Calcul de fonctions mathématiques

I. Calcul de l'exponentielle

Tout réel x strictement positif peut s'écrire[1] sous la forme :

$$x = \sum_{k=-\infty}^{n} a_k 2^k$$

tel que

- ⋆ pour tout k, $a_k \in \{0, 1\}$,
- ⋆ $a_n \neq 0$ et
- ⋆ la suite n'est pas constante égale à 1 à partir d'un certain rang.

C'est son écriture binaire. En utilisant la propriété fondamentale de l'exponentielle, on a alors :

$$e^x = \prod_{a_k \neq 0} e^{2^k}$$

Ainsi, pour calculer l'exponentielle d'un flottant, si l'on a calculé à l'avance et stocké les valeurs des termes de la forme e^{2^k}, le calcul d'une exponentielle se ramène à un produit.

Nous allons utiliser cette propriété pour avoir un algorithme permettant d'avoir l'exponentielle d'un réel positif en temps constant.

Dans un premier temps, nous allons utiliser la fonction `exp` de la bibliothèque `math` — qu'il faudra importer — pour déterminer les e^{2^k}, pour nous concentrer sur l'algorithme permettant de calculer l'exponentielle à partir de ces valeurs.. Par la suite, nous les calculerons directement.

Tout d'abord, notons que la représentation d'un flottant en mémoire ne permet pas de manipuler des nombres arbitrairement grands, ou arbitrairement proches de 1. En particulier, dans l'équation (**??**), les valeurs des indices k à prendre en compte seront bornées, nous ramenant à un produit fini.

Question 1 Pour répondre aux questions suivantes, vous pourrez utiliser un programme Python.

1. Quel est le plus grand entier k tel que le calcul de e^{2^k} ne provoque pas d'erreur ?
2. Quel est le plus grand entier k tel que la valeur calculée de $e^{2^{-k}}$ est différente de 1 ?
3. En déduire l'intervalle $[\![a, b]\!]$ des puissances de 2 « utiles » pour calculer une exponentielle en flottant.

Question 2 Écrire une fonction `decomposition` qui, étant donné un flottant x, renvoie la liste des puissances de deux apparaissant dans la décomposition de x. Cette décomposition s'obtient à l'aide d'un algorithme glouton : en considérant les puissances par ordre décroissant, 2^k intervient dans la décomposition de x si et seulement si $x \geq 2^k$, et dans ce cas il reste à décomposer $x - 2^k$.

Par exemple, on veut :

```
>>> decomposition(5.75)
[2, 0, -1, -2]
>>> decomposition(pi)
[1, 0, -3, -6, -11, -12, -13, -14, -15, -16, -18, -19, -21,
 -23, -25, -29, -33, -38, -40, -41, -43, -47, -48]
```

Normalement, la fonction termine toujours[2].

Question 3 En précalculant les valeurs de e^{2^k} pouvant servir (on pourra les stocker dans une liste), écrire une fonction `expo` qui calcule l'exponentielle d'un flottant donné en argument en temps constant (et sans utiliser `exp`, bien sûr, mais nous ne savez pas si elle est en temps constant). Vous pourrez utiliser ou adapter la fonction `decomposition`.

Vous prendez soin de tester la fonction. Il est possible que les résultats obtenus diffèrent de la fonction `exp` de la bibliothèque `math`, mais la différence doit être minime :

```
>>> exp(pi) # version de la bibliothèque math
23.140692632779267
>>> expo(pi) # version programmée "à la main"
23.14069263277926
```

Il reste à calculer les valeurs des e^{2^k}. Pour cela, on va utiliser le développement usuel:

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \cdots + \frac{x^n}{n!} + \cdots = \lim_{n\to\infty} \sum_{k=0}^{n} \frac{x^k}{k!}$$

En pratique, quelle que soit la valeur de x, pour n assez grand, le terme $x^n/n!$ sera nul, donc notre somme en apparence infinie sera approximée par une somme

[1] de façon unique, mais cela n'est pas intéressant ici

[2] Sauriez-vous dire pourquoi ?

finie.

Question 4 Écrire la fonction `expo_dl` qui calcule une exponentielle à l'aide de la formule précédente. On pourra utiliser une boucle `while` en s'arrêtant lorsque le nouveau terme à ajouter est nul.

À titre, d'illustration, on pourra renvoyer, en plus de l'exponentielle calculée, l'ordre maximal utilisé dans le calcul. Et, bien sûr, on comparera la fonction écrite aux valeurs d'exponentielle calculée à l'aide de la fonction de la bibliothèque. Par exemple,

```
>>> expo_dl(pi)
(23.140692632779274, 29)
>>> exp(pi)
23.140692632779267
>>> expo_dl(100)
(2.6881171418161356e+43, 194)
>>> exp(100)
2.6881171418161356e+43
```

II. Calcul du logarithmes, approches historiques

Nous allons maintenant nous intéresser à diverses méthodes de calculs utilisées à travers l'histoire pour calculer des logarithmes, l'utilisation de l'informatique nous permettant de les implémenter facilement et de les comparer.

On définira la fonction suivante :

```
def dist(x, y):
    return -log10(abs(x - y))
```

qui permet d'avoir une estimation du nombre de chiffres décimaux significatifs communs entre `x` et `y` :

```
>>> dist(pi, 355 / 113)
6.573872470811142
>>> pi
3.141592653589793
>>> 355 / 113
3.1415929203539825
```

1. Méthode de Briggs

Au début du XVIIème siècle, Henry Briggs publia une table de logarithmes. Ayant remarqué qu'une fonction logarithmique log (quelqu'en soit la base) vérifie les deux propriétés suivantes :

- ⋆ $\log\sqrt{a} = \frac{1}{2}\log a$,
- ⋆ si a est très proche de 0, alors $\log(1+a)$ est proportionnel à a (le coefficient dépendant de la base du logarithme),

il procéda ainsi :

1. on pose $u_0 = 2$ et $v_0 = 10$;
2. on en calcule les racines carrées successives, soit en écriture moderne :
$$\forall\, n \in \mathbf{N},\ u_{n+1} = \sqrt{u_n}\ \ \textbf{et}\ \ v_{n+1} = \sqrt{v_n}\ ;$$
3. le rapport $\dfrac{u_n - 1}{v_n - 1}$ converge alors vers $\dfrac{\ln 2}{\ln 10} = \log_{10} 2$.

Petite explication mathématique

On a $u_n = 2^{\frac{1}{2^n}} \longrightarrow 1$, donc
$$\frac{1}{2^n}\ln 2 = \ln u_n \underset{n\to\infty}{\sim} u_n - 1$$
et de même, $v_n - 1 \underset{n\to\infty}{\sim} \frac{1}{2^n}\ln 10$. On en déduit que l'on a bien
$$\frac{u_n-1}{v_n-1} \xrightarrow[n\to\infty]{} \frac{\ln 2}{\ln 10}.$$

Ainsi, pour calculer $\log_{10}(2)$, Henry Briggs itérera le processus 54 fois (oui oui, il a enchaîné 54 calculs de racine carrée à la main).

Question 5

1. Écrire une fonction `briggs()` qui calcule les 54 premiers termes des suites (u_n) et (v_n) et affiche la distance entre $\log_{10}(2)$ (calculé à l'aide des fonctions standards de Python) et le quotient $\frac{u_n-1}{v_n-1}$.
2. Quel phénomène étonnant observez-vous ? Pouvez-vous l'expliquer ?

2. L'aire sous une courbe

Les calculs précédents faisant intervenir les logarithmes en base 10. Cependant, on découvrit un peu plus tard que l'aire sous la courbe de la fonction $x \mapsto \frac{1}{x}$ est aussi une fonction logarithme, que l'on appelle maintenant logarithme népérien. Cette fonction logarithmique est la plus pertinente mathématiquement.

Nous allons programmer une méthode pour calculer $\ln 2$ de façon approchée à l'aide de l'intégrale de la fonction inverse entre 1 et 2, sachant que :

$$\ln(2) = \int_1^2 \frac{1}{t}\,\mathrm{d}t$$

Question 6

1. Écrire une fonction `rectangle` qui, étant donné un entier n, retourne l'intégrale calculée à l'aide de la méthode des rectangles avec n subdivisions, soit dans notre cas :
$$\frac{1}{n}\sum_{k=0}^{n-1} f\left(1+\frac{k}{n}\right)$$
avec pour f la fonction inverse.
2. Faire apparaître expérimentalement qu'avec cette méthode, on gagne un chiffre significatif en multipliant le nombre de subdivisions par 10.

Une amélioration de la méthode des rectangles est celle dite « des trapèzes » où l'on calcule une somme d'aires de trapèzes[3], ce qui correspond à la formule :

$$\frac{1}{n}\sum_{k=0}^{n-1}\frac{1}{2}\left(f\left(1+\frac{k}{n}\right)+f\left(1+\frac{k+1}{n}\right)\right)$$

Question 7

1. Écrire une fonction `trapèze` qui, étant donné un entier n, calcule l'intégrale précédente avec la méthode des trapèzes pour n subdivisions.
2. Comparer la vitesse de convergence à la celle de la méthode des rectangles.

Deux remarques s'imposent :

- ⋆ Difficile d'avoir plus d'une dizaine de décimales, alors que 52 bits de mantisse correspondent plutôt à une quinzaine.
- ⋆ Le nombre d'opérations pour obtenir une précision décente reste vraiment très élevé.

3. Calculs en séries

Les progrés mathématiques du XVIIème vont permettre de développer de nouvelles techniques bien plus efficaces, notamment à l'aide des séries numériques.

[3]Voir figure au tableau.

3.1. Série de Mercator

En 1668, dans son traîté « Logarithmotechnia », Nicolas Mercator (connu pour ses projections cartographiques) démontre la formule suivante (écrite de façon moderne) :

$$\forall x \in\,]-1,1],\ \ln(1+x) = \sum_{n=1}^{\infty}\frac{(-1)^{n+1}x^n}{n}$$

Avec des points de suspension, cela donne :

$$\ln(1+x) = x - \frac{x^2}{2}+\frac{x^3}{3}-\frac{x^4}{4}+\frac{x^5}{5}-\frac{x^6}{6}+\cdots$$

Pour $x=1$, on reconnaît la série harmonique alternée :

$$\ln 2 = 1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\frac{1}{5}-\frac{1}{6}+\cdots = \sum_{n=0}^{\infty}\frac{(-1)^n}{n+1}$$

Question 8 Écrire une fonction `mercator1` qui prends en argument un entier `n`et retourne la somme partielle précédente évaluée jusqu'à ce rang n.

Question 9 Évaluez son (in-)efficacité en calculant, pour plusieurs valeurs de n, la distance entre la valeur obtenue et $\ln(2)$.

On peut être un peu déçu par cette méthode de calcul. Mais la formule précédente s'applique aussi à d'autres valeurs de x. Pour le calcul de $\ln 2$, on peut remarquer que l'on a :

$$\ln 2 = -\ln\frac{1}{2} = -\ln\left(1-\frac{1}{2}\right).$$

Question 10

1. Écrire une fonction `mercator2` basée sur le calcul de $-\ln(1-\frac{1}{2})$.
2. Évaluez l'efficacité de la fonction obtenue.

3.2. Série de Gregory

Une amélioration simple pour l'utilisation de la série de Mercator est dûe à James Gregory, qui utilise intelligemment les propriétés du logarithme. Il soutrait les séries correspondant à $\ln(1+x)$ à $\ln(1-x)$, pour obtenir :

$$\forall x\in\,]-1,1[,\ \ln\left(\frac{1+x}{1-x}\right) = 2x+2\frac{x^3}{3}+2\frac{x^5}{5}+2\frac{x^7}{7}+\cdots = 2\sum_{n=0}^{\infty}\frac{x^{2n+1}}{2n+1}$$

Question 11 Pour quelle valeur de x peut-on calculer $\ln 2$?

Question 12 Écrire une fonction `gregory` correspondant à cette méthode, et visualiser son efficacité.

III. Encore plus de questions

Question 13 Pouvez-vous utilisez la fonction `expo` précédente pour en déduire une fonction `loga` ?

Pour finir, on peut adapter la méthode de calcul de l'exponentielle aux sinus et cosinus, les relations d'addition à utiliser étant bien sûr :

$$\cos(a+b) = \cos a \cos b - \sin a \sin b \qquad \sin(a+b) = \sin a \cos b + \cos a \sin b$$

Question 14 Essayer d'en déduire des fonctions calculant en temps constant les sinus et cosinus d'un angle. Pour obtenir les valeurs calculées à l'avance, on pourra bien sûr utiliser leurs développements en série :

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \cdots + (-1)^n \frac{x^{2n}}{(2n)!} + \cdots$$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + \cdots$$