Chapter 4

Numerical Differentiation and Integration

4.1 Numerical Differentiation

In this section, we introduce how to numerically calculate the derivative of a function. First, the derivative of the function $f$ at $x_0$ is defined as

$$f'(x_0) := \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$ 

This formula gives an obvious way to generate an approximation to $f'(x_0)$: simply compute

$$\frac{f(x_0 + h) - f(x_0)}{h}$$

for small values of $h$. Although this way may be obvious, it is not very successful, due to our old nemesis round-off error. But it is certainly a place to start.

To approximate $f'(x_0)$, suppose that $x_0 \in (a, b)$, where $f \in C^2[a, b]$, and that $x_1 = x_0 + h$ for some $h \neq 0$ that is sufficiently small to ensure that $x_1 \in [a, b]$. We construct the first Lagrange polynomial $P_{0,1}(x)$ for $f$ determined by $x_0$ and $x_1$, with its error term:

$$f(x) = P_{0,1}(x) + \frac{(x - x_0)(x - x_1)}{2!} f^{\prime\prime}(\xi(x))$$

$$= \frac{f(x_0)(x - x_0 - h)}{-h} + \frac{f(x_0 + h)(x - x_0)}{h} + \frac{(x - x_0)(x - x_0 - h)}{2} f^{\prime\prime}(\xi(x))$$

for some $\xi(x)$ between $x_0$ and $x_1$. Differentiating gives

$$f'(x) = \frac{f(x_0 + h) - f(x_0)}{h} + D_x \left[ \frac{(x - x_0)(x - x_0 - h)}{2} f^{\prime\prime}(\xi(x)) \right]$$

$$= \frac{f(x_0 + h) - f(x_0)}{h} + \frac{2(x - x_0) - h}{2} f^{\prime\prime}(\xi(x))$$

$$+ \frac{(x - x_0)(x - x_0 - h)}{2} D_x \left( f^{\prime\prime}(\xi(x)) \right).$$

Deleting the terms involving $\xi(x)$ gives

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h}.$$
One difficulty with this formula is that we have no information about \( D_x \left( f''(\xi(x)) \right) \), so the truncation error cannot be estimated. When \( x \) is \( x_0 \), however, the coefficient of \( D_x \left( f''(\xi(x)) \right) \) is 0, and the formula simplifies to

\[
f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{h}{2} f''(\xi).
\]

(4.1)

For small values of \( h \), the difference quotient \( \left[ f(x_0 + h) - f(x_0) \right]/2 \) can be used to approximate \( f'(x_0) \) with an error bounded by \( M|h|/2 \), where \( M \) is a bound on \( |f''(x)| \) for \( x \) between \( x_0 \) and \( x_0 + h \). This formula is known as the forward-difference formula for \( h > 0 \) and the backward-difference formula if \( h < 0 \).

**Example 4.1.1.** Use the forward-difference formula to approximate the derivative of \( f(x) = \ln x \) at \( x_0 = 1.8 \) using \( h = 0.1, h = 0.05 \), and \( h = 0.01 \), and determine bounds for the approximation errors.

**Solution.** The forward-difference formula

\[
\frac{f(1.8 + h) - f(1.8)}{h}
\]

with \( h = 0.1 \) gives

\[
\frac{\ln 1.9 - \ln 1.8}{0.1} = \frac{0.64185389 - 0.58778667}{0.1} = 0.5496722.
\]

Because \( f''(x) = -1/x^2 \) and \( 1.8 < \xi < 1.9 \), a bound for this approximation error is

\[
\frac{|h|f''(\xi)|}{2} = \frac{|h|}{2\xi^2} < \frac{0.1}{2(1.8)^2} = 0.0154321.
\]

The approximation and error bounds when \( h = 0.05 \) and \( h = 0.01 \) are found in a similar manner and the results are shown in the following table.

| \( h \) | \( f(1.8 + h) \) | \( \frac{f(1.8 + h) - f(1.8)}{h} \) | \( \frac{|h|}{2(1.8)^2} \) |
|---|---|---|---|
| 0.1 | 0.64185389 | 0.5406722 | 0.0154321 |
| 0.05 | 0.61518564 | 0.5479795 | 0.0077160 |
| 0.01 | 0.59332685 | 0.5540180 | 0.0015432 |

Since \( f'(x) = 1/x \), the exact value of \( f'(1.8) \) is 0.555, and in this case the error bounds are quite close to the true approximation error.

To obtain general derivative approximation formulas, suppose that \( x_0, x_1, \ldots, x_n \) are \( n + 1 \) distinct numbers in some interval \( I \) and that \( f \in C^{n+1}(I) \). Using the Lagrange interpolation formula, we have

\[
f(x) = \sum_{k=0}^{n} f(x_k)L_k(x) + \frac{(x-x_0)\cdots(x-x_n)}{(n+1)!} f^{(n+1)}(\xi(x)),
\]

for some \( \xi(x) \in I \), where \( L_k(x) \) denotes the \( k \)-th Lagrange polynomial for \( f \) at \( x_0, x_1, \ldots, x_n \). Differentiating this expression gives

\[
f'(x) = \sum_{k=0}^{n} f(x_k)L'_k(x) + D_x \left[ \frac{(x-x_0)\cdots(x-x_n)}{(n+1)!} \right] f^{(n+1)}(\xi(x)) + \frac{(x-x_0)\cdots(x-x_n)}{(n+1)!} D_x \left[ f^{(n+1)}(\xi(x)) \right].
\]
We again have a problem estimating the truncation error unless \( x \) is one of the numbers \( x_j \). In this case, the last term vanishes and the formula becomes

\[
f'(x_j) = \sum_{k=0}^{n} f(x_k) L_k'(x_j) + \frac{f^{(n+1)}(\xi(x_j))}{(n+1)!} \prod_{k=0, k \neq j}^{n} (x_j - x_k),
\]

which is called an \((n+1)\)-point formula to approximate \( f'(x_j) \).

In general, using more evaluation points in (4.2) produces greater accuracy, although the number of functional evaluations and growth of round-off error discourages this somewhat. The most common formulas are those involving three and five evaluation points. We first derive some useful three-point formulas and consider aspects of their errors. Since

\[ L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} \implies L_0'(x) = \frac{2x - x_1 - x_2}{(x_0 - x_1)(x_0 - x_2)}, \]

Similarly, we have

\[ L_1(x) = \frac{2x - x_0 - x_2}{(x_1 - x_0)(x_1 - x_2)} \quad \text{and} \quad L_2(x) = \frac{2x - x_0 - x_1}{(x_2 - x_0)(x_2 - x_1)}. \]

Hence, the equation (4.2) becomes (with \( n = 2 \))

\[
f'(x) = f(x_0) \left[ \frac{2x_j - x_1 - x_2}{(x_0 - x_1)(x_0 - x_2)} \right] + f(x_1) \left[ \frac{2x_j - x_0 - x_2}{(x_1 - x_0)(x_1 - x_2)} \right]
+ f(x_2) \left[ \frac{2x_j - x_0 - x_1}{(x_2 - x_0)(x_2 - x_1)} \right] + \frac{1}{6} f^{(3)}(\xi_j) \prod_{k=0, k \neq j}^{2} (x_j - x_k),
\]

for each \( j = 0, 1, 2 \), where the notation \( \xi_j \) indicates that this points depends on \( x_j \).

**Three-point formulas**

The formula from (4.3) become especially useful if the nodes are equally spaced, that is, when

\[ x_1 = x_0 + h \quad \text{and} \quad x_2 = x_0 + 2h \quad \text{for some} \ h \neq 0. \]

We will assume equally-spaced nodes throughout the remainder of this section. Using (4.3) with \( x_j = x_0, x_1 = x_0 + h, \) and \( x_2 = x_0 + 2h \), it gives

\[ f'(x_0) = \frac{1}{h} \left[ -\frac{3}{2} f(x_0) + 2f(x_1) - \frac{1}{2} f(x_2) \right] + \frac{h^2}{3} f^{(3)}(\xi_0). \]

Doing the same for \( x_j = x_1 \) gives

\[ f'(x_1) = \frac{1}{h} \left[ -\frac{1}{2} f(x_0) + \frac{1}{2} f(x_2) \right] - \frac{h^2}{6} f^{(3)}(\xi_1), \]

and for \( x_j = x_2 \),

\[ f'(x_2) = \frac{1}{h} \left[ \frac{1}{2} f(x_0) - 2f(x_1) + \frac{3}{2} f(x_2) \right] + \frac{h^2}{3} f^{(3)}(\xi_2). \]

Since \( x_1 = x_0 + h \) and \( x_2 = x_0 + 2h \), these formulas can also be expressed as

\[ f'(x_0) = \frac{1}{h} \left[ -\frac{3}{2} f(x_0) + 2f(x_0 + h) - \frac{1}{2} f(x_0 + 2h) \right] + \frac{h^2}{3} f^{(3)}(\xi_0), \]
\[ f'(x_0 + h) = \frac{1}{h} \left[ -\frac{1}{2} f(x_0) + \frac{1}{2} f(x_0 + 2h) \right] - \frac{h^2}{6} f^{(3)}(\xi_1), \]

and
\[ f'(x_0 + 2h) = \frac{1}{h} \left[ \frac{2}{3} f(x_0) - 2f(x_0 + h) + \frac{3}{2} f(x_0 + 2h) \right] + \frac{h^2}{3} f^{(3)}(\xi_2). \]

As a matter of convenience, the variable substitution \( x_0 \) for \( x_0 + h \) is used in the middle equation to change this formula to an approximation for \( f'(x_0) \). A similar change, \( x_0 \) for \( x_0 + 2h \), is used in the last equation. This gives three formulas for approximating \( f'(x_0) \):

\[ f'(x_0) = \frac{1}{2h} \left[ -3f(x_0) + 4f(x_0 + h) - f(x_0 + 2h) \right] + \frac{h^2}{3} f^{(3)}(\xi_0), \]
\[ f'(x_0) = \frac{1}{2h} \left[ f(x_0 + h) - f(x_0 - h) \right] - \frac{h^2}{6} f^{(3)}(\xi_1), \]

and
\[ f'(x_0) = \frac{1}{2h} \left[ 3f(x_0) - 4f(x_0 + h) + f(x_0 + 2h) \right] + \frac{h^2}{3} f^{(3)}(\xi_2). \]

Finally, note that the last of these equations can be obtained from the first by simply replacing \( h \) with \(-h\), so there are actually only two formulas:

**Three-point Endpoint formula**

\[ f'(x_0) = \frac{1}{2h} \left[ -3f(x_0) + 4f(x_0 + h) - f(x_0 + 2h) \right] + \frac{h^2}{3} f^{(3)}(\xi_0), \quad (4.4) \]

where \( \xi_0 \) lies between \( x_0 \) and \( x_0 + 2h \).

**Three-point Midpoint formula**

\[ f'(x_0) = \frac{1}{2h} \left[ f(x_0 + h) - f(x_0 - h) \right] - \frac{h^2}{6} f^{(3)}(\xi_1), \quad (4.5) \]

where \( \xi_1 \) lies between \( x_0 - h \) and \( x_0 + h \).

Although the errors in both (4.4) and (4.5) are \( O(h^2) \), the error in (4.5) is approximately half the error in (4.4). This is because (4.5) uses data on both sides of \( x_0 \) and (4.4) uses data on only one side. Note also that \( f \) needs to be evaluated at only two points in (4.5), whereas in (4.4), three evaluations are needed. The approximation in (4.4) is useful near the ends of an interval, because information about \( f \) outside the interval may not be available.

The methods presented in (4.4) and (4.5) are called **three-point formulas**. Similarly, there are **five-point formulas** that involve evaluating the function at two additional points. The error term for these formulas is \( O(h^4) \). One common five-point formula is used to determine approximations for the derivative at the midpoint.

**Five-point Midpoint formula**

\[ f'(x_0) = \frac{1}{12h} \left[ f(x_0 - 2h) - 8f(x_0 - h) + 8f(x_0 + h) - f(x_0 + 2h) \right] + \frac{h^4}{30} f^{(5)}(\xi), \quad (4.6) \]

where \( \xi \) lies between \( x_0 - 2h \) and \( x_0 + 2h \). The derivation of this formula is considered in later section. The other five-point formula is used for approximations at the endpoints.
Five-point Midpoint formula

\[ f'(x_0) = \frac{1}{12h} \left[ -25f(x_0) + 48f(x_0 + h) - 36f(x_0 + 2h) \\
+ 16f(x_0 + 3h) - 3f(x_0 + 4h) \right] + \frac{h^4}{5} f^{(5)}(\xi) \]  

(4.7)

where \( \xi \) lies between \( x_0 \) and \( x_0 + 4h \). Left-endpoint approximations are found using this formula with \( h > 0 \) and right-endpoint approximations with \( h < 0 \).

**Example 4.1.2.** Values for \( f(x) = xe^x \) are given in the following table. Use all the applicable three-point and five-point formulas to approximate \( f'(2.0) \).

<table>
<thead>
<tr>
<th>( x )</th>
<th>1.8</th>
<th>1.9</th>
<th>2.0</th>
<th>2.1</th>
<th>2.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(x) )</td>
<td>10.889365</td>
<td>12.703199</td>
<td>14.778112</td>
<td>17.148957</td>
<td>19.855030</td>
</tr>
</tbody>
</table>

**Solution.** The data in the table permit us to find four different three-point approximations. We can use the endpoint formula (4.4) with \( h = 0.1 \) or with \( h = -0.1 \), and we can use the midpoint formula (4.5) with \( h = 0.1 \) or with \( h = 0.2 \).

- Using the endpoint formula (4.4) with \( h = 0.1 \), it gives
  \[ f'(2.0) \approx \frac{1}{0.2} \left[ -3f(2.0) + 4f(2.1) - f(2.2) \right] = 5[-3(14.778112) + 4(17.148957) - 19.855030] = 22.032310. \]

- Using the endpoint formula (4.4) with \( h = -0.1 \), it gives \( f'(x_0) \approx 22.054525. \)

- Using the midpoint formula (4.5) with \( h = 0.1 \), it gives
  \[ f'(x_0) \approx \frac{1}{0.2} \left[ f(2.1) - f(1.9) \right] = 5(17.148957 - 12.7703199) = 22.228790. \]

- Using the midpoint formula (4.5) with \( h = 0.2 \), it gives \( f'(x_0) \approx 22.414163. \)

- The only five-point for which the table gives sufficient data is the midpoint formula (4.6) with \( h = 0.1 \). This gives
  \[ f'(x_0) \approx \frac{1}{1.2} \left[ f(1.8) - 8f(1.9) + 8f(2.1) - f(2.2) \right] = \frac{1}{1.2} \left[ 10.889365 - 8(12.703199) + 8(17.148957) - 19.855030 \right] = 22.166999. \]

If we had no other information we would accept the five-point midpoint approximation using \( h = 0.1 \) as the most accurate, and expect the true value to be between that approximation and the three-point mid-point approximation that is in the interval \([22.166, 22.229]\).

The true value in this case is \( f'(2.0) = (2 + 1)e^2 = 22.167168 \), so the approximation errors are actually:

- Three-point endpoint with \( h = 0.1 \): \( 1.35 \times 10^{-1} \);
- Three-point endpoint with \( h = -0.1 \): \( 1.13 \times 10^{-1} \);
• Three-point midpoint with \( h = 0.1: 6.16 \times 10^{-2}; \)
• Three-point midpoint with \( h = 0.2: 2.47 \times 10^{-1}; \)
• Five-point midpoint with \( h = 0.1: 1.69 \times 10^{-4}. \)

Methods can also be derived to find approximations to higher derivatives of a function using only tabulated values of the function at various points. The derivation is algebraically tedious, however, so only a representative procedure will be presented.

Expand a function \( f \) in a third Taylor polynomial about a point \( x_0 \) and evaluate at \( x_0 + h \) and \( x_0 - h \). Then, we have

\[
f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2} f''(x_0)h^2 + \frac{1}{6} f'''(x_0)h^3 + \frac{1}{24} f^{(4)}(\xi_1)h^4,
\]
and

\[
f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2} f''(x_0)h^2 - \frac{1}{6} f'''(x_0)h^3 + \frac{1}{24} f^{(4)}(\xi_{-1})h^4,
\]

where \( x_0 - h < \xi_{-1} < x_0 < \xi_1 < x_0 + h \). If we add these equations, the terms involving \( f'(x_0) \) and \(-f'(x_0)\) cancel, so we have

\[
f(x_0 + h) + f(x_0 - h) = 2f(x_0) + f''(x_0)h^2 + \frac{1}{24}[f^{(4)}(\xi_1) + f^{(4)}(\xi_{-1})]h^4.
\]

Solving this equation for \( f''(x_0) \) gives

\[
f''(x_0) = \frac{1}{h^2}[f(x_0 - h) - 2f(x_0) + f(x_0 + h)] - \frac{h^2}{24}[f^{(4)}(\xi_1) + f^{(4)}(\xi_{-1})]. \tag{4.8}
\]

Suppose \( f^{(4)} \) is continuous on \([x_0 - h, x_0 + h]\). Since \( \frac{1}{2}[f^{(4)}(\xi_1) + f^{(4)}(\xi_{-1})] \) is between \( f^{(4)}(\xi_1) \) and \( f^{(4)}(\xi_{-1}) \), the Intermediate Value Theorem implies that a number \( \xi \) exists between \( \xi_1 \) and \( \xi_{-1} \), and hence in \((x_0 - h, x_0 + h)\) with

\[
f^{(4)}(\xi) = \frac{1}{2}[f^{(4)}(\xi_1) + f^{(4)}(\xi_{-1})].
\]

This allows us to rewrite (4.8) in its final form

\[
f''(x_0) = \frac{1}{h^2}[f(x_0 - h) - 2f(x_0) + f(x_0 + h)] - \frac{h^2}{12} f^{(4)}(\xi) \tag{4.9}
\]

for some \( \xi \in (x_0 - h, x_0 + h) \). If \( f^{(4)} \) is continuous on \([x_0 - h, x_0 + h]\), it is also bounded, and the approximation is \( O(h^2) \).

**Example 4.1.3.** In Example 4.1.2, we used the data to approximate the first derivative of \( f(x) = xe^x \) at \( x = 2.0 \). Use the second derivative formula (4.9) to approximate \( f''(2.0) \).

**Solution.** The data permits us to determine two approximations for \( f''(2.0) \). Using (4.9) with \( h = 0.1 \) gives

\[
f''(x_0) \approx \frac{1}{0.01}[f(1.9) - 2f(2.0) + f(2.1)] = 100[12.703199 - 2(14.778112) + 17.148957]
= 29.593200
\]
and using \(4.9\) with \(h = 0.2\) gives
\[
 f''(x_0) \approx \frac{1}{0.04} [f(1.8) - 2f(2.0) + f(2.2)] = 25[10.889365 - 2(14.778112) + 19.855030] \\
= 29.704275.
\]

Because \(f''(x) = (x + 2)e^x\), the exact value is \(f''(2.0) = 29.556224\). Hence, the actual errors are \(-3.7 \times 10^{-2}\) and \(-1.48 \times 10^{-1}\), respectively.

**ROUND-OFF ERROR INSTABILITY**

It is particularly important to pay attention to round-off error when approximating derivatives. To illustrate the situation, let us examine the three-point midpoint formula \(4.5\)
\[
f'(x_0) = \frac{1}{2h} [f(x_0 + h) - f(x_0 - h)] - \frac{h^2}{6} f^{(3)}(\xi)
\]
more closely. Suppose that in evaluating \(f(x_0 + h)\) and \(f(x_0 - h)\) we encounter round-off errors \(e(x_0 + h)\) and \(e(x_0 - h)\). Then, our computations actually use the values \(\hat{f}(x_0 + h)\) and \(\hat{f}(x_0 - h)\), which are related to the true values by
\[
f(x_0 + h) = \hat{f}(x_0 + h) + e(x_0 + h) \quad \text{and} \quad f(x_0 - h) = \hat{f}(x_0 - h) + e(x_0 - h).
\]
The total error in the approximation,
\[
f'(x_0) - \frac{\hat{f}(x_0 + h) - \hat{f}(x_0 - h)}{2h} = \frac{e(x_0 + h) - e(x_0 - h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi_1),
\]
is due both to round-off error, the first part, and to truncation error. If we assume that the round-off error \(e(x_0 \pm h)\) are bounded by some number \(\varepsilon > 0\) and that the third derivative of \(f\) is bounded by a number \(M > 0\), then
\[
\left| f'(x_0) - \frac{\hat{f}(x_0 + h) - \hat{f}(x_0 - h)}{2h} \right| \leq \frac{\varepsilon}{h} + \frac{h^2}{6} M.
\]

To reduce the truncation error, \(h^2 M/6\), we need to reduce \(h\). But as \(h\) is reduced, the round-off error \(\varepsilon/h\) grows. In practice, then, it is seldom advantageous to let \(h\) be too small, because in that case the round-off error will dominate the calculations.

**Example 4.1.4.** Consider using the values in the following table to approximate \(f'(0.900)\), where \(f(x) = \sin x\). The true value is \(\cos(0.900) = 0.62161\).

<table>
<thead>
<tr>
<th>(x)</th>
<th>(\sin x)</th>
<th>(x)</th>
<th>(\sin x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.800</td>
<td>0.71736</td>
<td>0.901</td>
<td>0.78395</td>
</tr>
<tr>
<td>0.850</td>
<td>0.75128</td>
<td>0.902</td>
<td>0.78457</td>
</tr>
<tr>
<td>0.880</td>
<td>0.77074</td>
<td>0.905</td>
<td>0.78643</td>
</tr>
<tr>
<td>0.890</td>
<td>0.77707</td>
<td>0.910</td>
<td>0.78950</td>
</tr>
<tr>
<td>0.895</td>
<td>0.78021</td>
<td>0.920</td>
<td>0.79560</td>
</tr>
<tr>
<td>0.898</td>
<td>0.78208</td>
<td>0.950</td>
<td>0.81342</td>
</tr>
<tr>
<td>0.899</td>
<td>0.78270</td>
<td>1.000</td>
<td>0.84147</td>
</tr>
</tbody>
</table>

**Solution.** The formula
\[
f'(0.900) \approx \frac{f(0.900 + h) - f(0.900 - h)}{2h}
\]
with different values of \(h\) gives the approximation and the results are listed in the following table.
Approximation

<table>
<thead>
<tr>
<th>$h$</th>
<th>to $f'(0.900)$</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.001</td>
<td>0.62500</td>
<td>0.00339</td>
</tr>
<tr>
<td>0.002</td>
<td>0.62550</td>
<td>0.00089</td>
</tr>
<tr>
<td>0.005</td>
<td>0.62200</td>
<td>0.00039</td>
</tr>
<tr>
<td>0.010</td>
<td>0.62150</td>
<td>0.00011</td>
</tr>
<tr>
<td>0.020</td>
<td>0.62150</td>
<td>0.00011</td>
</tr>
<tr>
<td>0.050</td>
<td>0.62140</td>
<td>0.00021</td>
</tr>
<tr>
<td>0.100</td>
<td>0.62055</td>
<td>0.00106</td>
</tr>
</tbody>
</table>

The optimal choice for $h$ appears to lie between 0.005 and 0.05. We can use calculus to verify that a minimum for the error function

$$e(h) = \frac{\varepsilon}{h} + \frac{h^2}{6} M,$$

occurs at $h = (3\varepsilon/M)^{1/3}$, where

$$M = \max_{x\in[0.800,1.000]} |f'''(x)| = \max_{x\in[0.800,1.000]} |\cos x| = \cos 0.8 \approx 0.69671.$$

Because values of $f$ are given to five decimal places, we will assume that the round-off error is bounded by $\varepsilon = 5 \times 10^{-6}$. Therefore, the optimal choice of $h$ is approximately

$$h = \left(\frac{3\varepsilon}{M}\right)^{1/3} \approx 0.028,$$

which is consistent with the results in the table above.

In practice, we cannot compute an optimal $h$ to use in approximating the derivative, since we have no knowledge of the third derivative of the function. But we must remain aware that reducing the step size will not always improve the approximation.

4.2 Richardson’s Extrapolation

In this section, we introduce a method to generate high-accuracy results while using low-order formulas. This method is called Richardson’s extrapolation. It can be applied whenever it is known that an approximation technique has an error term with a predictable form, one that depends on a parameter, usually the step size $h$.

Suppose that for each number $h \neq 0$, we have a formula $N_1(h)$ that approximates an unknown constant $M$, and that the truncation error involved with the approximation has the form

$$M - N_1(h) = K_1 h + K_2 h^2 + K_3 h^3 + \cdots,$$

for some collection of (unknown) constants $K_1, K_2, \ldots$. The truncation error is $O(h)$, so unless there was a large variation in magnitude among the constants, in general, $M - N_1(h) \approx K_1 h$.

The object of extrapolation is to find an easy way to combine these rather inaccurate rate $O(h)$ approximations in an appropriate way to produce formulas with a higher-order truncation error.

To see specifically how we can generate the extrapolation formulas, consider the $O(h)$ formula for approximating $M$:

$$M = N_1(h) + K_1 h + K_2 h^2 + K_3 h^3 + \cdots.$$
The formula is assumed to hold for all positive $h$, so we replace the parameter $h$ by half its value. Then, we have
\[ M = N_1 \left( \frac{h}{2} \right) + K_1 \frac{h}{2} + K_2 \frac{h^2}{4} + K_3 \frac{h^3}{8} + \cdots. \] (4.11)
Subtracting (4.10) from twice (4.11) eliminates the term involving $K_1$ and gives
\[ M = N_1 \left( \frac{h}{2} \right) + \left[ N_1 \left( \frac{h}{2} \right) - N_1(h) \right] + K_2 \left( \frac{h^2}{2} - h^2 \right) + K_3 \left( \frac{h^3}{4} - h^3 \right) + \cdots. \] (4.12)
Define
\[ N_2(h) = N_1 \left( \frac{h}{2} \right) + \left[ N_1 \left( \frac{h}{2} \right) - N_1(h) \right] = 2N_1 \left( \frac{h}{2} \right) - N_1(h). \]
Then (4.12) is an $O(h^2)$ approximation formula for $M$:
\[ M = N_2(h) - \frac{K_2}{2} h^2 - \frac{3K_3}{4} h^3 - \cdots. \] (4.13)

**Example 4.2.1.** In Example 4.1.1 we use the forward-difference method with $h = 0.1$ and $h = 0.05$ to find approximations to $f'(1.8)$ for $f(x) = \ln x$. Assume that this formula has truncation error $O(h)$ and use extrapolation on these values to see if this results in a better approximation.

**Solution.** In Example 4.1.1 we found that $f'(1.8) \approx 0.5406722$ with $h = 0.1$ and $f'(1.8) \approx 0.5479795$ with $h = 0.05$. This implies that
\[ N_1(0.1) = 0.5406722 \quad \text{and} \quad N_1(0.05) = 0.5479795. \]
Extrapolating these results gives the new approximation
\[ N_2(0.1) = N_1(0.05) + (N_1(0.05) - N_1(0.1)) = 0.555287. \]
The $h = 0.1$ and $h = 0.05$ results were found to be accurate within $1.5 \times 10^{-2}$ and $7.7 \times 10^{-3}$, respectively. Because $f'(1.8) = 1/1.8 = 0.55555\cdots$, the extrapolated value is accurate to within $2.7 \times 10^{-4}$.

In general, extrapolation can be applied whenever the truncation error for a formula has the form
\[ \sum_{j=1}^{m-1} K_j h^{\alpha_j} + O(h^{\alpha_m}) \]
for a collection of constants $K_j$ and when $\alpha_1 < \alpha_2 < \alpha_3 < \cdots < \alpha_m$. Many formulas used for extrapolation have truncation errors that contain only even powers of $h$, that is, have the form
\[ M = N_1(h) + K_1 h^2 + K_2 h^4 + K_3 h^6 + \cdots. \] (4.14)
Assume that approximation has the form of (4.14). Replacing $h$ with $h/2$ gives the approximation formula
\[ M = N_1 \left( \frac{h}{2} \right) + K_1 \frac{h^2}{4} + K_2 \frac{h^4}{16} + K_3 \frac{h^6}{64} + \cdots. \]
Subtracting (4.14) from 4 times this equation eliminates the $h^2$ term,
\[ 3M = \left[ 4N_1 \left( \frac{h}{2} \right) - N_1(h) \right] + K_2 \left( \frac{h^4}{4} - h^4 \right) + K_3 \left( \frac{h^6}{16} - h^6 \right) + \cdots. \]
Dividing this equation by 3 produces an $O(h^4)$ formula

$$M = \frac{1}{3} \left[ 4N_1 \left( \frac{h}{2} \right) - N_1(h) \right] + \frac{K_2}{3} \left( \frac{h^4}{4} - h^4 \right) + \frac{K_3}{3} \left( \frac{h^6}{16} - h^6 \right) + \cdots. $$

Defining

$$N_2(h) = \frac{1}{3} \left[ 4N_1 \left( \frac{h}{2} \right) - N_1(h) \right] = N_1 \left( \frac{h}{2} \right) + \frac{1}{3} \left[ N_1 \left( \frac{h}{2} \right) - N_1(h) \right],$$

produces the approximation formula with truncation error $O(h^4)$:

$$M = N_2(h) - K_2 \frac{h^4}{4} - K_3 \frac{5h^6}{16} + \cdots. \quad (4.15)$$

Now replace $h$ in (4.15) with $h/2$ to produce a second $O(h^4)$ formula

$$M = N_2 \left( \frac{h}{2} \right) - K_2 \frac{h^4}{64} - K_3 \frac{5h^6}{1024} + \cdots.$$

Subtracting (4.15) from 16 times this equation eliminates the $h^4$ term and gives

$$15M = \left[ 16N_2 \left( \frac{h}{2} \right) - N_2(h) \right] + K_3 \frac{15h^6}{64} + \cdots.$$

Dividing this equation by 15 produces the new $O(h^6)$ formula

$$M = \frac{1}{15} \left[ 16N_2 \left( \frac{h}{2} \right) - N_2(h) \right] + \frac{h^6}{64} + \cdots.$$

We now have the $O(h^6)$ approximation formula

$$N_3(h) = \frac{1}{15} \left[ 16N_2 \left( \frac{h}{2} \right) - N_2(h) \right] = N_2 \left( \frac{h}{2} \right) + \frac{1}{15} \left[ N_2 \left( \frac{h}{2} \right) - N_2(h) \right].$$

Continuing this procedure gives, for each $j = 2, 3, \cdots$, the $O(h^{2j})$ approximation

$$N_j(h) = N_{j-1} \left( \frac{h}{2} \right) + \frac{N_j-1(h/2) - N_j-1(h)}{4^{j-1} - 1}.$$

**Example 4.2.2.** Taylor’s expansion can be used to show that the Three-Point Midpoint formula (4.5) to approximate $f'(x_0)$ can be expressed with an error formula:

$$f'(x_0) = \frac{1}{2h} \left[ f(x_0 + h) - f(x_0 - h) \right] - \frac{h^2}{6} f''(x_0) - \frac{h^4}{120} f''(5)(x_0) - \cdots.$$

Find approximations of order $O(h^2)$, $O(h^4)$, and $O(h^6)$ for $f''(2.0)$ when $f(x) = x e^x$ and $h = 0.2$.

**Solution.** The constants $K_1 = -f'''(x_0)/6$, $K_2 = -f''(5)(x_0)/120$, $\cdots$, are not likely to be known, but this is not important. We only need to know that these constants exist in order to apply extrapolation. We have the $O(h^2)$ approximation

$$f'(x_0) = N_1(h) - \frac{h^2}{6} f''(x_0) - \frac{h^4}{120} f''(5)(x_0) - \cdots,$$

where

$$N_1(h) = \frac{1}{2h} \left[ f(x_0 + h) - f(x_0 - h) \right].$$
This gives us the first $O(h^2)$ approximations

$$N_1(0.2) = \frac{1}{0.4} [f(2.2) - f(1.8)] = 22.414160 \quad \text{and} \quad N_2(0.1) = \frac{1}{0.2} [f(2.1) - f(1.9)] = 22.228786.$$ 

Combining these to produce the first $O(h^4)$ approximation gives

$$N_2(0.2) = N_1(0.1) + \frac{1}{3}(N_1(0.1) - N_1(0.2)) = 22.166995.$$ 

To determine an $O(h^6)$ formula we need another $O(h^4)$ result, which requires us to find the third $O(h^2)$ approximation

$$N_1(0.05) = \frac{1}{0.1} [f(2.05) - f(1.95)] = 22.182564.$$ 

We can now find the $O(h^4)$ approximation

$$N_2(0.1) = N_1(0.05) + \frac{1}{3}(N_1(0.05) - N_1(0.1)) = 22.167157$$ 

and finally the $O(h^6)$ approximation

$$N_3(0.2) = N_2(0.1) + \frac{1}{15}(N_2(0.1) - N_2(0.2)) = 22.167168.$$ 

We would expect the final approximation to be accurate to at least the value 22.167 because the $N_2(0.2)$ and $N_3(0.2)$ give the same value. In fact, $N_3(0.2)$ is accurate to all the listed digits. 

The following table shows the order in which the approximations are generated when

$$M = N_1(h) + K_1 h^2 + K_2 h^4 + K_3 h^6 + \cdots.$$ 

<table>
<thead>
<tr>
<th></th>
<th>$O(h^2)$</th>
<th>$O(h^4)$</th>
<th>$O(h^6)$</th>
<th>$O(h^8)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>$N_1(h)$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2:</td>
<td>$N_1\left(\frac{h}{2}\right)$</td>
<td>$N_2(h)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:</td>
<td>$N_1(h)$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4:</td>
<td>$N_2\left(\frac{h}{2}\right)$</td>
<td>$N_2(h)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5:</td>
<td>$N_1(h)$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6:</td>
<td>$N_3(h)$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7:</td>
<td>$N_1\left(\frac{h}{2}\right)$</td>
<td>$N_2(h)$</td>
<td>$N_3\left(\frac{h}{2}\right)$</td>
<td>$N_3(h)$</td>
</tr>
<tr>
<td>8:</td>
<td>$N_1(h)$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:</td>
<td>$N_3\left(\frac{h}{2}\right)$</td>
<td>$N_3(h)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10:</td>
<td>$N_2(h)$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

It is conservatively assumed that the true result is accurate at least to within the agreement of the bottom two results in the diagonal, in this case, to within $|N_3(h) - N_4(h)|$.

Each column beyond the first in the extrapolation table is obtained by a simple averaging process, so the technique can produce high-order approximations with minimal computational cost. However, as $k$ increases, the round-off error in $N_1(h/2^k)$ will generally increase because the instability of numerical differentiation is related to the step size $h/2^k$. Also, the higher-order formulas depend increasingly on the entry to their immediate left in the table, which is the reason we recommend comparing the final diagonal entries to ensure accuracy.

The technique of extrapolation is used throughout the course. The most prominent applications occur in approximating integrals and for determining approximate solutions to differential equations.
4.3 Elements of Numerical Integration

In this section, we introduce methods for evaluating the definite integral of a function that has no explicit antiderivative or whose antiderivative is not easy to obtain. The basic method involved is called numerical quadrature. It uses a sum \( \sum_{i=0}^{n} a_i f(x_i) \) to approximate the integral \( \int_{a}^{b} f(x) \, dx \).

The methods of quadrature are based on the interpolation polynomials. The basic idea is to select a set of distinct nodes \( \{x_0, x_1, \ldots, x_n\} \) from the interval \([a, b]\). Then, integrate the Lagrange interpolating polynomial

\[
P_n(x) = \sum_{i=0}^{n} f(x_i) L_i(x)
\]

and its truncation error term over \([a, b]\) to obtain

\[
\int_{a}^{b} f(x) \, dx = \int_{a}^{b} \sum_{i=0}^{n} f(x_i) L_i(x) \, dx + \int_{a}^{b} \frac{n}{(n+1)!} f^{(n+1)}(\xi(x)) \, dx
\]

where \( \xi(x) \) is in \([a, b]\) for each \( x \) and

\[
a_i = \int_{a}^{b} L_i(x) \, dx, \quad \text{for each } i = 0, 1, \ldots, n.
\]

The quadrature formula is, therefore,

\[
\int_{a}^{b} f(x) \, dx \approx \sum_{i=0}^{n} a_i f(x_i),
\]

with error given by

\[
E(f) = \frac{1}{(n+1)!} \int_{a}^{b} \prod_{i=0}^{n} (x-x_i)f^{(n+1)}(\xi(x)) \, dx.
\]

We first consider formulas produced by using first and second Lagrange polynomials with equally-spaced nodes. This gives the Trapezoidal rule and Simpson’s rule.

**Trapezoidal rule**

Let \( x_0 = a, x_1 = b, h = b-a \). Denote the linear Lagrange polynomial

\[
P_1(x) = \frac{(x-x_1)}{(x_0-x_1)} f(x_0) + \frac{(x-x_0)}{(x_1-x_0)} f(x_1).
\]

Then,

\[
\int_{a}^{b} f(x) \, dx = \int_{x_0}^{x_1} \left[ \frac{(x-x_1)}{(x_0-x_1)} f(x_0) + \frac{(x-x_0)}{(x_1-x_0)} f(x_1) \right] \, dx + \frac{1}{2} \int_{x_0}^{x_1} f''(\xi(x))(x-x_0)(x-x_1) \, dx (4.16)
\]

The product \((x-x_0)(x-x_1)\) does not change sign on \([x_0, x_1]\), so the Weighted Mean Value Theorem for Integrals (Theorem [1.1.21]) can be applied to the error term to give, for some \( \xi \in (x_0, x_1) \),

\[
\int_{x_0}^{x_1} f''(\xi(x))(x-x_0)(x-x_1) \, dx = f''(\xi) \int_{x_0}^{x_1} (x-x_0)(x-x_1) \, dx = -\frac{h^3}{6} f''(\xi).
\]
Consequently, (4.16) becomes

\[
\int_a^b f(x) \, dx = \left[ \frac{(x - x_1)^2}{2(x_0 - x_1)} f(x_0) + \frac{(x - x_0)^2}{2(x_1 - x_0)} f(x_1) \right]_{x_0}^{x_1} - \frac{h^3}{12} f''(\xi)
\]

\[
= \frac{(x_1 - x_0)}{2} [f(x_0) + f(x_1)] - \frac{h^3}{12} f''(\xi).
\]

Using the notation \( h = x_1 - x_0 \) gives the Trapezoidal rule:

\[
\int_a^b f(x) \, dx = \frac{h}{2} [f(x_0) + f(x_1)] - \frac{h^3}{12} f''(\xi).
\]

When \( f \) is a positive function, \( \int_a^b f(x) \, dx \) is approximated by the area in a trapezoid, as shown in the following figure. The error term for the Trapezoidal rule involves \( f'' \) and the rule gives the exact result when applied to any function whose second derivative is identically zero, that is, any polynomial of degree one or less.

**SIMPSON’S RULE**

Simpson’s rule results from integrating over \([a, b]\) the second Lagrange polynomial with equally-spaced nodes \( x_0 = a, x_2 = b, \) and \( x_1 = a + h, \) where \( h = (b - a)/2.\)
Using the similar technique, one can derive and obtain the following Simpson’s rule:

\[ \int_{a}^{b} f(x) \, dx = \frac{h}{3}[f(x_0) + 4f(x_1) + f(x_2)] - \frac{h^5}{90}f^{(4)}(\xi). \]

The error term in Simpson’s rule involves the fourth derivative of \( f \), so it gives exact results when applied to any polynomial of degree three or less.

**Example 4.3.1.** Compare the Trapezoidal rule and Simpson’s rule approximations to \( \int_{0}^{2} f(x) \, dx \) when \( f(x) \) is

(a) \( x^2 \);  (b) \( x^4 \);  (c) \( (x + 1)^{-1} \);
(d) \( \sqrt{1 + x^2} \);  (e) \( \sin x \);  (f) \( e^x \).

**Solution.** On \([0, 2]\) the Trapezoidal and Simpson’s rule have the forms:

\[
\text{Trapezoid: } \int_{0}^{2} f(x) \, dx \approx f(0) + f(2), \quad \text{and} \\
\text{Simpson’s: } \int_{0}^{2} f(x) \, dx \approx \frac{1}{3}[f(0) + 4f(1) + f(2)].
\]

When \( f(x) = x^2 \), they give

\[
\text{Trapezoid: } \int_{0}^{2} f(x) \, dx \approx 0^2 + 2^2 = 4, \quad \text{and} \\
\text{Simpson’s: } \int_{0}^{2} f(x) \, dx \approx \frac{1}{3}[0^2 + 4 \cdot 1^2 + 2^2] = \frac{8}{3}.
\]

The approximation from Simpson’s rule is exact because its truncation error involves \( f^{(4)} \), which is identically 0 when \( f(x) = x^2 \). The results to three places for the functions are summarized in the following table. Notice that in each instance Simpson’s rule is significantly superior.

<table>
<thead>
<tr>
<th></th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
<th>(f)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact value</td>
<td>2.667</td>
<td>6.400</td>
<td>1.099</td>
<td>2.958</td>
<td>1.416</td>
<td>6.389</td>
</tr>
<tr>
<td>Trapezoidal</td>
<td>4.000</td>
<td>16.000</td>
<td>1.333</td>
<td>3.326</td>
<td>0.909</td>
<td>8.389</td>
</tr>
<tr>
<td>Simpson’s</td>
<td>2.667</td>
<td>6.667</td>
<td>1.111</td>
<td>2.964</td>
<td>1.425</td>
<td>6.421</td>
</tr>
</tbody>
</table>

The standard derivation of quadrature error formulas is based on determining the class of polynomials for which these formulas produce exact results. The next definition is used to facilitate the discussion of this derivation.

**Definition 4.3.2.** The **degree of accuracy**, or **precision**, of a quadrature formula is the largest positive integer \( n \) such that the formula is exact for \( x^k \), for each \( k = 0, 1, \ldots, n \).

The Trapezoidal and Simpson’s rules have degrees of precision one and three, respectively, by definition.

The Trapezoidal and Simpson’s rules are examples of a class of methods known as **Newton-Cotes formulas**. There are two types of Newton-Cotes formulas, open and closed.
Closed Newton-Cotes Formulas

The $n + 1$-point closed Newton-Cotes formula uses nodes $x_i = x_0 + ih$, for $i = 0, 1, \cdots, n$, where $x_0 = a$, $x_n = b$, and $h = (b - a)/n$. It is called closed because the endpoints of the closed interval $[a, b]$ are included as nodes. See the following figure for illustration.

![Figure 4.3: Graphical illustration for closed Newton-Cotes formula.](image)

The formula assumes the form

$$\int_a^b f(x) \, dx \approx \sum_{i=0}^n a_i f(x_i), \quad \text{where} \quad a_i = \int_a^b L_i(x) \, dx = \int_a^b \prod_{j=0, j\neq i}^n \frac{(x - x_j)}{(x_i - x_j)} \, dx.$$ 

The error analysis associated with the closed Newton-Cotes formulas can be summarized in the following theorem.

**Theorem 4.3.3.** Suppose that $\sum_{i=0}^n a_i f(x_i)$ denotes the $n + 1$-point closed Newton-Cotes formula with $x_0 = a$, $x_n = b$, and $h = (b - a)/n$. There exists $\xi \in (a, b)$ for which

$$\int_a^b f(x) \, dx = \sum_{i=0}^n a_i f(x_i) + \frac{h^{n+3} f^{(n+2)}(\xi)}{(n+2)!} \int_0^1 t^2(t-1)\cdots(t-n) \, dt,$$

if $n$ is even and $f \in C^{n+2}[a, b]$, and

$$\int_a^b f(x) \, dx = \sum_{i=0}^n a_i f(x_i) + \frac{h^{n+2} f^{(n+1)}(\xi)}{(n+1)!} \int_0^1 t(t-1)\cdots(t-n) \, dt,$$

if $n$ is odd and $f \in C^{n+1}[a, b]$.

Note that when $n$ is an even number the degree of precision is $n + 1$, although the interpolation polynomial is of degree at most $n$. When $n$ is odd, the degree of precision is only $n$. We remark that when $n = 1$, the Newton-Cotes formula is the Trapezoidal rule and when $n = 2$, the formula becomes the Simpson’s rule.

Open Newton-Cotes Formulas

The open Newton-Cotes formulas do not include the endpoints of $[a, b]$ as nodes. They use the nodes $x_i = x_0 + ih$, for each $i = 0, 1, \cdots, n$, where $h = (b - a)/(n + 2)$ and $x_0 = a + h$. This implies that $x_n = b - h$, so we label the endpoints by setting $x_{-1} = a$ and $x_{n+1} = b$.

**Theorem 4.3.4.** Suppose that $\sum_{i=0}^n a_i f(x_i)$ denotes the $n + 1$-point open Newton-Cotes formula with $x_{-1} = a$, $x_{n+1} = b$, and $h = (b - a)/(n + 2)$. There exists $\xi \in (a, b)$ for which

$$\int_a^b f(x) \, dx = \sum_{i=0}^n a_i f(x_i) + \frac{h^{n+3} f^{(n+2)}(\xi)}{(n+2)!} \int_0^1 t^2(t-1)\cdots(t-n) \, dt,$$
if \( n \) is even and \( f \in C^{n+2}[a,b] \), and
\[
\int_a^b f(x) \, dx = \sum_{i=0}^{n} a_i f(x_i) + \frac{h^{n+2} f^{(n+1)}(\xi)}{(n+1)!} \int_0^n t(t-1) \cdots (t-n) \, dt,
\]
if \( n \) is odd and \( f \in C^{n+1}[a,b] \).

As in the case of the closed methods, we have the degree of precision comparatively higher for the even methods than for the odd methods. Some of the common open Newton-Cotes formulas with their error terms are as follows:

\( n = 0 \): Midpoint rule
\[
\int_{x_1}^{x_{-1}} f(x) \, dx = 2h f(x_0) + \frac{h^3}{3} f'(\xi), \quad \text{where} \quad x_{-1} < \xi < x_1.
\]

\( n = 1 \):
\[
\int_{x_1}^{x_{-1}} f(x) \, dx = \frac{3h}{2} [f(x_0) + f(x_1)] + \frac{3h^3}{4} f''(\xi), \quad \text{where} \quad x_{-1} < \xi < x_2.
\]

### 4.4 Composite Numerical Integration

The Newton-Cotes formulas are generally unsuitable for use over large integration intervals. High-degree formulas would be required, and the values of the coefficients in these formulas are difficult to obtain. In this section, we discuss a piecewise approach to numerical integration that uses the low-order Newton-Cotes formulas. These are the techniques most often applied.

We choose an even integer \( n \). Subdivide the interval \([a,b]\) into \( n \) subintervals, and apply Simpson’s rule on each consecutive pair of subintervals. See Figure 4.4. With \( h = (b - a)/n \) and

\[ x_j = a + jh, \text{ for each } j = 0, 1, \cdots, n, \text{ we have} \]
\[
\int_a^b f(x) \, dx = \sum_{j=1}^{n/2} \int_{x_{2j-2}}^{x_{2j}} f(x) \, dx
\]
\[
= \sum_{j=1}^{n/2} \left\{ \frac{h}{3} [f(x_{2j-2}) + 4f(x_{2j-1}) + f(x_{2j})] - \frac{h^5}{90} f^{(4)}(\xi_j) \right\},
\]

Figure 4.4: Graphical illustration for composite Simpson’s rule.
for some \( \xi_j \in (x_{2j-2}, x_{2j}) \), provided that \( f \in C^4[a, b] \). Using the fact that for each \( j = 1, 2, \ldots, (n/2) - 1 \) we have \( f(x_{2j}) \) appearing in the term corresponding to the interval \([x_{2j-2}, x_{2j}]\) and also in the term corresponding to the interval \([x_{2j}, x_{2j+2}]\), we can reduce this sum to

\[
\int_a^b f(x) \, dx = \frac{h}{3} \left[ f(x_0) + 2 \sum_{j=1}^{(n/2)-1} f(x_{2j}) + 4 \sum_{j=1}^{n/2} f(x_{2j-1}) + f(x_n) \right] - \frac{h^5}{90} \sum_{j=1}^{n/2} f^{(4)}(\xi_j).
\]

The error associated with this approximation is

\[
E(f) = -\frac{h^5}{90} \sum_{j=1}^{n/2} f^{(4)}(\xi_j),
\]

where \( x_{2j-2} < \xi_j < x_{2j} \), for each \( j = 1, 2, \ldots, n/2 \). If \( f \in C^4[a, b] \), the Extreme Value Theorem implies that \( f^{(4)} \) attains its maximum and minimum in \([a, b]\). Since

\[
\min_{x \in [a, b]} f^{(4)}(x) \leq f^{(4)}(\xi_j) \leq \max_{x \in [a, b]} f^{(4)}(x),
\]

we have

\[
\frac{n}{2} \min_{x \in [a, b]} f^{(4)}(x) \leq \frac{n}{2} \sum_{j=1}^{n/2} f^{(4)}(\xi_j) \leq \frac{n}{2} \max_{x \in [a, b]} f^{(4)}(x)
\]

and

\[
\min_{x \in [a, b]} f^{(4)}(x) \leq \frac{2}{n} \sum_{j=1}^{n/2} f^{(4)}(\xi_j) \leq \max_{x \in [a, b]} f^{(4)}(x).
\]

By the Intermediate Value Theorem, there is a constant \( \mu \in (a, b) \) such that

\[
f^{(4)}(\mu) = \frac{2}{n} \sum_{j=1}^{n/2} f^{(4)}(\xi_j).
\]

Thus,

\[
E(f) = -\frac{h^5}{90} \left( \frac{n}{2} f^{(4)}(\mu) \right) = -\frac{h^5}{180} n f^{(4)}(\mu)
\]

or since \( h = (b - a)/n \), we have

\[
E(f) = -\frac{(b - a)}{180} h^4 f^{(4)}(\mu).
\]

These observations produce the following result.

**Theorem 4.4.1.** Let \( f \in C^4[a, b] \), \( n \) be even, \( h = (b - a)/n \), and \( x_j = a + jh \), for each \( j = 0, 1, \ldots, n \). There exists a constant \( \mu \in (a, b) \) for which the **Composite Simpson’s rule** for \( n \) subintervals can be written with its error term as

\[
\int_a^b f(x) \, dx = \frac{h}{3} \left[ f(x_0) + 2 \sum_{j=1}^{(n/2)-1} f(x_{2j}) + 4 \sum_{j=1}^{n/2} f(x_{2j-1}) + f(x_n) \right] - \frac{(b-a)}{180} h^4 f^{(4)}(\mu).
\]

Note that the error term for Composite Simpson’s rule is \( O(h^4) \), whereas it was \( O(h^5) \) for the standard Simpson’s rule. However, these rates are not comparable because for standard Simpson’s rule we have \( h \) fixed at \( (b - a)/2 \), but for Composite Simpson’s rule we have \( h = (b - a)/n \), for \( n \) an even integer. This permits us to considerably reduce the value of \( h \) when the Composite Simpson’s rule is used.

The subdivision approach can be applied to any of the Newton-Cotes formulas. We summarize the result for the Trapezoidal and Midpoint rules.
Theorem 4.4.2. Let $f \in C^2[a,b]$, $h = (b-a)/n$, and $x_j = a + jh$, for each $j = 0, 1, \ldots, n$. There exists a constant $\mu \in (a,b)$ for which the Composite Trapezoidal rule for $n$ subintervals can be written with its error term as

$$
\int_a^b f(x) \, dx = \frac{h}{2} \left[ f(a) + 2 \sum_{j=1}^{n-1} f(x_j) + f(b) \right] - \frac{(b-a)}{12} h^2 f''(\mu).
$$

Theorem 4.4.3. Let $f \in C^2[a,b]$, $n$ be even, $h = (b-a)/(n+2)$, and $x_j = a + (j+1)h$, for each $j = -1, 0, 1, \ldots, n + 1$. There exists a constant $\mu \in (a,b)$ for which the Composite Midpoint rule for $n + 2$ subintervals can be written with its error term as

$$
\int_a^b f(x) \, dx = 2h \sum_{j=1}^{n/2} f(x_{2j}) + \frac{(b-a)}{6} h^2 f''(\mu).
$$

Example 4.4.4. Determine values of $h$ that will ensure an approximation error of less than 0.000002 when approximating $\int_0^\pi \sin x \, dx$ and employing (a) Composite Trapezoidal rule and (b) Composite Simpson’s rule.

(a) The error form for Composite Trapezoidal rule for $f(x) = \sin x$ on $[0, \pi]$ is

$$
\left| \frac{\pi h^2}{12} f''(\mu) \right| = \left| \frac{\pi h^2}{12} (-\sin \mu) \right| \leq \frac{\pi h^2}{12} < 2 \times 10^{-5}
$$

to ensure sufficient accuracy with this technique. Since $h = \pi/n$ implies that $n = \pi/h$, we need

$$
\frac{\pi^3}{12n^2} < 2 \times 10^{-5} \implies n > \left( \frac{\pi^3}{12(2 \times 10^{-5})} \right)^{1/2} \approx 359.44
$$

and the Composite Trapezoidal rule requires $n \geq 360$.

(b) The error form for the Composite Simpson’s rule for $f(x) = \sin x$ on $[0, \pi]$ is

$$
\left| \frac{\pi h^4}{180} f^{(4)}(\mu) \right| = \left| \frac{\pi h^4}{180} \sin \mu \right| \leq \frac{\pi h^4}{180} < 2 \times 10^{-5}
$$

to ensure sufficient accuracy with this technique. Using again the fact that $n = \pi/h$ we have

$$
\frac{\pi^5}{180n^4} < 2 \times 10^{-5} \implies n > \left( \frac{\pi^5}{180(2 \times 10^{-5})} \right)^{1/4} \approx 17.07.
$$

Hence, the Composite Simpson’s rule requires only $n \geq 18$. Moreover, the Composite Simpson’s rule with $n = 18$ gives

$$
\int_0^\pi \sin x \, dx \approx \frac{\pi}{54} \left[ 2 \sum_{j=1}^{8} \sin \left( \frac{j\pi}{9} \right) + 4 \sum_{j=1}^{9} \sin \left( \frac{(2j-1)\pi}{18} \right) \right] = 2.0000104.
$$

This is accurate to within about $10^{-5}$ because the true value of integral is 2.

Composite Simpson’s rule is the clear choice if you wish to minimize computation. For comparison purposes, consider the Composite Trapezoidal rule using $h = \pi/18$ for the integral in the example above. This approximation uses the same function evaluations as Composite Simpson’s rule but the approximation in this case

$$
\int_0^\pi \sin x \, dx \approx \frac{\pi}{36} \left[ 2 \sum_{j=1}^{17} \sin \left( \frac{j\pi}{18} \right) + \sin 0 + \sin \pi \right] = 1.9949205
$$

is accurate only to about $5 \times 10^{-3}$. 
4.7. GAUSSIAN QUADRATURE

Round-Off error stability

In Example 4.4.4 we saw that ensuring an accuracy of $2 \times 10^{-5}$ for approximating the integral required 360 subdivisions of $[0, \pi]$ for the Composite Trapezoidal rule and only 18 for the Composite Simpson’s rule. In addition to the fact that less computation is needed for the Simpson’s technique, you might suspect that because of fewer computations this method would also involve less round-off error. However, an important property shared by all the composite integration techniques is a stability with respect to round-off error. That is, the round-off error does not depend on the number of calculations performed.

To demonstrate this fact, suppose we apply the Composite Simpson’s rule with $n$ subintervals to a function $f$ on $[a, b]$ and determine the maximum bound for the round-off error. Assume that $f(x_i)$ is approximated by $\tilde{f}(x_i)$ and that

$$f(x_i) = \tilde{f}(x_i) + e_i, \quad \text{for each } i = 0, 1, \ldots, n,$$

where $e_i$ denotes the round-off error associated with using $\tilde{f}(x_i)$ to approximate $f(x_i)$. Then the accumulated error, $e(h)$, in the Composite Simpson’s rule is

$$e(h) = \frac{h}{3} \left[ e_0 + 2 \sum_{j=1}^{(n/2)-1} e_{2j} + 4 \sum_{j=1}^{n/2} e_{2j-1} + e_n \right]$$

If the round-off errors are uniformly bounded by $\varepsilon$, then

$$e(h) \leq \frac{h}{3} \left[ |e_0| + 2 \sum_{j=1}^{(n/2)-1} |e_{2j}| + 4 \sum_{j=1}^{n/2} |e_{2j-1}| + |e_n| \right].$$

But $nh = b - a$, so we have

$$e(h) \leq (b - a)\varepsilon,$$

a bound independent of $h$ and $n$. This means that, even though we may need to divide an interval into more parts to ensure accuracy, the increased computation that is required does not increase the round-off error. This result implies that the procedure is stable as $h$ approaches zero. Recall that this was not true of the numerical differentiation procedures.

4.7 Gaussian Quadrature

The Newton-Cotes formulas were derived by integrating interpolating polynomials. The error term in the interpolating polynomial of degree $n$ involves the $n+1$-st derivative of the function being approximated, so a Newton-Cotes formula is exact when approximating the integral of any polynomial of degree less than or equal to $n$.

All the Newton-Cotes formulas use values of the function at equally-spaced points. This restriction is convenient when the formulas are combined to form the composite rules, but it can significantly decrease the accuracy of the approximation.

Gaussian quadrature chooses the points for evaluation in an optimal, rather than equally-spaced way. The nodes $x_0, x_1, \ldots, x_n$ in the interval $[a, b]$ and coefficients $c_0, c_1, \ldots, c_n$, are chosen to minimize the expected error obtained in the approximation

$$\int_a^b f(x) \, dx \approx \sum_{i=0}^n c_i f(x_i).$$
To measure the degree of accuracy, we assume that the best choice of these values produces the exact result for the largest class of polynomials, that is, the choice that gives the greatest degree of precision.

The coefficients \( c_0, c_1, \ldots, c_n \) in the formula are arbitrary, and the nodes \( x_0, x_1, \ldots, x_n \) are restricted only by the fact that they must lie in \([a, b]\), the interval of integration. This gives us \( 2n + 2 \) parameters to choose. If the coefficients of a polynomial are considered parameters, the class of polynomials of degree at most \( 2n + 1 \) also contains \( 2n + 2 \) parameters. This is the largest class of polynomials for which it is reasonable to expect a formula to be exact.

We show how to select the coefficients and nodes when \( n = 1 \) and the interval of integration is \([-1, 1]\). We will discuss the more general situation later. Suppose that we want to determine \( c_0, c_1, x_0, \) and \( x_1 \) so that the integration formula

\[
\int_{-1}^{1} f(x) \, dx \approx c_0 f(x_0) + c_1 f(x_1)
\]

gives the exact result whenever \( f(x) \) is a polynomial of degree 2(1) + 1 = 3 or less, that is, when \( f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \), for some collection of constants \( a_0, a_1, a_2, \) and \( a_3 \). Since we have

\[
\int (a_0 + a_1 x + a_2 x^2 + a_3 x^3) \, dx = a_0 \int 1 \, dx + a_1 \int x \, dx + a_2 \int x^2 \, dx + a_3 \int x^3 \, dx,
\]

this is equivalent to showing that the formula gives exact results when \( f(x) \) is 1, \( x \), \( x^2 \), and \( x^3 \). Hence, we need \( c_0, c_1, x_0, \) and \( x_1 \) so that

\[
c_0 \cdot 1 + c_1 \cdot 1 = \int_{-1}^{1} 1 \, dx = 2, \quad c_0 x_0 + c_1 x_1 = \int_{-1}^{1} x \, dx = 0,
\]

\[
c_0 x_0^2 + c_1 x_1^2 = \int_{-1}^{1} x^2 \, dx = \frac{2}{3} \quad \text{and} \quad c_0 x_0^3 + c_1 x_1^3 = \int_{-1}^{1} x^3 \, dx = 0.
\]

It shows that this system of equations has the unique solution

\[
c_0 = 1, \quad c_1 = 1, \quad x_0 = -\frac{\sqrt{3}}{3}, \quad \text{and} \quad x_1 = \frac{\sqrt{3}}{3},
\]

which gives the approximation formula

\[
\int_{-1}^{1} f(x) \, dx \approx f \left( -\frac{\sqrt{3}}{3} \right) + f \left( \frac{\sqrt{3}}{3} \right).
\]

This formula has degree of accuracy 3, that is, it produces the exact result for every polynomial of degree 3 or less.

**Legendre Polynomials**

The technique we have described could be used to determine the nodes and coefficients for formulas that give exact results for higher-degree polynomials. The set that is relevant to our problem is the **Legendre polynomials**, a collection of polynomials \( \{ P_0(x), P_1(x), \ldots, P_n(x), \ldots \} \) with properties:

1. For each \( n \), \( P_n(x) \) is a monic polynomial of degree \( n \) (i.e., the coefficient of highest order term is 1).
2. \( \int_{-1}^{1} P(x)P_{n}(x) \, dx = 0 \) whenever \( P(x) \) is a polynomial of degree less than \( n \).

3. We have \( P_{n}(x) = \frac{n!}{(2n)!} \frac{d^n}{dx^n} [(x^2 - 1)^n] \) for \( n \geq 1 \) and \( P_0(x) = 1 \).

The first few Legendre polynomials are

\[
P_0(x) = 1, \quad P_1(x) = x, \quad P_2(x) = x^2 - \frac{1}{3}, \quad \text{and} \quad P_3(x) = x^3 - \frac{3}{5}x.
\]

The roots of these polynomials are distinct, lie in the interval \((-1, 1)\), have a symmetry with respect to the origin, and, most importantly, are the correct choice for determining the parameters that give us the nodes and coefficient for our quadrature method.

The nodes \( x_0, x_1, \ldots, x_n \) needed to produce an integral approximation formula that gives exact results for any polynomial of degree less than \( 2n + 2 \) are the roots of the \( n + 1 \)-st degree Legendre polynomial. This is established by the following result.

**Theorem 4.7.1.** Suppose that \( x_0, x_1, \ldots, x_n \) are the roots of the \( n + 1 \)-st Legendre polynomial \( P_{n+1}(x) \) and that for each \( i = 0, 1, \ldots, n \), the numbers \( c_i \) are defined by

\[
c_i = \int_{-1}^{1} \prod_{j=0, j \neq i}^{n} \frac{x-x_j}{x_i-x_j} \, dx.
\]

If \( P(x) \) is any polynomial of degree less than \( 2n + 2 \), then

\[
\int_{-1}^{1} P(x) \, dx = \sum_{i=0}^{n} c_i P(x_i).
\]

That is, the degree of accuracy of the quadrature formula \( \int_{-1}^{1} f(x) \, dx \approx \sum_{i=1}^{n} c_i f(x_i) \) is \( 2n + 1 \).

**Proof.** Let us first consider the situation for a polynomial \( P(x) \) of degree less than \( n + 1 \). Rewrite \( P(x) \) in terms of \( n \)-th Lagrange coefficient polynomials with nodes at the roots of \( n + 1 \)-st Legendre polynomial \( P_{n+1}(x) \). The error term for this representation involves the \( n + 1 \)-st derivative of \( P(x) \). Since \( P(x) \) is of degree less than \( n + 1 \), the \( n + 1 \)-st derivative of \( P(x) \) is 0, and this representation is of exact. So

\[
P(x) = \sum_{i=0}^{n} P(x_i)L_{i}^{(n)}(x) = \sum_{i=1}^{n} \prod_{j=1, j \neq i}^{n} \frac{x-x_j}{x_i-x_j} P(x_i)
\]

and

\[
\int_{-1}^{1} P(x) \, dx = \int_{-1}^{1} \left[ \sum_{i=0}^{n} \prod_{j=1, j \neq i}^{n} \frac{x-x_j}{x_i-x_j} P(x_i) \right] \, dx
\]

\[
= \sum_{i=0}^{n} \left[ \int_{-1}^{1} \prod_{j=1, j \neq i}^{n} \frac{x-x_j}{x_i-x_j} \, dx \right] P(x_i) = \sum_{i=0}^{n} c_i P(x_i).
\]

Hence, the result is true for polynomials of degree less than \( n + 1 \).

Consider a polynomial \( P(x) \) of degree at least \( n + 1 \) but less than \( 2n + 2 \). Divide \( P(x) \) by the \( n + 1 \)-st Legendre polynomial \( P_{n+1}(x) \). This gives two polynomials \( Q(x) \) and \( R(x) \), each of degree less than \( n + 1 \) with

\[
P(x) = Q(x)P_{n+1}(x) + R(x).
\]
Note that \( x_i \) is a root of \( P_{n+1}(x) \) for each \( i = 0, 1, \ldots, n \), so we have

\[
P(x_i) = Q(x_i)P_{n+1}(x_i) + R(x_i) = R(x_i).
\]

We now invoke the unique power of the Legendre polynomials. First, the degree of the polynomial \( Q(x) \) is less than \( n + 1 \), so by Legendre property (2),

\[
\int_{-1}^{1} Q(x)P_{n+1}(x) = 0.
\]

Then, since \( R(x) \) is a polynomial of degree less than \( n + 1 \), the opening argument implies that

\[
\int_{-1}^{1} R(x) \, dx = \sum_{i=0}^{n} c_i R(x_i).
\]

Putting these facts together verifies that the formula is exact for the polynomial \( P(x) \):

\[
\int_{-1}^{1} P(x) \, dx = \int_{-1}^{1} [Q(x)P_{n+1}(x) + R(x)] \, dx = \int_{-1}^{1} R(x) \, dx = \sum_{i=0}^{n} c_i R(x_i) = \sum_{i=0}^{n} c_i P(x_i).
\]

This completes the proof.

The constants \( c_i \) needed for the quadrature rule can be generated from the equation in Theorem 4.7.1, but both these constants and the roots of the Legendre polynomials are extensively tabulated. The quadrature formula formed by these special nodes (called Gaussian nodes) and weights are called Gaussian quadrature. The following table lists these values for \( n = 2, 3, 4, \) and 5.

<table>
<thead>
<tr>
<th>( n )</th>
<th>Root ( r_{n,i} )</th>
<th>Coefficients ( c_{n,i} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.5773502692</td>
<td>1.0000000000</td>
</tr>
<tr>
<td></td>
<td>-0.5773502692</td>
<td>1.0000000000</td>
</tr>
<tr>
<td>3</td>
<td>0.7745966692</td>
<td>0.5555555556</td>
</tr>
<tr>
<td></td>
<td>0.0000000000</td>
<td>0.8888888889</td>
</tr>
<tr>
<td></td>
<td>-0.7745966692</td>
<td>0.5555555556</td>
</tr>
<tr>
<td>4</td>
<td>0.8611363116</td>
<td>0.3478548451</td>
</tr>
<tr>
<td></td>
<td>0.3399810436</td>
<td>0.6521451549</td>
</tr>
<tr>
<td></td>
<td>-0.3399810436</td>
<td>0.6521451549</td>
</tr>
<tr>
<td></td>
<td>-0.8611363116</td>
<td>0.3478548451</td>
</tr>
<tr>
<td>5</td>
<td>0.9061798459</td>
<td>0.2369268850</td>
</tr>
<tr>
<td></td>
<td>0.5384693101</td>
<td>0.4786286705</td>
</tr>
<tr>
<td></td>
<td>0.0000000000</td>
<td>0.5688888889</td>
</tr>
<tr>
<td></td>
<td>-0.5384693101</td>
<td>0.4786286705</td>
</tr>
<tr>
<td></td>
<td>-0.9061798459</td>
<td>0.2369268850</td>
</tr>
</tbody>
</table>

**Example 4.7.2.** Approximate \( \int_{-1}^{1} e^x \cos x \, dx \) using Gaussian quadrature with \( n = 3 \).

**Solution.** Let \( f(x) = e^x \cos x \). The entries in the table above give us

\[
\int_{-1}^{1} e^x \cos x \, dx \approx 0.5f(0.7745966692) + 0.8f(0) + 0.5f(-0.7745966692) = 1.9333904.
\]

Integration by parts can be used to show that the true value of the integral is 1.9334214, so the absolute error is less than \( 3.2 \times 10^{-5} \).
An integral \( \int_a^b f(x) \, dx \) over an arbitrary \([a, b]\) can be transformed into an integral over \([-1, 1]\) by using the change of variables:

\[
t = \frac{2x - a - b}{b - a} \implies x = \frac{1}{2} \left[ (b - a)t + a + b \right].
\]

This permits Gaussian quadrature to be applied to any interval \([a, b]\) because

\[
\int_a^b f(x) \, dx = \int_{-1}^{1} f \left( \frac{(b - a)t + a + b}{2} \right) \frac{(b - a)}{2} \, dt. \tag{4.17}
\]

**Example 4.7.3.** Consider the integral \( \int_1^3 x^6 - x^2 \sin(2x) \, dx = 317.3442466 \).

(a) Compare the results for the closed Newton-Cotes formula with \( n = 1 \), the open Newton-Cotes formula with \( n = 1 \), and Gaussian Quadrature when \( n = 2 \).

(b) Compare the results for the closed Newton-Cotes formula with \( n = 2 \), the open Newton-Cotes formula with \( n = 2 \), and Gaussian Quadrature when \( n = 3 \).

**Solution.**

(a) Each of the formula in this part requires 2 evaluations of the function \( f(x) = x^6 - x^2 \sin(2x) \). The Newton-Cotes approximations are

\[
\text{Closed } n = 1 : \quad \frac{2}{2} [f(1) + f(3)] = 731.6054420; \\
\text{Open } n = 1 : \quad \frac{3(2/3)}{2} [f(5/3) + f(7/3)] = 188.7856682.
\]

Gaussian quadrature applied to this problem requires that the integral first be transformed into a problem whose interval of integration is \([-1, 1]\). Using (4.17), gives

\[
\int_1^3 x^6 - x^2 \sin(2x) \, dx = \int_{-1}^{1} (t + 2)^6 - (t + 2)^2 \sin(2(t + 2)) \, dt.
\]

Let \( F(t) = (t + 2)^6 - (t + 2)^2 \sin(2t + 4) \). Gaussian quadrature with \( n = 2 \) then gives

\[
\int_1^3 x^6 - x^2 \sin(2x) \, dx \approx F(-0.5773502692) + F(0.5773502692) = 306.8199344;
\]

(b) Each of the formula in this part requires 3 evaluations. The Newton-Cotes approximations are

\[
\text{Closed } n = 2 : \quad \frac{1}{3} [f(1) + 4f(2) + f(3)] = 333.2380940; \\
\text{Open } n = 2 : \quad \frac{4(1/2)}{3} [2f(1.5) - f(2) + 2f(2.5)] = 303.5912023.
\]

Gaussian quadrature with \( n = 3 \), once the transformation has been done, gives

\[
\int_1^3 x^6 - x^2 \sin(2x) \, dx \approx 317.2641516.
\]

The Gaussian quadrature results are clearly superior in each instance.