

Python 10 : Simulations aléatoires

Il est possible de simuler des nombres de manière aléatoires avec l'outil informatique. On obtient alors un programme, qui, contrairement à un algorithme standard, donne un résultat non nécessairement identique à chaque exécution.

Dans la suite on suppose avoir importé la bibliothèque `numpy.random` à l'aide de :

```
import numpy.random as rd
```

1) Simuler des nombres aléatoirement

La commande `rd.randint(a,b+1)` renvoie un nombre entier entre a et b de manière aléatoire et uniforme. C'est-à-dire qu'elle renvoie une réalisation d'une loi uniforme X sur $[[a, b]]$.

On peut l'utiliser pour simuler un tirage dans une urne par exemple.

Disons que l'on a une urne contenant N blanches et M noires.

`X=rd.randint(1,M+N+1)` : # on génère un nombre uniformément dans $[[1, M + N]]$.

if `1<=X<=N` : # on regarde si on a eu une blanche

`print('on a pioché une blanche')`

else :

`print('on a pioché une noire')`

Pour simuler un événement de probabilité p donnée :

La commande `rd.random()`¹ génère aléatoirement un nombre réel de l'intervalle $]0,1[$ de manière uniforme. Ainsi, si p est un réel de $]0,1[$ la probabilité que `rd.random()` soit inférieur à p vaut p (par exemple la probabilité qu'un nombre aléatoire dans $]0,1[$ soit inférieur à $\frac{1}{4}$ vaut bien $\frac{1}{4}$).

2) Exemples simples

Simulons le rang de la première obtention d'un Pile lors d'une succession de lancer de pièce équilibrée. Ce rang est noté X :

```
X=1
```

```
while rd.random()<1/2 : #tant qu'on a un échec (de probabilité 1/2)
```

```
X+=1 # on ajoute 1 à X
```

```
print(X)
```

¹ `rd.rand()` renvoie la même chose

ECG

Simulons le nombre de pile obtenus lors d'une succession de 10 lancers de pièce équilibrée. Ce nombre est noté X .

```
X=0
for k in range(10):           # on répète 10 expériences
    if rd.random()<1/2:       # si on a un succès, on augmente X de 1
        X+=1
print(X)
```