TD S12 (TI) - Corrigé

I1 - Nombre à virgule flottante

$x = \underbrace{0}_{s}\ \underbrace{00100000011}_{e'}\ \underbrace{11010011100101011000000000000011110000000000000000}_{m}$

$s = 0 \Rightarrow x > 0$

$e' = 1 + 2 + 2^8 = 259 \Rightarrow e = e' - 1023 = -764$

$m = 2^{-1} + 2^{-2} + 2^{-4} + 2^{-7} + 2^{-8} + 2^{-9} + 2^{-12} + 2^{-13} + 2^{-14} + 2^{-16} + 2^{-17} + 2^{-31} + 2^{-32} + 2^{-33} + 2^{-34}$

$= 0,826999398$

$x = +1,826999398 \times 2^{-764} = 1.8828759879171846e-230$

2. $N = 6,02 \cdot 10^{23} = (-1)^s \cdot (1+m) \times 2^{\overbrace{e' - 1023}^{e}} \rightarrow s\ e'\ m$

s, m, e' ?

$s = 0$

$1 \leq \underbrace{\quad}_{} < 2$

Cherchons e : $N = \underbrace{(1+m)}_{2^r} \times 2^e$ avec $0 \leq r < 1$

$\Rightarrow \log_2 N = \log_2(2^r) + \log_2(2^e)$

$\log_2 N = e + r$ ← partie décimale.

$\uparrow \in \mathbb{Z}$

$\Rightarrow e = \lfloor \log_2 N \rfloor$ puis codé en base 2.

$e = 78$

puis $r = \log_2 N - e \Rightarrow 1 + m = 2^r$

$\Rightarrow 1 + m = 2^{\log_2 N - e}$

$\Rightarrow m = 2^{\log_2 N - e} - 1.$

$m = 0,99185015\ldots$

↳ puis convertir m en base 2 "décimal"

- - - - -

2. Comparaison avec des flottants

```
c=0
x=1.0
y=x+1.0
while y-x==1.0:
    x=x*2.0
    y=x+1.0
    c=c+1
```

1/ Si $x, y \in \mathbb{R}$, alors à chaque itération $y-x=1$ donc la boucle est infinie

2/ Test sur machine : le programme s'arrête

3/ Ajout d'un compteur c

Arrêt pour $c = 53 = 52 + 1$

4/ Arrêt $\Leftrightarrow \exists$ une itération n tq : $y - x \neq 1$ vrai

Montrons qu'il y a absorption i.e $y = x + 1 = x$

A l'itération i : $x_i = 2^i$

$y_i = 2^i + 1$

donc $\frac{1}{x_i} = 2^{-i}$

Si $\frac{1}{x_i} < 2^{-52}$ alors $x_{i+1} = x$

$\Leftrightarrow 2^{-i} < 2^{-52}$

$\Leftrightarrow i \geq 53 \Longrightarrow$ absorption à la 53^{e} itérations !

Ce que montre bien le compteur.

I3. Résolution d'une équat° du second degré

```
def zero_poly_deg_2(a,b,c):

    delta=b**2-4*a*c

    if delta>0:
        x1=1/(2*a)*(-b-sqrt(delta))
        x2=1/(2*a)*(-b+sqrt(delta))
    elif delta<0:
        x1=1/(2*a)*(-b-sqrt(-delta)*1j)
        x2=1/(2*a)*(-b+sqrt(-delta)*1j)
    else:
        x1=x2=-b/(2*a)

    return x1,x2
```

zéros_poly_deg_2 (1.2, 1.5e8, 4e-4)
renvoie (-125000000, 0.0).
Or il est évident que 0 n'est pas solution.
Mathématiquement, les racines s'écrivent :

$x_\pm = \frac{1}{2a}(-b \pm \sqrt{\Delta})$

avec $\Delta = b^2 - 4ac$

Calculons Δ : $\Delta = (1,5\times10^8)^2 - 4\times1,2\times4\times10^{-4}$

$= \underbrace{2,25\times10^{16}}_{\alpha} - \underbrace{9,6\times10^{-4}}_{\beta}$

or $\frac{\beta}{\alpha} \sim 10^{-20} < 2^{-52} \Rightarrow fl(\alpha-\beta) = \alpha = b^2$! Absorption !

Dans, ici, $\Delta = b^2 \Rightarrow r_\pm = \frac{1}{2,4}(-b \pm b)$

$\Leftrightarrow \begin{cases} r_+ = 0 \\ r_- = \frac{2b}{2,4} = \frac{1,5}{1,2}\times10^8 = 1,25\times10^8. \end{cases}$

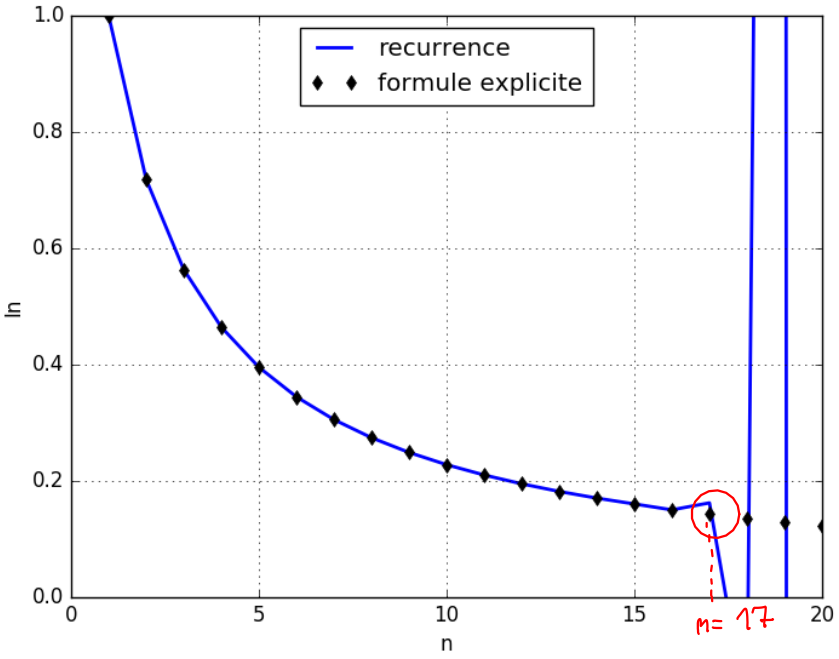
L'erreur du calcul des racine résulte de l'absorption qui se produit lors du calcul de Δ.

I4- Divergence numérique

Soit la suite I_n définie :

- explicitement par : $I_n = \int_0^1 x^n e^x dx$
- par récurrence par : $\begin{cases} I_0 = e-1 \\ I_n = e - nI_{n-1}. \end{cases}$

Expérimentalement, on observe les résultats suivants :



On observe une divergence entre le calcul explicite et le calcul itératif fondé sur la relation de récurrence définissant la suite.

Cette divergence est notable à partir de $n = 17$ et s'amplifie au rang suivant.

Pour comprendre, calculons les termes de la suite I_n.

$I_0 = e - 1$ mais e étant irrationnel, dans $\mathbb{F}$, $fl(e) \neq e$.

On peut supposer que l'erreur sur e est de l'ordre de $\varepsilon = 2^{-32}$

Donc $fl(e) = e + \varepsilon$

D'où $fl(I_0) = I_0 + \varepsilon$ erreur de représentation de I_0.

Calculons en ne tenant compte que l'erreur de représentation de I_0.

$I_1 = e - I_0 \Rightarrow fl(I_1) = fl(e) - fl(I_0) = e + \varepsilon - (I_0 + \varepsilon)$

$= e - I_0$

$= e - (e - 1) = 1$

Jusqu'ici tout va bien !

De même pour $I_2, I_3, \dots I_n$.

$I_2 = e - 2I_1 \Rightarrow fl(I_2) = fl(e) - 2fl(I_1) = e + \varepsilon - 2$

$I_3 = e - 3I_2 \Rightarrow fl(I_3) = fl(e) - 3fl(I_2) = e + \varepsilon - 3(e + \varepsilon - 2)$

$= e + \varepsilon - 3e - 3\varepsilon + 6 = -2e + 6 - 2\varepsilon$

Les erreurs ne se compensent plus !

$I_4 = e - 4I_3 \Rightarrow fl(I_4) = fl(e) - 4fl(I_3) = e + \varepsilon - 4(-2e + 6 - 2\varepsilon)$

$= 9e - 24 + 9\varepsilon$

$\vdots$

Plus généralement :

$I_n = e - nI_{n-1}$

$= e - n(e - (n-1)I_{n-2})$

$= e - n(e - (n-1)(e - (n-2)I_{n-3}))$

$\vdots$

$$I_n = e - n(e - (n-1)(e - (n-2)(\dots - 2(e - 1 \times (e - I_0)))\dots))$$

En développant l'expression, il apparaît le terme $n!I_0$ qui contient l'ordre de grandeur de l'erreur commise à savoir :

$fl(n!I_0) = n!\,fl(I_0) = n!I_0 + n!\varepsilon$ ← ordre de grandeur de l'erreur commise au rang n. : δ_n

A.N. : $n = 17$, $\delta_{17} \approx 0{,}09$
$n = 18$, $\delta_{18} \approx 1{,}42$
$n = 19$, $\delta_{19} \approx 27$

} cohérent avec le graphe !

Finalement, les oscillations apparentes de la suite résulte de l'erreur de représentation de e en machine. (⚠ toutes les suites ne sont pas aussi sensible à cette erreur).

I 5 - Erreur sur les multiplications (facultatif)

Nous raisonnons sur l'exemple de la multiplication (cf question 3)

1/ Mq : $\left|\frac{fl(A\times B) - A\times B}{A\times B}\right| \leq \varepsilon$

Soit : $A = (1+m)\times 2^{e_1}$ $\quad$ $B = (1+m')\times 2^{e_2}$ $\quad$ Mantisse. $0 \leq m < 1$, $0 \leq m' < 1$

au pire $\Rightarrow$ $fl(A) = (1+m\pm\varepsilon)\times 2^{e_1}$, avec $fl(B) = (1+m'\pm\varepsilon)\times 2^{e_2}$,

Alors : $fl(A)\times fl(B) = (1+m\pm\varepsilon)(1+m'\pm\varepsilon)\times 2^{e_1+e_2}$

$= \left((1+m)(1+m') + [\pm(1+m)\pm(1+m')]\varepsilon + \underbrace{\varepsilon^2}_{O(\varepsilon)}\right)\times 2^{e_1+e_2}$.

$fl(A\times B) = \left((1+m)(1+m') + [\pm(1+m)\pm(1+m')]\varepsilon\right)\times 2^{e_1+e_2}$.

Soit : au pire $\left|\frac{fl(A\times B) - A\times B}{A\times B}\right| = \frac{\left((1+m)(1+m') + [\pm(1+m)\pm(1+m')]\varepsilon\right)2^{e_1+e_2} - (1+m)(1+m')\cdot 2^{e_1+e_2}}{(1+m)(1+m')\times 2^{e_1+e_2}}$

$= \left|\frac{\pm(1+m')\pm(1+m')}{(1+m)(1+m')}\right|\varepsilon \quad \approx \quad \varepsilon$

$\underbrace{\phantom{\left|\frac{\pm(1+m')\pm(1+m')}{(1+m)(1+m')}\right|}}_{\sim 1 \text{ car } 0\leq m<1 \text{ et } 0\leq m'<1}$

Finalement, $\boxed{\left|\frac{fl(A\times B) - A\times B}{A\times B}\right| \leq \varepsilon.}$

"Semi-rigoureux" en fait l'erreur ne dépasse pas $\max\left(\left|\frac{2+m+m'}{(1+m)(1+m')}\right|\right)\varepsilon$ peut dépasser 1 pour $m = m' = 95$ par exemple

2/ Sur 64 bits, 52 sont réservés au codage de la mantisse donc :

$\varepsilon = 2^{-52} \approx 2{,}22\times 10^{-16}$

3/ Erreur maximale sur 10^6 multiplications successives.

On a montré que :

$fl(A)\times fl(B) = A\times B + [\pm(1+m)\pm(1\pm m')]\varepsilon\times 2^{e_1+e_2} + \varepsilon^2\times 2^{e_1+e_2}$

De la même façon, avec $C = (1+m'')2^{e_3}$ et $fl(C) = (1+m''\pm\varepsilon)2^{e_3}$

$fl(A)\times fl(B)\times fl(C) = A\times B\times C + [\pm(1+m)\pm(1+m')\pm(1+m'')]\varepsilon + (2\varepsilon^2 + \beta\varepsilon^3)\times 2^{e_1+e_2}$

au pire $\approx 6 = 3\times 2$; $O(\varepsilon)$: négligeable.

$\Rightarrow fl(A\times B\times C) \approx A\times B\times C + [\pm(1+m)\pm(1+m')\pm(1+m'')]\varepsilon\times 2^{e_1+e_2}$

au pire $\approx 6 = 3\times 2$.

Soit M le produit de 10^6 facteurs. Les erreurs d'ordre $\sim \varepsilon^n$ sont négligées. Seuls subsistent l'erreur relative $\sim \varepsilon$.

$fl(M) < M + 10^6\times 2\times\varepsilon\times 2^n \Rightarrow$ erreur relative $\sim 2\times 10^6\times\varepsilon \sim \underline{4\times 10^{-10}}$

⚠ Les erreurs s'accumulent.