Numerical Integration - a Cook's Tour

(no substitute for reading Davis and Rabinowitz (1975))

We begin with a quick review of numerical integration in one dimension, which subdivides into Gaussian, Newton-Cotes rules, and Curtis-Clenshaw rules.

Then we explore multi-dimensional integrals, where Monte Carlo, and Smolyak type rules have been receiving attention, for example in the recent article of Babuska et al.(2010).

Gaussian Integration:

$$\int_a^b w(x)f(x)dx = \sum_{i=1}^{N} w_i f(x_i)$$

is called an N-point rule, where $w(x) > 0$ is a weight function, and w_i and x_i are the *weights* and *ordinates* of the integration rule. The limits of integration, a and b, are customarily replaced by 0 and 1 in deriving the rule. A simple linear change of variable extends the rule to any finite interval (a, b).

Our normal way of presenting this material to students is to say that we will derive the weights w_i and ordinates x_i by insisting that the integration rule correctly integrates as many as we can of the functions 1, x, x^2, x^3 ... in turn. The motivation for this is that if we can expand a general function as a Taylor's series, integrating the function and integrating the low degree polynomials in x are closely related.

Applying this approach to the two point Gauss rule with $w(x) = 1$, we get the familiar equations

$$\int_0^1 1 \; dx = 1 = w_1 + w_2 \tag{1}$$

$$\int_0^1 x \; dx = 1/2 = w_1 \; x_1 + w_2 \; x_2 \tag{2}$$

$$\int_0^1 x^2 \; dx = 1/3 = w_1 \; {x_1}^2 + w_2 \; {x_2}^2 \tag{3}$$

$$\int_0^1 x^3 \; dx = 1/4 = w_1 \; {x_1}^3 + w_2 \; {x_2}^3 \tag{4}$$

These equations are non-linear in the x_i, and the hint in our first year text is to assume $w_1 = w_2$, find x_1 and x_2, and check that all four equations are satisfied.

Could we give a better hint? Yes we could, because our hint will not help if $w(x) \neq 1$, nor does it generalise for higher N.

We could instead say that x_1 and x_2 are the roots of a quadratic equation $x^2 + a_1 x + a_0 = 0$ and then combine together equations 1, 2, and 3 to give

$$\begin{aligned} w_1({x_1}^2 + a_1 x_1 + a_0) + w_2({x_2}^2 + a_1 x_2 + a_0) \\ = 1/3 + 1/2 \; a_1 + a_0 = 0 \end{aligned} \tag{5}$$

and equations 2, 3, and 4 to give

$$\begin{aligned} w_1({x_1}^3 + a_1 {x_1}^2 + a_0 x_1) + w_2({x_2}^3 + a_1 {x_2}^2 + a_0 x_2) \\ = 1/4 + 1/3 \; a_1 + 1/2 \; a_0 = 0 \end{aligned} \tag{6}$$

This approach will work when $w(x) \neq 1$, and generalises to higher values of N, and would therefore be a better hint to give. Once x_1 and x_2 are found, it is easy to solve the linear equations for w_1 and w_2, which do indeed turn out to be equal (to 1/2).

The generalisation to higher N is called the method of moments. If

$$M_i = \int_0^1 w(x) x^{i-1} dx \tag{7}$$

the coefficients of the polynomial satisfy the matrix equation

$$T\underline{a} = -\underline{b} \tag{8}$$

where $T_{i,j} = M_{i+j-1}$, and $b_i = M_{i+N-1}$.

End of story? Not quite. In the past, this approach left us with two difficult numerical problems to manage, which now are more manageable with Mathematica or Maple. The matrix T quickly becomes ill-conditioned, (with $w(x) = 1$, T is part of the Hilbert matrix). The second problem of extracting the roots of a polynomial, given its coefficients, is also a poorly conditioned problem. Even more fundamentally, can we even *know* that the roots x_i of the polynomial are firstly, *real*, and secondly, *lie in the interval* $(0, 1)$, which they need to do if our integration scheme is to make sense?

To make progress on both fronts, we should invoke the classical theory of orthogonal polynomials:

For every $w(x) > 0$ on an interval (a, b) there exists a family of orthogonal polynomials $\phi_i(x)$ satisfying

$$\int_a^b w(x)\phi_i(x)\phi_j(x) = 0 \quad \text{for } i \neq j \tag{9}$$

Apart possibly from a constant factor, these polynomials are the polynomials we sought to identify using the method of moments.

If $P_{2N-1}(x)$ is any polynomial of degree $2N-1$, we can write it as $\phi_N(x)q_{N-1}(x) + r_{N-1}(x)$ so that

$$\begin{aligned} & \int_a^b w(x)P_{2N-1}(x)dx \\ = & \int_a^b w(x)\{\phi_N(x)q_{N-1}(x) + r_{N-1}(x)\}dx \\ = & \sum_{i=1}^{N} w_i \ \{\phi_N(x_i)q_{N-1}(x_i) + r_{N-1}(x_i)\} \end{aligned} \tag{10}$$

If the x_i are chosen to be the zeros of $\phi_N(x)$ we automatically obtain

$$\int_a^b w(x)\phi_n(x)q_{N-1}(x)dx = 0 = \sum_i w_i\phi_N(x_i)q_{N-1}(x_i) \tag{11}$$

and are free to choose the weights w_i, so that the integrals

$$\int_a^b w(x)r_k(x)dx = \sum_{i=1}^{N} w_i r_k(x_i) \tag{12}$$

are correctly evaluated for

$r_k(x) = x^{k-1}$, or $\phi_{k-1}(x)$, for $k = 1 \ldots N$.

If the polynomials are orthonormal, satisfying

$$\int_a^b w(x)\phi_i(x)\phi_j(x)dx = \delta_{i,j} \tag{13}$$

the weights can be obtained from the Christoffel-Darboux identity

$$w_i = [\sum_{j=0}^{N-1} \phi_j(x_i))^2]^{-1} \tag{14}$$

which shows that the weights are positive, as we might expect for a reliable integration scheme.

These polynomials satisfy a recurrence relation of the form

$$\phi_{n+1}(x) = (a_n x + b_n)\ \phi_n(x) + c_n\ \phi_{n-1}(x) \qquad (15)$$

If the coefficients of the recurrence relation are available in closed form, Golub and Welsch(1969) identified a route to the roots x_i:

$$x\underline{\phi} = T\underline{\phi} + \underline{r} \qquad (16)$$

where

$$T = \begin{bmatrix} -b_0/a_0 & 1/a_0 & & & \\ -c_1/a_1 & -b_1/a_1 & 1/a_1 & & \\ & -c_2/a_2 & -b_2/a_2 & 1/a_2 & \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & -c_n/a_n & -b_n/a_n \end{bmatrix},$$

and $\underline{\phi} = [\phi_0, \phi_1, \ldots, \phi_n]^T$, $\underline{r} = [0, 0, \ldots, \phi_{n+1}(x)/a_n]^T$.

Solving the eigenvalue problem gives the zeros of ϕ_{n+1}. These methods were plagued by loss of accuracy in the computation, but help is now available via packages such as Maple or Mathematica, which have the ability to compute polynomials and their roots for weight functions of the form $(x-a)^{-\alpha}(b-x)^{-\beta}$. Using this family, it should be possible to match a singularity at either end of the integration for many weight functions $w(x)$ of interest by simply selecting the appropriate $W(x)$ from the family, and computing its x_i and w_i. A reliable integration rule for $w(x)$ will be

$$\int_a^b w(x)f(x)dx = \sum_{i=1}^{n} w_i (w(x_i)/W(x_i)) f(x_i) \qquad (17)$$

If a weight function $w(x)$ cannot be matched with those available in Maple, alternate procedures outlined by Gautschi (1968), involving Richardson extrapolation of numerical integrals, are available.

The mention of Richardson extrapolation leads us naturally to next consider Romberg integration and Newton-Cotes integration rules.

I believe it is important to give some time to the idea of Romberg integration in our first year courses for the following reasons:

It introduces students to the idea of Richardson extrapolation in numerical processes. If $T(h)$, the trapezoidal estimate of an integral using a step size h, approximates the true answer T with an error relation

$$T(h) = T + a_2 h^2 + a_4 h^4 + a_6 h^6 + \ldots \tag{18}$$

we may eliminate the error term proportional to h^2 by Richardson extrapolation.

$S(h/2) = (4 * T(h/2) - T(h))/3$ has a leading error term proportional to h^4. The fact that if we combine the rules symbolically, S(h/2) is the familiar Simpson's rule, and applying Richardson extrapolation to S(h) to eliminate the h^4 error term leads on to the not-so-familiar Weddell's 3/8 rule gives a unifying view of the Newton-Cotes rules.

The fact that we obtain three separate estimates of the integral from $T(h)$, $T(h/2)$, and $S(h/2)$, without any additional function evaluations, leads on naturally to the ideas of accuracy termination criteria, and the adaptive refinement techniques used in various automatic integration packages.

Starting from the elementary trapezoidal rule, successive trapezoidal approximations to $\int_{-1}^{+1} f(x)dx$ yield

$$T_1 = [\, f(-1) + f(1)] \tag{19}$$

$$\begin{aligned} T_2 &= 0.5 * [\, f(-1) + 2f(0) + f(1)] \\ &= 0.5T_1 + f(0) \end{aligned} \tag{20}$$

$$\begin{aligned} T_3 &= 0.25 * [\, f(-1) + 2f(-0.5) + 2f(0 + 2f(0.5) + f(1)] \\ &= 0.5T_2 + 0.5[f(-0.5) + f(0.5)] \end{aligned} \tag{21}$$

These successive estimates exhibit the **nested** property, which means that the function values used to compute the previous approximation to the integral are used in the current step. This would not be true in Gaussian integration for example, where the ordinates in a three point rule are completely distinct, and in fact interlace, the ordinates of the four point rule.

We can also also see immediately that

$$\begin{aligned} S_2 &= [4T_2 - T_1]/3 \\ &= [f(-1) + 4f(0) + f(1)]/3 \end{aligned} \tag{22}$$

yields the elementary Simpson rule, as previously remarked.

Another notable one dimensional family are the **Curtis-Clenshaw** family. They share the **nested** property with the Newton-Cotes rules, and behave like the Gaussian rules in being able to integrate increasing degrees of polynomials exactly. The key underlying idea is to approximate the function that we want to integrate by an interpolant at a sequence of ordinates, and then integrate the interpolant exactly. The simplest way to derive the Curtis-Clenshaw family is to change variables:

$$\int_{-1}^{+1} f(x)dx = \int_0^{\pi} f(cos\theta)\ sin\theta\ d\theta \tag{23}$$

We can approximate $f(cos\theta)$ by its Fourier series

$$f(cos\theta) = \frac{1}{2}\, a_0 + \sum_{k=1}^{\infty} a_k\ cos(k\theta) \tag{24}$$

where

$$a_k = \frac{2}{\pi} \int_0^{\pi} f(cos\theta)\ cos(k\theta)\ d\theta \tag{25}$$

so that now we can do the integrations in equation 23 exactly, term by term for each term in equation 24.

What we seem to have done is to replace the original problem of integrating $f(x)$ by the problem of evaluating the integrals in equation 25 for the coefficients a_k.

So now we would like to revisit the trapezoidal rule:

$$\int_0^{\pi} f(cos\theta)\, d\theta \approx \frac{\pi}{2N}[f(cos\theta_0)cos\ k\theta_0 + 2\sum_{j=0}^{N-1} f(cos\theta_j)cos\ k\theta_j + f(cos\theta_N)cos\ k\theta_N] \quad (26)$$

for $k = 0, 1, 2\ldots$ and $\theta_j = \frac{j\pi}{N}$. These summations can be evaluated by the Fast Fourier Transform (FFT or DCT) and the integrals $\int_0^{\pi} cos\ k\theta\ sin\theta\ d\theta$ are readily evaluated.

If we look at the points where the function being integrated has to be evaluated, we see that the values $f(cos\theta_j)$, $\theta_j = \frac{j\pi}{N}$ are nested, so that we can change from $N \rightarrow 2N$ just by evaluating $f(cos\theta)$ at the midpoints $\theta = \frac{(2j+1)\pi}{2N}$.

Thus by precomputing the weights w_j, the Curtis-Clenshaw integration rule can be written:

$$\int_{-1}^{1} f(x)dx = \sum_{j=0}^{N} w_j f(x_j), \quad x_j = cos(\frac{j\pi}{N}) \quad (27)$$

The Curtis-Clenshaw approach may have an advantage when attempting to integrate functions with an integrable singularity, or with weight functions other than 1. Suppose

$$f(x) \approx s(x) \quad (28)$$

where s(x) is integrable with a singularity.

$$\begin{aligned}\int_{-1}^{1} f(x)dx &= \int_{-1}^{1} \frac{f(x)}{s(x)}\ s(x)\ dx \\ &= \int_{-1}^{1} g(x)\ s(x)\ dx \\ &= \int_{0}^{\pi} g(cos\theta)\ s(cos\theta)\ sin\theta\ d\theta\end{aligned} \tag{29}$$

where $g(x)$ no longer has a singularity.

We can then create the integration rule

$$\int_{-1}^{1} f(x)\ dx = \sum_{j=0}^{N} \mu_j\ g(x_j) \tag{30}$$

where now we have to precompute the weights μ_j by evaluating the integrals

$$\int_{0}^{\pi} cos\ k\theta\ s(cos\theta)\ sin\theta\ d\theta,$$

which either we may be able to do analytically, or we can evaluate once, to high accuracy, numerically.

Numerical evaluation of multidimensional integrals, which are required in areas such as statistical physics and financial mathematics, are becoming increasingly common as the power of our computing platforms continues to expand. Computations yielding UQ (uncertainty quantification) are appearing increasingly in the literature.

For low dimensional integrals, we might be tempted to use a tensor product rule. For example, in two dimensions we would have

$$\int_0^1 dx_2 \int_0^1 dx_1 f(x_1, x_2) = \sum_i \sum_j w_i w_j f(x_i, x_j) \tag{31}$$

If N function evaluations are required for the one dimensional rule, N^2 function evaluations are required in 2 dimensions. However, applying the same approach to a 20 dimensional integral, using just a 3 point Gaussian rule in each variable, would require $3^{20} \approx 10^9$ function evaluations, the so called "curse of dimensionality" . In the past, we have often turned to the Monte Carlo estimate of the M dimensional integral

$$I(f) \approx (1/N) \sum_{i=1}^{N} f(x_{1,i}, x_{2,i}, \ldots, x_{M,i}) \tag{32}$$

where each of the N points ($x_{j,i}$ $j = 1 \ldots M$) in M-dimensional space is selected using a psuedo-random number generator.

The difference between the two approaches is stark as the dimensionality of the integral increases. The behaviour of the worst case error is revealing. If we imagine that $F(x_1, \ldots, x_N) = g(x_1)$, i.e. that the multidimensional function is just a function of one variable, the error after 10^9 function evaluations using the tensor product rule on a 20 dimensional problem is no better than it could be with 3 function evaluations - a depressing result.

For the Monte Carlo method, we have different types of error results:

$$E(F) \approx \frac{1}{N} \sum_{i=1}^{N} F(\mathbf{x}_i) \tag{33}$$

with an error term $\sim \sigma(F)\ N^{-\frac{1}{2}}$ where $\sigma(F)$ is the variance of F over the volume of integration. Knowing the functional dependence of the error on the number of Monte Carlo sample points offers the possibility of monitoring the progress of the calculation and terminating when the calculation when the error reaches a pre-set input value.

Such Monte Carlo estimates have their own problems:

1. Estimates of the integral from different random sequences are different.

2. Generating the random sequences may be difficult for complex shapes

3. The error dependence $\sim N^{-\frac{1}{2}}$ may be unacceptably slow.

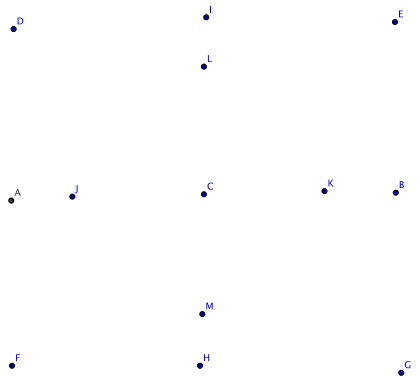
However, Monte Carlo methods have become more attractive in the age of multi-core computing, because each estimate of the function $F(\mathbf{x_j})$ at each point $\mathbf{x_j}$ can be evaluated independently of the rest, simultaneously, on however many cores are available. Of course, the same is true of tensor product integration, but the number of points in the tensor product case may quickly become prohibitive as the dimension of the sample space rises.

These days there is a thriving industry based on an idea of Smolyak(1963) which combines together the notions of interpolation and integration, including the work of our own Ian Sloan eg Sloan and Smith(1978) and much subsequent work. When a multi-dimensional function is approximated by an interpolatory polynomial, each term in the interpolatory polynomial can be integrated analytically. Combining the contribution of the function value $f(\mathbf{x_i})$ to each coefficient of the interpolatory polynomial with the result obtained from integrating the corresponding term in the polynomial allows us to again generate a rule of the type

$$I(f) \approx \sum_{i=1}^{N} w_i \, f(\mathbf{x}_i) \tag{34}$$

where now both the points $\mathbf{x_i}$ and the weights w_i are determined by the choice of the interpolating polynomial.

One such scheme, based on Curtis-Clenshaw integration is described by Novak and Ritter(1996), and illustrated below for a two dimensional integral.



This is based on a hierachy of Curtis-Clenshaw rules, labelled $1, 2, 3 \ldots$. The first is just the the single point midpoint rule. The second contains the end points of the interval, plus the midpoint. The third in the sequence contains the points $cos(\frac{j\pi}{4}), j = 0 \ldots 4$. The fourth contains $cos(\frac{j\pi}{8}), j = 0 \ldots 8$.

U^{i_1} represents integration using one of these rules to integrate with respect to variable x_1.

U^{i_2} represents integration using one of these rules to integrate with respect to variable x_2.

$U^{i_1} \otimes U^{i_2} \otimes \ldots U^{i_d}$ represents a tensor product integration rule for a function $f(x_1, x_2, \ldots x_d)$ where the number of integration points in dimension k is indicated by i_k.

A composite rule $A(q, d)$ defined by Novak and Ritter takes the form

$$A(q,d) = \sum_{|i|=q-d+1}^{q} (-1)^{q-|i|} \binom{d-1}{q-|i|} (U^{i_1} \otimes \ldots U^{i_d}) \qquad (35)$$

where $|i| = i_1 + i_2 + \ldots i_d$. For example,

$A(4,2) = -(U^1 \otimes U^2) - (U^2 \otimes U^1) + (U^1 \otimes U^3) + (U^2 \otimes U^2) + (U^3 \otimes U^1)$

The key as to why $A(q,d)$ does not generate integration points at the same rate as a single tensor product rule is the use of nested integration rules which means that the same points are used repeatedly. The following table illustrates the growth in the number of evaluation points with dimension d and q, with 3^d listed for comparison:

d	q	$A(q,d)$	3^d
2	4	13	9
2	5	29	9
2	6	65	9
2	7	145	9
5	9	801	243
5	10	2433	243
10	14	8801	59049
10	15	41265	59049
15	19	40001	1.43e+7
15	20	261497	1.43e+7
20	24	120401	3.49e+9
20	25	1.02e+6	3.49e+9
30	34	582801	2.06e+14

To use $A(q,d)$ to integrate a function we have to combine all the weights associated with each particular point, so that the integrand is evaluated once only at each point.

The fact that the signs of the components of $A(q,d)$ are alternating gives rise to some points with negative weights, for example the weight associated with the midpoint in $A(4,2)$. Such a situation never arises in tensor products of Gaussian rules.

If a function varies more rapidly in one particular direction k, it would be possible to vary the sequence of Curtis-Clenshaw points from the standard $(1,3,5,9,17\ldots)$ sequence to $(1,9,17\ldots)$ so that more points are sampled in that particular direction.

References:

Babuska, I., Nobile, F., and Tempone, R. (2010) A stochastic collocation method for elliptic partial differential equations with random input data. SIAM Review 52,2, 317-356

Clenshaw, C.W. and Curtis, A.R. (1960) A method for numerical integration on an automatic computer. Numer. Math. 2, 197-205.

Davis, P.J. and Rabinowitz, P. (1975) Methods of Numerical Integration. Academic Press. New York, San Francisco, London.

Gautschi, W. (1968) Construction of Gauss-Christoffel quadrature formulas. Math Comp. 22, 251-270.

Golub, G.H. and Welsch, J.H.(1969) Calculation of Gauss quadrature rules. Math.Comp. 23, 221-230.

Novak, E. and Ritter, K. (1996) High dimensional integration of smooth functions over cubes. Numer. Math. 75, 79-97.

Sloan, I.H. and Smith, W.E. (1978) Product integration with Clenshaw-Curtis and related points. Numer. Math. 30, 415-428

Smolyak, S.A.(1963) Quadrature and interpolation formulas for tensor products of certain classes of functions. Soviet Math. Dokl. 4, 240-243.