1 Numerical integration

Numerical integration is an algorithm, also called quadrature, to compute an approximation to a definite integral in the form of a finite sum,

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

where the abscissas x_i and the weights w_i are chosen such that the quadrature is particularly well suited for a given problem. Different quadratures use different strategies of choosing the abscissas and weights.

1.1 Quadratures with equally spaced abscissas (Newton-Cotes)

Classical Newton-Cotes quadratures use predefined, usually equally-spaced, abscissas. A quadrature is called *closed* if the abscissas include the end-points of the interval or the mid-point (which becomes end-point after halving the interval). Otherwise it is called open.

If the integrand is diverging at the end-points (or at the mid-point of the interval) the close quadratures generally can not be used.

For an n-point classical quadrature the n free parameters w_i can be chosen such that the quadrature integrates exactly a set of n functions. In the case of Newton-Cotes quadratures these functions are polynomials $[1, x, x^2, \dots, x^{n-1}]$. The n-point Newton-Cotes quadrature thus integrates exactly the first n terms of the function's Taylor expansion[1]

$$f(a+t) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!} t^k \;. \tag{3}$$

The nth order term $\frac{f^{(n)}(a)}{n!}t^n$ will not be integrated exactly by an n-point quadrature and will then result in the quadrature's error[2]

$$\epsilon_n \propto \int_0^h \frac{f^{(n)}(a)}{n!} t^n dt = \frac{f^{(n)}(a)}{n!(n+1)} h^{n+1} \;. \tag{4}$$

If the function is smooth and the interval h is small enough the Newton-Cotes quadrature can give a good approximation.

[1] assuming that the integral is rescaled as

$$\int_a^b f(x)dx = \int_0^{h=b-a} f(a+t)dt \;. \tag{2}$$

[2] Actually the error is often one order in h higher due to symmetry of the the polynomials t^k.

Here are a few examples of classical quadratures: one-point closed,

$$\int_0^h f(x)dx \approx hf(\frac{1}{2}h) \;, \tag{5}$$

three-point closed,

$$\int_0^h f(x)dx \approx \frac{h}{6}\left(f(0) + 4f(\frac{1}{2}h) + f(h)\right) \;, \tag{6}$$

two-point open,

$$\int_0^h f(x)dx \approx \frac{h}{2}\left(f(\frac{1}{3}h) + f(\frac{2}{3}h)\right) \;, \tag{7}$$

four-point open,

$$\int_0^h f(x)dx \approx \frac{h}{6}\left(\begin{array}{c} 2f(\frac{1}{6}h) + f(\frac{2}{6}h) \\ +f(\frac{4}{6}h) + 2f(\frac{5}{6}h) \end{array}\right) . \tag{8}$$

1.2 Quadratures with optimized abscissas (Gauss)

Gauss quadratures use optimally chosen weights w_i and, in addition, also abscissas x_i. The number of free parameters is thus $2n$ (n abscissas and n weights) the Gaussian quadratures are of order $2n-1$ compared to only order $n-1$ for equally spaced abscissas.

Here is, for example, a two-point Gauss-Legendre quadrature rule [3]

$$\int_{-1}^1 f(x)dx \approx f\left(-\sqrt{\frac{1}{3}}\right) + f\left(+\sqrt{\frac{1}{3}}\right) . \tag{10}$$

Unfortunately, the optimal points generally can not be reused at the next iteration in an adaptive algorithm (cannot be nested).

1.3 Subdivision of the interval vs higher order quadrature

The higher order quadratures, say $n > 10$, suffer from round-off errors as the weights w_i generally have alternating signs. Again, using high order polynomials is dangerous as they typically oscillate wildly and may lead to Runge phenomenon. Therefore if the error of the quadrature is yet too big for a sufficiently large n quadrature, the best strategy is to subdivide the interval in two and then use the quadrature on the half-intervals. Indeed, if the error is of the order h^k, the subdivision would lead to reduced error, $2\left(\frac{h}{2}\right)^k < h^k$, if $k > 1$.

[3] assuming that the integral is rescaled as

$$\int_a^b f(x)dx = \int_{-1}^1 \frac{b-a}{2} f\left(\frac{a+b}{2} + \frac{b-a}{2}t\right) dt \;. \tag{9}$$

1.4 Adaptive quadratures

Adaptive quadrature is an algorithm where the integration interval is subdivided into adaptively refined subintervals until the given accuracy is reached.

Adaptive algorithms are built on pairs of quadrature rules, a higher order rule (eg open-4) and a lower order rule (eg open-2). The higher order rule is used to compute the approximation to the integral. The difference between the higher order rule and the lower order rule gives an estimate of the error. If the error is larger than the required accuracy, the interval is subdivided into (two) smaller subintervals and the procedure applies recursively to the subintervals. The reuse of the function evaluations made at the previous step is very important for the efficiency of the algorithm. The equally-spaced abscissas naturally provide such a reuse.

1.5 Gauss-Kronrod quadratures

Gauss-Kronrod quadratures represent a compromise between equally spaced abscissas and optimal abscissas: n points are reused from the previous iteration (n weights as free parameters) and then m optimal points are added (m abscissas and m weights as free parameters). Thus the accuracy of the method is $n + 2m - 1$. There are several special variants of these quadratures fit for particular types of the integrands.