Algorithme de CORDIC

1 Introduction

L'algorithme CORDIC (*COordinate Rotation DIgital Computing*), inventé en 1959 pour calculer les valeurs des fonctions trigonométriques ou logarithme, a permis aux premiers calculateurs de poche (HP 35 en 1972) d'être rapides malgré la taille réduite de leur mémoire. C'est en effet un algorithme économe en calculs. Nous allons le décrire pour le calcul du logarithme népérien d'un nombre décimal x compris entre 1 et 10.

2 L'algorithme

- On prend habituellement pour une calculatrice $n = 10$. Le programme utilise le logarithme de certains nombres $\ln 10$, ln2 et $\ln(1+10^{-i})$ pour $i \in [\![0, n]\!]$ soit : $\ln 10,\ \ln 2,\ \ln(1,1),\ \ln(1,01),\ \ln(1,001),\ \ldots$

- Les $n+1$ valeurs des logarithmes sont calculées d'une autre façon et sont stockées dans la base de données de la calculatrice.

- Si $n = 10$ la précision de la valeur approchée par excès y de $\ln x$ est de 10^{-9}. Les valeurs stockées sont des approximation à 10^{-12}.

z	$\ln z$
10	$a = 2{,}302\,585\,092\,994$
2	$a_0 = 0{,}693\,147\,180\,560$
1,1	$a_1 = 0{,}095\,310\,179\,804$
1,01	$a_2 = 0{,}009\,950\,330\,853$
1,001	$a_3 = 0{,}000\,999\,500\,333$
1,000 1	$a_4 = 0{,}000\,099\,995\,000$
1,000 01	$a_5 = 0{,}000\,009\,999\,950$
1,000 001	$a_6 = 0{,}000\,000\,999\,999$
1,000 000 1	$a_7 = 0{,}000\,000\,100\,000$
1,000 000 01	$a_8 = 0{,}000\,000\,010\,000$
1,000 000 001	$a_9 = 0{,}000\,000\,001\,000$
1,000 000 000 1	$a_{10} = 0{,}000\,000\,000\,100$

```
from math import*
def cordic(n,x):
    y=log(10)
    for i in range(n+1):
        z=1+10**-i
        while x*z<=10:
            x=x*z
            y=y-log(z)
    return y
```

On peut remarquer que les seuls produits à faire sont des produits par $(1+10^{-i})$ ce qui revient à faire un décalage de virgule puis une addition, qui sont deux opérations très simples. Les seules autres opérations à effectuer sont des soustractions pour trouver y. Ceci explique que cet algorithme est particulièrement efficace et rapide.

On teste le programme :

x	4,5	5,6
y	1,504 077 397	1,722 766 598

Remarque : On peut remarquer que si l'on tape directement sur la calculatrice $\ln 4,5$ et $\ln 5,6$, on obtient exactement les mêmes résultats au dernier chiffre près.

3 Explication

Soit x un réel vérifiant $1 \leqslant x \leqslant 10 \overset{\uparrow(-1)}{\Leftrightarrow} \frac{1}{10} \leqslant \frac{1}{x} \leqslant 1 \overset{\times 10}{\Leftrightarrow} 1 \leqslant \frac{10}{x} \leqslant 10$

1) Comme $2^0 = 1$ et $2^4 = 16 \geqslant 10$,
il existe un entier α_0 avec $0 \leqslant \alpha_0 \leqslant 3$ tel que : $2^{\alpha_0} \leqslant \frac{10}{x} < 2^{\alpha_0+1}$

2) On divise l'encadrement par 2^{α_0} : $1 \leqslant \frac{10}{2^{\alpha_0} \times x} \leqslant 2$

Comme $1,1^0 = 1$ et $1,1^8 \approx 2,14 \geqslant 2$,
il existe un entier α_1 avec $0 \leqslant \alpha_1 \leqslant 8$ tel que : $1,1^{\alpha_1} \leqslant \frac{10}{2^{\alpha_0} \times x} < 1,1^{\alpha_1+1}$

$\overset{\times 2^{\alpha_0}}{\Leftrightarrow} 2^{\alpha_0} \times 1,1^{\alpha_1} \leqslant \frac{10}{x} < 2^{\alpha_0} \times 1,1^{\alpha_1+1}$

3) On montre par récurrence que : $\forall n \in \mathbb{N},\ \exists\, \alpha_i \in [\![0,10]\!],\ i \in [\![0,n]\!]$

$$2^{\alpha_0} \times 1,1^{\alpha_1} \times \cdots \times (1+10^{-n})^{\alpha_n} \leqslant \frac{10}{x} < 2^{\alpha_0} \times 1,1^{\alpha_1} \times \cdots \times (1+10^{-n})^{\alpha_n+1}$$

Remarque :

- Cette récurrence se montre en généralisant la méthode aux items 1) et 2).
- D'après l'inégalité de Bernoulli : $(1+10^{-n})^{10} > 1+10^{-(n-1)}$

4) On pose : $y = 2^{\alpha_0} \times 1,1^{\alpha_1} \times 1,01^{\alpha_2} \times \cdots \times (1+10^{-n})^{\alpha_n}$

D'après les propriétés algébriques de la fonction ln :

$$\ln y = \alpha_0 \ln(2) + \alpha_1 \ln(1,1) + \cdots + \alpha_n \ln(1+10^{-n})$$

De la relation de récurrence, on a :

$y \leqslant \frac{10}{x} < y(1+10^{-n}) \overset{\ln\nearrow}{\Leftrightarrow} \ln y \leqslant \ln 10 - \ln x < \ln y + \ln(1+10^{-n})$

De la relation $\ln(1+x) < x$, on a :

$$\ln y \leqslant \ln 10 - \ln x < \ln y + 10^{-n}$$
$$\overset{-\ln 10}{\Leftrightarrow} \ln y - \ln 10 \leqslant -\ln x < \ln y - \ln 10 + 10^{-n}$$
$$\overset{\times(-1)}{\Leftrightarrow} \ln 10 - \ln y - 10^{-n} < \ln x \leqslant \ln 10 - \ln y$$

Donc $\ln 10 - \ln y = \underbrace{\ln(10) - [\alpha_0 \ln 2 + \alpha_1 \ln 1,1 + \cdots + \alpha_n \ln(1+10^{-n})]}_{\text{valeur approchée de } \ln x \text{ à } 10^{-n}}$

5) On prend $n = 10$.
Les valeurs a et a_i du tableau étant données à 10^{-12} près, on a :

$$\ln 10 - \ln y \approx a - [\alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_{10} a_{10}]$$

Comme les coefficients a_i sont donnés à 10^{-12} près et que les coefficients α_i sont inférieurs à 10, alors les termes $\alpha_i a_i$ sont donnés à $10 \times 10^{-12} = 10^{-11}$ près.

Il y a 12 termes (11 $\alpha_i a_i$ et a), soit une précision de $12 \times 10^{-11} \approx 10^{-10}$.

$a - [\alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_{10} a_{10}]$ donne la valeur de $\ln x$ à 2×10^{-10}

6) Dans l'algorithme :
   - on prend les coefficient a_i les uns après les autres dans la boucle itérative ;
   - on soustrait α_i fois a_i à y dans la boucle conditionnelle.

4 Généralisation à un nombre quelconque

1) Soit x un nombre réel de la calculatrice, donc $10^{-100} < x < 10^{100}$.

   On encadre x entre deux puissance de 10 consécutives :
   - négatives si $x < 1$
   - positives si $x > 10$.

   Pour $n \in [\![-100\ ,\ 100]\!]$, on a : $10^n \leqslant x < 10^{n+1} \overset{\times 10^{-n}}{\Leftrightarrow} 1 \leqslant 10^{-n}x < 10$.

   Comme $1 \leqslant 10^{-n}x < 10$, on est revenu au calcul précédent, c'est à dire :

$$\begin{aligned} \ln(10^{-n}x) &\approx a - [\alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_{10} a_{10}] \\ -n\ln 10 + \ln x &\approx a - [\alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_{10} a_{10}] \\ \ln x &\approx n\ln 10 + a - [\alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_{10} a_{10}] \\ \ln x &\approx na + a - [\alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_{10} a_{10}] \end{aligned}$$

   - Comme a est donné à 10^{-12} près, $|na| < 100a$ est donné à 10^{-10} près.
   - Comme le calcul précédent était une approximation à 2×10^{-10} près
   - $\ln x$ est donc donné à 3×10^{-10} près.

2) Pour écrire un nouveau programme qui donne une approximation de $\ln x$ d'un réel quelconque de la calculatrice, on distinguera deux cas : $x < 1$ et $x \geqslant 10$.
   - Si $x > 10$, on divise par 10 tant que $x > 10$ en augmentant m de 1 à chaque boucle afin d'avoir $10^m \leqslant x < 10^{m+1}$.
   - Si $x < 1$, on multiplie par 10 tant que $x < 1$ en soustrayant m de 1 à chaque boucle afin d'avoir $10^m \leqslant x < 10^{m+1}$.

   On teste la nouvelle fonction cordic :

x	4 567	0,001 467
y	8,426 611 813	-6,524 535 78

Remarque : On peut remarquer que si l'on tape directement sur la calculatrice $\ln 4\,567$ et $\ln 0{,}001\,467$, on obtient exactement les mêmes résultats.

```
from math import*
def cordic(n,x):
    y=log(10)
    m=0
    while x>10:
        x=x/10
        m+=1
    while x<1
        x=10*x
        m-=1
    for i in range(n+1):
        z=1+10**-i
        while x*z<=10:
            x=x*z
            y=y-log(z)
    return m*log(10)+y
```