Numerical Analysis ?

- Algorithms (approximation)
- Analysis (errors)
- Implementation (python)

We can only approximate well posed problems. i.e.

1) there exist a unique solution
2) it depends continuously on the data

→ we simplify in a controlled manner

infinite → finite

differential → algebraic

non-linear → linear problems

Example:

Given a function "f", $f \in C^1[a,b]$, a point x_0, $x_0 \in (a,b)$

compute "$f'(x_0)$" (derivative of f in x_0)

→ use finite difference

$$f'(x_0) \approx \frac{f(x_0+h) - f(x_0)}{h}$$

with small h (s.t. $x_0+h \in (a,b)$)

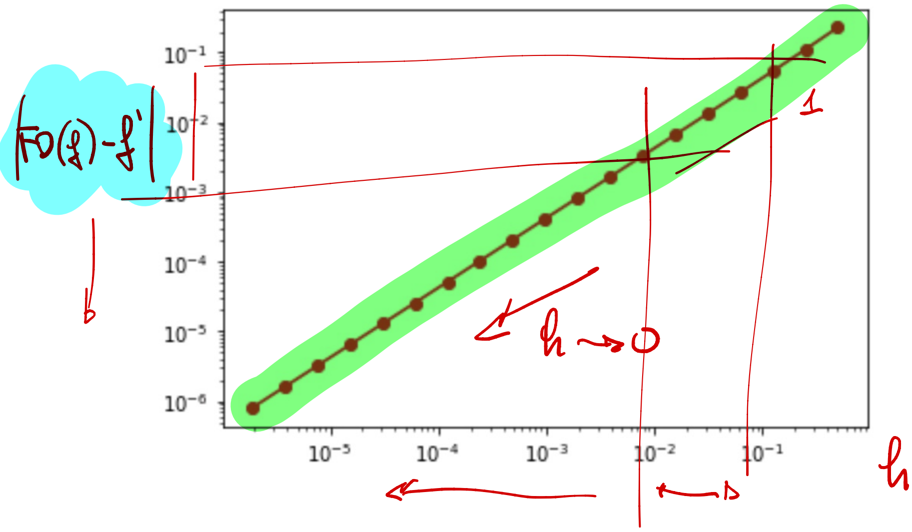
Error: Taylor expansion around x_0:

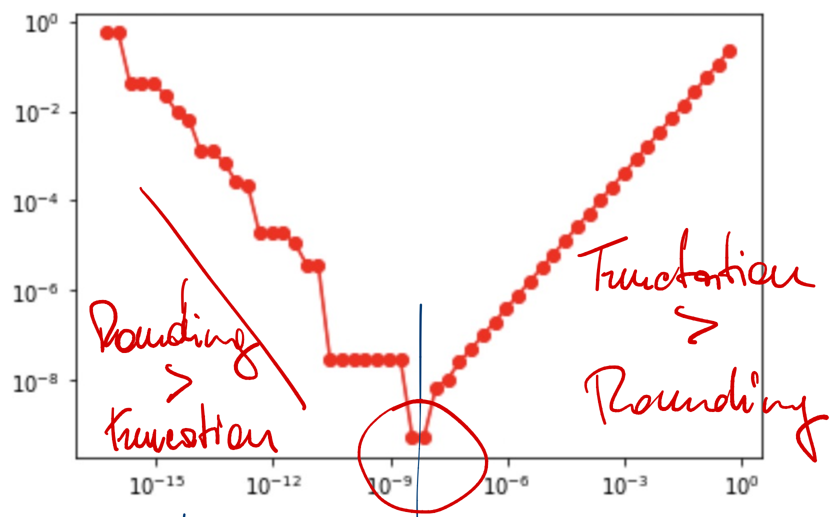
$$f(x_0+h) = f(x_0) + h f'(x_0) + \frac{h^2}{2} f''(x_0) + \ldots$$

$$f(x_0+h) \approx \sum_{i=0}^{n} \frac{f^{(i)}(x_0)\, h^i}{i!}$$

$$FD := \frac{f(x_0+h)-f(x_0)}{h} \simeq f'(x_0) + \frac{1}{2} f''(x_0)\, h + O(h^2)$$

Error : $|FD(f) - f'| \simeq \frac{f''(x_0)\, h}{2} + O(h^2)$





$\sim 10^{-15}$ $\quad 10^{-8}$

$\sqrt{\varepsilon}$ ~ optimal h

ε: precision of machine (Rounding precision)

Data error (Data)

Computational Error (Algorithm)

Truncation error (Algorithm error in Exact Arithmetics)

Rounding Error (Finite Arithmetic)

Pb: Compute value of a function "F"

$$F : \mathbb{R} \longrightarrow \mathbb{R}$$

at a argument x

- x : true value of input
- $F(x)$: desired value of output
- $\hat{x}$: approximate input (may be "off" for rounding)
- $\hat{F}(\hat{x})$: computed output (may be "off" for both rounding and truncation)

$$\hat{F}(\hat{x}) - F(x) =$$

$$\underbrace{\hat{F}(\hat{x}) - \hat{F}(x)}_{1} + \underbrace{\hat{F}(x) - F(x)}_{2}$$

$$\underbrace{\hat{F}(\hat{x}) - F(\hat{x})}_{3} + \underbrace{F(\hat{x}) - F(x)}_{4}$$

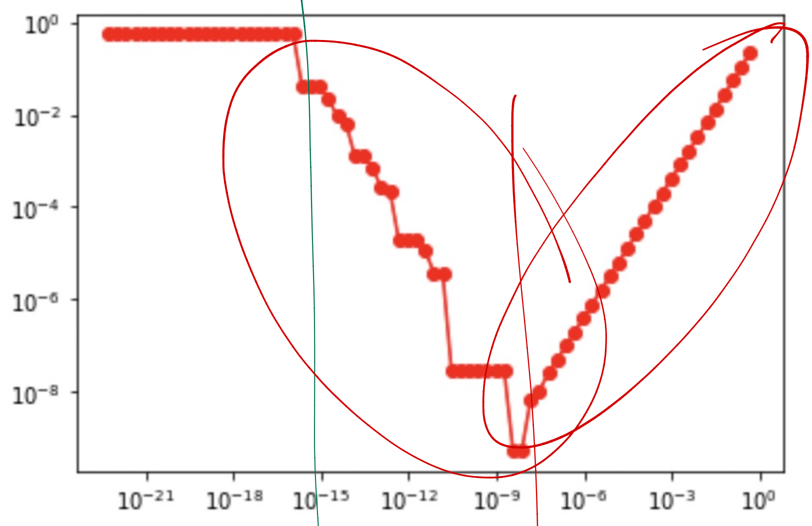
Rounding precision : ε the largest number s.t.

$$fl(1+\varepsilon) = fl(1)$$

fl = floating point representation of a number

$$fl(f(x+h)) = fl(f(x)) \rightarrow f(x+h) - f(x) = 0$$

false in exact arithmetic



optimal $h \simeq \sqrt{\varepsilon}$

$\sin(1) = 0.8414709848078965$