Cours

3 Tirages d'une variable aléatoire

Capacités exigibles

- Utiliser les fonctions de base des bibliothèques **random** et/ou **numpy** (leurs spécifications étant fournies) pour réaliser des tirages d'une variable aléatoire
- Utiliser la fonction **hist** de la bibliothèque **matplotlib.pyplot** (sa spécification étant fournie) pour représenter les résultats d'un ensemble de tirages d'une variable aléatoire
- Déterminer la moyenne et l'écart-type d'un ensemble de tirages d'une variable aléatoire

A Histogramme, moyenne et écart-type

A.1 Tracé d'un histogramme

Pour représenter graphiquement un ensemble de valeurs sous forme d'un histogramme, on aura chargé préalablement le module **pyplot** de la bibliothèque **matplotlib** avec l'instruction :

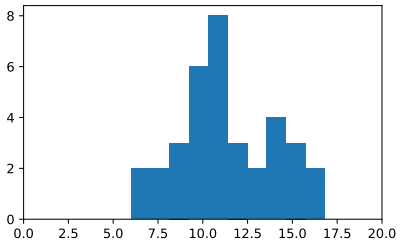
`import matplotlib.pyplot as plt`

Puis, on emploiera l'instruction :

`plt.hist(`*ensemble des valeurs étudiées***`)`**

Voici un exemple avec les moyennes individuelles d'une ancienne promotion sur une année complète :

```
1  import matplotlib.pyplot as plt
2
3  # Moyennes individuelles des étudiants
4  X = [7.4,10.8,6.0,10.6,11.4,15.1,14.1,13.8,15.3,
5       13.8,8.8,10.7,10.2,8.5,13.4,16.2,11.1,9.1,9.8,12.8,
6       9.7,9.6,10.6,14.6,16.8,11.7,11.1,9.6,10.4,9.3,15.2,
7       10.5,7.1,6.2,12.0]
8
9  # Tracé de l'histogramme
10 plt.hist(X)
11 plt.xlim(0,20)
12 plt.show()
```



A.2 Calcul d'une moyenne et d'un écart-type

Il faudra importer la bibliothèque **numpy** avec **`import numpy as np`**, puis :

- pour le calcul de la **moyenne** : **`np.average(`***liste des valeurs étudiées***`)`**
- pour le calcul de l'**écart-type de la population** σ : **`np.std(`***liste des valeurs étudiées***`)`**
- pour le calcul de l'**écart-type de l'échantillon** s_x (ou s_{n-1}) : **`np.std(`***liste des valeurs étudiées***`,ddof=1)`**

Poursuivons l'exemple précédent pour illustrer :

```
13 import numpy as np
14
15 moyenne = np.average(X)
16 sigma = np.std(X)
17 sx = np.std(X,ddof=1)
18
19 print(moyenne,sigma,sx)
```

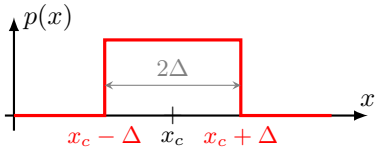
⇛ `11.23... 2.74... 2.78...`

N. Gaudouen

B Loi uniforme (ou rectangulaire)

Qu'est-ce que c'est ?

Une variable aléatoire X suit une loi rectangulaire de largeur 2Δ centré sur une valeur x_c lorsque la loi de probabilité $p(x)$ prend la forme ci-contre. Il y a donc équiprobabilité d'obtenir un tirage aléatoire dans l'intervalle $[x_c - \Delta, x_c + \Delta]$. En dehors de cet intervalle, la probabilité est nulle.



Tirage aléatoire avec Python

On aura chargé préalablement le module **random** de la bibliothèque **NumPy** avec l'instruction :

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

Si on connaît les valeurs $x_{min} = x_c - \Delta$ et $x_{max} = x_c + \Delta$, on utilisera l'instruction :

```
rd.uniform(xmin, xmax , N)
```

qui génère un tableau NumPy unidimensionnel de N valeurs tirées au sort de manière équiprobable dans l'intervalle $[x_{min}, x_{max}]$.

Si on connaît les valeurs x_c et Δ, on utilisera plutôt l'instruction :

```
rd.uniform(xc - Delta , xc + Delta, N)
```

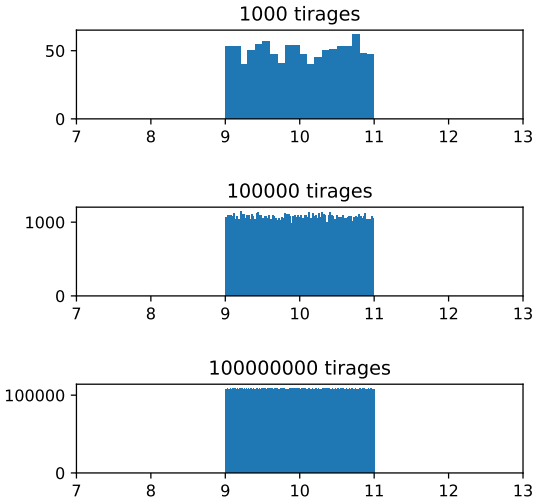
ou de manière équivalente :

```
xc + rd.uniform(-Delta , Delta, N)
```

En effet, `rd.uniform(-Delta , Delta, N)` génèrera un tableau NumPy de N valeurs dans $[-\Delta, \Delta]$. Donc, en additionnant x_c à ce tableau, on obtient bien des valeurs dans $[x_c - \Delta, x_c + \Delta]$.

Exemple

```
1  import matplotlib.pyplot as plt
2  import numpy.random as rd
3
4  N = 1000   # nombre de tirages
5  xmin = 9.
6  xmax = 11.
7
8  a = rd.uniform(xmin,xmax,N)
9
10 # ou bien, de manière équivalente :
11 xc = 10.
12 Delta = 1.
13 a = xc + rd.uniform(-Delta,Delta,N)
14
15
16 # tracé de l'histogramme
17 plt.hist(a, bins='rice')
18 plt.title("%i tirages" %N)
19 plt.show()
```



R *L'option* **`bins='rice'`** *permet d'optimiser le choix du nombre de barres à afficher.*

Poursuivons avec le calcul de la moyenne et de s_x :

```
20 import numpy as np
21
22 moyenne = np.average(a)
23 sx = np.std(a,ddof=1)
24
25 print("Pour",N,"tirages :")
26 print("moyenne =",moyenne)
27 print("sx =",sx)
```

⇛

```
Pour 1000 tirages :
moyenne = 9.982311156469073
sx = 0.5793309511215071
```

R *On retrouve logiquement une moyenne et un écart-type de l'échantillon proches respectivement de la valeur centrale x_c et de l'écart-type $\frac{\Delta}{\sqrt{3}} = 0,58$ attendu pour ce type de distribution.*

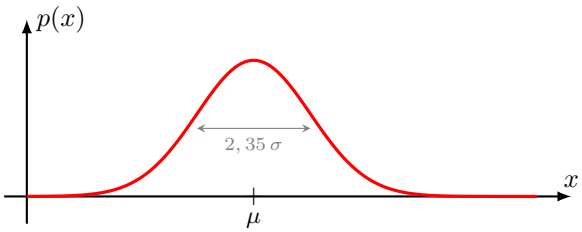
Cours

C Loi normale (ou gaussienne)

Qu'est-ce que c'est ?

Une variable aléatoire X suit une loi normale (ou gaussienne) d'espérance μ et d'écart-type σ lorsque la loi de probabilité s'écrit sous la forme :

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



R *La largeur à mi-hauteur vaut environ* $2,35\,\sigma$.

Tirage aléatoire avec Python

On aura chargé préalablement le module **random** de la bibliothèque **NumPy** avec l'instruction :

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

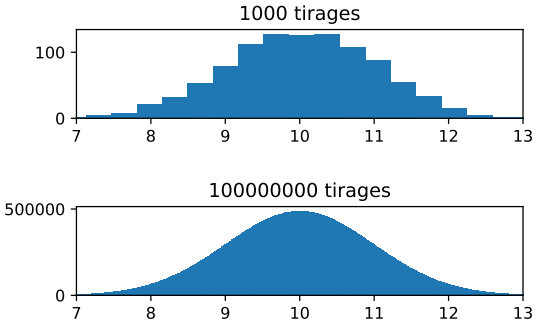
Puis, connaissant l'écart-type σ et l'espérance μ, on utilisera l'instruction :

```
rd.normal(mu, sigma, N)
```

permettant de générer un tableau NumPy unidimensionnel contenant N valeurs tirées au sort suivant la loi décrite précédemment.

Exemple

```
1  import matplotlib.pyplot as plt
2  import numpy.random as rd
3
4  N = 1000 # nombre de tirages
5  mu = 10.
6  sigma = 1.
7
8  b = rd.normal(mu,sigma,N)
9
10 # tracé de l'histogramme
11 plt.hist(b,bins='rice')
12 plt.title("%i tirages" %N)
13 plt.show()
```



R *L'option* ***bins='rice'*** *permet d'optimiser le choix du nombre de barres à afficher.*

Poursuivons avec le calcul de la moyenne et de s_x :

```
14 import numpy as np
15
16 moyenne = np.average(b)
17 sx = np.std(b,ddof=1)
18
19 print("Pour",N,"tirages :")
20 print("moyenne =",moyenne)
21 print("sx =",sx)
```

⇛

```
Pour 1000 tirages :
moyenne = 9.945380010087232
sx = 0.9693775291004446
```

R *On retrouve logiquement une moyenne et un écart-type de l'échantillon proches respectivement de la valeur de l'espérance* μ *et de l'écart-type* σ *choisie.*

N. Gaudouen