

Quelques mots sur l'exploitation de mesures expérimentales...



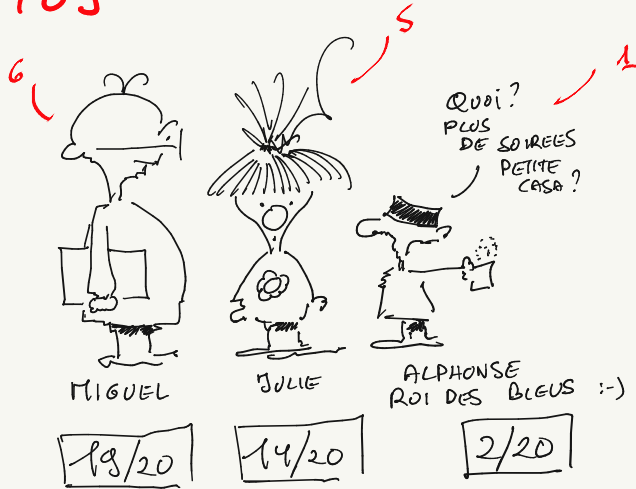
Mesure du poids avec une estimation de l'erreur
Mesure du volume avec une estimation de l'erreur

Comment combiner au mieux ces mesures ?

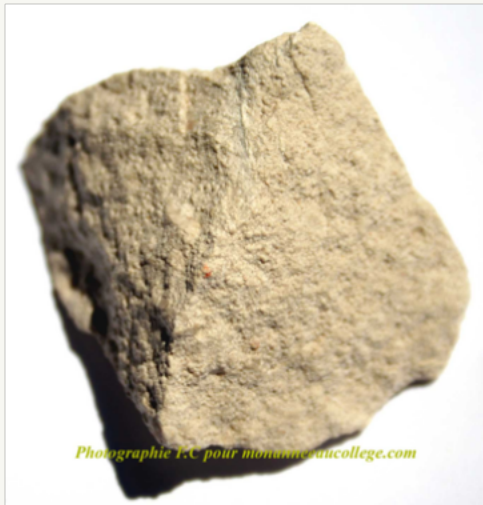
```
Vol      = array([ 10,  8,  8, 10, 99])
ErrVol   = array([  5,  2,  5,  1,  2])
Mass     = array([ 29.1, 25.7, 22.2, 27.5, 266.0])
ErrMass  = array([ 0.1, 0.1, 0.1, 0.1, 0.1])

Rho = Mass / Vol
ErrRho = (ErrMass * Vol + ErrVol * Mass) / (Vol*Vol)
VarRho = ErrRho*ErrRho
```

GROUPE 1769



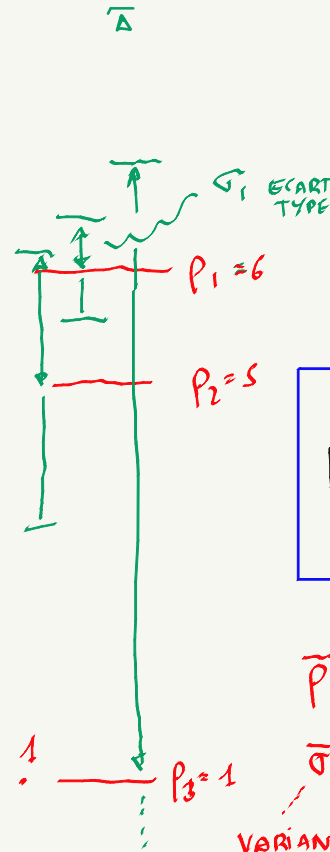
$$\alpha_1 = 18/35 \quad 14/35 = \alpha_2 \quad 2/35 = \alpha_3$$



Photographie 1: C pour monommes au college.com

$$6 \quad 1 \quad 5$$

5,314



$$p(M, V) = \frac{M}{V}$$

$$dp = \underbrace{\frac{\partial p}{\partial M}}_{1/V} dM + \underbrace{\frac{\partial p}{\partial V}}_{-M/V^2} dV$$

$$|dp| = \frac{1}{V^2} [V |dM| + M |dV|]$$

$$\bar{p} = \sum \alpha_i p_i$$

COMMENT MINIMISER $\bar{\sigma}$?

$$\bar{\sigma}^2 = \sum \alpha_i^2 \sigma_i^2$$

$$\sum \alpha_i = 1$$

ESTIMATION SANS BIAIS

$$\alpha_i = \frac{(1/\sigma_i)}{\sum (1/\sigma_i)}$$



ECART TYPE = 15

$$\bar{\sigma} = 0,63$$

$$\bar{\sigma} = \sqrt{\sum \alpha_i^2 \sigma_i^2} = 15 \sqrt{\sum \alpha_i^2} \approx 15 / \sqrt{800} \approx 0,63$$

$\sqrt{800 \cdot \frac{1}{800^2}}$



$$\rho = \frac{M}{V}$$



$$\frac{M + dM}{V + dV}$$

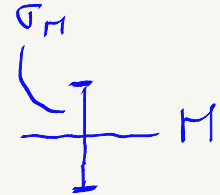
$$\rho + d\rho = \frac{M}{V} + \frac{\partial \rho}{\partial M} dM + \frac{\partial \rho}{\partial V} dV$$

$-\frac{M}{V^2}$

$$d\rho = \frac{1}{V} dM - \frac{M}{V^2} dV$$

$\varepsilon_M = 0.1$ $\varepsilon_V \approx 0.1$

$$|d\rho| = \frac{1}{V} |dM| + \frac{M}{V^2} |dV|$$



$$\underbrace{M}_{4.3 \text{ kg}} + \underbrace{\sigma_M}_{0.01 \text{ kg}}$$

$$\begin{matrix} +0.01 \\ -0.01 \end{matrix} \} \varepsilon_M \quad \text{ERREUR DE MESURE}$$

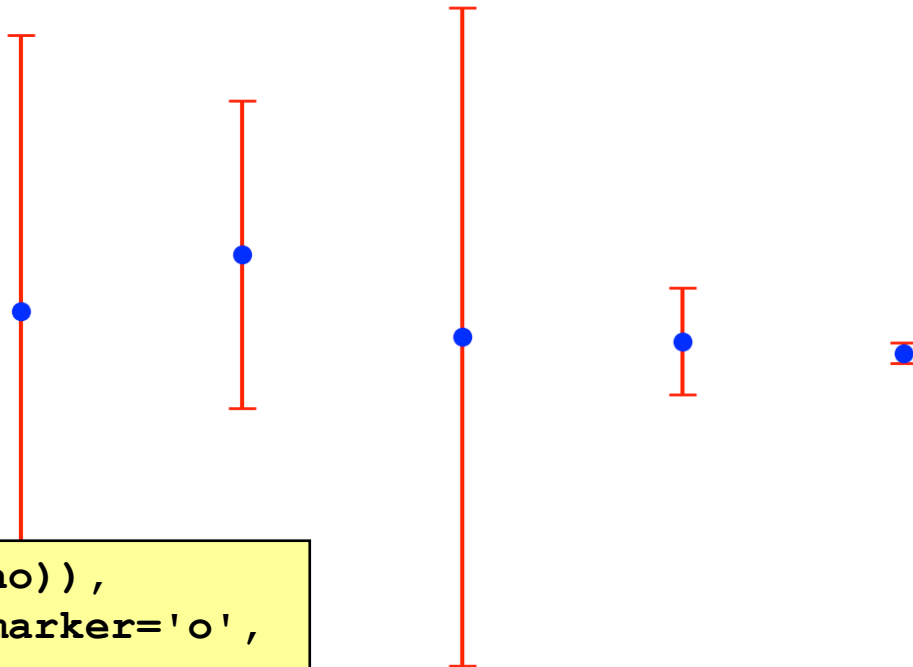
$$\underbrace{V}_{1 \text{ m}^3} + \underbrace{\sigma_V}_{0.1 \text{ m}^3}$$

$+ 0.1 \text{ m}^3$
 $- 0.1 \text{ m}^3$

$$\frac{19}{35} 6 + \frac{14}{35} 5 + \frac{2}{35} 1$$

$$\frac{114 + 70 + 2}{35} = \frac{186}{35} = 5,314$$

Toutes les mesures n'ont pas la même précision...



```
plt.errorbar(arange(len(Rho)),  
             Rho, ErrRho, linestyle='', marker='o',  
             markerfacecolor='blue',  
             markeredgecolor='blue',  
             color='red', capsize=5)
```

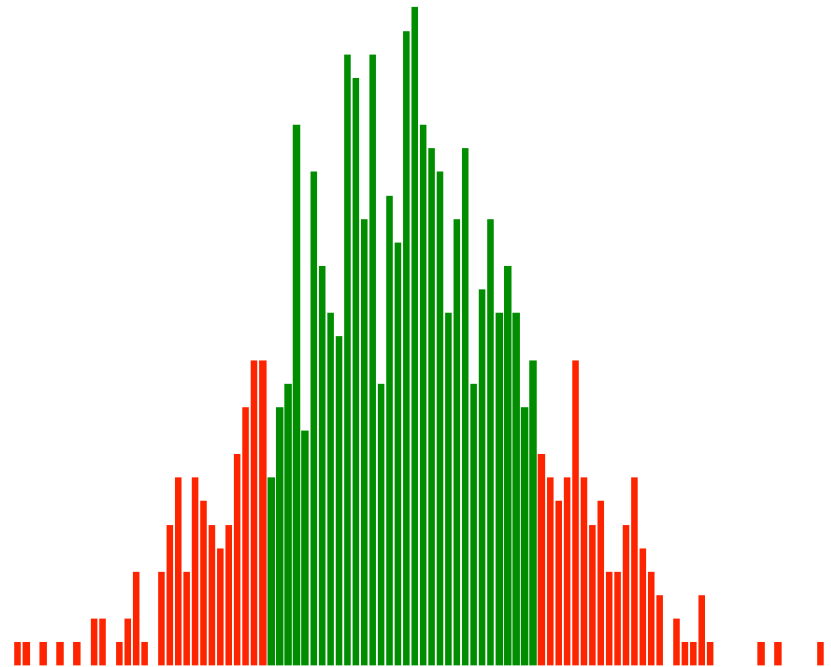
Et pourtant, la donnée la plus imprécise permettra d'améliorer la précision de la meilleure donnée, si, si !

Faisons plein de mesures virtuelles...



```
r = 500 + 15*randn(800)
plt.hist(r,100)
```

*66% des mesures tombent
dans l'intervalle [485,515] !*



numpy.random.randn

`numpy.random.randn(d0, d1, ..., dn)`

Return a sample (or samples) from the “standard normal” distribution.

If positive, int_like or int-convertible arguments are provided, `randn` generates an array of shape `(d0, d1, ..., dn)`, filled with random floats sampled from a univariate “normal” (Gaussian) distribution of mean 0 and variance 1 (if any of the `d_i` are floats, they are first converted to integers by truncation). A single float randomly sampled from the distribution is returned if no argument is provided.

This is a convenience function. If you want an interface that takes a tuple as the first argument, use `numpy.random.standard_normal` instead.

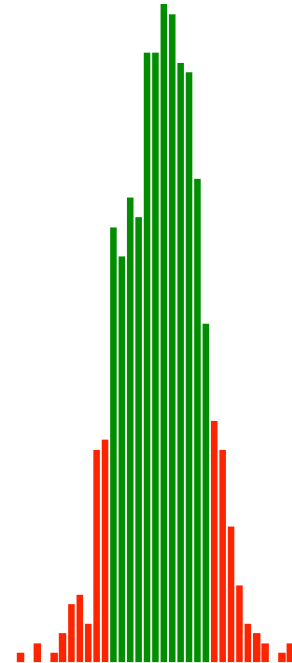
Que vaut l'écart type de la moyenne de n mesures ?

*66% de la moyenne de 10
mesures tombent dans
l'intervalle [495,505] !*

Effectuons 800 mesures avec un écart de 15
Moyenne : 499.9263731
Ecart type de la moyenne : 0.5303301

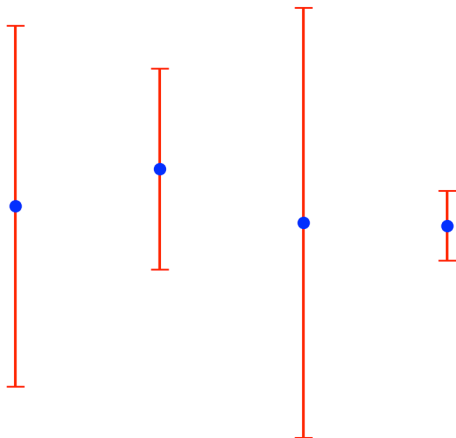
```
print('Ecart type de la moyenne : %14.7f `'  
      % (15/sqrt(800)))
```

```
r = zeros(800);  
for i in range(10):  
    r = r + 500 + 15*randn(800)  
r = r/10
```



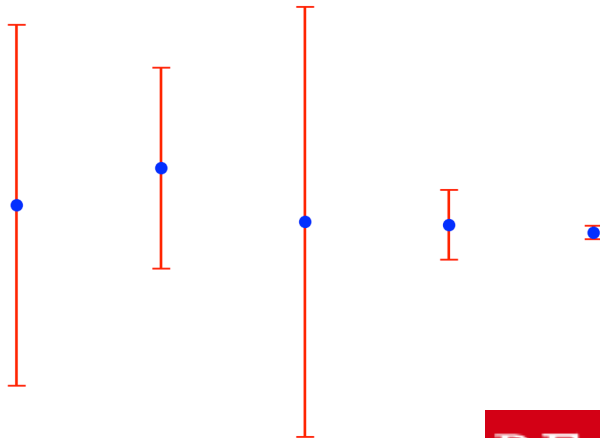
Comment mesurer au nanomètre avec une simple latte ?

Mesure 0 et écart type	:	2.9100000	(1.4650000)
Mesure 1 et écart type	:	3.2125000	(0.8156250)
Mesure 2 et écart type	:	2.7750000	(1.7468750)
Mesure 3 et écart type	:	2.7500000	(0.2850000)
Mesure 4 et écart type	:	2.6868687	(0.0552903)
Meilleure estimation et écart type	:	2.6918441	(0.0540956)



```
VarRho  = ErrRho*ErrRho
coeff = 1 / (VarRho)
coeff = coeff / sum(coeff)
BestEstimate = coeff @ Rho
VarBestEstimate = (coeff*coeff) @ VarRho
EcartBestEstimate = sqrt(VarBestEstimate)
```

Comment combiner des sondages un peu imprécis ?



REAL
CLEAR
POLITICS

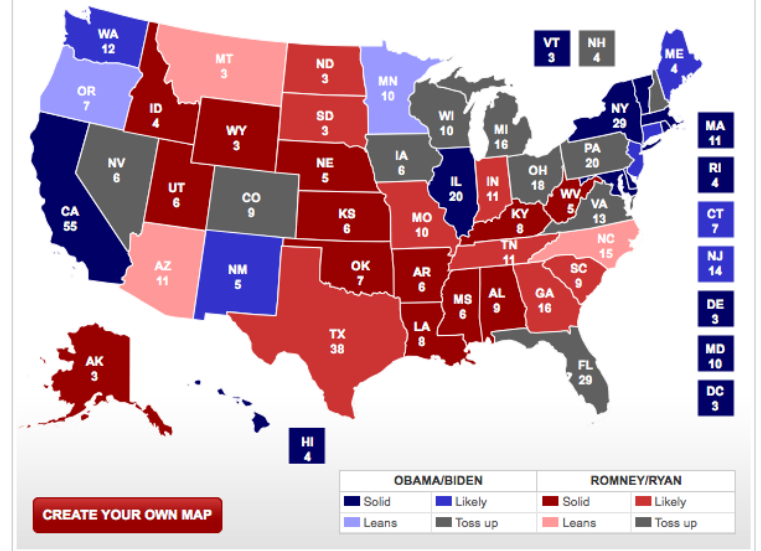
BATTLE FOR WHITE HOUSE

RCP ELECTORAL MAP NO TOSS UPS MAP RCP SENATE MAP SENATE NO TOSS UPS RCP HOUSE MAP
STATE CHANGES STATE CHANGES RACE CHANGES RACE CHANGES



Recent Elections: 2008 | 2004 | 2000 | 1996 | 1992 | 1988 | 1984 | 1980 | 1976 | 1972

Likely Obama (42)	Leans Obama (17)	Toss Up (131)	Leans Romney (29)	Likely Romney (101)
Connecticut (7) Maine (4) New Jersey (14) New Mexico (5) Washington (12) Solid Obama	Minnesota (10) Oregon (7)	Colorado (9) Florida (29) Iowa (8) Michigan (16) Nevada (6) New Hampshire (4) Ohio (18) Pennsylvania (20) Virginia (13) Wisconsin (10)	Arizona (11) Montana (3) North Carolina (15)	Georgia (16) Indiana (11) Missouri (10) North Dakota (3) South Carolina (9) South Dakota (3) Tennessee (11) Texas (38) Solid Romney



2

ET EN METHODES NUMERIQUES

Oui :->

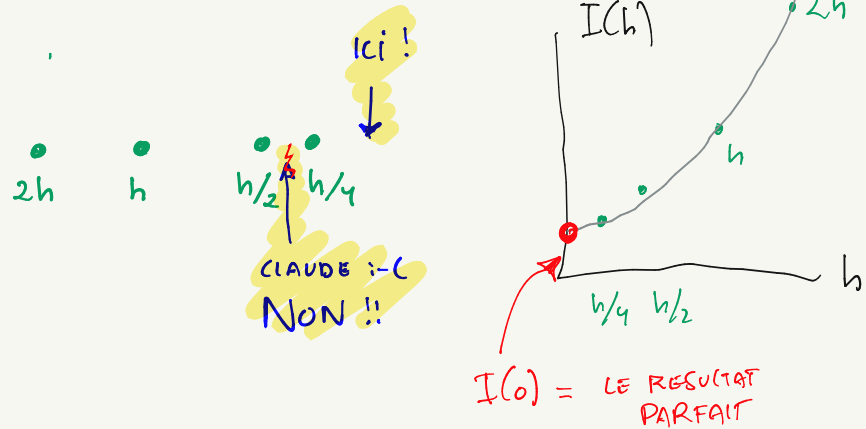
MODELES
CLIMATIQUES



OUI
CELA SE
RECHAUFFE !!

ON
SUPPOSE
DES MODELES
"INDEPENDANTS"

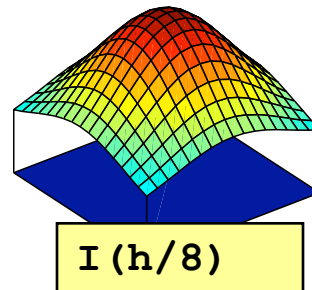
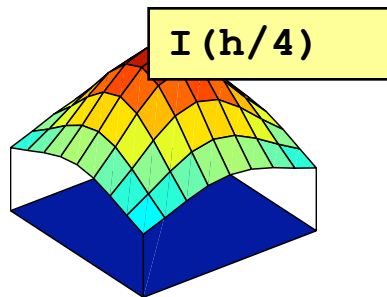
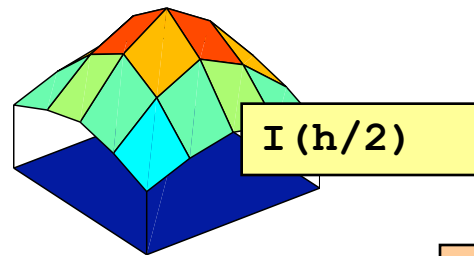
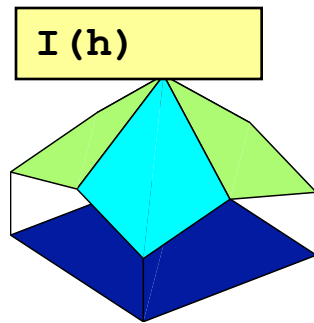
Non :-C



LES CALCULS
SONT BASES
SUR LA
MEME METHODE
ET LA MEME FONCTION

C'EST PAS INDEPENDANT

Pourrait-on utiliser la même idée pour les calculs numériques ?

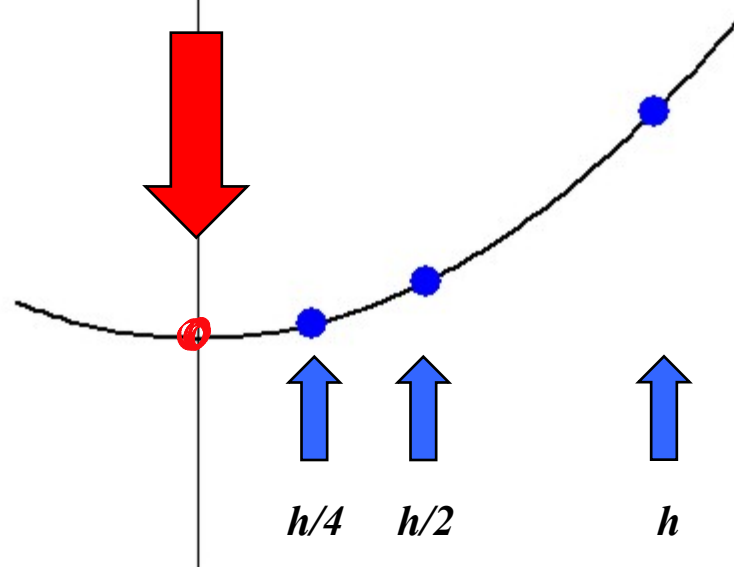


Est-il possible d'effectuer une combinaison linéaire de ces quatre valeurs numériques de l'intégrale afin d'obtenir une approximation encore plus précise que la valeur obtenue avec le plus petit pas ?

Extrapolation de Richardson

$$f\left(\frac{h}{2^i}\right) \quad i = 0, 1, 2, 3, \dots$$

Comment estimer $f(0)$?

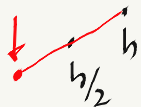


RICHARDSON

$$\begin{array}{rcl}
 -1 f(h) & = & -1 f(0) + \cancel{-1 \frac{h}{2} f'(0) +} \\
 2 f(\frac{h}{2}) & = & 2 f(0) + \cancel{2 \frac{h}{2} f'(0) +}
 \end{array}$$

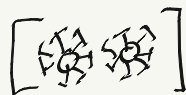
~~PEIT~~

$$f(0) = [2 f(h/2) - f(h)] + \frac{h^2}{4} f'' + \frac{h^3}{8} f''' \quad [\odot]$$



$$\begin{array}{rcl}
 -1 f(h/2) & = & -1 f(0) + \cancel{-1 \frac{h}{2} f'(0) +} \\
 2 f(h/4) & = & 2 f(0) + \cancel{2 \frac{h}{4} f'(0) +}
 \end{array}$$

$$f(0) = [2 f(h/4) - f(h/2)] + \frac{h^2}{16} f'' + \frac{h^3}{64} f'''$$



$$\begin{array}{rcl}
 -1 \frac{h^2}{2} f''(0) + & -1 \frac{h^3}{6} f'''(0) \\
 2 \frac{h^2}{8} f''(0) + & 2 \frac{h^3}{48} f'''(0)
 \end{array}$$

TRES PETIT TRES TRES PETIT

$$f''(\frac{h^2}{4} - \frac{h^2}{2}) = -\frac{h^2}{4} f'' \quad \quad (\frac{h^3}{24} - \frac{h^3}{6}) = \frac{3h^3}{24} = \frac{h^3}{8}$$

$$\begin{array}{rcl}
 -1 \frac{h^2}{8} f''(0) + & -1 \frac{h^3}{96} f'''(0) \\
 2 \frac{h^2}{32} f''(0) + & 2 \frac{h^3}{384} f'''(0)
 \end{array}$$

6x64 :-)

$$\frac{h^2}{16} - \frac{h^2}{8} = -\frac{h^2}{16}$$

$$\frac{h^3}{192} - \frac{h^3}{48} = \frac{-3h^3}{192} = -\frac{h^3}{64}$$

$$f(0) = [2f(h/2) - f(h)] + \cancel{\frac{h^2}{4} f''} + \frac{h^3}{8} f''' \quad [\text{☼}]$$

$$f(0) = [2f(h/4) - f(h/2)] + \cancel{\frac{h^2}{16} f''} + \frac{h^3}{64} f''' \quad [\text{☼☼}]$$

$$f(0) = \frac{4[\text{☼☼}] - [\text{☼}]}{3} + \frac{h^3}{48} f'''$$

$$+ \left[\underbrace{\frac{4}{3} \frac{1}{64}}_{\frac{1}{3} \frac{1}{16}} - \underbrace{\frac{1}{3} \frac{1}{8}}_{\frac{1}{3} \frac{1}{8}} \right] h^3 f'''$$

$\frac{1}{3} \left(-\frac{1}{16} \right)$

$[\text{☺}]$

Extrapolation de Richardson

$$f(h) = f(0) + h f'(0) + \frac{h^2}{2} f''(0) + \dots$$

$$f\left(\frac{h}{2}\right) = f(0) + \frac{h}{2} f'(0) + \frac{h^2}{8} f''(0) + \dots$$

2 mesures

Elimination du terme du premier ordre

$$f\left(\frac{h}{2^i}\right) \quad i = 0, 1, 2, 3, \dots$$

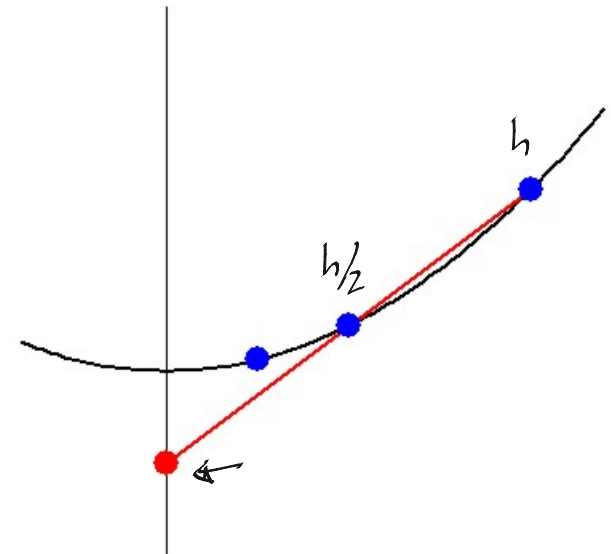
Comment estimer $f(0)$?

$$f\left(\frac{h}{2}\right) - \frac{1}{2}f(h) = \left(1 - \frac{1}{2}\right)f(0) + \left(\frac{h^2}{8} - \frac{h^2}{4}\right)f''(0) + \dots$$

$$= \left(1 - \frac{1}{2}\right)f(0) - \left(1 - \frac{1}{2}\right)\frac{h^2}{4}f''(0) + \dots$$

↓

$$f(0) = \frac{\left(f\left(\frac{h}{2}\right) - \frac{1}{2}f(h)\right)}{\left(1 - \frac{1}{2}\right)} + \frac{h^2}{4}f''(0) + \dots$$



Extrapolation de Richardson

$$f\left(\frac{h}{2}\right) = f(0) + \frac{h}{2} f'(0) + \frac{h^2}{8} f''(0) + \dots$$

$$f\left(\frac{h}{4}\right) = f(0) + \frac{h}{4} f'(0) + \frac{h^2}{32} f''(0) + \dots$$

3 mesures

Elimination du terme du second ordre

$$f\left(\frac{h}{4}\right) - \frac{1}{2}f\left(\frac{h}{2}\right) = \left(1 - \frac{1}{2}\right) f(0) + \left(\frac{h^2}{32} - \frac{h^2}{16}\right) f''(0) + \dots$$

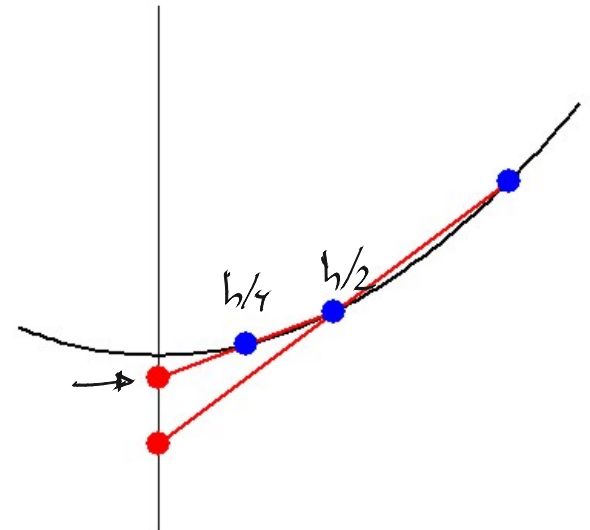
$$= \left(1 - \frac{1}{2}\right) f(0) - \left(1 - \frac{1}{2}\right) \frac{h^2}{16} f''(0) + \dots$$

$$\downarrow$$

$$f(0) = \frac{\left(f\left(\frac{h}{4}\right) - \frac{1}{2}f\left(\frac{h}{2}\right)\right)}{\left(1 - \frac{1}{2}\right)} + \frac{h^2}{16} f''(0) + \dots$$

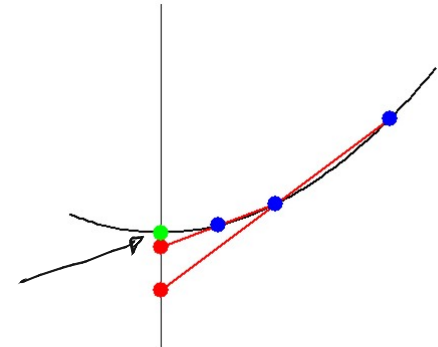
$$f\left(\frac{h}{2^i}\right) \quad i = 0, 2, 3, \dots$$

Comment estimer $f(0)$?



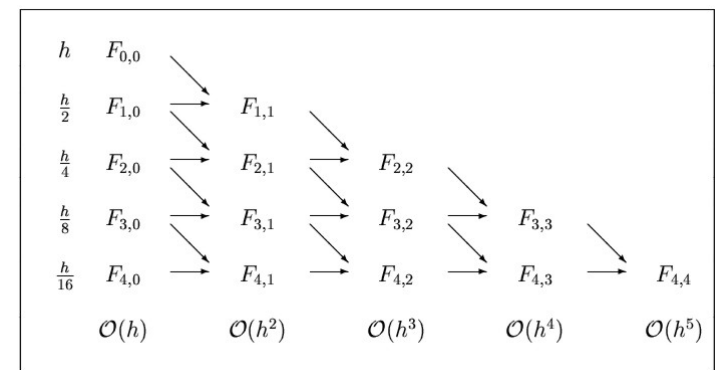
... et finalement

$$f(0) = \frac{\left(\frac{f(\frac{h}{4}) - \frac{1}{2}f(\frac{h}{2})}{(1 - \frac{1}{2})} \right) - \frac{1}{4} \left(\frac{f(\frac{h}{2}) - \frac{1}{2}f(h)}{(1 - \frac{1}{2})} \right)}{\left(1 - \frac{1}{4}\right)} + \frac{h^3}{48} f'''(0) + \dots$$



$$F_{i,k} = \frac{\left(F_{i,k-1} - \frac{1}{2^k} F_{i-1,k-1} \right)}{\left(1 - \frac{1}{2^k} \right)}$$

Disposition pratique des calculs



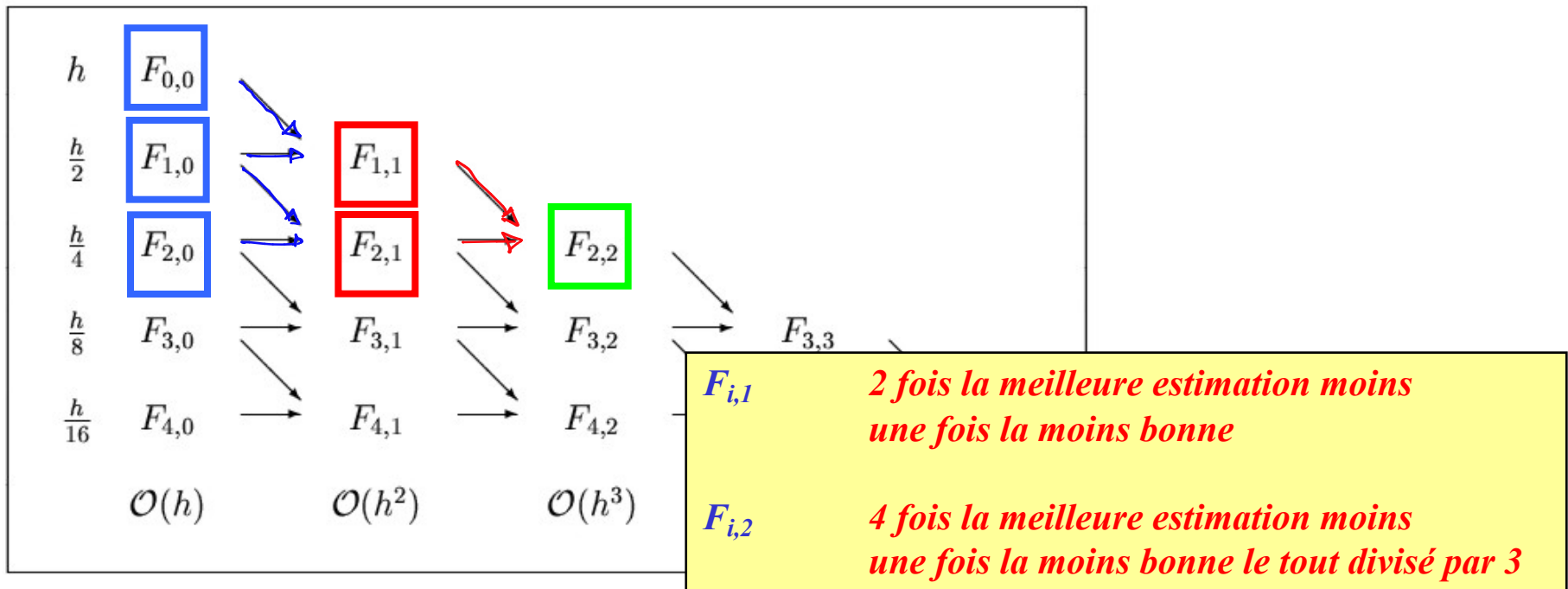
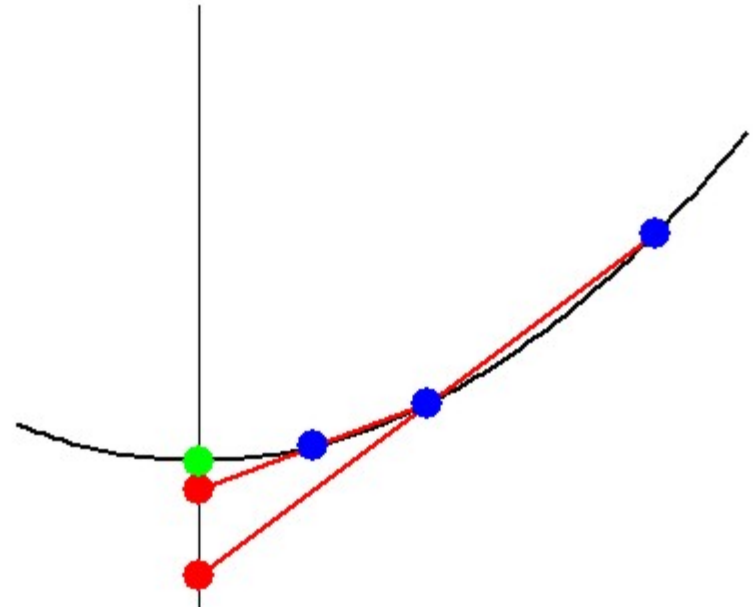
$F_{i,1}$

*2 fois la meilleure estimation moins
une fois la moins bonne*

$F_{i,2}$

*4 fois la meilleure estimation moins
une fois la moins bonne le tout divisé par 3*

Extrapolation de Richardson



Romberg = application de l'extrapolation de Richardson à la méthode composite des trapèzes avec des pas décroissants

$$I^h = \frac{h}{2} (U_0 + U_1)$$

$$I^{h/2} = \frac{h}{4} (U_0 + 2U_1 + U_2)$$

$$I^{h/4} = \frac{h}{8} (U_0 + 2U_1 + 2U_2 + 2U_3 + U_4)$$

$$I^{h/8} = \frac{h}{16} (U_0 + 2U_1 + 2U_2 + 2U_3 + \dots + 2U_7 + U_8)$$

$$I^{h/16} = \frac{h}{32} (U_0 + 2U_1 + 2U_2 + 2U_3 + \dots + 2U_{15} + U_{16})$$

Méthode composite des trapèzes

$h = (b-a)$ pour toutes les expressions

Extrapolation de Richardson

Estimer $I(0)$ sachant que

$$\begin{aligned} I(h) &= I^h \\ I(h/2) &= I^{h/2} \\ I(h/4) &= I^{h/4} \\ I(h/8) &= I^{h/8} \\ I(h/16) &= I^{h/16} \end{aligned}$$

ROMBERG



I_h

$$I_{00}$$

$$I_{11} = 4 \frac{I_{10} - I_{00}}{3}$$

ON A ÉLIMINÉ LE TERM EN $\mathcal{O}(h^2)$



$I_{h/2}$

$$I_{10}$$

$$I_{11}$$

ON ÉLIMINE LE TERM EN $\mathcal{O}(h^4)$

$$\frac{16 I_{21} - I_{11}}{15}$$



$I_{h/4}$

$$I_{20}$$

$$I_{21}$$

$$I_{22}$$

$$\frac{64 I_{32} - I_{22}}{63}$$



$I_{h/8}$

$$I_{30}$$

$$I_{31}$$

$$I_{32}$$

$$I_{33}$$

$\mathcal{O}(h^2)$

TRAPEZES

$\mathcal{O}(h^4)$

SIMPSON

$\mathcal{O}(h^6)$

BOOLE

$\mathcal{O}(h^8)$





I_h

$$\boxed{I_{00}}$$



$I_{h/2}$

$$\boxed{I_{10}}$$



$I_{h/4}$

$$\boxed{I_{20}}$$



$I_{h/8}$

$$\boxed{I_{30}}$$

$$\mathcal{O}(h^2)$$

$$\mathcal{O}(h^4)$$

$$\mathcal{O}(h^6)$$

$$\mathcal{O}(h^8)$$

← CAR ON GETTING ON TERM EN h^2

$$2^2 \frac{I_{10} - I_{00}}{3} = 4 \frac{I_{10} - I_{00}}{3} = I_{11}$$

← CAR $h \rightarrow h/2$

$$I_{22} = \frac{16 I_{21} - I_{11}}{15}$$

$$I_{33} = \frac{64 I_{32} - I_{22}}{63}$$



$$I - I_{00} = \cancel{C h^2} + \cancel{D h^4} + E h^6$$

$$I - I_{10} = \cancel{C \left(\frac{h}{2}\right)^2} + \cancel{D \left(\frac{h}{2}\right)^4} + E \left(\frac{h}{2}\right)^6$$

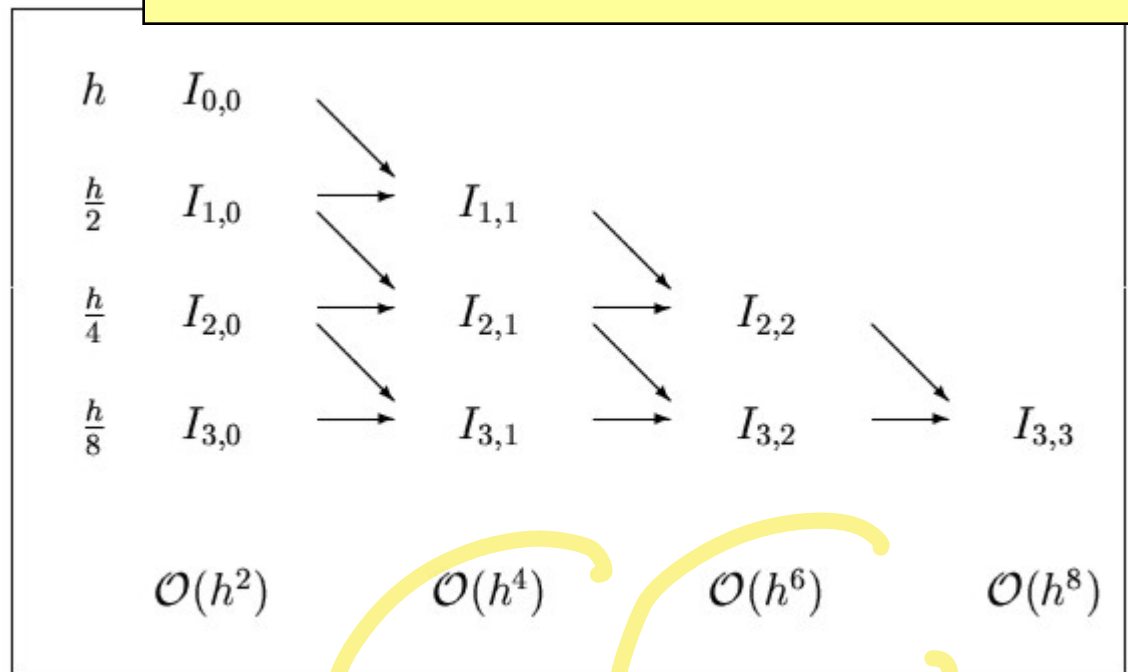
... et pratiquement

$$I_{i,k} = \frac{\left(I_{i,k-1} - \frac{1}{2^{2k}} I_{i-1,k-1} \right)}{\left(1 - \frac{1}{2^{2k}} \right)}$$

Le terme d'erreur ne fait intervenir que des puissances paires....

$I_{i,1}$

*4 fois la meilleure estimation moins
une fois la moins bonne le tout divisé par 3*

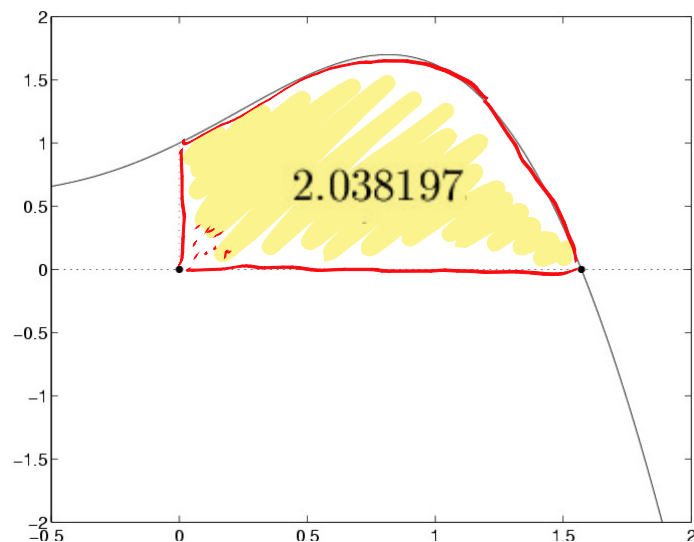


Simpson

Boole

Exemple

$$I = \int_0^{\pi/2} (x^2 + x + 1) \cos x \, dx$$



i	$I_{i,0}$	$I_{i,1}$	$I_{i,2}$	$I_{i,3}$	$I_{i,4}$
0	0.785398				
1	1.726813	2.040617			
2	1.960534	2.038441	2.038296		
3	2.018794	2.038214	2.038199	2.038197	
4	2.033347	2.038198	2.038197	2.038197	2.038197
	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$	$\mathcal{O}(h^{10})$

Romberg :

Implémentation récursive ?

```
h=1 ; d=0
Iq=Integrale exacte
I=Iq-1
while (Iq-I)~=0 :
    I=romberg(d,d,h)
    d=d+1 ;
```



Mon groupe favori.... lundi à 22h15.

« Afin de limiter les boucles trop ennuyeuses nous avons utilisé fonction interne. Mais nous ne nous sommes rendus compte que trop tard qu'une telle fonction ne peut s'appeler elle-même, et c'est ce qui, nous le présumons, engendre une erreur lors de l'exécution du programme.... »

Romberg



```
h=1 ; d=0
Iq=Integrale exacte
I=Iq-1
while (Iq-I) ~=0 :
    I=romberg(d,d,h)
    d=d+1 ;
```

$$I_{i,k} = \frac{I_{i,k-1} - \frac{1}{2^{2k}} I_{i-1,k-1}}{(1 - \frac{1}{2^{2k}})}$$

```
def romberg(i,k,h) :
```

```
    if k==0 :
```

```
        Integration par les trapezes...
```

```
    else :
```

```
        I=(romberg(i,k-1,h)-
```

```
            (romberg(i-1,k-1,h) / ((2**(2*k))) ) / (1 - (1/ (2**(2*k))) )
```

```
    return I
```

Implémentation récursive de la méthode de Romberg

Ecriture très compacte

Efficacité très faible

Quelques moments plus tard...



```
h=1 ; d=0  
Iq=Integrale exacte  
I=Iq-1  
while (Iq-I) !=0 :  
    I=romberg(d,d,h)  
    d=d+1 ;
```

Mon groupe favori, mardi à 19h52

X propos de la monstrosité...

HUm .

Bien , donc c vrmnt très nul , dixit ma soeur is my clock , suffit de remplacer par while ((Iq - I) != 0)

Voilà on est nul

Et le matin...

Mon groupe favori, mercredi à 01h07

« Voici en annexe la version finale. Il a été amélioré et fonctionne à présent bien. »

Mon groupe favori, mercredi à 09h

« Voici en annexe la version finale. Il a été amélioré et fonctionne à présent bien. Ce programme fut fait dans son intégrité par notre groupe 1183. Toute similitude avec un autre programme déjà existant serait purement fortuite... Il nous a semblé que le mail envoyé hier contenait peut-être des fichiers vides suite à une erreur du serveur mail ; nous nous permettons donc de

La version 3 du programme...

```
d=0
p=1/1000
I=0
I_old=0
while ( round(I/p) < round(I_old/p) or I_old==0 ) :
    I_old=I
    I=romberg(f,d,d,0,1)
    d=d+1
```

*On n'est jamais sûr que ce test sera satisfait...
Possibilité réelle de programme ne s'arrêtant
jamais jamais jamais*



La récursivité avec Romberg, c'est aussi une...

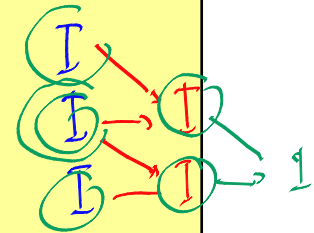


```
h=1;d=0
Iq = Integrale exacte
I = Iq-1
while (d!=6):
    I=romberg(d,d,h)
    d=d+1
```

```
def romberg(i,k,h):
    if k==0 :
        Integration par les trapezes...
    else :
        I=(romberg(i,k-1,h) -
            (romberg(i-1,k-1,h) / (2** (2*k) ))) / (1 - (1/ (2** (2*k) )))
        print(' Etape %i %i -> I  %21.14e' % (i,k,I))
    return I
```

Beaucoup de calculs répétés inutilement...

```
>>>>
I =
    1.3244
Romberg : etape 1 1 -> I = 1.19688584716124e+000 ✓
I =
    1.1969
Romberg : etape 2 1 -> I = 1.19509423681214e+000
Romberg : etape 1 1 -> I = 1.19688584716124e+000 ✓
Romberg : etape 2 2 -> I = 1.19497479612220e+000
I =
    1.1950
Romberg : etape 3 1 -> I = 1.19496650020032e+000
Romberg : etape 2 1 -> I = 1.19509423681214e+000 ✓
Romberg : etape 3 2 -> I = 1.19495798442620e+000
Romberg : etape 2 1 -> I = 1.19509423681214e+000 ✓
Romberg : etape 1 1 -> I = 1.19688584716124e+000 ✓
Romberg : etape 2 2 -> I = 1.19497479612220e+000
Romberg : etape 3 3 -> I = 1.19495771757388e+000
I =
    1.1950
```



Romberg : etape 4 1 -> I = 1.19495821928187e+000

Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓

Romberg : etape 4 2 -> I = 1.19495766722064e+000

Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓

Romberg : etape 2 1 -> I = 1.19509423681214e+000

Romberg : etape 3 2 -> I = 1.19495798442620e+000

Romberg : etape 4 3 -> I = 1.19495766218563e+000

Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓

Romberg : etape 2 1 -> I = 1.19509423681214e+000

Romberg : etape 3 2 -> I = 1.19495798442620e+000

Romberg : etape 2 1 -> I = 1.19509423681214e+000

Romberg : etape 1 1 -> I = 1.19688584716124e+000

Romberg : etape 2 2 -> I = 1.19497479612220e+000

Romberg : etape 3 3 -> I = 1.19495771757388e+000

Romberg : etape 4 4 -> I = 1.19495766196842e+000

I =

1.1950

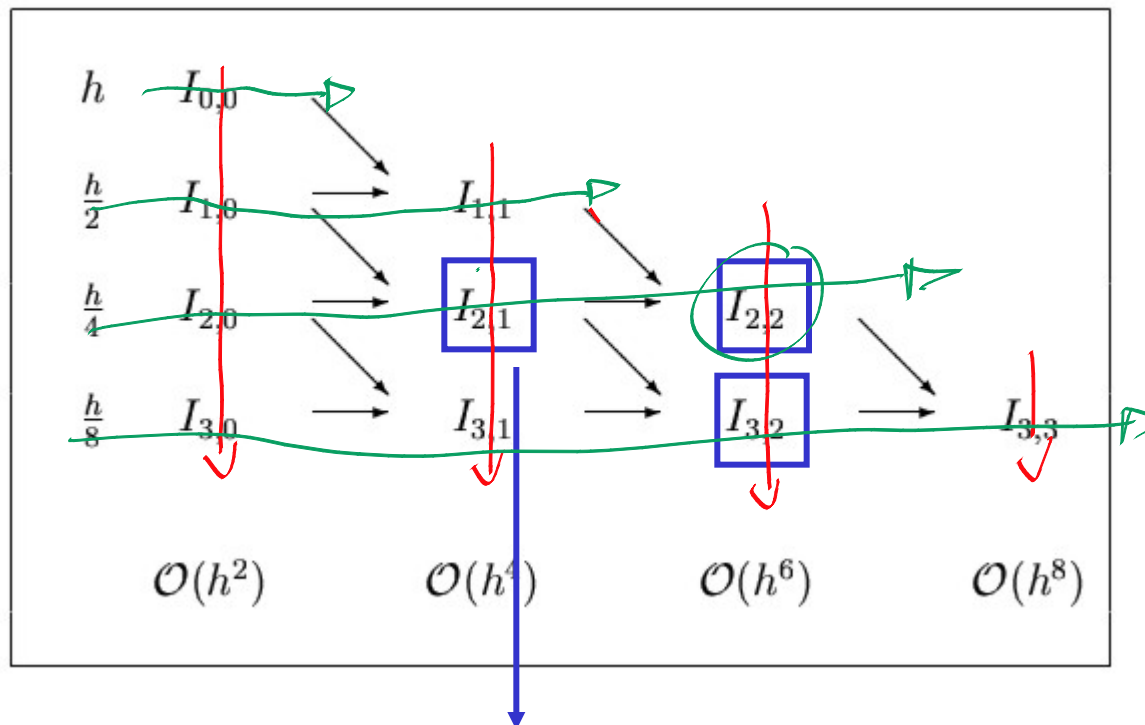


```
Romberg : etape 5 1 -> I = 1.19495769682481e+000
Romberg : etape 4 1 -> I = 1.19495821928187e+000
Romberg : etape 5 2 -> I = 1.19495766199434e+000
Romberg : etape 4 1 -> I = 1.19495821928187e+000
Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓
Romberg : etape 4 2 -> I = 1.19495766722064e+000
Romberg : etape 5 3 -> I = 1.19495766191138e+000
Romberg : etape 4 1 -> I = 1.19495821928187e+000
Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓
Romberg : etape 4 2 -> I = 1.19495766722064e+000
Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓
Romberg : etape 2 1 -> I = 1.19509423681214e+000
Romberg : etape 3 2 -> I = 1.19495798442620e+000
Romberg : etape 4 3 -> I = 1.19495766218563e+000
Romberg : etape 5 4 -> I = 1.19495766191030e+000
Romberg : etape 4 1 -> I = 1.19495821928187e+000
Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓
Romberg : etape 4 2 -> I = 1.19495766722064e+000
Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓
Romberg : etape 2 1 -> I = 1.19509423681214e+000
Romberg : etape 3 2 -> I = 1.19495798442620e+000
Romberg : etape 4 3 -> I = 1.19495766218563e+000
Romberg : etape 3 1 -> I = 1.19496650020032e+000 ✓
Romberg : etape 2 1 -> I = 1.19509423681214e+000
Romberg : etape 3 2 -> I = 1.19495798442620e+000
Romberg : etape 2 1 -> I = 1.19509423681214e+000
Romberg : etape 1 1 -> I = 1.19688584716124e+000
Romberg : etape 2 2 -> I = 1.19497479612220e+000
Romberg : etape 3 3 -> I = 1.19495771757388e+000
Romberg : etape 4 4 -> I = 1.19495766196842e+000
Romberg : etape 5 5 -> I = 1.19495766191025e+000
```

```
I =
    1.1950
```

```
>>
```

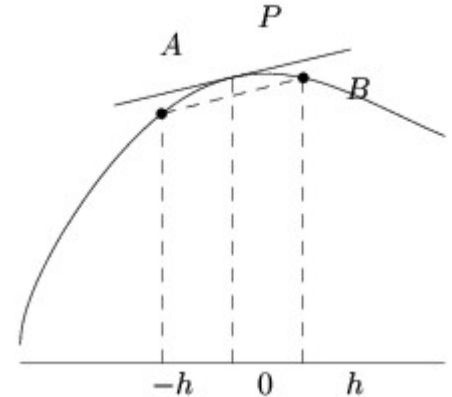
Il est beaucoup plus efficace de suivre le tableau des calculs....



Est nécessaire pour I_{22} et I_{32}

Plan du cours de méthodes numériques

Comment résoudre
numériquement un
problème aux
valeurs initiales ?



Comment interpoler
une fonction ?

Comment dériver
numériquement
une fonction ?

Comment approximer
une fonction ?

Comment résoudre
numériquement un
problème aux
conditions frontières ?

Comment intégrer
numériquement
une fonction ?

Et les équations
non linéaires ?

Et les méthodes itératives ?

*Comment résoudre numériquement
une équation différentielle ordinaire ?*

*Comment résoudre numériquement
une équation aux dérivées partielles ?*

Comment résoudre
numériquement une
équation aux dérivées
partielles ?

$$D = u'(x)$$

$$= \lim_{h \rightarrow 0} \frac{u(x+h) - u(x)}{h}$$

$$D^{h_k} = \frac{u(x+h_k) - u(x)}{h_k} \quad k = 1, 2, \dots, n, \dots$$

Dérivation numérique...

Méthodes conceptuellement simples...

Différences finies d'ordre n

Différences finies centrées et unilatérales

... mais **numériquement instables**

Erreurs d'arrondi

Concept de stabilité numérique

Valeur exacte



$$E^h = D - D^h$$



Approximation numérique

Une petite expérience numérique....

Théoriquement, la méthode avec une arithmétique exacte devrait converger vers la valeur exacte

PYTHON :

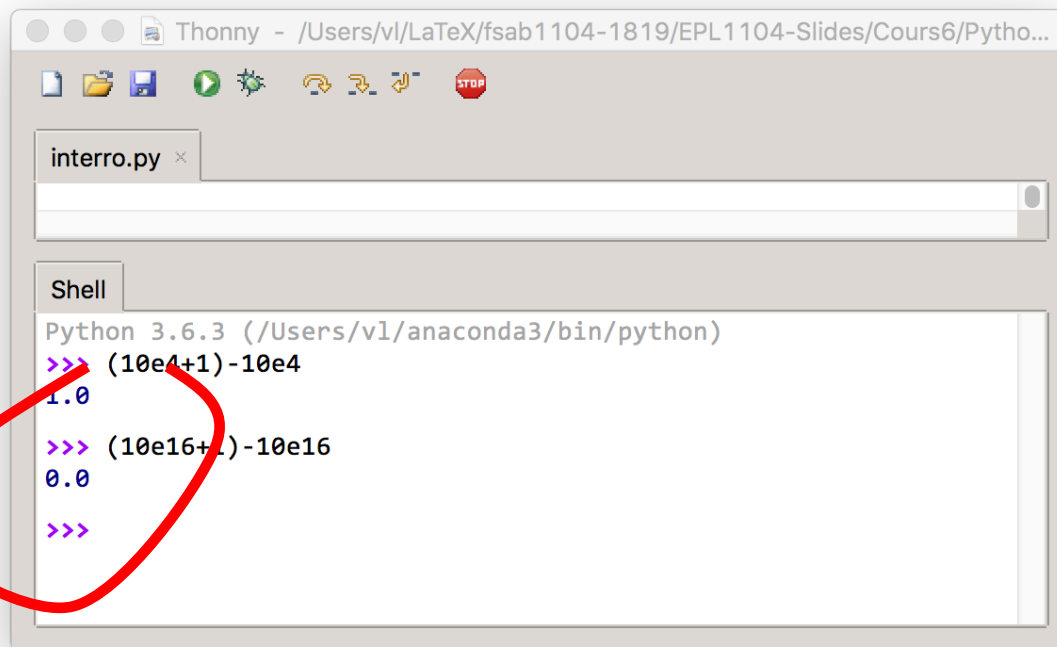
Pratiquement avec Matlab, l'erreur diminue et puis recommence à augmenter...

Il s'agit d'une catastrophe numérique !

h_k	$\exp(1 + h_k)$	D^{h_k}	E^{h_k}
$h_0 = 1$	7.389056099	4.670774270	-1.952492442
$h_1 = 0.1$	3.004166024	2.858841955	-0.140560126
$h_2 = 0.01$	2.745601015	2.731918656	-0.013636827
$h_3 = 0.001$	2.721001470	2.719641423	-0.001359594
$h_4 = 0.0001$	2.718553670	2.718417747	-0.000135919
$h_5 = 0.00001$	2.718309011	2.718295420	-0.000013591
$h_6 = 10^{-6}$	2.718284547	2.718283187	-0.000001359
$h_7 = 10^{-7}$	2.718282100	2.718281964	-0.000000140
$h_8 = 10^{-8}$	2.718281856	2.718281777	0.000000007
$h_9 = 10^{-9}$	2.718281831	2.718281600	-0.000000060
$h_{10} = 10^{-10}$	2.718281829	2.718278935	-0.000001548
$h_{11} = 10^{-11}$	2.718281828	2.718270053	-0.000032634
$h_{12} = 10^{-12}$	2.718281828	2.718270053	-0.000432314
$h_{13} = 10^{-13}$	2.718281828	2.713385072	0.000455864
$h_{14} = 10^{-14}$	2.718281828	2.664535259	-0.035071273
$h_{15} = 10^{-15}$	2.718281828	2.664535259	-0.390342640
$h_{16} = 10^{-16}$	2.718281828	0.000000000	2.718281828
$h_{17} = 10^{-17}$	2.718281828	0.000000000	2.718281828
$h_{18} = 10^{-18}$	2.718281828	0.000000000	2.718281828
$h_{19} = 10^{-19}$	2.718281828	0.000000000	2.718281828

Erreurs d'arrondi : c'est quoi ?

Un exemple très simple qui montre l'importance des erreurs d'arrondi



```
Thonny - /Users/vl/LaTeX/fsab1104-1819/EPL1104-Slides/Cours6/Pytho...  
interro.py x  
Shell  
Python 3.6.3 (/Users/vl/anaconda3/bin/python)  
>>> (10e4+1)-10e4  
1.0  
>>> (10e16+1)-10e16  
0.0  
>>>
```

$$(10^4 + 1) - 10^4$$

$$(10^{16} + 1) - 10^{16}$$

Valeur exacte

Nombre de chiffres significatifs

$$\left| \frac{x - \tilde{x}}{x} \right| < \frac{1}{2} 10^{-d}$$

Représentation dans l'ordinateur

Calculateur avec 4 chiffres pour représenter un nombre :

Effectuons l'opération

1.348 + 9.999

Valeur exacte

11.347

Valeur représentée par le calculateur

11.35

Arithmétique
à précision
finie

*Comment utiliser astucieusement la
mémoire disponible ?*

x	$\tilde{x}$	nombre de chiffres significatifs : d
<u>1.000001</u>	<u>0.999998</u>	$\left \frac{x - \tilde{x}}{x} \right \approx 0.000003 < \frac{1}{2} 10^{-5}$ 5
0.001234	0.001231	$\left \frac{x - \tilde{x}}{x} \right \approx 0.002 < \frac{1}{2} 10^{-2}$ 2
<u>0.000013</u>	<u>0.000009</u>	$\left \frac{x - \tilde{x}}{x} \right \approx 0.31 < \frac{1}{2}$ 0

$$\begin{array}{rclclcl}
 0.5000 & 10^{12} & = & 5.000 & 10^{11} & = & 5 \overbrace{00000000000}^{11 \text{ zéros}} .0000000000000000 \\
 -0.3275 & 10^{04} & = & -3.275 & 10^{03} & = & -3275.0000000000000000 \\
 0.5723 & 10^{01} & = & 5.723 & 10^{00} & = & 5.723000000000000000 \\
 0.9422 & 10^{-11} & = & 9.422 & 10^{-12} & = & 0.\underbrace{00000000000}_{11 \text{ zéros}} 9422
 \end{array}$$

**Représentation
normalisée : le premier
chiffre de la mantisse
est non nul**

Signe



Exposant



$\pm 0.m \ 10^{\pm e}$



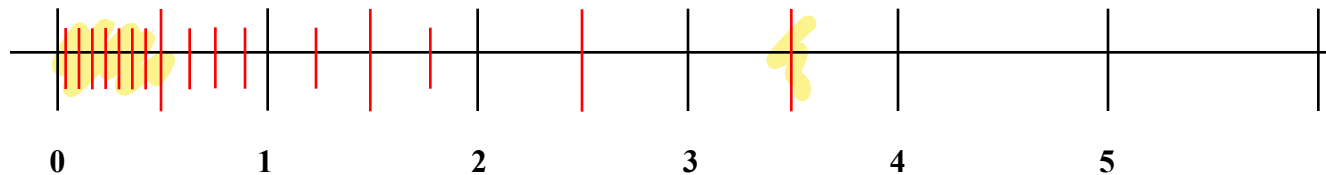
Mantisse

**Arithmétique
en virgule
flottante**

*ou l'utilisation astucieuse de la
mémoire disponible...*

IEEE single (32bits)	mantisse	23+1 bits
	exposant	8 bits
	limites (range)	10^{-38} 10^{+38}
	arrondi à l'unité	2^{-24} (approximativement 10^{-8})
IEEE double (64 bits)	mantisse	52+1 bits
	exposant	11 bits
	limites (range)	10^{-308} 10^{+308}
	arrondi à l'unité	2^{-53} (approximativement 10^{-16})

En réalité, un ordinateur utilise des nombres binaires...



$$(1 + 10^{-16}) - 1$$

↘ 1 :-)

Il est important de réaliser que les nombres représentables ne sont pas également espacés

Exemple représenté : mantisse de 3 bits, exposant compris entre -1 et 3.

**Calcul en simple (32bits)
ou double précision (64bits)**

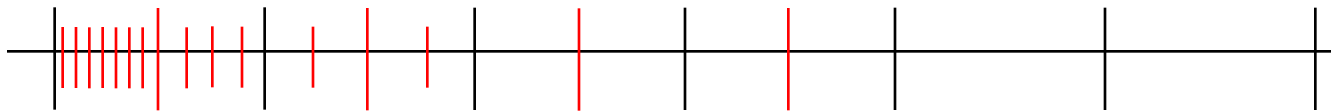
Python : l'instruction du jour...

```
>>> import numpy as np
>>> np.finfo(float).eps
2.220446049250313e-16
>>> np.finfo(float).min
-1.7976931348623157e+308
>>> np.finfo(float).max
1.7976931348623157e+308
>>> np.finfo(float).tiny
2.2250738585072014e-308
```

realmax = 1.7977e+308

realmin = 2.22507e-308

Il y a beaucoup plus de nombres
représentables près de $x_{\min}$ que
près de $x_{\max}$



$$x_{\min} = 2.2250738507201e-308$$

$$\underline{x}_{\min} = 2.2250738507202e-308$$

$$x_{\max} = 1.7976931348623158e+308$$

$$\underline{x}_{\max} = 1.7976931348623157e+308$$

$$x_{\max} - \underline{x}_{\max} \text{ est de l'ordre de } 10^{292}$$

$$\underline{x}_{\min} - x_{\min} \text{ est de l'ordre de } 10^{-323}$$

Les dangers de la soustraction...

ou pourquoi les formules de différences sont numériquement instables.

Représentation avec 12 chiffres

Effectuons l'opération $x-y$

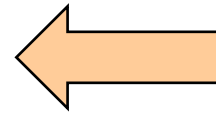
Valeur exacte de x	3.1234567845578
Valeur de x dans l'ordinateur	3.12345678456
Valeur exacte de y	3.1234567844660
Valeur de y dans l'ordinateur	3.12345678447
Résultat exact	0.0000000000918
Valeur approchée	0.00000000009

*12 chiffres significatifs
pour les données*

*1 chiffre significatif
pour le résultat*

Les facéties du calcul en virgule flottante...

```
>>> a=1; b=1
>>> while (a+b) != a:
...     b=b/2
...
>>> print(b)
1.1102230246251565e-16
```



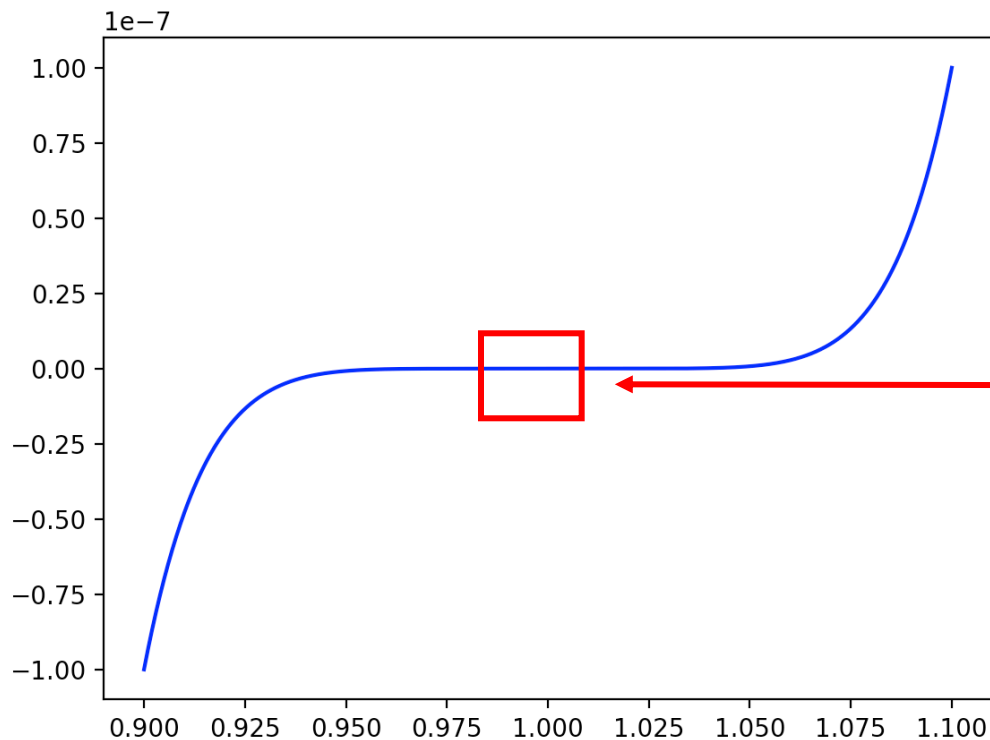
*Ce programme ne devrait jamais se terminer dans l'espace des nombres réels...
Mais il se termine très très rapidement dans l'espace des nombres représentables par **python***

Les règles de l'associativité et de la distributivité sont violées !

```
>>> a=1.0e+308; b=1.1e+308; c=-1.001e+308
>>> print(a+(b+c))
1.099e+308
>>> print((a+b)+c)
inf
>>>
```

$$\underbrace{(4+2)}_6 + 6 = 12$$
$$4 + \underbrace{(2+6)}_8 = 12$$

Les graphes... le meilleur et le pire !



Faisons un zoom !

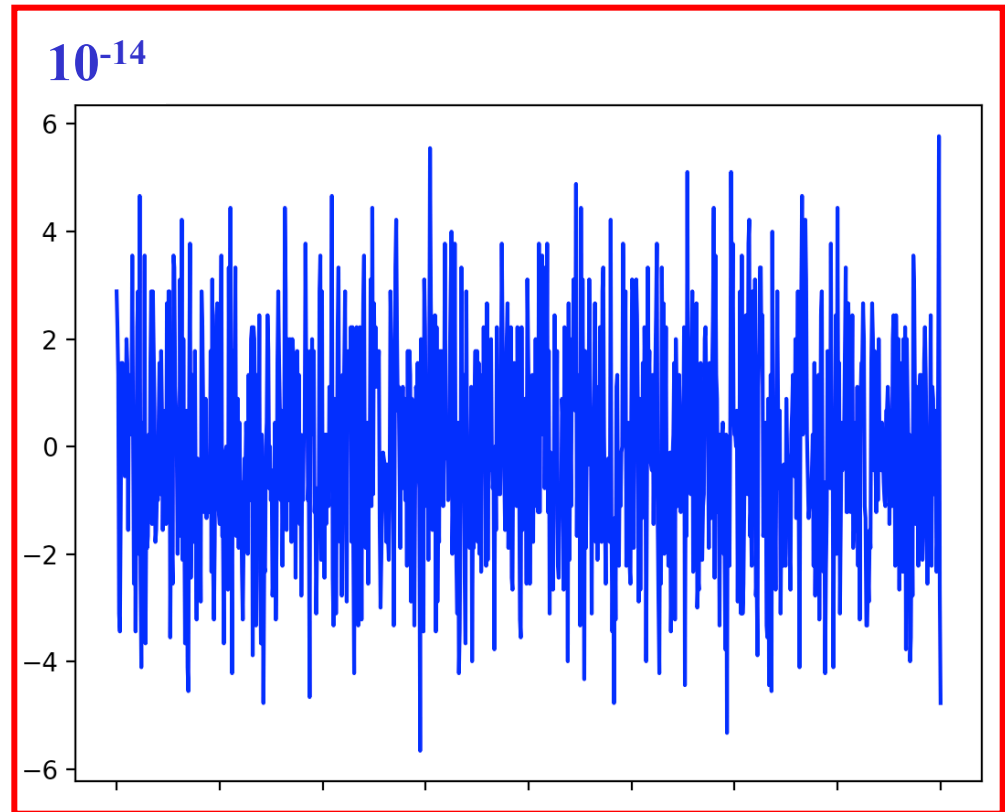
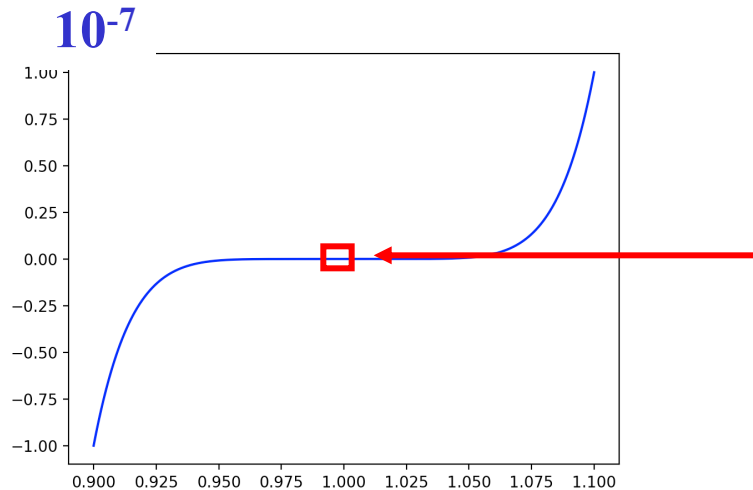
$$\underbrace{x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3}_{p(x)} - \underbrace{21x^2 + 7x + 1}$$

```
p = [1,-7,21,-35,35,-21,7,-1]
x = linspace(0.9,1.1,1000)
plt.plot(x,polyval(p,x),'-b')
```

$$(1-x)^7$$

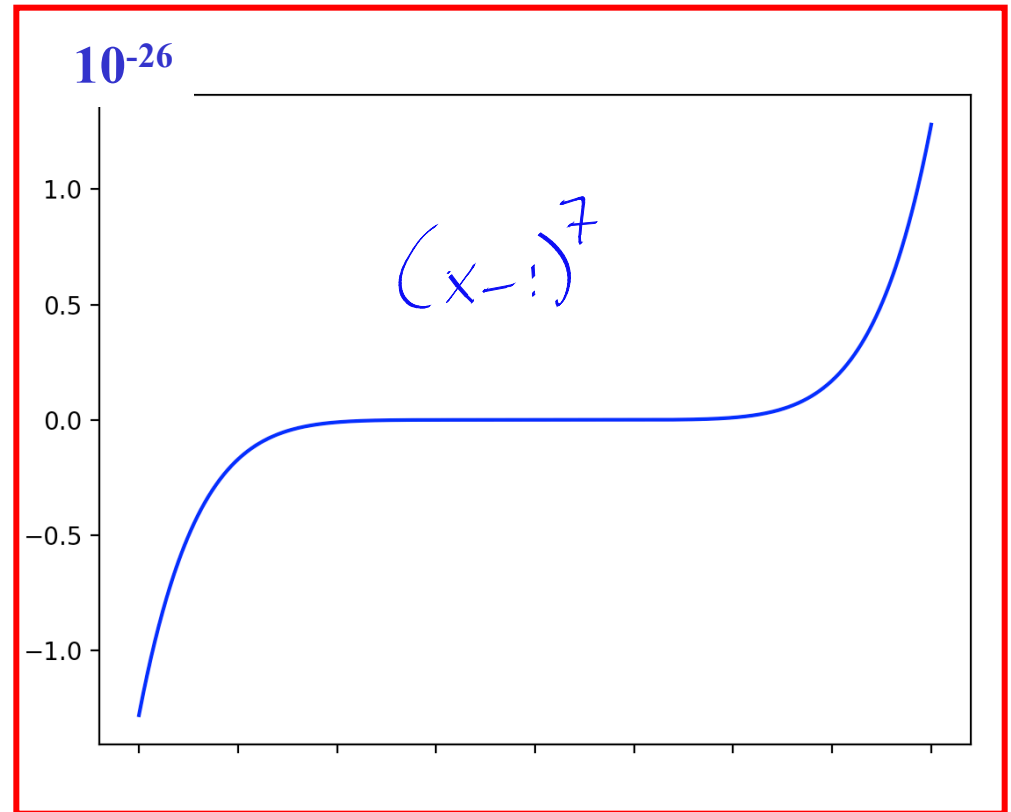
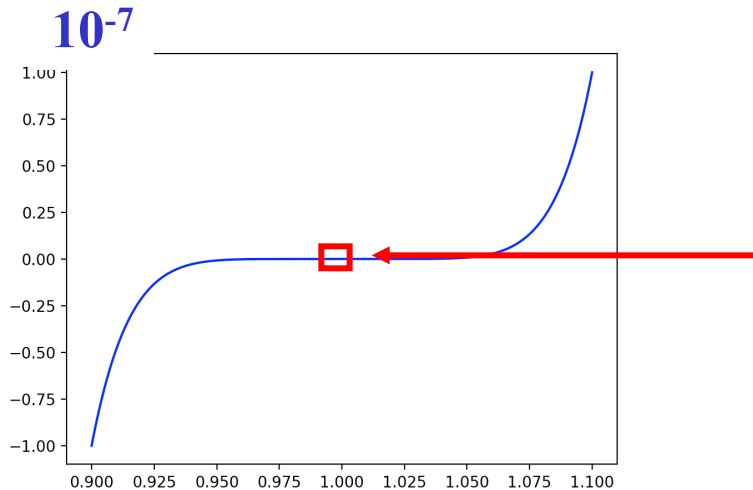
Et voilà un résultat débile :-)

```
p = [1,-7,21,-35,35,-21,7,-1]
x = linspace(1-2e-4,1+2e-4,1000)
plt.plot(x,polyval(p,x),'-b')
```



Et le même graphe correct :-)

```
x = linspace(1-2e-4,1+2e-4,1000)
plt.plot(x,power((x-1.0),7),'-b')
```



MODELISONS L'ORDINATEUR !

$$D^h = \frac{U_h - U_{-h}}{2h} \simeq v'(0)$$

$$U_h = \cancel{U_0} + h U_0' + \cancel{\frac{h^2}{2} U_0''} + \frac{h^3}{6} U_0''' + \dots$$

$$- U_{-h} \quad \quad \quad - \quad \quad \quad + \quad \quad \quad -$$

$$U_h - U_{-h} = \quad \quad \quad 2h U_0' \quad \quad \quad \frac{2h^3}{6} U_0'''$$

$$D^h = U_0' + \frac{h^2}{6} U_0'''$$

$$\boxed{v'(0)} =$$

$$\frac{U_h - U_{-h}}{2h}$$

$$\boxed{\frac{\tilde{U}_h - \tilde{U}_{-h}}{2h}}$$

CALCUL
DE L'ORDINATEUR

$$+ (-1) \frac{h^2}{6} v'''(\xi)$$

$$+ \underbrace{\frac{\tilde{e}_h - \tilde{e}_{-h}}{2h}}_{\frac{2\varepsilon}{2h}}$$

ERREUR

$$v'(0) =$$

$$\frac{U_h - U_{-h}}{2h}$$

$$\frac{\tilde{U}_h - \tilde{U}_{-h}}{2h}$$

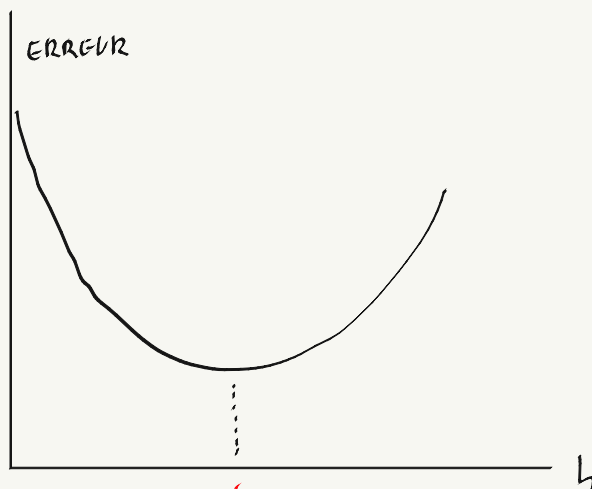
CALCUL
DE L'ORDINATEUR

$$+ (-1) \frac{h^2}{6} v'''(\xi)$$

$$+ \underbrace{\frac{\tilde{e}_h - \tilde{e}_{-h}}{2h}}_{\frac{2\varepsilon}{2h}}$$

ERREUR

$$(-v'''(\xi))$$



$$h = \sqrt{\frac{3\varepsilon}{C}}$$

$$\underbrace{\frac{\varepsilon}{h} + C \frac{h^2}{6}}_{f(h)} = \text{ERREUR}$$

$$f'(h) = -\frac{\varepsilon}{h^2} + \frac{2Ch}{6} = 0$$

$$h^3 = \frac{3\varepsilon}{C}$$

Propagation des erreurs d'arrondi pour les différences

$$u'(0) = \underbrace{\frac{(U_h - U_{-h})}{2h}}_{D^h} + \underbrace{(-1)\frac{h^2}{6}u^{(3)}(\xi)}_{E^h}$$

Erreur de discrétisation

Erreurs d'arrondi sur les données

$$\begin{aligned} U_h &= \tilde{U}_h + \tilde{e}_h \\ U_{-h} &= \tilde{U}_{-h} + \tilde{e}_{-h} \end{aligned}$$

Pratiquement...

Erreur de discrétisation



Propagation des
erreurs
d'arrondi



$$\begin{aligned} u'(0) &= \frac{(U_h - U_{-h})}{2h} + (-1)\frac{h^2}{6}u^{(3)}(\xi) \\ &\downarrow \\ u'(0) &= \underbrace{\frac{(\tilde{U}_h - \tilde{U}_{-h})}{2h}}_{\tilde{D}^h} + \underbrace{\frac{(\tilde{e}_h - \tilde{e}_{-h})}{2h}}_{\tilde{E}^h} + \underbrace{(-1)\frac{h^2}{6}u^{(3)}(\xi)}_{E^h} \end{aligned}$$

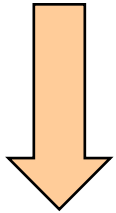
On néglige l'erreur d'arrondi commise sur les opérations pour calculer la différence, car elle est peu importante par rapport à l'erreur propagée



Erreur totale

Estimation du pas optimal

Calcul de h qui minimise l'erreur totale en valeur absolue



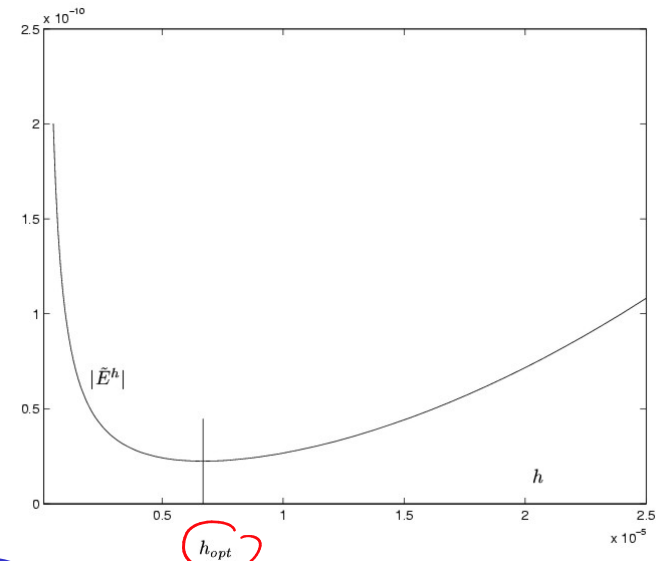
$$h = \left(\frac{3\epsilon}{C} \right)^{1/3}$$

$$|\tilde{E}^h| \leq \frac{\epsilon}{h} + \frac{C h^2}{6}$$

Exemple

calcul de la dérivée de $\cos(0.7)$

h	$\tilde{D}^h$	$\tilde{E}^h$
1	-0.54209049171057	0.10212719552713
0.1	-0.64314452781256	0.00107315942513
0.01	-0.64420695032992	0.00001073690777
0.001	-0.64421757986810	0.00000010736959
0.0001	-0.64421768616374	0.00000000107395
1.0000000e-005	-0.64421768722345	0.00000000001424
1.0000000e-006	-0.64421768725120	-0.00000000001351
1.0000000e-007	-0.64421768697365	0.00000000026404
1.0000000e-008	-0.64421769030432	-0.000000000306663
1.0000000e-009	-0.64421767920209	0.000000000803561
1.0000000e-010	-0.64421745715748	0.00000023008021
1.0000000e-011	-0.64421801226899	-0.00000032503130
1.0000000e-012	-0.64420691003875	0.00001077719894
1.0000000e-013	-0.64448446579490	-0.00026677855721
1.0000000e-014	-0.64392935428259	0.00028833295510
1.0000000e-015	-0.61062266354384	0.03359502369385
1.0000000e-016	-0.555111151231258	0.08910617492511
1.0000000e-017	0.000000000000000	0.64421768723769
1.0000000e-018	0.000000000000000	0.64421768723769
1.0000000e-019	0.000000000000000	0.64421768723769



Application de l'extrapolation de Richardson à des différences centrées avec des pas décroissants

$$u'(0) = \frac{(U_h - U_{-h})}{2h} - \frac{h^2}{6} u^{(3)}(\xi)$$

$$D_h \quad D_{h/10} \quad \rightarrow \quad D_* = 100 \frac{D_{h/10} - D_h}{99}$$

Différences centrées $h=0.1$

Estimer $D(0)$ sachant que

$$\begin{aligned} D(h) &= -0.64314452781256 \\ D(h/10) &= -0.64420695032992 \\ D(h/100) &= -0.64421757986810 \\ D(h/1000) &= -0.64421768616374 \end{aligned}$$

Extrapolation de Richardson

$$\frac{100 D_{10} - D_{00}}{99}$$

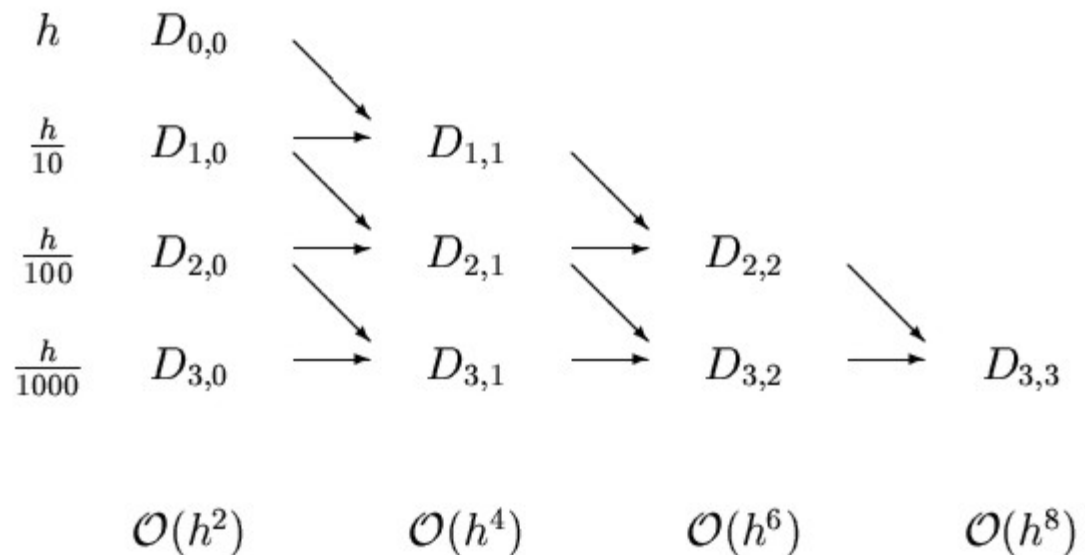
$$D_{11} = \frac{D_{10} - \frac{1}{100} D_{00}}{1 - \frac{1}{100}}$$

$$D_{i,k} = \frac{\left(D_{i,k-1} - \frac{1}{10^{2k}} D_{i-1,k-1} \right)}{\left(1 - \frac{1}{10^{2k}} \right)}$$

... et pratiquement

Le terme d'erreur ne fait intervenir que des puissances paires....

$D_{i,1}$ 100 fois la meilleure estimation moins une fois la moins bonne le tout divisé par 99



$h=0.1, h/10, h/100, h/1000$

Exemple

i	$D_{i,0}$	$D_{i,1}$	$D_{i,2}$	$D_{i,3}$
0	-0.64314452781256			
1	-0.64420695032992	-0.64421768187050		
2	-0.64421757986810	-0.64421768723717	-0.64421768723771	
3	-0.64421768616374	-0.64421768723743	-0.64421768723743	-0.64421768723743
	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$

Inutile, pourquoi ?

$h=0.1, h/2, h/4, h/8$

i	$D_{i,0}$	$D_{i,1}$	$D_{i,2}$	$D_{i,3}$
0	-0.64314452781256			
1	-0.64394929675235	-0.64421755306561		
2	-0.64415058332564	-0.64421767885006	-0.64421768723569	
3	-0.64420091086648	-0.64421768723743	-0.64421768723765	-0.64421768723766
	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$

$$\frac{10^6 D_{32} - D_{22}}{10^6 - 1}$$

ERREUR D'ARRONDI
DU CALCUL EN VIRGULE
FLOTTANTE
PRECISION
 10^{-6} !!!