Often we want to integrate over infinite & semi-infinite domains. How do we deal with this?

Example: $$I = \int_0^\infty e^{-x} \cos^2 x^2 \, dx$$

One way is to simply truncate at some large number $A$:

$$\text{Say } I \approx Q = \int_0^A e^{-x} \cos^2 x^2 \, dx$$

This produces an error:

$$I - Q = \int_A^\infty e^{-x} \cos^2 x^2 \, dx < \int_A^\infty e^{-x} \, dx = e^{-A}$$

which is exponentially small!

Unfortunately, this is not usually true.

In general $$I - Q = \int_A^\infty f(x) \, dx$$

if $f(x) \cong x^{-3/2}$ at large $x$ then

$$\int_A^\infty x^{-3/2}\,dx = -\frac{1}{2}A^{-1/2}$$

Thus for $A = 10^4$, error is still $\mathcal{O}(10^{-2})$!

A better way is to transform the integral by mapping the domain.

Let $x = p(t)$

If we pick $p(t) = -\ln t$ or $\frac{t}{1-t}$

we map the domain $[0, \infty]$ onto $[0, 1]$

e.g. if $x = -\ln t$

then: $$\int_0^\infty f(x)\,dx = -\int_1^0 f(-\ln t)\,\frac{dt}{t}$$

$$= \int_0^1 f(-\ln t)\,\frac{dt}{t}$$

This can work very well:

if $f(x) = e^{-x}$ then:

$$\int_0^\infty e^{-x}\,dx = \int_0^1 e^{\ln t}\,\frac{dt}{t} = \int_0^1 dt = 1\,!$$

or rather poorly:

if $f(x) = \frac{1}{1+x^2}$

$$\int_0^\infty \frac{1}{1+x^2}\,dx = \int_0^1 \frac{1}{1+(\ln t)^2}\,\frac{1}{t}\,dt$$

which has an integrable singularity at the origin! In general, the choice of mapping depends on the function - you want to choose a mapping that captures the asymptotic behavior of the function!

You can also combine mapping with truncation: map the integrand so that it is exponentially small in $t$, and then truncate it.

So far we have just looked at one-dimensional quadrature. Often we need to integrate over 2 or more dimensions, e.g.

$$I = \iint_D f(x,y)\,dA$$

Say, calculate the lift on a wing by integrating the pressure over the wing surface!

We can write the n-point 2-D quadrature rule as:

$$I = \sum_{i=1}^{n} w_i f(x_i, y_i) + R_n$$

This formula is of polynomial degree $d$ if $R_n = 0$ for any bivariate polynomial of degree $d$ but non-zero for some polynomial of degree $d+1$

What is a bivariate polynomial?

It is a linear combination of terms

$x^p y^q$ where $p+q \leq 2$

For example, the most general polynomial of bivariate degree 2 is:

$$a x^2 + b y^2 + c x y + d x + e y + f = 0$$

There are 6 parameters, thus to integrate this exactly we require at least 3 nodes:

$$I = w_1 f(x_1, y_1) + w_2 f(x_2, y_2)$$

For each node we have 3 adjustable parameters: $x_i$, $y_i$, and $w_i$

The problem is that the location and weights are dependent on the domain shape!

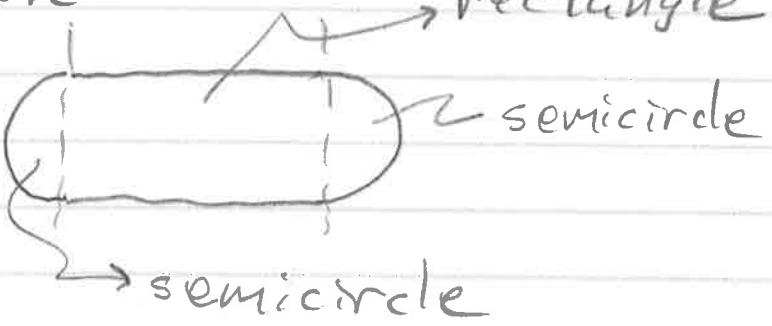
Optimum node locations and weights have been derived for rectangles, circles, and triangles.

What happens for more complex shapes?

There are a number of choices:

1) Map onto a simple domain via a change of variables. For example, a simple stretching maps an ellipse onto a circle, and polar coords. is a map of a circle onto a rectangle! Complex variables, through a technique called <u>conformal mapping</u> provides many useful mappings.

2) Break the domain of integration into a union of simpler shapes if possible. Example:




Solve for the integral on each of these bits separately

3) Approximate the domain with a union of triangles or other shapes:

error due to approximation.

The error gets smaller as the triangles are smaller (finer discretization)

4) Imbed the domain in a rectangle or other shape and let $f(x,y)=0$ outside the domain. This is very simple to do, but leads to a discontinuity in $f(x,y)$ - and hence to quadrature errors

---

In general, mapping is the best approach - if you can do it! As always, there is a tradeoff between the difficulty in setting the problem up (e.g. a complex mapping) and the amount of computer work required (e.g., dealing with discontinuities).

Suppose the domain is a rectangle (or a rectangular panel in a more complex domain)

We wish to solve:

$$\int_a^b \int_\alpha^\beta f(x,y)\,dx\,dy$$

Let $\int_\alpha^\beta f(x,y)\,dx = \sum_{i=1}^{n} w_i f(x_i,y) + R_1(y)$

and $\int_a^b g(y)\,dy = \sum_{j=1}^{n} p_j g(y_j) + R_2$

Thus:

$$\int_a^b \int_\alpha^\beta f(x,y)\,dx\,dy = \int_a^b \left[\sum_{i=1}^{n} w_i f(x_i,y) + R_1(y)\right] dx$$

$$= \sum_{i=1}^{n} w_i \int_a^b f(x_i,y)\,dy + \int_a^b R_1(y)\,dy$$

Let $f(x_i,y) = g_i(y)$

Thus:

$$I = \sum_{i=1}^{n} w_i \left[ \sum_{j=1}^{m} P_j f(x_i, y_j) + R_2 \right] + \int_a^b R_1 \, dy$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{m} w_i P_j f(x_i, y_j) + \underbrace{\sum_{i=1}^{n} w_i R_2 + \int_a^b R_1 \, dy}_{R \text{ (total error)}}$$

This is a product formula: essentially we evaluate the integral over one variable first and then over the other. $n$ & $m$ are, in general, different.

If the two one-dim. rules are of degree $d_1 = 2n-1$, $d_2 = 2m-1$ (e.g., we use Gaussian quadrature) then the product rule will be exact for polynomials of the form:

$$x^p y^q : \quad p \le d_1, \; q \le d_2$$

The product rule is exact for a bivariate polynomial of degree $\min(d_1, d_2)$

but it will be exact for some polynomials of degree $Q_1 + Q_2$.

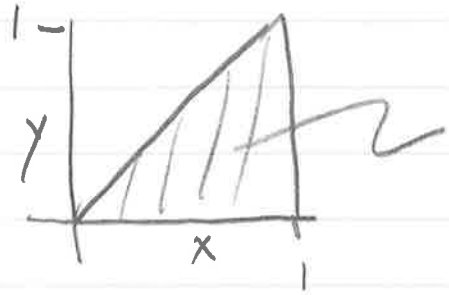
While this type of product rule is simple to code, it may not be the best choice. A 3x3 product rule using Gaussian quadrature is of bivariate degree 5 with 9 nodes.

There exists some rule of degree 5 with only 7 nodes! It's probably not worth trying to obtain it, however!

Mappings:

The most effective technique for doing multi-dim. quadrature is mapping. Usually we want to map a function onto a rectangle. This may do strange things.

Example: map a triangle onto a rectangle:




$0 \le x \le 1$

$0 \le y \le x$

We can map this onto a rectangle:

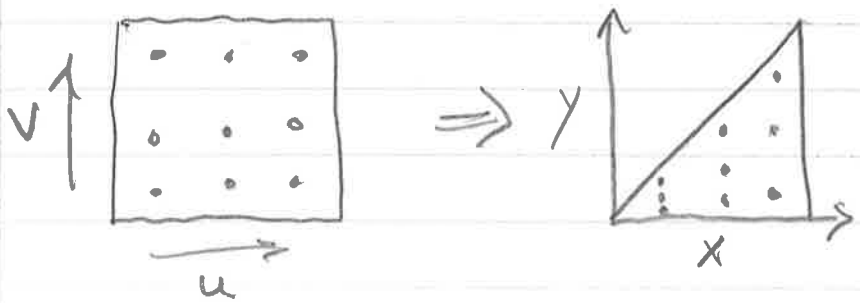
$u = x \quad , \quad v = y/x$

So that:

$$I \equiv \int_0^1 \int_0^x f(x,y)\, dy\, dx = \int_0^1 \int_0^1 f(u, uv)\, u\, du\, dv$$

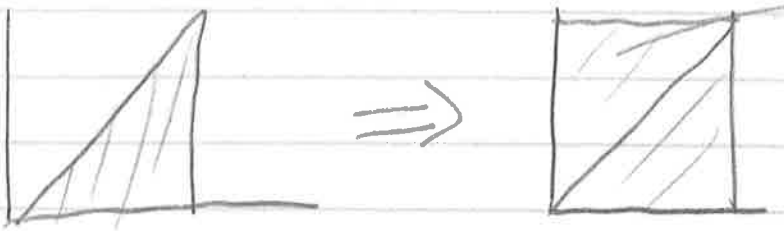
Note that $dy\,dx$ is transformed to $u\,du\,dv$!

This sort of transformation may not be desirable, because it will place the quadrature nodes unevenly in the original space:



Thus the function may be undersampled in some regions and oversampled in others. You should take this (and the spatial behavior of the function) into account when deciding on a mapping!

As an alternative, we could have continued the function by reflection



choose function to be the same as its mirror image!

Thus we pick $f(x,y)\Big|_{y>x} = f(y,x)$

and evaluate the integral over $0<x<1$ and $0<y<1$. The original integral will be half of this value.

While this leaves the points evenly distributed, it introduces a discontinuity in the derivative on the line $y=x$.

How can we apply adaptive quadrature to a 2-D integral?

If we have mapped to a rectangle, this is very simple:

We first evaluate over one variable, and then (in an outer loop or program) evaluate over the other!

Thus:

$$I = \int_a^b \left[ \int_\alpha^\beta f(x,y)\,dy \right] dx = \int_a^b g(x)\,dx$$

where $$g(x) = \int_\alpha^\beta f(x,y)\,dy$$

Suppose the quadrature routine over y returns:

$$g(x) = g^*(x) + E_y(x)$$

where $E_y(x)$ is the error and $g^*$ is the numerical result.

In the second integral (over x) we are not integrating $g(x)$, but rather the numerical result $g^*(x)$:

$$\int_a^b g^*(x)\,dx = \int_a^b [\, g(x) - E_y(x) \,]\,dx$$

and this will have some error as well!

Thus the combined result returns:

$$I = \int_a^b g(x)\,dx = I^* + \int_a^b E_y(x)\,dx + E_x$$

Thus both the integral over y and x contribute to the error.

We require (in adaptive quadrature) that the total error be less than some threshold $\varepsilon$:

$$\left| \int_a^b E_y(x)\,dx + E_x \right| < \varepsilon$$

This is accomplished by setting individual tolerances for both the x and y quadratures. Thus:

$$|E_y| < \varepsilon_y \;, \quad |E_x| < \varepsilon_x$$

where:

$$\left| (b-a)\,\varepsilon_y + \varepsilon_x \right| < \varepsilon$$

In general we require the inner tolerance to be smaller than the outer tolerance, say:

$$(b-a)\,\varepsilon_y \approx 0.1\,\varepsilon, \quad \varepsilon_x \approx 0.9\,\varepsilon$$

Feeding these values to the adaptive quadrature routines will result in the desired accuracy.

# Monte Carlo Integration

So far all quadrature methods we have looked at are polynomial based - we are approximating a function locally with a high degree polynomial. Monte Carlo integration is completely different.

Suppose we have:

$$I = \int_a^b f(x)\,dx \equiv (b-a)\langle f(x)\rangle$$

where $\langle f(x)\rangle$ is the average of $f$ over $[a, b]$.

The integral is thus just the average of the function times the width of the interval.

In Monte Carlo integration we want to compute this average!

We just pick $N$ points chosen at random over the domain and then average the corresponding function values.

$$I \approx \Theta_N \equiv (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

As $N$ goes to infinity, the error in $\Theta_N$ will go to zero.

What is the error? For large $N$, $\Theta_N$ approximates a normal distribution.

Thus:

$$E(\Theta_N) = I$$

$$\sigma^2_{\Theta_N} = E\left\{ \frac{(b-a)^2}{N} \frac{1}{N}\sum_{i=1}^{N} (f(x_i) - \bar{f})^2 \right\}$$

$$= \frac{(b-a)^2}{N}\sigma_f^2$$

where $\sigma_f^2$ is the variance of $f(x)$ over the interval.

The problem with this method is that it converges <u>very</u> slowly with $N$.

Even the Trapezoidal rule has an error which goes as $\frac{1}{N^2}$, while Monte Carlo integration error goes as $\sim \frac{1}{N^{1/2}}$!

To get 3 significant digits, you need around $10^6$ points.

The primary advantage of this procedure is in multi-dimensional integrals:

$$I = \int_D f(\underset{\sim}{x})\, dV$$

where $\underset{\sim}{x}$ is an $m$-dimensional vector. If $m$ is large, this is difficult to code using polynomial based quadrature. Also, if (say) $m = 12$ and we use just 3 point quadrature in each dimension, we require over 500,000 points! with this number of points Monte Carlo integration may well be more accurate!

The most important use is in simulation of random media and related topics.

You choose a number of random configurations of particles distributed in a matrix (say) and calculate effective properties such as thermal conductivity or dielectric constant for each configuration. You then just average the values together!

Monte Carlo integration was originally developed by von Neumann to solve the neutron diffusion problem important in designing the A-bomb. We use similar approaches for measuring shear-induced diffusion in suspensions.

Prof. Maginn uses these techniques to examine shape/size selectivity of zeolite catalysts.