



Math93.com

TD n°1 - Algorithmique

Statistiques

Échantillonnage



le module random

En probabilités, on est amené à utiliser des générateurs pseudo-aléatoires, proposés par la bibliothèque `random`, qu'on ouvre avec la commande `import random` au début du programme. Les fonctions d'usage le plus fréquent sont :

- `random.random()` qui renvoie un nombre réel (pseudo)-aléatoire de l'intervalle semi-ouvert $[0,1[$;
- `random.randint(a,b)` qui renvoie un nombre entier (pseudo)-aléatoire compris entre a et b (bornes incluses) ;
Par exemple `random.randint(1,100)` renvoie un entier compris entre 1 et 100
- `random.choice(L)` qui renvoie un élément tiré au sort de la liste L .

Exercice 1. Un échantillon de taille n

La fonction ci-dessous simule n lancers d'un dé équilibré à 6 faces numérotées de 1 à 6, et renvoie le résultat dans une liste nommée L .



Code Python

```
import random
def echantillon(n) :
    ''' Simule n lancers d'un dé à 6 faces '''
    assert n >= 1
    L = []
    for i in range(n) :
        L.append(random.randint(1,6))
    return L
```



Liste et `L.append(i)`

- `L = []` : crée une liste vide
- `L.append(i)` : ajoute un élément i en fin de la liste L .

Remarque : En anglais, "append" signifie "joindre" ou "apposer".

1. Tester ce programme avec différentes valeurs de n .
2. S'inspirant de l'écriture mathématique d'un ensemble en compréhension, Python propose une syntaxe utile pour la création d'une liste en compréhension. On peut ainsi utiliser l'expression


$$\left[\text{random.randint}(1,6) \text{ for } i \text{ in range}(n) \right]$$

pour créer la liste simulant un échantillon de taille n de lancers.

Modifier le programme précédent en créant une nouvelle fonction, `echantillon2(n)` qui sera bien plus simple.


Exercice 2. Nombre de pairs sur un échantillon

3. Écrire une fonction qui compte le nombre de chiffres pairs obtenus sur échantillon de taille n de l'exercice 1.

 **Code Python**

```
def nombredepairs(n) :
    assert n >= 1
    L = echantillon(n)
    compteur = 0
    for i in L :
        if # À compléter. :
            compteur = compteur + 1
    return compteur
```

4. Écrire maintenant une fonction qui donne la fréquence de nombres pairs obtenus.

 **Code Python**

```
def frequencedepairs(n) :
    assert n >= 1
    # À compléter.
```


5. On note p la probabilité d'obtenir un nombre pair en lançant un dé équilibré à 6 faces. Déterminer l'intervalle de fluctuation au seuil 95% de la fréquence f de pairs obtenus sur un échantillon de taille n .

- Conditions : $\begin{cases} n \dots \\ p \dots \end{cases}$
- $I_n = \dots$

6. Vérifier si la fréquence observée appartient à l'intervalle de fluctuation.

- $f = \dots$

7. Écrire une fonction `intervalle(p,n)` qui renvoie les bornes de l'intervalle de fluctuation au seuil 95% de la fréquence f , avec une probabilité p , sur un échantillon de taille n . Vous devrez introduire une assert avec les conditions de validation de l'intervalle.

 **Code Python**

```
def intervalle(p, n) :
    assert n \dots and p \dots and p \dots
    return (... , ...)
```


Racine carrée

La racine carrée du nombre x se note `math.sqrt(x)` après avoir importé le module `math` au début de votre programme sous la forme : `import math`.

Exercice 3. Fluctuation d'échantillonnage

8. Lancer plusieurs fois dans la console la fonction `frequencedepairs(n)` jusqu'à obtenir une fréquence qui n'appartient pas à l'intervalle de fluctuation associé.

9. Écrire une fonction (booléenne) nommée `test(n)` qui renvoie `TRUE` si la fréquence de l'échantillon appartient à l'intervalle de fluctuation et `FALSE` sinon.

 **Code Python**

```
def test(n) :
    p = 0.5
    (a , b) = intervalle(p , n)
    if ..... : # À compléter.
        return True
    else :
        return False
```

10. On rappelle la propriété du cours :


Propriété 1 (Intervalle de fluctuation au seuil 95% (admis))

Soit un caractère dont la proportion dans une population est p .
Pour n assez grand et p ni proche de 0, ni proche de 1, il y a environ 95% des échantillons de taille n issus de cette population qui sont tels que la fréquence f observée appartienne à l'intervalle :

$$I = \left[p - \frac{1}{\sqrt{n}} ; p + \frac{1}{\sqrt{n}} \right]$$

Les conditions sont : $\begin{cases} n \geq 25 \\ 0,2 \leq p \leq 0,8 \end{cases}$.

On cherche à tester la validité de la propriété pour une centaine d'échantillons, pour un milliers ...
Pour cela on va créer une fonction nommée `multi(n , nbechantillons)` qui envoie le pourcentage d'échantillons de taille n tels que la fréquence f appartienne à l'intervalle de fluctuation asymptotique. On utilisera pour cela la fonction `test(n)` précédente.


 **Code Python**

```
def multi(n , nbechantillons) :
    compteur = 0
    # À compléter.
    return .....
```

Exercice 4. Complément : un histogramme

On peut facilement construire l'histogramme des effectifs liés à chaque numéro de faces.
On importe le sous-module `pyplot` de `matplotlib` qu'on renomme au passage `plt`, par commodité.
On l'importe en écrivant en début de programme `import matplotlib.pyplot as plt`.

Sur un Python installé, pas de problème par contre sur `repl` ou les logiciels online, le graphique doit être converti en image.
Sur `repl`, il faut en outre convertir votre fichier en projet avec l'icône *add new file* en haut à gauche.

 **Code Python**

```
import random
import matplotlib.pyplot as plt
fig = plt.figure() # nécessaire seulement sur repl.
plt.hist(echantillon(50),20)
fig.savefig('graph.png') # ou simplement plt.show() sur un Python installé comme anaconda
```