INFORMATIQUE PSI P2B

Année 2023/24

TP n° 3 (noté)

Les fichiers, au format `.py`, seront nommés `NOM_DE_FAMILLE-TP3.py` et seront envoyés 5 min avant la fin de l'heure à tpinforohart@gmail.com

Le sujet comporte trois exercices.

Vincent Rohart
`http://vrohart.e-monsite.com/`

Exercice 1 — Petite énigme

1. Écrire un script Python qui résout l'énigme ci-dessus.
2. Programmer une fonction de signature `enigme(n)` qui renvoie la liste des entiers de n chiffres (dans leur écriture décimale) dont l'ordre s'inverse quand on les multiplie par 9.
3. Tracer un graphique représentant les points `(n, enigme(n))` pour n allant de 1 à 10.

Exercice 2 — Programmation orientée objet

On rappelle qu'un polynôme formel *(ou simplement polynôme) est une suite* $(a_n)_{n\in\mathbb{N}}$ *de scalaires nuls à partir d'un certain rang. Il faut distinguer* polynôme *et* fonctions polynômes.

Ainsi, le polynôme $X^3 + 2X - 5$ pourra être formalisé en Python par la liste `[-5,2,0,1]` mais aussi par la liste `[-5,2,0,1,0,0,0]`.

1. Créer une classe `Polynome` dont les objets seront des listes de flottants (des polynômes formels donc). Programmer la méthode `__init__` de telle façon à ce que l'attribut d'un polynôme soit la liste de ses coefficients.
2. Programmer une méthode `deg` qui renvoie le degré d'un polynôme.
3. Surcharger la méthode spéciale `__add__` pour que $P + Q$ désigne la somme de deux polynômes si P et Q en sont. Tester votre méthode en calculant $P + Q$ si $P = 1 + X + 2X^2$ et $Q = -X + X^3$.
4. Créer la classe `FracRat` des fractions rationnelles, celles de la forme $\frac{P}{Q}$ avec P et Q des polynômes ($Q \neq 0$), pour qu'elle hérite des méthodes de la classe `Polynome`. Surcharger alors la méthode `__add__` pour qu'elle soit utilisable dans cette classe.

Exercice 3 — Analyse numérique

Le module `scipy.integrate` et sa méthode `quad` permettent de calculer des intégrles de façon approchée :

https://www.concours-centrale-supelec.fr/CentraleSupelec/SujetsOral/Multi/Python-AN.pdf

1. Utiliser la documentation ci-dessus pour donner une valeur approchée de $\int_0^1 \frac{\mathrm{d}x}{1+x^2}$. Quelle est sa valeur exacte ?

   On admet que pour tout polynôme $\mathrm{P} \in \mathbb{R}_3[\mathrm{X}]$ et tous réels $a < b$,

   $$\frac{1}{b-a}\int_a^b \mathrm{P}(t)\mathrm{d}t = \frac{\mathrm{P}(a)+4\mathrm{P}(c)+\mathrm{P}(b)}{6},$$

   où $c = \frac{a+b}{2}$. C'est *la formule des trois niveaux*. Si $f : [a,b] \to \mathbb{R}$ est une fonction quelconque, on pose alors

   $$\mathrm{I}(f) = (b-a) \times \frac{f(a)+4f(c)+f(b)}{6}.$$

2. Ici $f = x \mapsto \dfrac{1}{1+x^2}$. Calculer, avec Python, l'erreur commise en approchant $\displaystyle\int_0^1 f(x)\mathrm{d}x$ par $\mathrm{I}(f)$.
3. Pour chaque $n \in \mathbb{N}^*$, on décide de découper $[0,1]$ en n parties égales : $[x_k, x_{k+1}]$ avec $x_k = \frac{k}{n}$, $k \in [\![0,n]\!]$ et d'approcher $\displaystyle\int_{x_k}^{x_{k+1}} f(t)\mathrm{d}t$ par $\mathrm{I}(f|_{[x_k,x_{k+1}]})$.

   Créer une fonction de signature `simpson(f,n)` qui prend une fonction f et un entier n et qui renvoie le flottant

   $$\sum_{k=0}^{n-1} \mathrm{I}(f|_{[x_k,x_{k+1}]}).$$

4. Ici encore $f = x \mapsto \dfrac{1}{1+x^2}$. Tracer un graphique représentant la suite $n \mapsto$ `simpson(f,n)`, pour $n \in [\![1,20]\!]$.

5. Comparer cette méthode avec celle du module `scipy.integrate` : quelle est la valeur minimale de n pour atteindre le même niveau de précision que `scipy.integrate`.

   La méthode de Monte-Carlo est une méthode probabiliste pour calculer des intégrales, simples ou multiples, de fonctions de classe quelconque (par exemple seulement continue). En contrepartie, elle converge souvent très lentement.

   Considérons une fonction continue $f : [a,b] \to \mathbb{R}$, supposée positive, et majorée par M. Le nombre $\int_a^b f$ représente l'aire d'un certain domaine D du plan contenu dans le rectangle $[a,b] \times [0,\mathrm{M}]$.

   Si l'on place « au hasard » (de façon uniforme) des points $\mathrm{M}_i(x_i,y_i)$ avec $x_i \in [a,b]$ et $y_i \in [0,\mathrm{M}]$, la probabilité pour que M_i soit dans le domaine D est le rapport de l'aire de D par celle du rectangle $[a,b] \times [0,\mathrm{M}]$, soit

   $$p = \frac{1}{\mathrm{M}\cdot(b-a)}\int_a^b f(t)\mathrm{d}t.$$

   La probabilité p peut quant à elle s'évaluer en comptant le nombre de points M_i qui sont dans D grâce à la relation

   $$\mathrm{M}_i \in \mathrm{D} \iff y_i < f(x_i).$$

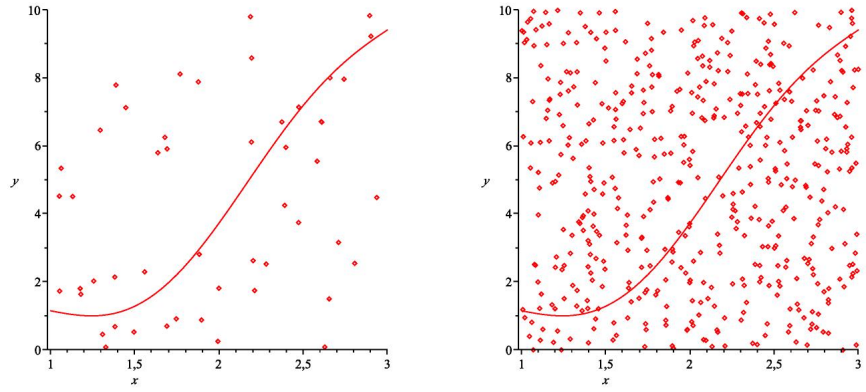

FIGURE 1 – Méthode de MONTE-CARLO pour $f : x \mapsto \sin(3x) + x^2$ sur $[1; 3]$ avec 50, puis 500 points.

6. Implémenter une fonction `mc(f,a,b,M,n)` qui renvoie une approximation de $\int_a^b f(x)\mathrm{d}x$, quand f est positive et majorée par M.
7. Tester la fonction pour le calcul de $\int_0^1 x^2\mathrm{d}x$. Trouver le plus petit n pour que `mc(f,a,b,M,n)` $\approx \int_0^1 x^2\mathrm{d}x$ à 10^{-3} près.
8. Adapter la fonction `mc` pour calculer $\iint_\mathrm{D} f(x, y)\mathrm{d}x\mathrm{d}y$ quand D est un domaine simple de la forme $\{(x, y) \in \mathbb{R}^2 \mid a \leqslant x \leqslant b \wedge \varphi(x) \leqslant y \leqslant \psi(x)\}$.