FEUILLE DE TD/TP 2

Pour tous les exercices dans lesquels vous proposerez une méthode de simulation, il est conseillé de faire une vérification par histogramme. Les exercices plus difficiles sont indiqués par un astérisque.

Exercice 1.

(1) Montrer que l'algorithme suivant permet de simuler une réalisation de loi uniforme sur le disque unité de $\mathbb{R}^2$, ie la loi de densité $p : \mathbb{R}^2 \to \mathbb{R}$ définie par

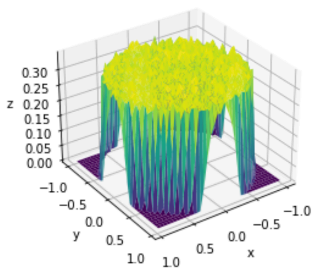
$$\forall (u, v) \in \mathbb{R}^2,\ p(u, v) = \frac{1}{\pi} \mathbb{1}_{u^2+v^2 \leq 1}.$$

(a) Simuler (U, V) couple de deux v.a. i.i.d. de loi uniforme sur $[-1, 1]$.
(b) Tant que $U^2 + V^2 > 1$, répéter (a).
(c) Renvoyer la valeur de (U, V) en fin de boucle.

Il s'agit d'une application directe de l'algorithme du rejet vu en cours.

(2) Coder l'algorithme en Python. Pour la représentation graphique, on pourra utiliser le package `mplot3d` de `mpl_toolkits`.

```
##simulation de la densit\'e uniforme sur le disque

def simudisque():
    x=2*np.random.rand(2)-1
    while(x[0]**2+x[1]**2>1):
        x=2*np.random.rand(2)-1
    return(x)

##repr\'esentation graphique

#simulation d'un vecteur de taille 1E6

r=np.zeros(2*int(1E6)).reshape(int(1E6),2)
for i in range(int(1E6)):
    r[i,]=simudisque()

## on cree les classes de côt\'es de longueur .05
## et on compte le nombre de tirages dans chaque classe
## on normalise par le  nombre de tirages
## et par l'aire de chaque classe

s=np.arange(-1,1,.05)
n=len(s)
x=s
y=s
z=np.zeros(n**2).reshape(n,n)
for i in range(n):
    for j in range(n):
            z[i,j]=sum((r[:,0]>=s[i])&(r[:,0]<s[i]+.05)&(r[:,1]>=s[j])&(r[:,1]<s[j]+.05))
            z[i,j]=0.05**(-2)*z[i,j]/1E6



fig = plt.figure()
ax = plt.axes(projection='3d')
X,Y=np.meshgrid(x,y
ax.plot_surface(X, Y, z, rstride=1, cstride=1,
                cmap='viridis', edgecolor='none')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z');

ax.view_init(30, 55)
fig
```



Histogramme 3D

L'exercice revient à montrer que, pour (U,V) suivant une loi uniforme sur le disque unité, les v.a. $X = ZU$ et $Y = ZV$, avec $Z = [-2\ln(R^2)/R^2]^{1/2}$ et $R^2 = U^2+V^2$, sont des gaussiennes $\mathcal{N}(0,1)$ indépendantes.

Exercice 2.

(1) On considère un couple de variables (U,V) de loi uniforme sur le disque unité de $\mathbb{R}^2$. On pose $R^2 = U^2+V^2$. Montrer que le couple (X,Y) défini par

$$X = \sqrt{-2\frac{\ln R^2}{R^2}}U, Y = \sqrt{-2\frac{\ln R^2}{R^2}}V$$

a même loi que le couple

$$(\sqrt{-2\ln\rho}\cos 2\pi\Theta, \sqrt{-2\ln\rho}\sin 2\pi\Theta)$$

où ρ et Θ sont deux variables i.i.d. de loi $\mathcal{U}([0;1])$. En déduire la loi du couple (X,Y).

On utilise pour cela la méthode de la fonction muette. Il s'agit d'une méthode générique pour identifier la loi d'un vecteur aléatoire. L'idée est simple. On se donne une fonction f continue bornée de $\mathbb{R}^2$ dans $\mathbb{R}$ et l'on cherche à écrire l'espérance $\mathbb{E}[f(X,Y)]$ sous la forme

$$\mathbb{E}\big[f(X,Y)\big] = \int_{\mathbb{R}^2} f(x,y)d\mu(x,y),$$

pour μ une mesure de probabilité sur $\mathbb{R}^2$ (indépendante de f), auquel cas la loi de (X,Y) est précisément la mesure de probabilité μ. Un exemple typique est le cas où

$$\mathbb{E}\big[f(X,Y)\big] = \int_{\mathbb{R}^2} f(x,y)p(x,y)\,\mathrm{d}x\,\mathrm{d}y,$$

c'est à dire μ est la mesure de probabilité sur $\mathbb{R}^2$ de densité p ; alors, le vecteur (X,Y) suit la loi de densité p.

Appliquons cette méthode ici. Pour une fonction muette f continue bornée de $\mathbb{R}^2$ dans $\mathbb{R}$, on a

$$\mathbb{E}\big[f(X,Y)\big]\frac{1}{\pi}\int_{\mathbb{R}^2} f\left(\sqrt{-2\frac{\ln(u^2+v^2)}{u^2+v^2}}u, \sqrt{-2\frac{\ln(u^2+v^2)}{u^2+v^2}}v\right)\mathbb{1}_{\{u^2+v^2\le 1\}}dudv.$$

On effectue le changement de coordonnées polaires: $u = r\cos(\theta)$, $v = r\sin(\theta)$, pour $r > 0$ et $\theta \in]0,2\pi[$. (Précision technique : pour que le changement de variables en coordonnées polaires soit bien un difféomorphisme entre deux ouverts, on doit en fait considérer l'intégrale en (u,v) sur l'ensemble $\mathbb{R}^2 \setminus \{(x,0)\,/\,x\ge 0\}$. Le Borélien $\{(x,0)\,/\,x\ge 0\}$, étant de mesure nulle, cela ne change rien à la valeur de l'intégrale.)

On a alors $dudv = r\,\mathrm{d}r\,\mathrm{d}\theta$ et il vient

$$\begin{aligned}\mathbb{E}\big[f(X,Y)\big] &= \frac{1}{\pi}\int_0^{\infty}\int_0^{2\pi} f\left(\sqrt{-2\ln(r^2)}\cos(\theta), \sqrt{-2\ln(r^2)}\sin(\theta)\right)\mathbb{1}_{\{r\le 1\}}r\,\mathrm{d}r\,\mathrm{d}\theta \\ &= \frac{1}{\pi}\int_0^{2\pi}\left[\int_0^1 f\left(\sqrt{-2\ln(r^2)}\cos(\theta), \sqrt{-2\ln(r^2)}\sin(\theta)\right)r\,\mathrm{d}r\right]\mathrm{d}\theta.\end{aligned}$$

On peut faire les changements de variable $\rho = r^2$ soit $2r\,\mathrm{d}r = \mathrm{d}\rho$ et d'autre part $\gamma = \frac{\theta}{2\pi}$ soit $\mathrm{d}\gamma = \frac{\mathrm{d}\theta}{2\pi}$. De fait,

$$\begin{aligned}\mathbb{E}\big[f(X,Y)\big] &= \int_0^1\left[\int_0^1 f\big(\sqrt{-2\ln\rho}\cos(2\pi\gamma), \sqrt{-2\ln\rho}\sin(2\pi\gamma)\big)d\rho\right]\mathrm{d}\gamma \\ &= \int_0^1\int_0^1 f\big(\rho\cos(2\pi\gamma), \rho\sin(2\pi\gamma)\big)d\rho\,\mathrm{d}\gamma.\end{aligned}$$

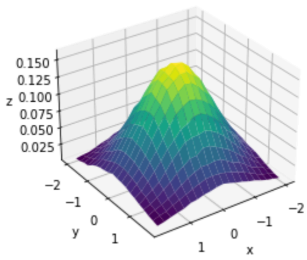
Cela démontre le premier point de la question. L'égalité en loi à la base de l'algorithme de Box-Muller nous dit alors que (X,Y) est un couple de v.a. indépendantes de même loi $\mathcal{N}(0,1)$.

(2) Utiliser le résultat de la question précédente pour montrer que l'algorithme suivant permet de simuler un couple de v.a. i.i.d. de loi gaussiennes centrées réduites.
   (a) Simuler (U,V) couple de deux v.a. i.i.d. de loi uniforme sur $[-1,1]$.
   (b) Tant que $U^2+V^2 > 1$, répéter (a).
   (c) Renvoyer la valeur de (U,V) et de $R^2 = U^2+V^2$ en fin de boucle.
   (d) Poser $Z = [-2\ln(R^2)/R^2]^{1/2}$.
   (e) Poser $X = ZU$ et $Y = ZV$ et renvoyer (X,Y).

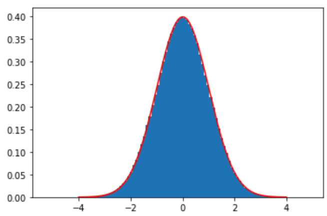
Il s'agit d'une conséquence directe de la question précédente et de l'algorithme du rejet.

(3) Coder l'algorithme en Python. Vérifier à l'aide d'un histogramme que la variable X simulée suit bien une loi gaussienne centrée réduite.

```
def simugaussienne():
    s=simudisque()
    r=s[0]**2+s[1]**2 #rayon au carr\'e
    z=np.sqrt(-2*np.log(r)*r**(-1))
    x=z*s[0]
    y=z*s[1]
    return [x,y]

## Representation graphique (dessin 3d)

# simulation d'un vecteur de taille 1E6
r=np.zeros(2*int(1E6)).reshape(int(1E6),2)
for i in range(int(1E6)):
    r[i,]=simugaussienne()


# on cree les classes de cotes de longueur .2
# on normalise par le nombre de tirages
# et par le nombre de classes


s=np.arange(-2,1.9,.2)
n=len(s)
x=s
y=s
z=np.zeros(n**2).reshape(n,n)
for i in range(n):
    for j in range(n):
        z[i,j]=sum((r[:,0]>=s[i])&(r[:,0]<s[i]+.2)&(r[:,1]>=s[j])&(r[:,1]<s[j]+.2))
        z[i,j]=0.2**(-2)*z[i,j]/1E6


#Representation 3d

fig = plt.figure()
ax = plt.axes(projection='3d')
X,Y=np.meshgrid(x,y)
ax.plot_surface(X, Y, z, rstride=1, cstride=1,
                cmap='viridis', edgecolor='none')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')


#Histogramme pour la loi de x

plt.hist(r[:,0],100,density=True)
absc=np.arange(-4,4,.01)
ordo=(np.pi*2)**(-.5)*np.exp(-absc**2/2)
plt.plot(absc,ordo,color='r')
plt.show()
```



Histogramme 3D



Histogramme de r[:,0]

Exercice 3.

(1) Soient X et Y deux v.a. i.i.d. de loi exponentielle de paramètre 1.

(a) Montrer que la loi conditionnelle de X sachant $2Y > (1-X)^2$ a pour densité

$$x \mapsto \frac{2}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \mathbb{1}_{[0,+\infty[}(x).$$

On utilise toujours la méthode de la fonction muette. Soit f continue bornée. Par définition,

$$\mathbb{E}\left[f(X)|2Y > (1-X)^2\right] = \frac{1}{\mathbb{P}(2Y > (1-X)^2)} \mathbb{E}\left[f(X)\mathbb{1}_{\{2Y>(1-X)^2\}}\right].$$

On se focalise sur le deuxième terme. Il vient, par le théorème de Fubini,

$$\begin{aligned}
\mathbb{E}\left[f(X)\mathbb{1}_{\{2Y>(1-X)^2\}}\right] &= \int_0^\infty \int_0^\infty f(x)\mathbb{1}_{\{2y>(1-x)^2\}} e^{-(x+y)}\, \mathrm{d}x\, \mathrm{d}y \\
&= \int_0^\infty f(x) e^{-x} \left[\int_{(1-x)^2/2}^\infty e^{-y}\, \mathrm{d}y\right] \mathrm{d}x \\
&= \int_0^\infty f(x) e^{-x} e^{-\frac{1}{2}(1-x)^2}\, \mathrm{d}x
\end{aligned}$$

$$= e^{-\frac{1}{2}} \int_0^\infty f(x) e^{-\frac{x^2}{2}} \, dx.$$

En choisissant $f = 1$, on obtient

$$\mathbb{P}\big(2Y > (1-X)^2\big) = e^{-\frac{1}{2}} \int_0^\infty e^{-\frac{x^2}{2}} \, dx = \frac{1}{2} e^{-\frac{1}{2}} \int_{-\infty}^\infty e^{-\frac{x^2}{2}} \, dx = \frac{\sqrt{2\pi}}{2} e^{-\frac{1}{2}}.$$

On en déduit que

$$\mathbb{E}\big[f(X) | 2Y > (1-X)^2\big] = \frac{2}{\sqrt{2\pi}} \int_0^\infty f(x) e^{-\frac{x^2}{2}} \, dx,$$

ce qui montre que la loi conditionnelle de X sachant $2Y > (1-X)^2$ a pour densité

$$x \mapsto \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \mathbb{1}_{[0,+\infty[}(x).$$

(b) Soit S une v.a. de loi Bernoulli de paramètre $1/2$, indépendante du couple (X, Y). Montrer que la loi conditionnelle de $(2S-1)X$ sachant $2Y > (1-X)^2$ suit une loi normale centrée réduite.

On suit la démonstration de la première question. Pour une fonction muette f comme précédemment, en utilisant l'indépendance de S et de (X, Y), on calcule

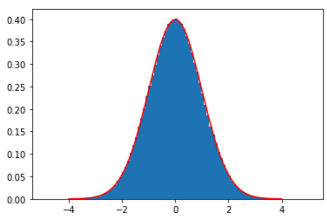
$$\mathbb{E}\big[f\big((2S-1)X\big) | 2Y > (1-X)^2\big] = \frac{1}{2}\mathbb{E}\big[f\big(-X\big) | 2Y > (1-X)^2\big] + \frac{1}{2}\mathbb{E}\big[f(X) | 2Y > (1-X)^2\big].$$

On applique maintenant la première question. On obtient

$$\mathbb{E}\big[f\big((2S-1)X\big) | 2Y > (1-X)^2\big] = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^\infty f(x) e^{-\frac{x^2}{2}}) \, dx.$$

Cela montre que la loi conditionnelle de $(2S-1)X$ sachant $2Y > (1-X)^2$ suit une loi normale centrée réduite.

(2) En déduire un algorithme de simulation de la loi $\mathcal{N}(0, 1)$ et le coder en Python.



Histogramme de r

```
## Simulation de la gaussienne

def simugaussienne2():
    x=-np.log(np.random.rand(2))
    s=(np.random.rand(1)<1/2).astype(int)
    while (2*x[1]<=(1-x[0])**2):
        x=-np.log(np.random.rand(2))
    return((2*s-1)*x[0])

# simulation d'un vecteur de taille 1E6 r=rep(0,1E6)
r=np.zeros(int(1E6))
for i in range(int(1E6)):
    r[i]=simugaussienne2()


# histogramme pour la loi de x

plt.hist(r,100,density=True)
absc=np.arange(-4,4,.01)
ordo=(np.pi*2)**(-.5)*np.exp(-absc**2/2)
plt.plot(absc,ordo,color='r')
plt.show()
```

Exercice 4.

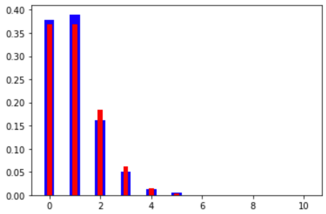
(1) Soit $(T_n)_{n\geq 1}$ une suite de v.a. indépendendantes de loi exponentielle de paramètre $\lambda > 0$. Montrer que la v.a.

$$N = \max\{n \geq 1 : T_1 + \cdots + T_n \leq 1\},$$

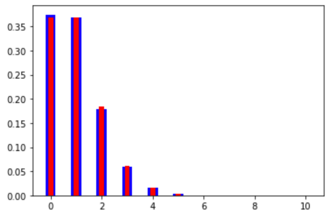
avec la convention $N = 0$ si l'ensemble ci-dessus est vide, suit une loi de Poisson de paramètre $\lambda > 0$.

Cf Cours.

(2) Déduire deux méthodes de simulation d'une loi de Poisson de paramètre λ, l'une avec inversion de la fonction de répartition et l'autre s'appuyant sur le résultat ci-dessus. Comparer les vitesses d'exécution.



Première méthode



Deuxième méthode

```
## Simulation de la loi de Poisson

def simupoisson(lambd,u):
    i=0
    F=np.exp(-lambd)
    while (F<u):
        i=i+1
        F=F+np.exp(-lambd)*lambd**i/math.factorial(i)
    return(i)

#representation sur 0,...,10
lambd=1
r=range(11)
p=[np.exp(-lambd)*lambd**s*(math.factorial(s))**(-1) for s in r]

x=np.zeros(1000)
for i in range(1000):
    x[i]=simupoisson(lambd,np.random.rand(1))

q=np.zeros(11)
for i in range(11):
    q[i]=np.sum(x==i)/1000

#histogramme
plt.bar(range(11),q,color='b',width=.4)
#comparaison avec loi theorique
plt.bar(range(11),p,color='r',width=.2)
plt.show()

## Simulation de la loi de Poisson via le processus de Poisson

def simupoisson2(lambd):
    # attention : ici, on ne met pas le tirage de l'uniforme en parametre {
    n=0 #valeur pas defaut
    t=-np.log(np.random.rand(1))/lambd #on tire la premiere exponentielle
    while (t<=1):
        n=n+1
        t=t-np.log(np.random.rand(1))/lambd
    return(n)

#representation sur 0,...,10
lambd=1
r=range(11)
p=[np.exp(-lambd)*lambd**s*(math.factorial(s))**(-1) for s in r]
x=np.zeros(10000)
for i in range(10000):
    x[i]=simupoisson2(lambd)

q=np.zeros(11)
for i in range(11):
    q[i]=np.sum(x==i)/10000

#historamme
plt.bar(range(11),q,color='b',width=.4)
#comparaison avec loi theorique
plt.bar(range(11),p,color='r',width=.2)
plt.show()



##Comparaison des temps d'execution sur 1E6 simulations

# première m\'ethode

start = time.time()
for i in range(int(1E6)):
    simupoisson(lambd,np.random.rand(1))
end = time.time()
```

```
elapsed=end-start
print(elapsed)

#deuxième m\'ethopde
start = time.time()
for i in range(int(1E6)):
    simupoisson2(lambd)
end = time.time()
elapsed=end-start
print(elapsed)
```

Exercice 5. On désigne par f la fonction définie par

$$\forall x \in \mathbb{R},\ f(x) = \sqrt{\frac{2}{\pi}} e^{-\frac{x^2}{2}} \mathbb{1}_{\{x \geq 0\}}.$$

(1) Montrer que la fonction f est une densité de probabilité.

La fonction f est positive (et borélienne). De plus, par symétrie de la fonction gaussienne,

$$\int_{\mathbb{R}} f(x)\,\mathrm{d}x = \int_0 \sqrt{\frac{2}{\pi}} e^{-\frac{x^2}{2}}\,\mathrm{d}x = \int_{\mathbb{R}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\,\mathrm{d}x = 1.$$

(2) Pour $\lambda > 0$ fixé, trouver une constante $c_\lambda > 0$ telle que

$$\forall x \in \mathbb{R},\ f(x) \leq c_\lambda \lambda e^{-\lambda x} \mathbb{1}_{\{x \geq 0\}}.$$

Considérons la fonction h définie par

$$\forall x \geq 0, \quad h(x) = \frac{f(x)}{\lambda e^{-\lambda x}} = \frac{\sqrt{2}}{\sqrt{\pi}\lambda} e^{\lambda x - \frac{x^2}{2}}.$$

On cherche le maximum de cette fonction. Pour simplifier les calculs, on étudie plutôt les variations de la fonction $\ln h$ qui sont les mêmes que celles de h : pour tout $x > 0$,

$$(\ln h)'(x) = \big(\lambda x - \frac{x^2}{2} + \ln \frac{\sqrt{2}}{\sqrt{\pi}\lambda}\big)' = \lambda - x$$

Ainsi, la fonction $\ln h$ et donc la fonction h sont maximales en $x = \lambda$ et

$$h(\lambda) = \frac{\sqrt{2}}{\sqrt{\pi}\lambda} e^{\frac{\lambda^2}{2}}$$

Donc, pour tout $x \in \mathbb{R}$,

$$f(x) \leq h(\lambda)\lambda e^{-\lambda x} \mathbb{1}_{\{x \geq 0\}} \quad \text{soit} \quad c_\lambda = \frac{\sqrt{2}}{\sqrt{\pi}\lambda} e^{\frac{\lambda^2}{2}}.$$

(3) En déduire une méthode de simulation de la loi de densité f.

D'après la question précédente, on peut simuler selon la loi de densité f en utilisant une méthode de rejet avec la loi exponentielle $\mathcal{E}(\lambda)$, $\lambda > 0$. On pose donc

$$\alpha(x) = \frac{f(x)}{c_\lambda \lambda e^{-\lambda x}} \mathbb{1}_{x \geq 0} = e^{-\frac{(x-\lambda)^2}{2}} \mathbb{1}_{x \geq 0}$$

et l'algorithme suivant renvoie alors une variable de densité f :

(a) Simuler (U, X), indépendants entre eux et de ce qui précède, de loi respective $\mathcal{U}([0;1])$ et $\mathcal{E}(\lambda)$.
(b) Tant que $U > \alpha(X)$, répéter (a).
(c) Renvoyer la valeur de X en fin de boucle.

(4) Trouver λ tel que le temps moyen de calcul dans la méthode proposée soit le plus petit possible.

Les résultats du cours nous disent que le temps moyen de calcul suit une loi géométrique de paramètre $1/c_\lambda$ et donc de moyenne c_λ. On cherche donc à minimiser cette fonction de λ. De manière équivalente en passant au logarithme, on minimise $\ell(\lambda) = \frac{\lambda^2}{2} - \ln \lambda$. Comme $\ell'(\lambda) = \lambda - 1/\lambda$, la fonction est donc minimale en $\lambda = 1$.

(5) Écrire un code Python qui met en place une méthode de simulation de type rejet pour la loi de densité f en utilisant la densité exponentielle de paramètre 1 pour la domination. Mettre ensuite en place une méthode de simulation de type rejet pour la loi de densité f en utilisant la densité exponentielle de paramètre 2 pour la domination. Comparer les temps d'exécution.

```
# lambda=1

def ratioalpha(x):
    return(np.exp(-(x-1)**2/2))

def simuf():
    y=-np.log(np.random.rand(1))
    while (ratioalpha(y)<np.random.rand(1)):
        y=-np.log(np.random.rand(1))
    return(y)

r=np.zeros(int(1E6))

for i in range(int(1E6)):
    r[i]=simuf()

# histogramme pour la loi de x
plt.hist(r,100,density=True) #histogramme
absc=np.arange(0,5,.01)
ordo=np.sqrt(2/np.pi)*np.exp(-absc**2/2)
plt.plot(absc,ordo,color="r")
plt.title('Simulation de la loi gaussienne')

# lambda=2

def ratioalpha2(x):
    return(np.exp(-(x-2)**2/2))

def simuf2():
    y=-np.log(np.random.rand(1))/2
    while (ratioalpha2(y)<np.random.rand(1)):
        y=-np.log(np.random.rand(1))/2
    return(y)

r=np.zeros(int(1E6))
for i in range(int(1E6)):
    r[i]=simuf2()

# histogramme pour la loi de x
plt.hist(r,100,density=True) #histogramme
absc=np.arange(0,5,.01)
ordo=np.sqrt(2/np.pi)*np.exp(-absc**2/2)
plt.plot(absc,ordo,color='r')
plt.title('Simulation de la loi gaussienne')

# comparaison des temps d'execution

start = time.time()
for i in range(int(1E6)):
    simuf()
end = time.time()
elapsed=end-start
print(elapsed)

start = time.time()
for i in range(int(1E6)):
    simuf2()
end = time.time()
elapsed=end-start
print(elapsed)
```

Exercice 6. Pour $a > 0$ donné, on désigne par f la fonction définie par

$$\forall x \in \mathbb{R}, \quad f(x) = Ce^{-x}\mathbb{1}_{[0;a]}(x).$$

(1) Trouver la valeur de la constante C pour que f soit une densité.

Une densité étant d'intégrale 1 sur $\mathbb{R}$, on obtient $C = 1/(1-e^{-a})$.

(2) Trouver une constante $c_1 > 1$ telle que

$$\forall x \in \mathbb{R}, \; f(x) \leq c_1 \frac{\mathbb{1}_{[0;a]}(x)}{a}.$$

On cherche à majorer la fonction $x \to \frac{ae^{-x}}{1-e^{-a}}$. Cette fonction étant décroissante, elle est maximale en 0 et on obtient

$$c_1 = \frac{a}{1-e^{-a}}$$

(3) Trouver une constante $c_2 > 1$ telle que

$$\forall x \in \mathbb{R}, \; f(x) \leq c_2 \mathbb{1}_{[0;+\infty[}(x)e^{-x}.$$

On obtient cette fois $c_2 = \frac{1}{1-e^{-a}}$.

(4) On veut mettre en place une méthode de rejet pour simuler la loi de densité f en utilisant la loi uniforme sur $[0; a]$ ou la loi exponentielle de paramètre 1. Laquelle vaut-il mieux choisir ?

Le critère naturel pour comparer l'efficacité des deux méthodes est le nombre moyen d'itérations dans la procédure qui est respectivement la constante c_1 ou c_2. De manière assez naturelle, la réponse dépend alors de la valeur du paramètre a. Ainsi, si $a \leq 1$, la première méthode est préférable et, si $a > 1$, la seconde est plus efficace.

(5) Coder chacune des deux méthodes et comparer les temps d'exécution.

```
# premiere methode

def ratioalpha(x):
    return(np.exp(-x))

def simuf(a):
    y=a*np.random.rand(1)
    while (ratioalpha(y)<np.random.rand(1)):
        y=a*np.random.rand(1)
    return(y)

a=2
r=np.zeros(int(1E6))
for i in range(int(1E6)):
    r[i]=simuf(a)

# histogramme pour la loi de x
plt.hist(r,100,density=True) #histogramme
absc=np.arange(0,a,.01)
ordo=(1-np.exp(-a))**(-1)*np.exp(-absc)
plt.plot(absc,ordo,color='r')
```

```
# deuxieme methode

def ratioalpha2(x,a):
    return((x<=a).astype(int))

def simuf2(a):
    y=-np.log(np.random.rand(1))
    while (ratioalpha2(y,a)<np.random.rand(1)):
        y=-np.log(np.random.rand(1))
    return(y)

r=np.zeros(int(1E6))
for i in range(int(1E6)):
    r[i]=simuf2(a)

# histogramme pour la loi de x
plt.hist(r,100,density=True)
absc=np.arange(0,a,.01)
ordo=(1-np.exp(-a))**(-1)*np.exp(-absc)
plt.plot(absc,ordo,color='r')

# comparaison des temps d'execution

a=2

start = time.time()
for i in range(int(1E6)):
    simuf(a)
end = time.time()
elapsed=end-start
print(elapsed)

start = time.time()
for i in range(int(1E6)):
    simuf2(a)
end = time.time()
elapsed=end-start
print(elapsed)

#repeter avec a=.5
```

Exercice 7. On souhaite simuler suivant la loi de densité

$$f\ :\ x \in \mathbb{R} \mapsto \frac{1}{Z} e^{-x^2} \mathbb{1}_{x \geq 1}, \quad \text{avec } Z = \int_1^{+\infty} e^{-x^2}\, dx.$$

(1) Trouver une densité g (suivant laquelle on sait simuler) et une constante C telles que $f \leq Cg$.

(2) Soit $X \sim \mathcal{N}(0, 1/2)$. Montrer que f est la densité de la loi de X sachant $X \geq 1$ (on pourra calculer l'espérance sur une fonction test).

(3) Coder la simulation de la loi de densité f. Sur l'histogramme de vérification, on tracera la fonction en traçant $x \mapsto c\exp(-x^2)$, avec c obtenu comme `e*histog[0][0]` comme dans l'exercice 1.

```
def ratioalpha(x):
    return(np.exp(-x**2+x))

def simuf():
    y=1-np.log(np.random.rand(1))
    while (ratioalpha(y)<np.random.rand(1)):
        y=1-np.log(np.random.rand(1))
    return(y)
```

```
r=np.zeros(int(1E6))
for i in range(int(1E6)):
    r[i]=simuf()

# histogramme pour la loi de x
histog=plt.hist(r,100,density=True) #histogramme
absc=np.arange(1,4,.01)
ordo=np.exp(-absc**2+1)*histog[0][0]
plt.plot(absc,ordo,color='r')
```

*** Exercice 8.** Soient f et $g : \mathbb{R} \mapsto \mathbb{R}$ des densités et h la fonction définie par :

$$\forall x \in \mathbb{R}, \quad h(x) = \frac{\max(f(x), g(x))}{\int_{\mathbb{R}} \max(f(t), g(t))dt}.$$

(1) On veut simuler une v.a. Z de densité h suivant une méthode d'acceptation/rejet. On considère une suite de variables i.i.d. $(X_n)_{n\in\mathbb{N}}$ de loi de densité f et une autre suite indépendante de la première $(Y_n)_{n\in\mathbb{N}}$ de loi de densité g. On considère également deux autres suites de v.a.i.i.d. $(U_n)_{n\in\mathbb{N}}$ et $(V_n)_{n\in\mathbb{N}}$ de loi uniforme sur $[0,1]$. Ces suites sont supposées indépendantes entre elles et indépendantes de $(X_n)_{n\in\mathbb{N}}$ et de $(Y_n)_{n\in\mathbb{N}}$. On pose alors

$$T = \min\{n \in \mathbb{N}^*,\ U_n f(X_n) \le g(X_n) \text{ ou } V_n g(Y_n) \le f(Y_n)\}$$

et

$$Z = \begin{cases} Y_T & \text{si } U_T f(X_T) \le g(X_T) \\ X_T & \text{si } V_T g(Y_T) \le f(Y_T) \text{ et } U_T f(X_T) > g(X_T) \end{cases}$$

Montrer que Z est de loi de densité h.

Pour toute fonction ℓ continue bornée, on a

$$\begin{aligned}
&\mathbb{E}[\ell(Z)] \\
&= \sum_{n\ge 1} \mathbb{E}\left[\left(\ell(Y_n)\mathbb{1}_{U_n f(X_n)\le g(X_n)} + \ell(X_n)\mathbb{1}_{V_n g(Y_n)\le f(Y_n) \text{ et } U_n f(X_n)>g(X_n)}\right)\mathbb{1}_{T=n}\right] \\
&= \sum_{n\ge 1} \mathbb{E}\left[\left(\ell(Y_n)\mathbb{1}_{U_n f(X_n)\le g(X_n)} + \ell(X_n)\mathbb{1}_{V_n g(Y_n)\le f(Y_n) \text{ et } U_n f(X_n)>g(X_n)}\right)\mathbb{1}_{\forall i<n,\ U_i f(X_i)>g(X_i) \text{ et } V_i g(Y_i)>f(Y_i)}\right].
\end{aligned}$$

Par indépendance des variables et les lois de celles-ci, on obtient :

$$\mathbb{E}[\ell(Z)] = \tag{1}$$
$$\sum_{n\ge 1}\left(\mathbb{E}[\ell(Y_1)]\,\mathbb{P}(U_1 f(X_1) \le g(X_1)) + \mathbb{E}\left[\ell(X_1)\mathbb{1}_{U_1 f(X_1)>g(X_1)}\right]\mathbb{P}(V_1 g(Y_1) \le f(Y_1))\right)\left(\mathbb{P}(U_1 f(X_1) > g(X_1))\,\mathbb{P}(V_1 g(Y_1) > f(Y_1))\right)^{n-1}.$$

Or, le théorème de Fubini nous montre que :

$$\begin{aligned}
\mathbb{E}\left[\ell(X_1)\mathbb{1}_{U_1 f(X_1)>g(X_1)}\right] &= \int_{\mathbb{R}}\int_0^1 \ell(x)\mathbb{1}_{uf(x)>g(x)} f(x)\,\mathrm{d}u\,\mathrm{d}x \\
&= \int_{\mathbb{R}} \ell(x)f(x)\mathbb{1}_{f(x)>0}\int_{\min(1,g(x)/f(x))}^1 \mathrm{d}u\,\mathrm{d}x \\
&= \int_{\mathbb{R}} \ell(x)(f(x) - \min(f(x), g(x)))\,\mathrm{d}x
\end{aligned}$$

Pour simplifer les notations, on pose

$$m = \int_{\mathbb{R}} \min(f(x), g(x))\,\mathrm{d}x.$$

Ainsi, en prenant la fonction $\ell \equiv 1$ dans l'égalité précédente, on obtient $\mathbb{P}(U_1 f(X_1) > g(X_1)) = 1 - m$ et, on peut obtenir, de même $\mathbb{P}(V_1 g(Y_1) > f(Y_1)) = 1 - m$.

En revenant à (1), cela nous donne :

$$\begin{aligned}
\mathbb{E}[\ell(Z)] &= \left(m\int_{\mathbb{R}} \ell(x)g(x)\,\mathrm{d}x + m\int_{\mathbb{R}} \ell(x)(f(x) - \min(f(x), g(x)))\,\mathrm{d}x\right)\sum_{n\ge 1}(1-m)^{2(n-1)}. \\
&= \frac{m}{1-(1-m)^2}\int_{\mathbb{R}} \ell(x)(g(x) + f(x) - \min(f(x), g(x)))\,\mathrm{d}x
\end{aligned}$$

Or,

$$\frac{m}{1-(1-m)^2} = \frac{1}{2-m} = \frac{1}{\int_{\mathbb{R}} g(x) + f(x) - \min(f(x), g(x))\,\mathrm{d}x}$$

Il suffit maintenant de remarquer que $g(x) + f(x) - \min(f(x), g(x)) = \max(g(x), f(x))$ pour voir que Z est bien de densité h.

(2) On suppose ici que f densité de la loi $\mathcal{N}(0,1)$ et g densité de la loi $\mathcal{N}(3/2,1)$. Écrire un programme qui simule des variables aléatoires de densité h.

```
def simuh():
    x=np.random.normal(0,1,1)
    y=np.random.normal(3/2,1,1)
    u=np.random.rand(1)
    v=np.random.rand(1)
    while ((u*np.exp(-x**2/2)>np.exp(-(x-3/2)**2/2))&(v*np.exp(-(y-3/2)**2/2)>np.exp(-y**2/2))):
        x=np.random.normal(0,1,1)
        y=np.random.normal(3/2,1,1)
        u=np.random.rand(1)
        v=np.random.rand(1)
    z=x
    if (u*np.exp(-x**2/2)<=np.exp(-(x-3/2)**2/2)):
        z=y
    return(z)


r=np.zeros(int(1E5))
for i in range(int(1E5)):
    r[i]=simuh()

# histogramme pour la loi de x
histog=plt.hist(r,100,density=True) #histogramme
index=np.min(np.where(histog[1]>=0))
absc=np.arange(-3,6,.01)
ordo=histog[0][index]*np.max(np.array([np.exp(-absc**2/2),np.exp(-(absc-3/2)**2/2)]),0)
plt.plot(absc,ordo,color='r')
```