative* everywhere in the interval $[a, b]$, in particular, at the knots.

For $S_{2,n}(x)$ to be an **interpolatory quadratic spline**, we must also have

- $S_{2,n}(x)$ interpolates the data, that is,

$$S_{2,n}(x_i) = f_i, \quad i = 0, 1, \ldots, n.$$

Within each interval $(x_{i-1}, x_i)$, the corresponding quadratic polynomial is continuous and has continuous derivatives of all orders. Therefore, $S_{2,n}(x)$ or one of its derivatives can be discontinuous only at a knot. Observe that the function $S_{2,n}(x)$ has two quadratic pieces incident at the interior knot $x_i$; to the left of $x_i$, it is a quadratic $p_i(x)$ while to the right it is a quadratic $p_{i+1}(x)$. Thus, a necessary and sufficient condition for $S_{2,n}(x)$ to have continuous first derivative is for these two quadratic polynomials incident at the interior knot to match in first derivative value. So we have a set of *smoothness conditions*: that is, at each interior knot,

$$p_i'(x_i) = p_{i+1}'(x_i), \quad i = 1, 2, \ldots, n-1.$$

In addition, to interpolate the data, we have a set on *interpolation conditions*: that is, on the $i$-th interval,

$$p_i(x_{i-1}) = f_{i-1}, \quad p_i(x_i) = f_i, \quad i = 1, 2, \ldots, n.$$

This way of writing the interpolation conditions also forces $S_{2,n}(x)$ to be continuous at the knots.

Since each of the $n$ quadratic pieces has three unknown coefficients, our description of the function $S_{2,n}(x)$ involves $3n$ unknown coefficients. Assuring continuity of the first derivative imposes $(n-1)$ linear constraints on its coefficients, and interpolation imposes an additional $2n$ linear constraints. Therefore, there are a total of $3n-1$ linear constraints on the $3n$ unknown coefficients. In order that we have the same number of equations as unknowns, we need 1 more (linear) constraints.

**Example 3.1.** Construct a quadratic spline interpolating

$$(-1, 0), \quad (0, 1), \quad (1, 3).$$

*Solution:* Assume that the quadratic spline $S_{2,2}(x)$ is of the form

$$p_1(x) = a_1 + b_1 x + c_1 x^2, \quad on \quad [-1, 0]$$

$$p_2(x) = a_2 + b_2 x + c_2 x^2, \quad on \quad [0, 1]$$

Since $S_{2,2}(x)$ interpolates the data points, we have

$$p_1(-1) = a_1 - b_1 + c_1 = 0 \tag{10}$$
$$p_1(0) = a_1 = 1 \tag{11}$$
$$p_2(0) = a_2 = 1 \tag{12}$$
$$p_2(1) = a_2 + b_2 + c_2 = 3 \tag{13}$$

In addition, $(p_1(0))' = (p_2(0))'$ gives

$$b_1 = b_2. \tag{14}$$

Above are the five linear equations about the six unknown coefficients $a_1, b_1, c_1, a_2, b_2, c_2$, which have infinitely many solutions. If we impose one more constrain, for example, $(S_{2,N}(-1))' = 0$,

$$b_1 - 2c_1 = 0.$$

Then there are six linear equations for six unknown coefficients, from which it can be solved that

$$a_1 = 1, \quad b_1 = 2, \quad c_1 = 1, \quad a_2 = 1, \quad b_2 = 2, \quad c_2 = 0.$$

Therefore the quadratic spline function

$$S_{2,2}(x) = \begin{cases} 1 + 2x + x^2, & on \quad [-1, 0] \\ 1 + 2x \quad on & [0, 1]. \end{cases}$$

# 4 An example: the construction of a complete interpolatory quadratic spline

To construct an interpolatory quadratic spline, we first define the numbers

$$z_i = S'_{2,n}(x_i), \quad 0 \le i \le n.$$

Since $S_{2,n}(x)$ is a quadratic spline, $S'_{2,n}(x)$ is a linear spline (by checking the definitions). Therefore $S'_{2,n}(x)$ is given by the straight line joining the points $(x_{i-1}, z_{i-1})$ and $(x_i, z_i)$:

$$p'_i(x) = z_{i-1} + \frac{z_i - z_{i-1}}{h_i}(x - x_{i-1}),$$

where $h_i \equiv x_i - x_{i-1}$. If this expression is integrated once, we obtain

$$p_i(x) = z_{i-1}(x - x_{i-1}) + \frac{z_i - z_{i-1}}{2h_i}(x - x_{i-1})^2 + C_i \tag{15}$$

where $C_i$ is constant of integration. The interpolation conditions $p_i(x_{i-1}) = f(x_{i-1})$ and $p_i(x_i) = f(x_i)$ give

$$C_i = f(x_{i-1}), \quad i = 1, 2, \cdots n \tag{16}$$

and

$$z_{i-1}h_i + \frac{z_i - z_{i-1}}{2}h_i + C_i = f(x_i), \quad i = 1, 2, \cdots n.$$

or equivalently

$$\frac{z_i + z_{i-1}}{2}h_i = f(x_i) - f(x_{i-1}), \quad i = 1, 2, \cdots n. \tag{17}$$

The equation (17) is a system of $n$ linear equations for the $n + 1$ unknowns $z_0, z_1, \cdots, z_n$. To obtain one additional equation, we apply the endpoint derivative conditions, e.g. $p_1'(x_0) = 0$ which gives

$$z_0 = 0.$$

Now we obtain the system of linear equations,

$$A\mathbf{z} = \mathbf{d}, \tag{18}$$

where

$$\mathbf{z} = [z_0, z_1, \cdots, z_n]^T, \quad \mathbf{d} = [d_0, d_1, \cdots, d_n]^T,$$

with

$$d_i = \begin{cases} 0, & i = 0, \\ \frac{2(f(x_i) - f(x_{i-1}))}{x_i - x_{i-1}}, & 1 \leq i \leq n, \end{cases}$$

and $A$ is the *tridiagonal* matrix

$$A = \begin{bmatrix} 1 & 0 & & & \\ 1 & 1 & 0 & & \\ 0 & 1 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & & & \\ & & 0 & 1 & 1 \end{bmatrix}.$$

The system can be solved, therefore the quadratic spline function (15).

**Example 4.1.** Again, we consider to construct a quadratic spline interpolating

$$(-1, 0), \quad (0, 1), \quad (1, 3),$$

with $S_{2,2}'(x = -1) = 0$.
*Solution:* Let $\vec{z} = (z_0, z_1, z_2)'$, where $z_i = S_{2,2}'(x = x_i)$. Then $\vec{z}$ satisfies the linear equations,

$$A\mathbf{z} = \mathbf{d},$$

with the matrix A is

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

and

$$\mathbf{d} = (0, 2, 4)'.$$

Therefore, $\vec{z} = (0, 2, 2)'$. Pluging in $\vec{z}$ and $C_1 = 0$, $C_2 = 1$ (see equation (16)) into equation (15), gives

$$S_{2,2}(x) = \begin{cases} 1 + 2x + x^2, & on \quad [-1, 0] \\ 1 + 2x \quad on & [0, 1]. \end{cases}$$

It is observed that both approaches in this and previous example give the same quadratic spline interpolation.

# 5 Cubic Interpolating Splines

To derive a mathematical model of a cubic spline, suppose the data are $\{(x_i, f_i)\}_{i=0}^{n}$, where, as for linear splines,

$$a = x_0 < x_1 < \ldots < x_n = b, \quad h \equiv \max_i |x_i - x_{i-1}|.$$

A cubic spline $S_{3,n}(x)$ is a $C^2$ piecewise cubic polynomial. This means that:

- $S_{3,n}(x)$ is piecewise cubic; that is, between consecutive knots $x_i$,

$$S_{3,n}(x) = \begin{cases} p_1(x) & = a_1 + b_1 x + c_1 x^2 + d_1 x^3, \quad x \in [x_0, x_1], \\ p_2(x) & = a_2 + b_2 x + c_2 x^2 + d_2 x^3, \quad x \in [x_1, x_2], \\ & \vdots \\ p_n(x) & = a_n + b_n x + c_n x^2 + d_n x^3, \quad x \in [x_{n-1}, x_n]; \end{cases}$$

- $S_{3,n}(x)$ is $C^2$; that is, $S_{3,n}(x)$ *is continuous and has continuous first and second derivatives* everywhere in the interval $[a, b]$, in particular, at the knots.

For $S_{3,n}(x)$ to be an **interpolatory cubic spline**, we must also have

- $S_{3,n}(x)$ interpolates the data, that is,

$$S_{3,n}(x_i) = f_i, \quad i = 0, 1, \ldots, n.$$

Within each interval $(x_{i-1}, x_i)$, the corresponding cubic polynomial is continuous and has continuous derivatives of all orders. Therefore, $S_{3,n}(x)$ or one of its derivatives can be discontinuous only at a knot. Observe that the function $S_{3,n}(x)$ has two cubic pieces incident at the interior knot $x_i$; to the left of $x_i$, it is a cubic $p_i(x)$ while to the right it is a cubic $p_{i+1}(x)$. Thus, a necessary and sufficient condition for $S_{3,n}(x)$ to be continuous and have continuous first and second derivatives is for these two cubic polynomials incident at the interior knot to match in value, and in first and second derivative values. So we have a set of *smoothness conditions*: that is, at each interior knot,

$$p_i'(x_i) = p_{i+1}'(x_i), \quad p_i''(x_i) = p_{i+1}''(x_i), \quad i = 1, 2, \ldots, n-1.$$

In addition, to interpolate the data, we have a set on *interpolation conditions*: that is, on the $i$-th interval,

$$p_i(x_{i-1}) = f_{i-1}, \quad p_i(x_i) = f_i, \quad i = 1, 2, \ldots, n.$$

This way of writing the interpolation conditions also forces $S_{3,n}(x)$ to be continuous at the knots.

Since each of the $n$ cubic pieces has four unknown coefficients, our description of the function $S_{3,n}(x)$ involves $4n$ unknown coefficients. Assuring continuity of this function and its first and second derivatives imposes $3(n-1)$ linear constraints on its coefficients, and interpolation imposes an additional $n+1$ linear constraints. Therefore, there are a total of $4n - 2n = 3(n-1) + (n+1)$ linear constraints on the $4n$ unknown coefficients. In order that we have the same number of equations as unknowns, we need 2 more (linear) constraints.

There are various ways of specifying two additional constraints such as the following.

1. The so–called **natural cubic spline** results when we impose the conditions
$$p_1''(x_0) = 0, \quad p_n''(x_n) = 0.$$
The natural cubic spline is seldom used since it does not provide a sufficiently accurate approximation $S_{3,n}(x)$ near the ends of the interval $[a, b]$. This may be anticipated from the fact that we are forcing zero values on the second derivatives when these are not necessarily the values of the second derivative of the function which the data measures. A natural cubic spline is built up from cubic polynomials, so it is reasonable to expect that if the data come from a cubic polynomial, then the natural cubic spline $S_{3,n}(x)$ will reproduce the cubic polynomial. However, for example, if the data are measured from the function $f(x) = x^2$ then the natural spline $S_{3,n}(x) \neq f(x)$; the function $f(x) = x^2$ has nonzero second derivatives at the nodes $x_0$ and $x_n$ where the value of the second derivative of the natural cubic spline $S_{3,n}(x)$ is zero by definition.

2. Instead of imposing the natural cubic spline conditions, we could use the correct second derivative values:
$$p_1''(x_0) = f''(x_0), \quad p_n''(x_n) = f''(x_n).$$
These second derivative values of the data are not usually available but they can be replaced by accurate approximations.

3. A simpler, sufficiently accurate spline is determined using the so–called **not–a–knot** conditions. Recall that at each knot the spline $S_{3,n}(x)$ changes from one cubic to the next. The idea of the not–a–knot conditions is *not to change* cubic polynomials as one crosses both the first and last *interior* nodes, $x_1$ and $x_{n-1}$. [Then $x_1$ and $x_{N-1}$ are not knots.] These conditions are expressed mathematically as the not–a–knot conditions

$$p_1'''(x_1) = p_2'''(x_1), \quad p_{n-1}'''(x_{n-1}) = p_n'''(x_{n-1}).$$

   By construction, the first two pieces, $p_1(x)$ and $p_2(x)$, of the cubic spline $S_{3,n}(x)$ agree in value, first, and second derivative at $x_1$. If $p_1(x)$ and $p_2(x)$ also satisfy the not–a–knot condition at $x_1$, it follows that $p_1(x) \equiv p_2(x)$; that is, $x_1$ is no longer a knot.

4. In the **complete cubic spline**, the slope conditions

$$p_1'(x_0) = f'(x_0), \quad p_n'(x_n) = f'(x_n) \tag{19}$$

   are imposed. These first derivative values of the data may not be readily available but they can be replaced by accurate approximations.

# 6 An example: the construction of a complete interpolatory cubic spline

(Note: the notations below is a little bit different from what we adopt in the class. However, the ideas behind are the same.)

As an example of the construction of an interpolatory cubic spline, we consider case 4 of section 4.5, that of the complete cubic spline. First, we define the numbers $z_i = S''_{3,n}(x_i)$, $0 \le i \le n$. (The complete interpolatory cubic spline will be defined in terms of $z_i$, $0 \le i \le n$.) Since $S''_{3,n}$ is continuous at $x_i$, $1 \le i \le n-1$, the $z_i$ exist and, at each interior knot, satisfy

$$p_i''(x_i) = z_i = p_{i+1}''(x_i), \quad 1 \le i \le n-1.$$

Since $p_{i+1}(x)$ is a cubic polynomial on $[x_i, x_{i+1}]$, $p''_{i+1}$ is a linear function satisfying $p''_{i+1}(x_i) = z_i$ and $p''_i(x_{i+1}) = z_{i+1}$ and therefore is given by the straight line joining the points $(x_i, z_i)$ and $(x_{i+1}, z_{i+1})$:

$$p''_{i+1}(x) = \frac{z_i}{h_{i+1}}(x_{i+1} - x) + \frac{z_{i+1}}{h_{i+1}}(x - x_i),$$

where $h_{i+1} \equiv x_{i+1} - x_i$. If this expression is integrated twice, we obtain

$$p_{i+1}(x) = \frac{z_i}{6h_{i+1}}(x_{i+1} - x)^3 + \frac{z_{i+1}}{6h_{i+1}}(x - x_i)^3 + C(x - x_i) + D(x_{i+1} - x),$$

where $C$ and $D$ are constants of integration. The interpolation conditions $p_{i+1}(x_i) = f_i$ and $p_{i+1}(x_{i+1}) = f_{i+1}$ give

$$\frac{z_i}{6h_{i+1}}(x_{i+1} - x_i)^3 + D(x_{i+1} - x_i) = f_i$$

and

$$\frac{z_{i+1}}{6h_{i+1}}(x_{i+1}-x_i)^3 + C(x_{i+1}-x_i) = f_{i+1},$$

respectively, from which it follows that

$$C = \left(\frac{f_{i+1}}{h_{i+1}} - \frac{z_{i+1}h_{i+1}}{6}\right), \quad D = \left(\frac{f_i}{h_{i+1}} - \frac{z_i h_{i+1}}{6}\right).$$

Thus,

$$\begin{aligned} p_{i+1}(x) &= \frac{z_i}{6h_{i+1}}(x_{i+1}-x)^3 + \frac{z_{i+1}}{6h_{i+1}}(x-x_i)^3 + \left(\frac{f_{i+1}}{h_{i+1}} - \frac{z_{i+1}h_{i+1}}{6}\right)(x-x_i) \\ &\quad + \left(\frac{f_i}{h_{i+1}} - \frac{z_i h_{i+1}}{6}\right)(x_{i+1}-x). \end{aligned}$$

Once the values of $z_0, z_1, \cdots, z_n$ have been found, $S_{3,n}(x)$ can be evaluated for any $x$ in the interval $[x_0, x_n]$.

To determine $z_1, z_2, \cdots, z_{n-1}$, we use the continuity equations for $S'_{3,n}$. At an interior knot $x_i$, we must have $p'_i(x_i) = p'_{i+1}(x_i)$. Since

$$\begin{aligned} p'_{i+1}(x) &= -\frac{z_i}{2h_{i+1}}(x_{i+1}-x)^2 + \frac{z_{i+1}}{2h_{i+1}}(x-x_i)^2 + \left(\frac{f_{i+1}}{h_{i+1}} - \frac{z_{i+1}h_{i+1}}{6}\right) \\ &\quad - \left(\frac{f_i}{h_{i+1}} - \frac{z_i h_{i+1}}{6}\right), \end{aligned}$$

it follows that, after simplification,

$$p'_{i+1}(x_i) = -\frac{h_{i+1}}{3}z_i - \frac{h_{i+1}}{6}z_{i+1} - \frac{f_i}{h_{i+1}} + \frac{f_{i+1}}{h_{i+1}}, \tag{20}$$

and analogously

$$p'_i(x_i) = \frac{h_i}{6}z_{i-1} + \frac{h_i}{3}z_i - \frac{f_{i-1}}{h_i} + \frac{f_i}{h_i}. \tag{21}$$

When the right hand sides of (20) and (21) are set equal to each other, the result can be written as

$$h_i z_{i-1} + 2(h_i + h_{i+1})z_i + h_{i+1}z_{i+1} = \frac{6}{h_{i+1}}(f_{i+1}-f_i) - \frac{6}{h_i}(f_i - f_{i-1}), \quad 1 \le i \le n-1, \tag{22}$$

which is a system of $n-1$ linear equations for the $n+1$ unknowns $z_0, z_1, \cdots, z_n$.

To obtain two additional equations, we apply the endpoint derivative conditions $p'_1(x_0) = f'_0 \equiv f'(x_0)$ and $p'_n(x_n) = f'_n \equiv f'(x_n)$ of (19) to obtain, respectively,

$$2h_1 z_0 + h_1 z_1 = \frac{6}{h_1}(f_1 - f_0) - 6f'_0, \tag{23}$$

on using (20) with $i = 0$, and

$$h_n z_{n-1} + 2h_n z_n = 6f'_n - \frac{6}{h_n}(f_n - f_{n-1}), \tag{24}$$

on using (21) with $i = n$. Combining equations (23), (22), and (24), we obtain the system of linear equations,

$$A\mathbf{z} = \mathbf{d}, \tag{25}$$

where

$$\mathbf{z} = [z_0, z_1, \cdots, z_n]^T, \quad \mathbf{d} = [d_0, d_1, \cdots, d_n]^T,$$

with

$$d_i = \begin{cases} \frac{6}{h_1}(f_1 - f_0) - 6f_0', & i = 0, \\ \\ \frac{6}{h_{i+1}}(f_{i+1} - f_i) - \frac{6}{h_i}(f_i - f_{i-1}), & 1 \le i \le n-1, \\ \\ 6f_n' - \frac{6}{h_n}(f_n - f_{n-1}), & i = n, \end{cases}$$

and $A$ is the *tridiagonal* matrix

$$A = \begin{bmatrix} 2h_1 & h_1 & & & \\ h_1 & 2(h_1+h_2) & h_2 & & \\ & h_2 & 2(h_2+h_3) & h_3 & \\ & \ddots & \ddots & \ddots & \\ & & & & \\ & & h_{n-1} & 2(h_{n-1}+h_n) & h_n \\ & & & h_n & 2h_n \end{bmatrix}.$$

This system can be solved using the MATLAB function *tridisolve* available from

`www.mathworks.com/Moler`

# 7 The Error in Cubic Spline Interpolation

For each way of supplying the additional linear constraints discussed in the preceding, the system of $4n$ linear constraints has a unique solution as long as the knots are distinct. So, the cubic spline interpolant constructed using any one of the natural, the correct endpoint second derivative value, an approximated endpoint second derivative value, the not–a–knot conditions or the complete spline conditions is unique. This uniqueness result permits an estimate of the error associated with approximations by cubic splines from the error bound for polynomial interpolation,

$$\max_{x \in [x_{i-1}, x_i]} |f(x) - p_3(x)| \le \frac{h^4}{4!} \max_{x \in [a,b]} |f^{(4)}(x)|,$$

where $h = \max_i |x_i - x_{i-1}|$. We might anticipate that the error associated with approximation by a cubic spline would be $O(h^4)$ for $h$ small, as for the interpolating cubic polynomial. However, the maximum absolute error associated with the natural cubic spline approximation behaves like $O(h^2)$ as $h \to 0$. In contrast, the maximum absolute error for a cubic spline based on correct endpoint second derivative values, or on the not–a–knot conditions or the complete spline conditions behaves like $O(h^4)$. Unlike the natural cubic spline, the correct second derivative value, not–a–knot cubic splines and complete splines reproduce cubic polynomials. That is, in all of these cases, $S_{3,n}(x) \equiv f(x)$ on the interval $[a, b]$ whenever the data are measured from a cubic polynomial $f(x)$. This reproducibility property is a necessary condition for the cubic spline $S_{3,n}(x)$ to be an $O(h^4)$ approximation to a general function $f(x)$.