TP14 : Simulation des lois à densité usuelles, convergence en loi et Théorème central limite

Commencer par importer les bibliothèques suivantes dans chaque fichier **Python** utilisé :

```
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt
```

On rappelle que l'on a alors accès aux commandes suivantes :

- `rd.random()`
  ↪ génère un nombre aléatoire dans $[0,1]$, *i.e.* simule une v.a.r. $X \hookrightarrow \mathcal{U}([0,1])$
- `rd.random(d)`
  ↪ génère d nombres aléatoires dans $[0,1]$
- `rd.random([n,p])`
  ↪ génère un tableau à n lignes et p colonnes dont chaque case est un nombre aléatoire dans $[0,1]$
- `rd.uniform(a,b,d)`
  ↪ simule d fois une v.a.r. $X \hookrightarrow \mathcal{U}([a,b])$ (le paramètre d est optionnel)
- `rd.exponential(a,d)`
  ↪ simule d fois une v.a.r. $X \hookrightarrow \mathcal{E}\left(\frac{1}{a}\right)$ (le paramètre d est optionnel)
- `rd.normal(m,s,d)`
  ↪ simule d fois une v.a.r. $X \hookrightarrow \mathcal{N}(m,s)$ (le paramètre d est optionnel)

et

- `rd.binomial(n,p,d)`
  ↪ simule d fois de manière indépendante une v.a.r. $X \hookrightarrow \mathcal{B}(n,p)$ (le paramètre d est optionnel)
  ↪ en choisissant $n=1$, simule d fois de manière indépendante une v.a.r. $X \hookrightarrow \mathcal{B}(p)$
- `rd.geometric(p,d)`
  ↪ simule d de manière indépendante fois une v.a.r. $X \hookrightarrow \mathcal{G}(p)$ (le paramètre d est optionnel)
- `rd.poisson(a,p)`
  ↪ simule d fois de manière indépendante une v.a.r. $X \hookrightarrow \mathcal{P}(a)$ (le paramètre d est optionnel)
- `rd.randint(a,b,d)`
  ↪ simule d fois de manière indépendante une v.a.r. $X \hookrightarrow \mathcal{U}(\llbracket a, b-1 \rrbracket)$ (le paramètre d est optionnel)

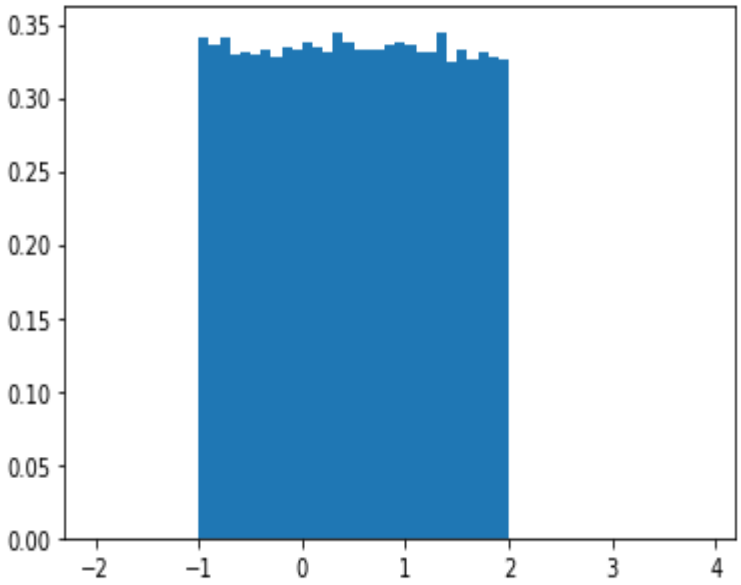
I. Lien entre histogramme des fréquences et densité

I.1. Simulation de la loi uniforme sur $[a, b]$

► Dessiner le graphe de la densité usuelle d'une v.a.r. $X \hookrightarrow \mathcal{U}([-1, 2])$.

► On représente le tracé obtenu en compilant le code suivant.

```
1 plt.hist(rd.uniform(-1,2,100000),np.arange(-2,4,0.1),density=True)
```



Qu'y a-t-il de remarquable ?

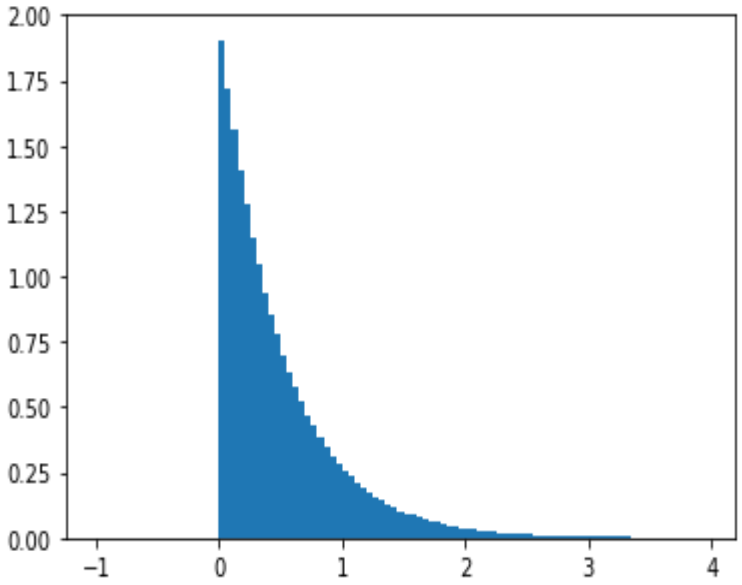
On retrouve la forme de la densité usuelle dessinée ci-dessus (comme si on avait colorié l'aire sous la courbe). Ainsi, on retiendra que l'histogramme des fréquences d'une v.a.r. à densité nous renseigne sur la forme des densités de cette v.a.r. .

I.2. Simulation de la loi exponentielle

▶ Dessiner le graphe de la densité usuelle d'une v.a.r. $X \hookrightarrow \mathcal{E}(2)$.

▶ On représente le tracé obtenu en compilant le code suivant.

```
1 plt.hist(rd.exponential(1/2,1000000),np.arange(-1,4,0.05),density=True)
```



Qu'y a-t-il de remarquable ?

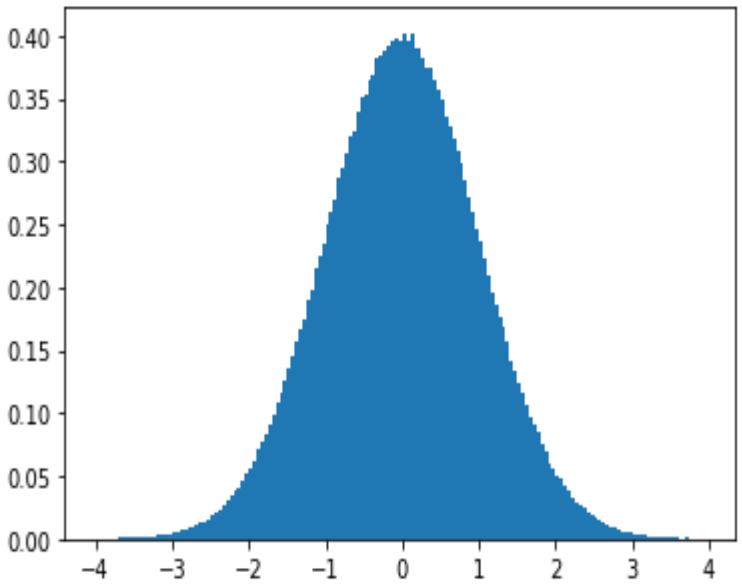
Comme dans la partie précédente, on retrouve la forme de la densité usuelle dessinée ci-dessus (comme si on avait colorié l'aire sous la courbe).

I.3. Simulation de la loi normale centrée réduite

- Dessiner le graphe de la densité usuelle d'une v.a.r. $X \hookrightarrow \mathcal{N}(0,1)$.

- On représente le tracé obtenu en compilant le code suivant.

```
1 plt.hist(rd.normal(0,1,1000000),np.arange(-4,4,0.05),density=True)
```



Qu'y a-t-il de remarquable ?

Comme dans la partie précédente, on retrouve la forme de la densité usuelle dessinée ci-dessus (comme si on avait colorié l'aire sous la courbe).

I.4. Visualisation d'une convergence en loi via les histogrammes de fréquences

Soit (U_n) une suite de v.a.r. indépendantes telle que, pour tout $n \in \mathbb{N}^*$, $U_n \hookrightarrow \mathcal{U}([0,1])$.
Pour tout $n \in \mathbb{N}^*$, on note $M_n = \max(U_1, \ldots, U_n)$ et $X_n = n(1 - M_n)$.

► Compléter la fonction suivante pour qu'elle renvoie une simulation de X_n.

```
def SimulX(n) :
    U = __rd.random(n)__
    M = __max(U)__
    return n*(1-M)
```

```
def SimulX(n) :
    U = rd.random(n)
    M = max(U)
    return n*(1-M)
```

► Compléter la fonction suivante pour qu'elle renvoie un tableau contenant N simulations de X_n.

```
def SimulTabX(n,N):
    T = __np.zeros(N)__
    for k in range(N):
        T[k] = __SimulX(n)__
    return T
```

```
def SimulTabX(n,N):
    T = np.zeros(N)
    for k in range(N):
        T[k] = SimulX(n)
    return T
```

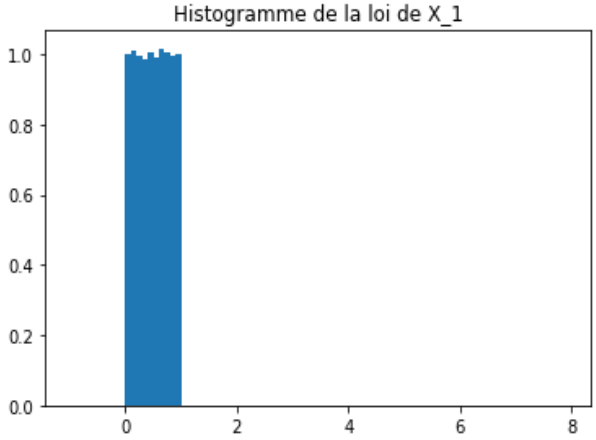
► Expliquer à quoi sert le script suivant.

```
for n in [1,2,3,4,10,20,40,100]:
    plt.hist(SimulTabX(n,100000),np.arange(-1,8,0.1),density=True)
    plt.title(f'Histogramme de la loi de X_{n}')
    plt.show()
```

Ce script permet d'afficher les histogrammes de fréquences des lois de X_n pour différentes valeurs de n, plus précisément pour $n \in \{1, 2, 3, 4, 10, 20, 40, 100\}$.

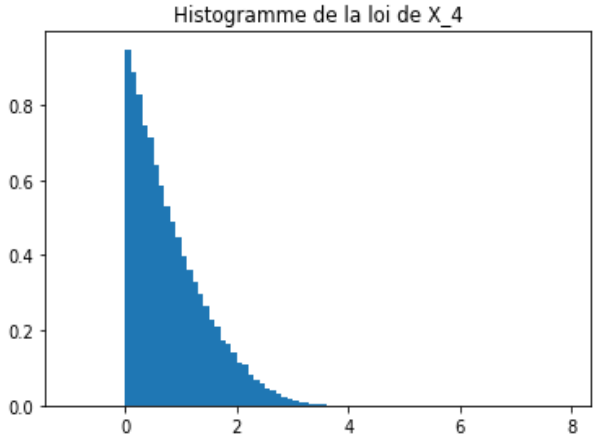
► On reproduit page suivante les résultats obtenus à l'aide du script précédent. Que peut-on conjecturer ? La convergence semble-t-elle rapide ?

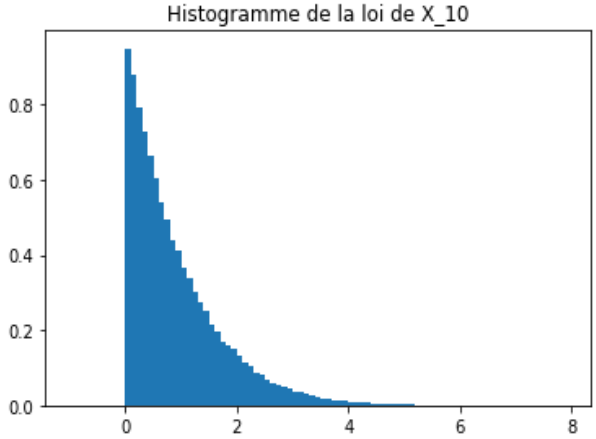
On conjecture que $X_n \xrightarrow[n\to+\infty]{\mathscr{L}} X$ où $X \hookrightarrow \mathcal{E}(1)$ et la convergence semble rapide (les histogrammes ne changent que très peu à partir de $n = 10$).

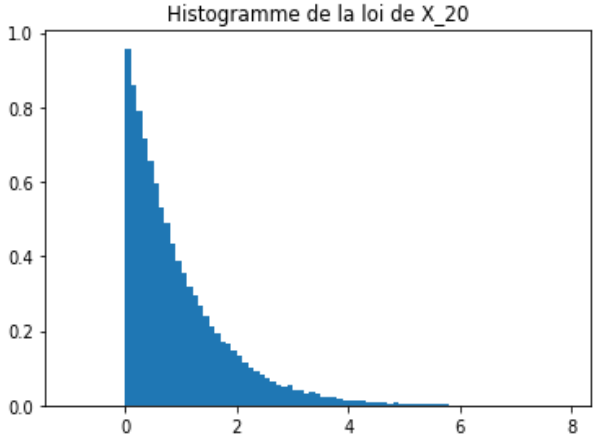

Histogramme de la loi de X_1
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8

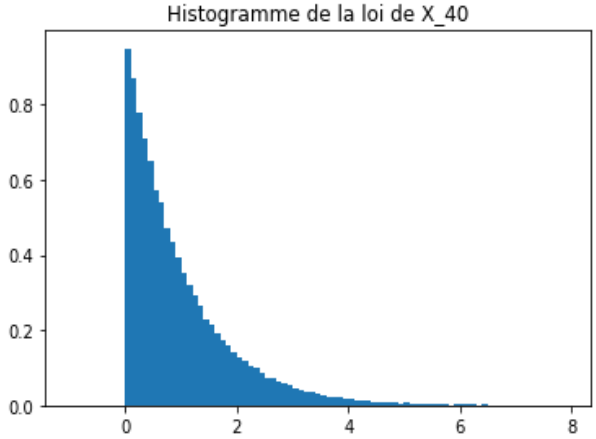

Histogramme de la loi de X_2
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8

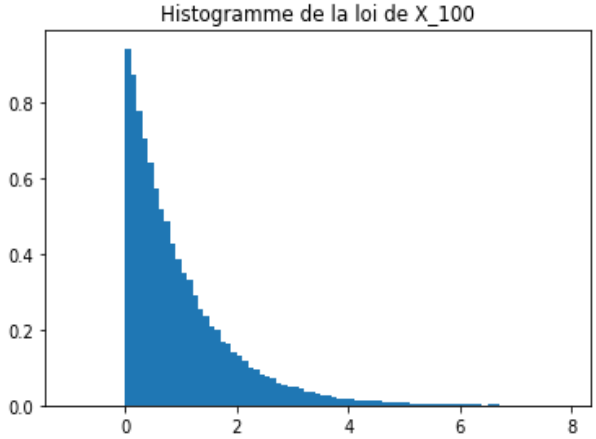

Histogramme de la loi de X_3
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8


Histogramme de la loi de X_4
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8


Histogramme de la loi de X_10
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8


Histogramme de la loi de X_20
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8


Histogramme de la loi de X_40
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8


Histogramme de la loi de X_100
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8

II. Théorème central limite

Le cadre du TCL permet de simuler une v.a.r. suivant la loi normale centrée réduite à l'aide d'autres lois usuelles. Dans la suite, on compare la densité de probabilité théorique avec celle obtenue par simulation. Pour ce faire, on procède comme suit :

× on produit `N` observations de la simulation de $S_n{}^*$,

× on trace l'histogramme des fréquences associés.

Il reste à préciser quelle loi doit suivre les variables X_k. On s'intéresse dans la suite aux cas suivants :

1) $X_k \hookrightarrow \mathcal{U}([0,1])$, *2)* $X_k \hookrightarrow \mathcal{B}(p)$, *3)* $X_k \hookrightarrow \mathcal{P}(a)$.

II.1. Simulation de la loi normale centrée réduite via les « 12 uniformes »

Dans le TCL, la convergence peut être très rapide. La loi de la variable $S_n{}^*$ est alors une bonne approximation de la loi $\mathcal{N}(0,1)$ pour des valeurs de n assez petites.
Dans ce paragraphe :

× on choisit $n = 12$.

× on considère des variables $X_1, \ldots, X_{12}$ indépendantes qui suivent toutes la loi $\mathcal{U}([0,1])$.

× on simule la v.a.r. ${S_{12}}^*$. Plus précisément, on représente l'histogramme des fréquences issu de N observations de ${S_{12}}^*$ et on compare ce résultat à la densité de la loi $\mathcal{N}(0,1)$.

► Que vaut ${S_{12}}^*$? En donner une expression simple à l'aide de $S_{12} = \sum_{k=1}^{12} X_k$.

$$ {S_{12}}^* = \frac{S_{12} - 12\, m}{\sigma\, \sqrt{12}} = \frac{S_{12} - 12\, \frac{1}{2}}{\frac{1}{\sqrt{12}}\, \sqrt{12}} = S_{12} - 6 $$

► Par quel appel simule-t-on un échantillon de 12 v.a.r. indépendantes suivant la loi $\mathcal{U}([0,1])$?
En déduire une simulation de la v.a.r. S_{12}.

`rd.random(12)` permet de simuler l'échantillon. Ainsi, `sum(rd.random(12))` simule S_{12}.

► Compléter la fonction suivante pour qu'elle renvoie un tableau contenant `N` simulations de S_{12}^*.

```
1  def SimulSCRUni(N):
2      T = __np.zeros(N)__
3      for k in range(N):
4          T[k] = __sum(rd.random(12)) - 6__
5      return T
```

```
1  def SimulSCRUni(N):
2      T = np.zeros(N)
3      for k in range(N):
4          T[k] = sum(rd.random(12)) - 6
5      return T
```

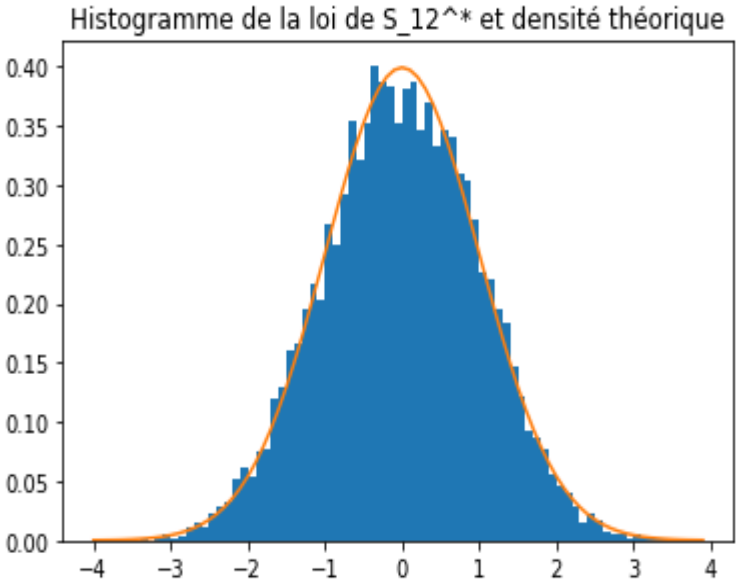
- Compléter la fonction suivante pour qu'elle renvoie $\varphi(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2}$.

```
def phi(t):
    return ___(1/np.sqrt(2*np.pi))*np.exp(-t**2/2)___
```

```
def phi(t):
    return (1/np.sqrt(2*np.pi))*np.exp(-t**2/2)
```

- On représente le tracé obtenu en compilant le code suivant.

```
plt.hist(SimulSCRUni(10000),np.arange(-4,4,0.1),density=True)
plt.plot(np.arange(-4,4,0.1),phi(np.arange(-4,4,0.1)))
plt.title('Histogramme de la loi de S_12^* et densité théorique')
plt.show()
```



Commenter.

L'histogramme des fréquences épouse correctement le graphe de la densité théorique φ. C'est remarquable pour une si petite valeur de n ($n = 12$ ici). La convergence du TCL est très rapide dans ce cas.

II.2. Simulation de la loi normale centrée réduite via la loi de Bernoulli

Dans ce paragraphe :

- × on choisit $p = \frac{1}{2}$.
- × on considère des variables $X_1, \ldots, X_n$ indépendantes qui suivent toutes la loi $\mathcal{B}(p)$.
- × on simule la v.a.r. ${S_n}^*$. Plus précisément, on représente l'histogramme des fréquences issu de N observations de ${S_n}^*$ et on compare ce résultat à la densité de la loi $\mathcal{N}(0,1)$.

- Rappeler la loi de $S_n = \sum_{k=0}^{n} X_k$, son espérance et sa variance.

$S_n \hookrightarrow \mathcal{B}\left(n, \frac{1}{2}\right)$, $\mathbb{E}(S_n) = \frac{1}{2}n$ et $\mathbb{V}(S_n) = \frac{1}{4}n$

- Que vaut ${S_n}^*$? En donner une expression simple à l'aide de S_n.

$${S_n}^* = \frac{S_n - \mathbb{E}(S_n)}{\sigma(S_n)} = \frac{S_n - \frac{1}{2}n}{\frac{1}{2}\sqrt{n}} = \frac{2S_n - n}{\sqrt{n}}$$

- Par quel appel simule-t-on une v.a.r. $Y \hookrightarrow \mathcal{B}\left(n, \frac{1}{2}\right)$?

`rd.binomial(n,1/2)` permet de simuler $Y \hookrightarrow \mathcal{B}\left(n, \frac{1}{2}\right)$.

- Compléter la fonction suivante pour qu'elle renvoie un tableau contenant `N` simulations de S_n^*.

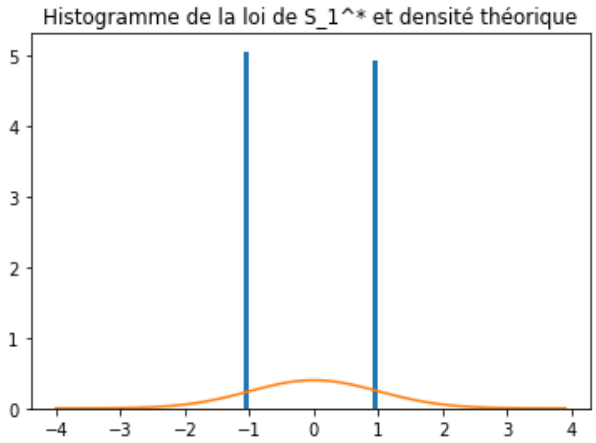
```
1  def SimulSCRBern(n,N):
2      T = __np.zeros(N)__
3      for k in range(N):
4          T[k] = ___(2*rd.binomial(n,1/2) - n)/np.sqrt(n)___
5      return T
```

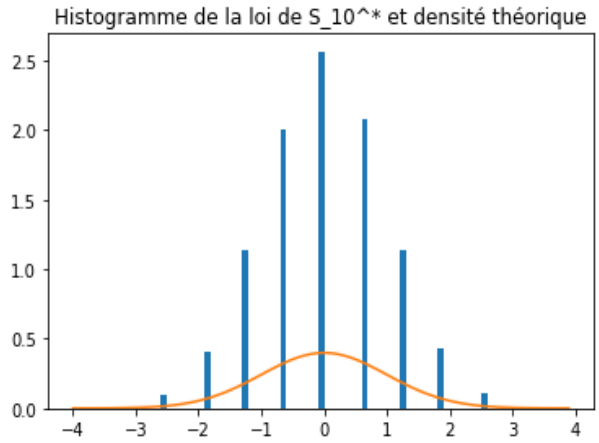
```
1  def SimulSCRBern(N):
2      T = np.zeros(N)
3      for k in range(N):
4          T[k] = (2*rd.binomial(n,1/2) - n)/np.sqrt(n)
5      return T
```

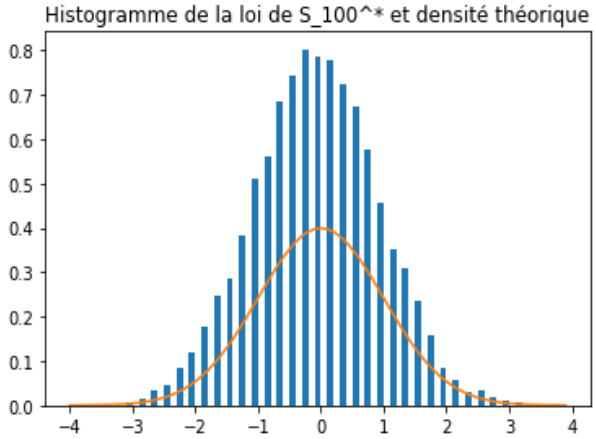
- On reproduit page suivante les résultats obtenus à l'aide du script suivant.
Comparer la vitesse de convergence avec celle de l'exemple précédent.

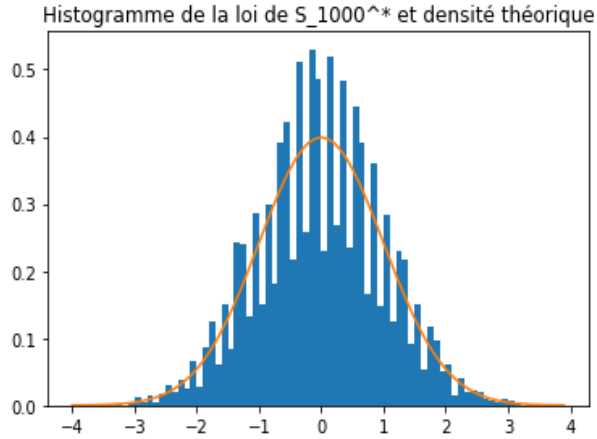
```
1  for k in range(8):
2      plt.hist(SimulSCRBern(10**k,10000),np.arange(-4,4,0.1),density=True)
3      plt.plot(np.arange(-4,4,0.1),phi(np.arange(-4,4,0.1)))
4      plt.title(f'Histogramme de la loi de S_{10**k}^* et densité théorique')
5      plt.show()
```

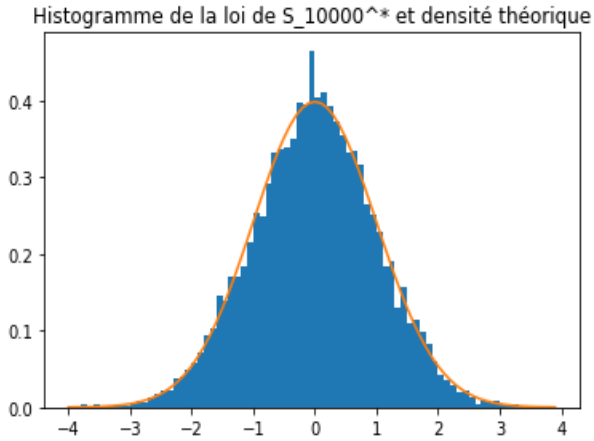
La convergence est lente comparée à l'exemple précédent. Il faut attendre $n = 10000$ pour avoir une loi ressemblant à la loi normale centrée réduite.

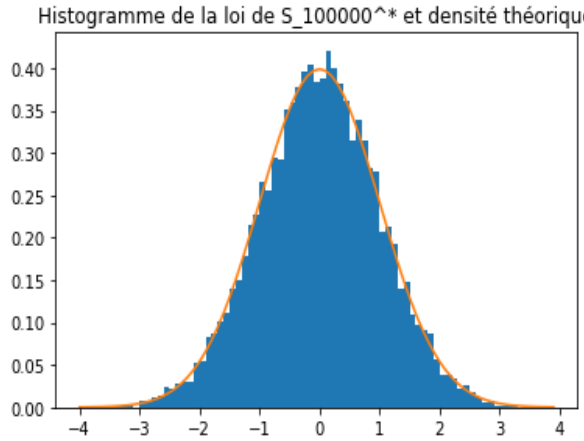

Histogramme de la loi de S_1^* et densité théorique

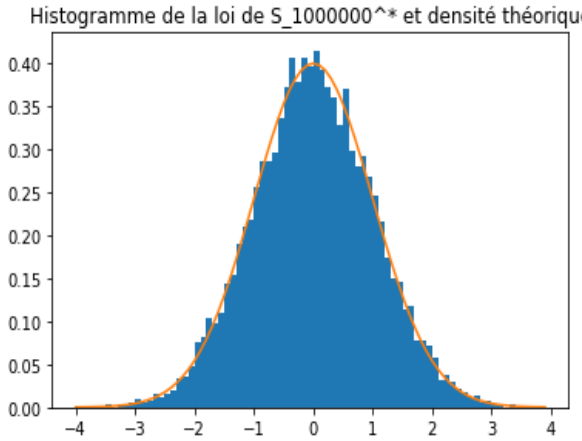

Histogramme de la loi de S_10^* et densité théorique

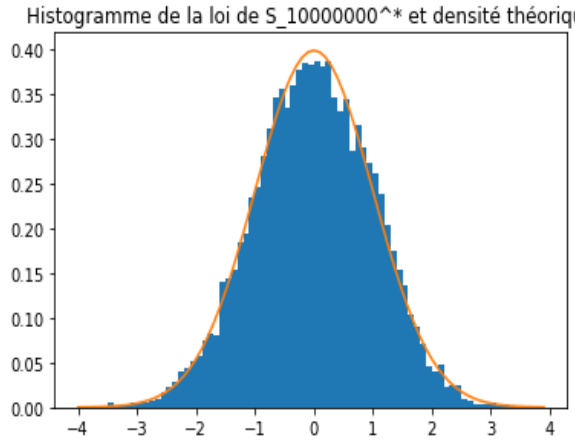

Histogramme de la loi de S_100^* et densité théorique


Histogramme de la loi de S_1000^* et densité théorique


Histogramme de la loi de S_10000^* et densité théorique


Histogramme de la loi de S_100000^* et densité théorique


Histogramme de la loi de S_1000000^* et densité théorique


Histogramme de la loi de S_10000000^* et densité théorique

II.3. Simulation de la loi normale centrée réduite via la loi de Poisson

Dans ce paragraphe :

× on choisit $a = 5$.

× on considère des variables $X_1, \dots, X_n$ indépendantes qui suivent toutes la loi $\mathcal{P}(a)$.

× on simule la v.a.r. ${S_n}^*$. Plus précisément, on représente l'histogramme des fréquences issu de N observations de ${S_n}^*$ et on compare ce résultat à la densité de la loi $\mathcal{N}(0,1)$.

▶ Rappeler la loi de S_n, son espérance et sa variance.

$S_n \hookrightarrow \mathcal{P}(5n)$, $\mathbb{E}(S_n) = 5n$ et $\mathbb{V}(S_n) = 5n$

▶ Que vaut ${S_n}^*$? En donner une expression simple à l'aide de S_n.

$${S_n}^* = \frac{S_n - \mathbb{E}(S_n)}{\sigma(S_n)} = \frac{S_n - 5n}{\sqrt{5n}}$$

▶ Par quel appel simule-t-on une v.a.r. $Y \hookrightarrow \mathcal{P}(5n)$?

`rd.poisson(5n)` permet de simuler $Y \hookrightarrow \mathcal{P}(5n)$.

▶ Compléter la fonction suivante pour qu'elle renvoie un tableau contenant `N` simulations de S_n^*.

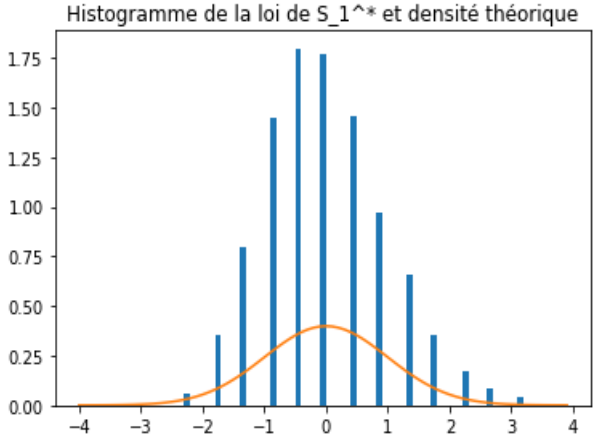
```
1  def SimulSCRPois(n,N):
2      T = __np.zeros(N)__
3      for k in range(N):
4          T[k] = ___(rd.poisson(5*n)-5*n)/np.sqrt(5*n)___
5      return T
```

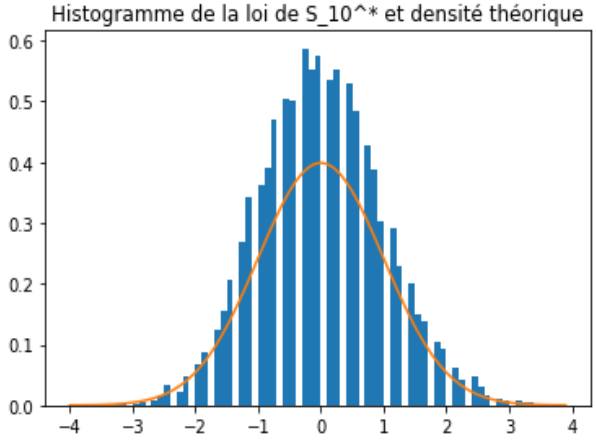
```
1  def SimulSCRPois(N):
2      T = np.zeros(N)
3      for k in range(N):
4          T[k] = (rd.poisson(5*n)-5*n)/np.sqrt(5*n)
5      return T
```

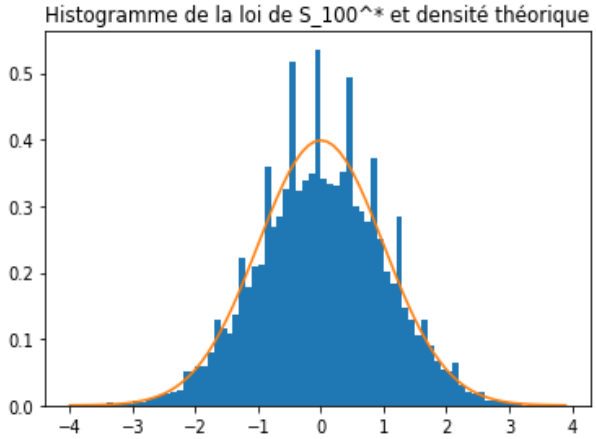
▶ On reproduit page suivante les résultats obtenus à l'aide du script suivant.
Comparer la vitesse de convergence avec celle de l'exemple précédent.

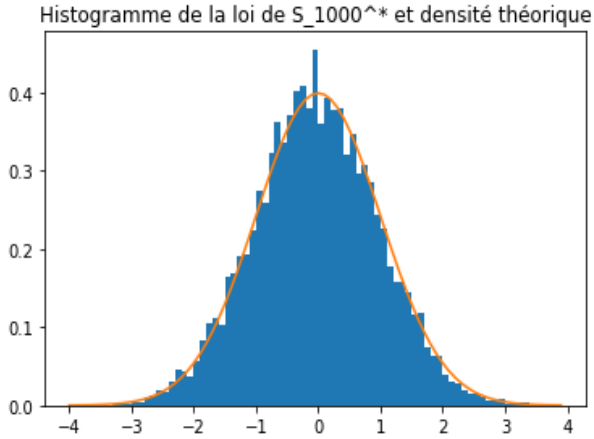
```
1  for k in range(8):
2      plt.hist(SimulSCRPois(10**k,10000),np.arange(-4,4,0.1),density=True)
3      plt.plot(np.arange(-4,4,0.1),phi(np.arange(-4,4,0.1)))
4      plt.title(f'Histogramme de la loi de S_{10**k}^* et densité théorique')
5      plt.show()
```

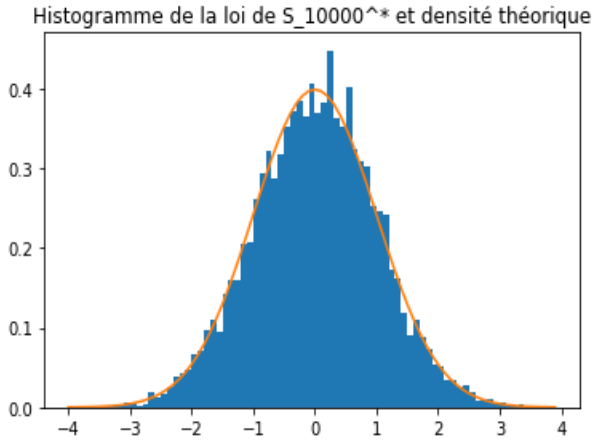
La convergence est un peu plus rapide que dans l'exemple précédent, mais reste beaucoup plus lente que dans le cas des « 12 uniformes ».

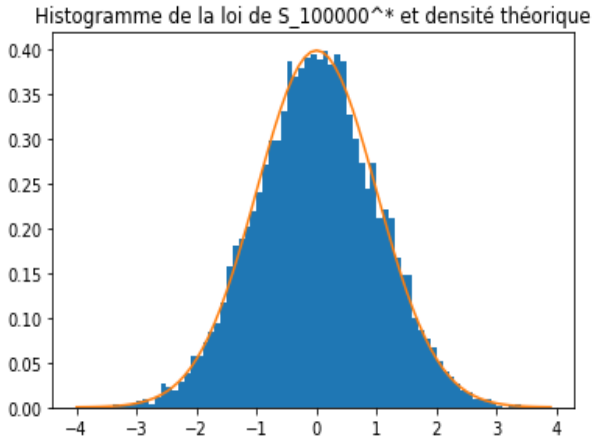

Histogramme de la loi de S_1^* et densité théorique

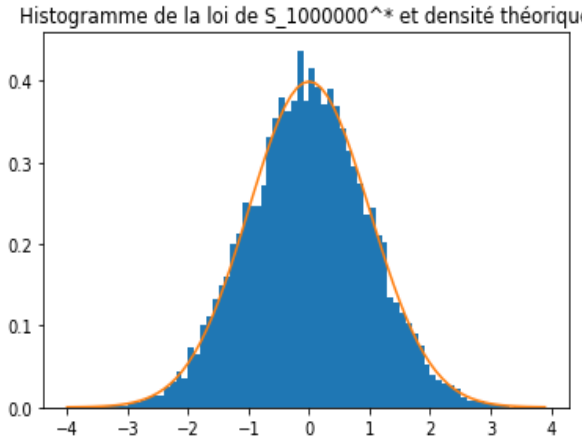

Histogramme de la loi de S_10^* et densité théorique

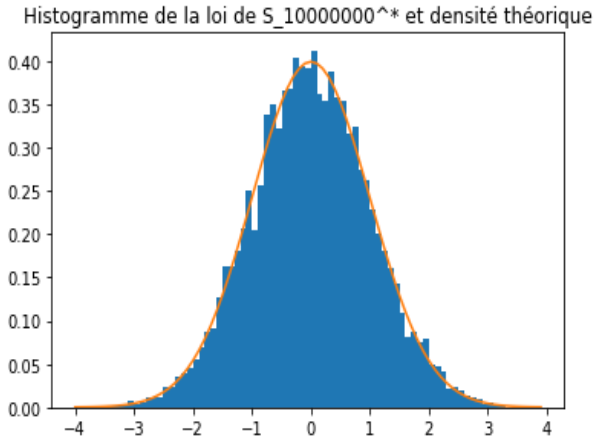

Histogramme de la loi de S_100^* et densité théorique


Histogramme de la loi de S_1000^* et densité théorique


Histogramme de la loi de S_10000^* et densité théorique


Histogramme de la loi de S_100000^* et densité théorique


Histogramme de la loi de S_1000000^* et densité théorique


Histogramme de la loi de S_10000000^* et densité théorique