Numerical Integration

Another (better) term: quadrature.

Problem: Given finite number of function values $f(x_i)$, $x_i \in [a,b]$ or the function $f(x)$ can be evaluated any $x \in [a,b]$, calculate

$$I(f) = \int_a^b f(x)dx.$$

Partition

$$a = x_1 < x_2 < \cdots < x_{n+1} = b,$$

and let $h_i = x_{i+1} - x_i$. Then

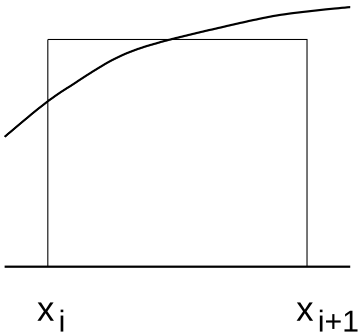
$$I(f) = \sum_{i=1}^{n} I_i \qquad I_i = \int_{x_i}^{x_{i+1}} f(x)dx$$

Quadrature rule: Approximation of I_i
Composite quadrature rule: Approximation of $I(f)$ as a sum of I_i

1 Rectangle Rule

Piecewise constant (degree zero polynomial)

$$y_i = \frac{x_i + x_{i+1}}{2}, \qquad i = 1, ..., n$$



The rectangle rule is:

$$I_i \approx h_i f(y_i)$$

and the composite rectangle rule is:

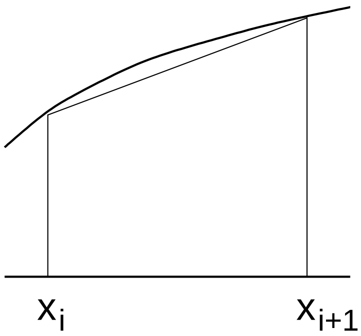
$$R(f) = \sum_{i=1}^{n} h_i f(y_i)$$

Function evaluations: n

2 Trapezoid Rule

Piecewise linear approximation (degree one polynomial)

$$I_i \approx h_i \frac{f(x_i) + f(x_{i+1})}{2}$$



Composite trapezoid rule:

$$T(f) = \sum_{i=1}^{n} h_i \frac{f(x_i) + f(x_{i+1})}{2}$$

Function evaluations: $n + 1$.
Convergence

$$\lim_{\max h_i \to 0} T(f) = I(f).$$

3 Error Estimates

How accurate are they (rectangle and trapezoid)? We consider their truncation errors.

Rectangle rule

Taylor expansion about $y_i = (x_i + x_{i+1})/2$:

$$f(x) = f(y_i) + \sum_{p=1}^{\infty} \frac{(x - y_i)^p}{p!} f^{(p)}(y_i).$$

Note that

$$\int_{x_i}^{x_{i+1}} (x - y_i)^p dx = \begin{cases} \frac{h_i^{p+1}}{(p+1)2^p} & \text{even p} \\ 0 & \text{odd p} \end{cases}$$

Then

$$\int_{x_i}^{x_{i+1}} f(x)dx = h_i f(y_i) + \frac{1}{24} h_i^3 f''(y_i) + \frac{1}{1920} h_i^5 f^{iv}(y_i) + \cdots$$

when h_i is small, the error

$$I(f) - R(f) \approx \frac{1}{24} \sum_{i=1}^{n} h_i^3 f''(y_i) + \frac{1}{1920} \sum_{i=1}^{n} h_i^5 f^{iv}(y_i).$$

Trapezoid rule

Substituting $x = x_i$ and $x = x_{i+1}$ in the Taylor expansion, we have

$$\begin{aligned} f(x_i) &= f(y_i) + \sum_{p=1}^{\infty} (-1)^p \frac{h_i^p}{2^p p!} f^{(p)}(y_i) \\ f(x_{i+1}) &= f(y_i) + \sum_{p=1}^{\infty} \frac{h_i^p}{2^p p!} f^{(p)}(y_i) \end{aligned}$$

Thus

$$\frac{f(x_i) + f(x_{i+1})}{2} = f(y_i) + \frac{1}{8} h_i^2 f''(y_i) + \frac{1}{384} h_i^4 f^{iv}(y_i) + \cdots$$

Recall that

$$\int_{x_i}^{x_{i+1}} f(x)dx = h_i f(y_i) + \frac{1}{24} h_i^3 f''(y_i) + \frac{1}{1920} h_i^5 f^{iv}(y_i) + \cdots$$

Using the above two equations, we have

$$\int_{x_i}^{x_{i+1}} f(x)dx = h_i \frac{f(x_i) + f(x_{i+1})}{2} - \frac{1}{12} h_i^3 f''(y_i) - \frac{1}{480} h_i^5 f^{iv}(y_i) + \cdots$$

The error is

$$I(f) - T(f) \approx -\frac{1}{12}\sum_{i=1}^{n} h_i^3 f''(y_i) - \frac{1}{480}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) + \cdots$$

Remarks

- Usually rectangle rule (degree zero) is more accurate than trapezoid rule (degree one). Surprise?
- Use $I(f) - R(f) \approx \frac{1}{3}(T(f) - R(f))$ to estimate the error in $R(f)$. Similarly, $I(f) - T(f) \approx \frac{2}{3}(R(f) - T(f))$. (But it is possible that $R(f) - T(f) = 0$ whereas $I(f) - R(f) \neq 0$).
- When each h_i is cut in half, $I(f) - R_{\frac{1}{2}}(f) \approx \frac{1}{4}(I(f) - R(f))$. (Why?) Similarly for the trapezoid rule. Doubling the number of panels in either the retangle rule or the trapezoid rule, it can be expected to roughly quadruple the accuracy. This can be used to estimate the error as well as improving the accuracy. (How?)

Example. Compute

$$\int_0^{\frac{\pi}{2}} \sin(x)dx$$

using the trapezoid rule.

m	QCTrap(sin, 0.000, 1.571, m)	error
3	0.9480594489685199	5.2e-2
5	0.9871158009727753	1.3e-2
9	0.9967851718861696	3.2e-3
17	0.9991966804850722	8.0e-4

This table shows that when the number of panels is doubled the accuracy is roughly quadrupled.

4 Simpson's Rule

Recall that

$$R(f) = I(f) - \frac{1}{24}\sum_{i=1}^{n} h_i^3 f''(y_i) - \frac{1}{1920}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) + \cdots$$

and

$$T(f) = I(f) + \frac{1}{12}\sum_{i=1}^{n} h_i^3 f''(y_i) - \frac{1}{480}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) + \cdots$$

Combining the above two equations to improve the accuracy (cancelling the $O(h_i^3)$ term),

$$\begin{aligned} S(f) &= \frac{2}{3}R(f) + \frac{1}{3}T(f) \\ &= I(f) + \frac{1}{2880}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) + \cdots \end{aligned}$$

Simpson's rule:

$$I_i = \frac{2}{3}h_i f(\frac{x_i + x_{i+1}}{2}) + \frac{1}{3}h_i \frac{f(x_i) + f(x_{i+1})}{2}$$

Composite Simpson's rule:

$$S(f) = \sum_{i=1}^{n} \frac{1}{6}h_i \left[f(x_i) + 4f(\frac{x_i + x_{i+1}}{2}) + f(x_{i+1}) \right]$$

Function evaluations: $2n + 1$

Error

$$I(f) - S(f) = -\frac{1}{2880}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) + \cdots$$

Remarks

- Simpson's rule may also be derived by using piecewise quadratic approximation (degree two).
- Actually, Simpson's rule is exact for cubic function (one extra order of accuracy), since the error term involves the fourth derivatives.

- Doubling the number of panels in Simpson's rule can be expected to reduce the error by roughly the factor of $\frac{1}{16}$.

A general technique: Richardson's extrapolation

Combining two approximations (e.g., $R(f)$ and $T(f)$) which have similar error terms to achieve a more accurate approximation (e.g., $S(f)$).

Example. Combining $S(f)$ and $S_{\frac{1}{2}}(f)$ to obtain an approximation which has error of order h_i^7. (What are the weights?) This gives Romberg quadrature.

A frame work: Newton-Cotes rules.

Use a polynomial interpolant $p(x)$ (Lagrange for example) to approximate $f(x)$

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx.$$

The m-point Newton-Cotes rule

$$Q_{\mathrm{NC}(m)} = \int_a^b p_{m-1}(x)dx$$

where p_{m-1} (of degree $m-1$) interpolates at

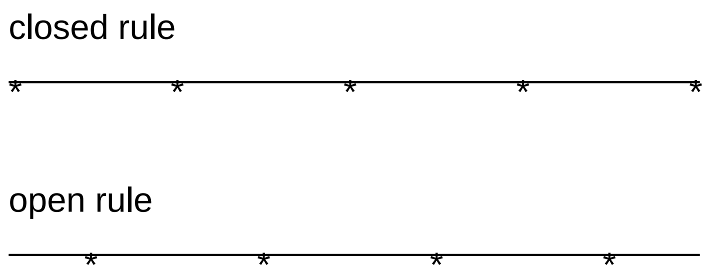
$$x_i = a + \frac{i-1}{m-1}(b-a), \quad i = 1, ..., m,$$

evenly spaced m points.

Special cases:

Trapezoidal rule, $m = 2$

Simpson's rule, $m = 3$



Rectangle rule is an open rule with $m = 1$, $x_i = a+(i-(1/2))h$, $h = (b-a)/m$, $i = 1, ..., m$.

- For $m = 3, 5, 6, 7, ...$, the open rules can undermine the numerical stability. (Negative weights can cause numerical cancellation.)

5 Adaptive Quadrature

What is adaptive quadrature? Given a predetermined tolerance ϵ, the algorithm automatically determines the panel sizes so that the computed approximation Q satisfies

$$\left| Q - \int_a^b f(x)dx \right| < \epsilon$$

Software interface: `quad(fname, a, b, tol)`

Why adaptive? The algorithm uses large panel sizes for smooth parts and small panel sizes for the parts where the function changes rapidly. Thus the prescribed accuracy is attained at as small a cost in computing time. (Number of function evaluations.)

The basic idea:

1. Compute two approximations, applying Simpson's rule

   one-panel formula

   $$P_i = \frac{h_i}{6}\left[f(x_i) + 4f(x_i + \frac{h_i}{2}) + f(x_i + h_i) \right]$$

   two-panel formula

   $$\begin{aligned} Q_i &= \frac{h_i}{12}\left[f(x_i) + 4f(x_i + \frac{h_i}{4}) + 2f(x_i + \frac{h_i}{2}) \right. \\ &\qquad \left. + 4f(x_i + \frac{3h_i}{4}) + f(x_i + h_i) \right] \end{aligned}$$

   - From P_i to Q_i, we need only two function evaluations $f(x_i + \frac{h_i}{4})$ and $f(x_i+\frac{3h_i}{4})$. The function values $f(x_i)$, $f(x_i+\frac{h_i}{2})$, and $f(x_i+h_i)$ evaluated in P_i are reused.

- Q_i can be viewed as the sum of two P's from two subintervals of length $h_i/2$

2. Compare P_i and Q_i to obtain an estimate of their accuracy.

$$\begin{aligned} I_i - P_i &= c h_i^5 f^{iv}(x_i + \frac{h_i}{2}) + \cdots \\ I_i - Q_i &= c\left(\frac{h_i}{2}\right)^5 \left[f^{iv}(x_i + \frac{h_i}{4}) + f^{iv}(x_i + \frac{3h_i}{4}) \right] + \cdots \end{aligned}$$

Using

$$f^{iv}(x_i + \frac{h_i}{4}) + f^{iv}(x_i + \frac{3h_i}{4}) \approx 2 f^{iv}(x_i + \frac{h_i}{2}),$$

we have

$$I_i - Q_i = 2c\left(\frac{h_i}{2}\right)^5 f^{iv}(x_i + \frac{h_i}{2}) + \cdots$$

Thus

$$I_i - Q_i = \frac{1}{2^4}(I_i - P_i) + \cdots$$

The above equation gives

$$I_i = \frac{2^4}{2^4 - 1}(Q_i - \frac{P_i}{2^4}) + \cdots,$$

which implies

$$I_i - Q_i = \frac{1}{2^4 - 1}(Q_i - P_i) + \cdots$$

Now the accuracy of Q_i is expressed in terms of $Q_i - P_i$.

Scheme

Bisect each subinterval until

$$\frac{1}{2^4 - 1}|Q_i - P_i| \leq \frac{h_i}{b - a}\epsilon,$$

then

$$\left| \int_a^b f(x)dx - \sum_{i=1}^n Q_i \right| \leq \frac{1}{2^4 - 1} \sum_{i=1}^n |Q_i - P_i| \leq \frac{\epsilon}{b - a} \sum_{i=1}^n h_i = \epsilon.$$

Algorithm. Adaptive quadrature.

```
[I, err] = AdaptQuad(fname,a,b,tol,maxLev)

if maxLev==0
   too many levels of recursion, quit;
end
compute one-panel quadrature R1;
compute two-panel quadrature R2;
use R1 and R2 to estimate error in R2;
if the estimated error < tol
   return R2 and estimated error;
else
   [I1, err1] =
     AdaptQuad(fname,a,mid,tol/2,maxLev-1);
   [I2, err2] =
     AdaptQuad(fname,mid,b,tol/2,maxLev-1);
   I = I1 + I2;
   err = err1 + err2;
end
```

Example. Compute

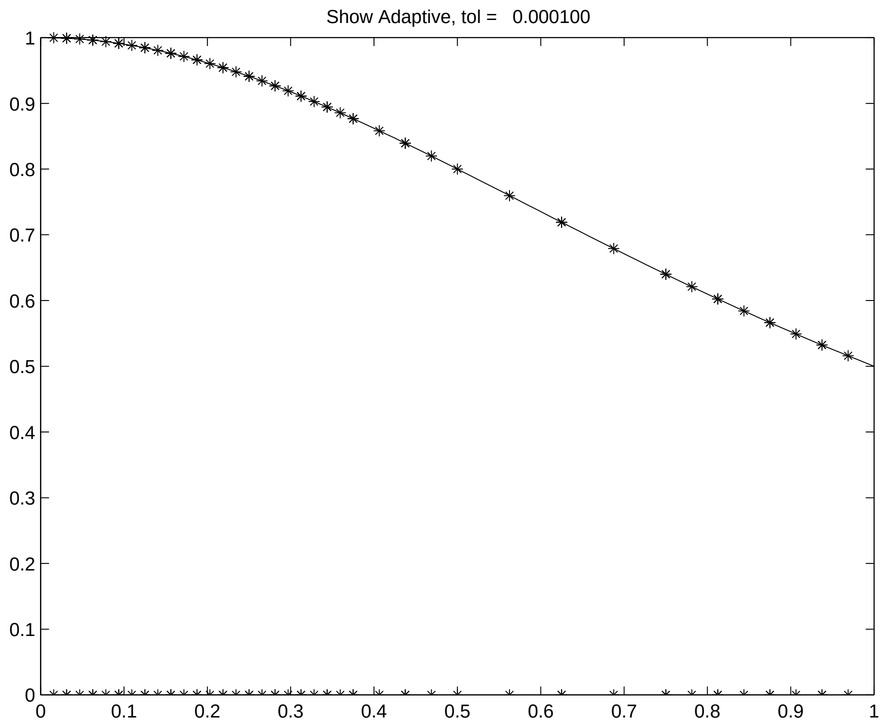
$$\frac{\pi}{4} = \int_0^1 \frac{1}{1+x^2} dx$$

using the adaptive rectangle rule.

`AdaptQRec('datan',0,1,0.0001,10)`: 0.785396

estimated error: 7.28×10^{-5}

actual error: 2.23×10^{-6}



6 Gaussian Quadrature

Newton-Cotes rules require the equal space of the knots x_i. Gaussian quadrature is another class of methods which removes the restriction. The basic idea is to use a weighted combination of function values at the knots to approximate the integration

$$I = \int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i).$$

The weights w_i and the knots x_i are chosen so that the error

$$E_n(f) = I - \sum_{i=1}^{n} w_i f(x_i)$$

equals zero for as high a degree polynomial as possible. Suppose, $f(x) = a_0 + a_1 x + \cdots + a_m x^m$, then

$$E_n(f) = \sum_{k=0}^{m} a_k \left(\int_{-1}^{1} x^k dx - \sum_{i=1}^{n} w_i x_i^k \right) = \sum_{k=0}^{m} a_k E_n(x^k).$$

When $n = 1$, there are two free parameters w_1 and x_1. We set

$$E_1(1) = \int_{-1}^{1} 1dx - w_1 = 0 \quad \text{and} \quad E_1(x) = \int_{-1}^{1} xdx - w_1 x_1 = 0,$$

which implies $w_1 = 2$ and $x_1 = 0$, i.e.,

$$\int_{-1}^{1} f(x)dx \approx 2f(0),$$

the rectangle rule and $E_1(f) = 0$ for a polynomial $f(x)$ of degree one.

When $n = 2$, solving

$$\begin{aligned} w_1 + w_2 &= 2 \\ w_1 x_1 + w_2 x_2 &= 0 \\ w_1 x_1^2 + w_2 x_2^2 &= \frac{2}{3} \\ w_1 x_1^3 + w_2 x_2^3 &= 0 \end{aligned}$$

we get

$$w_1 = w_2 = 1 \quad \text{and} \quad x_i = \pm \frac{\sqrt{3}}{3}.$$

For general n, it involves solving a nonlinear system of $2n$ equations. To alleviate this problem, instead of simple polynomials 1, x,..., x^m, we choose orthogonal polynomials $\phi_n(x)$ with respect to a weight function $w(x) \geq 0$, i.e.,

$$\int_a^b w(x)\phi_i(x)\phi_j(x)dx = 0, \quad i \neq j,$$

and take the zeros of $\phi_n(x)$ as the knots. We can find the weights w_i, so that

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

is exact for the polynomial $f(x)$ of degree $2n - 1$.

Example. Legendre polynomials defined by

$$P_0(x) = 1, \quad \text{and} \quad P_n(x) = \frac{(-1)^n}{2^n n!} \frac{d^n}{dx^n} \left[(1 - x^2)^n\right] \quad \text{for} \quad n \geq 1,$$

are orthogonal on $[-1, 1]$ with respect to $w(x) = 1$. For example,

$$P_0(x) = 1, \quad P_1(x) = x, \quad P_2(x) = \frac{1}{2}(3x^2 - 1), \quad \cdots$$

Gauss-Legendre knots and weights.

n	x_i	w_i
2	$\pm.5773502692$	1.0
3	$\pm.7745966692$	0.5555555556
	0.0	0.8888888889
4	$\pm.8611363116$	0.3478546451
	$\pm.3399810436$	0.6521451549

7 2D Quadrature

Consider a 2D integral

$$I = \int_a^b \int_c^d f(x, y) dy dx$$

and let

$$g(x) = \int_c^d f(x, y) dy.$$

Applying the composite trapezoid rule to

$$\int_a^b g(x) dx$$

we get the numerical integration

$$\sum_{i=1}^{m-1} \frac{g(x_i) + g(x_{i+1})}{2} h_x.$$

In vector form the above equals

$$h_x \mathbf{w}_x^{\mathrm{T}} \begin{bmatrix} g(x_1) \\ \vdots \\ g(x_m) \end{bmatrix}$$

where $\mathbf{w}_x^{\mathrm{T}} = [1/2, 1, ..., 1, 1/2]$.

Again, applying the composite trapezoid rule to each $g(x_i)$, we get

$$g(x_i) \approx \sum_{j=1}^{n-1} \frac{f(x_i, y_j) + f(x_i, y_{j+1})}{2} h_y.$$

In vector form

$$g(x_i) \approx h_y [f(x_i, y_1), ..., f(x_i, y_n)] \mathbf{w}_y$$

where $\mathbf{w}_y = [1/2, 1, ..., 1, 1/2]^{\mathrm{T}}$.

Finally, we have the numerical integration

$$Q = h_x h_y \mathbf{w}_x^{\mathrm{T}} F \mathbf{w}_y$$

where

$$F = \begin{bmatrix} f(x_1, y_1) & \cdots & f(x_1, y_n) \\ \vdots & \cdots & \vdots \\ f(x_m, y_1) & \cdots & f(x_m, y_n) \end{bmatrix}.$$

Example. Integrand function

$$f(x, y) = e^{-(x^2+2y^2)/4}$$

$a = -2$, $b = 2$, $c = -1$, $d = 1$, $m = n$

Subintervals	Integral	Relative Time
2	4.39508052	1.00
4	4.93166539	0.97
8	5.06690648	1.00
16	5.10073164	1.62
32	5.10918854	1.75
64	5.11130280	4.13
128	5.11183136	13.21
256	5.11196350	44.88

Summary

- Composite quadrature rules: Rectangle rule, trapezoid rule, Simpson's rule.
- Richardson's extrapolation technique: Combining two quadrature rules which have similar error terms to achieve a more accurate quadrature rule by cancelling the leading error term. This technique can also be used to estimate errors.
- Adaptive quadrature: Using error estimates to determine the panel sizes so that the computed approximation satisfies a predetermined tolerance.

- Gaussian quadrature: Use the zeros of an orthogonal polynomial and weights to achieve high accuracy.
- 2D quadrature: Formulation of the problem.

Software packages

IMSL qdag, qdags, twodq, qand

MATLAB quad, quad8

NAG d01ajf, d01daf, d01fcf

References

Kendall E. Atkinson, *An Introduction to Numerical Analysis*, 2nd Edition, John Wiley & Sons, 1989.

George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, Inc., 1977.