Numerical integration (also known as quadrature)

To numerically integrate a function $f(x)$ on an interval $[a,b]$, one usually uses approximations of the form

$$\int_a^b f(x)\mathrm{d}x \quad \approx \quad w_0 f(x_0) + \cdots + w_n f(x_n) = \sum_{i=0}^{n} w_i f(x_i),$$

where the points x_i and the weights w_i are chosen appropriately.

Comment. Such **quadrature rules** are typically judged by the maximal degree d of polynomials that they can integrate without error. For instance, to correctly integrate constant functions (degree 0 polynomials), the weights need to be such that they add up to $b-a$. (Why?!)

Common quadrature rules include:

- Newton–Cotes rules: equally spaced points x_i

  These are most useful if $f(x)$ already computed at equally spaced points, or if evaluation is fast.

  There are closed Newton–Cotes rules and open ones. Open means that a and b are not part of the x_i.

- Gaussian quadrature: the x_i are not equally spaced but chosen carefully

  Choosing the x_i is similar to our discussion of Chebyshev nodes in polynomial interpolation.

  Gaussian quadrature is particularly useful if $f(x)$ is expensive to compute.

Comment. In the case of integrable singularities, such as in $\int_0^1 \frac{1}{\sqrt{x}}\mathrm{d}x$, we cannot use closed Newton–Cotes.

Because of time constraints, we will focus on the simplest example of a closed Newton–Cotes rule, namely the trapezoidal rule.

We will then see that combining this with Richardson extrapolation, we can obtain higher order Newton–Cotes rules such as Simpson's rule.

The (composite) trapezoidal rule

Given equally spaced nodes $x_0, x_1, \ldots, x_n$ with $x_0 = a$ and $x_n = b$, we interpolate $f(x)$ on each segment $[x_{i-1}, x_i]$ by a linear function. Writing $h = (b-a)/n$ for the distance between nodes, the resulting integration rule is the following:

(trapezoidal rule) The following is an approximation of order 2:

$$\int_a^b f(x)\mathrm{d}x \approx \frac{h}{2}[f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)]$$

Why? On each segment $[x_{i-1}, x_i]$, we approximate $f(x)$ by a linear function so that the integral on that segment becomes the area of a trapezoid and we get

$$\int_{x_{i-1}}^{x_i} f(x)\mathrm{d}x \approx \underbrace{\text{width} \cdot \text{average height}}_{\text{area}} = h \cdot \frac{f(x_{i-1}) + f(x_i)}{2} = \frac{h}{2} f(x_{i-1}) + \frac{h}{2} f(x_i).$$

Make a sketch! Adding together the integrals over all segments, each node (except x_0 and x_n) will show up twice (hence the factor of 2 in front of $f(x_1), \ldots, f(x_{n-1})$) and we get the claimed integration rule. The fact that the trapezoidal rule provides an approximation of order 2 is proved in Theorem 123 below.

Sanity check. Note that the weights are $\frac{h}{2}$ for the first and last node, and h for the others. The sum of the weights is $2 \cdot \frac{h}{2} + (n-1) \cdot h = nh = b - a$. That is what we need to integrate constant functions without error. Indeed, from the construction it is clear that the composite trapezoidal rule integrates linear functions exactly.

Armin Straub
straub@southalabama.edu

Theorem 123. (trapezoidal rule with error term) If f is C^2 smooth, then

$$\int_a^b f(x)\mathrm{d}x = \frac{h}{2}[f(x_0)+2f(x_1)+\cdots+2f(x_{n-1})+f(x_n)] - \frac{(b-a)}{12}f''(\xi)h^2$$

for some $\xi \in [a,b]$. In particular, the trapezoidal rule is of order 2.

Proof. On each segment $[x_{i-1}, x_i]$, the error of the interpolation is

$$f(x) = \text{linear approximation} + \frac{1}{2}f''(\xi)(x-x_{i-1})(x-x_i).$$

Hence, when integrating

$$\int_{x_{i-1}}^{x_i} f(x)\mathrm{d}x = \underbrace{\frac{h}{2}f(x_{i-1})+\frac{h}{2}f(x_i)}_{\text{integral of linear approx.}} + \underbrace{\int_{x_{i-1}}^{x_i}\frac{1}{2}f''(\xi)(x-x_{i-1})(x-x_i)\mathrm{d}x}_{\text{error}_i},$$

so that the error when integrating is

$$\text{error}_i = \frac{1}{2}f''(\psi)\underbrace{\int_{x_{i-1}}^{x_i}(x-x_{i-1})(x-x_i)\mathrm{d}x}_{=\int_0^h x(x-h)\mathrm{d}x=\left[\frac{1}{3}x^3-\frac{h}{2}x^2\right]_0^h=-\frac{1}{6}h^3} = -\frac{1}{12}f''(\psi)h^3$$

where ψ is some value between x_{i-1} and x_i. Let us briefly justify the "pulling out" of $f''(\xi)$ even though ξ depends on x. Note that $(x-x_{i-1})(x-x_i)$ is always $\leqslant 0$ in the integral and, therefore, does not change sign. This means that the error integral lies between the corresponding integrals where we replace $f''(\xi)$ with its maximum value M and minimum value L; the values M and L no longer depend on x and therefore can be pulled out of the integral. The above computation then shows that error_i is between $-\frac{1}{12}h^3M$ and $-\frac{1}{12}h^3L$, hence must be equal to $-\frac{1}{12}h^3m$ for some $m \in [L,M]$. Since L and M are the minimum and maximum value of f'' on $[x_{i-1},x_i]$, and since f'' is continuous, it follows that $m = f''(\psi)$ for some ψ.

To get the overall error, we need to add the errors $-\frac{1}{12}f''(\psi_i)h^3$ from each segment $[x_{i-1},x_i]$, where $i=1,2,\ldots,n$ and where $\psi_i \in [x_{i-1},x_i]$. The result is

$$-\frac{1}{12}f''(\psi_1)h^3+\cdots+-\frac{1}{12}f''(\psi_n)h^3 = -\frac{nh}{12}\underbrace{\frac{f''(\psi_1)+\cdots+f''(\psi_n)}{n}}_{=\text{average}=f''(\xi)}h^2 = -\frac{b-a}{12}f''(\xi)h^2,$$

where ξ is some value between a and b. □

Comment. A closer inspection of our proof shows that the $f''(\xi)$ in the error formula converges, as $h \to 0$, to the average value of f'' on $[a,b]$. This means that we have a way to obtain an **error estimate** (rather than only an error bound). This observation is also useful because it shows that the error is of a form that allows us to perform Richardson extrapolation.

Advanced comment. Indeed, using the Euler–Maclaurin one can show that the error is

$$-\frac{f'(b)-f'(a)}{12}h^2+\frac{f'''(b)-f'''(a)}{720}h^4+\ldots-B_{2m}\frac{f^{(2m-1)}(b)-f^{(2m-1)}(a)}{(2m)!}h^{2m}+O(h^6),$$

where the B_{2m} are rational numbers known as Bernoulli numbers (provided, of course, that f is C^{2m-1} smooth). The fact that only even powers of h show up reflects the fact that the trapezoidal rule is symmetric (and therefore correctly integrates $(x-c)^n$ where $c=(a+b)/2$ and n is odd).

Note that this more precise form of the error tells us that the Richardson extrapolation of the trapezoidal rule will be of order 4 (rather than order 3).

Example 124. Use the trapezoidal rule to approximate $\int_1^3 \frac{1}{x}\,\mathrm{d}x = \log(3) \approx 1.09861$.

(a) Use $h=1$ and $h=1/2$.

(b) Using Richardson extrapolation, combine the previous two approximations to obtain an approximation of higher order. What are absolute and relative error?

Comment. We will see in the next section that this is equivalent to using Simpson's rule!

Solution. Let us write $f(x)=\frac{1}{x}$.

(a) $\int_1^3 f(x)\,\mathrm{d}x \approx \frac{h}{2}[f(1)+2f(2)+f(3)] = \frac{1}{2}\left[1+2\cdot\frac{1}{2}+\frac{1}{3}\right] = \frac{7}{6} \approx 1.1667$

Comment. Make a sketch! Can you explain why our approximation (for any h) will be an underestimate of the true value of the integral?

(b) $\int_1^3 f(x)\,\mathrm{d}x \approx \frac{h}{2}\left[f(1)+2f\left(\frac{3}{2}\right)+2f(2)+2f\left(\frac{5}{2}\right)+f(3)\right] = \frac{1}{4}\left[1+2\cdot\frac{2}{3}+2\cdot\frac{1}{2}+2\cdot\frac{2}{5}+\frac{1}{3}\right]$
$= \frac{67}{60} \approx 1.1167$

(c) Let us write $A(h)$ and $A(h/2)$ for our two approximations, and A^* for the true value of the integral. Since $A(h)$ is an approximation of order 2, we expect $A(h) \approx A^* + Ch^2$ for some constant C.

Correspondingly, $A\left(\frac{h}{2}\right) \approx A^* + \frac{1}{4}Ch^2$. Hence, $4A\left(\frac{h}{2}\right) - A(h) \approx (4-1)A^* = 3A^*$.

Therefore, the Richardson extrapolation is $\frac{1}{3}\left[4A\left(\frac{h}{2}\right) - A(h)\right] = \frac{1}{3}\left[4\cdot\frac{67}{60} - \frac{7}{6}\right] = \frac{11}{10} = 1.1$.

The absolute error is $|1.1 - \log(3)| \approx 0.00139$ and the relative error is $\left|\frac{1.1-\log(3)}{\log(3)}\right| \approx 0.00126$.

Example 125. `Python` Let us redo Example 124 by implementing the trapezoidal rule.

```
>>> def trapezoidal_rule(f, a, b, n):
        h = (b - a) / n
        integral = (f(a) + f(b)) / 2
        for i in range(1,n):
            integral += f(a + i*h)
        return h*integral

>>> def f(x):
        return 1/x
```

Comment. Writing `x += y` is a useful and common short alternative to `x = x + y`.

Choosing `n` to be 2 and 4 is equivalent to $h = \frac{3-1}{2} = 1$ and $h = \frac{3-1}{4} = \frac{1}{2}$ and so we get the same values as in Example 124:

```
>>> trapezoidal_rule(f, 1, 3, 2)

   1.1666666666666665

>>> trapezoidal_rule(f, 1, 3, 4)

   1.1166666666666667
```

As expected, further increasing n produces better approximations:

```
>>> [trapezoidal_rule(f, 1, 3, 10**n) for n in range(1,4)]

    [1.1015623265623264, 1.0986419169811203, 1.0986125849642736]
```

Indeed, the following convincingly illustrates that the error in the trapezoidal rule is $O(h^2)$.

```
>>> from math import log

>>> [trapezoidal_rule(f, 1, 3, 10**n) - log(3) for n in range(1,6)]

    [0.0029500378942166616, 2.9628313010565677e-05, 2.962961638264261e-07,
    2.9629636522088276e-09, 2.962430301067798e-11]
```

However, note that our computer had to work pretty hard to get the final approximation, because that entailed computing about 10^5 values. We clearly should use a higher order method in order to compute to higher accuracy. One option is to do what we did in the last part of Example 124.

Simpson's rule

Let us spell out what happens in general when we proceed as in the last part of Example 124. We start with $T(h)$, the trapezoidal rule applied with step size h, which is given by

$$T(h) = \frac{h}{2}[f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)].$$

Then, since $T(h)$ is an approximation of order $N = 2$, the Richardson extrapolation

$$S(h) := \frac{2^N T(h) - T(2h)}{2^N - 1} = \frac{4}{3}T(h) - \frac{1}{3}T(2h)$$

is an approximation of higher order, known as **Simpson's rule**. It takes the following form:

(Simpson's rule) Suppose n is even. The following is an approximation of order 4:

$$\int_a^b f(x)\mathrm{d}x \approx \frac{h}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)]$$

Proof. To see this, note that the trapezoidal approximations $T(h)$ and $T(2h)$ are given by

$$\begin{aligned} T(h) &= \frac{h}{2}[f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)], \\ T(2h) &= h[f(x_0) + 2f(x_2) + \cdots + 2f(x_{n-2}) + f(x_n)]. \end{aligned}$$

Here, we need n to be even so that $T(2h)$ uses the points x_0, x_2, ..., x_{n-2}, x_n. Therefore, the Richardson extrapolation is

$$\begin{aligned} S(h) = \frac{4}{3}T(h) - \frac{1}{3}T(2h) &= \frac{h}{3}[2f(x_0) + 4f(x_1) + \cdots + 4f(x_{n-1}) + 2f(x_n)] \\ &\quad -\frac{h}{3}[f(x_0) + 2f(x_2) + \cdots + 2f(x_{n-2}) + f(x_n)] \\ &= \frac{h}{3}[2f(x_0) + 4f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + 2f(x_n)], \end{aligned}$$

where the right-hand side is Simpson's rule. That this is an approximation of order 4 follows because the error of $T(h)$ is an even function in h, so that the order of $S(h)$ increases to 4 (rather than the otherwise expected 3). □

Alternative approach. We can also proceed like for the trapezoidal rule, except that we use quadratic instead of linear interpolations on each segment. More precisely, starting with equally spaced nodes $x_0, x_1, \ldots, x_n$ with n even, we can interpolate $f(x)$ on each segment $[x_{i-1}, x_{i+1}]$, with i odd, by a quadratic function $p(x)$. The integral on that segment works out to be

$$\int_{x_{i-1}}^{x_{i+1}} f(x)\mathrm{d}x \approx \int_{x_{i-1}}^{x_{i+1}} p(x)\mathrm{d}x \overset{\text{work}}{=} \frac{h}{3}[f(x_{i-1}) + 4f(x_i) + f(x_{i+1})],$$

where we wrote $h = (b-a)/n$ for the distance between nodes. Adding together the integrals over all segments, each node x_i, with i odd, will show up once with the above factor of $4h/3$ while x_i, with i even, (except x_0 and x_n) will show up twice with a factor of $h/3$. We thus again get Simpson's integration rule.

Comment. The above rule is often called **Simpson's 1/3 rule**. There is also **Simpson's 3/8 rule** which is derived similarly but is based on a cubic (instead of a quadratic) interpolation; it thus requires an additional node (the resulting error term is of the same order but about half).

Similar to Theorem 123, one can show that the error in Simpson's rule is as follows:

Theorem 126. (Simpson's rule with error term) If f is C^4 smooth, then

$$\int_a^b f(x)\mathrm{d}x = \frac{h}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + \cdots + 4f(x_{n-1}) + f(x_n)] - \frac{(b-a)}{180} f^{(4)}(\xi) h^4$$

for some $\xi \in [a, b]$.

Comment. In the alternative approach above, we constructed Simpson's rule so that quadratic polynomials would be integrated without error; the error term tells us that, in fact, cubic polynomials are also integrated without error. In other words, the error term is a pleasant surprise!

Example 127. Use Simpson's rule with $h = \frac{1}{2}$ to approximate $\displaystyle\int_1^3 \frac{1}{x}\,\mathrm{d}x = \log(3) \approx 1.09861$.

Solution.

$$\int_1^3 f(x)\,\mathrm{d}x \approx \frac{h}{3}\left[f(1) + 4f\left(\frac{3}{2}\right) + 2f(2) + 4f\left(\frac{5}{2}\right) + f(3)\right] = \frac{1}{6}\left[1 + 4\cdot\frac{2}{3} + 2\cdot\frac{1}{2} + 4\cdot\frac{2}{5} + \frac{1}{3}\right] = \frac{11}{10} = 1.1$$

Comment. Note that $\frac{11}{10}$ is exactly what we obtained in the last part of Example 124.

Indeed, recall that Simpson's rule with $h = \frac{1}{2}$ ($n = 4$) is equivalent to applying Richardson extrapolation to the trapezoidal approximations with $h = \frac{1}{2}$ ($n = 4$) and $h = 1$ ($n = 2$).

Example 128. Python Let us compute $\int_1^3 \frac{1}{x}\,dx = \log(3) \approx 1.09861$ by implementing Simpson's rule. The following code assumes that n is even.

```
>>> def simpson_rule(f, a, b, n):
        h = (b - a) / n
        integral = f(a) + f(b)
        for i in range(1,n):
            if i%2 == 1:
                integral += 4*f(a + i*h)
            else:
                integral += 2*f(a + i*h)
        return h/3*integral

>>> def f(x):
        return 1/x
```

With n set to 4, we obtain $\frac{11}{10}$ as in Examples 124 and 127:

```
>>> simpson_rule(f, 1, 3, 4)

    1.0999999999999999
```

Our approximations (here, $n = 10$ and 100) quickly approach the true value:

```
>>> [simpson_rule(f, 1, 3, 10**n) for n in range(1,3)]

    [1.0986605986605984, 1.0986122939305363]
```

Indeed, the following convincingly illustrates that the error in Simpson's rule is $O(h^4)$.

```
>>> from math import log

>>> [simpson_rule(f, 1, 3, 10**n) - log(3) for n in range(1,6)]

    [4.830999248861545e-05, 5.262426494567762e-09, 5.282441151166495e-13, -
    1.5543122344752192e-15, -7.105427357601002e-15]
```

Example 129. Python Various integration methods are already implemented in scipy.

```
>>> from scipy import integrate
```

For instance, the following is a way to use Simpson's rule with $n = 4$ (so that 5 points are used). The result matches the $\frac{11}{10}$ that we computed ourselves.

```
>>> def f(x):
        return 1/x

>>> xvalues = [1+1/2*i for i in range(5)]

>>> yvalues = [f(x) for x in xvalues]

>>> integrate.simps(yvalues, xvalues)

    1.0999999999999999
```

On the other hand, the following is a convenient way of "general purpose integration", where we only need to specify the end points:

```
>>> integrate.quad(f, 1, 3)

    (1.0986122886681096, 7.555511459798467e-14)
```

The second part of the result is an estimate for the absolute error.