TP0 – Simulation et illustration

On rappelle que pour exploiter tout le potentiel de Python dans un contexte mathématique, il est nécessaire d'installer ou d'importer un certain nombre de modules à chaque session. Pour cette séance, on pourra charger les modules suivants :

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

1 Générer des variables aléatoires

Il existe plusieurs functions ad-hoc dans Python permettant de générer les variables aléatoires de loi prescrite. Nous y viendrons plus tard. Dans un premier temps, rappelons que toutes les variables réelles peuvent s'obtenir à partir de variables de loi uniforme sur $[0,1]$ par la méthode d'inversion de la fonction de répartition. Si U est une variable de loi uniforme sur $[0,1]$ et F_X la fonction de répartition de la loi cible que l'on cherche à simuler, alors si F_X^{-1} est l'inverse généralisé de F_X

$$F_X^{-1}(t) = \inf\{x,\ F_X(x) \geq t\},$$

la variable $F_X^{-1}(U)$ a précisément pour fonction de répartion la fonction F_X et donc la loi cible. Par exemple si on cherche à simuler une variable $X \sim \mathcal{E}(\lambda)$ de loi exponentielle, on a

$$F_X(x) = \mathbb{P}(X \leq x) = 1 - e^{-\lambda x}, \quad \text{donc} \quad F_X^{-1}(t) = -\frac{1}{\lambda}\log(1-t),$$

de sorte que

$$-\frac{1}{\lambda}\log(1-U) \sim \mathcal{E}(\lambda).$$

Exercice 1. *Variables de Bernoulli*

1. Explicitez les fonction F_X et F_X^{-1} lorsque $X \sim \mathcal{B}(p)$ est une variable de Bernoulli.
2. En utilisant un instruction conditionnelle `if...`, écrire un programme pour simuler une variable de Bernoulli de paramètre p à partir de la commande `rd.rand()`.
3. Réécrire le programme ci-dessus en remplaçant le `if` par une évaluation booléenne.
4. Écrire un programme qui génère n variables de Bernoulli de paramètre p. On pourra à nouveau utiliser `rd.rand(n)`.
5. Modifier ce programme pour renvoyer des variables de Rademacher, i.e. des variables de Bernoulli à valeurs dans $\{-1,1\}$ au lieu de $\{0,1\}$.

Exercice 2. *Variables étagées*

1. Écrire une fonction python qui simule une variable aléatoire X de loi :

$$\mathbb{P}_X = \sum_{i=1}^{n} p_i \delta_{x_i},$$

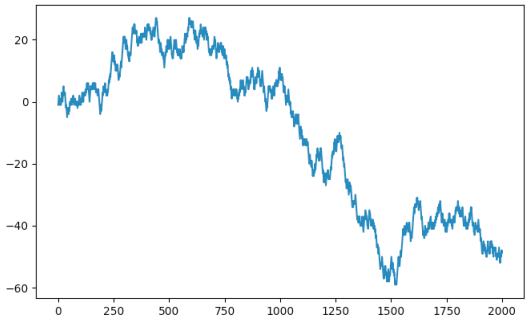
où $n \geq 2$, où les $(p_i)_{1 \leq i \leq n}$ sont des réels positifs de somme égale à un et où les x_i sont n nombres réels distincts. On pourra utiliser la fonction `np.cumsum` qui calcule les somme cumulées d'un tableau. Une première solution pourrait être par exemple

```
def loifinie(X,P):
    n=len(X)
    r=0
    Som=np.cumsum(P)
    b=rd.rand()
    for k in range(0,n-2):
        if (b>Som[k]) and (b<Som[k+1]):
           r= X[k+1]
    if b<Som[0]:
        r=X[0]
    return r
```

2. Modifiez éventuellement votre solution précédente de façon à ne pas utiliser de boucle, en utilisant les commandes `np.min` et `np.where`, ou encore `filter`.

Exercice 3. *Marche aléatoire*
Étant donné $p \in]0,1[$, on souhaite simuler une marche aléatoire sur $\mathbb{Z}$ issue de 0 telle qu'à chaque instant $n \geq 0$, $S_{n+1} = S_n + 1$ avec probabilité p et $S_{n+1} = S_n - 1$ avec probabilité $1-p$.

1. Écrire une fonction `marche_aleatoire(n,p)` traçant l'évolution de S_n en fonction de n. Pour cela, il faudra créer dans la fonction un vecteur de taille n contenant les valeurs successives de S_n et utiliser la fonction `plt.plot(.,.,label=p)` pour tracer S en fonction n.
2. Lancez la fonction pour différentes valeurs de p.
3. Que constatez-vous empiriquement lorsque $n \to \infty$ dans les cas : (i) $p < 0.5$, (ii) $p > 0.5$ et (iii) $p = 0.5$? On pourra superposer des graphiques et utiliser une légende avec les fonctions `label` et `plt.legend` pour plus de clarté.



2 Fonctions prédéfinies et histogrammes

Plusieurs modules et sous-modules de Python permettent directement de générer des variables aléatoires de loi prescrite. Il y a par exemple les modules `random`, `numpy.random` ou encore `scipy.stats`. Ce dernier sous-module contient en particulier les fonctions suivantes :

- `rvs` : random variables, i.e. simulation
- `pdf(x, parametre)` : probability density function i.e. densité en x
- `pmf(k, parametre)` : probability mass function i.e. densité discrête en k
- `cdf(x,parametre)` : cumulative density function i.e. fonction de répartition en x
- `ppf(q, parametre)` : percent point function i.e. q-ième quantile.

Ayant importé le sous-module via la commande `from scipy.stats import *`,

- pour générer une variable aléatoire de loi géométrique de paramètre $p = 1/3$, on écrira par exemple `geom.rvs(p=0.5)`
- pour générer 10 variables indépendantes de loi gaussienne $\mathcal{N}(0,1)$, on écrira de façon similaire `norm.rvs(size=10,loc=0,scale=1)`
- pour tracer la fonction de répartition d'une loi géométrique de paramètre $p = 1/3$, on pourra utiliser la fonction `x->geom.cdf(x,p=1/3)`
- pour tracer la densité de la loi exponentielle $\mathcal{E}(2)$, on pourra encore utiliser la fonction `x->expon.pdf(x,scale=1/2)`

Le tableau suivant résume les commandes selon les paramètres des lois usuelles.

loi(paramètres de forme)	Lois
`norm()`	Loi normale $\mathcal{N}(0,1)$.
`gamma(a=)`	Loi gamma $\gamma(a,1)$.
`beta(a=, b=)`	Loi béta $\beta(a,b)$.
`t(df=n)`	Loi Student de n degrés de liberté.
`cauchy()`	Loi de Cauchy.
`expon(scale=1/lambda)`	Loi exponentielle $\mathcal{E}(\lambda)$.
`uniform(loc=, scale=)`	Loi uniforme $\mathcal{U}(]loc, loc+scale[)$.
`randint(low, high)`	Loi uniforme $\mathcal{U}(\{low, low+1, \cdots, high-1\})$.
`binom(n=, p=)`	Loi binomiale $\mathcal{B}(n,p)$.
`geom(p=)`	Loi géométrique $\mathcal{G}(p)$.
`poisson(mu=)`	Loi de Poisson $\mathcal{P}(\mu)$.

On rappelle que si `X` est un tableau de nombres et N un entier naturel, la fonction

```
plt.hist(X,N,normed=1, facecolor='g')
```

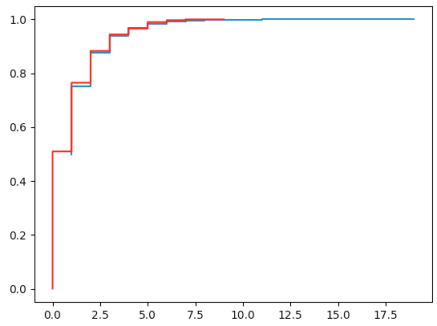
permet de représenter un histogramme de la répartition des éléments de `X` dans une partition de N sous-ensembles réguliers. On peut en option préciser la couleur de l'histogramme et à la place de l'entier N, on peut également considérer un tableau contenant une suite croissante de réels représentant la partition.

Exercice 4. *Fonctions de répartition et histogrammes empiriques*

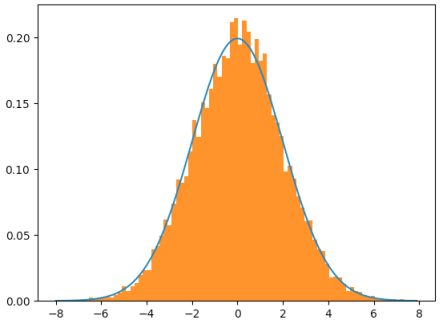
1. À l'aide de fonctions prédéfinies ou avec vos propres programmes, générer un échantillon $\{x_1, \ldots, x_n\}$ de n variables aléatoires indépendantes selon une loi usuelle discrète (géométrique, Poisson par exemple). À cet $n-$échantillon, on associe la fonction de répartition empirique

$$F_n(t) = \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{x_k \leq t}.$$

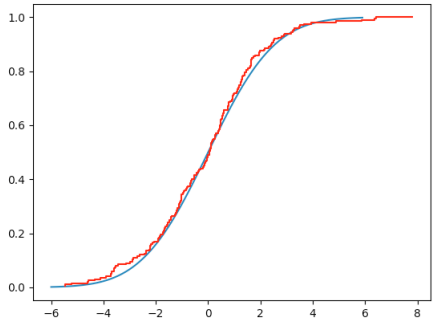
Tracer avec deux couleurs distinctes et sur un même graphique les fonctions de répartition empiriques et théoriques.



2. À l'aide de fonctions prédéfinies ou avec vos propres programmes, générer un échantillon de n variables aléatoires indépendantes selon une loi usuelle à densité (exponentielle, gaussienne par exemple). Tracer sur un même graphique l'histogramme empirique associé et la densité théorique correspondante. On pourra choisir une partition en N cases avec $N \approx \lfloor \sqrt{n} \rfloor$.



Faire de même avec la fonction de répartition empirique.



Exercice 5. *Théorème limite fondamentaux*
On considère une suite de variables aléatoires $(X_n)_{n\geq 1}$ indépendantes et toutes de même loi, de variance finie σ^2, par exemple des variables de Bernoulli, Poisson, géométriques, exponentielles, gaussiennes etc. On s'intéresse au comportement asymptotique de la somme $S_n := X_1 + \ldots + X_n$.

1. Pour vos lois usuelles favorites, tracer plusieurs trajectoires de la suite $(S_n/n - \mathbb{E}[X_1])_{n\geq 1}$.

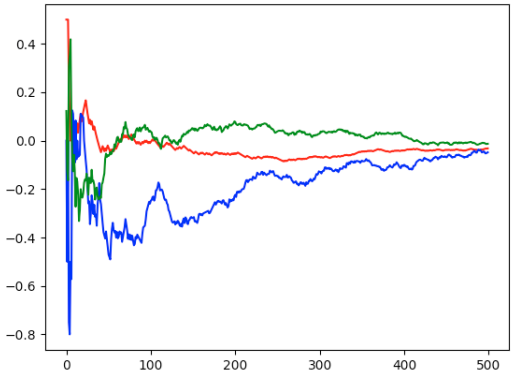


FIGURE 1 – Trajectoires de $S_n/n - \mathbb{E}[X_1]$ pour trois types de lois.

2. Pour $n = 1000$ et $m = 1000$, générer m réalisations de $\sqrt{n}(S_n/n - \mathbb{E}[X_1])$ et tracer l'histogramme empirique correspondant. On pourra superposer le graphe de la densité gaussienne $\mathcal{N}(0, \sigma^2)$.

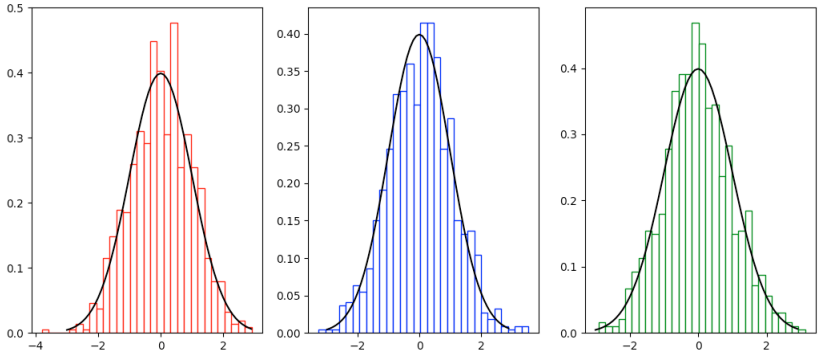


FIGURE 2 – Histogrammes empiriques et densité gaussienne pour trois type de lois.

3. Pour $n = 1000$ et $m = 1000$, générer m réalisations de $\sqrt{n}(S_n/n - \mathbb{E}[X_1])$ et la fonction de répartition empirique correspondante. On pourra superposer le graphe de la fonction de répartition gaussienne $\mathcal{N}(0, \sigma^2)$.