NAG Toolbox for MATLAB

Chapter Introduction

D01 – Quadrature

Contents

1 Scope of the Chapter

This chapter provides functions for the numerical evaluation of definite integrals in one or more dimensions and for evaluating weights and abscissae of integration rules.

2 Background to the Problems

The functions in this chapter are designed to estimate:

(a) the value of a one-dimensional definite integral of the form

$$\int_a^b f(x)\, dx \tag{1}$$

where $f(x)$ is defined by you, either at a set of points $(x_i,f(x_i))$, for $i = 1,2,\ldots,n$, where $a = x_1 < x_2 < \cdots < x_n = b$, or in the form of a function; and the limits of integration a,b may be finite or infinite.

Some methods are specially designed for integrands of the form

$$f(x) = w(x)g(x) \tag{2}$$

which contain a factor $w(x)$, called the weight-function, of a specific form. These methods take full account of any peculiar behaviour attributable to the $w(x)$ factor.

(b) the values of the one-dimensional indefinite integrals arising from (1) where the ranges of integration are interior to the interval $[a,b]$.

(c) the value of a multi-dimensional definite integral of the form

$$\int_{R_n} f(x_1,x_2,\ldots,x_n)\, dx_n \cdots dx_2 dx_1 \tag{3}$$

where $f(x_1,x_2,\ldots,x_n)$ is a function defined by you and R_n is some region of n-dimensional space.

The simplest form of R_n is the n-rectangle defined by

$$a_i \le x_i \le b_i, \qquad i = 1,2,\ldots,n \tag{4}$$

where a_i and b_i are constants. When a_i and b_i are functions of x_j $(j < i)$, the region can easily be transformed to the rectangular form (see page 266 of Davis and Rabinowitz 1975). Some of the methods described incorporate the transformation procedure.

2.1 One-dimensional Integrals

To estimate the value of a one-dimensional integral, a quadrature rule uses an approximation in the form of a weighted sum of integrand values, i.e.,

$$\int_a^b f(x)\, dx \simeq \sum_{i=1}^N w_i f(x_i). \tag{5}$$

The points x_i within the interval $[a,b]$ are known as the abscissae, and the w_i are known as the weights.

More generally, if the integrand has the form (2), the corresponding formula is

$$\int_a^b w(x)g(x)\, dx \simeq \sum_{i=1}^N w_i g(x_i). \tag{6}$$

If the integrand is known only at a fixed set of points, these points must be used as the abscissae, and the weighted sum is calculated using finite-difference methods. However, if the functional form of the integrand is known, so that its value at any abscissa is easily obtained, then a wide variety of quadrature rules are available, each characterised by its choice of abscissae and the corresponding weights.

The appropriate rule to use will depend on the interval $[a,b]$ – whether finite or otherwise – and on the form of any $w(x)$ factor in the integrand. A suitable value of N depends on the general behaviour of $f(x)$; or of $g(x)$, if there is a $w(x)$ factor present.

Among possible rules, we mention particularly the Gaussian formulae, which employ a distribution of abscissae which is optimal for $f(x)$ or $g(x)$ of polynomial form.

The choice of basic rules constitutes one of the principles on which methods for one-dimensional integrals may be classified. The other major basis of classification is the implementation strategy, of which some types are now presented.

(a) Single rule evaluation procedures

A fixed number of abscissae, N, is used. This number and the particular rule chosen uniquely determine the weights and abscissae. No estimate is made of the accuracy of the result.

(b) Automatic procedures

The number of abscissae, N, within $[a, b]$ is gradually increased until consistency is achieved to within a level of accuracy (absolute or relative) requested by you. There are essentially two ways of doing this; hybrid forms of these two methods are also possible:

(i) whole interval procedures (non-adaptive)

A series of rules using increasing values of N are successively applied over the whole interval $[a, b]$. It is clearly more economical if abscissae already used for a lower value of N can be used again as part of a higher-order formula. This principle is known as **optimal extension**. There is no overlap between the abscissae used in Gaussian formulae of different orders. However, the Kronrod formulae are designed to give an optimal $(2N + 1)$-point formula by adding $(N + 1)$ points to an N-point Gauss formula. Further extensions have been developed by Patterson.

(ii) adaptive procedures

The interval $[a, b]$ is repeatedly divided into a number of sub-intervals, and integration rules are applied separately to each sub-interval. Typically, the subdivision process will be carried further in the neighbourhood of a sharp peak in the integrand than where the curve is smooth. Thus, the distribution of abscissae is adapted to the shape of the integrand.

Subdivision raises the problem of what constitutes an acceptable accuracy in each sub-interval. The usual **global acceptability criterion** demands that the sum of the absolute values of the error estimates in the sub-intervals should meet the conditions required of the error over the whole interval. Automatic extrapolation over several levels of subdivision may eliminate the effects of some types of singularities.

An ideal general-purpose method would be an automatic method which could be used for a wide variety of integrands, was efficient (i.e., required the use of as few abscissae as possible), and was reliable (i.e., always gave results to within the requested accuracy). Complete reliability is unobtainable, and generally higher reliability is obtained at the expense of efficiency, and vice versa. **It must therefore be emphasised that the automatic functions in this chapter cannot be assumed to be 100% reliable. In general, however, the reliability is very high.**

2.2 Multi-dimensional Integrals

A distinction must be made between cases of moderately low dimensionality (say, up to 4 or 5 dimensions), and those of higher dimensionality. Where the number of dimensions is limited, a one-dimensional method may be applied to each dimension, according to some suitable strategy, and high accuracy may be obtainable (using product rules). However, the number of integrand evaluations rises very rapidly with the number of dimensions, so that the accuracy obtainable with an acceptable amount of computational labour is limited; for example a product of 3-point rules in 20 dimensions would require more than 10^9 integrand evaluations. Special techniques such as the Monte Carlo methods can be used to deal with high dimensions.

(a) Products of one-dimensional rules

Using a two-dimensional integral as an example, we have

$$\int_{a_1}^{b_1} \int_{a_2}^{b_2} f(x,y)\, dy\, dx \simeq \sum_{i=1}^{N} w_i \left[\int_{a_2}^{b_2} f(x_i, y)\, dy \right] \tag{7}$$

$$\int_{a_1}^{b_1} \int_{a_2}^{b_2} f(x,y)\,dy\,dx \simeq \sum_{i=1}^{N} \sum_{j=1}^{N} w_i v_j f\left(x_i, y_j\right) \tag{8}$$

where (w_i, x_i) and (v_i, y_i) are the weights and abscissae of the rules used in the respective dimensions.

A different one-dimensional rule may be used for each dimension, as appropriate to the range and any weight function present, and a different strategy may be used, as appropriate to the integrand behaviour as a function of each independent variable.

For a rule-evaluation strategy in all dimensions, the formula (8) is applied in a straightforward manner. For automatic strategies (i.e., attempting to attain a requested accuracy), there is a problem in deciding what accuracy must be requested in the inner integral(s). Reference to formula (7) shows that the presence of a limited but random error in the y-integration for different values of x_i can produce a 'jagged' function of x, which may be difficult to integrate to the desired accuracy and for this reason products of automatic one-dimensional functions should be used with caution (see Lyness 1983).

(b) Monte Carlo methods

These are based on estimating the mean value of the integrand sampled at points chosen from an appropriate statistical distribution function. Usually a variance reducing procedure is incorporated to combat the fundamentally slow rate of convergence of the rudimentary form of the technique. These methods can be effective by comparison with alternative methods when the integrand contains singularities or is erratic in some way, but they are of quite limited accuracy.

(c) Number theoretic methods

These are based on the work of Korobov and Conroy and operate by exploiting implicitly the properties of the Fourier expansion of the integrand. Special rules, constructed from so-called optimal coefficients, give a particularly uniform distribution of the points throughout n-dimensional space and from their number theoretic properties minimize the error on a prescribed class of integrals. The method can be combined with the Monte Carlo procedure.

(d) Sag–Szekeres method

By transformation this method seeks to induce properties into the integrand which make it accurately integrable by the trapezoidal rule. The transformation also allows effective control over the number of integrand evaluations.

(e) Automatic adaptive procedures

An automatic adaptive strategy in several dimensions normally involves division of the region into subregions, concentrating the divisions in those parts of the region where the integrand is worst behaved. It is difficult to arrange with any generality for variable limits in the inner integral(s). For this reason, some methods use a region where all the limits are constants; this is called a hyper-rectangle. Integrals over regions defined by variable or infinite limits may be handled by transformation to a hyper-rectangle. Integrals over regions so irregular that such a transformation is not feasible may be handled by surrounding the region by an appropriate hyper-rectangle and defining the integrand to be zero outside the desired region. Such a technique should always be followed by a Monte Carlo method for integration.

The method used locally in each subregion produced by the adaptive subdivision process is usually one of three types: Monte Carlo, number theoretic or deterministic. Deterministic methods are usually the most rapidly convergent but are often expensive to use for high dimensionality and not as robust as the other techniques.

3 Recommendations on Choice and Use of Available Functions

The following three sub-sections consider in turn functions for: one-dimensional integrals over a finite interval, and over a semi-infinite or an infinite interval; and multi-dimensional integrals. Within each sub-section, functions are classified by the type of method, which ranges from simple rule evaluation to automatic adaptive algorithms. The recommendations apply particularly when the primary objective is simply to compute the value of one or more integrals, and in these cases the automatic adaptive functions are generally the most convenient and reliable, although also the most expensive in computing time.

Note however that in some circumstances it may be counter-productive to use an automatic function. If the results of the quadrature are to be used in turn as input to a further computation (e.g., an 'outer' quadrature or an optimization problem), then this further computation may be adversely affected by the 'jagged performance profile' of an automatic function; a simple rule-evaluation function may provide much better overall performance. For further guidance, the article by Lyness 1983 is recommended.

3.1 One-dimensional Integrals over a Finite Interval

(a) Integrand defined at a set of points

If $f(x)$ is defined numerically at four or more points, then the Gill–Miller finite difference method (d01ga) should be used. The interval of integration is taken to coincide with the range of x values of the points supplied. It is in the nature of this problem that any function may be unreliable. In order to check results independently and so as to provide an alternative technique you may fit the integrand by Chebyshev-series using e02ad and then use functions e02aj and e02ak to evaluate its integral (which need not be restricted to the range of the integration points, as is the case for d01ga). A further alternative is to fit a cubic spline to the data using e02ba and then to evaluate its integral using e02bd.

(b) Integrand defined as a function

If the functional form of $f(x)$ is known, then one of the following approaches should be taken. They are arranged in the order from most specific to most general, hence the first applicable procedure in the list will be the most efficient. **However, if you do not wish to make any assumptions about the integrand, the most reliable functions to use will be d01aj (or d01at) and d01ah, although these will in general be less efficient for simple integrals.**

(i) Rule-evaluation functions

If $f(x)$ is known to be sufficiently well behaved (more precisely, can be closely approximated by a polynomial of moderate degree), a Gaussian function with a suitable number of abscissae may be used.

d01ba may be used if it is not required to examine the weights and abscissae.

d01bb or d01bc with d01fb may be used if it is required to examine the weights and abscissae.

d01bb is faster and more accurate, whereas d01bc is more general.

If $f(x)$ is well behaved, apart from a weight-function of the form

$$\left|x - \frac{a+b}{2}\right|^c \quad \text{or} \quad (b-x)^c(x-a)^d,$$

d01bc with d01fb may be used.

(ii) Automatic whole-interval functions

If $f(x)$ is reasonably smooth, and the required accuracy is not too high, the automatic whole-interval functions d01ar or d01bd may be used. d01ar incorporates high-order extensions of the Kronrod rule and is the only function which can also be used for indefinite integration.

(iii) Automatic adaptive functions

Firstly, several functions are available for integrands of the form $w(x)g(x)$ where $g(x)$ is a 'smooth' function (i.e., has no singularities, sharp peaks or violent oscillations in the interval of integration) and $w(x)$ is a weight function of one of the following forms.

1. if $w(x) = (b-x)^\alpha(x-a)^\beta(\log(b-x))^k(\log(x-a))^l$, where $k, l = 0$ or 1, $\alpha, \beta > -1$: use d01ap;
2. if $w(x) = \frac{1}{x-c}$: use d01aq (this integral is called the Hilbert transform of g);
3. if $w(x) = \cos(\omega x)$ or $\sin(\omega x)$: use d01an (this function can also handle certain types of singularities in $g(x)$).

Secondly, there are some functions for general $f(x)$. If $f(x)$ is known to be free of singularities, though it may be oscillatory, d01ak or d01au may be used.

The most powerful of the finite interval integration functions are d01aj and d01at, which can cope with singularities of several types, and d01ah. They may be used if none of the more specific situations described above applies. d01ah is likely to be more efficient, whereas d01aj and d01at are somewhat more reliable, particularly where the integrand has singularities other than at an end point, or has discontinuities or cusps, and is therefore recommended where the integrand is known to be badly behaved, or where its nature is completely unknown. It may sometimes be useful to use both functions as a check.

Most of the functions in this chapter require you to supply a function or (sub)program to evaluate the integrand at a single point. d01at and d01au use the same methods as d01aj and d01ak respectively, but have a different user interface which can result in faster execution, especially on vector-processing machines (see Gladwell 1986). They require you to provide a (sub)program to return an array of values of the integrand at each of an array of points. This reduces the overhead of function calls, avoids repetition of computations common to each of the integrand evaluations, and offers greater scope for vectorization of your code.

If $f(x)$ has singularities of certain types, discontinuities or sharp peaks **occurring at known points**, the integral should be evaluated separately over each of the subranges or d01al may be used.

3.2 One-dimensional Integrals over a Semi-infinite or Infinite Interval

(a) Integrand defined at a set of points

If $f(x)$ is defined numerically at four or more points, and the portion of the integral lying outside the range of the points supplied may be neglected, then the Gill–Miller finite difference method, d01ga, should be used.

(b) Integrand defined as a function

(i) Rule evaluation functions

If $f(x)$ behaves approximately like a polynomial in x, apart from a weight function of the form:

1. $e^{-\beta x}, \beta > 0$ (semi-infinite interval, lower limit finite); or
2. $e^{-\beta x}, \beta < 0$ (semi-infinite interval, upper limit finite); or
3. $e^{-\beta(x-\alpha)^2}, \beta > 0$ (infinite interval),

or if $f(x)$ behaves approximately like a polynomial in $(x+b)^{-1}$ (semi-infinite range), then the Gaussian functions may be used.

d01ba may be used if it is not required to examine the weights and abscissae.

d01bb or d01bc with d01fb may be used if it is required to examine the weights and abscissae.

d01bb is faster and more accurate, whereas d01bc is more general.

(ii) Automatic adaptive functions

d01am may be used, except for integrands which decay slowly towards an infinite end point, and oscillate in sign over the entire range. For this class, it may be possible to calculate the integral by integrating between the zeros and invoking some extrapolation process (see c06ba).

d01as may be used for integrals involving weight functions of the form $\cos(\omega x)$ and $\sin(\omega x)$ over a semi-infinite interval (lower limit finite).

The following alternative procedures are mentioned for completeness, though their use will rarely be necessary.

1. If the integrand decays rapidly towards an infinite end point, a finite cut-off may be chosen, and the finite range methods applied.

2. If the only irregularities occur in the finite part (apart from a singularity at the finite limit, with which d01am can cope), the range may be divided, with d01am used on the infinite part.

3. A transformation to finite range may be employed,e.g.,

$$x = \frac{1-t}{t} \qquad \text{or} \qquad x = -\log_e t$$

will transform $(0,\infty)$ to $(1,0)$ while for infinite ranges we have

$$\int_{-\infty}^{\infty} f(x)\,dx = \int_0^{\infty} (f(x) + f(-x))\,dx.$$

If the integrand behaves badly on $(-\infty,0)$ and well on $(0,\infty)$ or vice versa it is better to compute it as $\int_{-\infty}^{0} f(x)\,dx + \int_0^{\infty} f(x)\,dx$. This saves computing unnecessary function values in the semi-infinite range where the function is well behaved.

3.3 Multi-dimensional Integrals

A number of techniques are available in this area and the choice depends to a large extent on the dimension and the required accuracy. It can be advantageous to use more than one technique as a confirmation of accuracy particularly for high-dimensional integrations. Many of the functions incorporate the transformation procedure (sub)program **region** (a) which allows general product regions to be easily dealt with in terms of conversion to the standard n-cube region.

(a) Products of one-dimensional rules (suitable for up to about 5 dimensions)

If $f(x_1, x_2, \ldots, x_n)$ is known to be a sufficiently well behaved function of each variable x_i, apart possibly from weight functions of the types provided, a product of Gaussian rules may be used. These are provided by d01bb or d01bc with d01fb. Rules for finite, semi-infinite and infinite ranges are included.

For two-dimensional integrals only, unless the integrand is very badly behaved, the automatic whole-interval product procedure of d01da may be used. The limits of the inner integral may be user-specified functions of the outer variable. Infinite limits may be handled by transformation (see Section 3.2); end point singularities introduced by transformation should not be troublesome, as the integrand value will not be required on the boundary of the region.

If none of these functions proves suitable and convenient, the one-dimensional functions may be used recursively. For example, the two-dimensional integral

$$I = \int_{a_1}^{b_1} \int_{a_2}^{b_2} f(x,y)\,dy\,dx$$

may be expressed as

$$I = \int_{a_1}^{b_1} F(x)\,dx, \qquad \text{where} \qquad F(x) = \int_{a_2}^{b_2} f(x,y)\,dy.$$

The user segment to evaluate $F(x)$ will call the integration function for the y-integration, which will call another user segment for $f(x,y)$ as a function of y (x being effectively a constant). Note that, as Fortran 77 is not a recursive language, **a different library integration function must be used for each dimension**. Apart from this restriction, the following combinations are not permitted: d01aj and d01al, d01an and d01ap, d01ap and d01aq, d01aq and d01an, d01as and d01an, d01as and d01am, d01au and d01at. Otherwise the full range of one-dimensional functions are available, for finite/infinite intervals, constant/variable limits, rule evaluation/automatic strategies etc.

(b) Sag–Szekeres method

Two functions are based on this method.

d01fd is particularly suitable for integrals of very large dimension although the accuracy is generally not high. It allows integration over either the general product region (with built-in transformation to the n-cube) or the n-sphere. Although no error estimate is provided, two adjustable parameters may be varied for checking purposes or may be used to tune the algorithm to particular integrals.

d01ja is also based on the Sag–Szekeres method and integrates over the n-sphere. It uses improved transformations which may be varied according to the behaviour of the integrand. Although it can yield very accurate results it can only practically be employed for dimensions not exceeding 4.

(c) Number Theoretic method

Two functions are based on this method.

d01gc carries out multiple integration using the Korobov–Conroy method over a product region with built-in transformation to the n-cube. A stochastic modification of this method is incorporated hybridising the technique with the Monte Carlo procedure. An error estimate is provided in terms of the statistical standard error. The function includes a number of optimal coefficient rules for up to 20 dimensions; others can be computed using d01gy and d01gz. Like the Sag–Szekeres method it is suitable for large dimensional integrals although the accuracy is not high.

d01gd uses the same method as d01gc, but has a different interface which can result in faster execution, especially on vector-processing machines. You are required to provide two (sub)programs, the first to return an array of values of the integrand at each of an array of points, and the second to evaluate the limits of integration at each of an array of points. This reduces the overhead of function calls, avoids repetitions of computations common to each of the evaluations of the integral and limits of integration, and offers greater scope for vectorization of your code.

(d) A combinatorial extrapolation method

d01pa computes a sequence of approximations and an error estimate to the integral of a function over a multi-dimensional simplex using a combinatorial method with extrapolation.

(e) Automatic functions (d01gb and d01fc)

Both functions are for integrals of the form

$$\int_{a_1}^{b_1}\int_{a_2}^{b_2}\cdots\int_{a_n}^{b_n} f(x_1,x_2,\ldots,x_n)\,dx_n dx_{n-1}\cdots dx_1.$$

d01gb is an adaptive Monte Carlo function. This function is usually slow and not recommended for high-accuracy work. It is a robust function that can often be used for low-accuracy results with highly irregular integrands or when n is large.

d01fc is an adaptive deterministic function. Convergence is fast for well behaved integrands. Highly accurate results can often be obtained for n between 2 and 5, using significantly fewer integrand evaluations than would be required by d01gb. The function will usually work when the integrand is mildly singular and for $n \leq 10$ should be used before d01gb. If it is known in advance that the integrand is highly irregular, it is best to compare results from at least two different functions.

There are many problems for which one or both of the functions will require large amounts of computing time to obtain even moderately accurate results. The amount of computing time is controlled by the number of integrand evaluations allowed by you, and you should set this parameter carefully, with reference to the time available and the accuracy desired.

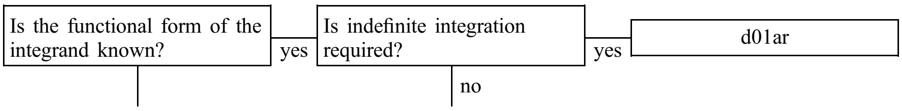
d01ea extends the technique of d01fc to integrate adaptively more than one integrand, that is to calculate the set of integrals

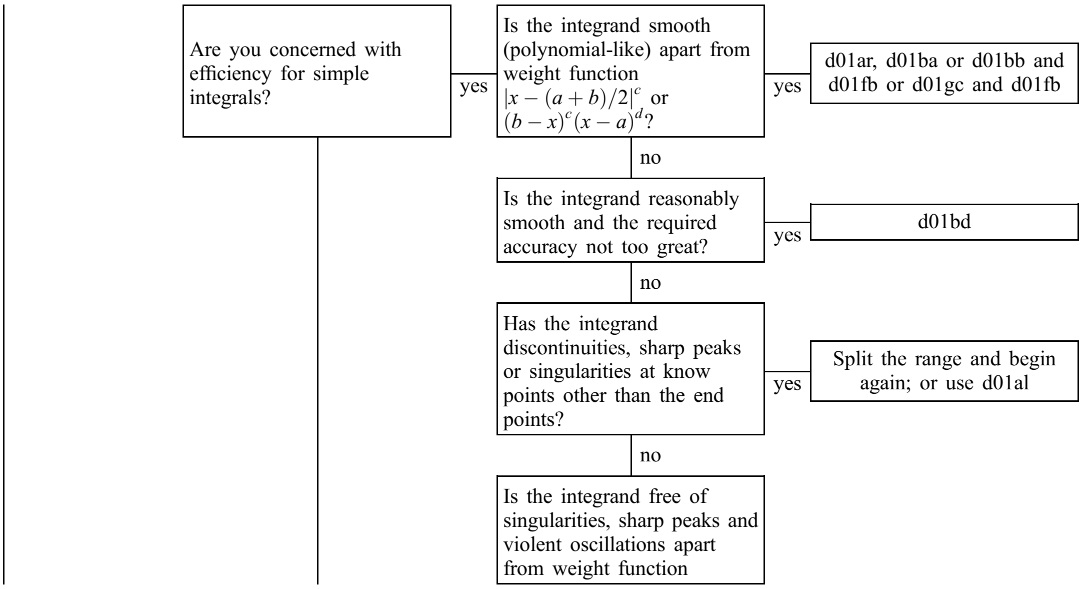
$$\int_{a_1}^{b_1}\int_{a_2}^{b_2}\cdots\int_{a_n}^{b_n} (f_1,f_2,\ldots,f_m)\,dx_n dx_{n-1}\cdots dx_1$$

for a set of similar integrands $f_1,f_2,\ldots,f_m$ where $f_i = f_i(x_1,x_2,\ldots,x_n)$.

4 Decision Trees

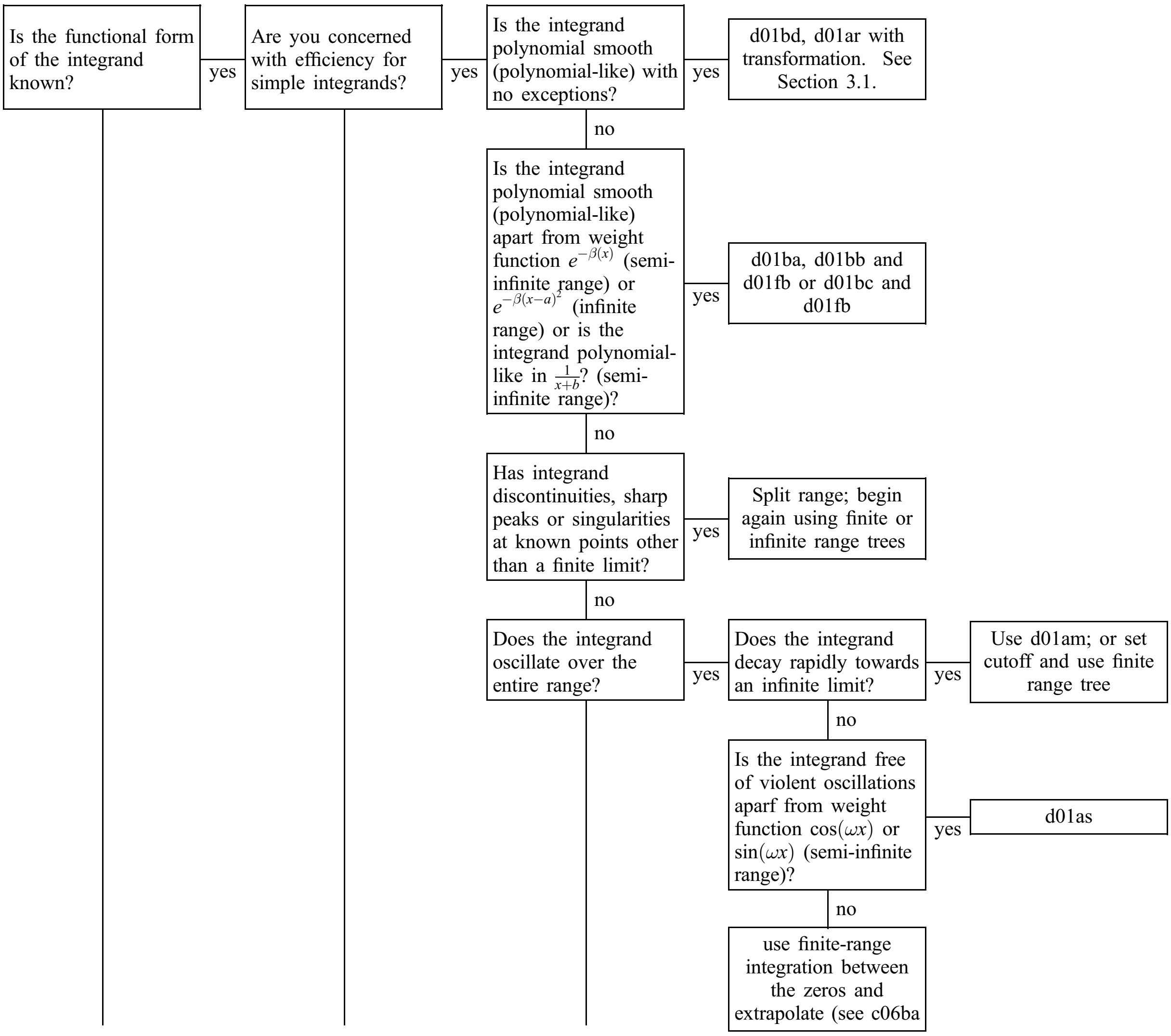
Tree 1: One-dimensional integrals over a finite interval

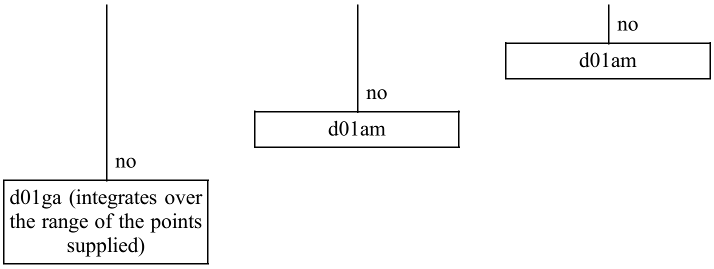

Is the functional form of the integrand known?
yes
Is indefinite integration required?
yes
d01ar
no


Are you concerned with efficiency for simple integrals?
yes
Is the integrand smooth (polynomial-like) apart from weight function $|x-(a+b)/2|^c$ or $(b-x)^c(x-a)^d$?
yes
d01ar, d01ba or d01bb and d01fb or d01gc and d01fb
no
Is the integrand reasonably smooth and the required accuracy not too great?
yes
d01bd
no
Has the integrand discontinuities, sharp peaks or singularities at know points other than the end points?
yes
Split the range and begin again; or use d01al
no
Is the integrand free of singularities, sharp peaks and violent oscillations apart from weight function

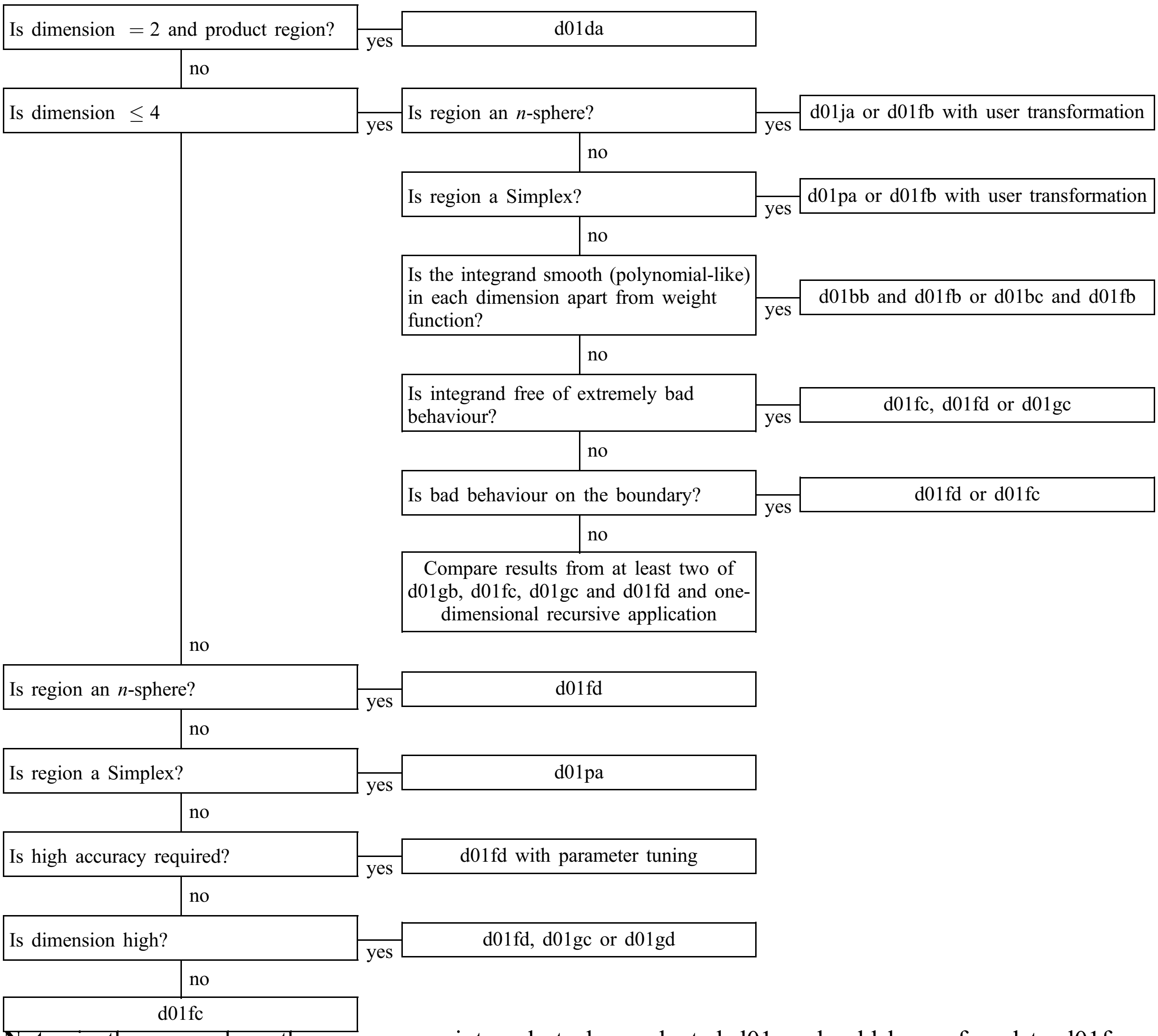
$(b-x)^{\alpha}(x-a)^{\beta}(\log(b-x))^{k}(\log(x-a))^{l}$?yesd01apnoIs the integrand free of singularities, sharp peaks and violent oscillations apart from weight function $\frac{1}{x-c}$?yesd01aqnoIs the integrand free of violent oscillations apart from weight function $\cos(\omega x)$ or $\sin(\omega x)$?yesd01annoIs the integrand free of singularities?yesd01ak or d01aunoIs the integrand free of discontinuities and of singularities except possibly at the end points?yesd01ahnod01aj or d01atnod01aj, d01at or d01ahnod01ga**Note**: d01at and d01au are likely to be more efficient than d01aj and d01ak, which use a more conventional user-interface, consistent with other functions in the chapter.

Tree 2: One-dimensional integrals over a semi-infinite or infinite interval


Is the functional form of the integrand known?
yes
Are you concerned with efficiency for simple integrands?
yes
Is the integrand polynomial smooth (polynomial-like) with no exceptions?
yes
d01bd, d01ar with transformation. See Section 3.1.
no
Is the integrand polynomial smooth (polynomial-like) apart from weight function $e^{-\beta(x)}$ (semi-infinite range) or $e^{-\beta(x-a)^2}$ (infinite range) or is the integrand polynomial-like in $\frac{1}{x+b}$? (semi-infinite range)?
yes
d01ba, d01bb and d01fb or d01bc and d01fb
no
Has integrand discontinuities, sharp peaks or singularities at known points other than a finite limit?
yes
Split range; begin again using finite or infinite range trees
no
Does the integrand oscillate over the entire range?
yes
Does the integrand decay rapidly towards an infinite limit?
yes
Use d01am; or set cutoff and use finite range tree
no
Is the integrand free of violent oscillations aparf from weight function $\cos(\omega x)$ or $\sin(\omega x)$ (semi-infinite range)?
yes
d01as
no
use finite-range integration between the zeros and extrapolate (see c06ba



Tree 3: Multi-dimensional integrals


Is dimension = 2 and product region?
yes
d01da
no
Is dimension ≤ 4
yes
Is region an *n*-sphere?
yes
d01ja or d01fb with user transformation
no
Is region a Simplex?
yes
d01pa or d01fb with user transformation
no
Is the integrand smooth (polynomial-like) in each dimension apart from weight function?
yes
d01bb and d01fb or d01bc and d01fb
no
Is integrand free of extremely bad behaviour?
yes
d01fc, d01fd or d01gc
no
Is bad behaviour on the boundary?
yes
d01fd or d01fc
no
Compare results from at least two of d01gb, d01fc, d01gc and d01fd and one-dimensional recursive application
no
Is region an *n*-sphere?
yes
d01fd
no
Is region a Simplex?
yes
d01pa
no
Is high accuracy required?
yes
d01fd with parameter tuning
no
Is dimension high?
yes
d01fd, d01gc or d01gd
no
d01fc

Note: in the case where there are many integrals to be evaluated d01ea should be preferred to d01fc.

MISSING 'ID' Is likely to be more efficient than d01gc, which uses a more conventional user-interface, consistent with other functions in the chapter.

5 Index

variant of d01gc especially efficient on vector machines .... `d01gd`
over a hyper-rectangle:
adaptive method .... `d01fc`
adaptive method,
multiple integrands .... `d01ea`
Gaussian quadrature rule-evaluation .... `d01fb`
Monte Carlo method .... `d01gb`
over an n-simplex .... `d01pa`
over an n-sphere $(n \geq 4)$,
allowing for badly behaved integrands .... `d01ja`
One-dimensional quadrature:
adaptive integration of a function over a finite interval:
allowing for singularities at user-specified break-points .... `d01al`
method suitable for oscillating functions .... `d01ak`
strategy due to Patterson,
suitable for well-behaved integrands .... `d01ah`
strategy due to Piessens and de Doncker,
allowing for badly behaved integrands .... `d01aj`
variant of d01aj especially efficient on vector machines .... `d01at`
variant of d01ak especially efficient on vector machines .... `d01au`
weight function $1/(x-c)$ Cauchy principal value (Hilbert transform) .... `d01aq`
weight function with end-point singularities of algebraico-logarithmic type .... `d01ap`
weight function $\cos(\omega x)$ or $\sin(\omega x)$ .... `d01an`
adaptive integration of a function over an infinite interval or semi-infinite interval .... `d01am`
adaptive integration of a function over an infinite interval or semi-infinite interval:
weight function $\cos(\omega x)$ or $\sin(\omega x)$ .... `d01as`
Gaussian quadrature rule-evaluation .... `d01ba`
integration of a function defined by data values only,
Gill–Miller method .... `d01ga`
non-adaptive integration over a finite interval .... `d01bd`
non-adaptive integration over a finite interval:
with provision for indefinite integrals also .... `d01ar`
Two-dimensional quadrature over a finite region .... `d01da`
Weights and abscissae for Gaussian qadrature rules:
more general choice of rule,
calculating the weights and abscissae .... `d01bc`
restricted choice of rule,
using pre-computed weights and abscissae .... `d01bb`

6 References

Davis P J and Rabinowitz P 1975 *Methods of Numerical Integration* Academic Press

Gladwell I 1986 Vectorisation of one dimensional quadrature codes *Numerical Integration: Recent Developments, and Applications* (ed P Keast and G Fairweather) 231–238 D Reidel Publishing Company, Holland

Lyness J N 1983 When not to use an automatic quadrature routine *SIAM Rev.* **25** 63–87

Piessens R, de Doncker–Kapenga E, Überhuber C and Kahaner D 1983 *QUADPACK, A Subroutine Package for Automatic Integration* Springer–Verlag

Sobol I M 1974 *The Monte Carlo Method* The University of Chicago Press

Stroud A H 1971 *Approximate Calculation of Multiple Integrals* Prentice–Hall