Travaux dirigés
Exponentiel n°12
Mathématiques pour l'informatique
—IMAC 2—

Exponentiel

La fonction exponentielle est massivement utilisée en mathématiques, en physique, économie, etc. Elle présente des propriétés tout à fait extraordinaires comme changer des produits en sommes, ou bien comme exprimer sa propre dérivée. Cette fonction est également connue pour sa croissance extrêmement rapide dans pour les nombres positifs et très lente pour les nombres négatifs. Elle elle s'inscrit donc dans un registre mêlant des nombres très proche de zéro ainsi que des nombres très grands, ce qui pose un challenge du côté de son traitement numérique.

▸ **Exercice 1.** ***Code***

1. *Récuperez le code sur le site de l'enseignant.*
2. *Lisez le et testez le.*

▸ **Exercice 2.** ***Fonction exponentielle et développement de Taylor***

Le développement de Taylor d'une fonction $f(x)$ autour de $x = a$ s'écrit :

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

où $f^{(n)}(x)$ correspond à la dérivée n-ième de $f(x)$. Autour de $a = 0$, le développement de Taylor de la fonction exponentielle se simplifie par :

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

1. *Écrire le pseudo-code de la fonction calculant exponentiel avec la formule de Taylor simplifiée.*
2. *Quelle est la complexité de cette méthode?*
3. *Codez une fonction calculant exponentiel de x avec le prototype suivant :*

```
template<typename T>
T taylor(T x)
```

Cette fonction implémentera le développement de Taylor décrit plus haut, sur 12 itérations pour des `float` *et sur 21 itérations pour des* `double` *(recommandé). Pour quoi il ne vaut mieux pas faire plus d'itérations?*

4. *Testez votre fonction sur des nombres $x \in [-2,2]$. Comparez votre résultat avec la fonction exponentielle de* `cmath`.
5. *Essayer avec des nombres $x \in [-10,10]$, que constatez-vous?*
6. *Sur des nombres encore plus grands, on peut constater que le développement de Taylor est plus précis sur les $x > 0$. Modifier votre code de telle sorte que votre programme calcule $y = e^{|x|}$, puis renvoie y si $x \geq 0$ et $1/y$ si $x < 0$.*

▸ **Exercice 3.** ***Fonction exponentielle et méthode de Horner***

Le developpement de Taylor de la fonction exponentielle fait intervenir la fonction factorielle qui est connue comme étant une source d'instabilité numérique (elle croit trop rapidement). Une alternative consiste à utiliser la méthode de Horner qui fait disparaitre cette fonction factorielle :

$$\begin{aligned} e^x &= \sum_{n=0}^{\infty} \frac{x^n}{n!} \\ &= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots \\ &= 1 + x\left(1 + \frac{x}{2}\left(1 + \frac{x}{3}\left(1 + \frac{x}{4}(\cdots)\right)\right)\right) \\ &\simeq 1 + x\left(1 + \frac{x}{2}\left(1 + \frac{x}{3}\left(1 + \frac{x}{4}\right)\right)\right) \end{aligned}$$

1. *Donnez le pseudo-code permettant de calculer e^x (toujours sur les $x \geq 0$), selon la méthode de Horner.*
2. *Codez cette fonction, en utilisant 17 itérations, avec le prototype suivant :*

```
template<typename T>
T horner(const T &x)
```

3. *Comparez votre résultat avec la fonction exponentielle de* `cmath` *et avec votre fonction* `taylor`.
4. *Comme pour la fonction* `taylor`, *la fonction* `horner` *est instable numériquement pour les x "grands". Une solution consiste à opérer une réduction d'intervalle sur l'évaluation de e^x. Il s'agit d'exprimer $e^x = e^r.2^k$, en choisissant judicieusement r et k. Il se trouve que 2^k est très simple à calculer (*`1 << k`*) et que r sera éventuellement très inférieur à x. Un bon choix de r et k peut se faire avec :*

$$\begin{aligned} k &= \left\lfloor \frac{x}{\log(2)} - \frac{\log(2)}{2} \right\rfloor + 1 \qquad \text{où } \lfloor x \rfloor \text{ est la partie entière de } x \\ r &= x - k\log(2) \end{aligned}$$

Donner le pseudo-code de cette méthode.

5. *Codez cette méthode avec le prototype suivant :*

```
template<typename T>
T reduced_horner(const T &x)
```

   À savoir, $\log(2)$ *est une constante précalculée dans* `cmath` *par la variable* `M_LN2`*.*

6. *Comparez avec les autres méthodes.*

▸ **Exercice 4.** ***Exponentielle rapide***

Il existe une méthode d'estimation rapide d'exponentielle de x *sur des* `double` *IEEE-754 (64 bits) dont les détails sont présentés sur le papier A Fast, Compact Approximation of the Exponential Function (1999) et dont voici le code :*

```
double exp_fast(const double &x){
    long int y = ((long int) (1512775 * x + 1072632447)) << 32;
    return * ( double * ) & (y);
}
```

Il s'agit d'une astuce exploitant le fait que l'exposant d'un nombre flottant s'exprime comme une puissance de 2, il suffit alors d'intégrer un $\log(2)$ *pour transformer cette puissance de 2 en exponentielle approximée.*

Comparez les résultats obtenus avec la solution exacte.