TP 2 - Intégration

Dans ce TP on veut calculer des intégrales

$$I := \int_a^b f(x)\mathrm{d}x$$

par approximations numériques. On supposera que f est continue. Avant de commencer :

— pour savoir comment coder correctement avec des commentaires, des options verbose et debug, des tests, faire un code qui calcule une approximation de $\ln(1+x)$ avec série de Taylor et regarder le corrigé `approx_ln.py`

— il y a une fonction scipy permettant d'obtenir des intégrales : `scipy.integrate.quad(f,a,b)`, donnant $\int_a^b f$. Elle retourne la valeur et la précision à laquelle elle a fait le calcul. Tester cette fonction sur une intégrale connue, comme par exemple $\int_0^1 x\mathrm{d}x$. Pour cela, écrire

```python
import scipy as sp

def test_scipy_integrate():
    def g(x):
        return x
    I, precision = sp.integrate.quad(g,0,1)
    print("Scipy integration is",I,", should be 1/2, prec",precision)

test_scipy_integrate()
```

S'il y a l'erreur "no module named 'scipy"', c'est que scipy n'est peut-être pas installé, l'installer avec `pip3 install scipy` ou `pip install scipy` selon si on utilise python ou python3.

Exercice 1 : Méthode des trapèzes

Soit la formule

$$Q_N := \sum_{i=0}^{N-1} \omega_i f(x_i)$$

pour approcher I. Ceci définit une classe d'approximations de I, appelées formules de quadrature. Chaque approximation étant un choix de manière de calculer ω_i.

Étant donné un entier naturel non nul N et une liste de N nombres $x_i \in [a,b]$, pour $i \in \{0,\dots,N-1\}$, la méthode des trapèzes consiste à calculer

$$T_N = \sum_{i=0}^{N-1} (x_{i+1} - x_i)\frac{f(x_i) + f(x_{i+1})}{2}.$$

On choisira une liste de x_i's régulièrement espacés, c'est-à-dire $x_{i+1} - x_i = \frac{b-a}{N}$, et $x_0 = a$.

(a) Montrer théoriquement que $T_N \underset{N\to+\infty}{\longrightarrow} \int_a^b f$.

(b) Programmer une fonction `trapeze(f,a,b,N)` qui calcule l'approximation de l'intégrale de f avec cette méthode.

(c) Faire une fonction de test `test_trapeze` de la fonction `trapeze` sur

$$I = \int_0^3 \sin(x + e^x)\mathrm{d}x, \tag{1}$$

où la valeur de référence sera calculée en utilisant la fonction `quad` de scipy qui donne des intégrales. On prendra $N = 10^4$ pour ce test. Quelle est l'erreur faite avec la valeur exacte ?

(d) On veut estimer l'erreur commise dans l'intégration de la fonction précédente par la méthode des trapèzes pour différentes valeurs de l'entier N. En guise de valeur de référence pour l'intégrale I, on utilisera encore la fonction de scipy. Regarder le comportement de `np.logspace` en manipulant par exemple `np.logspace(2, 4, num=50, endpoint=True, base=10.0)` et en faisant varier les paramètres de cette fonction. Illustrer graphiquement (par un tracé en échelle logarithmique, via `np.logspace` (il faudra transformer les valeurs de la liste retournée en prenant leur partie entière car N ne peut prendre que des valeurs entières)) la majoration suivante :

$$\exists C \leq 0,\ \forall N \in \mathbb{N}^*,\ \ |T_N - I| \leq \frac{C}{N^2}.$$

Donner la valeur numérique du C minimum qui vérifie cette inégalité.

Exercice 2 : Méthode de Monte-Carlo

La méthode de Monte-Carlo est une approche probabiliste permettant d'approximer la valeur d'une intégrale. L'idée de base est que l'intégrale peut être vue comme l'espérance d'une variable aléatoire uniforme X sur l'intervalle $[a, b]$

$$\int_a^b f(x)\,\mathrm{d}x = (b - a)E(f(X)).$$

Par la loi des grands nombres cette intégrale peut être approchée par la moyenne empirique

$$M_N := \frac{b-a}{N}\sum_{i=0}^{N-1} f(x_i)$$

où les x_i sont tirés aléatoirement dans l'intervalle $[a, b]$ avec une loi de probabilité uniforme. On a alors

$$\int_a^b f(x)\,dx \simeq M_N.$$

En python, la fonction `numpy.random.random()` permet de générer un nombre aléatoirement uniformément dans le segment $[0, 1[$. On pourra aussi utiliser `numpy.random.sample(N)`.

(a) Écrire une fonction `montecarlo(f,a,b,N)` qui détermine une approximation de l'intégrale par la méthode de Monte-Carlo. Tester la fonction sur l'intégrale (1) pour vérifier qu'il n'y a pas de bug.

(b) Modifier la fonction pour qu'elle retourne en plus de la moyenne également la variance empirique

$$V_N = \frac{(b-a)^2}{N} \sum_{i=0}^{N-1} \left(f(x_i) - \frac{M_N}{b-a} \right)^2 .$$

(c) Calculer M_∞ et V_∞ qui sont définis comme étant M_N et V_N pour N très grand, de telle manière à ce que M_∞ et V_∞ approximent correctement les valeurs exactes. Pour avoir M_∞, il serait également possible d'utiliser la fonction `scipy.integrate.quad(f,a,b)`. Étudier empiriquement la convergence en fonction de N en calculant M_N et V_N pour des valeurs de N de 1 à 10^5 en pas logarithmique, et faire un graphique. On a en fait

$$\sqrt{N}|M_N - M_\infty| \underset{N\to+\infty}{\to} c$$

pour un certain $c \in \mathbb{R}$. Tracer ce comportement sur un graph.

Exercice 3 : Méthode de Simpson

La méthode de Simpson consiste à approximer f sur chaque intervalle $[x_i, x_{i+1}]$ par un polynôme de degré deux. Le choix le plus naturel est le polynôme P_i de degré deux passant par les points

$$(x_i, f(x_i)), \qquad (m_i, f(m_i)), \qquad (x_{i+1}, f(x_{i+1})), \tag{2}$$

où $m_i := \frac{1}{2}(x_i + x_{i+1})$.

(a) On se donne $A, B, C \in \mathbb{R}$, non égaux. On rappelle le polynôme de Lagrange d'ordre 2, $L_{a,b,c}(x) := \frac{(x-c)(x-b)}{(a-c)(a-b)}$ est égal à 1 en a, et 0 en b et c. Déterminer le polynôme P_i qui passe par les trois points (2).

(b) Calculer les $\gamma_i := \int_{x_i}^{x_{i+1}} P_i(x)\mathrm{d}x$ à la main, et en faire une fonction `gamma(f,x,i)` qui prend aussi en argument x la liste des x_i.

(c) On peut donc approximer $F_N \simeq f$ avec $F_N := \sum_{i=0}^{N-1} P_i \chi_i$ où $\chi_i(x) = 1$ si $x \in [x_i, x_{i+1}]$ et $\chi_i(x) = 0$ sinon. Ceci induit donc l'approximation de Simpson

$$S_N := \int_a^b F_N(x)\mathrm{d}x = \sum_{i=0}^{N-1} \gamma_i.$$

Écrire une fonction `simpson(f,a,b,N)` qui calcule l'approximation de Simpson.

(d) Comparer les vitesses de convergence de l'approximation des trapèzes, de Monte-Carlo, et de Simpson en fonction de N, quelle est la meilleure méthode ? Faire un graphique en échelle logarithmique qui permet de visualiser. Ajouter des légendes.