TP 8 : Correction

1. Principe de l'estimation

Introduction : On s'intéresse à une variable aléatoire X que l'on sait simuler mais dont on ne connait que partiellement la loi : on sait que sa loi fait partie d'une famille de lois $(P_\theta)_{\theta\in I}$ mais il nous manque la valeur du paramètre θ.
Pour estimer le paramètre θ, on simule alors n variables aléatoires $(X_1, ..., X_n)$ indépendantes, de même loi que X, que l'on appelle un **n-échantillon de la loi de X**, et on calcule une valeur du type $T_n = f(X_1, ..., X_n)$ que l'on pense être "proche" de la valeur θ inconnue. T_n est alors appelé un **estimateur** de θ.

Exemple : le sondage

.

1. Un estimateur naturel de la proportion p de la population française votant pour B est $\frac{\text{nombre de personnes interrogées votant pour B}}{\text{nombre total de personnes interrogées}}$, i.e. :
$$T_n = \frac{\sum_{i=1}^{n} X_i}{n}$$
2.
3.

Individu n^o i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Candidat	A	A	B	A	B	B	B	B	A	A	B	A	A	A	B	A	A	A	B	A
x_i	0	0	1	0	1	1	1	1	0	0	1	0	0	0	1	0	0	0	1	0

(a) $t_{20} = \frac{8}{20} = \frac{2}{5}$.

Exercice 1. 1.
```
de=floor(rand()*6)+1
```
ou
```
de=grand(1,1,'uin',1,6)
```
2.
```
nbr_six_succ=0
    while nbr_six_succ <>4 do
        de=grand(1,1,'uin',1,6)
        if de==6 then
            nbr_six_succ=nbr_six_succ+1
        else
            nbr_six_succ=0
        end
    end
```
3.
```
function X=nb_de_lancers()
    nbr_six_succ=0
    X=0
    while nbr_six_succ<>4 do
        de=grand(1,1,'uin',1,6)
        X=X+1
        if de==6 then
            nbr_six_succ=nbr_six_succ+1
        else
            nbr_six_succ=0
        end
    end
endfunction

disp(nb_de_lancers())
```

4.
```
S=0
for k=1:1000 do
    S= S+nb_de_lancers()
end
T=S/1000
disp(T)
```
On remarque, en relançant plusieurs fois le programme que le nombre moyens de lancers nécessaires pour obtenir 4 "six" consécutifs est toujours autour de 1560. L'estimateur est bien stable (ce qui illustre la loi faible des grands nombres : l'estimateur de Monte Carlo converge tjrs vers l'espérance) et on peut dire que l'espérance de X est proche de 1560.

Exercice 2 (: **application de la méthode de Monte-Carlo pour calculer des valeurs approchées d'intégrales**).

1. (a) D'après le théorème de transfert :
$E(V) = E[\ln(1+U^2)] = \int_{-\infty}^{+\infty} \ln(1+t^2) f_U(t)dt = \int_0^1 \ln(1+t^2)dt = I$.
   (b) La méthode de Monte-Carlo consiste à estimer $I = E(V)$ par la moyenne d'un grand nombre de simulations de la variable aléatoire V ; d'où le programme suivant :
```
U=grand (1,100 000,'unf',0,1)
V=log(1+U.^2)
f=mean(V)
disp(f)
```

2.
```
U=grand (1,100 000,'unf',0,1)
V=log(1+U.^2)
f=mean(1./(1-V))
disp(f)
```

Exercice 3. 1. La suite (X_n) est une chaîne de Markov car c'est une suite de variables aléatoires telle que la variable X_{n+1} dépend de la position de la variable X_n.
Sachant $X_n = i$, X_{n+1} suit une loi uniforme sur $\{i-1; i+1\}$.

2.
```
function x=deplacements(n)
    x=0
    for k=1:n do
        if rand() < 1/2 then
            x=x-1
        else
            x=x+1
        end
    end
endfunction
```

3.
```
e=0
for k=1:10000 do
    x=deplacements(100)
    if abs(x)>10 then
        e=e+1
    end
end
t=e/10000
disp(t)
```
La loi faible des grands nombres est bien illustrée : l'estimateur de la fréquence est stable autour d'une même valeur (la probabilité) lorsque le nombre de simulations n est proche de $+\infty$. On obtient que la probabilité de l'ivrogne soit loin de son point de départ de plus de 10 unités est d'environ $0,27$.

2. Variabilité des estimateurs autour de θ

Exercice 4 (: comparaison de 2 estimateurs). 1. `function [t,u]=estimateurs(n,theta)`

```
    x=grand(1,n,'poi',theta)
    t=(1-1/n)^sum(x)
    e=sum(x==0)
    u=e/n
endfunction
```

2.
```
theta=1
t=zeros(1,1000)
u=zeros(1,1000)
for i=1:1000 do
    [t(i),u(i)]=estimateurs(100,theta)
end

b1=mean(t)-exp(-theta)
b2=mean(u)-exp(-theta)
disp(b1, 'b1=')
disp(b2, 'b2=')
```
on obtient :
```
 b1=

    0.0000093

 b2=

  - 0.0004894
```
On remarque que le biais empirique des deux estimateurs est quasi-nul.
On peut donc conjecturer que ce sont deux estimateurs sans biais de $e^{-\theta}$. Pour le moment, on ne sait pas lequel est le meilleur estimateur.

3.
```
x=0:0.01:0.7
scf(0), clf
histplot(x,t)
scf(1), clf ,
histplot(x,u)
```
On remarque que les valeurs prises par l'estimateur U_{100} sont plus dispersées (autour de $e^{-\theta}$) que les valeurs de T_{100}. T_{100} semble donc être un meilleur estimateur de $e^{-\theta}$ que U_{100} car ses valeurs sont plus proches du paramètre à estimer.
```
-->quart(t)
 ans  =

    0.3411661
    0.3660323
    0.3927110

-->quart(u)
 ans  =

    0.33
    0.37
    0.4
```
Les quartiles de T_{100} sont les moins dispersés comme on le voyait graphiquement.

```
v1=mean((t-mean(t)).^2)
r1= v1+b1^2
disp(r1, 'r1=')

v2=mean((u-mean(u)).^2)
r2= v2+b2^2
disp(r2, 'r2=')
```
On obtient :
```
 r1=
```

```
    0.0013162

 r2=

    0.0021391
```

Le risque quadratique empirique de T_{100} est bien plus faible que celui de U_{100}. T_{100} est un meilleur estimateur que U_{100}.

Exercice 5.

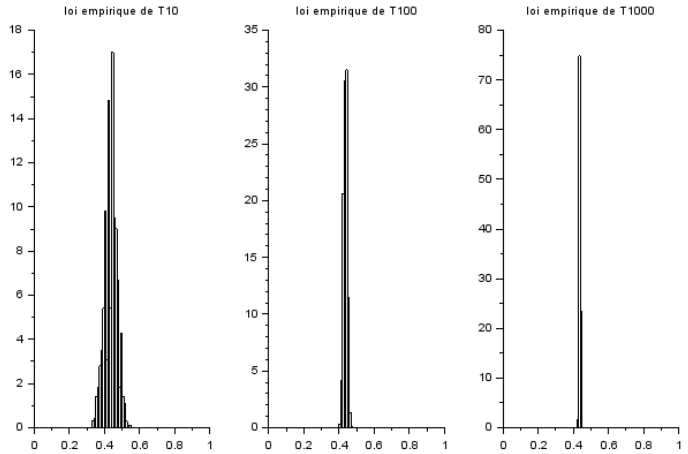
1. Les 20 envois de courriers sont indépendants et de même probabilité de succès (recevoir une réponse) de $1/4$. X compte le nombre de succès donc X suit une loi binomiale de taille 20 et de paramètre $1/4$.

   Sachant $X = i$, Y compte le nombre de réponses à $20 - i$ courriers donc suit une loi binomiale de taille $20 - i$ et de paramètre $1/4$.

2. ```
   function z=reponses()
       x=grand(1,1,'bin',20,1/4)
       y =grand(1,1,'bin',20-x,1/4)
       z=x+y
   endfunction
   ```

3. ```
   function t=estimation(n)
       z=zeros(1,n)
       for i=1:n do
           z(i)=reponses()
       end
       t=mean(z)/20
   endfunction
   ```

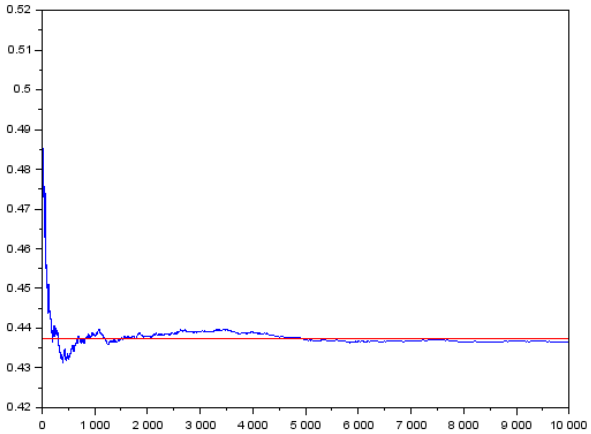
   Puis taper dans la console : `-->estimation(1000)`
   On remarque qu'on obtient toujours une estimation de θ proche de $0,435$.

4. 

   On remarque que les lois empiriques des estimateurs T_n sont de plus en plus condensées autour de θ lorsque n augmente. Le risque quadratique de la suite d'estimateurs (T_n) diminue donc lorsque n tend vers $+\infty$ et semble converger vers 0.
   La suite (T_n) converge donc en probabilité vers 0.

5. ```
   function t=suite_estimateur(n)
       z=zeros(1,n)
       t=zeros(1,n)
       for i=1:n do
           z(i)=reponses()
           t(i)=mean(z(1:i))/20
       end
   endfunction
   ```
   ```

```
scf(1), clf
t=suite_estimateur(10000)
plot(t)
plot([1 10000], [7/16 7/16], color='red')
```



La suite (T_n) converge en probabilités vers $7/16$ donc on peut conjecturer que $\theta = 7/16$.

3. Intervalle de confiances

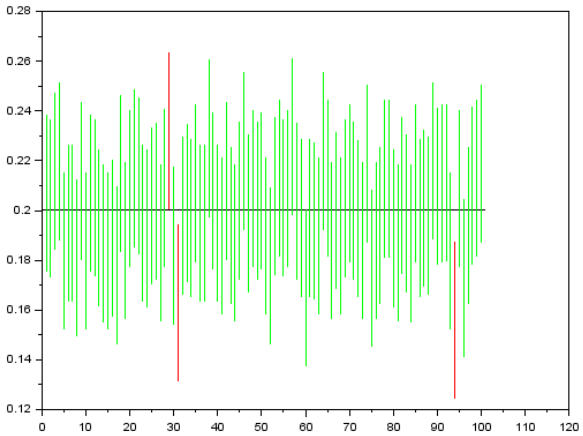
Exercice 6. Soit $X \hookrightarrow B(p)$ où p est l'inconnue à déterminer.
Soit $(X_1, ..., X_n)$ un n-échantillon de la loi de X. Nous allons vérifier informatiquement que l'intervalle $[\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]$ est un intervalle de confiance de p au niveau de confiance 95%.

1. ```
   function t=estimation_p(p,n)
       x=grand(1,n,'bin',1,p)
       t=mean(x)
   endfunction

   disp(estimation_p(0.3,1000))

   disp(estimation_p(0.3,10000))
   ```
   On remarque que l'estimateur de Monte-Carlo est proche du paramètre à estimer.
2. Ce programme trace le paramètre p à estimer en noir puis fait 100 estimations de p et vérifie pour chacun des estimateurs $\overline{X_n}$ obtenus si $p \in [\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]$ (l'intervalle de confiance) : si p appartient bien à l'intervalle de confiance, cet intervalle est tracé en vert, sinon il est tracé en rouge.

   Par exemple, pour $p = 0.2$ et $n = 1000$, on peut obtenir :


   ```

On remarque sur cet exemple que seuls 3 estimations sur 100 ne vérifient pas $p \in [\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]$ donc $P(p \in [\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]) \simeq 0,97$.
On a bien $P(p \in [\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]) \geq 0,95$ donc l'intervalle semble être un intervalle de confiance au niveau de confiance de 95%.

3.
```
p=input('entrer la valeur réelle du paramètre à estimer:' )
n=input('Entrer la longueur de l''échantillon:' )
S=0
for k=1:1000 do
    t=estimation_p(p,n)
    if p>= t-1/sqrt(n) & p<= t+1/sqrt(n) then
        S=S+1
    end
end
P=S/1000
disp(P)
```
On remarque qu'on obtient quasiment toujours une fréquence de l'évènement $p \in [\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]$ supérieure à 95% Ce qui confirme que très probablement $P(p \in [\overline{X_n} - \frac{1}{\sqrt{n}}; \overline{X_n} + \frac{1}{\sqrt{n}}]) \geq 0,95$ i.e. l'intervalle semble être un intervalle de confiance au niveau de confiance de 95%.