Chapter 10

Numerical Integration

Outline of Section

- The Trapezoidal Rule
- Simpson's Rule
- Romberg integration
- Relationship to ODE methods
- Improper Integrals

10.1 Introduction

One of the most commonly-encountered problems in everyday physics research is to efficiently evaluate a definite integral

$$I = \int_a^b f(x)\mathrm{d}x. \tag{10.1}$$

Numerical integration is an entire subfield of numerical analysis, with numerous sophisticated methods and codes available. These basically all fall into three main categories: Monte Carlo, quadrature and ODE methods. We have already seen Monte Carlo integration in Chapter 6. In this Chapter, we will deal predominantly with Newton-Coates quadrature methods, touching on ODE methods towards the end. We will also see how to transform badly-behaved integrals into a form that is more conducive to numerical evaluation.

10.2 Quadrature Methods

The basic idea of quadrature methods is to divide the integral into a number of sub-regions, and integrate over them independently. In a single dimension, this boils down to approximating the integral as a number of rectangle-like sub-regions, as shown in Fig. 10.1 – basically a fancy Riemann sum. The reason we say 'fancy' here is that, unlike in a Riemann sum, the tops of the rectangles are generally not taken to be flat, but rather slanted or with an even more complicated shape, in order to better approximate the behaviour of the integrand between samples.

There are thus three main decisions to be made in designing a quadrature algorithm:

- The number of samples to take of the integrand $f(x)$
- The distribution of the samples over the integration domain, i.e. the stepsize between samples h, and its variation with x
- The interpolating function to assume at the tops of the rectangle-like shapes.

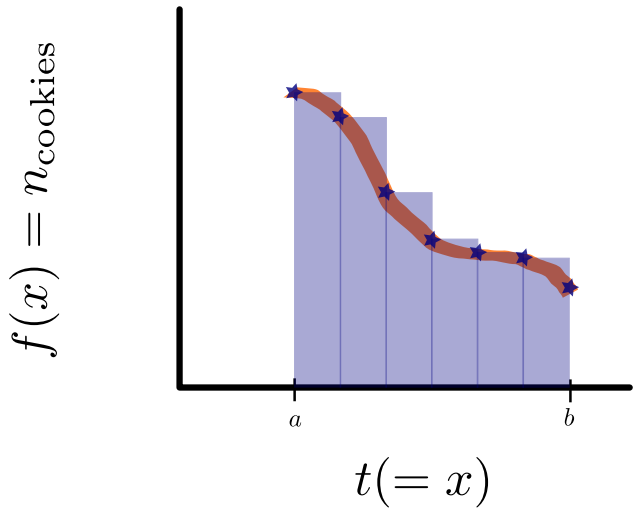

Figure 10.1: Basic anatomy of numerical integration by quadrature: the integral is divided up into rectangle-like shapes, the integrand is sampled at their edges, and the final integral is obtained by assuming some interpolating function passing between the sampled points. Cookies for dramatic effect only.

The simplest methods in this class are the *Newton-Coates rules*, which use a constant stepsize h; quadrature methods with a variable h across the integration domain are referred to as *Gaussian quadrature*. The example shown in Fig. 10.1 is therefore in fact a Newton-Coates method. Newton-Coates rules take a number of equal-spaced samples of the integrand, and then estimate the overall integral using a weighted sum of the samples. The different weightings correspond to different interpolating functions across the tops of the rectangles. The next three subsections deal with three different Newton-Coates methods; the difference between them is essentially in the choice of interpolating function.

10.2.1 The Trapezoidal Rule

Apart from a basic Riemann sum, the simplest way to approximate the integrand between samples is interpolate between the linearly. This is the so-called *Trapezoidal Rule*. The Trapezoidal Rule is a *two-point* rule, in that it estimates the integral in a region between $x = x_i$ and $x = x_{i+1}$ using the value of the integrand at only two points, namely the two integration limits:

$$\int_{x_i}^{x_{i+1}} f(x)\,\mathrm{d}x = h\left[\frac{1}{2}f(x_i) + \frac{1}{2}f(x_{i+1})\right] + O\left(h^3\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\right). \tag{10.2}$$

To compute an entire integral however, we need to patch together many instances of the Trapezoidal Rule into the *Extended Trapezoidal Rule*

$$\int_{x_0}^{x_{n-1}} f(x)\,\mathrm{d}x = h\left[\frac{1}{2}f(x_0) + f(x_1) + f(x_2) + \ldots + f(x_{n-2}) + \frac{1}{2}f(x_{n-1})\right] \tag{10.3}$$

This rule provides an estimate of the integral using n samples of the integrand, all connected via linear interpolation according to the Trapezoidal Rule. Turning this into a useful integration algorithm is then just a matter of wrapping it in a loop that steadily increases the number of samples until some convergence criterion is reached.

So, the basic algorithm for computing a definite integral using the Trapezoidal Rule to some desired relative accuracy ϵ is

(a) Evaluate $f(a)$ and $f(b)$

(b) Use these as a first estimate $I_1 = h_1 \frac{1}{2}\left[f(a) + f(b)\right]$

(c) Evaluate the midpoint $f(\frac{a+b}{2})$

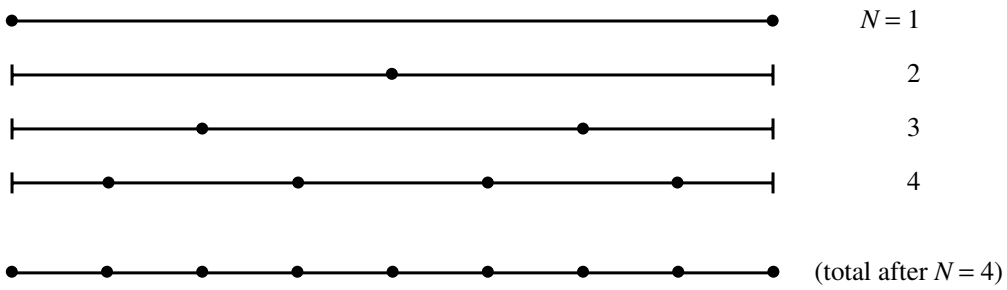

Figure 10.2: Sampling strategy for successive iterations of the Extended Trapezoidal Rule. Each new step fills in the midpoints between the previous points, never evaluating the integrand at the same value of x twice.

(d) Use this to update your estimate $I_2 = h_2 \left[\frac{1}{2}f(a) + f(\frac{a+b}{2}) + \frac{1}{2}f(b)\right]$

(e) If $\left|\frac{I_2 - I_1}{I_1}\right| < \epsilon$, terminate (convergence has been reached).

(f) If not, keep filling in intermediate points and making more trapezoids until the convergence criterion is satisfied.

Note in particular the use of the word 'intermediate' – there is no reason to ever evaluate the integrand twice at the same value of x. The idea is to decrease h by a factor of 2 at each iteration (Fig. 10.2), so that each successive iteration of the Extended Trapezoidal Rule can be built up from the previous one by simply reweighting the old estimate to account for the change in h, and adding in the new samples at $x = x_1, x_3, x_5, \ldots, x_{n-2}$ (remembering that our samples are indexed from 0 to $n-1$, not 1 to n):

$$T_{j+1} = \frac{1}{2}T_j + h \sum_{i=1}^{(n-1)/2} f(x_{2i-1}). \tag{10.4}$$

10.2.2 Simpson's Rule

The next level of sophistication is to add in an additional point to the basic Newton-Coates rule, and build an extended rule from this instead. The rule goes by the name of *Simpson's Rule*, and looks like

$$\int_{x_i}^{x_{i+2}} f(x)\,\mathrm{d}x = h\left[\frac{1}{3}f(x_i) + \frac{4}{3}f(x_{i+1}) + \frac{1}{3}f(x_{i+2})\right] + O\left(h^5 \frac{\mathrm{d}^4 f}{\mathrm{d}x^4}\right). \tag{10.5}$$

We can see immediately that this three-point rule is no longer doing linear interpolation. In fact, with the additional point, we can now uniquely define a polynomial of one degree higher, i.e. a quadratic. Simpson's Rule therefore improves on the Trapezoidal Rule by approximating the tops of the equal-width sub-integrals with quadratic curves rather than straight lines.

The corresponding *Extended Simpson's Rule* can then be build up in the same manner as for the Extended Trapezoidal Rule, by patching together successive copies of the basic 3-point Simpson's Rule:

$$\begin{aligned}\int_{x_0}^{x_{n-1}} f(x)\,\mathrm{d}x = h\Bigg[&\frac{1}{3}f(x_0) + \frac{4}{3}f(x_1) + \frac{2}{3}f(x_2) + \frac{4}{3}f(x_3)\ldots \\ &\ldots + \frac{4}{3}f(x_{n-4}) + \frac{2}{3}f(x_{n-3}) + \frac{4}{3}f(x_{n-2}) + \frac{1}{3}f(x_{n-1})\Bigg]\end{aligned} \tag{10.6}$$

Note in particular that the 3-point sub-rules here do not overlap at all, and are simply patched together end-to-end.

The implementation of the Extended Simpson's Rule in a full integration algorithm follows that for the Trapezoidal Rule fairly closely, except in two important ways. The first is pretty obvious – one needs to start with 3 points in the initial step, at the two limits of integration and the

midpoint. The second is more subtle, and interesting. Simpson's Rule is specifically designed so that the higher-order interpolant results in a higher-order method, i.e. so that the error terms of order h^3 and h^4 from the trapezoidal rule cancel. This means that it can be achieved by taking two successive iterations of the Trapezoidal Rule and combining them with the appropriate weights to cancel the lower-order errors induced by the two lower-order steps:

$$S_j = \frac{4}{3}T_{j+1} - \frac{1}{3}T_j. \tag{10.7}$$

This is quite cute, as it means that an implementation of the Extended Simpson's Rule can be easily piggybacked onto an existing implementation of the Extended Trapezoidal Rule. This gives a more accurate result, more or less 'for free' in terms of computational time.

10.2.3 Romberg integration

Unsurprisingly, successive iterations of Simpson's Rule can also be combined in such a way as to cause the $\mathcal{O}(h^5)$ and even higher-order errors to cancel. The tradeoff of this sort of exercise though is that the interpolating function across the tops of the sub-integrals becomes steadily more non-local the higher order one goes to. Much as higher-order spline interpolation starts to become a bit dicey due to non-local effects causing substantial higher derivatives to produce large excursions and 'ringing', higher-order Newton-Coates schemes can also start to suffer stability problems with rapidly-varying integrands. This isn't catastrophic, but it does offset any real speed gains that one can achieve from them, as it can mean that they need smaller h than one might naively expect, to avoid non-local effects.

Romberg integration is an algorithm that extrapolates the Newton-Coates strategy to arbitrarily high-order accuracy, at the same time as extrapolating the stepsize h to zero. This sounds almost too good to be true, and it basically is: except for extremely smooth functions, Romberg integration doesn't usually provide much speedup compared to a simple Simpson's Rule integrator – and it adds quite a lot more complexity. For moderately-varying integrands, variable-h methods such as ODE integration usually provide more significant speedup than Romberg integration. These methods are covered in the following Section.

10.3 Relationship to ODE methods

Doing a definite integral is mathematically equivalent to solving the initial value problem (IVP)

$$\frac{\mathrm{d}y}{\mathrm{d}x} = f(x); \quad y(a) = 0 \tag{10.8}$$

for $x = b$, i.e. $I \equiv y(b)$. We can see this by

$$\begin{aligned} I \equiv \int_a^b f(x)\,\mathrm{d}x &= \int_a^b \frac{\mathrm{d}y}{\mathrm{d}x}\,\mathrm{d}x \\ &= \int_{y(a)}^{y(b)} \mathrm{d}y \\ &= y(b) - y(a) \end{aligned}$$

Here the choice of the initial value $y(a)$ is entirely arbitrary, as we care only about its derivative. We can therefore choose any constant $y(a) = C \implies I = y(b) - C$. For simplicity, we can therefore just choose $C = 0$, giving $I = y(b)$.

To solve the integral, we therefore need to solve the ODE that describes the evolution of $y(x)$, starting from from $x = a$ and finishing at $x = b$. We have seen in the previous Chapter that there are a number of reasonable methods for this. RK45 with an adaptive stepsize is one of the most robust and efficient.

So what happens if we just directly use RK45 for doing definite integrals by recasting them as ODEs? Well, we end up with a nice adaptive numerical integration method – but we can actually do a bit better than that. Recall the general form for the RK4 step:

$$k_1 = hf(x_n, y_n) \tag{10.9}$$
$$k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \tag{10.10}$$
$$k_3 = hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \tag{10.11}$$
$$k_4 = hf(x_n + h, y_n + k_3) \tag{10.12}$$
$$y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4. \tag{10.13}$$

Now notice that in the case where the ODE comes from a transformed definite integral, f does not depend on y. Therefore, $k_2 = k_3$, and we only need to evaluate one of them. We can therefore collapse the equations into a reduced RK4 step:

$$k_1 = hf(x_n) \tag{10.14}$$
$$k_2 = hf(x_n + \frac{h}{2}) \tag{10.15}$$
$$k_3 = hf(x_n + h) \tag{10.16}$$
$$y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{2}{3}k_2 + \frac{1}{6}k_3. \tag{10.17}$$

This is another 25% more efficient than the regular RK45 step when solving definite integrals.

Hang on though – this last expression looks eerily familiar. In fact, it is *exactly* Simpson's rule with $h \to \frac{h}{2}$:

$$\int_{x_n}^{x_{n+1}} f(x)\,\mathrm{d}x = \frac{h}{2}\left[\frac{1}{3}f(x_n) + \frac{4}{3}f(x_{n+\frac{1}{2}}) + \frac{1}{3}f(x_{n+1})\right] + O(h^5) \tag{10.18}$$

In the end this is not so surprising when we think about the fact that RK4 and Simpson's rule are both designed from the start to cancel local errors of order h^4. Indeed,

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} \frac{\mathrm{d}y}{\mathrm{d}x}(x)\,\mathrm{d}x = y(x_n) + \int_{x_n}^{x_{n+1}} f(x)\,\mathrm{d}x, \tag{10.19}$$

so RK4 is just Simpson's Rule generalised to a variable stepsize and an $f(x)$ that also depends on $y(x) \equiv \int f(x)\,\mathrm{d}x$.

10.4 Improper Integrals

Everything we have seen so far in this Chapter corresponds to a *closed* Newton-Coates rule (or its variable-stepsize generalisation, in the case of the RK45 implementation of the generalised Simpson's method). This means that the integrand is evaluated directly at the edge of every sub-interval. However, it is also possible to construct *open* rules, which evaluate the integral in an interval without directly evaluating it at the limits. One such rule is the *midpoint rule*,

$$\int_{x_0}^{x_1} f(x)\,\mathrm{d}x = h\left[f\left(\frac{x_0 + x_1}{2}\right)\right], \tag{10.20}$$

which simply does a central Reimann sum, approximating the integrand in the interval by its value at the midpoint of the interval.

Open rules can be particularly useful when it is difficult or impossible to evaluate the integrand at one of the limits of integration. This may happen if the integrand is undefined at the limit, e.g. $\frac{0}{0}$, or if the integral diverges at the limit.

Notice that the midpoint rule only covers a small subdomain of integration, just like the 2-point Trapezoidal and 3-point Simpson's Rules introduced earlier. To use it for doing a real integration, we need to patch it together with many other basic rules to make an extended rule, and put it inside an iterative algorithm that increases the number of subdomains until we can get a convergent result. However, there is nothing that forces us to patch together only rules of exactly the same kind when creating our extended rules. For dealing with an integrand that is well-behaved right up to the limit of integration, but undefined exactly on the limit, a good approach is therefore simply to patch a single copy of the midpoint rule (at the limit where the integrand is undefined) on to many copies of Simpson's rule (for use in the interior and at the other limit).

In the case where an integrand diverges as one approaches one of the limits, or where one end of the integral extends to infinity, a little more care is needed.

Take the case of integrating to infinity first. Here, we have an integral

$$I = \int_a^b f(x)\mathrm{d}x. \tag{10.21}$$

where $a = -\infty$ or $b = \infty$. The best thing to do is to transform the asymptotic part of the integral via the transformation $x \to \frac{1}{t}$, which gives

$$\int_a^b f(x)\,\mathrm{d}x = \int_{1/b}^{1/a} \frac{1}{t^2} f\left(\frac{1}{t}\right)\,\mathrm{d}t, \tag{10.22}$$

i.e. making the previously badly-behaved limit correspond to $t = 0$. The integrand still can't be evaluated at this limit, but the asymptotic behaviour has been compressed into a finite range, allowing us to employ an open rule on the transformed integral, and simply avoid evaluating the integrand at exactly $t = 0$. Note that this trick only works for one limit at a time, and only when a and b have the same sign; otherwise, you will need to split the integral at some opportune location (often at $x = 0$) and do the two parts separately. This trick is also inefficient if the entire integrand is not asymptotically decreasing at least as quickly as x^{-2}; in this case it also pays to split the integral, so as to isolate the part where it *is* dropping faster than x^{-2}, and use the transformation only on that part (and a regular Simpson's Rule on the rest).

The case of a divergent integral requires even more thought. In this case, we have an integrable singularity at some special value of x; call this x_s. If we know the value of x_s, we can proceed fairly safely: split the integral at the singularity. Now you have two integrals each with singularities at the edges of their domains. We can then transform the nasty parts of each integral as $x \to \alpha$, with

$$\alpha = t^{\frac{1}{1-\gamma}} + x_s \quad \text{for lower limit } x_s \tag{10.23}$$

$$\alpha = x_s - t^{\frac{1}{1-\gamma}} \quad \text{for upper limit } x_s. \tag{10.24}$$

This gives (with either $a' = x_s$ or $b' = x_s$)

$$\int_{a'}^{b'} f(x)\,\mathrm{d}x = \frac{1}{1-\gamma} \int_0^{(b'-a')^{1-\gamma}} t^{\frac{\gamma}{1-\gamma}} f(\alpha)\,\mathrm{d}t, \tag{10.25}$$

which we can happily integrate with any extended rule that is open at $t = 0$. Here γ is a power-law index that we are basically free to choose to be anything between 0 and 1. Note that the most efficient integration will result if we choose γ to match the slope of our divergence as closely as possible. That is, if our function behaves like $f(x) \to (x - x_s)^{-\beta}$ as $x \to x_s$, then the most efficient choice is $\gamma = \beta$.

If you run into a situation where you know that there is a singularity somewhere in $a < x_s < b$, but don't actually know the value of x_s, then you're in significantly more trouble. In this case, you need to try to somehow 'sneak past' the singularity using variable-stepsize methods such as ODE integration, Gaussain quadrature or Monte Carlo methods. The challenge of course is that the area around $x = x_s$ generally contributes significantly to the total integral, so you *also* need to sample this area fairly densely to get a converged result – but at the same time, you need to

avoid evaluating the integrand so close to the pole that the result overflows the floating-point type you're using. It's not a lot of fun. In most cases, it's worth putting some time into instead trying to transform your problem so that the singularity either goes away, or sits at some known value of some suitably transformed integration variable.