TP : probabilités
Informatique pour tous

I Génération de p entiers parmi n

On considère l'algorithme suivant, où $p \leq n$:
Initialiser une liste `L` vide
Tant que $p > 0$:
    Avec probabilité $\frac{p}{n}$: ajouter n à `L` et diminuer p de 1.
    Diminuer n de 1.

1. Écrire une fonction implémentant cet algorithme. On utilisera `np.random.random()` qui donne un flottant aléatoire entre 0 et 1.
2. Vérifier et comprendre pourquoi cet algorithme renvoie p entiers uniformément au hasard entre 1 et n.

II Marche aléatoire

On définit un entier X égal à 0, puis on répète l'action suivante 1000 fois :
Avec probabilité $\frac{1}{2}$, augmenter X de 1. Sinon, diminuer X de 1.

1. Écrire une fonction renvoyant le temps nécessaire pour le premier retour de X en 0.
   Si X ne revient jamais en 0 on renverra -1.
2. Répéter l'expérience précédente 1000 fois et donner la proportion de ceux-ci qui sont revenus en 0.
3. Donner le temps moyen de retour en 0, parmi ceux qui sont effectivement revenus en 0.

III Méthode de Monte-Carlo

Nous allons voir une nouvelle méthode pour calculer une aire, basée sur des probabilités.

1. En utilisant les fonctions `sin` et `log` de numpy, définir en Python la fonction
$$f : x \in [-2, 2] \longmapsto \begin{cases} 1 + \sin(x), \text{ si } x < 0 \\ 1 + \log(1 + x), \text{ sinon} \end{cases}$$
2. Écrire `import matplotlib.pyplot as plt` puis afficher le graphe de f avec `plt.plot` puis `plt.show()`.
   On pourra construire une liste `X` de 100 abscisses dans $[-2, 2]$ (en commençant par une liste vide puis en rajoutant les points avec `X.append`).
   De même on construira la liste `Y` des ordonnées correspondantes pour enfin appeler `plt.plot(X, Y)`.
3. Approximer $\displaystyle\int_{-2}^{2} f$ (c'est à dire l'aire sous la courbe) en utilisant la méthode des rectangles et/ou des trapèzes.
4. La méthode de Monte-Carlo consiste à approximer une intégrale I en considérant un rectangle R contenant l'aire à approximer puis en générant des points aléatoirement dans R. L'aire de R multiplié par la proportion de points sous la courbe donne alors une approximation de I.
   Approximer $\displaystyle\int_{-2}^{2} f$ en utilisant cette méthode. On utilisera `np.random.random()` qui donne un flottant aléatoire entre 0 et 1
   Vérifier que l'approximation est proche de celle du 3., pour 1000 points aléatoires.