

Plaquette 5 – Simuler et résoudre des problèmes

Algorithmique et Python — Première STMG

Exercices

Méthode

Utiliser le générateur `random()` pour **simuler** une épreuve de Bernoulli, des échantillons et des expériences répétées ; comparer les fréquences simulées aux probabilités ; puis écrire des programmes **modulaires** qui cherchent un seuil ou un optimum.

Exercice 1 – Simuler une épreuve de Bernoulli

```
from random import random
```

```
def bernoulli(p):
    if random() < p:
        return 1
    return 0
```

- Pourquoi `bernoulli(0.3)` renvoie-t-elle 1 avec la probabilité 0,3 ?
- Comment utiliser cette fonction pour simuler un client qui achète avec la probabilité 0,3 ?

Exercice 2 – Fréquence dans un échantillon

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

- Que renvoie `frequence(100, 0.3)` ?
- Cinq appels donnent 0.22, 0.27, 0.27, 0.26 et 0.24. Commenter.

Exercice 3 – Simuler 500 échantillons

On simule 500 échantillons de taille 100 d'une loi de Bernoulli de paramètre 0,3 :

```
F = [frequence(100, 0.3) for k in range(500)]
```

La série F a pour moyenne 0,3018 et pour écart-type $s \approx 0,0467$; on compte 335 fréquences dans $[p - s; p + s]$, 478 dans $[p - 2s; p + 2s]$ et 500 dans $[p - 3s; p + 3s]$.

- Que contient F ?
- Calculer les pourcentages correspondants et commenter.

Exercice 4 – Compter dans un intervalle

Compléter la fonction qui compte, dans une liste F de fréquences, celles qui sont à au plus k écarts-types s de p .

```
def compte(F, p, s, k):
```

```
n = 0
for f in F:
    if ...:
        n = n + 1
return n
```

Exercice 5 – Taille de l'échantillon et dispersion

Pour $p = 0,3$, on simule 500 échantillons de taille n et l'on calcule l'écart-type s des fréquences : $s \approx 0,0903$ pour $n = 25$, $s \approx 0,0480$ pour $n = 100$ et $s \approx 0,0231$ pour $n = 400$. Que constate-t-on ? Relier à $\frac{1}{\sqrt{n}}$.

Exercice 6 – Simuler un dé

```
from random import randint
de = [randint(1, 6) for k in range(6000)]
print(de.count(6))
```

Un élève obtient 1009. Calculer la fréquence du 6 et la comparer à la probabilité.

Exercice 7 – Estimer une probabilité par simulation

```
def au_moins_un_6():
    for k in range(4):
        if randint(1, 6) == 6:
            return 1
    return 0

N = 10000
print(sum(au_moins_un_6() for k in range(N)) / N)
```

- Que simule `au_moins_un_6` ?
- Le programme affiche 0.5243. Comparer avec la probabilité exacte.

Exercice 8 – Simuler trois clients

```
def acheteurs():
    return sum(bernoulli(0.2) for k in range(3))

L = [acheteurs() for k in range(10000)]
print([L.count(k) / 10000 for k in range(4)])
```

Le programme affiche $[0.5148, 0.3802, 0.0961, 0.0089]$. Comparer avec la loi exacte du nombre d'acheteurs.

Exercice 9 – Repérer une erreur de simulation

Pour simuler un colis livré en retard avec la probabilité 0,1, un élève écrit `if random() > 0.1:` et obtient environ 90 % de retards. Expliquer et corriger.

Exercice 10 - Une fonction de seuil réutilisable

Écrire une fonction *seuil*(u_0 , q , S) qui renvoie le premier rang n tel que $u_0 \times q^n > S$. Donner *seuil*(1200, 1.08, 2000) et *seuil*(5000, 1.02, 6000), et interpréter le second.

Exercice 11 - Chercher un optimum par balayage

Le bénéfice d'une entreprise pour q articles est $B(q) = -q^2 + 80q - 1200$. Écrire un programme qui teste les quantités entières de 0 à 60 et renvoie celle qui donne le plus grand bénéfice.

Exercice 12 - Estimer l'espérance d'un jeu

Un jeu de kermesse : on lance un dé ; avec un 6, on gagne 8 € ; avec 4 ou 5, on gagne 1 € ; sinon, on perd 3 €.

```
def gain():
    d = randint(1, 6)
    if d == 6:
        return 8
    elif d >= 4:
        return 1
    else:
        return -3
```

Un élève calcule la moyenne de 10 000 gains simulés et trouve 0,1309. Calculer l'espérance exacte du gain et comparer.

Exercice 13 - Découper un programme en fonctions

On veut estimer, par simulation, la probabilité qu'au moins un des quatre colis d'une commande arrive en retard, sachant que chaque colis est en retard avec la probabilité 0,1, indépendamment. Proposer un programme découpé en trois fonctions, puis donner la probabilité exacte que la simulation doit approcher.

Exercice 14 - Attendre le premier succès

Un commercial obtient un rendez-vous à chaque appel avec la probabilité 0,3, indépendamment.

```
def attente(p):
    n = 1
    while random() >= p:
        n = n + 1
    return n
```

- Que renvoie *attente*(0.3) ?
- Sur 10 000 simulations, la valeur 3 apparaît avec la fréquence 0,1419. Calculer la probabilité que le premier rendez-vous soit obtenu au troisième appel, et comparer.

Exercice 15 - Synthèse : faut-il faire une promotion ?

Un magasin accueille 100 clients par jour. Sans promotion, chaque client achète avec la probabilité 0,25 et la marge est de 12 € par achat. Avec une promotion, chaque client achète avec la probabilité 0,35, mais la marge tombe à 9 € par achat.

- a) Écrire une fonction *marge_jour*(p, m) qui simule une journée et renvoie la marge totale.
- b) Calculer l'espérance de la marge journalière dans chaque cas.
- c) Une simulation de 1000 journées donne en moyenne 301,75 € sans promotion et 313,05 € avec. Conclure.