TP11 - Illustration des théorèmes limites

1 Loi gaussienne ou loi normale centrée réduite

Définition 1.0.1
Une variable aléatoire X suit une loi gaussienne ou loi normale centrée réduite si $\forall (a,b) \in \mathbb{R}^2,\ a \leq b$,

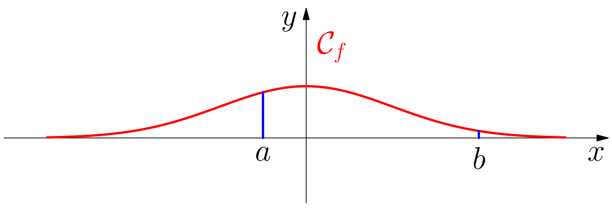
$$\boldsymbol{P(a \leq X < b) = \int_a^b f(x)\mathrm{d}x}$$

où la fonction f est définie par :

$$\forall x \in \mathbb{R},\ \boldsymbol{f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}}.$$

On le note :

$$\boldsymbol{X \hookrightarrow \mathcal{N}(0,1)}.$$



En Python, pour simuler une réalisation d'une telle variable aléatoire, on utilise la fonction `gauss` du module `random` :

```python
import random as rd
rd.gauss(0,1)
```

On cherche ici à mieux comprendre la loi normale. On considère $X \hookrightarrow \mathcal{N}(0,1)$. On va approcher la valeur de $P(1 \leq X < 2.35)$ de trois manières différentes.

1. Comme pour les variables aléatoires discrètes, $P(a \leq X < b) = F_X(b) - F_X(a)$ où F_X est la fonction de répartition de X. Cette fonction de répartition est tabulée (cf *TableNormale.png* sur CDP). À l'aide de cette table, donner une valeur approchée de $P(1 \leq X < 2.35)$.
2. Pour approcher la valeur d'une probabilité, on peut réaliser de nombreuses fois l'expérience aléatoire et calculer la fréquence de succès. Compléter le programme suivant pour calculer une valeur approchée de $P(1 \leq X < 2.35)$.

```python
import random as rd
N=100000
succes=...
for k in range(...):
    x=...
    if 1<=x<2.35:
        succes+=1
p2=...
```

3. On peut aussi estimer cette probabilité en approchant la valeur de $\int_1^{2.35} f(x)\mathrm{d}x$.
   (a) Définir la fonction `f` en Python.
   (b) Compléter le programme suivant pour approcher la valeur de cette intégrale par la méthode des rectangles à gauche.

```python
N=10000
h=...
x=...
I=...
for k in range(N):
    I+=h*f(x)
    x+=...
p3=I
```

2 Illustration numérique du théorème central limite

2.1 Le théorème central limite

Définition 2.1.1 Soit X une variable aléatoire d'espérance μ et de variance σ^2.
Soient $n \in \mathbb{N}^*$ et $(X_1, \dots, X_n)$ des variables aléatoires mutuellement indépendantes de même loi que X.
La **moyenne empirique $\boldsymbol{M_n}$** de cet échantillon est la variable aléatoire :

$$\boldsymbol{M_n = \frac{1}{n} \sum_{k=1}^{n} X_k}.$$

4. Déterminer l'espérance et la variance de la moyenne empirique en fonction de n, μ et σ.

Définition 2.1.2 On dit que la **suite de variables aléatoires $\boldsymbol{(X_n)_{n\in\mathbb{N}}}$ converge en loi vers** la variable $\boldsymbol{X}$ si et seulement si :

$$\boldsymbol{\forall a \in \mathbb{R} \text{ tel que } F_X \text{ continue en } a, \text{ on a } \lim_{n\to+\infty} F_{X_n}(a) = F_X(a)}.$$

Théorème 2.1.1 (Théorème central limite, première forme) Soit $(X_n)_{n\geqslant 1}$ une suite de variables aléatoires indépendantes de même loi admettant une espérance μ et une variance σ^2 non nulle. On pose :

$$\forall n \in \mathbb{N}^*, \qquad M_n^* = \frac{M_n - \mu}{\frac{\sigma}{\sqrt{n}}}$$

Alors $\boldsymbol{M_n^\star}$ **converge en loi vers une variable aléatoire suivant la loi normale centrée réduite.**

2.2 À l'aide de tirages répétés d'une loi exponentielle de paramètre 1

Soit $U \hookrightarrow \mathcal{U}([0,1[)$, c'est-à-dire que U correspond au tirage aléatoire d'un réel de l'intervalle $[0,1[$.

5. Quelle fonction Python permet de simuler une réalisation de U ?

On pose

$$X = -\ln(1-U).$$

On dit que X suit une loi exponentielle de paramètre 1, noté $X \hookrightarrow \mathcal{E}(1)$. On a $E(X) = 1$ et $V(X) = 1$.

6. Programmer une fonction `X()` qui simule une réalisation d'une variable aléatoire X qui suit une loi exponentielle de paramètre 1.
7. Écrire le théorème central limite pour une suite $(X_n)_{n\geqslant 1}$ de variables aléatoires indépendantes suivant la loi $\mathcal{E}(1)$.
8. Programmer une fonction `M(n)` qui simule une réalisation de la moyenne empirique M_n associée à la suite $(X_k)_{k\geqslant 1}$.
9. Programmer une fonction `Mstar(n)` qui simule une réalisation de la variable aléatoire centrée réduite associée à la moyenne empirique.
10. Tracer un histogramme représentant la répartition de 5000 réalisations de $M_n^\star$ pour $n = 100$. Pour ce faire, il faut créer une liste `L` contenant 5000 réalisations de $M_n^\star$ puis utiliser le code suivant :

```python
import numpy as np
import matplotlib.pyplot as plt
b=np.linspace(min(L),max(L),100) #On découpe l'intervalle [min(L),max(L)] en 100
# sous intervalles
plt.hist(L,b,density=True) #b indique les bases des rectangles à considérer et
# density=True demande un graphe normalisé
```

11. On veut comparer cet histogramme au graphe de la densité d'une variable alétoire suivant une loi normale centrée réduite, c'est-à-dire au graphe de la fonction :

$$\begin{aligned} f : \mathbb{R} &\to \mathbb{R} \\ t &\mapsto \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}. \end{aligned}$$

Tracer le graphe de cette fonction sur le même graphique que l'histogramme, donc en choisissant pour les points d'abscisses les mêmes que ceux de l'histogramme. Comparer les deux.

2.3 À l'aide du théorème de Moivre-Laplace

Théorème 2.3.1 (Théorème de Moivre-Laplace) Pour tout $n \in \mathbb{N}^\star$, on considère Y_n une variable aléatoire de loi $\mathcal{B}(n,p)$. On pose

$$Y_n^\star = \frac{Y - np}{\sqrt{np(1-p)}}.$$

Alors la suite $(Y_n^\star)_{n\geq 1}$ converge en loi vers une variable aléatoire de loi $\mathcal{N}(0,1)$.

Corollaire 2.3.2 (Approximation d'une loi binomiale par une loi normale)
Soit Y une variable aléatoire suivant une loi $\mathcal{B}(n,p)$ avec :

$$n \geqslant 30, \qquad np \geqslant 5 \qquad \text{et} \qquad n(1-p) \geqslant 5$$

alors on peut approcher la loi de $Y^\star = \dfrac{X - np}{\sqrt{np(1-p)}}$ par la loi normale $\mathcal{N}(0,1)$.
Cela revient à dire que l'on peut approcher la loi de Y par le loi $\mathcal{N}(np, np(1-p))$.

Soit $Y \hookrightarrow \mathcal{B}(400, \frac{1}{2})$.

12. Justifier que l'on peut approcher la loi de Y par celle d'une loi normale à préciser.
13. Programmer une fonction `Y()` qui simule une réalisation de Y.
14. Tracer un histogramme représentant la répartition de 5000 réalisations de Y en choisissant un nombre de rectangles égal à `max(L)-min(L)`.
15. Comparer cet histogramme à la courbe représentative de la fonction de densité d'une variable aléatoire suivant une loi normale aux paramètres adaptés. La fonction de densité d'une variable aléatoire suivant une loi normale de paramètres μ et σ est :

$$\begin{aligned} g : \mathbb{R} &\to \mathbb{R} \\ t &\mapsto \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}}. \end{aligned}$$

3 Illustration de la convergence d'une suite de va de loi binomiale vers une va de loi de Poisson

Approximation de la loi binomiale par la loi de Poisson
Soit $(n,p) \in \mathbb{N}^* \times]0,1[$. On suppose que :

$$n \geqslant 30 \qquad \text{et} \qquad p \leqslant 0,1 \qquad \text{et} \qquad np(1-p) \leqslant 10$$

Si $X \hookrightarrow \mathcal{B}(n,p)$, alors on peut approcher sa loi par la loi $\mathcal{P}(np)$.

Dans une chaîne de fabrication, 5% des pièces sont défectueuses. On prélève une pièce, on examine si elle est défectueuse et on la replace parmi toutes les autres. On répète 120 fois cette expérience. On désigne par Z la variable aléatoire qui à chaque tirage de 120 pièces associe le nombre de pièces défectueuses.

16. Déterminer la loi de Z puis calculer la probabilité d'obtenir exactement 5 pièces défectueuses.
17. Simuler numériquement une réalisation de Z.
18. Soient $(a,b) \in \mathbb{R}^2$ avec $a \leq b$, estimer numériquement la probabilité de l'évènement $P(a \leq Z \leq b)$.
19. Une approximation *Poissonnienne* de Z est-elle adaptée ? On note W une variable aléatoire de loi de Poisson qui approche la variable aléatoire Z.
20. On rappelle qu'on simule une réalisation d'une variable aléatoire qui suit une loi de Poisson de paramètre λ avec les commandes :

```python
from scipy import stats
W=stats.poisson(lambda)
w=W.rvs()
```

Soit $W \hookrightarrow \mathcal{P}(6)$, soient $(a,b) \in \mathbb{R}^2$ avec $a \leq b$, estimer numériquement la probabilité de l'évènement $P(a \leq W \leq b)$.

21. Conclure quant à la pertinence de l'approximation.

4 Méthode de Monte-Carlo : Estimer une aire

22. Que fait la fonction suivante, qui prend en entrée une fonction f et une liste X de longueur 2 ?

```
def relation (f,X):
    if f(X[0])>=X[1]:
        return(1)
    else:
        return(0)
```

23. Programmer une fonction `MonteCarlo(f,n)` qui prend en entrée une fonction f continue sur $[0,1]$ à valeur dans $[0,1]$ et à un entier n. Cette fonction simule n éléments de $[0,1]^2$ de manière uniforme, et compte les k points situés sous la courbe représentative de f. Enfin ce programme renvoie la valeur k/n.
24. Tester la fonction précédente pour $f(x) = x^3$ et les valeurs $n = 10, 100, 1000$. Comparer ces résultats avec $\displaystyle\int_0^1 t^3 \mathrm{d}t$.
25. Programmer la méthode des rectangles à gauche pour retrouver une approximation de la valeur de cette intégrale.
26. Tracer sur le même graphe l'évolution en fonction de n des erreurs d'approximation pour les deux méthodes. Laquelle semble la plus efficace ?
27. Modifier le programme `MonteCarlo` pour trouver une estimation de l'aire d'un disque de rayon $\frac{1}{2}$. Quel peut être l'intérêt de la méthode de Monte-Carlo par rapport à la méthode des rectangles ?