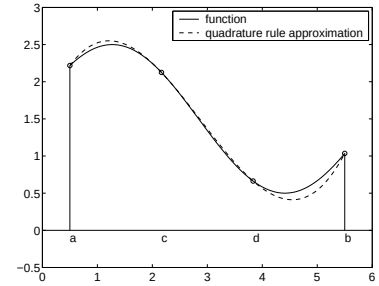
Numerical Integration (Quadrature) -- Introduction

It is often required to calculate the value of $I = \int_a^b f(x)dx$. If f is given in complicated analytic form, or if f is given only in tabular form, the standard mathematical techniques for integration may not be applicable. It this case, we apply numerical techniques to compute an approximation Q to I. A method for computing an approximation Q to I is called a *quadrature rule*.

Many quadrature rules are developed by first approximating $f(x)$ by a simpler function $p(x)$, say a polynomial, then taking $Q = \int_a^b p(x)dx$ as approximation to $I = \int_a^b f(x)dx$. When $p(x)$ is an interpolating polynomial of f, the quadrature rule is called *interpolatory*.



Quadrature rules

Assume we have data $\{(x_i, f(x_i)), i = 0, \cdots, n\}$, with x_i distinct. Then $p_n(x) = \sum_{i=0}^{n} f(x_i)l_i(x)$, where $l_i(x)$ are the Lagrange basis functions for x_i, $i = 0, \cdots, n$, is the polynomial of degree at most n interpolating the above data. The respective quadrature rule is

$$Q = \int_a^b p_n(x)dx = \int_a^b \sum_{i=0}^{n} f(x_i)l_i(x)dx = \sum_{i=0}^{n} f(x_i) \int_a^b l_i(x)dx = \sum_{i=0}^{n} f(x_i)A_i.$$

That is, a quadrature rule is given by a weighted sum of values of f on the data points, with weights $A_i = \int_a^b l_i(x)$. The weights depend only on a, b and $l_i(x)$, $i = 0, \cdots, n$, i.e. a, b and x_i, $i = 0, \cdots, n$. They do not depend on the function f. Thus, they can be computed once and for all for the given set of data points and interval of integration, and reused for any f.

Quadrature rules

Recall the polynomial interpolation error formula

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^{n}(x - x_i).$$

Clearly, if f happens to be a polynomial of degree at most n, the polynomial interpolant $p_n(x)$ based on $n+1$ distinct points is exact. Thus the respective quadrature rule based on $n+1$ distinct points is exact too.
A quadrature rule which is exact for all polynomials of degree k or less on all intervals and not exact on at least one polynomial of degree $k+1$ on at least one interval is said to have *polynomial degree* or *precision* or *degree of accuracy* k.

An interpolatory quadrature rule based on $n+1$ distinct points has polynomial degree at least n.

Quadrature rules

Quadrature rules based on equidistant data points are referred to as *Newton-Cotes* rules.

Quadrature rules that include the endpoints a and b of the interval of integration as data points are called *closed* rules, while
quadrature rules that do not include the endpoints a and b of the interval of integration as data points are called *open* rules.

When an interval is broken into subintervals (called *panels* or *pieces*) and the same quadrature rule is applied to each subinterval, we have a *composite* or *compound* rule. (More on this later.)

Integration is a linear operation. That is, if α and β are scalars,

$$\int(\alpha f + \beta g)dx = \alpha \int f dx + \beta \int g dx.$$

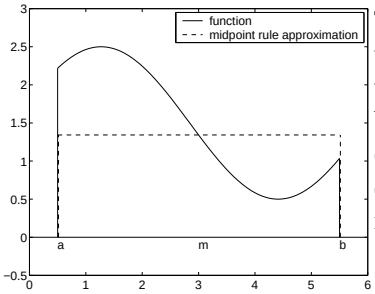
Quadrature rules follow linearity too:

$$Q(\alpha f + \beta g) = \sum_i A_i(\alpha f + \beta g)(x_i) = \sum_i A_i(\alpha f(x_i) + \beta g(x_i))$$
$$= \alpha \sum_i A_i f(x_i) + \beta \sum_i A_i g(x_i) = \alpha Q(f) + \beta Q(g).$$

Midpoint Rule

Assume $f(x)$ is approximated by a constant polynomial $p_0(x) = f(m)$, where $m = \dfrac{a+b}{2}$ is the midpoint of (a, b). Then

$$I = \int_a^b f(x)dx \approx \int_a^b f(m)dx = f(m)\int_a^b dx = f(m)(b-a) = Q_M.$$



That is, I is approximated by the area of the rectangle under the line $y = f(m)$ (which is parallel to the x-axis and matches f at m). If f happens to be a constant polynomial, then the midpoint rule is exact, therefore the midpoint rule has polynomial degree *at least* 0, by construction. It will be shown next that it has polynomial degree 1.

Error in midpoint rule

Recall *Taylor's Theorem*: If f is differentiable in $[a, b]$ and f'' exists in (a, b), then for any x and m in $[a, b]$,

$$f(x) = f(m) + (x-m)f'(m) + \frac{(x-m)^2}{2!}f''(\xi)$$

where ξ depends on x and either $x < \xi < m$ or $m < \xi < x$. (That is, $\xi \in (a, b)$). Thus

$$\int_a^b f(x)dx = \int_a^b f(m)dx + \int_a^b (x-m)f'(m)dx + \int_a^b \frac{(x-m)^2}{2!}f''(\xi)dx$$

$$= f(m)(b-a) + f'(m)\left[\frac{(x-m)^2}{2}\right]_a^b + \int_a^b \frac{(x-m)^2}{2!}f''(\xi)dx.$$

Notice that

$$\left[\frac{(x-m)^2}{2}\right]_a^b = \frac{(b-m)^2}{2} - \frac{(a-m)^2}{2} = 0.$$

Consider now the term $\displaystyle\int_a^b \frac{(x-m)^2}{2!}f''(\xi)dx$ and recall the

Error in midpoint rule

Mean Value Theorem for Integrals (MVTI): If f is continuous in $[a, b]$ and g continuous and one-signed in $[a, b]$ then

$$\int_a^b f(x)g(x)dx = f(\eta)\int_a^b g(x)dx$$

for some $\eta \in [a, b]$. Apply the MVTI to $\displaystyle\int_a^b \frac{(x-m)^2}{2!}f''(\xi)dx$.

Notice that $(x-m)^2$ is one-signed and continuous and assume f'' is continuous $[a, b]$. We have

$$\int_a^b \frac{(x-m)^2}{2!}f''(\xi)dx = \frac{f''(\eta)}{2}\int_a^b (x-m)^2dx$$

$$= \frac{f''(\eta)}{2}\left[\frac{(x-m)^3}{3}\right]_a^b = \frac{f''(\eta)}{2}\frac{(b-a)^3}{12} = f''(\eta)\frac{(b-a)^3}{24}.$$

Hence, the error in the midpoint rule is

$$\int_a^b f(x)dx - (b-a)f(m) = f''(\eta)\frac{(b-a)^3}{24}.$$

Error in midpoint rule

Repeat error in midpoint rule:

$$\int_a^b f(x)dx - (b-a)f(m) = f''(\eta)\frac{(b-a)^3}{24}.$$

Clearly, if f is a polynomial of degree at most 1, the midpoint rule is exact. Therefore, the midpoint rule has polynomial degree at least 1. It can be shown that it has polynomial degree exactly 1. Consider $f(x) = x^2$, $a = 0$ and $b = 1$. It is easy to see that $I = \displaystyle\int_a^b f(x)dx = \int_0^1 x^2dx = \left[\frac{x^3}{3}\right]_0^1 = \frac{1}{3}$ while $Q_M = f(m)(b-a) = f(\frac{1}{2})(1-0) = \frac{1}{4} \neq \frac{1}{3}$. Thus the midpoint rule has polynomial degree exactly 1.

Notes:

- If f is convex (i.e. $f'' > 0$), the midpoint rule underestimates I.
- If f is concave (i.e. $f'' < 0$), the midpoint rule overestimates I.
- If f'' is not continuous $[a, b]$, the error formula does not apply, i.e. we cannot tell how well the rule will approximate the integral.

Composite (Compound) Rules

In order to increase the accuracy of integral approximations, we would need to increase the number of data points.

One way of accommodating lots of data points is to construct a quadrature rule based on a high degree polynomial interpolant and integrating that interpolant.

As will be explained later, this may be fine up to a small number of data points (say at most 10), but it is not a good idea for lots of points.

Another way of accommodating lots of data points is to construct a quadrature rule based on a piecewise polynomial interpolant and integrating that interpolant.

This is equivalent to breaking the interval of integration into subintervals (called *panels* or *pieces*) and applying the same low degree quadrature rule to each subinterval. This technique gives rise to a *composite* or *compound* rule.

Example: Composite Midpoint Rule

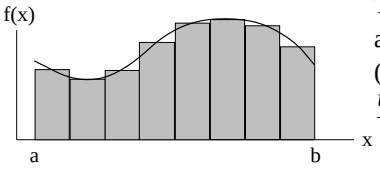
Composite Midpoint Rule

Let s be the number of panels and $h = \frac{b-a}{s}$ be the stepsize. Let $t_i = a + ih$, $i = 0, \cdots, s$, (note: $t_0 = a$ and $t_s = b$) be the knots defining the partition of $[a, b]$ into panels (subintervals). Then

$$I = \int_a^b f(x)dx = \sum_{i=1}^{s} \int_{t_{i-1}}^{t_i} f(x)dx$$

$$\approx \sum_{i=1}^{s} f(\frac{t_{i-1}+t_i}{2})(t_i - t_{i-1}) = h \sum_{i=1}^{s} f(\frac{t_{i-1}+t_i}{2}) = h \sum_{i=1}^{s} f(a + (i - \frac{1}{2})h).$$

This gives the composite midpoint rule. It is equivalent to approximating f by a piecewise constant polynomial defined with respect to the knots t_i, $i = 0, \cdots, s$, and integrating it.



That is, I (the area under the curve of f) is approximated by the sum of the areas of all (sub-)rectangles, each matching f at a midpoint $\frac{t_{i-1}+t_i}{2}$, $i = 1, \cdots, s$, of the panels.

Composite Midpoint Rule

• In the composite midpoint rule the knots are *not* data points.
• The composite midpoint rule uses s function evaluations, where s is the number of panels.

We will now study the accumulation of errors over all panels and come up with an error formula for the compound midpoint rule.

Error in compound midpoint rule:
Error in each panel: From the simple midpoint rule error, assuming f'' is continuous in $[a, b]$, we have $\int_{t_{i-1}}^{t_i} f(x)dx - hf(\frac{t_{i-1}+t_i}{2}) = f''(\eta_i)\frac{h^3}{24}$, where η_i is an unknown point in $[t_{i-1}, t_i]$.

Total error:

$$\int_a^b f(x)dx - h\sum_{i=1}^{s} f(\frac{t_{i-1}+t_i}{2}) = \sum_{i=1}^{s} f''(\eta_i)\frac{h^3}{24} = \frac{h^3}{24}\sum_{i=1}^{s} f''(\eta_i).$$

In order to simplify the expression $\sum_{i=1}^{s} f''(\eta_i)$, we use the

Composite Midpoint Rule

Mean Value Theorem for Sums (MVTS): If f is continuous in $[a, b]$, x_i, $i = 1, \cdots, n$, are in $[a, b]$ and w_i, $i = 1, \cdots, n$, are all of the same sign, then

$$\sum_{i=1}^{n} w_i f(x_i) = f(\eta)\sum_{i=1}^{n} w_i,$$

for some $\eta \in [a, b]$. Apply the MVTS to $\sum_{i=1}^{s} f''(\eta_i)$. Notice that, here, $w_i = 1$ (one-signed), and by assumption f'' is continuous $[a, b]$. We have

$$\sum_{i=1}^{s} f''(\eta_i) = f''(\eta)\sum_{i=1}^{s} 1 = f''(\eta)s.$$

Hence, the error in the composite midpoint rule is

$$\int_a^b f(x)dx - h\sum_{i=1}^{s} f(\frac{t_{i-1}+t_i}{2}) = \frac{h^3}{24}\sum_{i=1}^{s} f''(\eta_i)$$

$$= \frac{h^2}{24}\frac{b-a}{s} f''(\eta)s = \frac{h^2}{24}(b-a)f''(\eta)$$

for some $\eta \in [a, b]$.

Composite Midpoint Rule

Notes:

- The error is proportional to the length $b - a$ of the interval, as expected.
- The error is 0 (i.e. the rule is exact), if f is a polynomial of degree 1 or less (natural consequence of the midpoint rule).
- The error is 0 (i.e. the rule is exact), if f is a piecewise polynomial of degree 1 or less defined with respect to the knots t_i.
- The error converges to 0 as $s \to \infty$, i.e as $h \to 0$.
- If the number of panels doubles, the error is expected to decrease roughly by a factor of 4, if we assume that f'' does not vary a lot.
- If f'' is not continuous $[a, b]$, the error formula does not apply, i.e. we cannot tell how well the rule will approximate the integral.

Let's take a look at the simple and compound midpoint rule errors again:

$$\text{simple } \frac{(b-a)^3}{24} f''(\eta), \quad \frac{(b-a)h^2}{24} f''(\eta) \text{ composite}$$

Note that the η's are not necessarily the same, however there are clearly similarities between the two formulae.

Trapezoid Rule

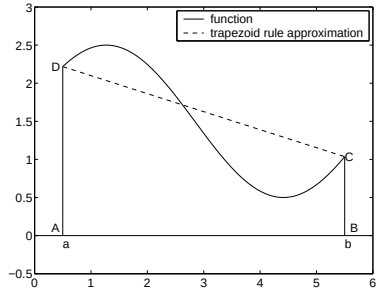
Assume $f(x)$ is approximated by a polynomial $p_1(x)$ of degree at most 1, passing through the points $(a, f(a))$ and $(b, f(b))$, i.e. by $p_1(x) = \frac{x-b}{a-b} f(a) + \frac{x-a}{b-a} f(b)$. Then

$$I = \int_a^b f(x)dx \approx \int_a^b p_1(x)dx = \frac{f(a)}{a-b}\int_a^b (x-b)dx + \frac{f(b)}{b-a}\int_a^b (x-a)dx$$

$$= \frac{f(a)}{a-b}\left[\frac{(x-b)^2}{2}\right]_a^b + \frac{f(b)}{b-a}\left[\frac{(x-a)^2}{2}\right]_a^b$$

$$= \frac{-f(a)}{a-b}\frac{(a-b)^2}{2} + \frac{f(b)}{b-a}\frac{(b-a)^2}{2}$$

$$= \frac{b-a}{2}(f(a) + f(b)) = Q_T.$$



That is, I is approximated by the area of the trapezoid ABCD.

Trapezoid Rule

If f happens to be a polynomial of degree at most 1, then the trapezoid rule is exact, therefore the trapezoid rule has polynomial degree *at least* 1, by construction. It will be shown next that it has polynomial degree (exactly) 1.

Consider $f(x) = x^2$, $a = 0$ and $b = 1$. It is easy to see that

$$I = \int_a^b f(x)dx = \int_0^1 x^2 dx = \left[\frac{x^3}{3}\right]_0^1 = \frac{1}{3}$$ while

$$Q_T = \frac{b-a}{2}(f(a) + f(b)) = \frac{1}{2}(f(0) + f(1))\frac{1}{2}(0+1) = \frac{1}{2} \neq \frac{1}{3}.$$

Thus the trapezoid rule has polynomial degree exactly 1.

Error in trapezoid rule

Recall the polynomial interpolation error

$$f(x) - p_1(x) = \frac{(x-a)(x-b)}{2!} f''(\xi(x))$$

for some $\xi \in \text{ospr}\{a, b, x\}$, that depends on x. Integrating we have,

$$\int_a^b f(x)dx - \int_a^b p_1(x)dx = \int_a^b \frac{(x-a)(x-b)}{2!} f''(\xi(x))dx.$$

Error in trapezoid rule

Repeat:

$$\int_a^b f(x)dx - \int_a^b p_1(x)dx = \int_a^b \frac{(x-a)(x-b)}{2!} f''(\xi(x))dx.$$

Notice that $(x-a)(x-b)$ is continuous and one-signed in $[a, b]$, and assume f'' is continuous in $[a, b]$. Applying the MVTI we have,

$$\int_a^b f(x)dx - \int_a^b p_1(x)dx = \frac{f''(\eta)}{2}\int_a^b (x-a)(x-b)dx = \frac{f''(\eta)}{2}\int_a^b (x^2 - (a+b)x + ab)dx$$

$$= \frac{f''(\eta)}{2}\left[\frac{x^3}{3} - (a+b)\frac{x^2}{2} + abx\right]_a^b = \frac{f''(\eta)}{2}\frac{-(b-a)^3}{6} = -f''(\eta)\frac{(b-a)^3}{12}.$$

Hence the error in the trapezoid rule is

$$\int_a^b f(x)dx - \frac{b-a}{2}(f(a) + f(b)) = -f''(\eta)\frac{(b-a)^3}{12},$$

for some $\eta \in [a, b]$. Clearly, if f is a polynomial of degree at most 1, the trapezoid rule is exact. This is not new, since it was shown above that the trapezoid rule has polynomial degree 1.

Alternative derivation of quadrature rules using model intervals and the method of undetermined coefficients

Quadrature rules can be alternatively derived by the following technique:
Assume we have a model interval, say $[-1, 1]$, and a quadrature rule

$$\int_{-1}^{1} g(t)dt \approx \sum_{i=0}^{n} A_i g(t_i)$$

for some given data points t_i, $i = 0, \cdots, n$, and some unknown coefficients (weights) A_i, $i = 0, \cdots, n$, which we would like to determine (hence the *method of undetermined coefficients*).
We can determine the weights by forcing the rule to be exact on $g(t) = 1, t, t^2, \cdots, t^n$. When $g(t)$ is chosen to be a monomial, it is easy to calculate the integral. We can then setup a linear system of equations with respect to the weights.

Example: Alternative derivation of trapezoid rule.

Alternative derivation of trapezoid rule

Consider the rule

$$\int_{-1}^{1} g(t)dt \approx A_0 g(-1) + A_1 g(1)$$

Let $g(t) = 1$: We have $\int_{-1}^{1} 1dt = [t]_{-1}^{1} = 2$ and $A_0 g(-1) + A_1 g(1) = A_0 \cdot 1 + A_1 \cdot 1$.

To make the rule exact for $g(t) = 1$, we need $A_0 + A_1 = 2$.

Let $g(t) = t$: We have $\int_{-1}^{1} t dt = \left[\frac{t^2}{2}\right]_{-1}^{1} = 0$ and $A_0 g(-1) + A_1 g(1) = A_0 \cdot (-1) + A_1 \cdot 1$.

To make the rule exact on $g(t) = t$, we need $-A_0 + A_1 = 0$. Thus we have,

$$\left\{ \begin{array}{c} A_0 + A_1 = 2 \\ -A_0 + A_1 = 0 \end{array} \right\} => 2A_1 = 2 => A_1 = 1 => A_0 = 1.$$

Hence the rule becomes

$$\int_{-1}^{1} g(t)dt \approx g(-1) + g(1) \quad = \frac{1-(-1)}{2}(g(-1) + g(1))$$

This is the trapezoid rule in $[-1, 1]$.
Can it be used for other intervals? Yes, with a change of intervals and variables. (See next section.)

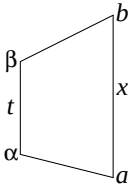
Alternative derivation of quadrature rules

Notes:

- If a rule is exact for $1, t, t^2, \cdots, t^m$, it is exact for all polynomials of degree up to m (by linearity).
- The method of undetermined coefficients is a more general technique that applies to several other problems, besides deriving integration rules. Assume we have a formula with some unknown parameters, then we setup conditions that the formula should satisfy, thus setting up a system of equations with respect to the parameters. For example, to construct the interpolating polynomial of degree at most n, we write it in terms of the monomials as $p_n(t) = \sum_{i=0}^{n} \alpha_i t^i$, and apply the interpolating conditions $p_n(t_i) = f(t_i)$, $i = 0, \cdots, n$. This leads to a linear system (Vandermonde matrix) with respect to the coefficients α_i. The procedure of setting up the formula for p_n and the conditions it should satisfy can be viewed as a method of undetermined coefficients, with the α_i playing the role of the undetermined coefficients.

Transforming quadrature rules from model intervals to other

Assume we want to approximate $\int_a^b f(x)dx$, where $\alpha \neq a$ and/or $\beta \neq b$, by the quadrature rule $\int_\alpha^\beta g(t)dt \approx \sum_{i=0}^{n} A_i g(t_i)$.



We use a linear transformation of variables

$$x = \frac{(b-a)t + a\beta - b\alpha}{\beta - \alpha}$$

to map $t = \alpha$ to $x = a$ and $t = \beta$ to $x = b$ (and vice versa). Then,

$$\int_a^b f(x)dx = \int_\alpha^\beta f(\frac{(b-a)t + a\beta - b\alpha}{\beta - \alpha}) \frac{b-a}{\beta - \alpha} dt$$

and

$$\int_a^b f(x)dx \approx \frac{b-a}{\beta - \alpha} \sum_{i=0}^{n} A_i f(\frac{(b-a)t_i + a\beta - b\alpha}{\beta - \alpha}).$$

Transforming quadrature rules from model intervals to other -- Example

Example: Trapezoid rule from $[-1,1]$ to $[a,b]$. As before, we have (with $\alpha = -1$, $\beta = 1$, $t_0 = -1$, $t_1 = 1$)

$$\int_{-1}^{1} g(t)dt \approx 1 \cdot g(-1) + 1 \cdot g(1)$$

and

$$\int_a^b f(x)dx \approx \frac{b-a}{1-(-1)}\left(1 \cdot f\left(\frac{-(b-a)+a-(-b)}{1-(-1)}\right) + 1 \cdot f\left(\frac{(b-a)+a-(-b)}{1-(-1)}\right)\right)$$

$$= \frac{b-a}{2}\left(f(a) + f(b)\right).$$

Notes:

- If a rule, given for a certain interval, is exact for $1, t, t^2, \cdots, t^m$, then the rule obtained from it by linear change of variables/intervals for another interval is also exact for $1, t, t^2, \cdots, t^m$.
- However, if a rule (given for a general interval) is exact for t^k, for some k, in a certain interval, it is not necessarily exact for $1, t, t^2, \cdots, t^{k-1}$, and it is not necessarily exact for t^k in all intervals.

Simpson's rule

Assume $f(x)$ is approximated by a polynomial $p_2(x)$ of degree at most 2, passing through the points $(a, f(a))$, $(m, f(m))$ and $(b, f(b))$, where $m = \frac{a+b}{2}$ is the midpoint of (a,b), i.e. by $p_2(x) = \frac{(x-m)(x-b)}{(a-m)(a-b)} f(a) + \frac{(x-a)(x-b)}{(m-a)(m-b)} f(m) + \frac{(x-a)(x-m)}{(b-a)(b-m)} f(b)$. Then

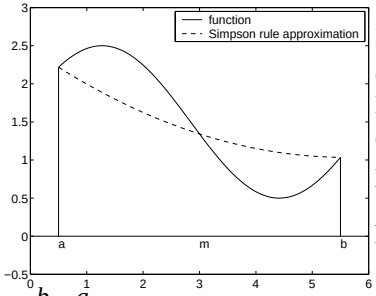
$$I = \int_a^b f(x)dx \approx \int_a^b p_2(x)dx = \frac{f(a)}{(a-m)(a-b)}\int_a^b (x-m)(x-b)dx$$

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

$$= \frac{f(a)}{(b-a)^2/2}(b-a)^3/12 + \frac{f(m)}{(b-a)^2/4}(b-a)^3/6 + \frac{f(b)}{(b-a)^2/2}(b-a)^3/12$$

$$= \frac{b-a}{6}\left(f(a) + 4f(m) + f(b)\right) = Q_S.$$

Simpson's rule



In the figure, I (the area under the solid line) is approximated by the area under the dashed line. By construction, Simpson's rule has polynomial degree at least 2. It will be shown next that it has polynomial degree 3. Consider $f(x) = x^3$. We have $I = \int_a^b f(x)dx = \int_a^b x^3 dx = \left[\frac{x^4}{4}\right]_a^b = \frac{b^4-a^4}{4}$

$= \frac{b-a}{4}(b^3 + b^2a + ba^2 + a^3)$, and $Q_S = \frac{b-a}{6}(a^3 + 4(\frac{a+b}{2})^3 + b^3)$

$= \frac{b-a}{6}(a^3 + \frac{1}{2}(a^3 + 3a^2b + 3ab^2 + b^3) + b^3) = \frac{b-a}{6}\frac{3}{2}(a^3 + a^2b + ab^2 + b^3) = I.$

Thus Simpson's rule is exact for $f(x) = x^3$ (on arbitrary intervals). Recall that, by construction, Simpson's is also exact for $f(x) = 1, x, x^2$. Since a polynomial of degree at most 3 can be written as a linear combination of $1, x, x^2, x^3$, by linearity Simpson's is exact for all polynomials of degree at most 3, and therefore has polynomial degree at least 3.

Error in Simpson's rule

Consider now $f(x) = x^4$, $a = 0$ and $b = 1$. It is easy to see that

$I = \int_a^b f(x)dx = \int_0^1 x^4 dx = \left[\frac{x^5}{5}\right]_0^1 = \frac{1}{5}$ while

$Q_S = \frac{1}{6}(f(0) + 4f(\frac{1}{2}) + f(1)) = \frac{1}{6}(0 + 4\frac{1}{16} + 1) = \frac{1}{6}\frac{5}{4} = \frac{5}{24} \neq \frac{1}{5}.$

Thus Simpson's rule has polynomial degree exactly 3.

Error in Simpson's rule

The natural way to go about deriving a formula for Simpson's rule error is to integrate

$$f(x) - p_2(x) = \frac{f^{(3)}(\xi)}{3!}(x-a)(x-m)(x-b).$$

However, $(x-a)(x-m)(x-b)$ does not have constant sign in $[a,b]$, therefore the MVTI does not apply. To get around this, consider the polynomial interpolant $p_3(x)$ of degree at most 3, that matches the data $(a, f(a))$, $(m, f(m))$, $(m, f'(m))$ and $(b, f(b))$, which we know it exists and is unique. We can show that

$$f(x) - p_3(x) = \frac{f^{(4)}(\xi)}{4!}(x-a)(x-m)^2(x-b)$$

Error in Simpson's rule

therefore

$$\int_a^b f(x)dx - \int_a^b p_3(x)dx = \int_a^b \frac{f^{(4)}(\xi)}{4!}(x-a)(x-m)^2(x-b)dx.$$

Note that $(x-a)(x-m)^2(x-b) \le 0$ in $[a,b]$, therefore, assuming $f^{(4)}$ is continuous in $[a,b]$, we have

$$\int_a^b f(x)dx - \int_a^b p_3(x)dx = \frac{f^{(4)}(\eta)}{4!}\int_a^b (x-a)(x-m)^2(x-b)dx = -\frac{f^{(4)}(\eta)}{2880}(b-a)^5.$$

Since Simpson's is exact for all polynomials of degree 3,

$$\int_a^b p_3(x)dx = \frac{b-a}{6}(p_3(a)+4p_3(m)+p_3(b)) = \frac{b-a}{6}(f(a)+4f(m)+f(b)),$$

the last equality obtained by construction of p_3. Thus,

$$\int_a^b f(x)dx - \frac{b-a}{6}(f(a)+4f(m)+f(b)) = -\frac{f^{(4)}(\eta)}{2880}(b-a)^5$$

for some $\eta \in [a,b]$. This is the error formula for Simpson's rule.

Corrected trapezoid rule

Assume $f(x)$ is approximated by a polynomial $p_3(x)$ of degree at most 3 passing through the points $(a, f(a))$, $(a, f'(a))$, $(b, f(b))$ and $(b, f'(b))$, i.e. $p_3(x)$ is the Hermite polynomial interpolant of degree at most 3. Then $I = \int_a^b f(x)dx \approx \int_a^b p_3(x)dx$ results in

$$\int_a^b f(x)dx \approx \frac{b-a}{2}(f(a)+f(b)) + \frac{(b-a)^2}{12}(f'(a)-f'(b)) = Q_{CT}.$$

This is the corrected trapezoid rule. By construction, the corrected trapezoid rule has polynomial degree at least 3. It will be shown next that it has polynomial degree exactly 3.

Consider $f(x) = x^4$, $a = 0$ and $b = 1$. As before, $I = \int_0^1 x^4 dx = \frac{1}{5}$,
while $Q_{CT} = \frac{1}{2}(0+1) + \frac{1}{12}(0-4) = \frac{1}{2} - \frac{4}{12} = \frac{1}{6} \ne \frac{1}{5}$.
Thus the corrected trapezoid rule has polynomial degree exactly 3.

Error in the corrected trapezoid rule

We integrate the Hermite polynomial interpolant error

$$f(x) - p_3(x) = \frac{f^{(4)}(\xi)}{4!}(x-a)^2(x-b)^2$$

and obtain (with the help of the MVTI)

$$\int_a^b f(x)dx - \int_a^b p_3(x)dx = \int_a^b \frac{f^{(4)}(\xi)}{4!}(x-a)^2(x-b)^2dx$$

$$= \frac{f^{(4)}(\eta)}{4!}\int_a^b (x-a)^2(x-b)^2dx = \frac{f^{(4)}(\eta)}{4!}\frac{(b-a)^5}{30} = \frac{f^{(4)}(\eta)}{720}(b-a)^5,$$

for some $\eta \in [a,b]$, assuming $f^{(4)}$ is continuous in $[a,b]$. This is the error formula for the corrected trapezoid rule.

Convergence of polynomial interpolatory rules

Recall the general form of polynomial interpolatory quadrature rules

$$Q_n = \int_a^b p_n(x)dx = \int_a^b \sum_{i=0}^n f(x_i^{(n)})l_i^{(n)}(x)dx = \sum_{i=0}^n f(x_i^{(n)})\int_a^b l_i^{(n)}(x)dx$$

$$= \sum_{i=0}^n f(x_i^{(n)})A_i^{(n)}, \text{ with weights } A_i^{(n)} = \int_a^b l_i^{(n)}(x)dx, \text{ where the superscript } (n) \text{ emphasizes}$$

that the rule is for $n+1$ data points. Note that the weights $A_i^{(n)}$ depend on $x_i^{(n)}$, since the Lagrange basis functions $l_i^{(n)}$ depend on $x_i^{(n)}$.

One would normally expect that by increasing n, the error $|I - Q_n|$ would converge to 0, i.e. $|I - Q_n| \underset{n\to\infty}{\to} 0$. It has been shown that this is true *only under certain conditions*.

Theorem: Let $n \in \mathbb{N}$ and $f \in \mathbb{C}[a,b]$. Let Q_n be the polynomial interpolatory rule defined above. Then

$|I - Q_n| \underset{n\to\infty}{\to} 0 \iff \exists \kappa > 0$, such that $\sum_{j=0}^n |A_i^{(n)}| \le \kappa$, $\forall n$.

We do not present the proof of this theorem, but discuss some practical implications of it.

Convergence of polynomial interpolatory rules

- If all weights are positive, we can approximate I by Q_n to arbitrary precision.
  To see why this is true, consider the function $f(x) = 1$. Any polynomial interpolatory rule is exact for such a function, i.e. $Q_n = I$ $\Rightarrow \sum_{i=0}^{n} 1 \cdot A_i^{(n)} = \int_a^b 1 dx \Rightarrow \sum_{i=0}^{n} A_i^{(n)} = b - a = \kappa > 0 \Rightarrow \sum_{i=0}^{n} |A_i^{(n)}| \le \kappa$, where κ is independent of n. Thus, $|I - Q_n| \underset{n\to\infty}{\to} 0$.
- Intuitively, if all weights are positive, we would expect the total round-off error when computing $Q_n = \sum_{i=0}^{n} A_i^{(n)} f(x_i^{(n)})$ to be smaller than if the weights had varying sign. This is because the round-off errors in $f(x_i^{(n)})$ are expected to be on the high side approximately as often as on the low side, in which case the constant sign of the $A_i^{(n)}$'s would help the errors on the high side to cancel out the errors on the low side.
- More generally, the quantity $\sum_{i=0}^{n} |A_i^{(n)}|$ is the *condition number* of the quadrature rule, that is, it is a measure of the sensitivity of Q_n to perturbations in the values of $f(x_i^{(n)})$. Large (small) $\sum_{i=0}^{n} |A_i^{(n)}|$ will cause a small perturbation in $f(x_i^{(n)})$ to result in a large (small) perturbation in Q_n.

The above discussion suggests that it is desirable to pick the points $x_i^{(n)}$ such that the weights $A_i^{(n)}$ are positive.

Newton-Cotes rules

Quadrature rules based on equidistant data points are referred to as *Newton-Cotes* (NC) rules.

For NC rules, the following error bounds hold:
If n is even (i.e. odd number of data, which include the midpoint), and $f \in \mathbb{C}^{n+2}[a, b]$,

$$|I - Q_n| \le C_n \left(\frac{b-a}{n}\right)^{n+3} \frac{f^{(n+2)}(\eta)}{(n+2)!},$$

and, if n is odd (i.e. even number of data) and $f \in \mathbb{C}^{n+2}[a, b]$,

$$|I - Q_n| \le \tilde{C}_n \left(\frac{b-a}{n}\right)^{n+2} \frac{f^{(n+1)}(\eta)}{(n+1)!},$$

where C_n and $\tilde{C}_n$ are some numbers that depend on n. Note that with n even (odd number of data), we gain one extra degree of precision, beyond the polynomial interpolant suggests.

Newton-Cotes rules

Although picking equidistant points seems natural and simple, it is not necessarily the best choice. It is known that

- The weights of an NC rule with $n \ge 10$ are always of varying sign.
- The weights of an NC rule grow without bound as n increases, i.e. $\sum_{i=0}^{n} |A_i^{(n)}| \underset{n\to\infty}{\to} \infty$.

For these reasons, NC rules with large n are not used in practice.
However, low n NC rules (such as Simpson's) are often used, and are extremely simple and helpful.

Compound (composite) quadrature rules

When an interval is broken into subintervals (called *panels* or *pieces*) and the same quadrature rule is applied to each subinterval, we have a *composite* or *compound* rule.

Compound quadrature rules often use a low n NC quadrature rule in each of the s subintervals, thus accommodating an increasing number of data, without suffering from unbounded weights.

Typical composite rules, their simple counterparts and errors:

Compound (composite) quadrature rules

Simple:	Rule	Error
Midpoint	$(b-a)\,f(m)$	$f''(\eta)\,\frac{(b-a)^3}{24}$
Trapezoid	$\frac{b-a}{2}\,(f(a)+f(b))$	$-f''(\eta)\,\frac{(b-a)^3}{12}$
Simpson's	$\frac{b-a}{6}\,(f(a)+4f(m)+f(b))$	$-f^{(4)}(\eta)\,\frac{(b-a)^5}{2880}$
Corr.Trap.	$\frac{b-a}{2}\,(f(a)+f(b))+\frac{(b-a)^2}{12}\,(f'(a)-f'(b))$	$f^{(4)}(\eta)\,\frac{(b-a)^5}{720}$

Compound:		
Midpoint	$h\sum_{i=1}^{s} f(a+(i-\frac{1}{2})h)$	$f''(\eta)\,\frac{h^2}{24}\,(b-a)$
Trapezoid	$\frac{h}{2}\,(f(a)+2\sum_{i=1}^{s-1} f(a+ih)+f(b))$	$-f''(\eta)\,\frac{h^2}{12}\,(b-a)$
Simpson's	$\frac{h}{6}\,[f(a)+2\sum_{i=1}^{s-1} f(a+ih)+4\sum_{i=1}^{s} f(a+(i-\frac{1}{2})h)+f(b)]$	$-f^{(4)}(\eta)\,\frac{h^4}{2880}\,(b-a)$
Corr.Trap.	$\frac{h}{2}\,(f(a)+2\sum_{i=1}^{s-1} f(a+ih)+f(b))+\frac{h^2}{12}\,(f'(a)-f'(b))$	$f^{(4)}(\eta)\,\frac{h^4}{720}\,(b-a)$

Error estimators for quadrature rules

Consider a composite quadrature rule and the associated error formula. It is clear that by increasing the number of panels (subintervals) the error decreases. Moreover, it decreases at a rate governed by the exponent of the step-size h as it appears in the error formula.

It is desirable to have a technique of estimating roughly the magnitude of the error, so that we know how large an n we should pick to have an error below a given tolerance `tol`.

Note that, if the function-integrand is given in analytic form and its derivatives can be calculated, we can also calculate bounds for the error in the quadrature formula.

Composite quadrature rules allow us to develop techniques to roughly estimate the error, without the need for derivative knowledge, minimization or maximization. These techniques are easy to program and provide good approximations to the quadrature error.

We will demonstrate such a technique using the composite trapezoidal rule.

Error estimators for quadrature rules -- Example

Let T_1 be the approximation to the integral I using the composite trapezoidal rule with s panels and E_1 the associated error, $E_1 = I - T_1 = -\,f''(\eta_1)(b-a)h^2/12$.
Let T_2 be the approximation to the integral I using the composite trapezoidal rule with $2s$ panels and E_2 the associated error, $E_2 = I - T_2 = -\,f''(\eta_2)(b-a)(\frac{h}{2})^2/12$.

If f'' does not vary a lot in $[a,b]$, we expect

$$E_2 \approx E_1/4 \tag{1}$$

or, equivalently,

$$E_1 \approx 4E_2. \tag{2}$$

We have

$$I - T_1 = E_1 \tag{3}$$

$$I - T_2 = E_2. \tag{4}$$

Subtracting (4) from (3) and using (2) we get

$$T_2 - T_1 = E_1 - E_2 \approx 3E_2 \Rightarrow E_2 \approx \frac{T_2 - T_1}{3}\,. \tag{5}$$

Thus, if we have T_1 and T_2 calculated, we can easily calculate an estimate to the error in T_2 using (5).

Error estimators for quadrature rules -- Example

We can use this technique recursively, by doubling the number of panels each time, until we reach a certain error tolerance.

Note also that $T_2 + E_2 = T_2 + \frac{T_2 - T_1}{3} = \frac{4T_2 - T_1}{3}$ is normally a better approximation to I than T_2. (More on this later, under *Romberg integration*.)

Algorithm for approximating an integral within a given tolerance using the composite trapezoidal rule with error estimator (nested composite trapezoidal rule)

do $s+1$ function evaluations $f_i = f(t_i)$, $i = 0, \cdots, s$
set $h = (b-a)/s$
calculate $T_1 = \frac{h}{2}\,(f_0 + 2\sum_{i=1}^{s-1} f_i + f_s)$
for $k = 2$, ... , until satisfied do
  set $t_{i-1/2} = \frac{t_{i-1} + t_i}{2}$, $i = 1, \cdots, s$
  do s function evaluations $f_{i-1/2} = f(t_{i-1/2})$, $i = 1, \cdots, s$
  calculate $T_k = \frac{h}{4}\,(f_0 + 2\sum_{i=1}^{s-1} f_i + 2\sum_{i=1}^{s} f_{i-1/2} + f_s)$
  or, more efficiently, $T_k = \frac{1}{2}\,T_{k-1} + \frac{h}{2}\sum_{i=1}^{s} f_{i-1/2}$ (6)
  estimate the error, $est = \frac{T_k - T_{k-1}}{3}$ (7)
  if $|est| \le$ `tol` break
  else set $s = 2s$, $h = h/2$ and reset the t_i's and f_i's
endfor
accept T_k as approximation to I

Error estimators for quadrature rules

Note that, each time we double the number of panels, we re-use the current function evaluations and past computations.

The above technique can be applied to all Newton-Cotes rules, and to other rules, as long as there is an error formula, with the stepsize h to some exponent and some other quantities assumed not to vary a lot.
Open rules (e.g. the midpoint rule), however, do not allow the re-use of (all) the current function evaluations.
To see how the current function evaluations are re-used, consider the following facts:

- The nodes of an NC rule are equidistant.
- The nodes in a s-panel partition remain nodes in a $2s$-panel part.
- The midpoints in a s-panel partition become nodes in a $2s$-panel partition.

Note:

- The coefficient in (7) (or (5)) and the coefficients in (6) may change depending on the NC rule chosen.

Error estimators for quadrature rules

Other notes:

- We can make the stopping criterion $|\frac{T_k - T_{k-1}}{3}| \le$ `tol` three times more severe by changing it to $|T_k - T_{k-1}| \le$ `tol`
- We can make the stopping criterion relative by changing it to $|T_k - T_{k-1}| \le |T_k|$ `tol`

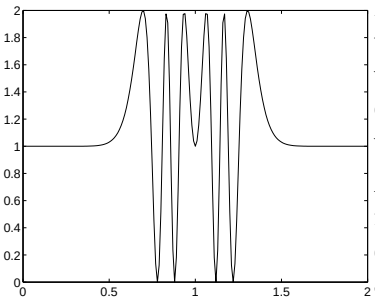
Other error estimators:

Another technique for estimating the error in a quadrature rule is to use a pair of rules, one of higher order than the other. Let Q_p and p be the approximation and order of the low order rule, and Q_q and q be the approximation and order of the high order rule. Then $Q_q - Q_p$ is a reasonable error estimate for Q_p. For specific pairs of rules, some more sophisticated formulae have been developed, e.g. $C(Q_q - Q_p)^{\alpha}$, where C and α are appropriate scalars.

Adaptive quadrature

The previous discussion suggests that a nested repetition of a compound quadrature rule results in an approximation to an integral within a user chosen error tolerance.

If the function-integrand varies a lot in some part of the interval (a, b) of integration and very little in another part, this flat (non-selective) repetition of a compound rule wastes a lot of function evaluations in the part of little variation.



In the figure to the left, the function varies very little in [0, 0.5] and [1.5, 2], while within [0.5, 1.5] it varies more. Moreover, even within [0.5, 1.5], we could possibly identify subintervals, where the variation is higher than in other.

Adaptive quadrature attempts to avoid the waste of function evaluations, by smartly selecting the parts of (a, b), where the rule should (is worth) be applied in a nested format. Adaptive quadrature dynamically refines the nodes in the parts of (a, b) where the accuracy has not reached a certain level, and assigns an appropriate portion of the total error to each subinterval / part of (a, b).

Adaptive quadrature

Adaptive quadrature techniques are often implemented by the use of recursion.

Let (Q, E) = LQ(f, a, b) be a routine that uses a standard quadrature rule in (a, b) for the function f, and returns the approximation Q to $\int_a^b f dx$, and an error estimate E. A simple implementation of the adaptive quadrature is as follows.

```
function Q = AQ(f, a, b, ε)
(Q, E) = LQ(f, a, b)
if |E| > ε then
    m = (a+b)/2
    return AQ(f, a, m, ε/2) + AQ(f, m, b, ε/2)
else
    return Q
endif
```

The above implementation uses an absolute stopping criterion and assigns about half the error to each of the two halves of the subinterval, in each repetition.

Adaptive quadrature

A relative criterion would be: `if |E| > ε * |Q|`. In the case of a relative criterion the tolerance need not be halved.

We may also set additional stopping criteria or limits in order to control the computational cost, and avoid divergent recursions, e.g. in cases of unbounded functions. Such criteria may include (a) a maximum nesting (recursion depth) level, (b) a maximum number of function evaluations, (c) a minimum length of subinterval, etc.

MATLAB has several adaptive quadrature routines, including `quad`, `quadl`, `quadgk` and `integral`.
Function `quad` uses Simpson's rule, `quadl` uses a combination of high order Gauss-Lobatto rules, while `quadgk` and `integral` use a combination of a Gauss and a Kronrod rule. (Lobatto rules are constructed in a similar way as Gauss rules, but include the endpoints. Kronrod rules add $n+1$ points to a n-point Gauss rule in a way to make the resulting rule of highest accuracy.)

Here is a simplified MATLAB implementation of adaptive quadrature, assuming `Q = LQ(f, a, b)` is a simple quadrature rule without error estimation:

Adaptive quadrature

```
function Q = AQ(f, a, b, ε)
Q0 = huge
Q = LQ(f, a, b)
if |Q - Q0| > ε |Q|
    Q = AQstp(f, a, b, ε, Q)
endif
function Q = AQstp(f, a, b, ε, Q0)
m = (a+b)/2
Q1 = LQ(f, a, m)
Q2 = LQ(f, m, b)
Q = Q1 + Q2
if (|Q - Q0| > ε |Q|)
    Q1 = AQstp(f, a, m, ε, Q1)
    Q2 = AQstp(f, m, b, ε, Q2)
    Q = Q1 + Q2
endif
```

Another implementation of adaptive quadrature uses a pair of quadrature rules as in the case of error estimators.

Gauss quadrature

Newton-Cotes (NC) rules with nodes x_i, $i = 0, \cdots, n$, (i.e. $n+1$ points) have polynomial degree (pd) $\geq n$. The rules we have seen have pd either exactly n (e.g. trapezoid), or $n+1$ (e.g. midpoint, Simpson).

What is the highest pd we can achieve with $n+1$ points?
Carl Friedrich Gauss (1777-1855) laid the foundations for answering this question.

Consider a general rule of the form $Q_n = \sum_{i=0}^{n} w_i f(x_i)$. There are $2n+2$ free parameters:

- $n+1$ weights w_i, $i = 0, \cdots, n$
- $n+1$ nodes x_i, $i = 0, \cdots, n$

In order to construct a quadrature rule with pd at least d, the following conditions must be satisfied:

$$\int_a^b x^k dx = \sum_{i=0}^{n} w_i x_i^k, \quad k = 0, 1, \cdots, d,$$

that is, the rule must be exact for $f(x) = x^k$, $k = 0, 1, \cdots, d$.
Let us choose $d = 2n+1$, so that we have a total of $2n+2$ conditions. Then the conditions are

Gauss quadrature

$$k = 0: \int_a^b dx = \sum_{i=0}^{n} w_i \quad \Rightarrow b - a = \sum_{i=0}^{n} w_i \tag{0}$$

$$k = 1: \int_a^b x dx = \sum_{i=0}^{n} w_i x_i \quad \Rightarrow \frac{b^2 - a^2}{2} = \sum_{i=0}^{n} w_i x_i \tag{1}$$

.

$$k = 2n+1: \int_a^b x^{2n+1} dx = \sum_{i=0}^{n} w_i x_i^{2n+1} \Rightarrow \frac{b^{2n+2} - a^{2n+2}}{2n+2} = \sum_{i=0}^{n} w_i x_i^{2n+1} \tag{2n+1}$$

Given a, b and n, is the above system of equations solvable for w_i, $i = 0, \cdots, n$, and x_i, $i = 0, \cdots, n$?

Note that the above equations, with the exception of the $k = 0$ case, are nonlinear. (There are products of unknowns $w_i x_i$ and powers of unknowns $w_i x_i^2$, etc.)

Nonlinear systems of equations do not necessarily have a solution, and even if they have, it may not be unique. Moreover, even if the solution is unique, it may be complex or imaginary.

Gauss quadrature

Gauss showed that the above nonlinear system has a unique real solution.
Theorem: The following can be shown: Given a, b and n,

- Equations (0)-(2n+1) have a unique real solution for w_i, $i = 0, \cdots, n$, and x_i, $i = 0, \cdots, n$, which gives rise to a quadrature rule of pd exactly $2n+1$ (i.e. the rule is not exact for a polynomial of degree $2n+2$).
- The weights w_i, $i = 0, \cdots, n$, are all positive. (Thus $|I - Q_n| \underset{n \to \infty}{\to} 0$.)
- The points x_i, $i = 0, \cdots, n$, include neither a nor b (i.e. the rule is open).
- The points x_i, $i = 0, \cdots, n$, are the roots of the $(n+1)$st degree Legendre orthogonal polynomial $q_{n+1}(x)$, with respect to the weight function $w(x) = 1$, and a continuous inner product in (a, b).

Other facts:

- A Gauss rule can be constructed by first constructing $q_{n+1}(x)$ (e.g. by the three term recurrence relation of Gram-Schmidt), then finding its roots x_i, $i = 0, \cdots, n$, then constructing the polynomial interpolant of the data $(x_i, f(x_i))$, $i = 0, \cdots, n$, then integrating the interpolant (thus computing the weights). Thus, Gauss rules are *interpolatory*.

Gauss quadrature

- The weights and points are usually given by complicated formulae and are often irrational numbers.
- In order to simplify the procedure, the literature (just about every Numerical Methods textbook) provides tables of the weights and points of Gauss rules for selected values of n and for some model interval, say $[-1, 1]$. We then map the points to the interval of interest by linear transformation, and scale the range of integration appropriately.

point	weight
2-point Gauss rule	
-0.57735026919625	1
0.57735026919625	1
4-point Gauss rule	
-0.861136311594052	0.347854845137454
-0.339981043584856	0.652145154862546
0.339981043584856	0.652145154862546
0.861136311594052	0.347854845137454

- The points are located symmetrically about the origin and are in $(-1, 1)$. Points that are located symmetrically about the origin have equal weights.

Gauss quadrature

- The set of points of a $(n+1)$-point Gauss rule is disjoint from the set of points of a $(m+1)$-point Gauss rule, for $n \neq m$, except if n and m are even, in which case only the midpoint is shared.
- The compound $(n+1)$-point Gauss rule with s panels shares no function evaluations with the compound $(n+1)$-point Gauss rule with $2s$ panels (with the exception of the midpoint if n is even.)
- Open rules can be used to compute integrals with singularities on the boundaries (endpoints).

  Example: Consider $\int_0^1 \frac{\sin x}{x}\,dx$. This integral is defined. By de l' Hospital's rule we have $\lim_{x\to 0} \frac{\sin x}{x} = \lim_{x\to 0} \frac{\cos x}{1} = 1$. However at $x = 0$, $\frac{\sin 0}{0}$ cannot be computed. An open rule, such as a Gauss rule, can be easily programmed to approximate the above integral, since it will never evaluate the integrand at $x = 0$, while a closed rule, such as trapezoid, Simpson's, etc, cannot.

Quadrature rules (simple)

Quadrature rule	$n+1$	d	respective poly.interp.	approx. $Q(f)$ to integral $I(f)$	error $I(f) - Q(f)$
rectangle	1	0	constant	$(b-a)f(a)$	$f'(\eta)\frac{(b-a)^2}{2}$
midpoint	1	1	constant	$(b-a)f(m)$	$f''(\eta)\frac{(b-a)^3}{24}$
trapezoid	2	1	linear	$\frac{b-a}{2}(f(a)+f(b))$	$-f''(\eta)\frac{(b-a)^3}{12}$
Simpson	3	3	quadratic	$\frac{b-a}{6}(f(a)+4f(m)+f(b))$	$-f^{(4)}(\eta)\frac{(b-a)^5}{2880}$
corr. trap.	4	3	Herm. cub.	$\frac{b-a}{2}(f(a)+f(b)) + \frac{(b-a)^2}{12}(f'(a)-f'(b))$	$f^{(4)}(\eta)\frac{(b-a)^5}{720}$
gen. NC	$n+1$	$\geq n$	deg $\leq n$	$\sum_{i=1}^{n} w_i f(x_i)$	$f^{(d+1)}(\eta)\frac{(b-a)^{d+2}}{K}$
2-pt Gauss	2	3	linear	$\frac{b-a}{2}[f(\frac{a+b}{2}-\frac{b-a}{2\sqrt{3}}) + f(\frac{a+b}{2}+\frac{b-a}{2\sqrt{3}})]$	$f^{(4)}(\eta)\frac{(b-a)^5}{4320}$
gen. Gauss	$n+1$	$2n+1$	deg $\leq n$		$f^{(d+1)}(\eta)\frac{(b-a)^{d+2}}{C}$

Quadrature rules (simple)

In the above,
$n+1$ number of data (and number of function evaluations), including derivative data,
d polynomial degree (degree of accuracy) of the quadrature rule,
η an unknown point in $[a, b]$, in general different for each rule,
$m = \frac{a+b}{2}$ midpoint of the interval and
K, C some constants, in general different for each rule.

Compound quadrature rules

Quadrature rule	n	d	respect. pc. poly.interp.	approx. $Q(f)$ to integral $I(f)$	erro $I(f) - Q(f$
rectangle	s	0	constant	$h\sum_{i=0}^{s-1} f(a+ih)$	$f'(\eta)\frac{h}{2}(b-a$
midpoint	s	1	constant	$h\sum_{i=1}^{s} f(a+(i-\frac{1}{2})h)$	$f''(\eta)\frac{h^2}{24}(b-a$
trapezoid	$s+1$	1	linear	$\frac{h}{2}(f(a)+2\sum_{i=1}^{s-1} f(a+ih)+f(b))$	$-f''(\eta)\frac{h^2}{12}(b-a$
Simpson	$2s+1$	3	quadratic	$\frac{h}{6}[f(a)+2\sum_{i=1}^{s-1} f(a+ih) +4\sum_{i=1}^{s} f(a+(i-\frac{1}{2})h)+f(b)]$	$-f^{(4)}(\eta)\frac{h^4}{2880}(b-a$
corr. trap.	$s+3$	3	Herm. cub.	$\frac{h}{2}(f(a)+2\sum_{i=1}^{s-1} f(a+ih)+f(b)) + \frac{h^2}{12}(f'(a)-f'(b))$	$f^{(4)}(\eta)\frac{h^4}{720}(b-a$
gen. NC		d			$f^{(d+1)}(\eta)\frac{h^{d+1}}{K}(b-a$

Compound quadrature rules

In the above,
n number of data (and number of function evaluations), including derivative data,
d polynomial degree of the quadrature rule,
η an unknown point in $[a, b]$, in general different for each rule,
$h = \dfrac{b-a}{s}$ the step-size of each panel,
s the number of panels and
K some constant, in general different for each rule.

Order of a (compound) quadrature rule: the exponent associated with the step-size h in the error formula for the rule. Notice that, for all rules studied, the order is $d+1$.

We say that the midpoint and trapezoid rules are second order, while Simpson's, corrected trapezoid and two-point Gauss are fourth order.

Comparison between Newton-Cotes and Gauss rules

Property	Newton-Cotes	Gauss
points equidistant	yes	no
points rational	yes	mostly no
weights positive	not all (for $n \geq 10$ some weights are always negative)	yes
weights rational	yes	mostly no
end-points included	yes or no (closed or open)	no (open)
polynomial degree (degree of accuracy) with $n+1$ points	n or $n+1$	$2n+1$
interpolatory	yes	yes
re-use of function evals when doubling the number of panels	often yes	mostly no

Romberg integration

Romberg integration uses a low order composite quadrature rule (e.g. composite trapezoidal) and a technique called *Richardson extrapolation* to obtain a high order quadrature rule. The basic mathematical ingredient of Romberg integration is, as in the case of error estimators, the formula for the error of the (composite) quadrature rule.

We will demonstrate Romberg integration using the composite trapezoidal rule, but the technique can be applied to all rules for which we have a formula giving a (Taylor) expansion of the error.

Let $I = \int_a^b f(x)dx$ be the integral to be approximated and let initially $h = (b-a)/s$.

Denote by $T_{k,1}$ the approximation to I using the trapezoid rule with stepsize $\dfrac{h}{2^{k-1}}$, $k = 1, 2, \cdots$. Recall that, from the error formulae for the trapezoid and the corrected trapezoid rules, we have

$$I - T_{1,1} = -f''(\eta)\frac{b-a}{12}h^2 = \frac{f'(a)-f'(b)}{12}h^2 + f^{(4)}(\eta')\frac{b-a}{720}h^4. \quad (1)$$

Romberg integration

In general, it can be shown that

$$I - T_{1,1} = c_2h^2 + c_4h^4 + c_6h^6 + \cdots \quad (2)$$

where $c_2, c_4, c_6, \cdots$ are scalars that do not depend on h. Similarly,

$$I - T_{2,1} = c_2(\frac{h}{2})^2 + c_4(\frac{h}{2})^4 + c_6(\frac{h}{2})^6 + \cdots \quad (3)$$

Using (2) and (3) (more specifically, $4\times(3)-(2)$), we get

$$3I - (4T_{2,1} - T_{1,1}) = c_2 \cdot 0 \cdot h^2 + c_4(\frac{1}{4}-1)h^4 + c_6(\frac{1}{16}-1)h^6 + \cdots$$

from which we get

$$I - \frac{1}{3}(4T_{2,1} - T_{1,1}) = d_4h^4 + d_6h^6 + \cdots \quad (4)$$

for some scalars $d_4, d_6, \cdots$ that do not depend on h. Thus, we have eliminated the lower order term (i.e. the h^2 term) of the error in the trapezoid rule, and obtained a fourth order approximation to I, which gives rise to a quadrature rule. More specifically, the *first level* Richardson extrapolation rule based on the trapezoid rule is given by

$$I \approx \frac{1}{3}(4T_{2,1} - T_{1,1}) \quad (5)$$

where the error is $O(h^4)$. Let $T_{2,2} = \dfrac{1}{3}(4T_{2,1} - T_{1,1})$.

Romberg integration

We now proceed to the second level Richardson extrapolation. As before,

$$I - T_{3,1} = c_2(\frac{h}{4})^2 + c_4(\frac{h}{4})^4 + c_6(\frac{h}{4})^6 + \cdots \tag{6}$$

Using (3) and (6) (more specifically, $4 \times (6) - (3)$), we get

$$3I - (4T_{3,1} - T_{2,1}) = c_2 \cdot 0 \cdot (\frac{h}{2})^2 + c_4(\frac{1}{4} - 1)(\frac{h}{2})^4 + c_6(\frac{1}{16} - 1)(\frac{h}{2})^6 + \cdots$$

from which we get

$$I - \frac{1}{3}(4T_{3,1} - T_{2,1}) = d_4(\frac{h}{2})^4 + d_6(\frac{h}{2})^6 + \cdots \tag{7}$$

Note that, again, the lower order term (i.e. the h^2 term) of the error in the trapezoid rule is eliminated, and a fourth order approximation to I is obtained, but this time the stepsize is half. That is, relation (7) is the same as (4), but for half stepsize.

Let $T_{3,2} = \frac{1}{3}(4T_{3,1} - T_{2,1})$. Using (4) and (7) (more specifically, $16 \times (7) - (4)$), we get

$$15I - (16T_{3,2} - T_{2,2}) = d_4 \cdot 0 \cdot h^4 + d_6(\frac{1}{4} - 1)h^6 + \cdots$$

from which we get

$$I - \frac{1}{15}(16T_{3,2} - T_{2,2}) = e_6 h^6 + \cdots \tag{8}$$

for some scalars $e_6, \cdots$ that do not depend on h.

Romberg integration

Thus, we have eliminated the next order term (i.e. the h^4 term) of the error in the trapezoid rule, and obtained a sixth order approximation to I, which gives rise to a quadrature rule. More specifically, the *second level* Richardson extrapolation rule based on the trapezoid rule is given by

$$I \approx \frac{1}{15}(16T_{3,2} - T_{2,2}) \tag{9}$$

where the error is $O(h^6)$. Let $T_{3,3} = \frac{1}{15}(16T_{3,2} - T_{2,2})$. We can continue this procedure and develop approximations to I of $O(h^8), O(h^{10}), \cdots, O(h^{2m})$ error. The procedure generates approximations to I which can be arranged in a table such as:

	$O(h^2)$	$O(h^4)$	$O(h^6)$		$O(h^{2m})$
h	$T_{1,1}$				
$\frac{h}{2}$	$T_{2,1}$	$T_{2,2}$			
$\frac{h}{4}$	$T_{3,1}$	$T_{3,2}$	$T_{3,3}$		
.	.	.	.		
$\frac{h}{2^{m-1}}$	$T_{m,1}$	$T_{m,2}$	$T_{m,3}$	$\cdots$	$T_{m,m}$

Romberg integration

- The $T_{i,j}$'s are given by a recursive definition

$$T_{i,j} = \frac{1}{4^{j-1} - 1}(4^{j-1}T_{i,j-1} - T_{i-1,j-1}), \quad j = 2, \cdots, i, \; i = 2, \cdots, m.$$

- The row index (i) in the table shows the halving of the stepsize, while the column index (j) shows the level (depth) of extrapolation. The order of the approximation to I is $2j$.
- When implementing Romberg integration, we only need to store the two latest rows (or columns) of the data in order to proceed to the next level.
- The number of function evaluations required to obtain $T_{m,m}$ is the same as the number required to obtain $T_{m,1}$, and this is approximately 2^{m-1} times the number required to obtain $T_{1,1}$. Thus, with the same number of function evaluations as those required by a low-order composite quadrature rule and a few extra calculations we obtain a high order quadrature rule.
- To get full advantage of the high order of Romberg integration the function-integrand must be in $\mathbb{C}^{2m}$.
- The computations involved in Romberg integration may become unstable for large m. But one or two levels of extrapolation are usually very effective.

Infinite (and semi-infinite) integrals

Infinite integrals are those definite integrals whose at least one endpoint is plus or minus infinity. Without loss of generality, we will consider the semi-infinite integral $I = \int_a^\infty f(x)dx$ and assume that I exists.

How can we compute (an approximation to) I?

(a) *Truncation*: Following the truncation approach, we find a 'sufficiently large' b, consider

$$I = \int_a^\infty f(x)dx \approx \int_a^b f(x)dx = I_b,$$

and apply a quadrature rule to I_b. Thus, we first approximate the integral by another (continuous) integral, then apply a discrete approximation to the latter.

How is b chosen?

(i) Start with some b, get an approximation to I_b, then keep increasing b, until the approximations to I_b seem to converge, e.g. when the absolute value of the (relative) difference between successive approximations is small.

Infinite (and semi-infinite) integrals

(ii) Use mathematical techniques to find a b such that $|I - I_b| \le tol$, for a user-chosen tolerance tol. Note that $\int_a^\infty f(x)dx = \int_a^b f(x)dx + \int_b^\infty f(x)dx$ $=> \int_a^\infty f(x)dx - \int_a^b f(x)dx = \int_b^\infty f(x)dx$. Thus $I - I_b = \int_b^\infty f(x)dx$. Now try to work an upper bound of the form $|\int_b^\infty f(x)dx| \le \cdots \le \cdots b \cdots = g(b)$, given in terms of b. To compute I within accuracy ε

- Choose b such that $g(b) \le \frac{\varepsilon}{2}$. (Let b be the smallest such b.)
- Compute I_b within accuracy $\frac{\varepsilon}{2}$, using some (possibly adaptive) quadrature rule with an error estimator. Let Q be the approximation to I_b.
- Take Q as the approximation to I.

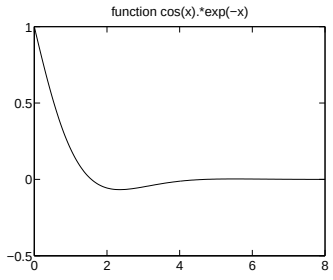
Then $|I - Q| = |I - I_b + I_b - Q| \le |I - I_b| + |I_b - Q| \le \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon$. Thus Q satisfies the ε accuracy.

Infinite (and semi-infinite) integrals

Example: Let $I = \int_a^\infty (\cos x)e^{-x}dx$. We have $|\int_b^\infty \cos x e^{-x}dx| \le \int_b^\infty |\cos x|e^{-x}dx \le \int_b^\infty e^{-x}dx = e^{-b}$. To compute I within accuracy ε

- Choose b such that $e^{-b} \le \frac{\varepsilon}{2} => e^{-b} \le \frac{\varepsilon}{2} => -b \le \ln(\frac{\varepsilon}{2}) => b \ge -\ln(\frac{\varepsilon}{2})$.
- Compute $I_b = \int_a^b \cos x e^{-x}dx$ within accuracy $\frac{\varepsilon}{2}$. Let Q be the approximation to I_b.
- Take Q as the approximation to I.

MATLAB example: Let $I = \int_0^\infty (\cos x)e^{-x}dx$. In the plot, we see how the integrand damps down as x increases. Through mathematical techniques (more specifically repeated application of integration by parts), we know that $I = 0.5$.



Infinite (and semi-infinite) integrals

In the following, we test how well the MATLAB function `quadl` approximates I, and the effect of the 'false' right endpoint.

```
>> Q = 0.5;
>> tol = 1e-8;
>> b = -log(tol/2)
b =
   19.1138
>> quadl('cos(x).*exp(-x)', 0,  8, tol) - Q
ans =
   1.9035e-04
>> quadl('cos(x).*exp(-x)', 0, 12, tol) - Q
ans =
  -4.2408e-06
>> quadl('cos(x).*exp(-x)', 0, 16, tol) - Q
ans =
   3.7686e-08
>> quadl('cos(x).*exp(-x)', 0, 20, tol) - Q
ans =
   5.2015e-10
```



```
>> quadl('cos(x).*exp(-x)', 0, 24, tol) - Q
ans =
  -2.5101e-11
>> [q, nfcn] = quadl('cos(x).*exp(-x)', 0, b, tol/2);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  -1.7600e-09  168
```

Infinite (and semi-infinite) integrals

(b) *Transformation*: Following the transformation approach, we transform the infinite interval into a finite one by a change of variable. It is important to note that not all changes of variable work well. Some may give rise to singularities of the integrand, others may give rise to infinite oscillations of the integrand, etc.

Example: Let $I = \int_a^\infty \cos x e^{-x} dx$.

Attempt 1:

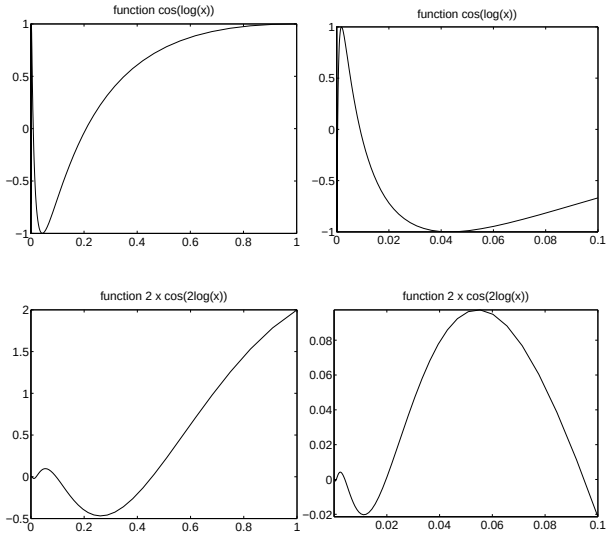
Use the transformation $x = -\ln t \Rightarrow t = e^{-x} \Rightarrow \frac{dx}{dt} = -\frac{1}{t}$.

Then, when $x = 0$, we have $t = 1$ and when $x = \infty$, we have $t = 0$.

Thus $\int_0^\infty \cos x e^{-x} dx = \int_1^0 \cos(-\ln t) t(-\frac{1}{t}) dt = \int_0^1 \cos(\ln t) dt$.

We now have a finite interval. However, we have created a singularity at the left endpoint. We can use an open rule to avoid the evaluation of the integrand at 0, or approximate the integral by $I_a = \int_a^1 \cos(\ln t) dt$, for a sufficiently small. Thus, we come back to the use of truncation.

Infinite (and semi-infinite) integrals



Moreover, the integrand oscillates infinitely often as $t \to 0$. Due to the oscillations, many points (with increasing density as $t \to 0$) are needed near 0 to properly capture the behaviour of the integrand. This forces an adaptive quadrature rule to use many function evaluations.

In the following, we test how well the MATLAB function `quadl` approximates I after the transformation, and the effect of the 'false' left endpoint.

```
>> a = tol/2
a =
   5.0000e-09
>> quadl('cos(log(x))', 1e-8,   1, tol) - Q
ans =
  -6.7109e-09
>> quadl('cos(log(x))', 1e-9,   1, tol) - Q
ans =
  -2.4136e-08
>> quadl('cos(log(x))', 1e-10,  1, tol) - Q
ans =
   4.6649e-08
>> quadl('cos(log(x))', 1e-11,  1, tol) - Q
ans =
  -3.8032e-09
>> [q, nfcn] = quadl('cos(log(x))', a, 1, tol/2);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  -5.3957e-09  378
```


Infinite (and semi-infinite) integrals

Attempt 2: Use the transformation

$x = -2\ln t \Rightarrow t = e^{-\frac{x}{2}} \Rightarrow t^2 = e^{-x} \Rightarrow \frac{dx}{dt} = -\frac{2}{t}$.

Then, when $x = 0$, we have $t = 1$ and when $x = \infty$, we have $t = 0$.

Thus $\int_0^\infty \cos x e^{-x} dx = \int_1^0 \cos(-2\ln t) t^2 (-\frac{2}{t}) dt = \int_0^1 2t \cos(2\ln t) dt$.

We now have a finite interval. We still have a singularity and infinite number of oscillations as $t \to 0$, but the amplitute is damped down by a factor of t. This is an easier integral to approximate by some (adaptive) quadrature rule, than the one obtained by the first attempt.

```
>> a = sqrt(tol/2)
a =
   7.0711e-05
>> quadl('2*x.*cos(2*log(x))', 1e-4,  1, tol) - Q
ans =
  -6.6061e-09
>> quadl('2*x.*cos(2*log(x))', 1e-5,  1, tol) - Q
ans =
   1.6572e-09
```

```
>> quadl('2*x.*cos(2*log(x))', 1e-6,  1, tol) - Q
ans =
   9.2978e-10
>> [q, nfcn] = quadl('2*x.*cos(2*log(x))',a,1,tol/2);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  -1.6697e-09  228
```


Infinite (and semi-infinite) integrals

Another example: Let $I = \int_0^\infty \cos^2(x^2)e^{-x}dx$.

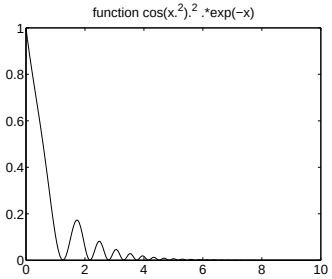
Attempt 1:

Use the transformation $x = -\ln t \Rightarrow t = e^{-x} \Rightarrow \frac{dx}{dt} = -\frac{1}{t}$.

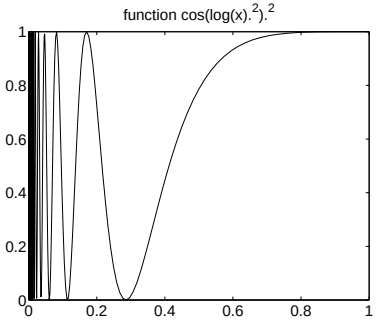
Then, when $x = 0$, we have $t = 1$ and when $x = \infty$, we have $t = 0$.

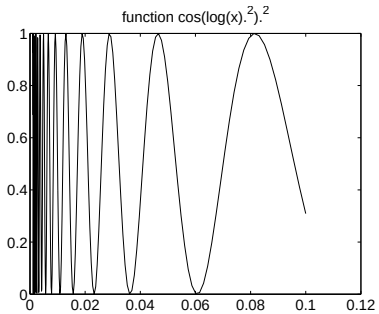
Thus $\int_0^\infty \cos^2 x^2 e^{-x}dx = \int_1^0 \cos^2((-\ln t)^2)t(-\frac{1}{t})dt = \int_0^1 \cos^2((\ln t)^2)dt$.

We now have a finite interval. Unfortunately, the integrand has a singularity and oscillates infinitely often as $t \to 0$. A computer program implementing some (adaptive) quadrature rule applied to the above integrand would have some trouble reaching a certain tolerance, due to the increasing density of points required near 0.



Infinite (and semi-infinite) integrals





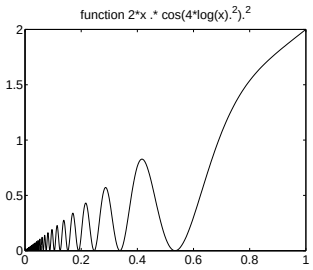
Infinite (and semi-infinite) integrals

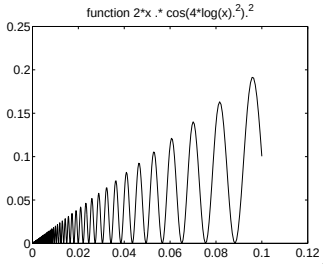
Attempt 2: Use the transformation

$x = -2\ln t \Rightarrow t = e^{-\frac{x}{2}} \Rightarrow t^2 = e^{-x} \Rightarrow \frac{dx}{dt} = -\frac{2}{t}$.

Then, when $x = 0$, we have $t = 1$ and when $x = \infty$, we have $t = 0$.

Thus $\int_0^\infty \cos^2 x^2 e^{-x}dx = \int_1^0 \cos^2((-2\ln t)^2)t^2(-\frac{2}{t})dt = \int_0^1 2t\cos^2(4(\ln t)^2)dt$.

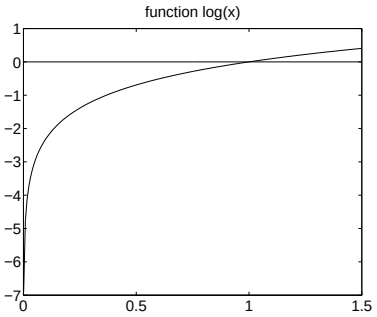




We now have a finite interval. We still have a singularity and infinite number of oscillations as $t \to 0$, but the amplitute is damped down by a factor of t. This is an easier integral to approximate by some (adaptive) quadrature rule, than the one obtained by the first attempt.

Singular integrals

When the integrand blows up at some point within the interval (including endpoints) of integration, we have a singular integral.



Example: $I = \int_0^1 \ln x dx$. The integral exists, i.e. the area above $\ln x$ between 0 and 1 is finite, but the integrand tends to infinity as $x \to 0$.

By integration by parts, we know that $\int \ln x dx = x \ln x - x$.

By de l' Hospital's rule we know that $\lim_{x\to 0} x \ln x = \lim_{x\to 0} \frac{\ln x}{1/x} = \lim_{x\to 0} \frac{1/x}{-1/x^2} = \lim_{x\to 0}(-x) = 0.$

Thus, $\int_0^1 \ln x dx = [x \ln x - x]_0^1 = -1 - 0 = -1.$

Note that it is quite common for the singularities to occur at the endpoint(s).

How to approximate such a singular integral?

(a) Use an *open* rule, e.g. a Gauss rule. Thus the evaluation of the integrand at the endpoints is avoided.

Singular integrals

(b) *Transformation*: Attempt to remove the singularity by a change of variable. It is important to note that not all changes of variable work well. Some may give rise to infinite integrals, or other undesirable behaviour.

Let $I = \int_a^b f(x)dx$. Apply the change of variable $x = p(t)$, $\frac{dx}{dt} = p'(t)$. Then, $I = \int_\alpha^\beta f(p(t))p'(t)dt$, where $p(\alpha) = a$ and $p(\beta) = b$.

How is $p(t)$ chosen so that the new integrand $f(p(t))p'(t)$ does not blow up in the new interval $[\alpha, \beta]$?

There is no general rule that is guaranteed to work in all cases, but a choice of $p(t)$ that often works is given by

$$p(t) = b - (b-a)u^2(2u+3) \text{ where } u = u(t) = \frac{t-b}{b-a}.$$

Then,

$$p'(t) = -(b-a)(2u(2u+3) + 2u^2),\ u(a) = -1, u(b) = 0,$$
$$p(a) = a,\ p(b) = b, \text{ thus } \alpha = a,\ \beta = b,$$
$$p'(a) = 0,\ p'(b) = 0.$$

Singular integrals

Then,

$$f(p(a))p'(a) = f(a) \cdot 0 \text{ and } f(p(b))p'(b) = f(b) \cdot 0.$$

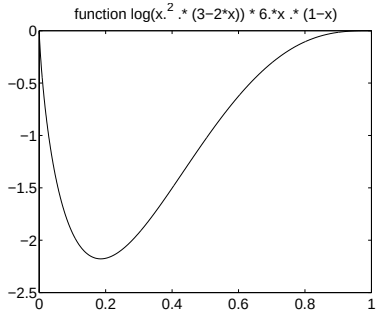
Thus, we have good chances to have cancelled the singularities at the endpoints.

Example: Consider $I = \int_0^1 \ln x dx$, and let

$$p(t) = 1 - 1(\frac{t-1}{1})^2(2\frac{t-1}{1} + 3) = 1 - (t-1)^2(2t+1) = \cdots = t^2(3-2t) \quad = 3t^2 - 2t^3.$$

Then, $I = \int_0^1 \ln(t^2(3-2t))6t(1-t)dt$,

which does not blow up in [0, 1].



Singular integrals

In the following, we test how well the MATLAB function `quadl` approximates I, using the above two transformations, and truncation from the left:

```
>> tol = 1e-8;
>> Q = -1;
>> a = 1e-5;
>> [q, nfcn] = ...
quadl('log(x.^2.*(3-2*x))*6.*x.*(1-x)',a,1,tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  7.7569e-009  168
>> [q, nfcn] = quadl('log(x)', a,    1, tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  1.2513e-004  318
>> [q, nfcn] = quadl('log(x)', 1e-6, 1, tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  1.4815e-005  318
>> [q, nfcn] = quadl('log(x)', 1e-7, 1, tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  1.7118e-006  348
```



```
>> [q, nfcn] = quadl('log(x)', 1e-8, 1, tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  1.9090e-007  378
>> [q, nfcn] = quadl('log(x)', 1e-9, 1, tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  2.1442e-008  408
>> [q, nfcn] = quadl('log(x)', 1e-10, 1, tol);
>> fprintf(' %12.4e %4d\n', q-Q, nfcn)
  3.0003e-010  408
```


Numerical integration in multiple dimensions

The one-dimensional quadrature rules studied so far can be extended for approximating multi-dimensional integrals. However, several issues arise, the most important being efficiency and the handling of arbitrary shape domains.

Consider, for example, the double integral $I = \int_a^b \int_c^d f(x,y)dydx$.

A Cartesian product rule based on a one-dimensional rule would be appropriate for such an integral. For example, the two-dimensional Simpson's rule:

$$I \approx \frac{(b-a)(d-c)}{36}\,(f(a,c)+4f(a,\frac{c+d}{2})+f(a,d)$$
$$+4f(\frac{a+b}{2},c)+16f(\frac{a+b}{2},\frac{c+d}{2})+4f(\frac{a+b}{2},d)$$
$$+f(b,c)+4f(b,\frac{c+d}{2})+f(b,d))$$

Note that the (2D) points of evaluation of the above 2D rule are formed by pairs of (1D) points of 1D rules. *All* combinations of the 1D points in the two dimensions are taken.

A composite version is straightforward. For an adaptive version, one needs to develop techniques to decide which direction(s) the refinement should proceed.

Numerical integration in multiple dimensions

If the domain is not rectangular, more care must be taken. Consider, for example, the double integral $I = \int\int_\Omega f(x,y)d\Omega$, where $\Omega \subset \mathbb{R}^2$.

A pair of (possibly adaptive) one-dimensional quadrature rules, one for the inner integral and one for the outer one would be appropriate for such an integral. Each time the outer rule does a function-integrand evaluation, the inner rule is called.

For example, if the outer rule is the x-rule and the inner one the y-rule, for a set of fixed x values, the inner rule goes through the respective x-lines (vertically), but the starting and ending points of each of those lines may be different.

Thus, the pair of one-dimensional quadrature rules works like a product rule, with 2D points formed by combining 1D points in the two dimensions, but *not* necessarily taking all combinations.

There are also interpolatory rules which are not based on products. Such rules are often based on a triangulation of the domain.

Numerical integration in multiple dimensions

For higher than two dimensions, similar ideas apply. However, efficiency issues arise, as the number of dimensions grows. Here is a table showing the number of function evaluations required by a Cartesian product rule, based on an n-point 1D rule and s panels in all dimensions. The table also shows the evolution of the error when the number of panels doubles, assuming 4th order of convergence.

	1D		2D		3D		4D		5D	
panels	fcnev	err	fcnev	err	fcnev	err	fcnev	err	fcnev	err
s	ns	e	$(ns)^2$	e	$(ns)^3$	e	$(ns)^4$	e	$(ns)^5$	e
$2s$	$2ns$	$e/16$	$4(ns)^2$	$e/16$	$8(ns)^3$	$e/16$	$16(ns)^4$	$e/16$	$32(ns)^5$	$e/16$

Clearly, the cost of improving the error by a certain factor increases rapidly with the dimension of the problem.

Numerical integration in multiple dimensions

An alternative for integrals in many dimensions is the Monte Carlo method. According to that, the function-integrand is sampled at N points randomly chosen in the domain, and the mean of the function values times the area (or volume, etc) of the domain is taken as an approximation to the integral. The error converges to 0 as $\frac{1}{\sqrt{N}}$, *independently* of the dimension. Here is a table showing the number of function evaluations required by a Cartesian product rule, and by Monte Carlo to achieve the respective errors.

	6D		8D		10D	
	fcnev	err	fcnev	err	fcnev	err
product rule	N $64N$	e_p $e_p/16$	N $256N$	e_p $e_p/16$	N $1024N$	e_p $e_p/16$
Monte Carlo	N $64N$	e_{mc} $e_{mc}/8$	N $256N$	e_{mc} $e_{mc}/16$	N $1024N$	e_{mc} $e_{mc}/32$

Clearly, the rate of cost increase over the rate of error reduction comes in favour of Monte Carlo, as the dimensions increase.