Exercices sur les probabilités

Exercice 1. Génération d'une loi de Poisson (*)

Écrire, en n'utilisant que `random.random()` qui génère un nombre aléatoire dans [0 ;1[une fonction `Poisson(l)` qui génère une loi de Poisson de paramètre l.

On pourra envisager d'inverser la fonction de répartition.

Exercice 2. Marche au hasard en 1d (**)

On se place sur la droite réelle, et on s'intéresse au sous-ensemble $\mathbb{Z}$. À l'instant initial t=0, un observateur est situé à l'origine. À chaque pas de temps, il avance, au hasard et de manière équiprobable d'une unité vers la gauche ou vers la droite.

1. Écrire une fonction `listepositions(n)` qui prend en argument un entier n positif, et qui renvoie une liste des positions successives que parcourt l'observateur qui marche au hasard.

2. Écrire une fonction `observateur(k,n)` qui prend en arguments deux entiers k et n strictement positifs et qui renvoie la liste des positions qu'occuperaient k observateurs marchant au hasard après n pas de temps. Quelle est la complexité de la fonction écrite ?

3. Représenter graphiquement la répartition de 10000 observateurs après 100 pas de temps.

4. Comment semble évoluer la probabilité qu'un observateur repasse par l'origine après un grand nombre de pas de temps ?

Exercice 3. Marche au hasard en 2d (***)

On se place dans le plan réel, et on s'intéresse au sous-ensemble $\mathbb{Z}^2$. À l'instant initial t=0, un observateur est situé en (0 ;0). À chaque pas de temps, il avance, au hasard et de manière équiprobable d'une unité vers le haut et le bas et d'une unité vers la gauche ou vers la droite.

1. Écrire une fonction `listepositions(n)` qui prend en argument un entier n positif, et qui renvoie une liste des positions successives que parcourt l'observateur qui marche au hasard.

2. Écrire une fonction `observateur(k,n)` qui prend en arguments deux entiers k et n strictement positifs et qui renvoie la liste des positions qu'occuperaient k observateurs marchant au hasard après n pas de temps. Quelle est la complexité de la fonction écrite ?

3. Représenter graphiquement (on pourra se servir de couleurs) la répartition de 10000 observateurs après 100 pas de temps.

4. Comment semble évoluer la probabilité qu'un observateur repasse par l'origine après un grand nombre de pas de temps ?

Exercice 4. Génération de matrices stochastiques(**)

Une matrice carrée réelle d'ordre n est dite stochastique si tous ses coefficients appartiennent à [0 ;1] et si la somme de tous les coefficients d'une ligne vaut 1 pour toutes les lignes de la matrice.

1. Écrire une fonction `generestochastique(n)` qui prend en argument un entier n strictement positif et créé une matrice stochastique aléatoire d'ordre n.

2. Écrire une fonction `teststochastique(A)` qui prend en argument une matrice `A` quelconque et renvoie un booléen qui indique si la matrice `A` est stochastique.

Exercice 5. Un problème d'urnes(**)

Soit n un entier naturel non nul. Une urne contient $2n$ boules indiscernables au toucher : n sont numérotées 0, les autres sont numérotées de 1 à n. On effectue au hasard deux tirages successifs et sans remise d'une boule dans cette urne. On note X le plus grand numéro obtenu, et Y le plus petit numéro obtenu lors de ces deux tirages.

1. Déterminer les valeurs prises par X.
2. Proposer une représentation informatique de l'urne et une fonction (écrite en langage Python) permettant de simuler une expérience telle que celle décrite ci-dessus et retournant la liste $[X, Y]$. On testera la fonction pour différentes valeurs de n.

Indication : la fonction Python `randint(start,stop)` de la bibliothèque `random` fournit un nombre entier aléatoire compris entre `start` et `stop` (ces valeurs étant incluses).
3. Proposer une fonction permettant de simuler m expériences (avec m grand) telles que celle décrite ci-dessus et retournant la liste des résultats de ces m expériences. On testera la fonction pour différentes valeurs des paramètres n et m.
4. Écrire une fonction simulant les m expériences successives pour évaluer la probabilité de l'évènement $(X = Y + 1)$ sous la forme d'une fréquence. On testera cette fonction pour différentes valeurs de m.
5. Que vaut la probabilité (théorique) de l'évènement $(X = Y + 1)$?
6. Proposer une illustration graphique confirmant l'accord entre la théorie (question 5) et la simulation (question 4).

Exercice 6. Génération d'une permutation aléatoire de $[\![1; n]\!]$ (**)

Écrire, en n'utilisant que `random.random()` qui génère un nombre aléatoire dans [0 ;1[une fonction sans argument qui renvoie une permutation aléatoire de l'intervalle entier $[\![1; n]\!]$.

Toutes les permutations devront être équiprobables.

Note : random.shuffle ne génère pas des permutations équiprobables ; comme la page suivante l'indique : https ://docs.python.org/2/library/random.html

Exercice 7. Modèles des urnes de Ehrenfest(**)

Le modèle des urnes est un modèle "stochastique" introduit en 1907 par les époux Ehrenfest pour illustrer certains des paradoxes apparus dans les fondements de la mécanique statistique naissante. Le mathématicien Mark Kac a écrit à son propos qu'il était « ... probablement l'un des modèles les plus instructifs de toute la physique. » Il étudie l'évolution d'un système complexe, où les relations de récurrence sont régies par des phénomènes aléatoires.

On considère deux urnes A et B, ainsi que N boules, numérotées de 0 à $N - 1$. Initialement, toutes les boules se trouvent dans l'urne A. Le processus stochastique associé consiste à répéter l'opération suivante :

"Choisir au hasard un numéro i compris entre 0 et $N - 1$, prendre la boule i, changer la boule i d'urne."

Nous allons utiliser une représentation informatique de cette situation à l'aide d'une liste L de longueur N composée de 0 et de 1 ; à un moment donné si la boule numéro i est dans l'urne A alors on a L[i]=1 ; si la boule numéro i est dans l'urne B alors on a L[i]=0.

1. Écrire une fonction `initial()` qui renvoie la liste L0 représentant la situation des boules à l'instant initial (toutes les boules sont dans l'urne A).

2. Écrire une fonction `transition(L)` qui prend en entrée la liste L représentant un état des urnes et renvoie la liste L après le choix d'un nombre au hasard et transfert de la boule associée.

3. Écrire une fonction `nombreA(L)` qui prend en entrée la liste L représentant un état des urnes et renvoie le nombre de boules présentes dans l'urne A.

4. Écrire une fonction `evolution(k)` qui **à l'aide des fonctions précédentes** :
- crée la liste L0 correspondant à l'état initial
- répète k fois les transitions en stockant dans une liste NA le nombre de boules dans l'urne A après chaque transition (on aura NA[0]=N et len(NA)=k+1)
- renvoie la liste NA

Dans cette modélisation, quel que soit le nombre de boules N fini, il existe toujours des retours à l'état initial, pour lesquelles toutes les boules sont dans l'urne A . Mais le temps moyen entre deux retours à l'état initial consécutifs croît très rapidement avec N, ce qui ne les rend pas facilement observables.

Formellement, on introduit une suite d'instants $\{t_n\}_{n=1,2,...}$ (finis) pour lesquels toutes les boules reviennent dans l'urne A (par convention, on pose $t_0 = 0$). On peut alors définir une nouvelle suite $\tau_n = t_n - t_{n-1}$ des durées finies entre deux retours à l'état initial consécutifs.

Le théorème de Kac (1947) affirme que cette durée moyenne vaut 2^N

$$\langle\ \tau\ \rangle\ =\ \lim_{p\to\infty}\ \frac{1}{p}\ \sum_{n=1}^{p}\ \tau_n = 2^N$$

5. Écrire une fonction `chercheN(L)` qui pour une liste L donnée en argument renvoie une liste contenant tous les indices i pour lesquels L[i]=N.

6. Écrire une fonction `differences(L)` qui pour une liste L donnée, contenant le nombre de boules dans l'urne A à chaque instant, renvoie une liste contenant les différences entre deux indices consécutifs où L[i]=N de cette liste. Quelle est la complexité de cette fonction ?

7. Écrire une fonction `moyennerec(L)` qui réalise le calcul de la durée moyenne entre deux retours consécutifs à l'état initial. L'argument est une liste L qui contient le nombres de boules dans l'urne A à chaque instant. On se servira des fonctions précédentes.

8. Proposer une validation expérimentale du théorème de Kac pour N=10.

Exercice 8. Loi uniforme sur une sphère (***)

Écrire une fonction qui génère les coordonnées sphériques d'un point d'une sphère centrée en 0 et de rayon R selon une loi uniforme.

On prendra garde à l'analyse mathématique du problème à effectuer en amont !