Exercices 6 et 7 de la feuille 1

Uniforme : valeurs 1 … 10 , prises avec proba 1/10

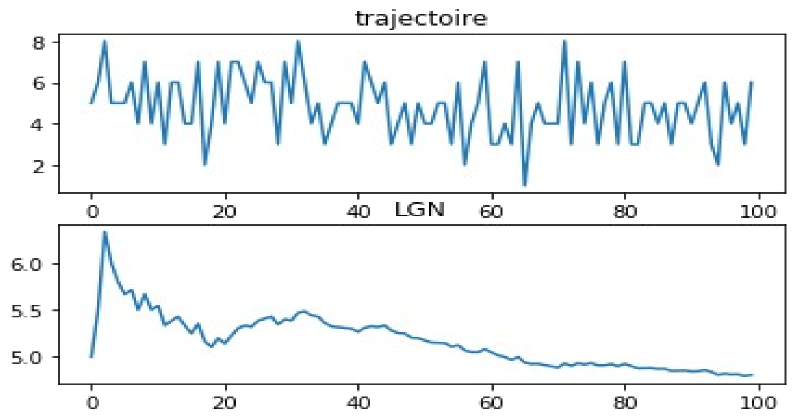
```
 X=ceil(rand(100)*10)
```

Binomiale :

```
# -*- coding:latin-1 -*-
from pylab import *

def binomperso(N,n,p): # simule une va de loi binomiale de param n,p,
                       # ech de taille N et trace ...

    res=zeros(N)
    for k in range(n):
        u=(rand(N)<p)*1.0  # simule une Bernoulli ET convertit en reels
        res=res+u     # une b(n,p) est une somme de n bernouilli indep

    sommes=cumsum(res)/range(1,N+1) # calcule les sommes cumulees

    clf()
    figure(1)
    subplot(3,1,1)
    legend('binomiale')
    plot(res)
    subplot(3,1,2)
    legend('cumulees')
    plot(sommes)
    subplot(3,1,3)
    hist(sommes)

    return res
```

Ou bien sans boucle :

```
def binomiale(n,p,N):
    U=rand(N,n)   # on simule n  echantillons de taille N de va uniformes
    V=1.0* (U<p)    # on simule les va de Bernoulli
    W=sum(V,1)         # on somme pour avoir les binomiales
    Z=cumsum(W)/(1+arange(N))   # LGN

    clf()   # les traces
    subplot(2,1,1)
    title('trajectoire')
    plot(W)
    subplot(2,1,2)
    title('LGN')
    plot(Z)
    show()
    return(W,Z)
```



Loi géométrique, loi binomiale négative

U suit une loi uniforme sur $[0,1]$. p dans $]0,1[$
X suit la loi géométrique : $P(X=1)=p$, $P(X=k)= p(1-p)^{k-1}$.
On pose, pour $n \geq 1$, S_n la somme de 1 à n des $P(X=k)$. Un calcul donne que
$S_n= 1-(1-p)^n$.
Et $S_0=0$
On divise $[0,1]$ en intervalles successifs de taille $P(X=k)$, leurs bornes seront les S_n :

$$0=S_0 < S_1=p < S_2 \dots < S_n < S_{n+1} < \dots$$

Si une coordonnée de U est entre S_n et S_{n+1}, (donc avec une probabilité $p(1-p)^n$) on renvoie le nombre $n+1$. Avec $S_0= 0$, $S_1=p$, entre 0 et p on doit renvoyer 1, par exemple.

On ne va pas faire de tests. Comment obtenir n (ou $n+1$) à partir de u qui est dans $[0,1]$ et qui est une réalisation d'une loi uniforme ?

$n \geq 0$: $1-(1-p)^n \leq u < 1-(1-p)^{n+1}$ si et seulement si .... (calcul)

$n+1 \geq \log(1-u) /\log(1-p) \geq n$.

On prend donc : n partie entière de $\log(1-u) /\log(1-p)$ **et on renvoie n+1** . Qu'on mette $<$ ou $\leq$ importe peu, une loi uniforme ne charge pas les points.

Programme :

```
# -*- coding:latin-1 -*-
from pylab import *

def geometoile(N,p): # simule un ech taille N d'une loi geom a vals ds 1,2,3, ..

    U=rand(N) # loi unif [0,1]
    G= floor(log(ones(N)-U)/log(1-p)) + ones(N)  # explications a part
        # on coupe [0,1] en une succession d'intervalles de proba p(1-p)^k-1

    return G
```

```
def binomneg(N,r,p): # loi binom neg, p dans ]0,1[, N taille ech,
            # r = nbre de contrats a placer

   init=zeros(N)
   for k in range(r):
      init=init + geometoile(N,p)

   return init
```

Ou bien sans boucle :

```
def binomneg(N,r,p):  # N taille echantillon,
   #r = nb de contrats a placer,
   #p = proba d'en placer un
   U=rand(N,r)
   X=floor(log(1.0-U)/log(1-p))+1.0
   Y=sum(X,1)
   Z= cumsum(Y)/(arange(N)+1.0)
   clf()
   plot(Z)
   show()
   return Y,Z
```

simulation : N=10 000, p= 0,22, r=5 moyenne   r/p= 22,727272,,, , trouvé 22,84