La marche de l'ivrogne

Exercice 1 — Élémentaire, mon cher Watson

in Élémentaire, mon cher Watson, Colin Bruce, Flammarion (p. 81)

– Rappelez-vous cet ivrogne que nous vîmes tituber dans Baker Street, la veille de Noël. Je viens d'apprendre qu'il s'appelait Davis. Il était, paraît-il marin à bord de l'*Illustrious*, un navire de Sa Majesté qui avait jeté l'ancre à Greenwich. Lorsque nous le vîmes, il tentait de retrouver le chemin de son bateau.

– Cela faisait un bon bout de chemin, commentais-je.

– Son intention n'était nullement de faire tout le trajet à pied. Il avait loué un petit canot et il avait ramé jusqu'à Fisherman's Wharf, un quai abandonné non loin d'ici, en se laissant porter par la marée montante. Visiblement, il pensait que la marée descendante lui permettrait de rentrer avant minuit, à la fin de sa permission. Mais en réalité, il n'est jamais rentré.

– Vu dans l'état dans lequel il était lorsque nous l'avons vu, il aurait pu lui arriver toutes sortes de choses.

– Et, de fait, il lui est arrivé quelque chose. La dernière partie de son chemin jusqu'au canot fut particulièrement mouvementée. Il faisait nuit noire sur Fisherman's Wharf. Le quai, très large, fait cent pas de long. Lorsqu'on arrive au bout, on tombe brutalement sur la rivière, sauf au centre, où une promenade en bois s'étend sur douze pas, de part et d'autre du milieu.

– Je ne vois pas comment il aurait pu retrouver son chemin, dis-je.

– C'est très facile pour quelqu'un qui n'a pas bu. Le quai est orienté plein nord. Au moment où il est arrivé, l'étoile Polaire devait être bien visible. Il lui suffisait de marcher vers elle, puis de sauter du bord du quai sur la passerelle, au centre de laquelle son canot était amarré. Mais nous savons qu'il était saoul et qu'au lieu de marcher droit il faisait des zigzags. D'ailleurs d'après ce que nous avons vu et ce qu'ont constaté par la suite d'autres témoins, son trajet était une parfaite marche au hasard, au sens mathématique, c'est-à-dire qu'il ne suivait aucune règle dans ses changement de direction et n'avait aucune préférence quant au côté vers lequel il titubait. Il faisait un pas vers l'avant, suivi d'un pas de côté involontaire, d'une façon tellement aléatoire qu'il serait difficile pour un homme sobre d'en faire autant. Ainsi, bien qu'il se fût dirigé

vers le nord, en direction du centre du quai, il se retrouva décalé de treize pas vers la droite lorsqu'il arriva à la rivière.

– Un nombre qui porte malheur, ajoutais-je.

– Cela lui a effectivement porté malheur. La nuit était froide, et une fine couche de glace s'était formée sur l'eau. En regardant vers le bas, il a aperçut une surface solide qu'il a dû prendre pour la passerelle en bois et il a sauté. On a retrouvé hier au fond de la rivière des clés et d'autres affaires lui appartenant. La marée a dû emporter son corps jusqu'à la mer. On ne le retrouvera sans doute jamais.

– Le pauvre homme. Cet accident semble toute fois assez banal. Pourquoi Lestrade a-t-il éprouvé le besoin de vous demander conseil ?

– C'est là que nous touchons à l'originalité de cette histoire. Davies avait pris l'avant veille une assurance-vie pour une somme de cent guinées. Il n'avait pas de famille, et la seule bénéficiaire de ce contrat était sa sœur. Il semble particulièrement curieux qu'il ait pris une assurance aussi chère, alors que l'on sait qu'il avait des dettes de jeu. La compagnie d'assurances pense qu'il a pu se suicider et refuse de payer – rappelez-vous que, par cette froide nuit de veille de Noël, plusieurs malheureux se sont jetés du haut des ponts de la Tamise.

– Lestrade doit être désolé pour la pauvre sœur. Notre inspecteur a le cœur moins dur qu'il voudrait nous le faire croire. Mais l'argument principal de la compagnie d'assurance, c'est que Davis faisait tout bonnement semblant d'être saoûl et que, après s'être assuré que de nombreux témoins l'avaient vu incapable de marcher en ligne droite, il s'est jeté à l'eau volontairement. D'ailleurs le fait qu'il soit passé devant la résidence du détective le plus connu de Londres n'était peut-être pas un hasard.

  […] Je frissonnais à l'idée que l'ivrogne qui nous avait intrigués cette nuit là était peut-être en réalité en train de marcher froidement vers la mort, tout en sachant que nous l'observions. […]

– Très bien. J'aurais alors le plaisir de vous laissez expliquer cette affaire à Lestrade.

La marche de l'ivrogne

Partie A – Une marche de l'ivrogne

```python
#-*- coding:utf8 -*-
# Python 3

"""
l'ivrogne fait 100 pas en allant aléatoirement vers la
gauche ou vers la droite. On cherche la probabilité
qu'il ne soit pas dans l'intervalle [-12 ; 12]
au bout des 100 pas.
"""

from random import randrange

def marche():
    """ simule une marche aléatoire de 100 pas et renvoie
    l'ordonnée de l'arrivée
    """
    y = 0
    for _ in range(100): # 100 pas
    # démontrer que le test "classique"
    # if randrange(0,2) == 0: # 0 = gauche
    #     y = y + 1
    # else:
    #     y = y - 1
    # peut être remplacé par
        y += 1 - 2 * randrange(0, 2)
    return y

def n_marches(n=1000):
    # exécute <n> (par défaut 1000) marches et renvoie la
    # probabilité que l'ivrogne tombe à l'eau
    est_tombe = 0
    for k in range(n):
```

```
33          arrivee = marche()
34          if (arrivee <= -13) or (arrivee >= 13):
35              est_tombe += 1
36      return est_tombe / n
```

Remarque pour le prof de maths : Yves et Médéric me l'ont confirmé... Si G est une v.a. qui suit la loi binomiale $\mathscr{B}(100;0{,}5)$ alors la v.a. $X = 100 - 2G$ décrit la marche de l'ivrogne.

G est la v.a. qui compte le nombre de pas à gauche. En modélisant le point de départ à l'origine du repère et en allant vers les valeurs positives des ordonnées (100 pas vers le Nord), pour connaître l'abscisse X de l'arrivée, il faut calculer $X = D - G = (100 - G) - G = 100 - 2G$ (D nombre de pas à droite) et donc $P(-12 \leqslant X \leqslant 12) = P(44 \leqslant G \leqslant 56) \approx 0{,}8$

Partie B – Pour visualiser les résultats

```
1  #-*- coding:utf8 -*-
2  # Python 3
3  #
4  # Marche de l'ivrogne
5  #=====================
6  # Marche aléatoire G / D sur 100 pas
7  # Recherche de la probabilité de ne pas être
8  # à la fin dans l'intervalle [-12 ; 12]
9
10 from random import randrange
11 import matplotlib.pyplot as plt
12 import numpy as np
13
14 def marche():
15     """ donne l'ordonnée de l'arrivée après 100 pas """
16     arrive = 0
17     for _ in range(100):
18         arrive += 1 - 2 * randrange(0,2)
19         # randrange (a,b) entier aléatoire dans [a, b[
20     return arrive
21
22 def batons(n, aff=True):
23     """ prépare et peut afficher le diagramme en bâtons de <
       ↪n>
24     marches aléatoires et renvoie la liste des <n> arrivées.
25     """
26     arrivees = np.array([marche() for _ in range(n)])
27     # <arrivees> ne contient que nb. pairs (à démontrer ?) :
28     # attention bornes (bins) !!
29     # rwidth : épaisseur des rectangles relatifs aux bornes
30     freq, bornes, patches = plt.hist(arrivees,
31         bins=range(arrivees.min()-1, arrivees.max()+2, 2),
```

```
32          facecolor='g', alpha=0.75, rwidth=.5) #, density=
        ↪True
33      if aff:
34          plt.grid()
35          plt.title('Arrivées')
36          plt.show()
37      return arrivees
38
39  def p_in_ab(l, a, b):
40      """ retourne la proportion de nombres de la liste <l>
41      dans l'intervalle [a; b].
42      utilisation : p_in_ab(batons(1000,aff=False), -12, 12)
43      """
44      return len([k for k in l if k in range(a, b+1)]) / len(l
        ↪)
```

Exercice 2 — En longeant un mur

Le mur est l'axe des abscisses, le trottoir a une largeur de 4 pas : si l'ivrogne fait cinq pas, il tombe dans le caniveau et y passe la nuit ; sinon il continue d'avancer jusqu'à arriver chez lui à n pas du café.

À quelle distance maximale du café doit il habiter (en pas) afin que la probabilité qu'il dorme chez lui soit supérieure ou égale à 0,5 ?

On suppose qu'en sortant du café l'ivrogne se dirige vers la droite (c'est bien de ce côté qu'est sa maison !)

Soit Y la variable aléatoire qui donne l'ordonnée (positive) du point d'arrivé.

1. Construire un arbre représentant cette situation pour 3 pas et déterminer la loi de Y. Puis reprendre la question pour 4 pas.
2. Dans chacun des cas précédents donner la probabilité que l'ivrogne ne dorme pas chez lui.
3. Écrire un programme Python simulant n marches aléatoires afin d'obtenir une estimation de l'ordonnée de l'arrivée pour un nombre de pas à déterminer.
4. Donner une représentation graphique de la loi de probabilité de la variable aléatoire Y.
5. Répondre à la question initiale.

La marche de l'ivrogne

La fonction arange de la bibliothèque numpy permet d'avoir un intervalle avec des bornes non entières.

La fontion savetxt de la bibliothèque numpy permet de sauver un tableau dans un fichier .csv et de préciser le séparateur de données.

```
#-*- coding:utf8 -*-
# Python 3
"""
l'ivrogne avance d'un pas en allant aléatoirement à gauche ou à droite.
il sort du café en longeant le mur sur son côté droit,
s'il ne peut pas aller à droite : il continue à longer le mur.
Le trottoir est large de 4 pas, s'il tombe dans le caniveau
(il a fait 5 pas à gauche) il ne se relève pas et fini sa nuit ainsi.
A quelle distance (en pas) maximale doit-il habiter du café
afin d'avoir une proba de dormir chez lui supérieure ou égale à 0,5 ?
"""

from random import randrange
from matplotlib import pyplot as plt
import numpy as np


def marche(pas_max):
    """simule une marche de l'ivrogne de longueur <pas_max>
    et renvoie l'ordonnée de l'ivrogne (entier de [0, 5])
    """
    pas, y = 0, 0
    while pas <= pas_max and y <= 4:
        if randrange(0,2) == 0: # 0 = droite
```

```
25              y = max(y - 1, 0) # aller à droite ou longer le
    ↩mur
26          else:
27              y = y + 1
28          pas = pas + 1
29      return y
30
31  def maison(n, pas_max):
32      """simule <n> marches de l'ivrogne de longueur <pas_max>
33      et affiche la probabilité de dormir à la maison.
34      """
35      tombe = 0
36      for k in range(n):
37          if marche(pas_max) > 4:
38              tombe = tombe + 1
39      print("proba de dormir à la maison", 1 - tombe / n)
40
41  def batons(n, pas_max, freq=True):
42      """ affiche le diagramme en bâtons de <n>
43      marches aléatoires de longueur <pas_max> et renvoie la
44      liste des ordonnées d'arrivées.
45      """
46      arrivees = np.array([marche(pas_max) for _ in range(n)])
47      # bins = le nombre d'intervalles [x0, x1[, [x1, x2[, ...
48      # [x(n-1), x(n)] avec x0 et x(n) les bornes de range
49      # rwidth = l'épaisseur des rectangles relative aux
    ↩bornes
50      # density=True pour avoir la fréquence de chaque valeur.
51      freq, bornes, patches = plt.hist(arrivees,
52                  range = (arrivees.min() - .5, arrivees.max()
    ↩ + .5),
53                  bins = 6,
54                  facecolor='g', rwidth=.5) #density=freq,
```

```
55      plt.grid()
56      plt.title('Arrivées')
57      plt.show()
58      # sauver le tableau contenant les ordonnées
59      # pour le chemin : utiliser /
60      # %d pour préciser que les nombres sont entiers
61      # \t le séparateur de données est la tabulation s'il y a
        ↪ plusieurs
62      # sur la même ligne (ce n'est pas le cas ici)
63      np.savetxt('//SRV1-2019/sstage1$/Documents/leon/arrivees
        ↪.csv', arrivees, delimiter = '\t', fmt = '%d')
64      return 0
```