Python td 11 - Estimateurs, méthode de Monte-Carlo, méthode de rejet

Tous les programmes ci-dessous commenceront par :

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

Exercice 1
Edhec 2015 ECS

Soit X une variable aléatoire suivant une loi normale centrée réduite.
On considère par ailleurs la fonction g définie par :

$$g(x) = \frac{e^{-x}}{\sqrt{\pi x}} \text{ si } x > 0 \text{ et } g(x) = 0 \text{ si } x \leq 0$$

On admet qu'il s'agit de la densité d'une variable aléatoire Z. On admet enfin que

$$|X| = \sqrt{2Z}$$

1. Ecrire une fonction Python intitulée `def Z():` qui simule la variable aléatoire Z.

2. On considère les commandes suivantes :

```
n=int(input("Entrer n : "))
W=rd.exponential(1,n)
S=(np.sum(W*np.sqrt(W))/n) /np.sqrt(np.pi)
```

   En remarquant que $x^2.g(x) = \frac{x\sqrt{x}}{\sqrt{\pi}}.e^{-x}$, montrer que s contient une valeur approchée de $\int_0^{+\infty} x^2 g(x)dx$ pour peu que l'on entre une valeur de n assez grande.

Exercice 2
Edhec 2018 ECE

On considère l'intégrale

$$I = \int_0^1 \ln(1+t^2)dt$$

1. Compléter le script Python suivant pour qu'il calcule et affiche, à l'aide de la méthode de Monte-Carlon une valeur approchée de I :

```
n=100000
U=rd.random(n)
V=np.log(1+U**2)
I=.......
print(I)
```

2. On pose $u_0 = 1$ et pour tout entier $n \in \mathbb{N}^*$, $u_n = \int_0^1 (\ln(1+t^2))^n dt$.
   On admet que

$$\sum_{k=0}^{+\infty} u_k = \int_0^1 \frac{1}{1-\ln(1+t^2)}dt$$

   Modifier le script précédent pour donner une valeur approchée de $\sum_{k=0}^{+\infty} u_k$.

Exercice 3
Oral HEC 2016 - Question sans préparation

Donner la finalité du programme suivant (on pourra penser à la loi faible des grands nombres) :

```
N=100000
S=0
for i in range(1,N+1):
    u=rd.random()
    S=S+4/N*1/(1+u**2)
print(S)
```

Exercice 4
Méthode de rejet

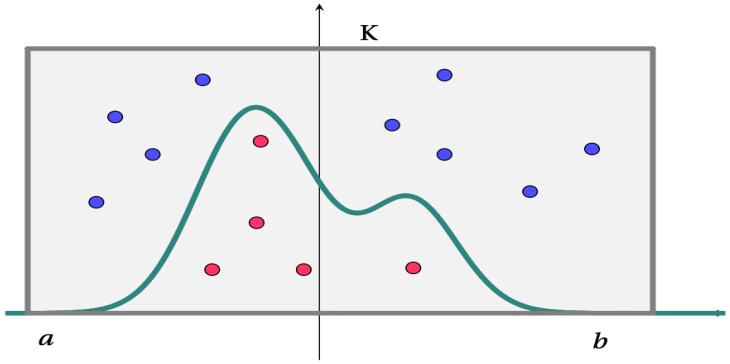
Soit X une variable à densité et f une densité de X. On suppose que la variable X est bornée et que la fonction f est bornée : il existe $K \in \mathbb{R}_+$, $(a,b) \in \mathbb{R}^2$ avec $a < b$ tels que

$$\forall x \in \mathbb{R}, \quad 0 \leq f(x) \leq K \text{ et } f(x) = 0 \text{ si } x \notin [a,b]$$

Rappelons que l'aire comprise entre la courbe, l'axe des abscisses et les droites d'équation $x = a$, $x = b$ correspond exactement à la probabilité que la variable aléatoire prenne les valeurs comprises entre a et b : Aire $= \int_a^b f(t)dt = P(a \leq X \leq b)$.

Pour simuler la variable X, on peut tirer un point au hasard sous la courbe représentative de f de manière uniforme. On prend alors pour réalisation de la loi l'abscisse de ce point.

Pour tirer un point sous la courbe de facon uniforme : on obtient des coordonnées aléatoires (x,y) en simulant des lois uniformes sur $[a,b] \times [0,K]$ jusqu'à ce que l'un des points tirés soit situé sous la courbe de f (d'où le nom de méthode de rejet).



1. Soit Y la variable égale au nombre d'essais avant d'avoir un point situé sous la courbe de f. Quelle est la loi de Y ?

2. Soit X une variable aléatoire dont une densité f est définie sur $\mathbb{R}$ par :

$$f(x) = \begin{cases} 6x(1-x) & \text{si } x \in [0,1] \\ 0 & \text{sinon} \end{cases}$$

   On dit alors que X suit la loi $\beta(2,2)$.

   (a) Etudier les variations de f, donner son maximum. Tracer sur Python le graphe de f.

   (b) Ecrire une fonction Python, intitulée `def beta()` qui simule la variable X par la méthode de rejet à l'aide de la commande `rd.random()`.