Numerical analysis (3/7): Derivation, Integration

University of Luxembourg

Philippe Marchner

Siemens Digital Industries Software, France

November 2nd, 2023

Outline

Outline

Introduction

The problem

We want to compute the derivative of a function f at a point x_i

$$\lim_{h\to 0} f'(x_i) = \frac{f(x_i+h)-f(x_i)}{h}$$

However f is not known explicitly but

1. by its values on a discrete set (for sufficiently close points so derivative makes sense)
2. by an algorithm or formula that (at least theoretically) allows to compute it at each point.

We will look for an approximation of this derivative number and try to get an error estimate

Numerical differentiation allows us to find an **estimate** of the derivative by only using the values of f at a finite number of points.

Outline

Derivation by Taylor approximation

Taylor-Young formula (1st order)

If f is of class $\mathcal{C}^2$ in $[x_i, x_{i+1}]$

$$f(x_i + h) = f(x_i) + hf'(x_i) + h^2\frac{f''(\xi)}{2}, \quad \xi \in [x_i, x_i + h]$$
$$\Rightarrow f'(x_i) = \frac{f(x_i + h) - f(x_i)}{h} + \mathcal{O}(h)$$

The *discretization error* is $\mathcal{O}(h)$, it is first order accurate

Remark: we can also derive $f'(x_i) = \frac{f(x_i)-f(x_i-h)}{h} + \mathcal{O}(h)$

Derivation by Taylor approximation

Rounding-error

$$\mathrm{fl}(f(x_i+h)-f(x_i)) = (f(x_i+h)-f(x_i))(1+\varepsilon), \quad |\varepsilon| \le \eta$$
$$\Rightarrow \left| f'(x_i) - \mathrm{fl}\left(\frac{f(x_i+h)-f(x_i)}{h}\right)\right| \le \frac{hM}{2} + \frac{2\eta}{h}, \quad M = \max_{\xi\in[x_i,x_i+h]} f''(\xi)$$

If h is chosen too small, the error grows as $\mathcal{O}(h^{-1})$! (see exercise 1st session)

Optimal step size - forward difference

The optimal step size is the minimum of $E(h) = \frac{hM}{2} + \frac{2\eta}{h}$, $h^* = 2\sqrt{\frac{\eta}{M}}$

Outline

Derivation by interpolation

In the sequel, we assume that f is known or can be evaluated at the points $\dots, x_{i-2}, x_{i-1}, x_i$, $x_{i+1}, x_{i+2}, \dots$ that we assume to be close and we denote by $h_i = x_{i+1} - x_i$.

Principle

To obtain an approximation of $f'(x_i)$:

- we approximate f in a neighborhood of x_i by a function which is "simple" to derive
- to this end, we will use an interpolation polynomial near x_i !
- and finally the obtained formulas will differ with respect to the number of points chosen to write the interpolation polynomial (generally 2 or 3)

Derivation by interpolation

Two-points formulas

if one uses the interpolation polynomial for the two points x_i, x_{i+1} :

$$P(x) = f(x_i) + f[x_i, x_{i+1}](x - x_i)$$

we then have $P'(x_i) = f[x_i, x_{i+1}]$, leading to

Right decentered formula

$$f'(x_i) \simeq \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

Derivation by interpolation

Two-points formulas

if one uses the interpolation polynomial for the two points x_{i-1}, x_i :

$$P(x) = f(x_{i-1}) + f[x_{i-1}, x_i](x - x_{i-1})$$

we then have $P'(x_i) = f[x_{i-1}, x_i]$

Left decentered formula

$$f'(x_i) \simeq \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}$$

Derivation by interpolation

Three-points formulas

if one uses the interpolation polynomial for the three points x_{i-1}, x_i, x_{i+1} :

$$P(x) = f(x_{i-1}) + f[x_{i-1}, x_i](x - x_{i-1}) + f[x_{i-1}, x_i, x_{i+1}](x - x_{i-1})(x - x_i)$$

we then have

$$P'(x_i) = f[x_{i-1}, x_i] + f[x_{i-1}, x_i, x_{i+1}](x_i - x_{i-1})$$

leading to

Centered formula

$$f'(x_i) \simeq f(x_{i+1})\frac{h_{i-1}}{h_i(h_{i-1}+h_i)} + f(x_i)\left(\frac{1}{h_{i-1}} - \frac{1}{h_i}\right) - f(x_{i-1})\frac{h_i}{h_{i-1}(h_{i-1}+h_i)}$$

Derivation by interpolation

Centered formula for equispaced points

For equispaced points i.e. $h_{i-1} = h_i = h$, the centered formula simplifies

$$f'(x_i) \simeq \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$

Remarks

- The above formula is nothing else than the mean of the two previous decentered formula in the case of equispaced points.
- More generally if $P_n(x) = \sum_{j=0}^{n} f(x_j)\, L_j(x)$, we have $P_n'(x_i) = \sum_{j=0}^{n} f(x_j)\, L_j'(x_i)$
  $\rightarrow$ in principle, we can improve the approximation with a higher degree polynomial

Error estimates

Remark:

- By definition of the derivative, the right/left decentered formula tend towards $f'(x_i)$, and we have seen by Taylor-Young formula that they are first order accurate

If f is $\mathcal{C}^3$ on $[x_{i-1}, x_{i+1}]$, then we have in the case of equispaced points $h_{i-1} = h_i = h$

$$f(x_{i+1}) = f(x_i) + hf'(x_i) + \frac{h^2}{2} f''(x_i) + \frac{f^{(3)}(\xi)}{6} h^3 \quad \text{with} \quad \xi \in [x_i, x_{i+1}]$$

$$f(x_{i-1}) = f(x_i) - hf'(x_i) + \frac{h^2}{2} f''(x_i) - \frac{f^{(3)}(\eta)}{6} h^3 \quad \text{with} \quad \eta \in [x_{i-1}, x_i]$$

so we can prove that

Error estimates - centered formula - equispaced points

$$\left| f'(x_i) - \frac{f(x_{i+1}) - f(x_{i-1})}{2h} \right| \leq \frac{M_3}{6} h^2, \quad M_3 = \max_{x \in [x_{i-1}, x_{i+1}]} f^{(3)}(x)$$

Outline

Error estimates

General case with n points: $a < x_0 < x_1 < \cdots < x_n < b$

If $P_n(x) = \sum_{k=0}^{n} f(x_k)\, L_k(x)$, the polynomial interpolation error is

$$E(x) = f(x) - P_n(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{k=0}^{n} (x - x_k), \quad \xi_x \in [a, b]$$

For the derivative at x_i, we have $P_n'(x_i) = \sum_{k=0}^{n} f(x_k)\, L_k'(x_i)$ and

$$E'(x_i) = f'(x_i) - P_n'(x_i) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{k=0, k\neq i}^{n} (x_i - x_k)$$

Remarks:

- An approximation with $n+1$ points has an error $\mathcal{O}(h^n)$
- The approximation is exact for polynomials of degree $\leq n$
- Centered formulas lead a lower error constant

Outline

Higher-order derivatives

- we use the same principle: we approximate f by an interpolation polynomial near x_i
- we must take care about the fact that the degree of the interpolation and the number of points must be large enough so that its n-th derivative is not zero!
- for example, for the second-order derivative, we need three points which in general are x_{i-1}, x_i, x_{i+1}, leading to

Second-order derivative - equispaced points

$$f''(x_i) \simeq \frac{f(x_{i-1}) - 2f(x_i) + f(x_{i+1})}{h^2}$$

Exercise: Derive the formula using the interpolating polynomial

Error estimates - second-order derivative - equispaced points

Now, if f is $\mathcal{C}^4$ on $[x_{i-1}, x_{i+1}]$, then we have

$$f(x_{i+1}) = f(x_i) + hf'(x_i) + \frac{h^2}{2} f''(x_i) + h^3 \frac{f^{(3)}(x_i)}{6} + \frac{f^{(4)}(\xi)}{24} h^4 \quad textwith \quad \xi \in [x_i, x_{i+1}]$$

$$f(x_{i-1}) = f(x_i) - hf'(x_i) + \frac{h^2}{2} f''(x_i) - h^3 \frac{f^{(3)}(x_i)}{6} + \frac{f^{(4)}(\eta)}{24} h^4 \quad \text{with} \quad \eta \in [x_{i-1}, x_i]$$

and hence

Error estimates - centered formula - equispaced points

$$\left| f''(x_i) - \frac{f(x_{i-1}) - 2f(x_i) + f(x_{i+1})}{h^2} \right| \leq \frac{M_4}{12} h^2, \quad M_4 = \max_{x \in [x_i, x_{i+1}]} f^{(4)}(x)$$

Remark: the general theory may also be developed.

- A k-th derivative requires $k+1$ points, and will have $\mathcal{O}(h)$ or $\mathcal{O}(h^2)$ error (with symmetry)
- Each extra points can improve the order by one

Outline

An example of application: finite difference schemes

Consider the one-dimensional **boundary-value problem**

$$\begin{cases} -u''(x) = f(x) & 0 < x < 1 \\ u(0) = u(1) = 0 \end{cases}$$

- we show that if f is continuous on $[0, 1]$ then this problem admits a unique solution $u \in \mathcal{C}^2([0, 1])$
- the finite difference method consists in computing an approximate solution at the points $x_1, \ldots, x_n$ for a given subdivision of the interval $[0, 1]$
- to simplify, we assume that we have a uniform subdivision

$$x_i = ih \quad \text{for all } 0 \leq i \leq n+1 \quad \text{with} \quad h = \frac{1}{n+1}$$

An example of application: finite difference schemes

- hence if we set u_i as the approximate value of $u(x_i)$ at point x_i, we are led to determine the vector
$$u_h = (u_1, \dots, u_n)$$
- since we have $-u''(x_i) = f(x_i)$, for any $1 \leq i \leq n$, if one replaces $u''(x_i)$ by
$$u''(x_i) \simeq \frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{h^2}$$
then the vector u_h is solution to the linear system
$$\frac{-u_{i-1} + 2u_i - u_{i+1}}{h^2} = f_i \quad \forall\, 1 \leq i \leq n$$
with $f_i = f(x_i)$ and $u_0 = u(0) = 0$, $u_{n+1} = u(1) = 0$ to satisfy the boundary conditions

An example of application: finite difference schemes

- then we solve the linear system

$$\frac{1}{h^2} A u_h = f_h$$

where $f_h = (f_1, \dots, f_n)$ and

$$A = \begin{bmatrix} 2 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}.$$

- we can check that A is a symmetric definite-positive tridiagonal matrix
- the system admits one and only one solution

Outline

Introduction

Being given a continuous function $f : [a, b] \to \mathbb{R}$, we try to compute the integral

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

Once again, we do not want to compute exactly the value of this integral but only an approximate value I_{app} with an *a priori* given accuracy ϵ i.e. such that

$$|I - I_{app}| < \epsilon$$

Remarks

we need to use numerical integration when

- we do not know the explicit form of the primitives of f : examples

$$f(x) = e^{-x^2}, \quad f(x) = \frac{e^x}{x}, \quad f(x) = \frac{1}{\log x},$$

- the function f is only known at some points $x_0, \ldots, x_n$,
- the function f is known by a complex algorithm that provides its value at any point.

Basic principle

- Once again, we "replace" the function f by a close enough function for which computing its primitive is quite easy $\rightarrow$ polynomial interpolation
- However, we must be careful with the polynomial interpolation (why ?)
- This is why we will privilege a piecewise polynomial approximation

Description of the principle

- we subdivise the interval $[a, b]$ in some smaller intervals $a = x_0 < x_1 < \cdots < x_N = b$
- By linearity we have
$$\int_a^b f(x)dx = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)dx$$
Hence, we are led to compute some integrals for which the length of the integration interval is relatively small.

Change of variable

- for each integral, the following change of variable

$$x = \frac{(x_{i+1} - x_i)s + (x_i + x_{i+1})}{2} = \frac{h_i s + (x_i + x_{i+1})}{2} \quad \text{with} \quad h_i = x_{i+1} - x_i$$

  yields

$$\int_{x_i}^{x_{i+1}} f(x)dx = \frac{h_i}{2} \int_{-1}^{1} g_i(s)ds$$

  setting

$$g_i(s) = f\left(\frac{h_i s + (x_i + x_{i+1})}{2}\right) \quad \forall s \in [-1, 1]$$

- Therefore, we focus on the approximation of integrals on the reference interval $[-1, 1]$

Outline

Interpolating the integrand over [-1,1]

- Being given n points $(s_j)_{1\leq j\leq n},\ n>0$ in $[-1,1]$, we approximate the function g by the interpolation polynomial of degree $\leq (n-1)$ at points $(s_j, g(s_j))_{1\leq j\leq n}$

$$g(s) \simeq \sum_{j=1}^{n} g(s_j) L_i(s) \quad \text{with} \quad L_j(s) = \prod_{k=1, k\neq j}^{n} \frac{s - s_k}{s_j - s_k}$$

- As a consequence, we have

$$\int_{-1}^{1} g(s)ds \simeq \int_{-1}^{1} \sum_{j=1}^{n} g(s_j) L_j(s) ds = \sum_{i=1}^{n} g(s_j) \int_{-1}^{1} L_j(s) ds = \sum_{j=1}^{n} w_j g(s_j)$$

Interpolating the integrand over [-1,1]

- We have obtained

$$\int_{-1}^{1} g(s)ds \simeq \sum_{j=1}^{n} w_j g(s_j) \quad \text{with} \quad w_j = \int_{-1}^{1} L_j(s)ds$$

Remarks

- if g is a polynomial of degree $\leq n-1$ then it coincides with its interpolation polynomial and the formula is exact.
- We have $\sum_{j=1}^{n} w_j = 2$ (because $\sum_{j=1}^{n} L_j(s) = 1$, can you guess why ?)

Quadrature formula - definition

Definition

- the formula $\int_{-1}^{1} g(s)ds \simeq \sum_{j=1}^{n} w_j g(s_j)$ is called elementary quadrature formula with n stages
- the (s_j) are the nodes of the quadrature formula and the (w_j) are the weights
- By linearity, the formula

$$\int_a^b f(x)dx \simeq \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)dx = \sum_{i=0}^{N-1} \frac{h_i}{2} \int_{-1}^{1} f\left(\frac{h_i s + (x_i + x_{i+1})}{2}\right) ds$$

leads composite quadrature formulas

Quadrature formula

Remarks

- Composite formulas can be obtained directly by replacing f by its piecewise interpolant
- The interest of coming back to a fixed interval $[-1, 1]$ is to allow a unified treatment that is independent of the interval $[x_i, x_{i+1}]$
- In particular, it shows that the weights (w_j) do not depend on i !

We will now derive various quadrature rules thanks to interpolation

Outline

Examples

- the left rectangle formula (0th order polynomial with $x_0 = -1$)

$$\int_{-1}^{1} g(t)dt \simeq 2g(-1)$$

  Composite rule

$$\int_a^b f(x)dx \simeq \sum_{i=0}^{N-1} h_i f(x_i)$$

- the right rectangles formula (0th order polynomial with $x_0 = 1$)

$$\int_{-1}^{1} g(t)dt \simeq 2g(1)$$

  Composite rule

$$\int_a^b f(x)dx \simeq \sum_{i=0}^{N-1} h_i f(x_{i+1})$$

Examples

- the midpoint formula (0th order polynomial with $x_0 = 0$)

$$\int_{-1}^{1} g(t)dt \simeq 2g(0)$$

Composite rule

$$\int_a^b f(x)dx \simeq \sum_{i=0}^{N-1} h_i f\left(\frac{x_i + x_{i+1}}{2}\right)$$

Examples

- the trapezoidal formula (1st order polynomial with $x_0 = -1, x_1 = 1$)

$$\int_{-1}^{1} g(t)dt \simeq 2\left(\frac{g(-1)+g(1)}{2}\right)$$

Composite rule

$$\int_a^b f(x)dx \simeq \frac{h_0}{2}f(a) + \sum_{i=1}^{N-2}\frac{h_i}{2}\left(f(x_i)+f(x_{i+1})\right) + \frac{h_{N-1}}{2}f(b)$$

and for a regular (uniformly distributed) subdivision

$$\int_a^b f(x)dx \simeq h\left[\frac{1}{2}f(a) + \sum_{i=1}^{N-1} f(x_i) + \frac{1}{2}f(b)\right]$$

Examples

- the Simpson's quadrature rule (2nd order polynomial with $x_0 = -1, x_1 = 0, x_2 = 1$)

$$\int_{-1}^{1} g(t)dt \simeq 2\left(\frac{1}{6}g(-1) + \frac{4}{6}g(0) + \frac{1}{6}g(1)\right)$$

Composite rule, regular subdivision

$$\int_a^b f(x)dx \simeq \frac{h}{6}\left[f(a) + 2\sum_{i=1}^{N-1} f(x_i) + f(b) + 4\sum_{i=0}^{N-1} f\left(\frac{x_i + x_{i+1}}{2}\right)\right]$$

Illustrative summary

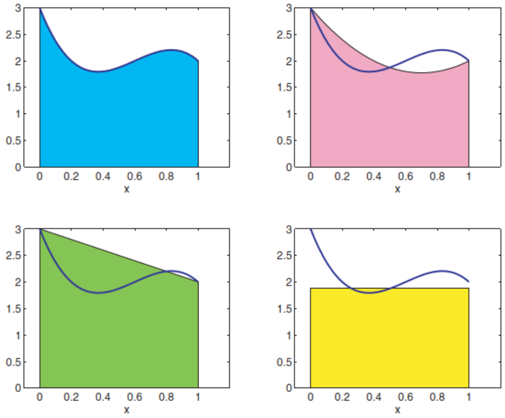

Figure 15.1. *Area under the curve. Top left (cyan): for $f(x)$ that stays nonnegative, I_f equals the area under the function's curve. Bottom left (green): approximation by the trapezoidal rule. Top right (pink): approximation by the Simpson rule. Bottom right (yellow): approximation by the midpoint rule.*

Exercise: derive the weights of these rules by integrating (by hand) Lagrange polynomials

Examples: higher-order formulas

The Newton-Cotes formulas

More generally, if we choose equidistant nodes (x_j) in the quadrature formulas $(n > 1)$

$$x_j = -1 + 2\frac{(j-1)}{(n-1)}, \quad 1 \leq j \leq n$$

the quadrature formulae $(x_j, w_j)_{1 \leq j \leq n}$ are called (closed) Newton-Cotes formulae.

Examples: higher-order formulas

The Newton-Cotes formulas

n	w_j							name
2	$\frac{1}{2}$	$\frac{1}{2}$						trapezoidal
3	$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$					Simpson
4	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{1}{8}$				Newton
5	$\frac{7}{90}$	$\frac{32}{90}$	$\frac{12}{90}$	$\frac{32}{90}$	$\frac{7}{90}$			Boole
6	$\frac{19}{288}$	$\frac{75}{288}$	$\frac{50}{288}$	$\frac{50}{288}$	$\frac{75}{288}$	$\frac{19}{288}$		-
7	$\frac{41}{840}$	$\frac{216}{840}$	$\frac{27}{840}$	$\frac{272}{840}$	$\frac{27}{840}$	$\frac{216}{840}$	$\frac{41}{840}$	Weddle

Examples: higher-order formulas

The Newton-Cotes formulas

- for n ($n \geq 10$) large, the weigths (w_j) explode and the signs are mixed which implies that the formulae are very sensitive to round-off errors
- some coefficients start to be such that $w_j < 0$ for $n \geq 9$
- this implies that the Newton-Cotes formulas are used for $n \leq 8$

Remark

we can similarly build some elementary quadrature formulas with some equidistant nodes on $[-1, 1]$, but excluding the endpoints -1 and 1. For example

$$x_j = -1 + \frac{2j-1}{n} \quad \forall\, 1 \leq j \leq n$$

Such formulas are called (open) Newton-Cotes formulas (like for example the midpoint rule)

Outline

Order of accuracy of elementary quadrature

Definition

We say that a n stages elementary quadrature formula $(x_j, w_j)_{1\leq j\leq n}$ is of order p if the integration formula is exact for any polynomial of degree less or equal to $p-1$

$$\int_{-1}^{1} g(s)ds = \sum_{j=1}^{n} w_j g(x_j) \quad \forall\ g \in \mathbb{R}_{p-1}[X]$$

Remark

By construction, a n stages elementary quadrature formula is at least of order n

Order of accuracy of elementary quadrature

Theorem

A n stages elementary integration formula $(w_j, x_j)_{1\leq j\leq n}$ is of order p if and only if

$$\forall\ 0 \leq q \leq p-1, \quad \sum_{j=1}^{n} w_j x_j^q = \begin{cases} \dfrac{2}{q+1} & \text{if } q \text{ even} \\ 0 & \text{if } q \text{ odd} \end{cases}$$

Deriving quadrature rules - another point of view

By fixing n distinct nodes $x_1, \ldots, x_n$ and considering an order $p = n$, a necessary and sufficient condition to satisfy the previous theorem is to solve the following linear system

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 0 \\ \frac{1}{3} \\ \vdots \\ \frac{1-(-1)^{n+1}}{2(n+1)} \end{pmatrix}$$

This is a Vandermonde matrix which is invertible. This system gives us a quadrature formula of order $p \geq n$. We can then find again the previous quadrature formulas.

Example

Let try $n = 3$ with the points $x_1 = -1$, $x_2 = 0$ and $x_3 = 1$

One gets the system

$$\begin{pmatrix} 1 & 1 & 1 \\ -1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ \frac{2}{3} \end{pmatrix}$$

Its solution gives

$$w_1 = \frac{1}{3}, \quad w_2 = \frac{4}{3}, \quad w_3 = \frac{1}{3}$$

We find the Simpson's formula

$$\int_{-1}^{1} g(t)dt \simeq 2\left(\frac{1}{6}g(-1) + \frac{4}{6}g(0) + \frac{1}{6}g(1)\right)$$

Example

Simpson's formula: checking our result

- $q = 0$

$$2\left(\frac{1}{6} \times 1 + \frac{4}{6} \times 1 + \frac{1}{6} \times 1\right) = 2 = \int_{-1}^{1} t^0 dt$$

- $q = 1$

$$2\left(\frac{1}{6} \times (-1)^1 + \frac{4}{6} \times (0)^1 + \frac{1}{6} \times (1)^1\right) = 0 = \int_{-1}^{1} t^1 dt$$

- $q = 2$

$$2\left(\frac{1}{6} \times (-1)^2 + \frac{4}{6} \times (0)^2 + \frac{1}{6} \times (1)^2\right) = \frac{2}{3} = \int_{-1}^{1} t^2 dt$$

As expected the Simpson's formula is at least of order 3

Example

Simpson's formula: checking our result

But we can also remark that

- $q = 3$

$$2\left(\frac{1}{6}\times(-1)^3+\frac{4}{6}\times(0)^3+\frac{1}{6}\times(1)^3\right)=0=\int_{-1}^{1}t^3dt$$

- $q = 4$

$$2\left(\frac{1}{6}\times(-1)^4+\frac{4}{6}\times(0)^4+\frac{1}{6}\times(1)^4\right)=\frac{2}{3}\neq\frac{2}{5}=\int_{-1}^{1}t^4dt$$

Then the Simpson's formula is of order 4
This is a consequence of a general property for symmetrical elementary quadrature formulas.

Definition

An elementary quadrature formula $(w_j, x_j)_{1 \leq j \leq n}$ is called symmetrical if and only if

$$x_j = -x_{n+1-j} \quad w_j = w_{n+1-j} \quad \forall\, 1 \leq j \leq n$$

Theorem

A symmetrical quadrature formula has always an even order. In other words if the formula is exact for polynomials of degree $\leq 2m - 1$ then it is automatically also exact for polynomials of order $\leq 2m$

Example

- The midpoint $(n = 1)$ and trapezoidal $(n = 2)$ rules are of second order
- Simpson's rule is of order 4 $(n = 3)$
- The Newton-Cotes are symmetrical, so choosing n odd increases the order by one "for free"

Outline

Brief study of the error

We now want to study the error related to the elementary quadrature formula

$$E(g) = \int_a^b g(x)dx - \sum_{j=1}^{n} w_j g(x_j), \quad a=-1,\ b=1$$
$$= \int_a^b g\left[x_1, x_2, \ldots, x_n, x\right](x-x_1)(x-x_2)\cdots(x-x_n)\,dx.$$

with $\psi_n(x) = (x-x_1)(x-x_2)\cdots(x-x_n)$.
Here we distinguish two cases:

- ψ_n has constant sign over $[a,b] \to$ mean value theorem
- $\int_a^b \psi_n(x)dx = 0 \to$ we use $g\left[x_1,\ldots,x_n,x\right] = g\left[x_1,\ldots,x_n,x_{n+1}\right] + (x_{n+1}-x)g\left[x_1,\ldots,x_n,x_{n+1},x\right]$ and redo as above with ψ_{n+1}

Finally, we obtain estimates by integrating $\int_a^b \psi_n(x)dx$

Error estimates - Composite quadrature rules

- midpoint formula (order 2)

$$E(f,h) \leq h^2 \frac{(b-a)}{24} \max_{x\in[a,b]} |f''(x)|$$

- trapezoidal rule (order 2)

$$E(f,h) \leq h^2 \frac{(b-a)}{12} \max_{x\in[a,b]} |f''(x)|$$

- Simpson's formula (order 4)

$$E(f,h) \leq h^4 \frac{(b-a)}{2880} \max_{x\in[a,b]} |f^{(4)}(x)|$$

Error estimates - Remark

- to increase the approximation order, we can add some distinct nodes n
- however, this also increases the computational cost of the method. Indeed, each interval requires n evaluations of the function

Question:

Can we choose the nodes (x_j) so that the approximation order is higher ?

Outline

Towards high-order quadrature

We fix the number of nodes n. If one fixes some distinct nodes (x_j), there exists a unique quadrature formula of order $p \geq n$

Theorem

Let $(w_j, x_j)_{1 \leq j \leq n}$ be an elementary quadrature formula of order p and let

$$M(x) = \prod_{j=1}^{n} (x - x_j).$$

Then, the order is higher or equal to $n + m$ iff

$$\int_{-1}^{1} M(x) q(x) dx = 0, \quad \forall\ q \in \mathbb{R}_{m-1}[X]$$

Example

$n = 3$ stages high-order quadrature formula

- to get a $n = 3$ stages formula of order ≥ 3, we require that

$$\begin{aligned} 0 &= \int_{-1}^{1} (x - x_1)(x - x_2)(x - x_3)\, 1\, dx \\ &= \int_{-1}^{1} x^3 - (x_1 + x_2 + x_3)x^2 + (x_1x_2 + x_1x_3 + x_2x_3)x - x_1x_2x_3 dx \\ &= -\frac{2}{3}(x_1 + x_2 + x_3) - 2x_1x_2x_3 \end{aligned}$$

- let us now keep on studying the $n = 3$ stages quadrature formula and try to determine x_1, x_2, x_3 to get an order ≥ 6
- redo the previous step with $q(x) = x$ and $q(x) = x^2$ to find 2 new equations

Example

$n = 3$ stages high-order quadrature formula

- From the Theorem, it is necessary and sufficient to get

$$\begin{cases} -\frac{1}{3}(x_1 + x_2 + x_3) - x_1x_2x_3 = 0 \\ \frac{1}{5} + \frac{1}{3}(x_1x_2 + x_1x_3 + x_2x_3) = 0 \\ -\frac{1}{5}(x_1 + x_2 + x_3) - \frac{1}{3}x_1x_2x_3 = 0 \end{cases}$$

By setting $\sigma_1 = x_1 + x_2 + x_3$, $\sigma_2 = x_1x_2 + x_1x_3 + x_2x_3$, $\sigma_3 = x_1x_2x_3$, we get

$$\begin{cases} \frac{1}{3}\sigma_1 + \sigma_3 = 0 \\ \frac{1}{5} + \frac{1}{3}\sigma_2 = 0 \\ \frac{1}{5}\sigma_1 + \frac{1}{3}\sigma_3 = 0 \end{cases}$$

- the solution to this system gives

$$\sigma_1 = 0, \quad \sigma_2 = -\frac{3}{5}, \quad \sigma_3 = 0$$

Example

$n = 3$ stages high-order quadrature formula

- now, we have

$$M(x) = (x - x_1)(x - x_2)(x - x_3) = x^3 - \sigma_1 x^2 + \sigma_2 x - \sigma_3$$

- hence

$$M(x) = x(x^2 - \frac{3}{5}) = x \left(x - \sqrt{\frac{3}{5}} \right) \left(x + \sqrt{\frac{3}{5}} \right)$$

- As a consequence

$$x_1 = -\sqrt{\frac{3}{5}}, \quad x_2 = 0 \quad x_3 = \sqrt{\frac{3}{5}}$$

Example

$n = 3$ stages high-order quadrature formula

- we get the weights (w_j) by solving the linear system

$$\begin{pmatrix} 1 & 1 & 1 \\ -\sqrt{\frac{3}{5}} & 0 & \sqrt{\frac{3}{5}} \\ \frac{3}{5} & 0 & \frac{3}{5} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 0 \\ \frac{1}{3} \end{pmatrix}$$

- and then

$$w_1 = \frac{5}{9}, \quad w_2 = \frac{8}{9}, \quad w_3 = \frac{5}{9}$$

- we then have obtained a quadrature formula of order $p = 6$ with only $n = 3$ stages!

$$\int_{-1}^{1} g(x)dx \simeq 2\left(\frac{5}{18}g\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{18}g(0) + \frac{5}{18}g\left(\sqrt{\frac{3}{5}}\right)\right)$$

Towards high-order quadrature

Question

Can we still do better?

Theorem

If p is the order of a n stages elementary quadrature formula, then we necessarily have

$$p \leq 2n$$

Towards high-order quadrature

Question

How to build an elementary quadrature formula of order $2n$ for $n \geq 4$?

- it is possible to do some similar computations as in the previous example
- nevertheless, the calculations involving $M(x) = \prod_{j=1}^{n}(x - x_j)$ are quite complex ...
- but the theorem remains valid for any polynomial M of degree n
- the idea is then to consider a well-chosen polynomial M

Towards high-order quadrature

We then search for a polynomial M such that

- M is of degree $= n$
- M has n distincts roots $(x_j)_{1 \leq j \leq n}$, all being required to lie in $[-1, 1]$
- $\displaystyle\int_{-1}^{1} M(x)g(x)dx = 0$, for any $g \in \mathbb{R}_{n-1}[X]$

Hence, by determining the n roots $(x_j)_{1 \leq j \leq n}$, next by computing the n weights $(w_j)_{1 \leq j \leq n}$ by solving the linear system, we will get an elementary quadrature formula of order $2n$.

Towards high-order quadrature

we can remark that the relations

$$\int_{-1}^{1} M(x)g(x)dx = 0, \quad \forall g \in \mathbb{R}_{n-1}[X] \quad (\star)$$

are orthogonality relations.
Indeed

$$\langle P, Q \rangle = \int_{-1}^{1} P(x)Q(x)dx$$

is an inner product on the vectorial space of real-valued polynomials and the relations $(\star)$ means that $M \in \mathbb{R}_{n-1}^{\perp}[X]$
Therefore, it seems natural to use the orthogonal Legendre polynomials P_k

Another view on orthogonal Legendre polynomials

Theorem

Let $k \in \mathbb{N}^*$. The polynomial P_k defined by

$$P_k(t) = \frac{1}{2^k k!} \frac{d^k}{dt^k} \left((t^2 - 1)^k\right)$$

is a polynomial of degree k such that

$$\int_{-1}^{1} P_k(t) g(t) dt = 0, \quad \forall\, g \in \mathbb{R}_{k-1}[X]$$

Remark

- the polynomials P_k are orthogonal
- the P_k are called orthogonal Legendre polynomials
- the (normalization) constant is chosen such that $P_k(1) = 1$

Another view on orthogonal Legendre polynomials

Theorem

The Legendre polynomials satisfy

$$(k+1)P_{k+1} = (2k+1)P_k - kP_{k-1}, \quad \forall\ k \geq 1$$

The first Legendre polynomials are

$$\begin{aligned} P_0(t) &= 1, & P_3(t) &= \tfrac{5}{2}t^3 - \tfrac{3}{2}t \\ P_1(t) &= t, & P_4(t) &= \tfrac{35}{8}t^4 - \tfrac{30}{8}t^2 + \tfrac{3}{8} \\ P_2(t) &= \tfrac{3}{2}t^2 - \tfrac{1}{2}, & P_5(t) &= \tfrac{63}{8}t^5 - \tfrac{70}{8}t^3 + \tfrac{15}{8}t \end{aligned}$$

Another view on orthogonal Legendre polynomials

To be able to use these polynomials to build some quadrature formulas, we need some informations on their roots that will play the role of the nodes for the quadrature formula

Theorem

Let $k \in \mathbb{N}^*$. All the roots of P_k are real, simple and are in $]-1,1[$.

Gauss quadrature

We can now build some n stages elementary quadrature formulas of order $2n$ by using $M(t) = P_n$ as the Legendre polynomial of degree n.

Theorem (Gauss)

For any $n \in \mathbb{N}^*$, there exists a unique n stages elementary quadrature formula of order $2n$. It is defined by

- the nodes $(x_j)_{1\leq j\leq n}$ are the n distinct roots of P_n
- the weights $(w_j)_{1\leq j\leq n}$ are obtained by solving a linear system

Gauss quadrature

- $n = 1$ - order 2

$$\int_{-1}^{1} g(t)dt \simeq 2g(0)$$

  midpoint formula

- $n = 2$ - order 4

$$\int_{-1}^{1} g(t)dt \simeq 2\left(\frac{1}{2}g\left(-\frac{1}{\sqrt{3}}\right) + \frac{1}{2}g\left(\frac{1}{\sqrt{3}}\right)\right)$$

- $n = 3$ - order 6

$$\int_{-1}^{1} g(t)dt \simeq 2\left(\frac{5}{18}g\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{18}g(0) + \frac{5}{18}g\left(\sqrt{\frac{3}{5}}\right)\right)$$

Gauss quadrature

- $n = 4$ - order 8

$$\int_{-1}^{1} g(t)dt \simeq 2\left(\alpha g\left(-\delta\right) + \alpha' g\left(-\delta'\right) + \alpha' g\left(\delta'\right) + \alpha g\left(-\delta\right)\right)$$

with

$$\delta = \sqrt{\frac{15+2\sqrt{30}}{35}}, \quad \delta' = \sqrt{\frac{15-2\sqrt{30}}{35}}$$

$$\alpha = \frac{1}{4} - \frac{\sqrt{30}}{72}, \quad \alpha' = \frac{1}{4} + \frac{\sqrt{30}}{72}$$

Outline

Computational cost of a numerical quadrature

- the cost is essentially related to the number of evaluations of the function f
- for example, for a 2-points Gauss quadrature, to get an approximation of

$$\int_{x_i}^{x_{i+1}} f(x)dx,$$

  requires two evaluations of f, that is a total of $2N$ evaluations of f.
- more generally, for a s points Gauss quadrature, it needs sN evaluations of f.

Computational cost of a numerical quadrature

- for the Simpson's rule, one gets

$$\int_{x_i}^{x_{i+1}} f(x)dx \simeq h\left(\frac{1}{3}f(x_i) + \frac{4}{3}f\left(\frac{x_i + x_{i+1}}{2}\right) + \frac{1}{3}f(x_{i+1})\right)$$

so *a priori* 3 evaluations of f

- but for the following approximation

$$\int_{x_{i+1}}^{x_{i+2}} f(x)dx \simeq h\left(\frac{1}{3}f(x_{i+1}) + \frac{4}{3}f\left(\frac{x_{i+1} + x_{i+2}}{2}\right) + \frac{1}{3}f(x_{i+2})\right)$$

we only need two evaluations of f! (if the method has been correctly hard coded!)

- for Simpson, the number of evaluations of f is then $2N + 1$
- more generally, for a s points Newton-Cotes formula, we need $(s-1)N + 1$ evaluations.

Summary of the contents

Numerical differentiation :

1. Interpolation allows to approximate derivatives at the discrete level
2. It allows to solve boundary value problem by differentiation matrices
3. These methods are subjected to round-off errors

Numerical integration :

1. Basic quadrature rules are built from Lagrange interpolation with equidistant points
2. A quadrature rule is defined by nodes and weights (x_j, w_j) over $[-1, 1]$
3. Higher order quadrature rules can be obtained thanks to orthogonal polynomials. One example is Gauss-Legendre quadrature, but they are many such quadrature rules !