Numerical Integration

Gordon K. Smyth

May 1997

Numerical integration is the study of how the numerical value of an integral can be found. Also called *quadrature*, which refers to finding a square whose area is the same as the area under a curve, it is one of the classical topics of numerical analysis*. Of central interest is the process of approximating a definite integral from values of the integrand when exact mathematical integration is not available. The corresponding problem for multiple dimensional integration is known as multiple integration or *cubature*.

Numerical integration has always been useful in biostatistics to evaluate distribution functions and other quantities. Emphasis in recent years on Bayesian and empirical Bayesian methods and on mixture models has greatly increased the importance of numerical integration for computing likelihoods and posterior distributions and associated moments and derivatives. Many recent statistical methods are dependent especially on multiple integration, possibly in very high dimensions.

This article describes classical quadrature methods and, more briefly, some of the more advanced methods for which software is widely available. The description of the elementary methods in this article borrows from introductory notes by Stewart [31]. An excellent general reference on numerical integration is [5]. More recent material can be found in [8] and [29]. Recent surveys of numerical integration with emphasis on statistical methods and applications are [10] and [9].

Trapezoidal Rule

The simplest quadrature rule in wide use in the *trapezoidal rule*. Like many other methods, it has both a geometric and an analytic derivation. The idea of the geometric derivation is to approximate the area under the curve $y = f(x)$ from $x = a$ to $x = b$ by the area of the trapezoid bounded by the points $(a, 0)$, $(b, 0)$, $(a, f(a))$ and $(b, f(b))$. This gives

$$\int_a^b f(x)dx \approx \frac{b-a}{2}\{f(a) + f(b)\}.$$

The analytic derivation is to interpolate $f(x)$ at a and b by a linear polynomial.

The trapezoidal rule cannot be expected to give accurate results over a larger interval. However by summing the results of many applications of the trapezoidal rule over smaller intervals, we can obtain an accurate approximation to the integral over any interval. We begin by dividing $[a, b]$ into n equal intervals by the points

$$a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b.$$

Specifically, if

$$h = \frac{b-a}{n}$$

is the common length of the intervals, then

$$x_i = a + ih, \quad i = 0, 1, \ldots, n.$$

Applying the trapezoidal rule to each interval $[x_{i-1}, x_i]$ gives the *composite trapezoidal rule*

$$\int_a^b f(x)dx \approx h\left\{\frac{f(x_0)}{2} + f(x_1) + \cdots + f(x_{n-1}) + \frac{f(x_n)}{2}\right\}.$$

An error formula for the composite trapezoidal rule can be obtained from polynomial approximation theory. Let $\mathrm{CT}_h(f)$ denote the approximation produced by the composite trapezoidal rule applied to f on $[a, b]$. If f is twice continuously differentiable on (a, b), then

$$\int_a^b f(x)dx - \mathrm{CT}_h(f) = -\frac{(b-a)f''(\xi)}{12}h^2$$

for some $\xi \in [a, b]$. Note that because the error decreases as h^2, that doubling the number of points reduces the error by a factor of four.

Simpson's Rule

More sophisticated quadrature rules can produce higher order error terms. Even more popular than the trapezoidal rule is *Simpson's rule*:

$$\int_a^b f(x)dx \approx \frac{b-a}{6}\left\{f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right\}.$$

Simpson's rule can be derived by interpolating $f(x)$ by a quadratic polynomial at a, $(a + b)/2$ and b.

As with the trapezoidal rule, Simpson's rule is usually applied to many short intervals. Letting the x_i be as above for n even and writing $f_i = f(x_i)$, the *composite Simpson rule* is

$$\int_a^b f(x)dx \approx \frac{h}{3}(f_0+4f_1+2f_2+4f_3+\cdots+2f_{n-2}+4f_{n-1}+f_n).$$

If $\mathrm{CS}_h(f)$ denotes the result of applying the composite Simpson rule to f over the interval $[a,b]$, and if f has continuous fourth derivative on (a,b), then

$$\int_a^b f(x)dx - \mathrm{CS}_h(f) = -\frac{(b-a)f^{(4)}(\xi)}{180}h^4,$$

where $\xi \in [a,b]$. Although Simpson's rule was derived to integrate quadratic polynomials exactly on each interval, the presence of the fourth derivative in the error term signals that it in fact integrates cubics exactly as well. This property follows from the fact that Simpson's rule is a special case of Gaussian quadrature, treated below.

Newton-Cotes Formulas

The trapezoidal rule integrates any linear polynomial exactly. In general we might look for an $n+1$ point rule which integrates exactly any polynomial of degree n. Such a quadrature rule is the *Newton-Cotes formula.*

Let $x_0, x_1, \ldots, x_n$ be distinct points in the interval $[a,b]$. We wish to determine constants $A_0, A_1, \ldots, A_n$ such that

$$\int_a^b f(x)dx = A_0 f(x_0) + \ldots + A_n f(x_n)$$

for any polynomial f of degree $\leq n$. This problem has an elegant solution in terms of *Lagrange polynomials.* The ith Lagrange polynomial over the points $x_0, \ldots, x_n$ is defined to be

$$\ell_i(x) = \frac{\prod_{j=0}^n (x - x_j)}{\prod_{j=0}^n (x_i - x_j)}, \quad i = 0, 1, \ldots, n.$$

Then

$$A_i = \int_a^b \ell_i(x)dx, \quad i = 0, 1, \ldots, n,$$

are the unique coefficients giving the Newton-Cotes formula. Strictly speaking, as Newton-Cotes usually refers to formulas with equally spaced abscissas, this is a slight generalization.

Although the A_i have an elegant analytic expression, they are difficult to evaluate stably. For rules of low degree, one can use the *method of undetermined coefficients.* We will illustrate the technique with a three-point formula over the interval $[0,1]$ on the points 0, $\frac{1}{2}$ and 1. The exactness property requires that the rule integrate any function that is identically one. In other words

$$1 \cdot A_0 + 1 \cdot A_1 + 1 \cdot A_2 = \int_0^1 1dx = 1.$$

The rule must also integrate the function x, which gives

$$0 \cdot A_0 + \frac{1}{2} \cdot A_1 + 1 \cdot A_2 = \int_0^1 xdx = \frac{1}{2}.$$

Finally, the rule must integrate the function x^2, which gives

$$0 \cdot A_0 + \frac{1}{4} \cdot A_1 + 1 \cdot A_2 = \int_0^1 x^2dx = \frac{1}{3}.$$

The solution of these three equations is

$$A_0 = A_2 = \frac{1}{6}$$

and

$$A_1 = \frac{2}{3},$$

which is just Simpson's rule.

Clenshaw-Curtis Integration

Newton-Cotes formulas with equally spaced abscissas are of practical use only for small point numbers, say $n \leq 8$. For n as low as nine, the coefficients A_i vary in sign. As n increases, the coefficients become large in absolute value, leading to unstable evaluation of the integral. This problem can be avoided by choosing the abscissas in a more sophisticated way. One choice for which the coefficients are not only positive but have stable analytic expressions is the Chebyshev points on $[a,b]$,

$$x_i = \frac{b+a}{2} + \frac{b-a}{2}\cos\frac{i\pi}{n}, \quad i = 0, 1, \ldots, n.$$

Define the modified Fourier coefficients,

$$a_j = \frac{2}{n}\sum_{i=0}^{n}{}'' f(x_i)\cos\frac{ij\pi}{n}$$

where $''$ indicates that the first and last terms in the sum are to be halved. If n is even, then the *Clenshaw-Curtis formula* can be written

$$\begin{aligned}\int_a^b f(x)dx \quad &\approx \quad \frac{b-a}{2}\left\{a_0 - \frac{2a_2}{(1)(3)} - \frac{2a_4}{(3)(5)} - \cdots\right.\\ &\left. -\frac{2a_{n-2}}{(n-3)(n-1)} - \frac{a_n}{(n-1)(n+1)}\right\}.\end{aligned}$$

Like other formulas of the Newton-Cotes type, Clenshaw-Curtis will integrate exactly polynomials of order n or less. In practice, it does rather better than other rules of the same order, because of the bounded variation properties of Chebyshev polynomials. The error of Clenshaw-Curtis integration can be estimated from the rate of decrease of the coefficients a_j.

O'Hara and Smith [23] suggest the use of bounds such as $\max(2|a_{n-4}|, 2|a_{n-2}|, |a_n|)$ for the approximation error.

Treatment of Singularities

Provided that the integrand f is sufficiently smooth, the Newton-Cotes formulas converge as $n \to \infty$. It sometimes happens however that one has to integrate a function with a singularity. Suppose for example

$$f(x) \approx \frac{c}{x^d}$$

with $0 < d < 1$ for x near zero. Then $\int_0^1 f(x)dx$ exists, but the Newton-Cotes formulas will not obtain good results because f is not at all polynomial on $[0, 1]$. A better approach is to incorporate the singularity into the quadrature rule itself.

First define

$$g(x) = x^d f(x),$$

and look for a rule that evaluates the integral

$$\int_0^1 g(x)x^{-d}dx$$

where g is a well-behaved function on $[0, 1]$. The function x^{-d} is called a *weight function*. Given any modest number of points $x_0, \ldots, x_n$ in the interval $(0, 1]$, the method of undetermined coefficients can easily determine an integration rule of the form

$$\int_0^1 g(x)x^{-d}dx = A_0 f(x_0) + \ldots + A_n f(x_n)$$

by substituting in $g(x) = 1$, $g(x) = x$, $g(x) = x^2$ etc.

The appearance of derivatives in the error terms for Newton-Cotes rules (and for the Gaussian rules below) shows that the method is troubled not only by singularities in the integrand, but by singularities in its derivatives as well. A weight function may therefore need to remove singularities in the derivatives as well as in the function itself.

Gaussian Quadrature

A polynomial of degree n is determined by its $n+1$ coefficients. We have seen that the $n+1$ coefficients $A_0, \ldots, A_n$ in the $n+1$-point Newton-Cotes formula can be chosen to make the rule exact for polynomials of degree n or less. The idea behind Gaussian quadrature is that the abscissas $x_0, \ldots, x_n$ represent another $n+1$ degrees of freedom, which may be used to extend the exactness of the rule to polynomials of degree $2n+1$.

Gauss quadrature formulae have the form

$$\int_a^b f(x)w(x)dx \approx A_0 f(x_0) + \ldots + A_n f(x_n),$$

where $w(x)$ is a weight function which is greater than zero on the interval $[a, b]$. The correct choice for $x_0, \ldots, x_n$ turns out to be the zeros of an orthogonal polynomial of order $n+1$.

Two functions f and g are said to be *orthogonal* with respect to $w(x)$ on $[a, b]$ if

$$\int_a^b f(x)g(x)w(x)dx = 0.$$

For each interval and each appropriate weight function, there is a unique sequence $\{p_i\}_{i=0}^{\infty}$ of polynomials with $\deg(p_i) = i$ such that each polynomial in the sequence is orthogonal to all others. This sequence can be generated by a simple recurrence, which is described in the article on polynomial interpolation*.

Let $x_0, \ldots, x_n$ be the zeros of the orthogonal polynomial p_{n+1}. It can be shown that the x_i are real, simple, and lie in the interval $[a, b]$. Set

$$A_i = \int_a^b \ell_i(x)w(x)dx, \quad i = 0, \ldots, n$$

where ℓ_i is the ith Lagrange polynomial over $x_0, \ldots, x_n$. For any function f let

$$\mathrm{G}_n f = A_0 f(x_0) + \cdots + A_n f(x_n).$$

Then

$$\mathrm{G}_n f = \int_a^b f(x)w(x)dx$$

for any polynomial f of degree less than or equal to $2n+1$.

An important point is that the coefficients A_i are positive. Moreover, $A_0 + A_1 + \cdots + A_n = \int_a^b w(x)dx$, so no coefficient can be larger than $\int_a^b w(x)dx$. Consequently we cannot have a situation in which large coefficients create large intermediate results that suffer cancellation when they are added.

Gaussian quadrature has error formulas similar to those for Newton-Cotes formulas. Specifically if f is $2n+2$ times continuously differentiable on (a, b), then

$$\int_a^b f(x)w(x)dx - \mathrm{G}_n f = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \int_a^b p_{n+1}^2 w(x)dx$$

where $\xi \in [a, b]$. If f does not satisy the smoothness property then the accuracy of Gaussian quadrature is generally reduced by at least an order of magnitude. However a consequence of the positivity of the coefficients A_i is that Gaussian quadrature converges for any continuous function as $n \to \infty$.

Particular Gauss formulas arise from particular choices of the interval $[a, b]$ and the weight function $w(x)$. The workhorse is *Gauss-Legendre* quadrature in which $[a, b] =$

$[-1, 1]$ and $w(x) = 1$, so that the formula approximates the integral

$$\int_{-1}^{1} f(x)dx.$$

The corresponding orthogonal polynomials are called Legendre polynomials.

If we take $[a, b] = [0, \infty)$ and $w(x) = e^{-x}$, we get a formula to approximate

$$\int_{0}^{\infty} f(x)e^{-x}dx.$$

This is called *Gauss-Laguerre* quadrature.

If we take $[a, b] = [-\infty, \infty]$ and $w(x) = e^{-x^2}$, we get a formula to approximate

$$\int_{-\infty}^{\infty} f(x)e^{-x^2}dx$$

This is *Gauss-Hermite* quadrature.

Computing the abscissas and coefficients for these and other Gauss rules in a stable and efficient manner is a challenging nonlinear problem. Two successful algorithms are those of Golub and Welsch [13] and Sack and Donovan [28]. A fortran program implementing the Golub-Welsch method can be obtaining by searching the NETLIB* database for GAUSSQ. The expense of computing the abscissas and coefficients is sufficiently great that they are usually stored and re-used rather than generated afresh for each problem.

Simpson's rule is actually a variant of the Gauss-Legendre 3-point rule in which x_0 and x_n are constrained to be the end-points. Rules with such constraints are called *Gauss-Radau* or *Gauss-Lobatto* quadrature [5].

Progressive Formula

Despite their optimal properties, the Gaussian formulas are not universally used in practice. The main reason for this is the difficulty of determining in advance the required number of points to achieve a given level of accuracy. In some cases, mathematical analysis of the function to be integrated makes it possible to use the analytic error bounds of the quadrature rules. It is more common however to estimate the error empirically by applying the same quadrature rule twice with different point-numbers. Often the point-number is doubled until the successive values of the integral agree to the required number of figures.

A succession of integration formulas with increasing point-numbers is said to be *nested* or *progressive* if each formula re-uses the abscissas of the earlier formulas. The composite Simpson and Clenshaw-Curtis rules with n doubling at each step are important examples of progressive formulas. Gaussian formulas are generally not progressive, as the abscissas at any point-number are different from those for any other point-number. The relative advantage of the Gauss formulas is therefore lost in the expense of computing addition abscissas and function evaluations.

One possibility is to construct progressive formulas starting or finishing with a Gaussian formula. Kronrod [19] gave a method for adding points to a Gauss-Legendre formula in an optimal way. The Kronrod rule adds $n + 1$ points to a n-point Gauss-Legendre formula, resulting in a rule which integrates exactly polynomials of order $3n + 1$ (n even) or $3n + 2$ (n odd). The desirable properties of Gaussian quadrature are preserved in that the abscissas remain in the integration interval and the coefficients A_i remain positive. When the n-point Gauss rule is combined with its Kronrod optimal extension, a very economical pair of formulas result for the simultaneous calculation of an approximation for an integral and the respective error estimate. The problem of extending arbitrary quadrature formulas in a progressive fashion was studied by Patterson [24] [25] [18], who also gave a stable computation for the Kronrod rules. Together, the Kronrod and Patterson methods provide a nested sequence of quadrature rules based on an initial Gauss rule, and are the basis of some of the most widely used integration programs.

Adaptive Methods

A quadrature rule is *adaptive* if it compensates for a difficult subrange of an integrand by automatically increasing the number of quadrature points in the awkward region. Adaptive strategies divide the integration interval into subintervals and, typically, employ a progressive formula in each subinterval with some fixed upper limit on the number of points allowed. If the required accuracy is not achieved by the progressive formula, then the subinterval is bisected and a similar procedure carried out on each half. This subdivision process is carried out recursively until convergence is achieved in each of the terminating subintervals. Most general purpose integration programs are adaptive, since such a strategy can be successful over a very wide range of integrands.

Multiple Integration: Product Rules

Multiple integration is concerned with the numerical approximation of integrals of two or more variables. It is not a simple extension of one-dimensional integration. The diversity of possible integration regions and singularities for d-dimensional functions is daunting. As a general rule, it is not possible to obtain the same accuracy with higher-dimensional integrals as with one-dimensional integrals for reasonable computing times.

The problem addressed by multiple integration is to evaluate integrals of the form

$$\int f \quad = \quad \int_{a_d}^{b_d} \int_{a_{d-1}(x_d)}^{b_{d-1}(x_d)} \cdots$$

$$\int_{a_1(x_1,\ldots,x_d)}^{b_1(x_1,\ldots,x_d)} f(x_1, x_2, \ldots, x_d) dx_1 dx_2 \ldots dx_d.$$

The most obvious approach is to treat the multiple integral as a nested sequence of one-dimensional integrals, and to use one-dimensional quadrature with respect to each argument in turn. The resulting multiple integration formula is a *product rule.*

Suppose that the integration region is a hyper-rectangle, so that the integration interval $[a_j, b_j]$ for x_j in the above integral is independent of $x_{j+1}, \ldots, x_d$. If Gauss quadrature is used to integrate f with respect to x_j, with abscissas $x_{j0}, x_{j1}, \ldots, x_{jn}$ and coefficients $A_{j0}, A_{j1}, \ldots, A_{jn}$, then the product rule is

$$\int f \approx \sum_{i_0, i_1, \ldots, i_d = 0}^{n} A_{0i_0} A_{1i_1} \cdots A_{di_d} f(x_{0i_0}, x_{1i_1}, \ldots, x_{di_d}).$$

This rule integrates exactly any sum of monomials $x_1^\alpha x_2^\beta \ldots x_n^\gamma$, where each $\alpha, \beta, \ldots, \gamma$ is an integer between zero and $2n+1$, a result which derives directly from the corresponding result for the one-dimensional Gauss rule.

The number of evaluations of f in the product rule is $(n+1)^d$, which grows exponentially with d. Rapid growth in the number of function evaluations usually limits the practical use of product rules to around 5 or 6 dimensions. One special case common in statistical applications is that in which $x_1, \ldots, x_d$ are exchangeable. This arises when $x_1, \ldots, x_d$ is an independent sample from some distribution and f is a function of the probability density. In that case only one evaluation of f is needed for all points which are permutations of one another. The total number of evaluations required is then $\binom{n+d}{d}$, which is considerably smaller than $(n+1)^d$, so that calculations for sample sizes up about 10 are manageable.

Despite the above limitations, Gauss product rules have been the basis of at least one general approach to implementing Bayesian analysis methods, discussed in [21] [30].

Rules of Polynomial Degree

As with quadrature, most cubature rules are designed to integrate a certain class of polynomials exactly. A rule is said to be of polynomial degree r if it integrates exactly any sum of monomials $x_1^{k_1} \cdots x_d^{k_d}$ with $k_1 + \cdots + k_d \le r$. Although Gauss product rules integrate certain monomials of higher order, they do not integrate x_j^{2n+2} exactly and are therefore of polynomial degree $2n+1$.

By allowing rules that are not product rules, it is usually possible to find rules which are more efficient than the product-Gauss rules in the sense of having polynomial degree $\ge 2n+1$ yet requiring fewer than $(n+1)^d$ points. Methods for constructing rules of prescribed polynomial degree are surveyed in [3]. For a compilation of such rules see [32] and [4].

Polynomial rules of degrees five and seven on the hyper-rectangle serve as basic integrating rules for the popular multiple integration program ADAPT [12], which is described further below.

Globally Adaptive Algorithms

One dimensional adaptive programs usually consider each subinterval in turn, subdividing each until a specified accuracy is obtained. This straightforward strategy is called *locally adaptive* because the behavior of the algorithm in each local subinterval depends only on the error estimates in that interval. However, for multiple integrals it is often unknown at the beginning of the calculation whether the given accuracy can be obtained in a reasonable amount of time. A popular adaptive strategy, originally proposed by van Dooren and de Ridder [33], always subdivides the integration subregion with the largest error. Such a strategy is known as *globally adaptive* because it makes subdivision decisions using information about all the current subregions. Although globally adaptive algorithms require more memory space to maintain the current subregion list and take more time to select subregions for subdivision, at each stage in the calculation the global estimate for $\int f$ is in some sense the best one available using the computation that has been done so far.

The globally adaptive program ADAPT [12] and its successor DCUHRE [1] [2] build on the work of van Dooren and de Ridder [33]. ADAPT uses the difference between nested pair of polynomial rules, of degrees seven and five respectively, to estimate the error in each subregion. Some of the degree seven integrand values are also used to compute fourth differences in directions parallel to each of the coordinate axes. When a subregion is selected for subdivision, it is divided in half in the direction of largest absolute fourth difference. This clever strategy for halving in only one direction, using fourth differences to measure integrand irregularity, is probably one of the main reasons for the practical effectiveness of the algorithm. The later program DCUHRE gives the user a choice of integration rules, uses a more sophisticated error estimate, and is organized to facilitate parallel integration of a vector of related integrands.

Lattice Methods

Lattice rules were originally called "number theoretic" or "quasi-random" methods [32]. The integration region is translated to the unit cube, and the integral approximated by a multiple sum of the form

$$Qf = \frac{1}{n_1 n_2 \cdots n_t} \sum_{j_t=0}^{n_t-1} \cdots \sum_{j_1=0}^{n_1-1} f\left(\frac{j_1}{n_1}\mathbf{z}_1 + \cdots + \frac{j_t}{n_t}\mathbf{z}_t\right),$$

where $\mathbf{z}_1, \ldots, \mathbf{z}_t$ are carefully selected integer vectors. This is the simple unweighted mean of the integrand evaluated over a regular lattice of abscissas in the unit cube. For the method to work well, the integrand must be transformed to be periodic in the cube so that its Fourier coefficients go to zero rapidly.

A lattice method originated by Korobov [17] and extended by Patterson and Cranley [26] is implemented in the NAG library* routines D01GCF and D01GDF. Lattice methods are not yet in widespread use, but there is some evidence [11] [29] that they can outperform other available methods when the number of dimensions is between about 10 to 20.

Monte Carlo Methods

The idea of estimating an integral by random sampling is a natural one in a statistical context. In the classical Monte Carlo method [20] [14] points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are chosen randomly in the integration region and the integral is estimated by

$$\bar{f} = \frac{V}{n} \sum_{i=1}^{n} f(\mathbf{x}_i),$$

where V is the volume of the integration region. Convergence is guaranteed almost surely by the central limit theorem under very weak conditions on f. Moreover, the rate of convergence is independent of the dimensionality. The error $\bar{f} - \int f$ is approximately normal with mean zero and standard deviation

$$\frac{\sigma(f)}{\sqrt{n}} V,$$

where

$$\sigma^2(f) = \frac{1}{V} \int f^2 - \left(\frac{1}{V} \int f \right)^2$$

is the variance of f. Finally, and most importantly, a free estimate of the error is available as $\sigma^2(f)$ may be estimated by the sample variance of $f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)$.

The slow $n^{-1/2}$ rate of convergence means that Monte Carlo methods are usually limited to low accuracy, say three significant figures. However this accuracy can be achieved with comparable work for any number of dimensions and for a very wide range of integration regions. In many statistical applications higher accuracy is not required; computational error need only be small relative to the inherent statistical uncertainty that enters the process of drawing inferences from data.

Practical use of the Monte Carlo method depends on techniques for reducing the $\sigma^2(f)$ variance term in the error. Central amongst these is *importance sampling*, in which $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are sampled from a distribution which is as much like f in shape as possible. This has the effect of sampling most densely in those parts of the integration region where the integrand is greatest. Specifically, write the integral as

$$\int f(\mathbf{x}) g(\mathbf{x}) d\mathbf{x},$$

where g is a density function on the integration region, and f is as close to a constant function as possible. Points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are sampled from g, and the integral is approximated as before by $\bar{f}$. The standard deviation of $\bar{f}$ is now

$$\frac{\sigma_g(f)}{\sqrt{n}}$$

where

$$\sigma_g^2(f) = \int f^2 g - \left(\int fg \right)^2$$

is the variance of f with respect to the density g.

In Bayesian problems the aim is to approximate posterior means of the form

$$E(f) = \frac{\int f(\mathbf{x}) g(\mathbf{x}) d\mathbf{x}}{\int g(\mathbf{x}) d\mathbf{x}}$$

where $g(\mathbf{x})$ is the un-normalized posterior density of $\mathbf{x}$, and $f(\mathbf{x})$ is some quantity of interest. If sampling is available from a suitable sampling density h, which is hopefully a good approximation to the posterior, then the posterior mean can be approximated by

$$\bar{f} = \frac{\sum_{i=1}^{n} f(\mathbf{x}_i) w(\mathbf{x}_i)}{\sum_{i=1}^{n} w(\mathbf{x})}$$

where w is the *weight function*

$$w(\mathbf{x}) = g(\mathbf{x}) / h(\mathbf{x}).$$

It can be easily shown that $\bar{f}$ is approximately normal with mean $E(f)$ and standard deviation $\sigma / \sqrt{n}$ where

$$\sigma^2 = E[\{g - E(g)\}^2 w]$$

and all expectations are taken with respect to the posterior distribution.

Other variance reduction techniques include stratified sampling and antithetic acceleration. Antithetic acceleration involves generating pairs of identically distributed but negatively correlated points $\mathbf{x}_i^1$ and $\mathbf{x}_i^2$. This tends to produce negatively correlated terms in the sum; the more negative the correlation the lower the variance of the sum. See [5], [14], [16] or [9] for references to variance reduction methods.

The NAG library subroutine D01GBF uses an adaptive Monte Carlo algorithm to integrate over a hyper-rectangle. The number of subregions is doubled at each iteration, and in each the integral and variance are estimated by Monte Carlo sampling. Algorithms also exist which are adaptive in terms of the importance sampling density. Such algorithms refine the importance sampling density adaptively so as to minimize σ^2 during the Monte Carlo process [5] [22].

Conclusions

If a large number of well behaved one-dimensional integrands are to be integrated, and the user is willing to do some analytic analysis to obtain efficiency, then it is hard to go past the classical Gauss quadrature methods. More usually though users will choose to use an automatic integration program of some kind, using computer time to save their own time and to gain reliability.

Reliable and well-documented software for numerical integration can be found by searching the NIST GAMS online catalogue at http://gams.nist.gov under category "h2". See [15] for brief reviews of much of this software. It is also worth searching the STATLIB* database for statistical functions based on these routines.

In one and two dimensions there is a wealth of reliable and effective programs. The leading one-dimensional package currently is QUADPACK by Piessen et al [27]. This is available from the NETLIB* database and is cross classified by GAMS. It has also been incorporated into the NAG*, IMSL* and SLATEC subroutine libraries. QUADPACK provides a suite of programs designed for different types of difficulties such as singularities and oscillatory integrands, and includes a decision tree to guide the user in choosing the appropriate routine. The program QAGS is a particularly robust general purpose integration program, as is the non-QUADPACK program CADRE [6] which is included in the IMSL* library. In statistical applications, however, the integrands are often smooth with a single dominant peak, so the more efficient programs QNG and QAG, which use higher order Gauss, Gauss-Kronrod and Patterson rules, may suffice.

So far there is no reliable suite of programs for multiple integration. Up to 10 or perhaps 15 dimensions, globally adaptive routines such as ADAPT and DCUHRE can be recommended. When the number of dimensions exceeds about 20, Monte Carlo methods are the only ones possible. Mark 17 of the NAG* library includes ten multiple integration programs, including one which implements a Monte Carlo method.

References

[1] Bernsten, J., Espelid, T. O., and Genz, A. (1991). An adaptive algorithm for the approximate calculation of multiple integrals. *ACM Transaction of Mathematical Software*, **17**, 437–451.

[2] Bernsten, J., Espelid, T. O., and Genz, A. (1991). Algorithm 698: DCUHRE: an adaptive multidimensional integration routine for a vector of integrals. *ACM Transaction of Mathematical Software*, **17**, 452–456.

[3] Cools, R. (1992). A survey of methods for constructing cubature formulae. In *Numerical integration: recent developments, software and applications*, T. O. Espelid and Genz, A. C. (eds.), Kluwer Academic Publishers, Dordrecht, pp. 1–24.

[4] Cools, R., and Rabinowitz, P. (1993). Monomial cubature rules since 'Stroud': a compilation. *J. Comput. Appl. Math.*, **48**, 309–326.

[5] Davis, P. J., and Rabinowitz, P. (1984). *Methods of Numerical Integration.* Academic Press, New York. (The leading general text.)

[6] de Boor, C. (1971). CADRE: An algorithm for numerical quadrature. In Rice, J. R. (Ed.), *Mathematical Software*, Academic Press, New York, pp. 417–449.

[7] Espelid, T. O., and Genz, A. C. (Eds.) (1992). *Numerical Integration: Recent Developments, Software and Applications.* Kluwer Academic Publishers, Dordrecht.

[8] Evans, G. (1993). *Practical Numerical Integration.* Wiley, New York.

[9] Evans, M., and Swartz, T. (1995). Methods for approximating integrals in statistics with special emphasis on Bayesian integration problems. *Statistical Science*, **10**, 254–272.

[10] Flournoy, N., and Tsutakawa, R. K. (eds.) (1991). *Statistical multiple integration.* Contemporary Mathematics 115, American Mathematical Society, Providence, RI.

[11] Genz, A. (1984). Testing multiple integration software. In Ford, B., Rault, J.-C., and Thomasset, F. (eds.), *Tools, Methods and Languages for Scientific and Engineering Computation*, North Holland, New York, pp. 208–217.

[12] Genz, A., and Malik, A. A. (1980). An adaptive algorithm for numerical integration over an N-dimensional rectangular region. *J. Comp. Appl. Math.*, **6**, 295–302.

[13] Golub, G. H., and Welsch, J. H., (1969). Calculation of Gaussian quadrature rules. *Mathematics of Computation*, **23**, 221–230. (Computation of Gaussian quadrature rules.)

[14] Hammersley, J. M., and Handscomb, D. C. (1964). *Monte Carlo Methods.* Methuen, London.

[15] Kahaner, D. K. (1991). A survey of existing multidimensional quadrature routines. In [10], pp. 9–22.

[16] Kalos, M. H., and Whitlock, P. A. (1986). *Monte Carlo Methods Volume 1: Basics.* Wiley, New York.

[17] Korobov, N. M. (1959). The approximate calculation of multiple integrals. *Doklady Akademii Nauk SSSR*, **124**, 1207–1210 (Russian).

[18] Krogh, F. T., and van Snyder, W. (1991). Algorithm 699: a new representation of Patterson's quadrature formulae. *ACM Trans. Math. Softw.*, **17**, 457–461.

[19] Kronrod, A. S. (1965). Nodes and weights of quadrature formulas. Consultants Bureau, New York. (Authorized translation of the Russian.)

[20] Metropolis, N. C., and Ulam, S. M. (1949). The Monte Carlo Method. *J. Amer. Statist. Assoc.*, **44**, 335–341.

[21] Naylor, J. C., and Smith, A. F. M. (1982). Applications of a method for the efficient computation of posterior distributions. *Appl. Stat.*, **31**, 214–225.

[22] Oh, M.-S. (1991). Monte Carlo integration via importance sampling: dimensionality effect and an adaptive algorithm. In [10], pp. 165–187.

[23] O'Hara, H., and Smith, F. H. (1968). Error estimation in the Clenshaw-Curtis quadrature formulae. *Computing J.*, **11**, 213–219.

[24] Patterson, T. N. L. (1968). The optimum addition of points to quadrature formulae. *Math. Comp.*, **22**, 847–856.

[25] Patterson, T. N. L. (1989). An algorithm for generating interpolating quadrature rules of the highest degree of precision with preassigned nodes for general weight functions. *ACM Trans. Math. Softw.*, **15**, 123–136.

[26] Patterson, T. N. L., and Cranley, R. (1976). Randomization of number theoretic methods for multiple integration. *SIAM J. Numer. Anal.*, **13**, 904–914.

[27] Piessens, R., De Doncker-Kapenga, E., Überhuber, C. W., and Kahaner, D. K. (1983). *Quadpack, a Subroutine Package for Automatic Integration.* Springer-Verlag, Berlin. (High quality Fortran functions for automatic integration in one-dimension.)

[28] Sack, R. A., and Donovan, A. F. (1972). An algorithm for Gaussian quadrature given modified moments. *Numer. Math.*, **18**, 465–478.

[29] Sloan, I. H., and Joe, S. (1994). *Lattice Methods for Multiple Integration.* Clarendon Press, Oxford.

[30] Smith, A. F. M., Skene, A. M., Shaw, J. E. H., and Dransfield, M. (1985). The implementation of the Bayesian paradigm. *Comm. Stat. A*, **14**, 1079–1102.

[31] Stewart, G. W. (1996). *Afternotes on Numerical Analysis.* Society for Industrial and Applied Mathematics, Philadelphia, PA. (An elegant and elementary introduction to classical one-dimensional quadrature rules.)

[32] Stroud, A. H. (1971). *Approximate Calculation of Multiple Integrals.* Prentice-Hall, Englewood Cliffs, NJ. (Extensive list of n-dimensional rules.)

[33] van Dooren, P., and de Ridder, L. (1976). An adaptive algorithm for numerical integration over an N-dimensional cube. *J. Comput. Appl. Math. 2*, **3**, 207–217.

Related Articles

Numerical Analysis
Polynomial Approximation