Université Clermont Auvergne U.E. Logiciels Scientifiques Année 2019-2020

TP 7 : Modélisation, simulation aléatoire

Les exercices dans cette feuille sont basés sur le principe de simulation : la probabilité d'un événement s'estime par sa fréquence observée, c'est-à-dire la proportion d'issues favorables dans l'ensemble des observations.

Plus généralement, on peut approcher l'*espérance mathématique*, (une quantité théorique calculée sur papier) par la moyenne *empirique*, c-à-d. la moyenne obtenue dans une simulation. On peut aussi visualiser la répartition des résultats (i.e. la *loi empirique*) à l'aide d'un histogramme. Dans le cours de Probabilités en S4, le principe de simulation prendra forme d'un théorème appelé *loi de grands nombres*.

On rappelle qu'au niveau de programmation, on structure le code de manière suivante :

- d'abord, on écrit une fonction qui réalise l'expérience aléatoire en question;
- puis, on l'exécute plusieurs fois pour en tirer des statistiques.

Exercice 1. *Pile ou face*

1. Construire une liste `L` composée de 0 et de 1 qui modélise une série de 100 lancers d'une pièce (100 jeux à pile ou face). On pourra utiliser la commande `randint` (dans `numpy.random`).
2. Calculer le nombre de piles dans la liste `L` (on pourra supposer que la pile est représenté par 1 dans la liste).
3. Écrire une fonction `pf()` (sans arguments) qui modélise une expérience de 100 jeux à pile ou face et retourne le nombre de piles obtenues. Tester.

   Quelle est la probabilité que le nombre de piles soit

   (a) strictement supérieur à 53 ?

   (b) inférieur ou égal à 40 ?

   Indication : on pourra répéter l'expérience plusieurs fois (par exemple 20000 fois) et calculer la proportion des cas où le nombre de piles obtenues vérifie la condition donnée.
4. Quelle est répartition de nombre des piles dans notre expérience ? Faire plusieurs expériences et représenter les résultats sous forme d'un histogramme. On choisira bien le nombre d'expériences et on construira les boîtes qui conviennent.

Exercice 2. *Bonne passe*

1. Quelle est la probabilité que dans une série de 100 jeux à pile ou face il y a (au moins) une suite de $k = 4$ piles consécutives ?

2. Faire un diagramme en bâtons qui visualise les probabilités d'une suite de piles pour k variant de 3 à 8.

Exercice 3. *Bernoulli*

1. Écrire une fonction `bern(a, b, p)` qui retourne la valeur `b` avec la probabilité `p` et retourne la valeur `a` avec une probabilité 1-`p`. Tester.

   Indication : Si `r = rand()`, alors `r` est inférieur à p avec la probabilité `p`,

2. Créer une liste contenant plusieurs valeurs aléatoires `bern(-1, 1, 0.3)` et afficher son histogramme. Penser à définir des boîtes adaptées aux données.

Exercice 4. *Jeu de dés*

Selon Siméon-Denis Poisson, le calcul de probabilités est né en 1654 d'un "*problème relatif aux jeux de hasard proposé à un austère janséniste*[1] *par un homme du monde* [2]".

1. Écrire une fonction `serie(n)` qui retourne un vecteur de longueur `n` composé de nombres aléatoires entiers entre 1 et 6 (tous les nombres sont équiprobables; cela représente une simulation d'une série de lancers d'un dé). Afficher un tel vecteur de longueur `n = 10`.

2. *Paradoxe du chevalier de Méré*

   (a) Quelle est la probabilité qu'en jetant 4 fois un dé on obtienne *au moins* une fois un 6 ? Répondre en faisant une simulation à l'aide de vos fonctions.

   (b) Quelle est la probabilité qu'en jetant 24 fois deux dés, on obtienne un double 6 ? Répondre en faisant une simulation.

   Indication : Pour simuler deux dés, on pourra utiliser deux listes `A` et `B` pour simuler deux dés indépendantes. L'événement "double 6" est alors réalisé lorsque les deux valeurs `A[i]` et `B[i]` sont égales à 6 pour une même valeur d'indice `i`$\in \{0, ..., 23\}$.

Exercice 5. *Marche aléatoire sur* $\mathbb{Z}$

1. Simuler et visualiser une marche au hasard sur $\mathbb{Z}$ (appelé aussi marche d'ivrogne) : un marcheur commence en $x = 0$ et à chaque seconde prend un pas à gauche ou à droite au hasard de façon équiprobable.

   Visualiser par exemple 10, 100 et 5000 pas.

   Indication : on pourra construire un tableau composé de -1 et de 1 (cf. `bern`). La position du marcheur est alors la somme cumulative de ce tableau (on utilisera la commande `np.cumsum`).

2. Superposer dans un même graphique 500 trajectoires de 500 pas.

[1] théologien

[2] Il s'agit de Blaise Pascal et d'Antoine Gombaud, dit chevalier de Méré.

3. Soit X_n la position de marcheur après n pas. Visualiser la loi empirique de X_{100}. *Indication* : on fera un histogramme avec des boîtes de largeur 2.

4. Normaliser votre histogramme (`normed=1`) et superposer avec la courbe représentative de la fonction suivante :
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$
où $\sigma = 10$. Quel théorème de la Terminale (S) rappelle la coïncidence observée ?

Exercice 6. *Marche aléatoire dans le plan $\mathbb{Z}^2$*

Partant du point de coordonnées $X_0 = (0, 0)$, on génère une suite de points dans le plan de la façon suivante : $X_{n+1} = X_n + D$ où D est un vecteur aléatoire valant (1,0), (0,1), (-1,0) ou (0,-1) avec les probabilités égales à 1/4.

1. Écrire une fonction `un_pas()` qui génère les vecteurs D ci-dessus.

2. Écrire une procédure `marche2d(n)` qui construit et visualise une trajectoire à `n` pas. Exécuter avec différentes valeurs de `n`.

Exercice 7. *Facultatif: Promenade d'un cavalier*

Objectif de cette exercice est de mesurer le temps (moyen) de parcours aléatoire d'un cavalier sur l'échiquier.

1. Écrire une fonction `un_pas(L,C)` qui étant donnée une position d'un cavalier sur l'échiquier standard 8×8 (`L` = ligne, `C` = colonne), effectue un mouvement au hasard et retourne la nouvelle position sous forme d'une liste `[L1, C1]`.

   Indication : Construire un vecteur des 8 mouvements possibles pour un cavalier sans contraintes puis, tenant compte de la position initiale `(L, C)`), choisir un mouvement au hasard jusqu'à ce que la position finale soit valable (c.à.d. dans l'échiquier).

2. Écrire une fonction `excursion(L, C)` qui simule une marche aléatoire d'un cavalier à partir d'une case `(L, C)` jusqu'à ce qu'il revienne à la case de départ. La fonction retournera le nombre de mouvements effectués.

3. Déterminer le temps moyen de retour pour chaque case de départ possible (par symétrie, il suffit de considérer une *moitié d'un quart* d'échiquier).

   Afficher le résultat sous forme d'un tableau (on pourra arrondir les résultats à 2 chiffres après la virgule).

   Créer un tableau `numpy` de forme 8x8 avec les valeurs obtenues et visualiser-le à l'aide de `plt.matshow`.