§1.

1D – Optimization

$f : \mathbb{R} \longrightarrow \mathbb{R}$ — min

no constraints

x^* min f

$\Downarrow$ CN

$f'(x^*) = 0$

I – About Math...

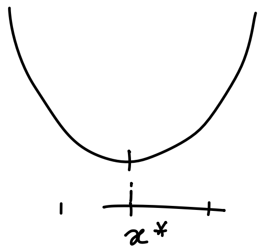
Order 1 conditions (f')

$\Longrightarrow$

NC1 — f $\mathcal{C}^1$

necessary

| if x^* is a min of f
| then $f'(x^*) = 0$



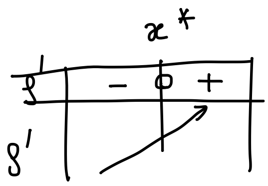
$\Longleftarrow$

SC1

sufficient

(1) if $f'(x^*) = 0$ and
f $\searrow$ before x^*, $\nearrow$ after x^*
then x^* is a minimum

(2) if $f'(x^*) = 0$ and
$f' < 0$ before x^*, $f' > 0$ after x^*
then x^* is a minimum $\longrightarrow$



Order 2 conditions f $\mathcal{C}^2$

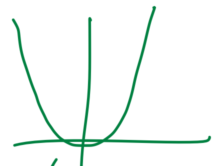
$<$ f', f''

$\Longrightarrow$

NC2

If x^* minimum of f then
| $f'(x^*) = 0$
| and $f''(x^*) \geqslant 0$

$f(x) = x^4$



min = 0

$f''(x) = 12 . x^2$

$f''(0) = 0$ . . . . . .

$\Longleftarrow$

SC2

If $f'(x^*) = 0$ and $f''(x^*) > 0$ then x^* is a min

Computationally:

→ use the derivatives — 1 family → Newton
know

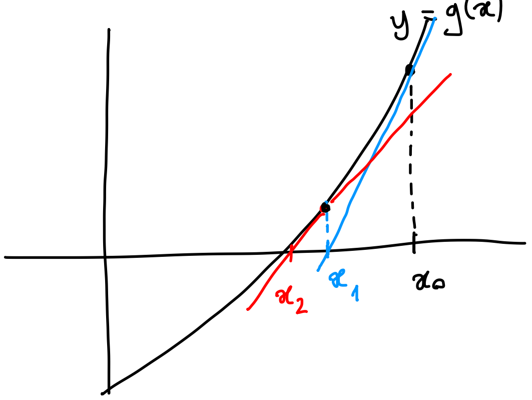
depends on the function we optimize

→ do no use the derivatives — interval reduction methods

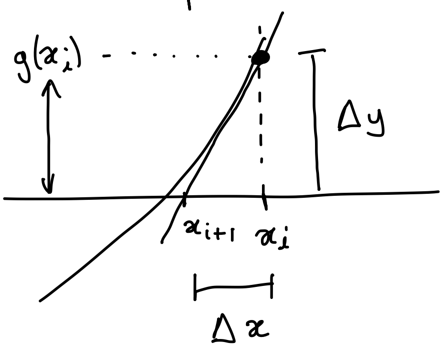
II. Algorithms using the derivatives → ∼ Newton

Newton → compute zeros / roots $g(x) = 0$?

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



slope of the tangent :



$$g'(x_i) = \frac{\Delta y}{\Delta x} = \frac{g(x_i)}{x_i - x_{i+1}}$$

$$\boxed{x_{i+1} = x_i - \frac{g(x_i)}{g'(x_i)}}$$

converges at quadratic speed if g $\mathcal{C}^2$ → zeros of g

Minimum of f ? (x^*) — $\boxed{\text{Newton on } f'}$

$\Downarrow$

$f'(x^*) = 0$

Algorithm : x_0 close to the minimum ....

Iterate :

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

Stop : min up to ε

V1 $\|x_{i+1} - x_i\|$

$\|x_{i+1} - x_i\| < \varepsilon$ absolute error ↳ when (min) small

$\frac{\|x_{i+1} - x_i\|}{\|x_i\|}$ relative error ↳ when (min) large

Convergence (if f $\underline{\underline{\mathcal{C}^3}}$ then converges at quadratic speed

min $\to$ $f' = 0$

V2 $\|f'(x_i)\| < \varepsilon$

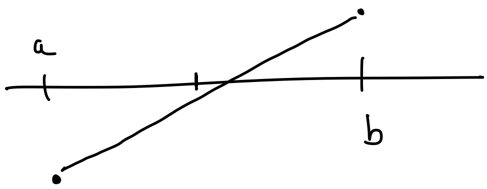
III. Algorithms without derivatives → interval reduction alg.

⇝ finding zeros of functions ($f(x) = 0$)

↙ Newton ✓ f' / $f\ \mathcal{C}^2$

↘ $[a, b] \ni$ zero such that f is $\underline{\underline{\text{monotonous}}}$ on $[a, b]$

dichotomy



Monotonous ⟷ pre-located $\underline{\underline{\text{a}}}$ zero

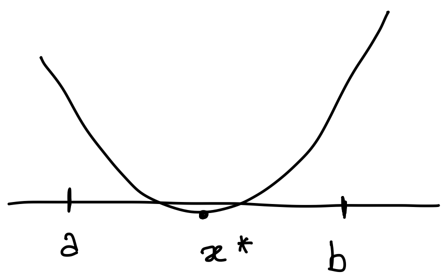
Here ....

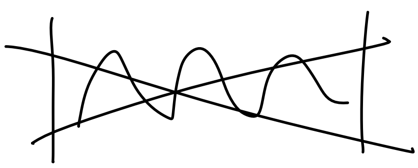
We need an interval where we have

$\boxed{\text{pre-located a minimum}}$

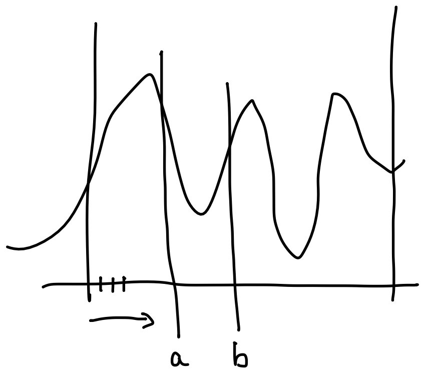
↕

$\boxed{\text{unimodal}}$



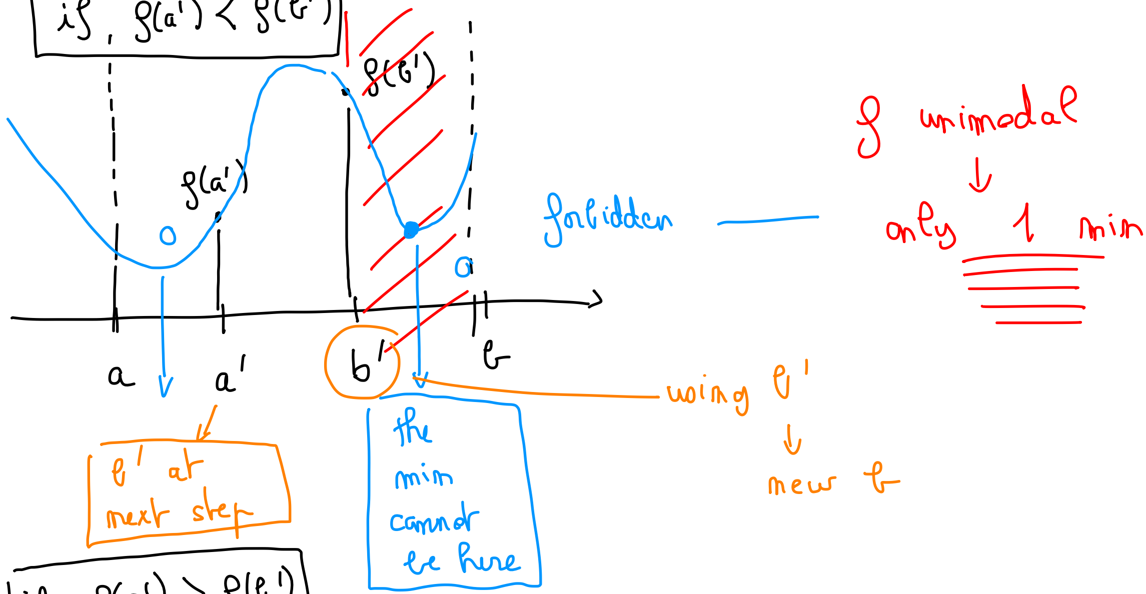


* def: f unimodal on $[a,b]$ if
- → $\exists\, x^* \in \,]a,b[$ min of f
- → $f \searrow$ over $]a,x^*[$, $\nearrow$ over $]x^*,b[$

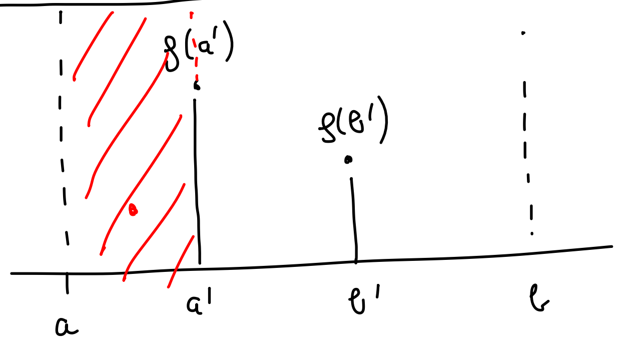


① Reduction methods → general idea

a) $\boxed{\text{if } f(a') < f(b')}$



b) $\boxed{\text{if } f(a') > f(b')}$



Various methods

a' / b' ?

② Golden section method

To chose a', b', follow the strategy:

- → the reduction rate should be constant

$$\frac{1}{\tau} = \frac{\text{length new interval}}{\text{" old "}} = \text{cst} \begin{cases} \text{case a)} \\ \text{case b)} \end{cases}$$

- → the point "not used" should be a' or b' at next step

$\rightsquigarrow$ a', b' are completely determined
$\downarrow$
golden number ratio $\quad \tau = \frac{1+\sqrt{5}}{2}$

Algo $[a,b]$ s.t. (such that) f strictly (non constant) unimodal on $[a,b]$ | ε : required precision

while $(((b-a) > \varepsilon)$ && $(\underline{iter} < N))$ $\longrightarrow$ Implementation

$\quad a' = a + \frac{b-a}{\tau^2}$ $\quad$ $y_a = f(a')$

$\quad b' = a + \frac{b-a}{\tau}$ $\longrightarrow$ $y_b = f(b')$ recovered from prev. iteration

$\quad$ if $(\underbrace{f(a')}_{y_a} < \underbrace{f(b')}_{y_b})$

$\quad\quad \{ b \leftarrow b'$ $\quad$ (a' will be b' at next step)

$\quad$ else if $(\underbrace{f(a')}_{y_a} > \underbrace{f(b')}_{y_b})$

$\quad\quad a \leftarrow a'$

$\quad$ else // case c)

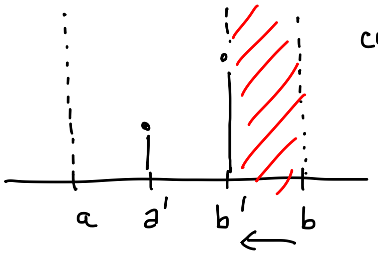
$\quad\quad$ stop – error ... $\quad$ we cannot conclude

$\quad\quad$ $\left\{ \begin{array}{l} a \leftarrow a' \\ b \leftarrow b' \end{array} \right.$

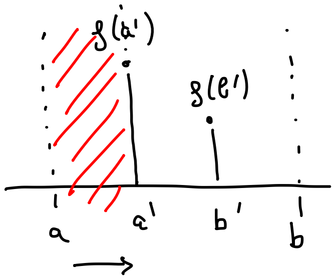
endwhile

return$_{(*)}$ $\left(\frac{a+b}{2}\right)$

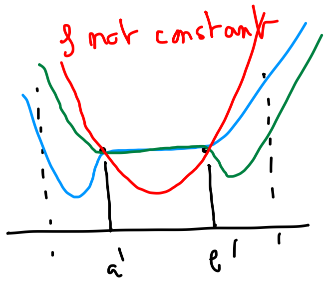
⚠ (if f non constant)



case a)



case b)

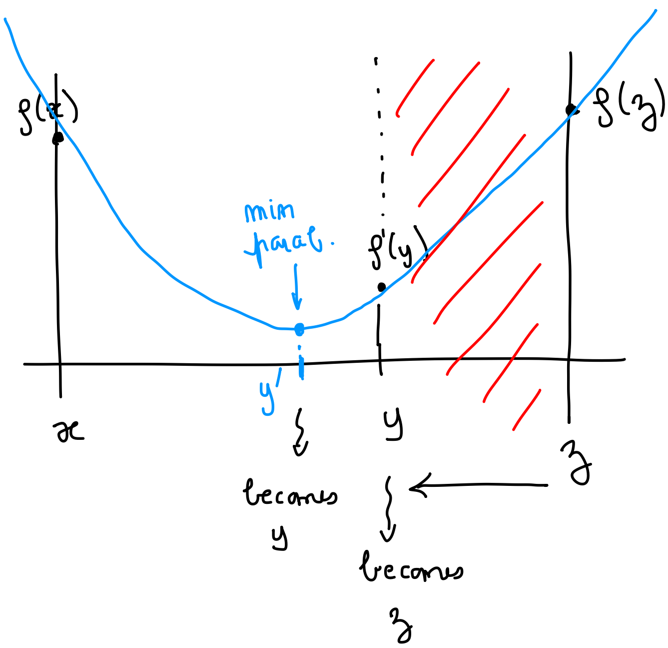


(*) $\min \in \,]a, b[$

Convergence guaranteed if f $\mathcal{C}^2$
$\downarrow$
speed : constant $\quad \frac{\text{length new int}}{\text{" old "}} = \frac{1}{\tau} = \frac{2}{\underbrace{1+\sqrt{5}}_{\sim 2,3}} \sim 3,3 \quad \sim \frac{2}{3,3} \sim 0,7$

If we want $\varepsilon = 10^{-6}$ ..... slow .....

3 Parabolic interpolation.



f unimod on $[x, z]$

$y \in]x, z[$ (first one random)

3 pts $\longrightarrow$ 1 parabola

f unimodal / close to the minimum

min of f
min of the parabola

$\longrightarrow$ same side of y

Convergence – f C^2 : guaranteed

speed : $\| x_{i+1} - x^* \| \leqslant M \cdot \| x_i - x^* \|^{1,37}$

$\exists M$ const. x^* : min

⊕ far better than constant speed (golden section)

slower than Newton

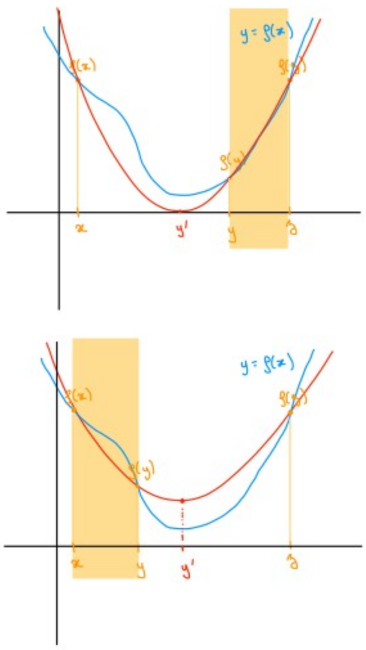
⊖ unstable if $[x, z]$ small ....

Brent method
- begin : parabolic (ε until 10^{-2} / 10^{-3})
- finish convergence : golden section

Matlab : fminbnd

C++
- gsl
- boost

MÉTHODES D'INTERPOLATION PARABOLIQUE



Entrée : fonction f, segment [x,z], seuil ϵ

Sortie : minimum x^*

initialiser y ∈]x,z[

tant que (z-y > ϵ)

$$y' \leftarrow \frac{x+y}{2} - \frac{f[x;y]}{2f[x;y;z]}$$

si (y' < y)

z ← y

y ← y'

sinon si (y' > y)

x ← y

y ← y'

sinon

erreur / break

fin tantque

x^* ← (x+z)/2

Avec les fonctions annexes :

$$f[x;y] = \frac{f(x) - f(y)}{x - y}$$

$$f[x;y;z] = \frac{f[x;z] - f[x;y]}{z - y}$$