

Chapter 5

Interpolation

5.1 Information from Data

The topic of this chapter is interpolation, which relates to passing a curve through a set of data points. To put this in context, extracting information from data is one of the central objectives in science and engineering and exactly what or how this is done depends on the particular setting. Two examples are shown in Figure 5.1. Figure 5.1(L) contains data obtained from measurements of high redshift type supernovae. As is often the case with computerized testing systems, there are many data points and there is some scatter in the values obtained. Because of this one would not be interested in finding a function that passes through all of these points, but rather a function that behaves in a qualitatively similar manner as the data. In this case one uses a fitting method, like least squares, to make the connection more quantitative. Exactly how this might be done will be considered in Chapter 8.

In comparison, the data in Figure 5.1(R) have a well-defined shape and for this reason are more amenable to interpolation. This is also true for the data shown in Figure 5.2. The hand data is typical of what arises in CAD applications, while the data on the right relates to a more mathematical application. To explain, the data points are obtained by evaluating the function

$$f(x) = \frac{1}{3} + \sum_{n=1}^{\infty} \frac{4(-1)^n}{n^{4/3}\pi^2} \cos(n\pi x) \quad (5.1)$$

at 10 points from the interval $-1 \leq x \leq 1$. What is seen is that the above relatively complicated function does not have a correspondingly complicated graph. This raises the question if we might be able to replace the function with a much simpler expression that would serve as a respectable approximation of the original.

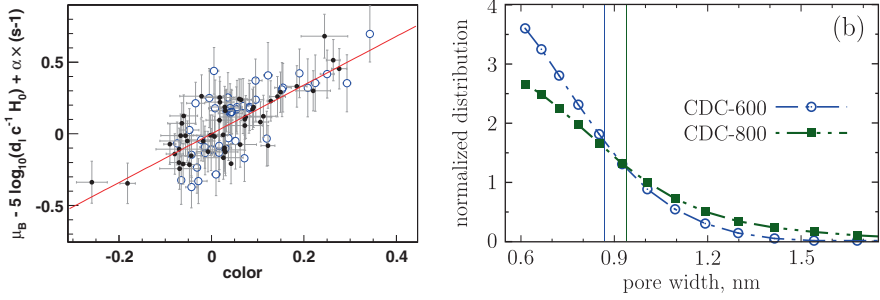


Figure 5.1 Left: data related to a supernovae redshift [Astier et al., 2006]. Right: data for nanopores in a supercapacitor [Kondrat et al., 2012].

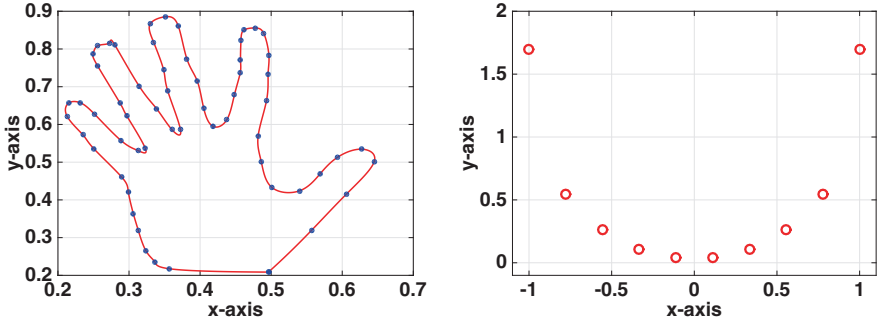


Figure 5.2 Left: geometric representation of a hand using interpolation [Burkardt, 2015]. Right: values of (5.1) at selected points in interval $-1 \leq x \leq 1$.

One of the better ways to test how well an interpolation method works is to try it out on different data sets. In what follows we will use the sets shown in Figure 5.3. Each consists of 12 equally spaced points over the interval $-1 \leq x \leq 1$. The top two were generated using functions; the one in the upper left comes from the 5th order polynomial

$$y(x) = (x + 0.9)(x + 0.1)^2(x - 0.2)(x - 0.8),$$

while the one on the upper right consists of points that lie on the circle $x^2 + y^2 = 1$. The lower two are used to mimic or resemble a periodic function and one with jumps. It is recommend that you spend a moment or two and sketch in what you think would be an acceptable interpolation function for each data set. This will help later when we see what the standard interpolation methods produce.

It is of interest to know that many of the interpolation methods derived in this chapter are summarized in Appendix C, Table C.1.

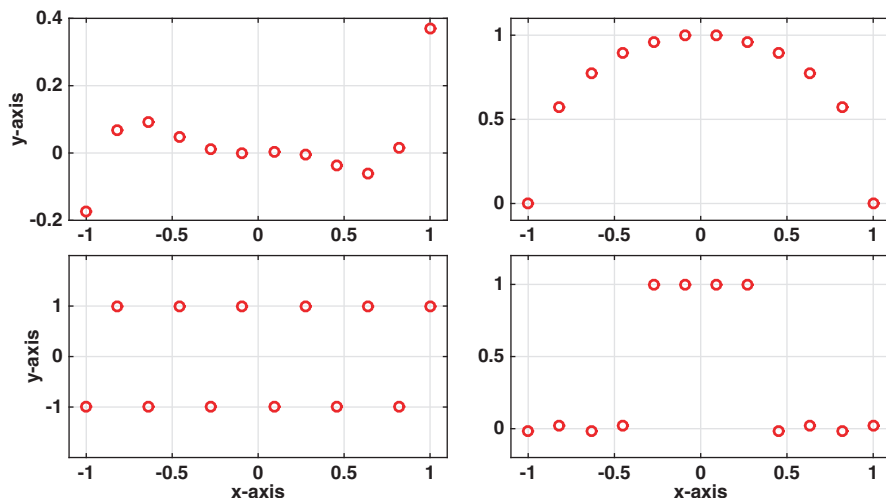


Figure 5.3 Data sets that are used to test the various interpolation methods.

5.2 Global Polynomial Interpolation

If one wants to find a function to connect two data points, the easiest choice is to use a straight line, in other words a linear function. Similarly, given three data points one would likely use a quadratic. To generalize this idea, suppose there are $n + 1$ points and they are $(x_1, y_1), (x_2, y_2), \dots, (x_{n+1}, y_{n+1})$, where $x_1 < x_2 < \dots < x_{n+1}$. We are going to find a single n th degree polynomial $p_n(x)$ that passes through each and every point. This is not particularly difficult and there are several ways to find the interpolation polynomial. However, as is often the case in numerical computing, some methods are much more sensitive to round-off error than other methods.

5.2.1 Direct Approach

Taking the direct approach, the simplest choice is to take

$$p_n(x) = a_0 + a_1x + \dots + a_nx^n. \quad (5.2)$$

The interpolation requirement is that $p_n(x_1) = y_1$, $p_n(x_2) = y_2$, \dots , $p_n(x_{n+1}) = y_{n+1}$. Using the above polynomial this produces the equations

$$\begin{aligned}
a_0 + a_1x_1 + \cdots + a_nx_1^n &= y_1 \\
a_0 + a_1x_2 + \cdots + a_nx_2^n &= y_2 \\
&\vdots \\
a_0 + a_1x_{n+1} + \cdots + a_nx_{n+1}^n &= y_{n+1}
\end{aligned}$$

This can be rewritten in matrix form as $\mathbf{V}\mathbf{a} = \mathbf{y}$, where $\mathbf{a} = (a_0, a_1, \dots, a_n)^T$, $\mathbf{y} = (y_1, y_2, \dots, y_{n+1})^T$, and

$$\mathbf{V} = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n+1} & x_{n+1}^2 & \cdots & x_{n+1}^n \end{pmatrix}. \quad (5.3)$$

This is a *Vandermonde matrix*. Given that it has a name you should not be surprised that it plays an important role in interpolation, but as will be seen shortly not all of its contributions are good.

Example

Each of the test data sets in Figure 5.3 contains 12 points. Fitting $p_{11}(x)$ to each set produces the curves shown in Figure 5.4. The top two look to be reasonable fits to the data while the bottom two are not. The over- and under-shoots seen in the bottom two curves often appear with higher degree polynomials and one of the drawbacks of using a global polynomial with larger data sets. ■

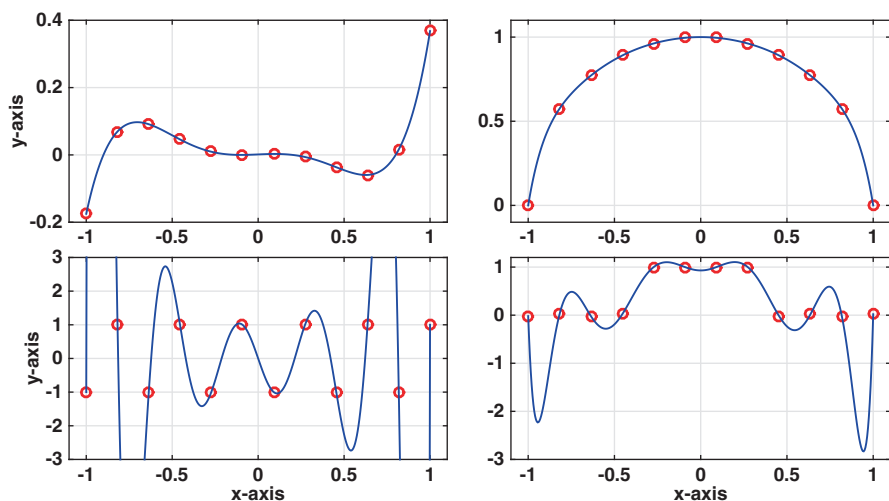


Figure 5.4 Using a global polynomial $p_{11}(x)$, as given in (5.2), for the data in Figure 5.3.

The previous example indicates that there are concerns about using a global polynomial, particularly when you have a large number of data points. In fact, there are significant problems not evident in the example related to the condition number of the Vandermonde matrix. As shown in Section 3.4, \mathbf{V} can be ill-conditioned for even small values of n . It is possible in some cases to rescale the data to improve the condition number, and an example of this is given in Exercise 5.28. However, it is possible to avoid this particular problem altogether, and this will be considered next. Even so, be warned that there is a second problem with a large number of equally spaced data points and this is discussed in Section 5.2.3.

5.2.2 Lagrange Approach

The easiest way to explain how to avoid using the Vandermonde matrix is to examine what happens with linear and quadratic functions. So, suppose the data points are (x_1, y_1) and (x_2, y_2) , where $x_1 \neq x_2$. The global polynomial in this case is linear and it can be written as

$$\begin{aligned} p_1(x) &= y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \\ &= y_1 \frac{x - x_2}{x_1 - x_2} + y_2 \frac{x - x_1}{x_2 - x_1} \\ &= y_1 \ell_1(x) + y_2 \ell_2(x), \end{aligned}$$

where

$$\ell_1(x) = \frac{x - x_2}{x_1 - x_2},$$

and

$$\ell_2(x) = \frac{x - x_1}{x_2 - x_1}.$$

The functions $\ell_1(x)$ and $\ell_2(x)$ are examples of *linear Lagrange interpolation functions* and they have the properties that $\ell_1(x_1) = \ell_2(x_2) = 1$, $\ell_1(x_2) = 0$, and $\ell_2(x_1) = 0$. In other words, each $\ell_i(x)$ is linear, equal to one when $x = x_i$ and equal to zero at the other x_j data point.

It is relatively easy to generalize this idea and write down the quadratic Lagrange interpolation functions. Namely, if the data points are (x_1, y_1) , (x_2, y_2) , and (x_3, y_3) then

$$\begin{aligned} \ell_1(x) &= \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)}, \\ \ell_2(x) &= \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}, \\ \ell_3(x) &= \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}. \end{aligned}$$

By design, each $\ell_i(x)$ is quadratic, equal to one when $x = x_i$ and equal to zero at the other x_j data points. Using these functions the corresponding quadratic interpolation polynomial is

$$p_2(x) = y_1\ell_1(x) + y_2\ell_2(x) + y_3\ell_3(x). \quad (5.4)$$

For this to be well-defined it is required that the x_i 's are distinct, which means that $x_1 \neq x_2$, $x_1 \neq x_3$, and $x_2 \neq x_3$. Also, although it looks different, (5.4) produces the same function as given in (5.2) in the case of when $n = 2$.

Generalizing the above results we have that given data points (x_1, y_1) , $(x_2, y_2), \dots, (x_{n+1}, y_{n+1})$, with the x_i 's distinct, the interpolation polynomial can be written as

$$p_n(x) = \sum_{i=1}^{n+1} y_i \ell_i(x), \quad (5.5)$$

where the *Lagrange interpolation functions* are defined as

$$\ell_i(x) = \frac{(x - x_1)(x - x_2) \cdots (x - x_{i-1})(x - x_{i+1}) \cdots (x - x_{n+1})}{(x_i - x_1)(x_i - x_2) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_{n+1})} \quad (5.6)$$

$$= \prod_{\substack{j=1 \\ j \neq i}}^{n+1} \frac{x - x_j}{x_i - x_j}. \quad (5.7)$$

In a similar manner, as in the linear and quadratic examples, each $\ell_i(x)$ is an n th degree polynomial, it is equal to one when $x = x_i$, and it equals zero when $x = x_j$ for $j \neq i$.

Example

To find the global polynomial that interpolates the data in Table 5.1, first note that the data is $(x_1, y_1) = (0, 1)$, $(x_2, y_2) = (1/2, -1)$, and $(x_3, y_3) = (1, 2)$. Consequently, the polynomial is

$$\begin{aligned} p_2(x) &= y_1\ell_1(x) + y_2\ell_2(x) + y_3\ell_3(x) \\ &= \ell_1(x) - \ell_2(x) + 2\ell_3(x), \end{aligned}$$

where $\ell_1(x) = 2(x - 1/2)(x - 1)$, $\ell_2(x) = -4x(x - 1)$, and $\ell_3(x) = 2x(x - 1/2)$. ■

x	0	1/2	1
y	1	-1	2

Table 5.1 Data for example.

The Lagrange interpolation formulas in (5.5) and (5.7) have an advantage over the direct formula, given in (5.2), in that the Vandermonde matrix is avoided. There is still a potential computational problem related to overflow, particularly for (5.6). This is the same problem explored in Exercise 1.7. It can help to rescale the data, and this is explained in Exercise 5.28. However, writing the formula in factored form as in (5.7) significantly reduces the possibility of overflow.

The price paid for avoiding the Vandermonde matrix is the effort needed to evaluate the ℓ_i 's, and this is often stated to be a drawback of the method. For a large number of data points, evaluating $\ell_i(x)$ can require about $2n^2$ flops. To translate this into computing time, if you use 20 data points and 2,000 evaluation points, the computing time is about 1 msec. Similarly, if you use 200 data points and 20,000 evaluation points, the computing time is about 1 sec. In other words, the computational time is not particularly significant unless you are working with a large data set. In such cases there are more efficient ways to write the interpolation formulas, using something called barycentric weights, and these are explored in Exercise 5.29. However, there is a more significant problem with this method and this is explained next.

5.2.3 Runge's Function

Now that the ill-conditioned matrix problem has been avoided it is time to explain the other problem with using a global interpolation polynomial. For this we can use the top two plots in Figure 5.4. It is seen that with the 10 data points we have obtained a fairly accurate approximation of the original functions. Always trying to improve things, one might think that by adding data points that the approximation will be even better. For many functions this does indeed happen but there are functions where it does not (and you would think it should). The example many use to demonstrate this is

$$f(x) = \frac{1}{1 + 25x^2}, \quad (5.8)$$

and this is known as *Runge's function*. This is plotted in Figure 5.5, along with $p_4(x)$ and $p_{12}(x)$. The interpolation polynomials are constructed using equally spaced data points. It is seen that in the center of the interval the approximation improves but it gets worse towards the endpoints. Increasing the number of data points makes the situation worse in the sense that the magnitude of $|p_n(x)|$ near the endpoints increases. For example, when $n = 40$ the maximum in $|p_n(x)|$ is about 10^4 , while for $n = 100$ the maximum in $|p_n(x)|$ is about 10^{14} . Moreover, this behavior is not limited to equally spaced points. If you take the points randomly from the interval, the maximum in $|p_n(x)|$ is often even larger.

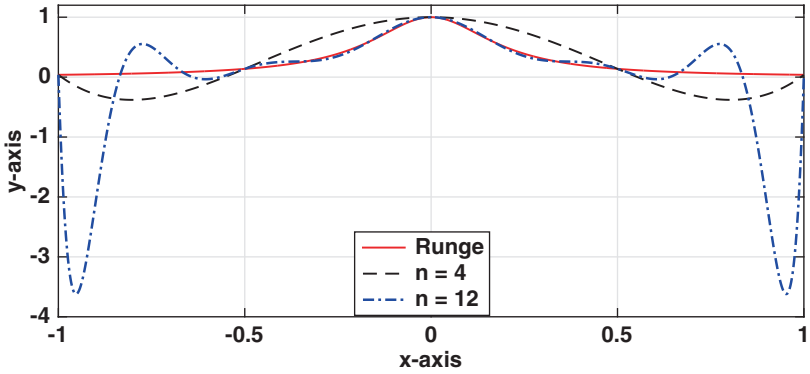


Figure 5.5 The Runge function (5.8) along with two interpolating polynomials.

The conclusion from the above discussion is that using a global interpolation polynomial works well for small data sets but has limited value as the number of data points increases. One solution for larger data sets is to break them into small groups, use interpolation on the subgroups, and then connect the information into a coherent whole. This is considered in the next section. If you are set on using a global polynomial then you need to consider where the x_i 's are placed in the interval, and this is considered in Section 5.5.4.

5.3 Piecewise Linear Interpolation

We will consider using linear interpolation between adjacent data points. This is effectively what is done in a child's connect the dots puzzle. An example is shown in Figure 5.6 where the line between (x_1, y_1) and (x_{16}, y_{16}) is pre-drawn in the puzzle.

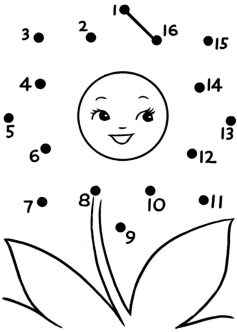


Figure 5.6 A typical child's puzzle of connect the dots.

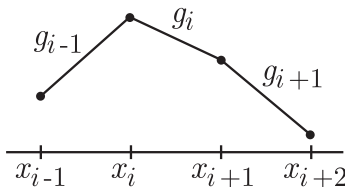


Figure 5.7 Intervals and functions used for piecewise linear interpolation.

To be able to write down the mathematical formula used for piecewise linear interpolation, assume that the data points are (x_1, y_1) , (x_2, y_2) , \dots , (x_{n+1}, y_{n+1}) , where $x_1 < x_2 < \dots < x_{n+1}$. The linear function connecting adjacent data points is given as

$$g_i(x) = y_i + \frac{y_{i+1} - y_i}{x_{i+1} - x_i}(x - x_i), \quad \text{for } x_i \leq x \leq x_{i+1}. \quad (5.9)$$

Assembling these into a complete description of the interpolation function we get

$$g(x) = \begin{cases} g_1(x) & \text{if } x_1 \leq x \leq x_2 \\ g_2(x) & \text{if } x_2 \leq x \leq x_3 \\ \vdots & \vdots \\ g_n(x) & \text{if } x_n \leq x \leq x_{n+1}. \end{cases} \quad (5.10)$$

An illustration of this is given in Figure 5.7.

It is possible to write $g(x)$ in a form that can be easier to use, and looks a lot simpler than the expression in (5.10). To do this we introduce the piecewise linear function $G_i(x)$, with $G_i(x_i) = 1$ and $G_i(x_j) = 0$ if $j \neq i$. The formula for this function is

$$G_i(x) = \begin{cases} 0 & \text{if } x \leq x_{i-1}, \\ \frac{x - x_{i-1}}{x_i - x_{i-1}} & \text{if } x_{i-1} \leq x \leq x_i, \\ \frac{x_i - x_{i+1}}{x - x_{i+1}} & \text{if } x_i \leq x \leq x_{i+1}, \\ 0 & \text{if } x_{i+1} \leq x. \end{cases} \quad (5.11)$$

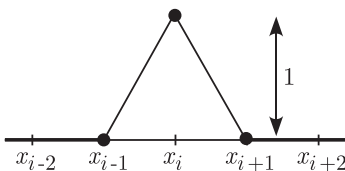


Figure 5.8 The piecewise linear function $G_i(x)$ defined in (5.11).

and a sketch of this is given in Figure 5.8. In the case of when the points are equally spaced, so $x_{i+1} - x_i = h$, this can be written in a more compact form as

$$G_i(x) = G\left(\frac{x - x_i}{h}\right), \quad (5.12)$$

where

$$G(x) = \begin{cases} 1 - |x| & \text{if } |x| \leq 1, \\ 0 & \text{if } 1 \leq |x|. \end{cases} \quad (5.13)$$

Note that the defining properties of $G_i(x)$ are very similar to the properties that were used to define the Lagrange interpolation functions $\ell_i(x)$ in Section 5.2.2.

With this, the piecewise linear interpolation function (5.10) can be written as

$$g(x) = \sum_{i=1}^{n+1} y_i G_i(x). \quad (5.14)$$

Just so it's clear, this expression produces the same interpolation function as the expanded version given in (5.10). Also, because of its shape, $G_i(x)$ has a variety of names, and they include the *hat function* and the *chapeau function*.

As a final comment, to define G_1 it is necessary to introduce x_0 , with $x_0 < x_1$, and for G_{n+1} we need to add in x_{n+2} , with $x_{n+1} < x_{n+2}$. Exactly where these two points are located is not important because they have no affect on the interpolation function over the interval $x_1 \leq x \leq x_{n+1}$.

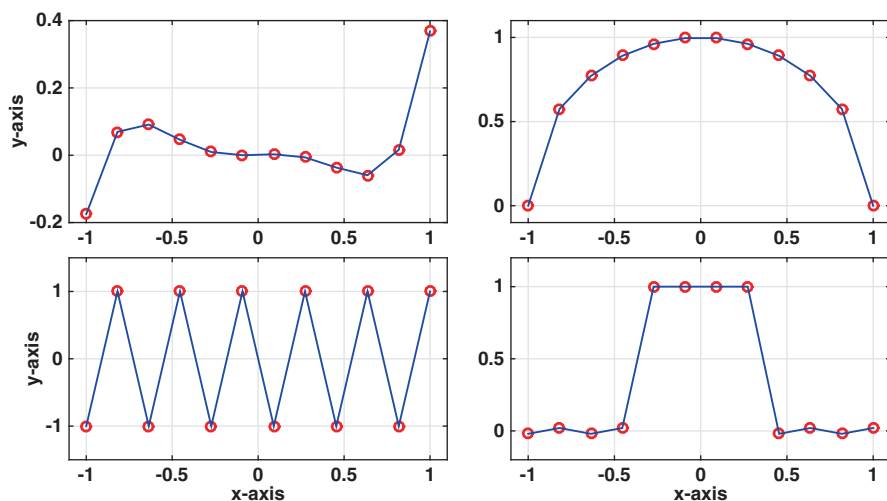


Figure 5.9 Using a piecewise linear function to fit the data in Figure 5.3.

Example

Using a piecewise linear function to fit the data in Figure 5.9 produces the curves shown in Figure 5.9. These curves are not bad fits but they are also not great. They are not bad because they do not contain the over- and under-shoots seen in Figure 5.9. However, they are not great because they are jagged. Note that in some applications, such as the one in Figure 5.6, jagged is what is desired but in many applications this is something one wants to avoid. ■

Example

Find the piecewise linear function that interpolates the data in Table 5.2. Note that in this case, $x_1 = 0$, $x_2 = 1/2$, and $x_3 = 1$.

Method 1: Using (5.10),

$$g(x) = \begin{cases} g_1(x) & \text{if } 0 \leq x \leq 1/2 \\ g_2(x) & \text{if } 1/2 \leq x \leq 1, \end{cases}$$

where

$$\begin{aligned} g_1(x) &= y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \\ &= 1 - 4x, \end{aligned}$$

and

$$\begin{aligned} g_2(x) &= y_2 + \frac{y_3 - y_2}{x_3 - x_2}(x - x_2) \\ &= -4 + 6x. \end{aligned}$$

Method 2: Using (5.14),

$$\begin{aligned} g(x) &= y_1 G_1(x) + y_2 G_2(x) + y_3 G_3(x) \\ &= G_1(x) - G_2(x) + 2G_3(x), \end{aligned}$$

x	0	1/2	1
y	1	-1	2

Table 5.2 Data for example.

where

$$G_1(x) = \begin{cases} 1 + 2x & \text{if } -1/2 \leq x \leq 0, \\ 1 - 2x & \text{if } 0 \leq x \leq 1/2, \\ 0 & \text{otherwise,} \end{cases}$$

$$G_2(x) = \begin{cases} 2x & \text{if } 0 \leq x \leq 1/2, \\ 2 - 2x & \text{if } 1/2 \leq x \leq 1, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$G_3(x) = \begin{cases} -1 + 2x & \text{if } 1/2 \leq x \leq 1, \\ 3 - 2x & \text{if } 1 \leq x \leq 3/2, \\ 0 & \text{otherwise.} \quad \blacksquare \end{cases}$$

5.4 Piecewise Cubic Interpolation

The principal criticism of piecewise linear interpolation is that the approximation function has corners. One method that is often used to avoid this is to replace the linear functions with cubics. Instead of (5.10), we have a interpolation function of the form (see Figure 5.10)

$$s(x) = \begin{cases} s_1(x) & \text{if } x_1 \leq x \leq x_2 \\ s_2(x) & \text{if } x_2 \leq x \leq x_3 \\ \vdots & \vdots \\ s_n(x) & \text{if } x_n \leq x \leq x_{n+1}, \end{cases} \quad (5.15)$$

where in the i th interval the function is

$$s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad \text{for } x_i \leq x \leq x_{i+1}. \quad (5.16)$$

To satisfy the interpolation conditions it is required that

$$s_i(x_i) = y_i, \quad s_i(x_{i+1}) = y_{i+1}, \quad \text{for } i = 1, 2, \dots, n. \quad (5.17)$$

This will determine two of the four constants in s_i . We will also require that the transition between intervals is as smooth as possible. First, the slopes must match and this means that

$$s'_i(x_{i+1}) = s'_{i+1}(x_{i+1}), \quad \text{for } i = 1, 2, \dots, n-1. \quad (5.18)$$

Second we will require that the second derivatives also match, and so

$$s''_i(x_{i+1}) = s''_{i+1}(x_{i+1}), \quad \text{for } i = 1, 2, \dots, n-1. \quad (5.19)$$

Conditions (5.17)–(5.19) are the basic requirements for $s(x)$ to be a cubic spline. However, $s(x)$ has $4n$ coefficients that we need to determine, and these conditions produce $4n - 2$ equations. In other words, we are short two conditions. What is usually done is to specify conditions at the left and right ends of the data interval. Some of the commonly made choices are as follows:

- *Natural Spline*: $s_1''(x_1) = 0$ and $s_n''(x_{n+1}) = 0$

This produces a spline with an interesting property related to curvature, and this will be explained later.

- *Clamped Spline*: $s_1'(x_1) = y_1'$ and $s_n'(x_{n+1}) = y_{n+1}'$

This requires knowing the value of the derivative at the endpoints, something that is not usually available.

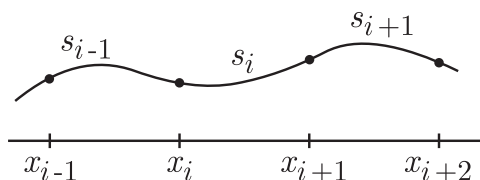


Figure 5.10 Intervals and functions used for piecewise cubic interpolation.

- *Not-a-Knot Spline*: $s_1'''(x_2) = s_2'''(x_2)$ and $s_{n-1}'''(x_n) = s_n'''(x_n)$

This is the default choice in MATLAB.

Whichever choice is made, the resulting function $s(x)$ provides a smooth interpolation of the given data points.

Example

To find the natural cubic spline that interpolates the data in Table 5.3, we use (5.15) and write

$$s(x) = \begin{cases} s_1(x) & \text{if } 0 \leq x \leq 1/2 \\ s_2(x) & \text{if } 1/2 \leq x \leq 1, \end{cases}$$

where

$$s_1(x) = a_1 + b_1x + c_1x^2 + d_1x^3,$$

and

$$s_2(x) = a_2 + b_2(x - 1/2) + c_2(x - 1/2)^2 + d_2(x - 1/2)^3.$$

x	0	1/2	1
y	1	-1	2

Table 5.3 Data for example.

From the interpolation conditions (5.17),

$$\begin{aligned}
 s_1(0) &= 1 : & a_1 &= 1, \\
 s_1(1/2) &= -1 : & a_1 + \frac{1}{2}b_1 + \frac{1}{4}c_1 + \frac{1}{8}d_1 &= -1, \\
 s_2(1/2) &= -1 : & a_2 &= -1, \\
 s_2(1) &= 2 : & a_2 + \frac{1}{2}b_2 + \frac{1}{4}c_2 + \frac{1}{8}d_2 &= 2.
 \end{aligned}$$

Also, from (5.18) and (5.19)

$$\begin{aligned}
 s'_1(1/2) &= s'_2(1/2) : & b_1 + c_1 + \frac{3}{4}d_1 &= b_2, \\
 s''_1(1/2) &= s''_2(1/2) : & 2c_1 + 3d_1 &= 2c_2.
 \end{aligned}$$

Finally, to qualify to be a natural cubic spline it is required that

$$\begin{aligned}
 s''_1(0) &= 0 : & c_1 &= 0, \\
 s''_2(1) &= 0 : & 2c_2 + 3d_2 &= 0.
 \end{aligned}$$

It is now a matter of solving the above equations, and after doing this one finds that

$$s_1(x) = 1 - \frac{13}{2}x + 10x^3,$$

and

$$s_2(x) = -1 + (x - 1/2) + 15(x - 1/2)^2 - 10(x - 1/2)^3. \quad \blacksquare$$

To find $s(x)$ it remains to solve $4n$ equations with $4n$ unknowns. It is possible to just solve the resulting matrix equation for the unknowns, but there are better ways to find the coefficients. One possibility is to mathematically simplify the equations, and reduce the problem to solving a system with n unknowns. Another approach is to use cubic B-splines. This will also reduce the problem down to having to solve for (approximately) n unknowns. The advantage of B-splines is that they are easier to code. They are also very useful for least squares fitting of data (see Exercise 8.32), as well when solving differential equations numerically [Holmes, 2007].

5.4.1 Cubic B-Splines

The idea is to write the interpolation function in the form

$$s(x) = \sum a_i B_i(x). \quad (5.20)$$

The functions $B_i(x)$ are called *cubic B-splines*, and a sketch of a typical B-spline is given in Figure 5.11. The above expression has a passing similarity to the expressions used for Lagrange interpolation and piecewise linear interpolation. However, one important difference is that the coefficient a_i in the above sum is not necessarily equal to the data value y_i .

The derivation of (5.20) consists of two steps. The first is the construction of the cubic B-splines $B_i(x)$. This only has to be done once, which means that these functions do not need to be rederived if the data set is changed. Once this is complete, then the problem used to find the a_i 's is determined. The values of the a_i 's do depend on the data, and so this problem must be solved each time the data set is changed.

Finding the B_i 's

Each $B_i(x)$ is a piecewise cubic function and has the form

$$B_i(x) = \begin{cases} 0 & \text{if } x \leq x_{i-2}, \\ q_{i-2}(x) & \text{if } x_{i-2} \leq x \leq x_{i-1}, \\ q_{i-1}(x) & \text{if } x_{i-1} \leq x \leq x_i, \\ q_{i+1}(x) & \text{if } x_i \leq x \leq x_{i+1}, \\ q_{i+2}(x) & \text{if } x_{i+1} \leq x \leq x_{i+2}, \\ 0 & \text{if } x_{i+2} \leq x, \end{cases} \quad (5.21)$$

where $q_j(x) = \bar{A}_j + \bar{B}_j(x - x_j) + \bar{C}_j(x - x_j)^2 + \bar{D}_j(x - x_j)^3$. We will assume that the x_i 's are equally spaced, with $h = x_{i+1} - x_i$, so the function $B_i(x)$ is symmetric about $x = x_i$. Note that one consequence of the symmetry is that $B'_i(x_i) = 0$.

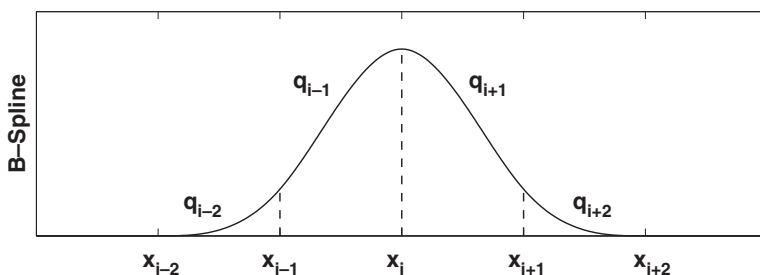


Figure 5.11 Sketch of the various components of a cubic B-spline.

The coefficients of the q_j 's in (5.21) are determined from the requirement that $B_i \in C^2(-\infty, \infty)$. This is accomplished by requiring the following:

$$\begin{aligned} x = x_{i-2}: \quad & q_{i-2}(x_{i-2}) = 0, \quad q'_{i-2}(x_{i-2}) = 0, \quad \text{and} \quad q''_{i-2}(x_{i-2}) = 0 \\ x = x_{i-1}: \quad & q_{i-2}(x_{i-1}) = q_{i-1}(x_{i-1}), \quad q'_{i-2}(x_{i-1}) = q'_{i-1}(x_{i-1}), \quad \text{and} \\ & q''_{i-2}(x_{i-1}) = q''_{i-1}(x_{i-1}) \\ x = x_i: \quad & q_{i-1}(x_i) = q_{i+1}(x_i), \quad q'_{i-1}(x_i) = q'_{i+1}(x_i), \quad \text{and} \\ & q''_{i-1}(x_i) = q''_{i+1}(x_i) \end{aligned}$$

with similar conditions at $x = x_{i+1}$ and $x = x_{i+2}$. From the conditions at $x = x_{i-2}$ one easily concludes that $q_{i-2}(x) = \bar{D}_{i-2}(x - x_{i-2})^3$. In a similar way, it is found that $q_{i+2}(x) = \bar{D}_{i+2}(x - x_{i+2})^3$, where from the symmetry, $\bar{D}_{i+2} = -\bar{D}_{i-2}$. From the smoothness requirements at x_{i-1} , and the requirement that $B'(x_i) = 0$, one finds that

$$q_{i-1}(x) = D_{i-2}[h^3 + 3h^2(x - x_{i-1}) + 3h(x - x_{i-1})^2 - 3(x - x_{i-1})^3].$$

A similar equation can be derived for $q_{i+1}(x)$. This leaves one undetermined constant and the convention is to take $B_i(x_i) = 2/3$, which means that $\bar{D}_{i-2} = 1/(6h^3)$. With this, $B_i(x)$ is completely defined and the functions in (5.21) are

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

By factoring the above polynomials it is possible to show that

$$B_i(x) = B\left(\frac{x - x_i}{h}\right), \quad (5.22)$$

where

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

A plot of the resulting function is shown in Figure 5.12. At x_{i-1} and x_{i+1} the curve makes such a smooth transition across the respective data point that you would not know that the cubics change there. The same is true at x_{i-2} and x_{i+2} where the function makes a smooth transition to zero.

Finding the a_i 's

The cubic spline interpolation function can now be written as

$$s(x) = \sum_{i=0}^{n+2} a_i B_i(x). \quad (5.23)$$

Just so it is clear, this function satisfies the smoothness conditions in (5.18) and (5.19), but does not yet satisfy the interpolation conditions in (5.17) or the specific end conditions (clamped, natural, etc.). Also, the sum is over all possible B_i 's that are nonzero on the interval $x_1 \leq x \leq x_{n+1}$. This has required us to include the $i = 0$ and $i = n + 2$ terms even though there is no x_{-1} , x_0 , x_{n+2} , or x_{n+3} in the original data set. We will deal with this shortly. First note that at x_i only B_{i-1} , B_i , and B_{i+1} are nonzero. In particular, using the values given in Table 5.4,

$$\begin{aligned} s(x_i) &= a_{i-1}B_{i-1}(x_i) + a_iB_i(x_i) + a_{i+1}B_{i+1}(x_i) \\ &= \frac{1}{6}(a_{i-1} + 4a_i + a_{i+1}). \end{aligned}$$

Because of the interpolation requirement (5.17) we have that

$$a_{i-1} + 4a_i + a_{i+1} = 6y_i, \quad \text{for } i = 1, 2, \dots, n+1. \quad (5.24)$$

To use this we need to know a_0 and a_{n+2} , and this is where the two additional conditions are used. We will use a natural spline, and for this note that

$$\begin{aligned} s''(x_i) &= a_{i-1}B''_{i-1}(x_i) + a_iB''_i(x_i) + a_{i+1}B''_{i+1}(x_i) \\ &= \frac{1}{h^2}(a_{i-1} - 2a_i + a_{i+1}). \end{aligned}$$

Solving $s''(x_1) = 0$ we get that $a_0 = 2a_1 - a_2$, and at the other end one finds that $a_{n+2} = 2a_{n+1} - a_n$. In (5.24), when $i = 1$ one finds that $a_1 = y_1$ and at the other end one gets that $a_{n+1} = y_{n+1}$. The rem-

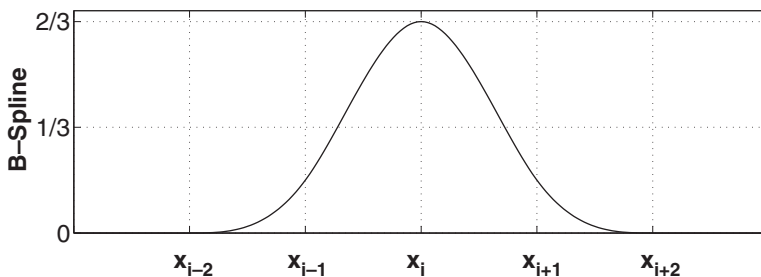


Figure 5.12 Plot of the cubic B-spline $B_i(x)$ defined in (5.22).

	x_{i-1}	x_i	x_{i+1}	x_j for $j \neq i, i \pm 1$
B_i	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$	0
B'_i	$\frac{1}{2h}$	0	$-\frac{1}{2h}$	0
B''_i	$\frac{1}{h^2}$	$-\frac{2}{h^2}$	$\frac{1}{h^2}$	0

Table 5.4 Values of the B-spline $B_i(x)$, as defined in (5.21), at the grid points used in its construction.

aining a_i 's are found by solving $\mathbf{A}\mathbf{a} = \mathbf{z}$ where $\mathbf{a} = (a_2, a_3, \dots, a_n)^T$, $\mathbf{z} = (6y_2 - y_1, 6y_3, \dots, 6y_{n-1}, 6y_n - y_{n+1})^T$, and \mathbf{A} is the $(n-1) \times (n-1)$ tridiagonal matrix

$$\mathbf{A} = \begin{pmatrix} 4 & 1 & & & 0 \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & \ddots & \ddots & \ddots \\ 0 & & & & 1 \\ & & & 1 & 4 \end{pmatrix}.$$

This positive definite and tridiagonal matrix equation can be solved using the Thomas algorithm (see Section 3.8). This procedure is very fast, and requires minimal storage, which means finding the coefficients for a cubic spline is fairly easy even for a large number of interpolation points.

As a final comment, for a clamped spline it is also necessary to solve an equation of the form $\mathbf{A}\mathbf{a} = \mathbf{z}$, where \mathbf{A} and \mathbf{z} are given in Exercise 5.30.

Example

Using a natural cubic spline to fit the data in Figure 5.3 produces the solid (blue) curves shown in Figure 5.13. For comparison, the curves obtained using a not-a-knot spline are also shown. The interpolation of the top two data sets is as good as what was obtained using the global polynomial, and there are few minor differences between the two spline functions. Moreover, both splines give a better representation than a global polynomial for the lower two data sets. For the lower right data set some small over- and under-shoots are present in the spline functions, but they are not as pronounced as those in Figure 5.4. It is also evident that there are also differences between the two spline functions, although these occur primarily in the regions close to the endpoints. ■

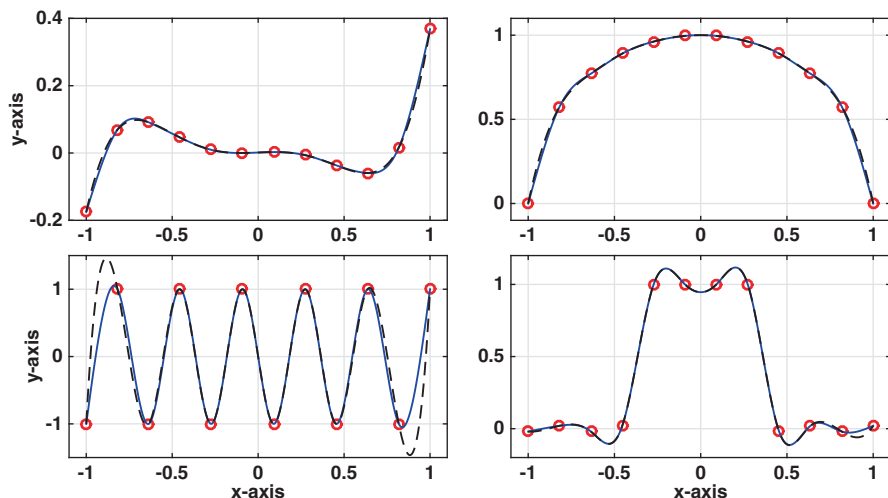


Figure 5.13 Using a natural cubic spline function, the solid (blue) curves, and a Not-a-Knot spline, the dashed (black) curves, to fit the data in Figure 5.3.

Example

To find the natural cubic spline that interpolates the data in Table 5.1, we use (5.23) and write

$$s(x) = a_0 B_0(x) + a_1 B_1(x) + a_2 B_2(x) + a_3 B_3(x) + a_4 B_4(x),$$

where $x_0 = -1/2$, $x_4 = 3/2$, and $B_i(x) = B(2(x - x_i))$. The interpolation requirements are

$$\begin{aligned} s(0) = 1 &: & a_0 + 4a_1 + a_2 &= 6, \\ s(1/2) = -1 &: & a_1 + 4a_2 + a_3 &= -6, \\ s(1) = 2 &: & a_2 + 4a_3 + a_4 &= 12, \end{aligned}$$

and the natural spline end conditions are

$$\begin{aligned} s''(0) = 0 &: & a_0 - 2a_1 + a_2 &= 0, \\ s''(1) = 0 &: & a_2 - 2a_3 + a_4 &= 0. \end{aligned}$$

Solving these equations it is found that

$$s(x) = \frac{17}{4}B_0(x) + B_1(x) - \frac{9}{4}B_2(x) + 2B_3(x) + \frac{15}{4}B_4(x). \quad \blacksquare$$

The question arises as to why the natural cubic spline works so well. There is a partial answer to this and it involves curvature. Recall that for a curve $y = f(x)$, the curvature at a point is defined as

$$\kappa = \frac{|f''(x)|}{[1 + (f')^2]^{3/2}}.$$

Assuming the curve is not particularly steep, one can approximate the curvature as $\kappa \approx |f''(x)|$. This is brought up because of the next result due to Holladay [1957].

Theorem 5.1. *If $q \in C^2[a, b]$ interpolates the same data points that the natural cubic spline function (5.15) interpolates, then*

$$\int_a^b [s''(x)]^2 dx \leq \int_a^b [q''(x)]^2 dx.$$

In these integrals, $a = x_1$ and $b = x_{n+1}$.

What this theorem states is that out of all smooth functions that interpolate the data, the *natural* cubic spline produces the interpolation function with the smallest total curvature (squared). This helps explain why the spline interpolations in Figure 5.13 do not suffer the significant under- and overshoots seen in Figure 5.4.

5.5 Function Interpolation

In using the data sets to test out the various interpolation methods we have been using a qualitative, or visual, determination of how well they do. We are now going to make the test more quantitative and this will restrict the applications. In particular, it is assumed that the data comes from the evaluation of a given function $f(x)$ and we are going to investigate how well the interpolation function approximates $f(x)$ between the data points.

In what follows the data points are $(x_1, y_1), (x_2, y_2), \dots, (x_{n+1}, y_{n+1})$, where $y_i = f(x_i)$. Also, unless stated explicitly to the contrary, the step size $h = x_{i+1} - x_i$ is assumed constant (so the x_i 's are equally spaced), with $a = x_1$ and $b = x_{n+1}$. What is of interest is whether the approximation gets better as the step size h gets smaller. In particular, does the error go to zero as h goes to zero?

5.5.1 Global Polynomial Interpolation

We begin with a global interpolation polynomial $p_n(x)$. As explained in Section 5.2.3, $p_n(x)$ can fail to provide a good approximation of a function if a large number of equally spaced points are used. However, it is effective for a

small number of points, as long as they are not too far apart. In fact, such approximations are central to several of the methods considered later in the text. The critical result needed to determine the error when using $p_n(x)$ is given in the following theorem:

Theorem 5.2. *If $f \in C^{n+1}[a, b]$, then*

$$f(x) = p_n(x) + \frac{f^{(n+1)}(\eta)}{(n+1)!} q_{n+1}(x), \quad \text{for } a \leq x \leq b, \quad (5.25)$$

where

$$q_{n+1}(x) = (x - x_1)(x - x_2) \cdots (x - x_{n+1}), \quad (5.26)$$

and η is a point in (a, b) .

Outline of Proof: To explain how this is proved, we will consider the case of when $n = 1$. In this case, $q_2(x) = (x - a)(x - b)$. The formula in (5.25) holds when $x = a$ or $x = b$, so assume that $a < x < b$. The key step is a trick, which consists of introducing the function

$$F(z) = f(z) - p_1(z) + \frac{f(x) - p_1(x)}{q_2(x)} q_2(z).$$

Given the way it is defined, $F(a) = 0$, $F(x) = 0$, and $F(b) = 0$. According to Rolle's theorem, there must be a point z_1 , where $a < z_1 < x$ and $F'(z_1) = 0$, and there must be a point z_2 , where $x < z_2 < b$ and $F'(z_2) = 0$. Using Rolle's theorem again, there must be a point η , where $z_1 < \eta < z_2$ and $F''(\eta) = 0$. From the above formula for $F(z)$, one finds that $F''(\eta) = 0$ reduces to (5.25). The case of when $n > 1$ is similar, except one uses Rolle's theorem $n + 1$ times (instead of twice). \square

An immediate consequence of this theorem is the following:

Theorem 5.3. *If $f \in C^{n+1}[a, b]$, then the global interpolation polynomial $p_n(x)$ satisfies*

$$|f(x) - p_n(x)| \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty \|q_{n+1}\|_\infty, \quad \text{for } a \leq x \leq b,$$

where

$$\|f^{(n+1)}\|_\infty = \max_{a \leq x \leq b} |f^{(n+1)}(x)|,$$

and

$$\|q_{n+1}\|_\infty = \max_{a \leq x \leq b} |q_{n+1}(x)|.$$

It should be pointed out that the above two theorems hold in the case of when the x_i 's are not equally spaced (this fact is used in Section 5.5.4).

To make use of Theorem 5.3, we need to determine $\|q_{n+1}\|_\infty$. This is not hard to do for small values of n , and to illustrate this, consider the case of when $n = 2$. For this, $q_2(x) = (x - x_1)(x - x_2)$, where $x_2 = x_1 + h$. The maximum, and minimum, of this function occur either at the endpoints or at a critical point inside the interval. First note that $q_2(x_1) = q_2(x_2) = 0$. As for the critical points, solving $q'_2(x) = 0$, one finds that $x = x_1 + \frac{1}{2}h$. From this it follows that $\|q_2\|_\infty = \frac{1}{4}h^2$. Similarly, if $n = 3$, then $q_3(x) = (x - x_1)(x - x_2)(x - x_3)$, where $x_2 = x_1 + h$ and $x_3 = x_1 + 2h$. Solving $q'_3(x) = 0$, one finds two solutions, $x = x_1 + h \pm \frac{1}{3}h\sqrt{3}$. From this it follows that $\|q_3\|_\infty = \frac{2}{9}h^3\sqrt{3}$. Continuing this, the values in Table 5.5 are obtained.

n	$\ q_{n+1}\ _\infty$
1	$\frac{1}{4}h^2$
2	$\frac{2}{9}\sqrt{3}h^3$
3	h^4
4	$\frac{1}{50}(\sqrt{145} - 1)\sqrt{150 + 10\sqrt{145}}h^5$
5	$\frac{16}{27}(7\sqrt{7} + 10)h^6$

Table 5.5 Value of $\|q_{n+1}\|_\infty$, which appears in the error formula in Theorem 5.3.

Example

Suppose that $f(x) = \cos(2\pi x)$ and we use the interpolation polynomial $p_2(x)$, with points x_1 , $x_2 = x_1 + h$, and $x_3 = x_1 + 2h$. According to Theorem 5.3, how small does h need to be to guarantee an error of 10^{-6} , irrespective of the choice for x_1 ? To answer this, since $f''' = -(2\pi)^3 \sin(2\pi x)$, then $\|f'''\|_\infty \leq (2\pi)^3$. With this we have that

$$\frac{1}{6}\|f'''\|_\infty\|q_3\|_\infty \leq \sqrt{3}\left(\frac{2\pi}{3}h\right)^3.$$

Consequently we will achieve the require error bound if $\sqrt{3}(2\pi h/3)^3 \leq 10^{-6}$, which means $h \leq 3^{5/6}/(200\pi) \approx 0.0040$. ■

Finding $\|q_{n+1}\|_\infty$ for larger values of n is difficult, and the usual approach is to find an upper bound on this number. One that is not hard to derive is (see Exercise 5.31)

$$\|q_{n+1}\|_\infty \leq \frac{1}{4}n!h^{n+1}. \quad (5.27)$$

Using this, the inequality in Theorem 5.3 can be written as

$$|f(x) - p_n(x)| \leq \frac{1}{4(n+1)} \|f^{(n+1)}\|_\infty h^{n+1}, \quad \text{for } a \leq x \leq b, \quad (5.28)$$

If this is used in the above example, the conclusion is that it is necessary to have $h \leq (3/2)^{1/3}/(100\pi) \approx 0.0036$. The fact that this is smaller than the requirement given in the example is a reflection of the inequality in (5.27).

5.5.2 Piecewise Linear Interpolation

The next easiest method to analyze is piecewise linear interpolation. An example is shown in Figure 5.14, where $f(x) = \cos(2\pi x)$ is approximated using 6 points over the interval $0 \leq x \leq 1$. How well the linear functions approximate $f(x)$ depends on the subinterval. This is why in the analysis for the general case in the next paragraph we first determine what happens over each subinterval $x_i \leq x \leq x_{i+1}$.

To determine how well $f(x)$ is approximated by $g_i(x)$, given in (5.9), over the subinterval $x_i \leq x \leq x_{i+1}$ we can use Theorem 5.3. In this case, $n = 1$ and $q_2(x) = (x - x_i)(x - x_{i+1})$. Consequently,

$$\begin{aligned} |f(x) - g_i(x)| &\leq \frac{1}{2} \max_{x_i \leq x \leq x_{i+1}} |f''(x)| \max_{x_i \leq x \leq x_{i+1}} |q_2(x)| \\ &= \frac{1}{8} h^2 \max_{x_i \leq x \leq x_{i+1}} |f''(x)| \\ &\leq \frac{1}{8} h^2 \|f''\|_\infty. \end{aligned}$$

This applies to each subinterval, which gives the next result.

Theorem 5.4. *If $f \in C^2[a, b]$ then the piecewise linear interpolation function (5.9) satisfies*

$$|f(x) - g(x)| \leq \frac{1}{8} h^2 \|f''\|_\infty, \quad \text{for } a \leq x \leq b,$$

where $\|f''\|_\infty = \max_{a \leq x \leq b} |f''(x)|$.

This means that the piecewise linear interpolation function converges to the original function and the error is second order (because of the h^2). Consequently, if the number of interpolation points is doubled, the error in the approximation should decrease by about a factor of $1/4$.

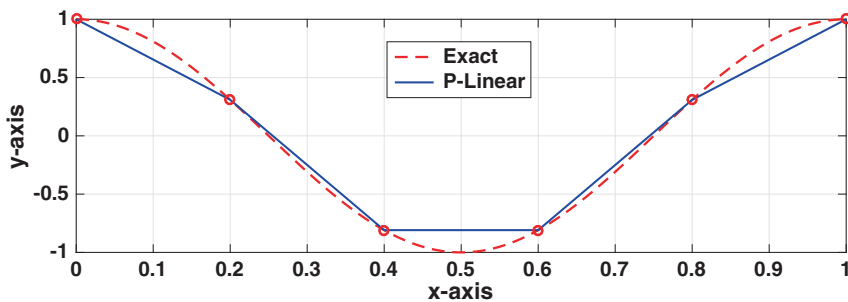


Figure 5.14 Piecewise linear interpolation of $f(x) = \cos(2\pi x)$.

Examples

1. Piecewise linear interpolation is used to approximate $f(x) = \cos(2\pi x)$ in Figure 5.14. According to Theorem 5.4, what is the error bound for this approximation?

The interval in this case is $0 \leq x \leq 1$. Since $f(x) = \cos(2\pi x)$, then $f''(x) = -4\pi^2 \cos(2\pi x)$ and from this it follows that $\|f''\|_\infty = 4\pi^2$. Given that $h = 1/5$, the error bound is $|f(x) - g(x)| \leq \pi^2/50$, where $\pi^2/50 \approx 0.2$. ■

2. For $f(x) = \cos(2\pi x)$, where $0 \leq x \leq 1$, how many interpolation points are needed to guarantee an error of 10^{-4} ?

Since $\|f''\|_\infty = 4\pi^2$, then we want $\pi^2 h^2/2 \leq 10^{-4}$. This gives

$$h \leq \sqrt{2} \times 10^{-2}/\pi.$$

If n is the number of interpolation points, then $h = 1/(n-1)$, and combining our results, $n \geq 1 + \sqrt{2}\pi^2 \times 10^2 \approx 1396.8$. Therefore, we need to take $n \geq 1397$. ■

3. According to Theorem 5.4, for what functions will there be zero error using piecewise linear interpolation, no matter what the interpolation interval?

This requires $\|f''\|_\infty = 0$, which means that $f''(x) = 0$ for $a \leq x \leq b$. Therefore, to have zero error it must be that $f(x) = \alpha + \beta x$, i.e., it must be a linear function in this interval. ■

5.5.3 Cubic Splines

This brings us to the question of how well the cubic splines do when interpolating a function. The answer depends on what type of spline function is used (natural, clamped, or not-a-knot). To illustrate, in Figure 5.15 the function $f(x) = \cos(2\pi x)$ is approximated using both a natural and a clamped cubic spline. The clamped spline provides a somewhat better approximation but this is not surprising because it has the advantage of using the derivative information at the endpoints. It is possible to determine the error for the clamped spline, but because this requires some effort to derive only the result will be stated (see Hall and Meyer 1976 for the proof).

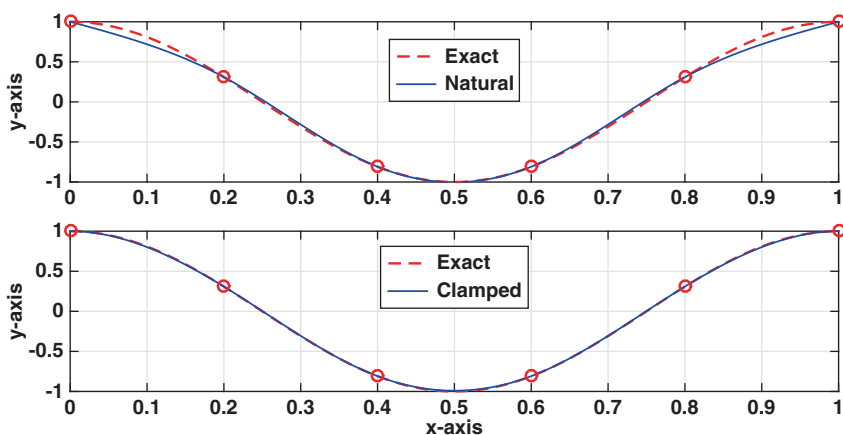


Figure 5.15 Natural and clamped cubic spline interpolation of $f(x) = \cos(2\pi x)$.

Theorem 5.5. *If $f \in C^4[a, b]$ then the clamped cubic spline interpolation function (5.15) satisfies*

$$|f(x) - s(x)| \leq \frac{5}{384} h^4 \|f''''\|_\infty, \quad \text{for } a \leq x \leq b,$$

where $\|f''''\|_\infty = \max_{a \leq x \leq b} |f''''(x)|$. Moreover, for $a \leq x \leq b$,

$$\begin{aligned} |f'(x) - s'(x)| &\leq \frac{1}{24} h^3 \|f''''\|_\infty, \\ |f''(x) - s''(x)| &\leq \frac{1}{3} h^2 \|f''''\|_\infty. \end{aligned}$$

This is an amazing result because it states that the clamped cubic spline can be used to approximate $f(x)$ and its first two derivatives. Moreover,

the error in approximating $f(x)$ is fourth-order (because of the h^4). As an example, if the number of interpolation points is increased by a factor of 10, the error bound decreases by about a factor of 10^4 . In comparison, according to Theorem 5.4, the error for piecewise linear interpolation decreases by a factor of 10^2 .

Examples

1. A clamped cubic spline is used to approximate $f(x) = \cos(2\pi x)$ in Figure 5.15. According to Theorem 5.5, what is the error bound for this approximation?

The interval is $0 \leq x \leq 1$. Since $f(x) = \cos(2\pi x)$, then $f''''(x) = (2\pi)^4 \cos(2\pi x)$ and $\|f''''\|_\infty = (2\pi)^4$. Given that $h = 1/5$, the error bound is $|f(x) - g(x)| \leq \pi^4/3000$, where $\pi^4/3000 \approx 0.03$. ■

2. For $f(x) = \cos(2\pi x)$, where $0 \leq x \leq 1$, how many interpolation points are needed to guarantee an error of 10^{-4} when using a clamped spline?

Since $\|f''''\|_\infty = (2\pi)^4$, then we want

$$\frac{5}{384}h^4(2\pi)^4 \leq 10^{-4}.$$

This can be rewritten as $h \leq (384/5)^{1/4}/(20\pi)$. If n is the number of interpolation points, then $h = 1/(n-1)$, and combining our results, $n \geq 1 + 20\pi(5/384)^{1/4} \approx 22.2$. Therefore, we need to take $n \geq 23$. ■

3. According to Theorem 5.5, for what functions will there be zero error using a clamped spline, no matter what the interpolation interval?

This requires $\|f''''\|_\infty = 0$, which means that $f''''(x) = 0$ for $a \leq x \leq b$. Therefore, to have zero error it must be that $f(x)$ is a cubic function. ■

The major drawback in the above theorem is that it requires the clamped end conditions and for many problems $f'(x_1)$ and $f'(x_{n+1})$ are not known. It is possible to obtain the same level of accuracy with a natural cubic spline, except near the endpoints, as long as enough points are used. This is seen in Figure 5.15, where the natural spline does almost as well approximating the function except over the subintervals next to the boundary points. The exact statement of the result is given next (see [Kershaw, 1971] for the proof).

Theorem 5.6. *If $f \in C^4[a, b]$ then for the natural cubic spline interpolation function $s(x)$:*

1. $|f(x) - s(x)| \leq K_1 h^2$, for $a \leq x \leq b$, where K_1 is a positive constant.
2. For large n , there are points x_ℓ and x_r , with $a \leq x_\ell < x_r \leq b$, so that

$$|f(x) - s(x)| \leq K_2 h^4, \quad \text{for } x_\ell \leq x \leq x_r,$$

where K_2 is a positive constant. Moreover, $x_\ell \rightarrow a$ and $x_r \rightarrow b$ as $n \rightarrow \infty$.

What this theorem states is that the error for a natural cubic spline is at least second-order. It also states that the error is actually fourth-order except near the endpoints. Moreover, the regions near the endpoints where it is not fourth-order shrink as n increases.

5.5.4 Chebyshev Interpolation

We saw earlier that the polynomial that interpolates all of the data points can be written as

$$p_n(x) = \sum_{i=1}^{n+1} y_i \ell_i(x), \quad (5.29)$$

where

$$\ell_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{n+1} \frac{x - x_j}{x_i - x_j}. \quad (5.30)$$

Given that we are now considering function interpolation, it is assumed that $y_i = f(x_i)$. One thing to note is that this does not require that the x_i 's are equally spaced. Also, we know that when n is large, and the points are equally spaced, the above polynomial should not be used for interpolation. The question we examine now is, is it possible to pick the locations of the x_i 's so $p_n(x)$ is capable of producing an accurate interpolation function. It is possible, and to explain how, we need Theorem 5.2. What is of interest here is the error term, which is

$$\frac{f^{(n+1)}(\eta)}{(n+1)!} q_{n+1}(x),$$

where $q_{n+1}(x)$ is given in (5.26). In Figure 5.5 we saw that the values of this can be huge. We want to prevent this from happening. For a given data set, so n is given, the assumption that $f \in C^{n+1}[a, b]$ means that there is a positive constant M_{n+1} so that $|f^{(n+1)}(x)| \leq M_{n+1}$. So, the part of the error function that we need to concentrate on is $q_{n+1}(x)$, and what we are specifically interested in is the value of

$$Q = \max_{a \leq x \leq b} |q_{n+1}(x)|. \quad (5.31)$$

This brings us to the following question: given n , how do we position the x_i 's in the interval $a \leq x \leq b$ to minimize Q .

The answer is easy to state, but deriving it requires some work. The result is given below, and afterwards the derivation is outlined.

Theorem 5.7. *The x_i 's that produce the smallest value of Q are*

$$x_i = \frac{1}{2}[a + b + (b - a)z_i], \text{ for } i = 1, 2, \dots, n + 1, \quad (5.32)$$

where

$$z_i = \cos\left(\frac{2i - 1}{2(n + 1)}\pi\right). \quad (5.33)$$

The x_i 's in the above theorem are called the *Chebyshev points* because they correspond to the zeros of the $(n + 1)$ th Chebyshev polynomial. Note that the Chebyshev polynomials are usually defined for $-1 \leq x \leq 1$, and when using $a \leq x \leq b$ they are referred to as the Chebyshev polynomials for the general interval. This distinction is not made in what follows.

There is a simple geometric interpretation for how the x_i 's are positioned in the interval that comes from (5.33). Placing a semi-circle over the interval, as in Figure 5.16, consider the $n + 1$ points on the semi-circle that are a constant angle π/n apart, with the first one (on the far right) at an angle $\pi/(2n)$. Their x coordinates are given in (5.32) and they are the corresponding Chebyshev points. This figure also shows why the Chebyshev points are closer together at the endpoints of the interval, as compared to their placement towards the center.

In the proof of Theorem 5.7, the following result is also obtained:

Theorem 5.8. *With the x_i 's given in (5.32), then the global interpolation polynomial $p_n(x)$ given in (5.5) satisfies*

$$|f(x) - p_n(x)| \leq \frac{1}{2^n(n + 1)!} \left(\frac{b - a}{2}\right)^{n+1} \|f^{(n+1)}\|_\infty, \text{ for } a \leq x \leq b,$$

where $\|f^{(n+1)}\|_\infty = \max_{a \leq x \leq b} |f^{(n+1)}(x)|$.

The fact that $2^n(n + 1)!$ grows rapidly with n means that the error with Chebyshev interpolation can be quite small. To get an estimate of just how small, according to Stirling's formula, for large values of n ,

$$n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n. \quad (5.34)$$

Using this approximation for the factorial, the inequality in Theorem 5.8 can be replaced with (see Exercise 5.31)

$$|f(x) - p_n(x)| \leq \sqrt{\frac{2}{n\pi}} R^{n+1} \|f^{(n+1)}\|_\infty, \text{ for } a \leq x \leq b, \quad (5.35)$$

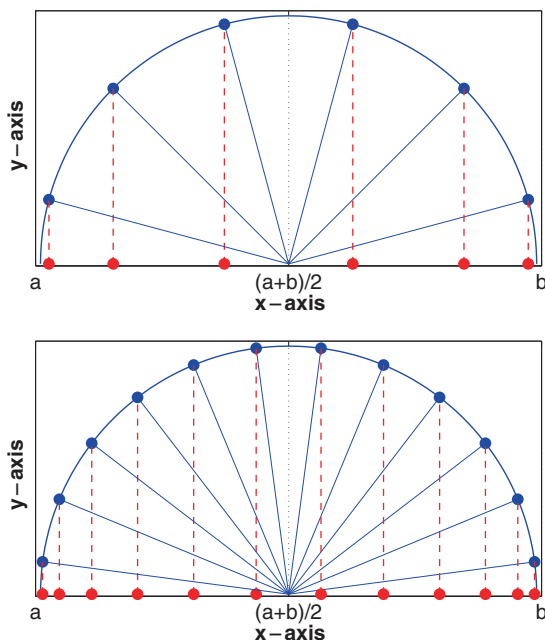


Figure 5.16 The Chebyshev points, which are the red dots along the x -axis, are determined by equally spaced points on the circumscribed semi-circle. In the top graph, $n = 5$, while in the bottom graph, $n = 11$.

where

$$R = \frac{(b-a)e}{4(n+1)}. \quad (5.36)$$

Consequently, if n is large enough that $R < 1$, then R^{n+1} approaches zero exponentially fast. Whether this means that the error for Chebyshev interpolation approaches zero exponentially fast, however, depends on how the

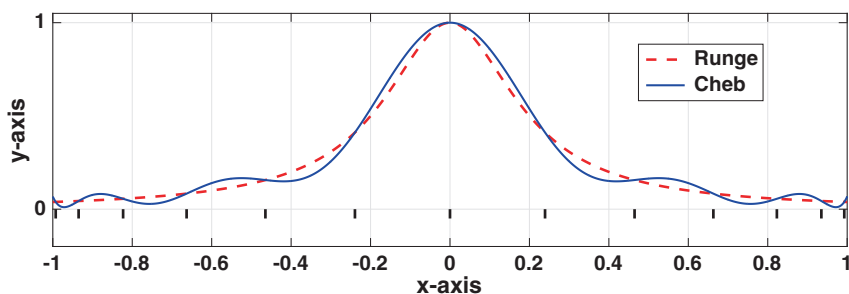


Figure 5.17 The Runge function (5.8) and the Chebyshev interpolation polynomial with $n = 12$.

$f^{(n+1)}$ term depends on n . This issue is considered in the examples to follow. Also, it's worth noting that the error using piecewise linear or cubic splines is not exponential, and the error in each case approaches zero as a fixed power of $h = (b - a)/(n - 1)$.

Examples

1. Using the Chebyshev points with $n = 12$ to fit Runge's function in (5.8), the interpolation function shown in Figure 5.17 is obtained. The improvement over using equally spaced points, which is shown in Figure 5.5, is dramatic. Also shown in Figure 5.17, by small horizontal bars, are the locations of the x_i 's. This shows the non-uniform spacing of the interpolation points, and they get closer together as you approach either of the endpoints of the interval. ■
2. For $f(x) = \cos(2\pi x)$, where $0 \leq x \leq 1$, according to Theorem 5.8, how many interpolation points are needed to guarantee an error of 10^{-4} when using Chebyshev interpolation?

Since $\|f^{(n+1)}\|_\infty = (2\pi)^{n+1}$, then we want $\pi^{n+1}/(2^n(n+1)!) \leq 10^{-4}$. From this one finds that $n \geq 9$. In comparison, earlier we found that piecewise linear requires $n \geq 1397$, while a clamped cubic spline requires $n \geq 23$. It is also interesting to note that if the number of points is doubled to $n = 18$, that according to Theorem 5.8, the error bound is about 10^{-14} , and if doubled again to $n = 36$, the error bound is an astonishing 10^{-36} . In contrast, for a clamped cubic spline, doubling the number of points decreases the error bound by a factor of $2^{-4} \approx 6 \times 10^{-2}$. This is a clear demonstration of the benefits of an exponentially converging approximation, but as we will see shortly, exponential convergence is limited to certain types of functions. ■

Chebyshev Polynomials

To explain how the x_i 's are determined, it is assumed that $a = -1$ and $b = 1$. We begin with a definition.

Definition 5.1. The Chebyshev polynomials $T_n(x)$ are defined using the following recursion formula:

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \text{ for } n = 1, 2, 3, \dots, \quad (5.37)$$

where $T_0(x) = 1$ and $T_1(x) = x$.

Using this definition, the first few Chebyshev polynomials are

$$\begin{aligned}
T_2(x) &= 2x^2 - 1, \\
T_3(x) &= 4x^3 - 3x, \\
T_4(x) &= 8x^4 - 8x^2 + 1, \\
T_5(x) &= 16x^5 - 20x^3 + 5x.
\end{aligned}$$

As is seen in the above expressions, $T_n(x)$ is an n th degree polynomial and the leading coefficient is 2^{n-1} .

Two of the key results needed for finding the x_i 's are contained in the following result:

Theorem 5.9.

1. If $P_n(x) = x^n + b_{n-1}x^{n-1} + \cdots + b_0$, where $n \geq 1$, then

$$\max_{-1 \leq x \leq 1} |P_n(x)| \geq 2^{1-n}.$$

2. If $P_n(x) = 2^{1-n}T_n(x)$, then

$$\max_{-1 \leq x \leq 1} |P_n(x)| = 2^{1-n}.$$

The usual proof of the first statement involves contradiction, and using the oscillatory properties of a polynomial. An illustration of how it is possible to prove it directly is given in Exercise 5.32.

Note that the $q_{n+1}(x)$ in (5.26) is an example of the function $P_{n+1}(x)$ appearing in the above theorem. The first result in the theorem states that no matter what we pick for the x_i 's, the Q in (5.31) satisfies $Q \geq 2^{-n}$. What the second result states is that if we pick $q_{n+1}(x) = 2^{-n}T_{n+1}(x)$, then $Q = 2^{-n}$, i.e., it achieves the stated minimum value. Therefore, we should pick the x_i 's so that

$$2^{-n}T_{n+1}(x) = (x - x_1)(x - x_2) \cdots (x - x_{n+1}).$$

In other words, the x_i 's are the zeros of $T_{n+1}(x)$.

We need an easy way to find the zeros of $T_{n+1}(x)$, and this is given in the next result.

Theorem 5.10.

$$T_n(x) = \cos(n \cos^{-1} x), \text{ for } n = 0, 1, 2, 3, \dots$$

This is a strange looking equation because the right-hand side does not look to be a polynomial. Nevertheless, the proof is rather simple, and basically

involves using trig identities to show that the right-hand side satisfies (5.37). With this it is easy to find the zeros of $T_{n+1}(x)$, and they are the values of x that satisfy

$$(n+1)\cos^{-1}x = \frac{\pi}{2}(2i-1), \text{ for } i = 1, 2, 3, \dots, n+1.$$

The values given in (5.32) are the resulting positions when the above result is transformed from $-1 \leq x \leq 1$ to $a \leq x \leq b$.

5.5.5 Chebyshev Versus Cubic Splines

We saw that Chebyshev interpolation has the potential to have an error that approaches zero exponentially fast as n increases. It also has the distinction that it produces the smallest Q , as explained in Theorem 5.7. The natural cubic splines, on the other hand, produce the smallest total curvature squared, as defined in Theorem 5.1. What this means is that we have two interpolation methods that can claim to be optimal. Given this, it is of interest to compare Chebyshev and cubic spline interpolation on some more challenging examples.

Example 1

We begin with the Runge's function in (5.8), and assume 13 data points are used. The resulting Chebyshev interpolation function is shown in Figure 5.17. For comparison, the corresponding natural cubic spline is shown in Figure 5.18. In comparing the two figures, it is clear that the spline provides a better approximation function. To have a more quantitative comparison, suppose $g(x)$

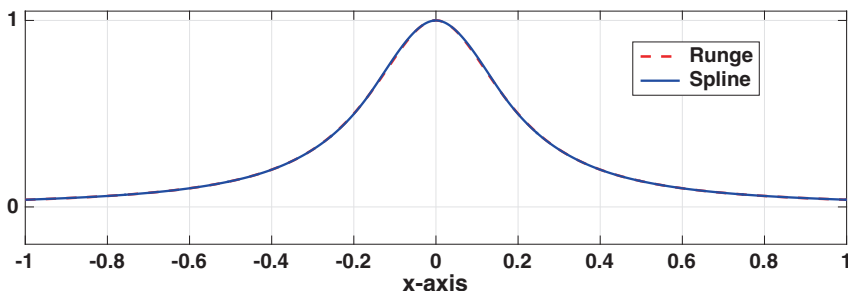


Figure 5.18 The function (5.8) and the natural cubic spline using 13 equally spaced points. The two curves are indistinguishable.

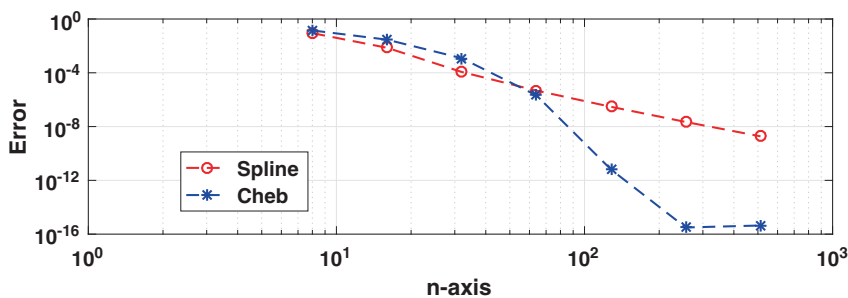


Figure 5.19 The error, as determined by the area integral in (5.38), when approximating Runge's function with a natural cubic spline, and with a Chebyshev interpolation function.

is an interpolation function of a given function $f(x)$. The error is using $g(x)$ to approximate $f(x)$ will be determined using the area between the two curves, which means

$$E = \int_a^b |f(x) - g(x)| dx. \quad (5.38)$$

The value of this integral is given in Figure 5.19, when $g(x)$ is the natural cubic spline, and when it is the Chebyshev interpolation function. It is seen that the spline produces a more accurate approximation when using up to about 60 interpolation points. The reason the spline does better is that $\|f^{(n+1)}\|_\infty$ grows rapidly with n , which effectively eliminates the exponential convergence for Chebyshev interpolation. For example, when $n = 13$, $\|f^{(n+1)}\|_\infty \approx 5 \times 10^{20}$, while $2^n(n+1)! \approx 3 \times 10^{13}$. It is not until n is rather large that the exponential convergence kicks in and Chebyshev begins to produce a better approximation than the spline. ■

Example 2

Suppose the function is $f(x) = \tanh(100x - 30)$, and the interval is $0 \leq x \leq 1$. Using 25 interpolation points, the resulting interpolation functions are shown in Figure 5.20. In this case, both have some difficulty with the rapid rise in the function. However, the under- and over-shoots in the spline function die out much faster than those for the Chebyshev function. The associated error for each interpolation function, as determined using (5.38), is shown in Figure 5.21. As in the last example, it is not until n is rather large that the exponential convergence enables Chebyshev to produce a better approximation than the spline. ■

It is evident from the examples that Chebyshev interpolation is capable of producing a more accurate approximation than cubic splines. However, this

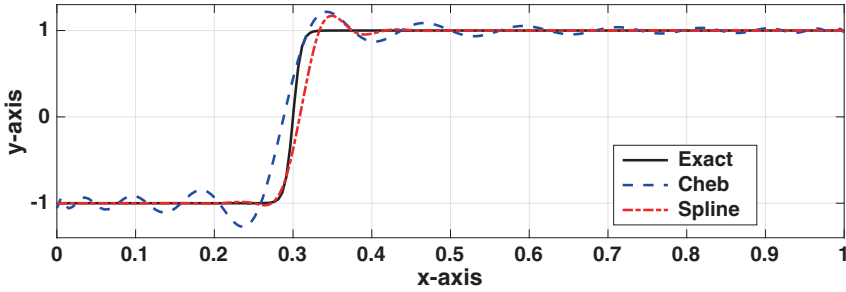


Figure 5.20 The function $f(x) = \tanh(100x - 30)$, along with Chebyshev and cubic spline interpolation functions using 25 data points.

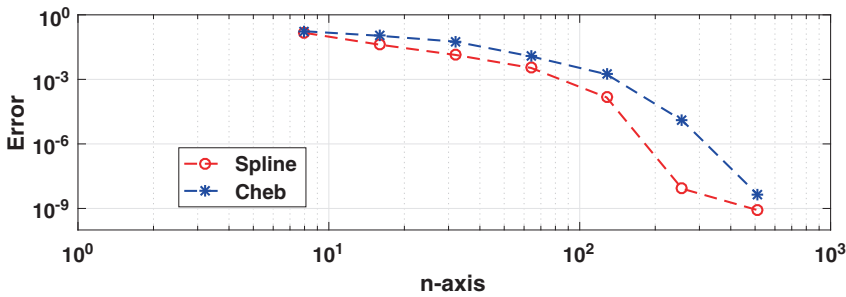


Figure 5.21 The error, as determined by the area integral in (5.38), when approximating $f(x) = \tanh(100x - 30)$ with a natural cubic spline, and with a Chebyshev interpolation function.

is based on the stipulation that the contribution of the $f^{(n+1)}(\eta)$ term in the error is not too large, and it can be difficult to know if this holds. This limits its usefulness, but it does not dampen the enthusiasm that some have for the method. To get insight into why they think this way, Trefethen [2012] should be consulted.

5.5.6 Other Ideas

There are a variety ways of modifying the interpolation procedure. For example, given a function $f(x)$ one can construct a piecewise cubic that interpolates $f(x_1), f(x_2), \dots, f(x_{n+1})$ as well as the derivative values $f'(x_1), f'(x_2), \dots, f'(x_{n+1})$. This is known as Hermite interpolation and it produces an approximation with an error that is $O(h^4)$. This puts it in the same category as the clamped cubic spline discussed earlier.

There is also the idea of being monotone. The objective here is that if the data appear to describe a monotonically increasing (or decreasing) function

over an interval then the interpolation function should behave the same way over that interval. As seen in the lower right data set in Figures 5.4 and 5.13, the global and spline functions fail at this while the piecewise linear function in Figure 5.9 works. However, for the latter there is the usual problem with corners. So, the goal is to find a method that does well at preserving monotonicity and is also smooth. This comes under the more general heading of finding a smooth shape preserving interpolation function. A review of such methods, such as the Akima algorithm and the Fritsch-Butland procedure, can be found in Huynh [1993]. It is also worth noting that shape preserving interpolation is of particular interest in the mathematical finance community, and those interested in this application should consult Hagan and West [2006].

5.6 Questions and Additional Comments

Below are some random questions and comments about interpolation.

1. To fix the corner problem that arises with piecewise linear interpolation, we used piecewise cubic interpolation. What's wrong with piecewise quadratics?

Answer: They have a couple of drawbacks. To explain, quadratics can interpolate the data and have a continuous first derivative (see Exercise 5.33). In comparison, cubic splines have continuous second derivatives, and so they are smoother. Another issue with quadratics is that they can produce what can best be described as bumpy curves, and this is illustrated in Exercise 5.33(d). However, it is possible to adjust the interpolation procedure to improve the situation and those interested might want to consult Marsden [1974] or Grasselli and Pelinovsky [2008].

2. The cubic B-spline $B_i(x)$ is nonzero for $x_{i-2} \leq x \leq x_{i+2}$. Why not use the smaller interval $x_{i-1} \leq x \leq x_{i+1}$?

Answer: It simply won't work (try it).

3. Can the interpolation methods be used to solve the puzzle in Figure 5.6?

Answer: Yes, but if you want to draw something that looks like a flower then you will need to rewrite the problem because our methods are based on interpolating a function. Possible solutions would be to break the data points into sections that can each be described as a function, or to use parametric coordinates. These ideas can be generalized, which leads

naturally to something called Bezier curves and surfaces, and geometric modeling. Those interested might consider looking at Salomon [2006] and Mortenson [1997].

4. If just one of the y_i 's is changed, what happens to the interpolation function?

Answer: It depends on what method you are using. For piecewise linear, the

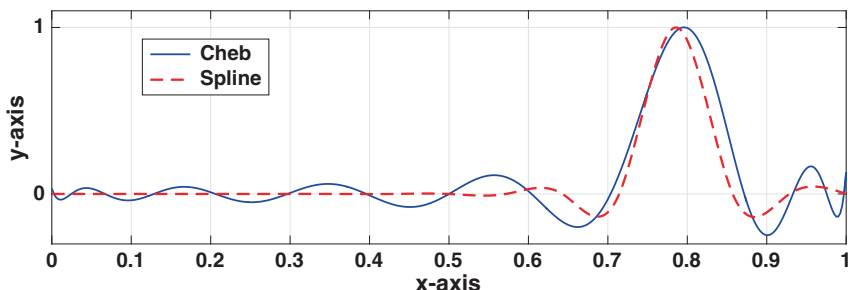


Figure 5.22 Interpolation using Chebyshev and cubic spline interpolation functions with 15 data points, with all values zero except for the one near 0.8.

interpolation function is only affected over the interval $x_{i-1} < x < x_{i+1}$. If you are using cubic splines or Chebyshev, then the interpolation function over the entire interval $a < x < b$ is affected. To get an idea of what happens, if all of the y_i 's are zero then the interpolation function is zero everywhere, irrespective of whether or not you are using a natural cubic spline or Chebyshev interpolation. Now, suppose one data point is changed and it is now nonzero. This situation is shown in Figure 5.22 using a natural cubic spline and Chebyshev interpolation, where the nonzero point is the one close to 0.8 (the exact point differs between the two methods due to how they position the points). In both cases, the changes in the interpolation function over the entire interval are less than the change at the given data point. In other words, if the data value is changed by a small amount, then the interpolation function over the interval is changed by no more than this value. However, it is also apparent that the changes in the Chebyshev function are more widespread than for the cubic spline.

5. In the early days of spline research, they used the cubic spline version of the piece linear hat functions $G_i(x)$. These functions were usually designated as $L_i(x)$ and they were defined as the cubic splines that satisfied $L_i(x_i) = 1$ and $L_i(x_j) = 0$ for $j \neq i$ (see, e.g., de Boor and Schoenberg 1976). These were called the fundamental functions, and one is shown in Figure 5.23. The reason for introducing them is that they have the nice property that the spline interpolation function is simply

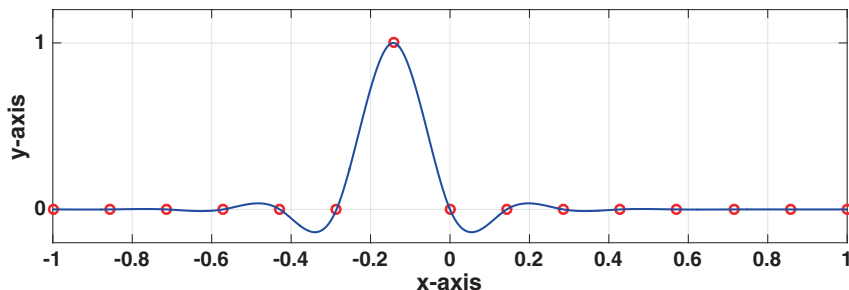


Figure 5.23 A fundamental function for (clamped) cubic splines. Shown is $L_7(x)$.

$$s(x) = \sum_{i=1}^{n+1} y_i L_i(x).$$

However, this hides all the work needed to determine the spline. In particular, finding the L_i 's requires the solution of $n + 1$ matrix equations.

Exercises

5.1. In this problem the data are: $(x_1, y_1) = (0, 0)$, $(x_2, y_2) = (1, 1)$, and $(x_3, y_3) = (2, 3)$.

- Find the global interpolation polynomial that fits these data.
- Find the piecewise linear interpolation function that fits these data.
- Find the natural cubic spline that fits these data.

5.2. This problem concerns the data in Table 5.6.

- Find the piecewise linear interpolation function $g(x)$ that fits these data.
- Find the global interpolation polynomial $p_3(x)$ that fits these data.

5.3. Use a Lagrange interpolating polynomial of degree 1 to find an approximate value for the following. Not all of the data points are needed, and you should explain which ones you use and why.

- $f(2.4)$ if $f(2.1) = 1$, $f(2.3) = 1.2$, $f(2.6) = 1.3$, $f(2.7) = 2$
- $f(-0.1)$ if $f(0.1) = 2$, $f(0) = 0.1$, $f(-0.2) = -0.1$, $f(0.4) = -0.5$
- $f(1)$ if $f(0.5) = -1$, $f(0.8) = -0.5$, $f(1.1) = 0.5$, $f(1.2) = 1$

5.4. Redo the previous problem but use a Lagrange interpolating polynomial of degree 2.

5.5. If $f(\theta) = \sin \theta$, then $f(0) = 0$, $f(\pi/4) = \sqrt{2}/2$, and $f(\pi/2) = 1$. Use these data points to answer the following questions. Note that the error that is asked for is the absolute value of the difference between the exact value $f(\pi/8) = \sqrt{2} - \sqrt{2}/2$ and the estimated value.

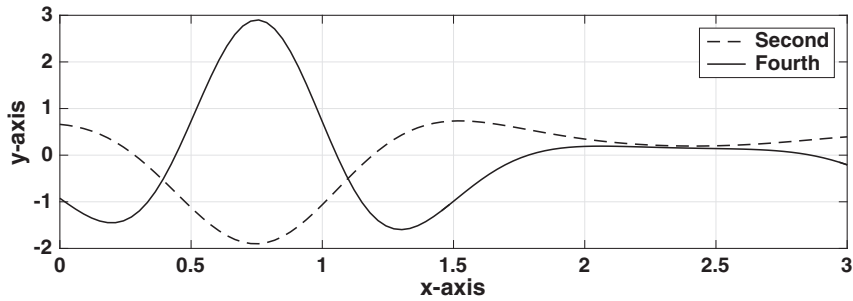


Figure 5.24 Graph used in Exercise 5.6.

x	-1	0	1	2
y	0	1	1	0

Table 5.6 Data for Exercise 5.2.

- (a) Using piecewise linear interpolation, what is the estimated value of $f(\pi/8)$? What is the error in this estimate?
- (b) Using a global interpolation polynomial, what is the estimated value of $f(\pi/8)$? What is the error in this estimate?
- (c) Using natural cubic spline interpolation, what is the estimated value of $f(\pi/8)$? What is the error in this estimate?
- (d) Using the additional information that $f'(0) = 1$ and $f'(\pi/2) = 0$, use clamped cubic spline interpolation to find an estimated value of $f(\pi/8)$. What is the error in this estimate?
- (e) Suppose Chebyshev interpolation is used. Determine the three Chebyshev points in the interval, and evaluate $f(\theta)$ at these points. What is the resulting estimated value of $f(\pi/8)$? What is the error in this estimate?

5.6. A function $f(x)$ is going to be approximated using an interpolation function for $0 \leq x \leq 3$. The second, $f''(x)$, and fourth, $f^{(4)}(x)$, derivatives of the function are plotted in Figure 5.24.

- (a) How many data points for piecewise linear interpolation are needed to guarantee the error is less than 10^{-8} ?
- (b) How many data points for a clamped cubic spline are needed to guarantee the error is less than 10^{-8} ?

5.7. The function $y = \log_{10} x$ is going to be approximated using an interpolation function for $1 \leq x \leq 10$.

- (a) How many data points for piecewise linear interpolation are needed to guarantee the error is less than 10^{-6} ?

- (b) How many data points for a clamped cubic spline are needed to guarantee the error is less than 10^{-6} ?
- (c) How many data points are needed when using Chebyshev interpolation to guarantee the error is less than 10^{-6} ?

5.8. For the following functions, determine a step size h that will guarantee that the error is less than 10^{-6} using piecewise linear interpolation.

- (a) $f(x) = x^{10}$, for $-1 \leq x \leq 1$.
- (b) $f(x) = \ln(x)$, for $1 \leq x \leq 2$.
- (c) $f(x) = 2\sin(3x) + 3\sin(2x)$, for $0 \leq x \leq \pi$.

5.9. Redo the previous problem but use a clamped cubic spline.

5.10. The Bessel function of order zero can be written as

$$J_0(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin s) ds.$$

- (a) Show that $|J_0(x)| \leq 1$, $|J'_0(x)| \leq 1$, $|J''_0(x)| \leq 1$, and in fact, for any positive integer k ,

$$\left| \frac{d^k}{dx^k} J_0(x) \right| \leq 1.$$

In what follows, determine how many interpolation points over the interval $0 \leq x \leq 10$ are needed so the error is no more than 10^{-6} .

- (b) Using piecewise linear interpolation.
- (c) Using a clamped cubic spline.
- (d) Using Chebyshev interpolation.
- (e) The Bessel function of order m can be defined as

$$J_m(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin s - ms) ds.$$

How do your answers in parts (b)–(d) change for this function?

5.11. This problem considers the function

$$g(x) = \begin{cases} 2 + 3x^2 + \alpha x^3 & \text{if } -1 \leq x \leq 0, \\ 2 + \beta x^2 - x^3 & \text{if } 0 \leq x \leq 1. \end{cases}$$

- (a) For what values of α and β , if any, is $g(x)$ a cubic spline for $-1 \leq x \leq 1$?
These values are to be used in the remainder of this problem.
- (b) What were the data points that gave rise to this cubic spline?
- (c) For what values of α and β is $g(x)$ a natural cubic spline?
- (d) For what values of α and β is $g(x)$ a clamped cubic spline?

5.12. This problem considers the function

$$g(x) = \begin{cases} 2(x+1)^3 + 5(x+1) - 13x & \text{if } -1 \leq x \leq 0, \\ 2(1-x)^3 + 9x + 5(1-x) & \text{if } 0 \leq x \leq 1. \end{cases}$$

- Show that this is a cubic spline, and determine the data values used in its construction.
- Is this a natural cubic spline?
- Is this a clamped cubic spline?

5.13. Consider the function

$$g(x) = \begin{cases} x^3 - 1 & \text{if } 0 \leq x \leq 1, \\ -x^3 + 6x^2 - 6x + 1 & \text{if } 1 \leq x \leq 2. \end{cases}$$

Is $g(x)$ a cubic spline for $0 \leq x \leq 2$? If it is, is it natural, clamped, or neither? Make sure to justify your answers.

5.14. The data considered here are the population of a country for the years $x_1 = 1900$, $x_2 = 1910$, $x_3 = 1920$, $x_4 = 1930$, \dots , $x_{12} = 2010$. The y_i 's are the corresponding population values, and they should be given per million. For example, the population of the USA is given in Table 5.7, and this is from the Wikipedia page *Demographics of the United States*.

- Fit this data with: i) a global polynomial using Lagrange interpolation, and ii) a natural cubic spline. Plot these two curves and the data on the same axis. Make sure to include a legend in your plot.
- What do each of the two interpolation functions give as the population in 2005?
- What do each of the two interpolation functions predict the population will be in 2015?

5.15. The data considered here are the temperatures over a 24 hour period, in two hour increments. So, $x_1 = 0$, $x_2 = 2$, $x_3 = 4$, $x_4 = 6$, \dots , $x_{13} = 24$. The y_i 's are the corresponding temperatures. For example, the temperatures in Troy, NY on June 21 are given in Table 5.8 (in $^{\circ}\text{F}$).

- Fit this data with: i) a global polynomial using Lagrange interpolation, and ii) a natural cubic spline. Plot these two curves and the data on the same axis. Make sure to include a legend in your plot.

x	1900	1910	1920	1930	1940	1950	1960	1970	1980	1990	2000	2010
y	76.21	92.23	106.0	123.2	132.2	151.3	179.3	203.3	226.5	248.8	281.4	308.7

Table 5.7 Sample population data for Exercises 5.14, 5.28, and 6.9.

- (b) What do each of the two interpolation functions give as the temperature at 11 AM?
- (c) What do the two interpolation functions predict the temperature will be at 1 AM the next day?
- (d) What do the two interpolation functions predict the temperature will be at 9 AM the next day? Explain why the spline predicts the value it does.

5.16. The data below describe a cross-section of an airfoil where the points (X, Y_u) define the upper surface and the points (X, Y_ℓ) describe the lower surface.

$X = [0, 0.005, 0.0075, 0.0125, 0.025, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]$

$Y_u = [0, 0.0102, 0.0134, 0.017, 0.025, 0.0376, 0.0563, 0.0812, 0.0962, 0.1035, 0.1033, 0.095, 0.0802, 0.0597, 0.034, 0]$

$Y_\ell = [0, -0.0052, -0.0064, -0.0063, -0.0064, -0.006, -0.0045, -0.0016, 0.001, 0.0036, 0.007, 0.0121, 0.017, 0.0199, 0.0178, 0]$

- (a) Draw the airfoil by fitting cubic splines separately to the upper and lower surfaces, and then plotting the results as a single figure. To make it look like an airfoil you will probably need to resize the plot window.
- (b) Redo (a) but use global polynomial interpolation instead of splines.

5.17. This problem considers some of the difficulties interpolating the function $f(x) = \sqrt{x}$.

- (a) If the interpolation interval is $1 \leq x \leq 10$, how many data points are needed for piecewise linear interpolation to guarantee that the error is less than 10^{-6} ?
- (b) Explain why Theorem 5.4 is not so useful if the interval is $0 \leq x \leq 1$.
- (c) One way to deal with the singularity at $x = 0$ is to break the interval into two segments, one is $0 \leq x \leq \delta$ and the other is $\delta \leq x \leq 1$, where δ is a small positive number. On the interval $0 \leq x \leq \delta$ the function is going to be interpolated with a single line. What is the equation for this line, and how small does δ need to be to guarantee that the approximation error is 10^{-6} ?
- (d) Assuming that δ is known, how many data points over the interval $\delta \leq x \leq 1$ are needed for piecewise linear interpolation to guarantee that the error is less than 10^{-6} ?

5.18. The function $f(x) = 1/(1 + x^2)$ is to be approximated using a piecewise linear function $g(x)$ over the interval $0 \leq x < \infty$. The requirement is that $|f(x) - g(x)| \leq 10^{-4}$ for $0 \leq x < \infty$. Explain how to determine the spacing of the x_i 's used in the construction of $g(x)$, and how you handle the approximation over the intervals $x_n \leq x \leq x_{n+1}$ and $x_{n+1} \leq x < \infty$, where x_{n+1} is the largest node you use.

5.19. This problem concerns interpolating the function $f(x) = \sin \pi x$ over the interval $1 \leq x \leq 3$, using three data points, with $x_1 = 1$, $x_2 = 2$, and $x_3 = 3$.

x	0	2	4	6	8	10	12	14	16	18	20	22	24
y	59	56	53	54	60	67	72	74	75	74	70	65	61

Table 5.8 Sample temperature data for Exercise 5.15. Note $x = 0$ and $x = 24$ correspond to midnight.

- Find the global interpolation polynomial that fits this data.
- Find the piecewise linear interpolation function that fits this data.
- Find the natural cubic spline that fits this data.
- Find the clamped cubic spline that fits this data.
- Chebyshev interpolation cannot use the stated x_i 's. What are the three Chebyshev interpolation points for this interval, and what is the resulting interpolation function?

5.20. In this problem $x_i = i$, for $i = 1, 2, 3, 4$, and

$$s(x) = B_0(x) - B_1(x) + B_2(x) - B_3(x) + B_4(x) - B_5(x),$$

for $1 \leq x \leq 4$.

- What data points (x_i, y_i) were used to produce this cubic spline?
- Is this a natural cubic spline?
- Is it a clamped cubic spline?

5.21. In this problem $x_i = i$, for $i = 1, 2, 3, 4$, and

$$s(x) = B_2(x) + 5B_4(x),$$

for $1 \leq x \leq 4$.

- What data points (x_i, y_i) were used to produce this cubic spline?
- Is this a natural cubic spline?
- Is it a clamped cubic spline?

5.22. Given a data set $(x_1, y_1), (x_2, y_2), \dots, (x_{101}, y_{101})$ suppose one of the following interpolation methods is to be used: Lagrange interpolation, piecewise linear interpolation, cubic spline interpolation using cubic B-splines.

- Order them by the number of flops needed to determine the interpolation function. Make sure to explain how you arrive at your answer. Assume the Thomas algorithm is used for the cubic spline (see Section 3.8).
- Order the methods by the number of flops needed to evaluate them at a given point (assume this point isn't in the data set). Make sure to explain how you arrive at your answer.

5.23. Suppose Chebyshev interpolation is applied to a function $f(x)$ using eight interpolation points. Also, suppose that no matter what interpolation interval is used, the error for the Chebyshev interpolation is zero. What conclusion can you make about the original function $f(x)$?

5.24. Given three points x_{i-1} , x_i , and x_{i+1} , this problem considers the quadratic interpolation formula

$$p_2(x) = y_{i-1}\ell_{i-1}(x) + y_i\ell_i(x) + y_{i+1}\ell_{i+1}(x).$$

It is assumed here that $x_i - x_{i-1} = h$ and $x_{i+1} - x_i = h$.

(a) Show that the above interpolation formula can be rewritten as

$$p_2(x) = y_i + \frac{1}{2h}(y_{i+1} - y_{i-1})(x - x_i) + \frac{1}{2h^2}(y_{i+1} - 2y_i + y_{i-1})(x - x_i)^2.$$

(b) Calculate $p'_2(x)$ and $p''_2(x)$.

(c) Suppose $p_2(x)$ interpolates $f(x)$ at x_{i-1} , x_i , and x_{i+1} , so $y_{i-1} = f(x_{i-1})$, $y_i = f(x_i)$, and $y_{i+1} = f(x_{i+1})$. Setting $x = x_i + \alpha h$, expand $f(x)$ and $p_2(x)$ about $h = 0$ and show that

$$f(x) = p_2(x) + \frac{1}{6}z(z^2 - h^2)f'''(x_i) + \frac{1}{24}z^2(z^2 - h^2)f''''(x_i) + \cdots,$$

where $z = x - x_i$.

(d) How does the result in part (c) compare to the result in Theorem 5.2 in the case of when $a = x_i - h$, $b = x_i + h$, and $n = 2$?

5.25. It is possible when solving for the coefficients for a cubic spline that one of the s_i 's turns out to be a linear function (versus a full cubic). This exercise explores this situation. Suppose there are three data points, with $x_1 = 0$, $x_2 = 1$, and $x_3 = 2$ and let

$$s(x) = \begin{cases} a + bx & \text{if } 0 \leq x \leq 1, \\ a_2 + b_2(x-1) + c_2(x-1)^2 + d_2(x-1)^3 & \text{if } 1 \leq x \leq 2. \end{cases}$$

(a) To be a cubic spline it is required that $s \in C^2(0, 2)$. What conditions must be imposed on the coefficients so this happens?

(b) Under what conditions, if any, is this a natural cubic spline?

5.26. This exercise explores some of the differences between a cubic polynomial and a cubic spline. In this problem the data are: $(x_1, y_1) = (0, 0)$, $(x_2, y_2) = (1, 1)$, $(x_3, y_3) = (2, 0)$, and $(x_4, y_4) = (3, 1)$.

(a) Find the global interpolation polynomial that fits this data, and then evaluate this function at $x = 1/2$.

(b) Find the natural cubic spline that fits this data, and then evaluate this function at $x = 1/2$.

(c) The cubic in part (a) satisfies the interpolation and smoothness conditions required of a spline, yet it produces a different result than the cubic spline in part (b). Why?

(d) What boundary conditions should be used so the cubic spline produces the cubic in part (a)?

5.27. The objective of this problem is to find a method that can evaluate $f(x) = \cos x$, for $0 \leq x \leq 2\pi$, with an error of no more than 10^{-6} . In doing this, the interpolation points are restricted to those x_i 's for which the exact value of $\cos x_i$ is known. It is useful to know that, by considering the angles in a polygon, it is possible to determine the exact values of $\cos x$ and $\sin x$ for $x = \pi/10, \pi/12, \pi/15$, etc. (these are given on the Wikipedia page *Exact trigonometric constants*).

- Show that if the values of $\cos x$ and $\sin x$ are known for $x = \pi/k$, then they are known at $x = m\pi/k$, for $m = 2, 3, 4, \dots$.
- For a given value of k , let $h = \pi/k$ and suppose that the interpolation points are $x_i = (i-1)h$, for $i = 1, 2, \dots, n+1$. Find n in terms of k .
- According to Theorem 5.4, how small must h be so the error using piecewise linear interpolation with $f(x) = \cos x$ is no more than 10^{-6} ? What is the smallest value of k so that $\pi/k \leq h$?
- According to Theorem 5.6, how small must h be so the error using a clamped cubic spline with $f(x) = \cos x$ is no more than 10^{-6} ? What is the smallest value of k so that $\pi/k \leq h$?
- For a given value of x , describe a procedure that uses the exact values of $\cos x$ and/or $\sin x$ to evaluate $f(x) = \cos x$, for $0 \leq x \leq 2\pi$, with an error of less than 10^{-6} .
- Write a MATLAB program that implements your algorithm in part (e) and compares the computed values with MATLAB's built in cosine function, for $x = 1, 2, 5$.

5.28. This problem explores how to scale the data to help improve the computability of the interpolation polynomial. We consider the direct approach to determine $p_n(x)$, and as usual the data points are $(x_1, y_1), (x_2, y_2), \dots, (x_{n+1}, y_{n+1})$, where $x_1 < x_2 < \dots < x_{n+1}$. The x values are going to be scaled by letting

$$z = \frac{x - \alpha}{\beta},$$

where α and β are given numbers with $\beta > 0$. The data point (x_i, y_i) in this case changes to (z_i, y_i) , where $z_i = (x_i - \alpha)/\beta$. Also, the interpolation polynomial also changes to

$$p_n(z) = a_0 + a_1 z + \dots + a_n z^n.$$

- The original data interval is $x_1 \leq x \leq x_{n+1}$. What is the data interval when using z ? What matrix equation must be solved to find the a_i 's in the above formula for $p_n(z)$?
- The values for α and β are going to be selected so the z data interval is $-1 \leq z \leq 1$. What are α and β in this case?
- Using the population data from Exercise 5.14, plot the interpolation function using the direct approach on the original x_i data set. Also compute the condition number for \mathbf{V} .

- (d) Using the population data from Exercise 5.14, scale the data based on the result from part (b), and then find the coefficients for $p_n(z)$. What is the condition number of the matrix in this case? Once the a_i 's are computed then in terms of the original x variable,

$$p_n(x) = a_0 + a_1 \left(\frac{x - \alpha}{\beta} \right) + a_2 \left(\frac{x - \alpha}{\beta} \right)^2 + \cdots + a_n \left(\frac{x - \alpha}{\beta} \right)^n.$$

Plot this function and compare the result with what you found in part (c).

5.29. This problem concerns a method to reduce the computational effort to evaluate the Lagrange interpolation function given in (5.5).

- (a) What is the flop count to evaluate (5.5) for a given value of x ?
 (b) Assuming that $x \neq x_i$, for any i , show that (5.5) can be written as

$$p_n(x) = \ell(x) \sum_{i=1}^{n+1} w_i y_i / (x - x_i),$$

where $\ell(x) = \prod_{j=1}^{n+1} (x - x_j)$ and

$$w_i = 1 / \prod_{\substack{j=1 \\ j \neq i}}^{n+1} (x_i - x_j).$$

This is known as the first form of the barycentric interpolation formula, and w_i 's are called barycentric weights.

- (c) Suppose the formula in part (b) is used to interpolate the constant function $f(x) = 1$. Use this to show that

$$\ell(x) \sum_{i=1}^{n+1} w_i / (x - x_i) = 1.$$

- (d) Use the result from part (c) to show that

$$p_n(x) = \frac{\sum_{i=1}^{n+1} w_i y_i / (x - x_i)}{\sum_{i=1}^{n+1} w_i / (x - x_i)},$$

This is called the second (true) form of the barycentric formula.

- (e) What is the flop count to evaluate the formula for $p_n(x)$ given in part (d)?

5.30. The cubic spline interpolation function is

$$s(x) = \sum_{i=0}^{n+2} a_i B_i(x).$$

For a natural spline we found the a_i 's by solving a matrix equation $\mathbf{A}\mathbf{a} = \mathbf{z}$. The purpose of this exercise is to find what this equation is for a clamped spline. Recall that for a clamped spline it is required that $s'(x_1) = y'_1$ and $s'(x_{n+1}) = y'_{n+1}$, where y'_1 and y'_{n+1} are given. Show $\mathbf{a} = (a_1, a_2, \dots, a_{n+1})^T$, $\mathbf{z} = (6y_1 + 2hy'_1, 6y_2, \dots, 6y_n, 6y_{n+1} - 2hy'_{n+1})^T$, and \mathbf{A} is the $n \times n$ tridiagonal matrix

$$\mathbf{A} = \begin{pmatrix} 4 & 2 & & & \\ 1 & 4 & 1 & & 0 \\ & 1 & 4 & 1 & \\ & & \ddots & \ddots & \ddots \\ 0 & & & 1 & 4 & 1 \\ & & & & 2 & 4 \end{pmatrix}.$$

Also, once \mathbf{a} is determined, then $a_0 = a_2 - 2hy'_1$ and $a_{n+2} = a_n + 2hy'_{n+1}$.

5.31. This problem concerns some of the inequalities arising for function interpolation.

- (a) For q_{n+1} , given in (5.26), assume x is not one of the x_i 's. So, there is an x_i so that $x_i < x < x_{i+1}$. With this, it is possible to write

$$q_{n+1}(x) = (x - x_i)(x - x_{i+1}) \prod_{j=1}^{i-1} (x - x_j) \prod_{j=i+2}^{n+1} (x - x_j).$$

Show that $|(x - x_i)(x - x_{i+1})| \leq \frac{1}{4}h^2$. Also show that $|\prod_{j=1}^{i-1} (x - x_j)| \leq i!h^{i-1}$ and $|\prod_{j=i+2}^{n+1} (x - x_j)| \leq (n+1-i)!h^{n-i}$. From this, derive (5.27). Make sure to comment about the case of when x equals one of the x_i 's.

- (b) It is possible to prove that for every positive integer n [Sandor and Debnath, 2000],

$$\sqrt{2\pi n} \left(\frac{n}{e}\right)^n \leq n!.$$

Use this, and Theorem 5.8, to derive (5.35).

5.32. This problem considers a direct proof of Theorem 5.9, at least for the case of when $n = 1$. This will help demonstrate how this result is independent of the coefficients of the polynomial.

- (a) If $P_1(x) = x + b_0$, explain why

$$\max_{-1 \leq x \leq 1} |P_1(x)| = \max\{|1 + b_0|, |-1 + b_0|\}.$$

- (b) Sketch the two absolute values in part (a) as a function of b_0 . Use this to explain why $\max_{-1 \leq x \leq 1} |P_1(x)| = 1 + |b_0|$. From this derive the result stated in the theorem.

5.33. This problem derives the formulas for a piecewise quadratic interpolation function. This function is written as

$$w(x) = \begin{cases} w_1(x) & \text{if } x_1 \leq x \leq x_2 \\ w_2(x) & \text{if } x_2 \leq x \leq x_3 \\ \vdots & \vdots \\ w_n(x) & \text{if } x_n \leq x \leq x_{n+1}, \end{cases}$$

where

$$w_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2, \quad \text{for } x_i \leq x \leq x_{i+1}.$$

The interpolation requirements are $w_i(x_i) = y_i$ and $w_i(x_{i+1}) = y_{i+1}$. Also, $w'(x)$ is required to be continuous, which means that $w'_i(x_{i+1}) = w'_{i+1}(x_{i+1})$ for $i = 1, 2, \dots, n-1$.

- (a) Explain why the stated requirements are not enough, and it is necessary to impose one additional condition. In this problem, it is assumed that b_1 is specified. Explain why this is the same as specifying the value of the slope $w'(x_1)$.
- (b) From the stated requirements, deduce that $a_i = y_i$ for $i = 1, 2, \dots, n+1$. Also, $b_i = -b_{i-1} + 2(y_i - y_{i-1})/h$ for $i = 2, 3, \dots, n$, and $c_i = (y_{i+1} - y_i - hb_i)/h^2$ for $i = 1, 2, \dots, n$.
- (c) Explain why the results from parts (a) and (b) mean that one can determine the coefficients for w_1 , then determine the coefficients for w_2 , then for w_3 , etc.
- (d) Use $w(x)$ to interpolate $f(x) = \cos(6\pi x)$, over the interval $0 \leq x \leq 1$. Plot $w(x)$ and $f(x)$ for the following cases: i) $n = 5$, ii) $n = 10$, iii) $n = 15$, and iv) $n = 30$. Comment on how well the interpolation method works for this particular function.

Introduction to Scientific Computing and Data Analysis

Holmes, M.H.

2016, XIV, 497 p. 177 illus., 138 illus. in color.,

Hardcover

ISBN: 978-3-319-30254-6