Numerical derivative and quadrature

quadrature = calculating a definite integral

one dimensional

- To fit data by a suitable function → integrate/differentiate this function. Applicable if the data are subject of errors (experimental data).
  Example: Shomate equation

$$C^\circ_{p\mathrm{m}}(T) = A + BT + CT^2 + DT^3 + E/T^2$$

- To replace a derivative by a difference,
  – several points in the neighborhood are needed
  – the accuracy decreases
  To replace a quadrature by a sum over selected points in the interval
  – the accuracy increases

multidimensional

- partial derivative: repeat in all variables/directions
- quadrature: up to about 3D–5D, several 1D quadratures nested
  more dimensions: Monte Carlo, Conroy integration

Numerical derivative

The difference formulas can be derived from the Taylor series.

1st derivative:

$f(h) = \mathcal{O}(h^n)$ iff $\exists M > 0$ and $h_0 > 0 : |f(h)| \le M h^n \ \forall\ h \le h_0$

$$
\begin{aligned}
\frac{f(x+h)-f(x)}{h} &= f'(x) + \frac{h}{2}f''(x) + \frac{h^2}{6}f'''(x) + \cdots \\
&= f'(x) + \mathcal{O}(h) \\
\Rightarrow \quad f'(x) &= \frac{f(x+h)-f(x)}{h} + \mathcal{O}(h)
\end{aligned}
$$

This is the 1st order formula in h = error is $\mathcal{O}(h)$ = error is on the order of h

More accurate

$$f'(x) = \frac{f(x+h)-f(x-h)}{2h} + \mathcal{O}(h^2)$$

$$f'(x) = \frac{-f(x+2h)+8f(x+h)-8f(x-h)+f(x-2h)}{12h} + \mathcal{O}(h^4) \qquad (1)$$

Right derivative – if function is known for arguments $\ge x$

$$f'(x) = \frac{-f(x+2h)+4f(x+h)-3f(x)}{2h} + \mathcal{O}(h^2)$$

Numerical derivative II

Similarly for the 2nd derivative, the simplest formula:

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + \mathcal{O}(h^2)$$

Which step *h*?

Let ε = numerical precision: the smallest number > 1 is $1+\varepsilon$
Typical error of several operations = several ε

- 64 bit (`double`, `REAL*8`): $\varepsilon = 2^{-52} \doteq 2 \times 10^{-16}$, today's standard
  (typically 1×10^{-15})
- 80 bit (extended, `long double`, `REAL*10`): $\varepsilon = 2^{-63} \doteq 1 \times 10^{-19}$
  (maximum float precision of the FPU on x86 architecture)
- 32 bit (`float`, `REAL*4`): $\varepsilon = 2^{-23} \doteq 1 \times 10^{-7}$
  (minimum speed gain except GPU)

Rule of the thumb: For the best h: rounding error ≈ method error.

Example. Which h is optimum in eq. (1)?

rounding error $\propto \varepsilon/h$, method error $\propto h^4$, $\Rightarrow$ $h \approx \varepsilon^{1/5} = 1 \times 10^{-3}$

Numerical quadrature

Numerical integration in interval $[a, b]$. Assumption: several derivatives exist (are finite) in the interval. If not (e.g., $\sqrt{x}$ in $[0, 1]$), a substitution helps.

General formula:

$$\int_a^b f(x)\mathrm{d}x = \sum w_i f(x_i), \quad x_i \in [a, b]$$

Methods:

- equidistant arguments (Newton–Cotes):
  – closed: use $f(a)$, $f(b)$
  – open: only points in (a, b)
- non-equidistant arguments (Gauss): usually more efficient (if we can calculate the function at arbitrary point)

Improper intervals:

- substitution → finite interval
- special methods

Typically an interval is divided into shorter subinterval and a suitable method is used repeatedly

Newton–Cotes formulas

Trapezoidal rule:

$$\int_a^b f(x)\mathrm{d}x = \frac{b-a}{2}[f(a)+f(b)] + \mathcal{O}((b-a)^2)$$

Several dividing points:

$$\int_a^b f(x)\mathrm{d}x = \left[\frac{f(a)}{2} + f(a+h) + f(a+2h) + \cdots + \frac{f(b)}{2}\right]h + \mathcal{O}(h^2), \quad h = \frac{b-a}{n}$$

Rectangular rule (open): 2× more accurate than trapezoid

$$\int_a^b f(x)\mathrm{d}x = (b-a)\left[f(\frac{a+b}{2})\right] + \mathcal{O}((b-a)^2)$$

Simpson's rule:

$$\int_a^b f(x)\mathrm{d}x = \frac{b-a}{6}\left[f(a) + 4f(\frac{a+b}{2}) + f(b)\right] + \mathcal{O}((b-a)^4)$$

Order and error: because of linearity, it's enough to verify for 1, x, x^2, ...

Example. Verify that the Simpson rule integrates exactly 1, x, x^2, x^3, but not x^4, therefore, it is an $\mathcal{O}(h^4)$ method

Gauss quadrature

- Two-point formula of the 4th order has a half error w.r.t. Simpson and needs less ponts (by one)

$$\int_a^b f(x)\mathrm{d}x = \frac{b-a}{2}\left[f\left(\frac{a+b}{2}-\frac{a-b}{2\sqrt{3}}\right)+f\left(\frac{a+b}{2}+\frac{a-b}{2\sqrt{3}}\right)\right]+O((b-a)^4)$$

- The four-point formula of the 8th order with a good numerical stability:

```
double Gauss8(double (*f)(double),double a,double b,int n)
// int[a,b] f(x) dx using 4-point Gauss quadrature with n subintervals
{
  const double
    q1=0.4305681557970262876, // sqrt((15+sqrt(120))/140)
    q2=0.1699905217924281324, // sqrt((15-sqrt(120))/140)
    w=0.1739274225687269287; // 1/4-sqrt(5/864)
  double h=(b-a)/n;
  double w1=h*w, w2=h/2-w1;
  double h1=h*q1, h2=h*q2;
  int i;
  double sum=0,x;

  for (i=0; i<n; i++) {
    x=h*(i+0.5)+a;
    sum+=(f(x-h1)+f(x+h1))*w1+(f(x-h2)+f(x+h2))*w2;
  }
  return sum;
}
```

Higher-order methods may be less stable, better use 4–8th in several subintervals

Richardson extrapolation

The error formulas often have the error given by

$$S = S(h) + Ah^n + Bh^{n+2} + \cdots \quad n \text{ is usually even}$$

Generally we may have

$$S = S(h) + Ah^n + Bh^{n+1} + \cdots$$

More accurate result:

$$S = \overset{S_2(h/2)}{\frac{2^n S(h/2) - S(h)}{2^n - 1}} + \begin{cases} \mathcal{O}(h^{n+2}) \\ \mathcal{O}(h^{n+1}) \end{cases}$$

We can repeat this trick with pair $S_2(h/4)$ a $S_2(h/2)$, etc.

Warning, this process fails if the function is not smooth enough (does not have enough derivatives)

Example. Show that one step of the Richardson extrapolation of the trapezoidal rule is equivalent to the Simpson formula

ODE – the initial value problem

$$y' = f(x, y),\ \ y(x_0) = y_0$$

ODE = Ordinary Differential Equation

- y may be a vector (system of ODEs)
- one higher-order eq. can be transformed to a system of ODEs of the 1st order (but a numerical method tailored to the original ODE may be more efficient)

$$y'' = f(x, y, y'),\quad \text{subst. } z = y' \quad \Rightarrow \quad z' = f(x, y, z),\ \ y' = z$$

- Runge-Kutta: ⊕ history not needed
  ⊕ easy change of the step (adaptive)
  ⊕ good stability
  ⊖ several evaluations of the rhs/step
- predictor–corrector: ⊕ more efficient (less rhs/step),
  ⊖ the history must be calculated in advance
  ⊖ step change is difficult
  ⊖ stability problems
- methods for dynamical systems ($\ddot{r} = f(r, \dot{r}, t)$):
  – symplectic or at least time-reversible (⇒ energy conservation)
  – predictor–corrector

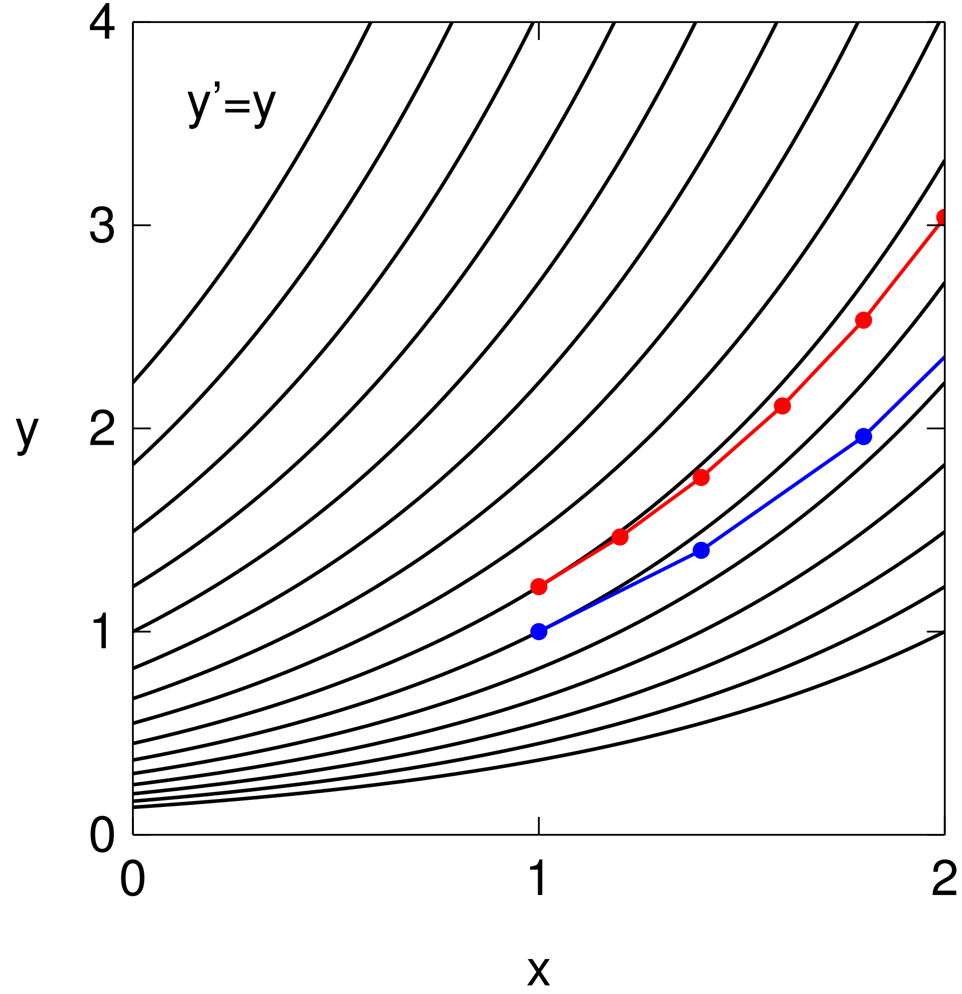
Warm-up: Euler's method

$$y' = f(x,y), \;\; y(x_0) = y_0$$

1 step:

$$y(x+h) = y(x) + hy' + \mathcal{O}(h^2) = y(x) + hf(x,y) + \mathcal{O}(h^2)$$

This method is $\mathcal{O}(h^2)$ locally $\Rightarrow$ $\mathcal{O}(h)$ globally
($\propto 1/h$ steps needed on a finite interval)



Runge–Kutta 2nd order (RK2)

if the function argument is missing, x is assumed

Let's improve Euler – trapezoidal style:

$$\begin{aligned} k_1 &:= f(x,y) \\ k_2 &:= f(x+h, y(x)+hk_1) \\ y(x+h) &:= y(x)+\tfrac{h}{2}(k_1+k_2) \\ x &:= x+h \end{aligned}$$

Order derivation is based on: $y'' = \mathrm{d}f(x,y)/\mathrm{d}x = f_x + f_y y' \Rightarrow$

$$\begin{aligned} y(x+h) &= y(x)+\frac{h}{2}(k_1+k_2) \\ &\overset{\mathcal{O}(h^3)}{\approx} y(x)+\frac{h}{2}(y'+y'+hf_x+hf_yf) = y(x)+hy'(x)+\frac{h^2}{2}y''(x) \end{aligned}$$

Local error $\mathcal{O}(h^3)$ (or better), global error is (at least) $\mathcal{O}(h^2)$

Let's improve Euler – rectangular (half-step) style:

$$\begin{aligned} k_1 &:= f(x,y) \\ k_2 &:= f(x+\tfrac{h}{2}, y(x)+\tfrac{h}{2}k_1) \\ y(x+h) &:= y(x)+hk_2 \\ x &:= x+h \end{aligned}$$

The same order, smaller error coefficient

Runge–Kutta 4th order (RK4)

Popular method of the 4th order (local error $\mathcal{O}(h^5)$):

$$
\begin{aligned}
k_1 &:= f(x, y) \\
k_2 &:= f(x + \tfrac{h}{2}, y(x) + \tfrac{h}{2}k_1) \\
k_3 &:= f(x + \tfrac{h}{2}, y(x) + \tfrac{h}{2}k_2) \\
k_4 &:= f(x + h, y(x) + hk_3) \\
y(x + h) &:= y(x) + \tfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
x &:= x + h
\end{aligned}
$$

Predictor-corrector – intro

We know a history = values (and/or derivatives, i.e., rhs).

- predictor: we predict $y^P(x+h)$:
  – using rhs (usually more stable and more accurate)
  – without evaluating the rhs
  (Gear methods – polynomial extrapolation)
- [optional modificator]
- corrector: we calculate the corrected $y^C(x+h)$:
  – rhs calculated once
  – rhs calculated twice, three times,. . .
  – rhs calculated iteratively until some precision limit is reached

Problem – **stability**: the single-step errors propagate to further steps. The method must guarantee that the errors do not cumulate and do not explode (exponentially/exponential oscillations)

If the coefficients of a method are large with alternating sigs, the method will likely be unstable

Predictor-corrector – 3rd order example

Let's first rewrite RK2 to the predictor-corrector form:

$$\begin{aligned} y^P(x+h) &= y(x) + hf(x, y(x)) + \mathcal{O}(h^2) \\ y^C(x+h) &= y(x) + \frac{h}{2}[f(x, y(x)) + f(x+h, y^P(x+h))] + \mathcal{O}(h^3) \end{aligned}$$

the 2nd step is $\mathcal{O}(h^3)$ because it is trapezoid, and the error in $y^P(x+h)$ is $h\mathcal{O}(h^2)$.

Let's try to improve both steps. **Predictor:**

$$y^P(x+h) = y(x) + \frac{h}{2}[3f(x) - f(x-h)] + \mathcal{O}(h^3)$$

where $f(x) \equiv f(x, y(x)))$ and $f(x-h) \equiv f(x-h, y(x-h))$ (from the previous step).

We shall look for the **corrector** in the form:

$$y^C(x+h) = y(x) + h[af(x-h) + bf(x) + cf(x+h, y^P(x+h))]$$

The test function method using equation $y = y'$ (see matenum4.mw) $\Rightarrow$

$$y^C(x+h) = y(x) + \frac{h}{12}[-f(x-h) + 8f(x) + 5f(x+h, y^P(x+h)] + \mathcal{O}(h^4)$$

= this is a 3rd order method (locally $\mathcal{O}(h^4)$)

3rd order example – stability

The method is $\mathcal{O}(h^4)$ locally, so we can write (neglecting $\mathcal{O}(h^5)$):

$$y(x-ih) = y^{\text{exact}}(x-ih) + \epsilon_i h^4$$

Let's use the test equation $y' = y$ with $y(0) = 1$ (the solution is $y = \mathrm{e}^x$). Using Maple:

$$\epsilon_{i+1} = \epsilon_i - 13/144$$

- any error of $y(x-h)$, $y(x-2h)\ldots$ does not propagate (with precision up to h^4)
- an error $\propto h^4$ is generated in one step

Milne method

Corrector = Simpson formula.

$$\begin{aligned} y^P(x+h) &= y(x-3h) + \frac{4h}{3}[2f(x) - f(x-h) + 2f(x-2h)] \\ y^C(x+h) &= y(x-h) + \frac{h}{3}[f(x-h) + 4f(x) + f(x+h, y^P(x+h)] \end{aligned}$$

Local error = $\mathcal{O}(h^5)$. Let $y(ih)$ be subject to error $\epsilon_i h^5$. The error propagates as follows (see matenum4.mw using $y' = y$):

$$\epsilon_{i+1} := \frac{1}{90} + \epsilon_{i-1}$$

Discussion:

- $\epsilon_{i+1} := \frac{1}{90} + \epsilon_i$ would be OK, a constant error cannot be removed in principle (unless the order increases)

  example of an unstable method: $\epsilon_{i+1} := \frac{4}{3}\epsilon_i + 1/90$

  example of a stable method: $\epsilon_{i+1} := \frac{3}{4}\epsilon_i + 1/90$
- the Milne method method is at the edge of stability (a sort of very particular problem)
- the stride by 2 means that ϵ_{even} and ϵ_{odd} may differ (oscillations caused by higher orders)

A typical equation for error propagation in predictor-corrector methods is (in h^n, where n is the local order)

$$\epsilon_{i+1} := a_c + a_0\epsilon_i + a_1\epsilon_{i-1} \cdots a_n\epsilon_{i-n}$$

This is a **linear difference equation**. A general solution is:

$$\epsilon_i = \sum_x b_x x^i + b_c$$

where the sum is over all roots of the so called **characteristic polynomial**:

$$x^{n+1} = c_0 x^n + \cdots + c_n x^0$$

(In case of multiple roots the basis is $\{x^i, ix^i \ldots\}$, similarly as for systems of homogeneous linear differential equations.)

The errors should not exponentially grow, thus $|x| < 1$ must be satisfied.

Example of a difference equation

Fibonacci sequence: $F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$ for $n > 1$

$$F_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$$