Plaquette 4 – Simuler avec Python : fonctions et séries de fréquences

Échantillonnage et fluctuation — Seconde

Exercices

Méthode

On écrit des **fonctions** qui renvoient le résultat d'une expérience aléatoire, puis on appelle ces fonctions un grand nombre de fois pour obtenir une **série statistique** (de résultats ou de fréquences). On lit et on complète des fonctions qui calculent la moyenne et l'écart type de cette série au fil de la boucle, sans stocker les valeurs, et l'on interprète ce que la série révèle de la fluctuation d'échantillonnage.

Exercice 1 – Expérience et répétition

On considère les deux fonctions suivantes.

```
from random import randint

def experience():
    if randint(1, 6) >= 5:
        return 1
    return 0

def repetition(n):
    total = 0
    for i in range(n):
        total = total + experience()
    return total / n
```

a) Que renvoie *experience()* ? Avec quelle probabilité renvoie-t-elle 1 ?

b) Que renvoie *repetition(n)* ? Autour de quelle valeur s'attend-on à la trouver pour $n = 10\,000$?

Exercice 2 – La moyenne d'une série de lancers

La fonction suivante lance n fois un dé et renvoie la moyenne des faces obtenues.

```
from random import randint

def moyenne_des(n):
    s = 0
    for i in range(n):
        s = s + randint(1, 6)
    return s / n
```

a) Lors d'un appel *moyenne_des(4)*, les lancers donnent 2, 6, 3, 5. Que renvoie la fonction ?

b) La moyenne des six faces d'un dé vaut 3,5. Des appels ont donné 3,42 pour $n = 100$, 3,519 pour $n = 1\,000$ et 3,5083 pour $n = 10\,000$. Commenter.

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

Exercice 3 – Moyenne d'une série de fréquences

On dispose d'une fonction *frequence(n)* qui renvoie la fréquence de « face » sur n lancers d'une pièce équilibrée. On écrit :

```
def moyenne_frequences(k, n):
    s = 0
    for i in range(k):
        s = s + frequence(n)
    return s / k
```

a) Combien de lancers sont simulés au total lors de l'appel *moyenne_frequences(20, 50)* ?

b) Lors d'un appel *moyenne_frequences(4, 50)*, les quatre appels à *frequence(50)* renvoient 0,42 ; 0,56 ; 0,5 ; 0,48. Que renvoie la fonction ?

c) Ces quatre fréquences s'écartent de 0,5, mais leur moyenne beaucoup moins. Pourquoi ?

Exercice 4 – Écart type d'une série de fréquences

On complète la fonction précédente pour calculer aussi l'écart type des k fréquences obtenues :

```
from math import sqrt

def ecart_type_frequences(k, n):
    s = 0
    s2 = 0
    for i in range(k):
        f = frequence(n)
        s = s + f
        s2 = s2 + f * f
    m = s / k
    return sqrt(s2 / k - m * m)
```

a) Que contiennent s et $s2$ à la fin de la boucle ?

b) Exécuter la fonction à la main lorsque les quatre fréquences obtenues sont 0,4 ; 0,5 ; 0,6 ; 0,5.

c) Que mesure le nombre renvoyé ?

Exercice 5 – Adapter une fonction à une autre expérience

La fonction suivante simule le lancer d'une pièce.

```
from random import random

def lancer():
    if random() < 0.5:
        return "face"
    return "pile"
```

a) Écrire une fonction *frequence_face(n)* qui appelle n fois *lancer()* et renvoie la fréquence de « face ».

b) Modifier *lancer()* pour simuler une pièce truquée qui tombe sur « face » dans 60 % des cas.

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

c) Écrire une fonction *tirage()* qui simule le tirage d'une boule dans une urne contenant 3 boules rouges, 5 vertes et 2 jaunes, et renvoie sa couleur.

Exercice 6 – Combien de lancers pour un 6, en moyenne ?

La fonction *attente()* renvoie le nombre de lancers d'un dé nécessaires pour obtenir le premier « 6 ». On écrit :

```
def attente_moyenne(k):
    s = 0
    for i in range(k):
        s = s + attente()
    return s / k
```

a) Que renvoie *attente_moyenne(k)* ?

b) Des appels ont donné 6,65 pour $k = 100$, 5,937 pour $k = 1\,000$ et 6,0347 pour $k = 10\,000$. Quelle conjecture peut-on faire ?

c) Un joueur affirme : « puisqu'il faut en moyenne 6 lancers, si je n'ai pas eu de 6 après 5 lancers, le prochain sera forcément un 6 ». Qu'en penser ?

Exercice 7 – Le plus grand de deux dés

On lance deux dés et on note le plus grand des deux résultats.

a) Écrire une fonction *maximum_deux_des()* qui simule l'expérience. On pourra utiliser la fonction *max* de Python : *max(3, 5)* vaut 5.

b) En dressant le tableau des 36 issues, déterminer la probabilité que le plus grand résultat soit 6.

c) Sur 1 000 simulations, le maximum a valu 6 à 311 reprises. Comparer.

Exercice 8 – Une fonction qui renvoie un booléen

On considère la fonction suivante.

```
from random import randint

def double():
    a = randint(1, 6)
    b = randint(1, 6)
    return a == b
```

a) Quel est le type de la valeur renvoyée ? Dans quel cas vaut-elle *True* ?

b) Calculer la probabilité que *double()* renvoie *True*.

c) Écrire une fonction *frequence_doubles(n)* qui renvoie la fréquence des doubles sur n lancers. Sur 1 000 lancers, une simulation a compté 152 doubles : commenter.

Exercice 9 – Exécuter une partie à la main

On joue à pile ou face : on lance une pièce jusqu'à obtenir « face », mais au plus 4 fois. On gagne 10 € si « face » sort au premier lancer, 5 € au deuxième, 2 € au troisième, 1 € au quatrième, et rien sinon.

```
from random import random
```

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

```
def partie():
    gain = 10
    lancers = 0
    while lancers < 4:
        lancers = lancers + 1
        if random() < 0.5:
            return gain
        if gain == 10:
            gain = 5
        elif gain == 5:
            gain = 2
        else:
            gain = 1
    return 0
```

a) Exécuter *partie()* à la main lorsque *random()* renvoie 0,8, puis 0,6, puis 0,3.

b) Que renvoie la fonction si *random()* renvoie quatre fois un nombre supérieur à 0,5 ?

Exercice 10 – Un return mal placé

Un élève écrit la fonction suivante pour obtenir la fréquence des « pile » sur n lancers :

```
from random import random

def frequence_pile(n):
    c = 0
    for i in range(n):
        if random() < 0.5:
            c = c + 1
        return c / n
```

a) Il obtient toujours 0 ou $\frac{1}{n}$. Expliquer pourquoi.

b) Corriger la fonction.

Exercice 11 – Gain moyen d'un jeu

À une fête foraine, une partie coûte 1 €. On lance un dé : on gagne 5 € si l'on obtient « 6 », rien sinon.

```
from random import randint

def gain():
    if randint(1, 6) == 6:
        return 5
    return 0

def gain_moyen(n):
    s = 0
    for i in range(n):
        s = s + gain()
    return s / n
```

a) Que renvoie *gain_moyen(n)* ?

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

b) Une simulation de 10 000 parties a donné *gain_moyen(10000)* = 0,832. Comment ce résultat s'explique-t-il avec la fréquence des « 6 » ?

c) Le jeu est-il avantageux pour le joueur ?

Exercice 12 – Proportion d'échantillons proches de la probabilité

On dispose de *frequence(n)*, qui renvoie la fréquence de « face » sur n lancers d'une pièce équilibrée. On écrit :

```
def proportion_proches(k, n):
    c = 0
    for i in range(k):
        f = frequence(n)
        if abs(f - 0.5) <= 0.1:
            c = c + 1
    return c / k
```

a) Que teste la condition *abs(f - 0.5) <= 0.1* ? (*abs* renvoie la valeur absolue.)

b) Des appels avec $k = 500$ ont renvoyé 0,79 pour $n = 25$, 0,952 pour $n = 100$ et 1 pour $n = 400$. Interpréter ces résultats.

Exercice 13 – Simuler un tirage avec remise

Une urne contient 4 boules rouges et 6 boules noires. On tire une boule, on note sa couleur, on la remet, et on tire une seconde boule.

a) Écrire une fonction *deux_rouges()* qui renvoie *True* si les deux boules tirées sont rouges.

b) On admet que la probabilité de tirer deux boules rouges vaut 0,16. Combien de fois l'événement devrait-il se produire, environ, sur 5 000 simulations ?

c) Un élève simule 50 tirages et trouve 14 fois deux rouges. Il en déduit que la probabilité admise est fausse. Qu'en penser ?

Exercice 14 – Lire une fonction mystère

On considère la fonction suivante.

```
from random import randint

def mystere(n):
    c = 0
    for i in range(n):
        a = randint(1, 6)
        b = randint(1, 6)
        if a + b >= 10 or a == b:
            c = c + 1
    return c / n
```

a) Décrire en français l'événement dont la fonction calcule la fréquence.

b) En dressant le tableau des 36 issues, calculer la probabilité de cet événement.

c) Quelle valeur approchée s'attend-on à obtenir avec *mystere(100000)* ?

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

Exercice 15 – Synthèse : les paris du chevalier de Méré

Au XVII^{e} siècle, le chevalier de Méré pariait sur deux jeux :

— jeu 1 : obtenir au moins un « 6 » en lançant 4 fois un dé ;
— jeu 2 : obtenir au moins un double six en lançant 24 fois deux dés.

Il pensait que les deux jeux étaient aussi favorables l'un que l'autre. Pascal et Fermat étudièrent la question.

a) Écrire une fonction *jeu1()* qui renvoie *True* si au moins un « 6 » apparaît en 4 lancers.

b) Compléter la fonction *jeu2()* de même :

```
def jeu2():
    for i in range(24):
        if ...:
            return True
    return False
```

c) Des simulations ont donné les fréquences de gain suivantes :

Parties simulées	**1 000**	**10 000**	**100 000**
Jeu 1	0,510	0,520	0,517
Jeu 2	0,507	0,496	0,490

Quel jeu semble favorable au joueur ? Pourquoi 1 000 parties ne suffisaient-elles pas pour s'en rendre compte ?

d) On admet que les probabilités de gain valent environ 0,518 pour le jeu 1 et 0,491 pour le jeu 2. Les simulations sont-elles cohérentes avec ces valeurs ?

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).