.

Thème 3 : Théorème *Central Limit*

Avant-Propos : Conformément au programme officiel, ce thème du programme a pour objectif de mettre en évidence le théorème dit *limite central* (ou *central limit*) ainsi que ses conséquences.
Le théorème en lui-même n'est pas au programme mais on pourra l'énoncer (ceci sera fait en seconde partie du TP).

Pensez à charger les bibliothèques permettant de réaliser des simulations aléatoires (`numpy.random`) ainsi que la visualisation d'histogrammes ou de courbes (`mathplotlib`) : les TP de ce thème utiliseront donc les bibliothèques d'importation suivantes :

```
from math import*
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt
```

Distribution des moyennes empiriques

L'objectif de cette partie est d'observer le comportement commun aux observations de *moyennes empiriques calculées sur des réalisations de phénomènes aléatoires suivant une même loi* indépendamment du choix initial de la-dite loi.
Les lois étudiées ne sont pas forcément usuelles mais resteront discrètes

Aux dés : un exemple usuel

On se donne la commande `rd.randint` pour réaliser les simulations de lancés de dés usuels (6 faces, non truqués) pour cette première section.

1. On séchauffe : générez à l'aide de Python quelques lancés de dés (5 à 10) puis observez l'histogramme de répartition.
2. Calculer la moyenne *empirique* de vos résultats.
3. On va créer une fonction renvoyant la moyenne empirique selon le nombre demandé de lancés :

```
def empirmeande(n):
        X=0
        for k in range(n):
        X = rd.randint(1,6)+X
     X=X/n
     return(X)
```

   Ecrire la définition formelle de la variable X fournie en sortie par ce programme, en fonction de n et de $U_1, \dots, \dots U_n$ les VAR générées par l'instruction `rd.randint`$(1, 6)$
4. Quelle est la loi commune aux VAR générées par l'instruction `rd.randint`$(1, 6)$ dans ce programme ?
   En rappeler l'espérance μ et l'écart-type σ.
5. On fixe $n = 100$
   (a) Créer une liste L contenant N =250 simulations de `empirmeande`$(\mathtt{n})$
   (b) En utilisant la commande `plt.hist`$(\mathtt{L}, \mathtt{range} = (0, 6), \mathtt{bins} = 6, \mathtt{color} =' \mathtt{blue}', \mathtt{edgecolor} =' \mathtt{black}')$ observez l'histogramme de retour. Faites varier le paramètre `bins` $= 6$. Que constatez-vous ?
   (c) Observer l'évolution du graphique pour $N = 500$, $N = 1000$ puis $N = 10\,000$ en adapatant le choix de `bins`.
   Une forme semble-t-elle se dégager pour l'histogramme ?

6. On fixe $n = 1000$
   (a) Créer une liste L contenant N =2500 simulations de `empirmeande(n)`
   (b) En utilisant la commande $\texttt{plt.hist}(\texttt{L}, \texttt{range} = (0, 6), \texttt{bins} = 6, \texttt{color} =' \texttt{blue}', \texttt{edgecolor} =' \texttt{black}')$ observez l'histogramme de retour.
   (c) Observer l'évolution du graphique pour $N = 5000$, $N = 10000$ puis $N = 100\ 000$. Une forme semble-t-elle se dégager pour l'histogramme ?

Tickets à gratter : version simplifiée

On se donne la commande `rd.random` pour réaliser les simulations la loi uniforme sur $[0; 1[$ (cas à densité).
On considère un ticket à gratter dont le tableau des gains G (après déduction du prix) est donné (en euros) par :

valeur	-2	0	+2	+ 8	+20
probabilité	69/90	7/90	9/90	4/90	1 /90
répartition	69/90	76/90	...	...	...

NB : ce tableau est formé à partir des données d'un véritable ticket, à l'exception de la dernière colonne que l'on a cherché à rendre la plus proche possible de la réalité en résumant les gains supérieurs à 10 euros en une seul donnée la plus conforme (et lisse) possible.

1. Ne constatez-vous pas un élément paradoxal dans la répartiton des gains ?
   Compléter la dernière ligne du tableau (c'est F_G, la fonction de répartition de G en quelques valeurs pertinentes)
2. Déterminer $\mathbb{E}[G]$ et $\sigma(G)$ où G est la variable aléatoire associée à ce gain.
3. Compléter le programme suivant pour qu'il simule une réalisation de G :

```
def Gtick():
        U=np.random()
        if U<= 69/90:
                G=-2
        else :
                if U<=76/90:
                        ..........
.
.
.
.
.
.
.
.
        return(G)
```

On considère que l'on va étudier le comportement moyen des gains de tickets achetés en masse. On considérera ces tickets indépendants et de même nature (soit de même loi de gain que G).
En retirant les "gros lots", cette hypothèse est plus viable bien que pas tout à fait exacte en pratique

4. On va créer une fonction renvoyant la moyenne empirique selon le nombre demandé de tickets :

```
def empirmeanG(n):
        X=0
        for k in range(n):
        X = Gtick()+X
     X=X/n
     return(X)
```

Ecrire la définition formelle de la variable X fournie en sortie par ce programme, en fonction de n et de $U_1, \dots, \dots U_n$ les VAR générées par l'instruction `rd.randint(1, 6)`

5. On fixe $n = 9000$
   (a) Créer une liste L contenant N =180 000 simulations de `empirmeande(n)`
   (b) En utilisant la commande `plt.hist(L, range` $= (-2, 20)$, `bins` $= 22$, `color` $=$ `'blue'`, `edgecolor` $=$ `'black'`) observez l'histogramme de retour. Faites ensuite varier le paramètre `bins`
   (c) Observer l'évolution du graphique pour $N = 500$, $N = 1000$ puis $N = 10\,000$. Noubliez pas de modifier `bins`
6. Reprenez les représentations qui précèdent avec $n = 90\,000$ et adaptez N en conséquence.
   On espére juste que les machines vont tenir le coup : ne soyez ni trop gourmand ni trop pressés

Un premier bilan

Dans les protocoles précédents, il semble se dégager une forme commune.
Nous allons chercher à l'étudier à partir des variables *centrées-réduites*, donc sans tenir compte de l'échelle des valeurs obtenues par les simulations.

1. Pour X une VAR quelconque, rappeler la définition de X^*, variable centrée-réduite associée à X.
2. Donner l'expression de G^* (associée aux tickets à gratter) ainsi que de U_1^* (lancé du premier dé)
3. Représenter les histogrammes associés aux variables aléatoires $\overline{G}_n^*\sqrt{n}$ et $\overline{U}_n^*\sqrt{n}$ où l'on donne :

$$\overline{G}_n^* = \frac{G_1^* + G_2^* + \cdots + G_n^*}{n} \qquad \text{et} \qquad \overline{U}_n^* = \frac{U_1^* + U_2^* + \cdots + U_n^*}{n}$$

   avec $N = 100\,000$ réalisations pour $n = 900$ puis $n = 4500$.
4. Soit φ définie sur $\mathbb{R}$ par : $\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$. Compléter le script suivant pour représenter graphiquement f :

```
def phi(x):
        y = (1/(.....))*np.exp(....)
    return(y)
X=linspace(-5,5,1000)
Y=phi(X)
plot(........)
plt.show()
```

   En cas de problème technique, remplacer `Y = phi(X)` *par une boucle* `for` *et* `Y[k] = phi(X[k])`
5. Comparer les aspects de la courbe et de vos histogrammes. Une remarque ?

Théorème : [Central Limit]

(connaissance non exigible)
On considère $(X_k)_{k\in\mathbb{N}}$ une suite de VAR mutuellement indépendantes, de même loi admettant une espérance μ et une variance σ^2. On note $\overline{X}_n = \dfrac{X_1 + \cdots + X_n}{n}$ pour tout $n \in \mathbb{N}^*$ et on désigne par $\overline{X}_n^* = \dfrac{\overline{X}_n - \mu}{\sigma}$ la variable aléatoire centrée-réduite associée. Alors on a :

$$\forall x \in \mathbb{R} \quad \lim_{n\to+\infty} \mathbb{P}[\overline{X}_n^*\sqrt{n} \leq x] = \int_{-\infty}^{x} \varphi(t)\,\mathrm{d}t = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}\,\mathrm{d}t$$

Loi de la moyenne empirique $\overline{X}_n$

On se donne la commande `rd.random` pour réaliser les simulations la loi uniforme sur $[0; 1[$ (cas à densité).
Conformément aux attentes du programme officiel, on va tester directement le protocole effectué dans les séances précédentes sur X de loi $\mathcal{U}[0; 1[$.

Vocabulaire :

La moyenne empirique $\overline{X}_n$ des n variables aléatoires $X_1, X_2, \ldots, X_n$ est la variable aléatoire définie comme :

$$\overline{X}_n = \frac{1}{n}\sum_{k=1}^{n} X_k = \frac{X_1 + \cdots + X_n}{n}$$

En pratique, on l'utilise dans le cas où les variables aléatoires $X_1, X_2, \ldots, X_n$ sont de même loi et en supposant qu'elle sont mutuellement indépendantes.

1. En vous aidant des sections précédentes, définir une fonction python `rdempirik(n)` qui prend n, entier en entrée et renvoie la moyenne empirique de n réalisations de lois $\mathcal{U}[0;1[$ obtenues avec la commande `rd.random`.
2. Déterminer les valeurs de $\mathbb{E}\left[\overline{X}_n\right]$ et $\sigma\left(\overline{X}_n\right)$ en fonction de $n \in \mathbb{N}^*$.
3. En déduire une expression (formelle) de $\overline{X}_n^*$, variable centrée-réduite associée à $\overline{X}_n$.
4. Compléter le programme suivant pour qu'il crée un vecteur-ligne L_N de N réalisations de $\overline{X}_n$ lorsque n et N sont donnés en entrée :

```
def simuL(N,n):
        L=np.zeros(N)
        for k in rang(N):
                X=rdempirik(n)
                X=.....................
                L[k] = X
        return(L)
```

5. Créer un histogramme associé à `simuL(N,n)` pour $N = 1000$ et $n = 100$.
6. Procéder à d'autres tests en comparant la forme de l'histogramme avec la forme de la courbe de la fonction φ définie dans la section **premier bilan**.
   On cherchera à placer la courbe et l'histogramme dans une même fenêtre graphique
   Attention ! L'histogramme renvoie des *effectifs* en ordonnées et φ n'est pas sur la même échelle de mesure. Il faudra donc adapter l'un à l'autre.
7. Quelles sont vos observations ?
8. Comparer vos histogrammes avec des exécutions du script suivant :

```
def normali(N):
    X = np.zeros([N])
    for k in range(N):
        X[k] = rd.normal()
    Gu = plt.hist(X, range = (-5,5), bins = 1000)
    return(Gu)
Gu=normali(100000)
plt.show()
```

Vous chercherez à choisir vos paramètres en concordance avec le script fourni.