LAB2: NUMERICAL INTEGRATION AND ITS APPLICATIONS
DUE: SUNDAY, OCTOBER 25

Most integrals cannot be evaluated exactly. For such integrals, we instead have to rely on numerical integration, also called quadrature[1], to calculate an approximate value for the integral. In practice, numerical integration is often used even for the integrals that can be evaluated exactly. Numerical integration methods are simple compared to exact methods, and can be efficient and highly accurate. They can also be used to derive simple numerical methods for solving ordinary differential equations.

1. Introduction

In a numerical integration scheme the integral is replaced by a finite sum:

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} f(x_i)w_i. \tag{1}$$

Here $w_0, w_2, \dots w_n$ are real numbers, called *quadrature weights*. The sets of *quadrature points* points $\{x_i\}_{i=0}^n$ in $[a,b]$ and weights $\{w_i\}_{i=0}^n$ together form a *quadrature rule*. While an exact evaluation of the left hand side (1) would require knowing the antiderivative of f, the right hand side only requires that the values of f are known at the quadrature points.

Note that if we have a quadrature rule for one interval $[a,b]$, then we can easily adapt it to any other interval $[\tilde{a},\tilde{b}]$ by transformation of the variables. The mapping $\phi : x \mapsto \frac{b-x}{b-a}\tilde{a} + \frac{x-a}{b-a}\tilde{b}$ maps the interval $[a,b]$ onto $[\tilde{a},\tilde{b}]$, hence

$$\begin{aligned}\int_{\tilde{a}}^{\tilde{b}} f(\tilde{x})\,d\tilde{x} &= \int_a^b f\circ\phi(x)\phi'(x)\,dx \\ &= \int_a^b f\left(\frac{b-x}{b-a}\tilde{a} + \frac{x-a}{b-a}\tilde{b}\right)\left(\frac{\tilde{b}-\tilde{a}}{b-a}\right)dx \\ &\approx \sum_{i=0}^{n} f\left(\frac{b-x_i}{b-a}\tilde{a} + \frac{x_i-a}{b-a}\tilde{b}\right)\left(\frac{\tilde{b}-\tilde{a}}{b-a}\right)w_i.\end{aligned} \tag{2}$$

This shows that a quadrature rule for the interval $[\tilde{a},\tilde{b}]$ with points $\{\tilde{x}_i\}_{i=0}^n$ and weights $\{\tilde{w}_i\}_{i=0}^n$ is obtained by setting

$$\begin{aligned}\tilde{x}_i &= \frac{b-x_i}{b-a}\tilde{a} + \frac{x_i-a}{b-a}\tilde{b}, \\ \tilde{w}_i &= \frac{\tilde{b}-\tilde{a}}{b-a}w_i.\end{aligned}$$

[1]Quadrature is a historical term for determining the area under a curve. Some authors reserve the word quadrature to mean numerical integration in one dimension exclusively, using the word cubature for integration in two dimensions. However, in modern usage, quadrature is usually synonymous with numerical integration in any dimension.

That is, the the quadrature points in $[\tilde{a}, \tilde{b}]$ are distributed with the same relative distances as those in $[a, b]$, and the weights are simply scaled by the ratio of the interval lengths.

2. Simple quadrature rules

Polynomials are good for approximating smooth functions, and can easily be integrated exactly. A simple quadrature rule is based on the approximation

$$\int_a^b f(x)\,dx \approx \int_a^b p(x)\,dx \tag{3}$$

where p is a polynomial of degree n interpolating f in the points $x_0, \ldots, x_n$. That is,

$$p(x_i) = f(x_i), \quad i = 0, \ldots, n. \tag{4}$$

Recall that a polynomial of degree n is precisely determined by its values at any $n+1$ distinct points, so the approximation in (3) is well-defined.

The *Lagrange polynomials* with respect to the points $x_0, \ldots, x_n$ can be defined setting

$$L_i(x) = \prod_{j \neq i}^{n} \frac{x - x_j}{x_i - x_j},$$

with the convention that the empty product equals 1. These polynomials have degree n and have roots at x_j for $j \neq i$, and takes the value 1 at x_i. This important property can be written compactly as

$$L_i(x_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j, \end{cases} \qquad 0 \leq i, j \leq n.$$

In particular, the polynomial p in (3)–(4) can be expanded Lagrange polynomials,

$$p(x) = \sum_{j=0}^{n} f(x_j) L_j(x),$$

which is easily verified from $p(x_i) = f(x_i)$. We then obtain the quadrature weights w_i as the integral of the Lagrange polynomial L_i:

$$\int_a^b f(x)\,dx \approx \int_a^b p(x)\,dx = \sum_{i=0}^{n} f(x_i) \underbrace{\int_a^b L_i(x)\,dx}_{=w_i} = \sum_{i=0}^{n} f(x_i) w_i. \tag{5}$$

We consider some examples of quadrature rules with 1, 2, and 3 quadrature points in the interval $[a, b]$. See also Figure 1.

Midpoint rule: A polynomial of degree zero is constant, and is determined by its value at one point. If we take this point to be the midpoint of the interval $[a, b]$, we get the midpoint rule. In (5) we have $L_i(x) = 1$, so find that the weight is equal to the interval length.

$$\begin{aligned} &\text{points:} && x_0 = \frac{a+b}{2} \\ &\text{weights:} && w_0 = b - a. \end{aligned}$$

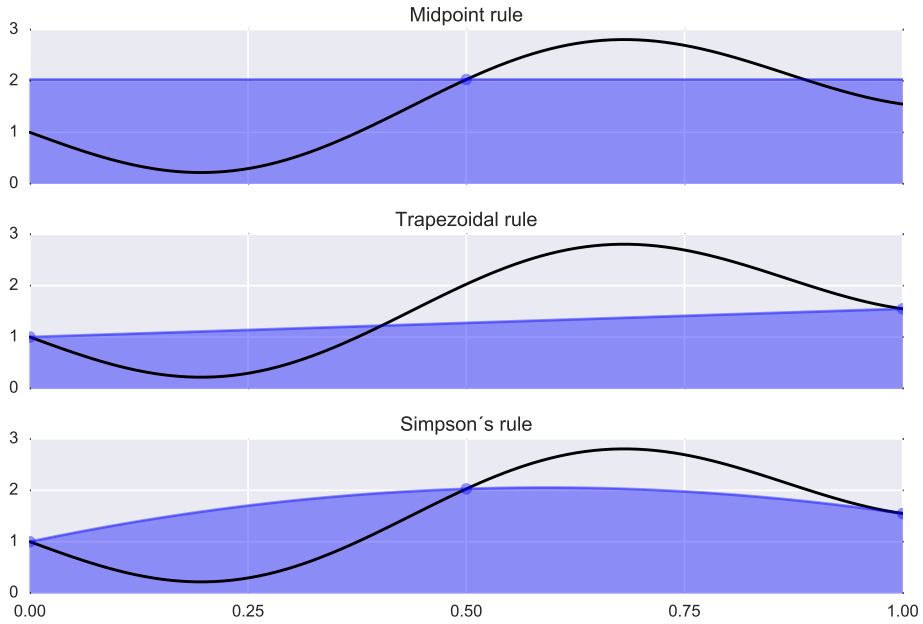

FIGURE 1. Three simple quadrature rules. The shaded area equals the approximate integral of the function f (black). For the midpoint rule the interpolant is piecewise constant, for the trapezoidal rule it is piecewise linear, and for Simpson's rule it is piecewise quadratic.

Trapezoidal rule: If we take the quadrature points to be the endpoints of the interval, we get the trapezoidal rule. The Lagrange polynomials are then $L_0 = \frac{b-x}{b-a}$ and $L_1 = \frac{x-a}{b-a}$, resulting in weights equaling half the interval length.

$$\begin{aligned} &\text{points:} && x_0 = a, \quad x_1 = b \\ &\text{weights:} && w_0 = w_1 = \frac{b-a}{2} \end{aligned}$$

Simpsons rule: For Simpsons rule we have three quadrature points, the two endpoints and the midpoint. The weights are determined the same way as before.

$$\begin{aligned} &\text{points:} && x_0 = a, \quad x_1 = \frac{b+a}{2}, \quad x_2 = b \\ &\text{weights:} && w_0 = w_2 = \frac{b-a}{6}, \quad w_1 = \frac{2}{3}(b-a). \end{aligned}$$

Note that all the above quadrature rules have equidistant points. Rules with four or more points can be derived the same way, and this class of numerical integration methods with equidistant quadrature points is called Newton-Cotes quadrature.

3. Composite quadrature rules

Composite quadrature rules can be constructed from simple ones by dividing the interval $[a, b]$ into a number of subintervals, and applying a simple quadrature rule to each subinterval. This way we easily obtains a more accurate quadrature rule by increasing the number of quadrature points. For example, we divide the interval into N subintervals $[a_k, b_k]$, $k = 1, \ldots, N$, and apply the trapezoidal rule to each to obtain

$$\int_a^b f(x)\,dx = \sum_{k=1}^{N} \int_{a_k}^{b_k} f(x)\,dx \approx \sum_{k=1}^{N} \big(f(a_k) + f(b_k)\big)\frac{b_k - a_k}{2}.$$

If we now assume that all the subintervals have the same length, i.e. $b_k - a_k = h = \frac{b-a}{N}$, and let $x_k = a + (k-1)h$, $k = 1, \ldots, N+1$, then we get

$$\begin{aligned}\int_a^b f(x)\,dx &\approx \sum_{k=1}^{N} \big(f(x_k) + f(x_{k+1})\big)\frac{h}{2} \\ &= f(x_1)\frac{h}{2} + \sum_{k=2}^{N} f(x_k)h + f(x_{N+1})\frac{h}{2}.\end{aligned}$$

We can see that the composite quadrature rule has of $N+1$ equidistant points in $[a, b]$ with weights $h/2$ for the two end points and h for the remaining points.

Composite versions of other simple quadrature rules can derived the same way, with some presented below. For simplicity, the interval is chosen to be $[0, 1]$, and $h = N^{-1}$. See also Figure 2.

Composite midpoint rule:

$$\begin{aligned}&\text{points:} && x_i = \left(i - \frac{1}{2}\right)h, && i = 1, \ldots, N, \\ &\text{weights:} && w_i = h, && i = 1, \ldots, N\end{aligned}$$

Composite trapezoidal rule:

$$\begin{aligned}&\text{points:} && x_i = (i-1)h, && i = 1, \ldots, N+1 \\ &\text{weights:} && w_i = \begin{cases} \frac{h}{2} & i = 1 \text{ or } i = N+1 \\ h & 1 < i < N+1 \end{cases}\end{aligned}$$

Composite Simpsons rule:

$$\begin{aligned}&\text{points:} && x_i = (i-1)\frac{h}{2}, && i = 1, \ldots, 2N+1, \\ &\text{weights:} && w_i = \begin{cases} \frac{h}{6} & i = 1 \text{ or } i = 2N+1 \\ \frac{4h}{6} & i = 2, 4, 6, \ldots, 2N \\ \frac{2h}{6} & i = 3, 5, 7, \ldots, 2N-1 \end{cases}\end{aligned}$$

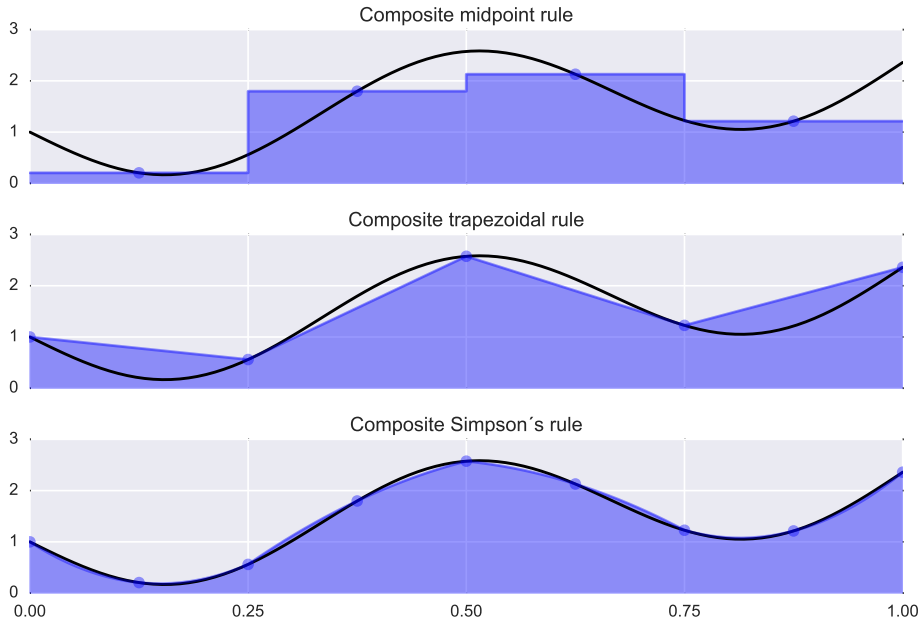

FIGURE 2. Three composite quadrature rules with $n = 5$ subintervals. The shaded area equals the approximate integral of the function f (black). For the midpoint rule the approximation is piecewise constant, for the trapezoidal rule it is piecewise linear, and for Simpson's rule it is piecewise quadratic.

3.1. **Quadrature error.** For the composite quadrature rules described above, it can be shown that the error is bounded according to the estimate

$$\left| \int_a^b f(x)\,dx - \sum_{i=1}^{N} f(x_i) w_i \right| \leq Ch^r, \tag{6}$$

where again $h = (b-a)/N$ is the length of the subintervals, C is a constant independent of h (but depending on f), and r is the rate of convergence. A derivation of the estimate can be found in standard textbooks[2].

4. Application to the numerical solution of ordinary differential equations

To show another application of simple quadrature rules, we aim to approximate the solution of

$$y'(t) = f(t, y(t)), \quad t > t_0; \quad y(t_0) = y_0, \tag{7}$$

where $f : [t_0, \infty) \times \mathbb{R} \to \mathbb{R}$, $y_0 \in \mathbb{R}$ is a given number.

The theta method. A family of one-step approximation procedures can be derived using simple quadrature rules for integration as follows. Let $\theta \in [0, 1]$ and $h > 0$.

[2]See, for example, Süli's book, Chapter 7.

Integrate (7) from t_0 to $t_0 + h$ and approximate the integral by a quadrature to get

$$y(t_0+h) = y(t_0) + \int_{t_0}^{t_0+h} f(\tau, y(\tau))\,\mathrm{d}\tau \approx y_0 + \theta h f(t_0, y_0) + (1-\theta) h f(t_0+h, y(t_0+h)).$$

Here we used a quadrature with quadrature points t_0 and $t_0 + h$ and quadrature weights θh and $(1-\theta)h$, respectively. Motivated by this, given a sequence of equidistant time-instances $t_0, t_1 = t_0 + h, t_2 = t_0 + 2h, ...$ we define the approximation

$$y_1 = y_0 + \theta h f(t_0, y_0) + (1-\theta) h f(t_1, y_1),$$

or, more generally,

$$y_{n+1} = y_n + \theta h f(t_n, y_n) + (1-\theta) h f(t_{n+1}, y_{n+1}), \quad n = 0, 1, 2, \ldots. \tag{8}$$

The family of methods defined by (8) is called the *theta method*. We highlight some special cases:

(1) When $\theta = 1$, we get

$$y_{n+1} = y_n + h f(t_n, y_n), \quad n = 0, 1, 2, \ldots, \tag{9}$$

which is called the *Euler method*.

(2) When $\theta = 0$, we get

$$y_{n+1} = y_n + h f(t_{n+1}, y_{n+1}), \quad n = 0, 1, 2, \ldots,$$

which is called the *implicit Euler*, or *backward Euler* method.

(3) When $\theta = \frac{1}{2}$, we get

$$y_{n+1} = y_n + \frac{1}{2} h f(t_n, y_n) + \frac{1}{2} h f(t_{n+1}, y_{n+1}), \quad n = 0, 1, 2, \ldots,$$

which is called the *Crank-Nicolson* method or the *trapezoidal rule* (c.f. trapezoidal rule for numerical integration).

Unless $\theta = 1$ (the Euler method), the theta method is implicit. This means that at each time step one has to solve a possibly nonlinear algebraic equation. Therefore, it may seem like there is no point using values of θ other than 1. However, when $\theta = \frac{1}{2}$, and f is twice continuously differentiable, then the asymptotic order of the method is 2 which vaguely means that the approximation error decays quadratically as $h \to 0$. For other values of θ the asymptotic order is 1; that is, the approximation error decays linearly as $h \to 0$, but the method might have other desirable properties which makes it useful for a class of problems (such as, for so-called stiff differential equations when $\theta = 1$), or long-term approximation of differential equations.

Exercise 1.

(a) Implement the composite midpoint, trapezoidal rule and Simpsons rules in Matlab, as functions taking as input the integrand, the left and right endpoints of the interval, and the number of subintervals. That is, the quadrature rules should be implemented in a `.m` file starting with something like `function quad_midpoint(f, a, b, N)`.

Test the implemented quadrature rules on an integral where the exact value is known, for example

$$\int_0^{\pi/2} \sin(x)\, dx = 1.$$

It may be useful to define the integrand as an inline function. For example, the function above can be implemented in Matlab with `f = @(x) sin(x);`.

(b) Determine numerically the convergence rate r in the estimate (6) for the three methods you have implemented in (a). Hint: If we look at the the ratio of the errors with $\frac{h}{2}$ and h (2N and N, respectively), we have

$$\frac{\text{error}(h)}{\text{error}(\frac{h}{2})} \sim \frac{2^r h^r}{h^r} = 2^r,$$

and we get the estimated convergence rate

$$r \sim \log_2 \left(\frac{\text{error}(h)}{\text{error}(\frac{h}{2})} \right).$$

Compute the errors for $n = 1, 2, 4, 8, \ldots$ and see if the estimated rate converges. You can also make a `loglog` plot with h on the x-axis and the errors on the y-axis and determine the slope.

(c) Suppose we make a mistake when implementing the composite trapezoidal rule, neglecting the half-valued weights at the endpoints and instead set all the weights equal (i.e. $w_i = h$ for $i = 1, \ldots, N+1$). What happens to the convergence rate?

Exercise 2. Consider the ordinary differential equation

$$y'(t) = y(t) - y(t)^3, \quad t > 0; \quad y(0) = y_0.$$

The exact solution is given by

$$y(t) = \frac{y_0}{\sqrt{y_0^2 - \left(y_0^2 - 1\right) * e^{-2t}}}.$$

(a) Implement the Euler, backward Euler and Crank-Nicolson methods with stepsize h to approximate the solution at $T > 0$. The input parameters should be y_0, T, N where $h = T/N$. For the backward Euler and Crank Nicholson methods use a solver you implemented in Lab 1 to solve the nonlinear algebraic equation at each time level.

(b) Demonstrate the asymptotic order of the methods as $h \to 0$ implemented in (a) when $T = 1$ by comparing the exact solution and approximate solutions for different values of h. A recommended value $y_0 = 0.1$. Write an algorithm that checks if the rate is not changing much and it's close to an integer number. Here the recommended precision is 0.01 (both for comparing with the previous rate and the difference between the closest integer and the rate).