

CONTENTS

1 Suites	3
1.0.1 ESC 2017 - Exercice 1 (voie T)	3
1.0.2 ESCP 2016 - Exercice 1 (voie T)	3
1.0.3 ESCP 2017 - Exercice 2 (voie T)	4
1.0.4 ESCP 2017 - Exercice 3 (voie T)	5
1.0.5 ECRICOME 2016 (voie T)	5
1.0.6 ECRICOME 2015 - Exercice 1 (Voie E)	7
1.0.7 ECRICOME 2015 (Voie S)	8
1.0.8 ECRICOME 2017 - Exercice 1 (Voie S)	9
1.0.9 ECRICOME 2018 - Exercice 2 (voie S)	13
1.0.10 EML 2015 - Exercice 2 (Voie E)	14
1.0.11 EML 2016 - Exercice 3 (Voie E)	14
1.0.12 EML 2017 Exercice 1 (Voie E)	14
1.0.13 EML 2016 Problème 2 (Voie S)	15
1.0.14 EML 2017 - Problème 2 (voie S)	15
1.0.15 EDHEC 2016 - Exercice 2 (Voie E)	15
1.0.16 EDHEC 2015 - Exercice 1 (Voie S)	16
1.0.17 EDHEC 2016 - Exercice 1 (Voie S)	16
1.0.18 EDHEC 2017 - Exercice 1 (Voie S)	17
1.0.19 ECRICOME 2019 - Problème (Voie S)	17
1.0.20 EDHEC 2019 - Exercice 2 (Voie E)	18
1.0.21 ESSEC II 2016 (Voie E)	19
1.0.22 Oral HEC 2015 - Exercice principal E 77	20
2 Statistiques univariées	21
2.0.1 EDHEC 2017 - Exercice 3 (Voie E)	21
2.0.2 HECII 2017 - Partie III (voie S)	22
2.0.3 Oral HEC 2015 Exercice sans préparation E 69	24
3 Simulation de variable aléatoires discrètes	25
3.0.1 ESCP 2016 - Exercice 4 (Voie T)	25
3.0.2 ECRICOME 2017 - Exercice 2 (voie T)	26
3.0.3 ECRICOME 2015 - Exercice 3 (Voie E)	27
3.0.4 ECRICOME 2016 - Exercice 3 (Voie E)	28
3.0.5 ECRICOME 2017 - Exercice 3 (Voie E)	30
3.0.6 ECRICOME 2016 (Voie S)	32
3.0.7 ECRICOME 2017 - Problème (Voie S)	34
3.0.8 ECRICOME 2019 - Problème (Voie S)	35
3.0.9 EDHEC 2015 - Problème (Voie E)	37
3.0.10 EDHEC 2016 - Problème (Voie E)	37
3.0.11 EDHEC 2018 - Exercice 2 (Voie E)	37
3.0.12 EDHEC 2015 - Problème (Voie S)	39
3.0.13 EDHEC 2018 (épreuve annulée) - Exercice 3 (Voie S)	39
3.0.14 EDHEC 2018 (sujet Clermont) - Problème (Voie S)	40
3.0.15 EDHEC 2019 - Exercice 2 (Voie E)	41

3.0.16	Oral HEC 2015 Exercice sans préparation S 132	41
3.0.17	Oral HEC 2015 Exercice sans préparation S 143	42
3.0.18	Oral HEC 2016 Exercice sans préparation S 179	42
4	Chaînes de Markov	45
4.0.1	EDHEC 2017 - Problème (Voie E)	45
4.0.2	EDHEC 2016 - Problème (Voie S)	45
4.0.3	Oral HEC 2015 Exercice sans préparation S 140	46
5	Statistiques bivariées	47
5.0.1	HEC 2016 (voie E)	47
6	Simulation de variable aléatoires continues	49
6.0.1	EML 2015 - Exercice 1 (Voie E)	49
6.0.2	EDHEC 2016 - Exercice 3 (voie E)	49
6.0.3	EDHEC 2018 (épreuve annulée) - Problème - Partie 2 (Voie S)	50
6.0.4	EDHEC 2018 (sujet final) - Exercice 3 (Voie S)	51
6.0.5	EDHEC 2018 - Exercice 3 (Voie E)	51
6.0.6	EDHEC 2019 - Problème (Voie E)	51
6.0.7	ESSEC I 2015 (Voie E)	52
6.0.8	ESSECI 2017 (Voie E)	53
6.0.9	HEC 2015 (Voie E)	54
6.0.10	Oral HEC 2016 Exercice sans préparation S 89	55
7	Fonctions à deux variables	57
7.0.1	EDHEC 2017 - Exercice 1 (Voie E)	58
7.0.2	EDHEC 2018 (sujet Clermont) - Exercice 2 (Voie S)	59
8	Estimation	61
8.0.1	ESCP 2017 - Exercice 4 (voie T)	61
8.0.2	EML 2017 - Exercice 3 (voie E)	64
8.0.3	EDHEC 2015 - Exercice 2 (voie E)	66
8.0.4	EDHEC 2015 - Exercice 2 (Voie S)	67
8.0.5	EDHEC 2017 - Problème (Voie S)	67
8.0.6	EDHEC 2018 - Exercice 2 (Voie E)	68
8.0.7	HEC 2015 (Voie E)	69
8.0.8	HEC 2017 - Problème (voie E)	70
8.0.9	ESSEC I 2016 (Voie E)	71
8.0.10	HEC I 2017 (voie S)	71
8.0.11	Oral HEC 2016 Exercice sans préparation S 173	72
8.0.12	Oral HEC 2016 Exercice principal S 171	72
9	Autre	73
9.0.1	ECRICOME 2017 - Exercice 3 (Voie T)	73
9.0.2	EDHEC 2015 - Exercice 1 (Voie E)	73
9.0.3	EDHEC 2017 - Exercice 1 (Voie S)	73
9.0.4	EDHEC 2019 - Exercice 1 (Voie S)	74
9.0.5	EDHEC 2019 - Problème (Voie S)	75
9.0.6	HEC II-ESCP 2015 (voie S)	76

SUITES

1.0.1 ESC 2017 - Exercice 1 (voie T)

On considère les trois suites $(u_n)_{n \geq 1}$, $(v_n)_{n \geq 1}$ et $(w_n)_{n \geq 1}$ définies par $u_1 = 1$, $v_1 = 0$ et $w_1 = 2$ et, pour tout entier $n \geq 1$,

$$u_{n+1} = u_n; v_{n+1} = v_n + 2w_n \text{ et } w_{n+1} = 2u_n + w_n.$$

On admet que pour tout entier $n \geq 1$, on a : $v_n = 2n(n-1)$ et $w_n = 2n$.

On considère le programme suivant qui permet de calculer les premiers termes des suites $(u_n)_{n \geq 1}$, $(v_n)_{n \geq 1}$ et $(w_n)_{n \geq 1}$.

- Compléter la ligne 1 afin que soit mémorisé dans la variable A la matrice A.
- Pour mémoriser les termes successifs de la suite $(v_n)_{n \geq 1}$ de v_2 à v_{10} , quelle instruction parmi celles-ci faut-il ajouter en ligne 10 ? (On justifiera la réponse).
A. $v(i) = X(i)$ B. $v(i) = X$ C. $v(i) = X(2)$
D. une autre instruction à préciser
- Proposer de la même manière une instruction pour la ligne 11 qui permette de mémoriser les premiers termes de la suite $(w_n)_{n \geq 1}$.

```
A = . . . .
u=zeros(1,10)
v=zeros(1,10)
w=zeros(1,10)
u(1)=1, v(1)=0, w(1)=2
X=[1;0;2]
for i=2:10
    X=A*X
    u(i)=1
    . . . .
    . . . .
end
```

1.0.2 ESCP 2016 - Exercice 1 (voie T)

Soit $(u_n)_{n \geq 0}$ la suite définie par $u_0 = 1$ et pour tout entier $n \in \mathbb{N}$, $u_{n+1} = \ln(1 + u_n^2)$.

- Compléter le programme *Scilab* suivant pour qu'il calcule et affiche u_n , pour une valeur de n entrée par l'utilisateur.
- On considère le programme *Scilab* suivant :

```
n=input('entrer la valeur de n')
u=.....
for k=1:n
    u=.....
end
disp(u)
```

```
n=0
u=1
while u>=0.0001
    u=log(1+u^2)
    n=n+1
end
disp(n)
```

(...) On montre ensuite mathématiquement que la suite converge vers 0 (...)

On exécute ce programme. Le résultat affiché est 6.

Quelle est la signification de ce résultat ?

1.0.3 ESCP 2017 - Exercice 2 (voie T)

Soit $(u_n)_{n \geq 1}$ et $(v_n)_{n \geq 1}$ deux suites définies par :

$$u_1 = 1, v_1 = 2 \text{ et } \forall n \geq 1, u_{n+1} = \frac{u_n^2}{u_n + v_n} \text{ et } v_{n+1} = \frac{v_n^2}{u_n + v_n}.$$

Recopier et compléter les lignes (6) et (7) du programme *Scilab* suivant afin qu'il calcule et affiche les valeurs de u_n et v_n pour une valeur de n entrée par l'utilisateur.

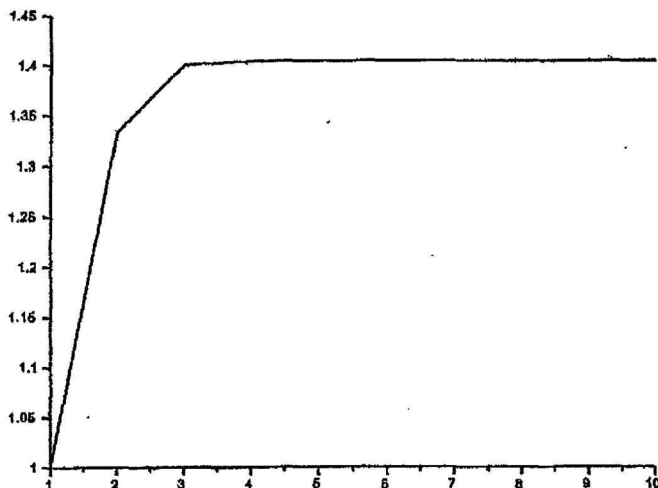
```
(1) n=input('entrer la valeur de n :')
(2) u=1
(3) v=2
(4) for k=2:n
(5)     a=u
(6)     u=.....
(7)     v=.....
(8) end
(9) disp(u)
(10) disp(v)
```

6. On considère le programme précédent avec les instructions supplémentaires :

```
(1) n=input('entrer la valeur de n :')
(2) u=1
(3) v=2
(4) s=ones(1,n)
(5) for k=2:n
(6)     a=u
(7)     u=.....
(8)     v=.....
(9)     s(k)=u
(10) end
(11) disp(u)
(12) disp(v)
(13) x=1:n
(14) y=cumsum(s)
(15) plot2d(x,y)
```

a. Que contiennent les variables s et y à l'issue du programme ?

b. Ce programme fournit la sortie graphique suivante pour la valeur $n = 10$.



Quel résultat ce graphique permet-il de conjecturer ?

1.0.4 ESCP 2017 - Exercice 3 (voie T)

On pose $I_0 = \int_0^1 e^{-x^2} dx$ et pour tout entier $n \geq 1$, $I_n = \int_0^1 x^n e^{-x^2} dx$.

On admet que $I_1 = \frac{1}{2} - \frac{1}{2e}$ et que pour tout entier $k \geq 1$, on a : $I_{2k+1} = kI_{2k-1} - \frac{1}{2e}$.

À l'aide de cette relation et de la valeur de I_1 , compléter le script *Scilab* suivant afin qu'il calcule et affiche la valeur de I_{2n+1} pour une valeur de n entrée par l'utilisateur.

```
(1) n=input('entrer la valeur de n :')
(2) I=1/2-1/(2*%e)
(3)   for k=1:n
(4)     I=.....
(5) end
(6) disp(I)
```

1.0.5 ECRICOME 2016 (voie T)

EXERCICE 2

On a montré qu'une suite de réels $(I_n)_{n \in \mathbb{N}}$ était définie par :

$$I_{n+1} = \frac{1}{e} + (n+1)I_n \quad \text{et} \quad I_0 = \frac{1}{e}$$

Écrire un script Scilab qui, demandant à l'utilisateur un entier strictement positif n , calcule et affiche la valeur de I_n .

EXERCICE 3

On a montré que la loi d'une VA X était donnée par :

$$X(\Omega) = \llbracket 3; +\infty \llbracket \quad \forall k \geq 3, \quad P(X = k) = \frac{1}{2} \left[\left(\frac{4}{5} \right)^{k-2} - \left(\frac{2}{3} \right)^{k-2} \right]$$

1. Compléter ce script SCILAB afin qu'il calcule et affiche les probabilités $P(X = k)$ pour $k \in \llbracket 3, 22 \rrbracket$.

```
for k = 3 : 22
    u(k) = .....
end
plot(u, '+')
```

Le graphique obtenu en sortie est présent sur la figure 1 (page suivante).
Quelle semble être la valeur la plus probable de X ?

2. Quelle instruction peut-on écrire dans la Console SCILAB qui affiche les vingt valeurs de la fonction de répartition de X aux points d'abscisses entières comprises entre 3 et 22.

Le graphique obtenu en sortie est présent sur la figure 2 (page suivante).
Déterminer graphiquement un réel m tel que $P(X \leq m) = P(X \geq m)$.

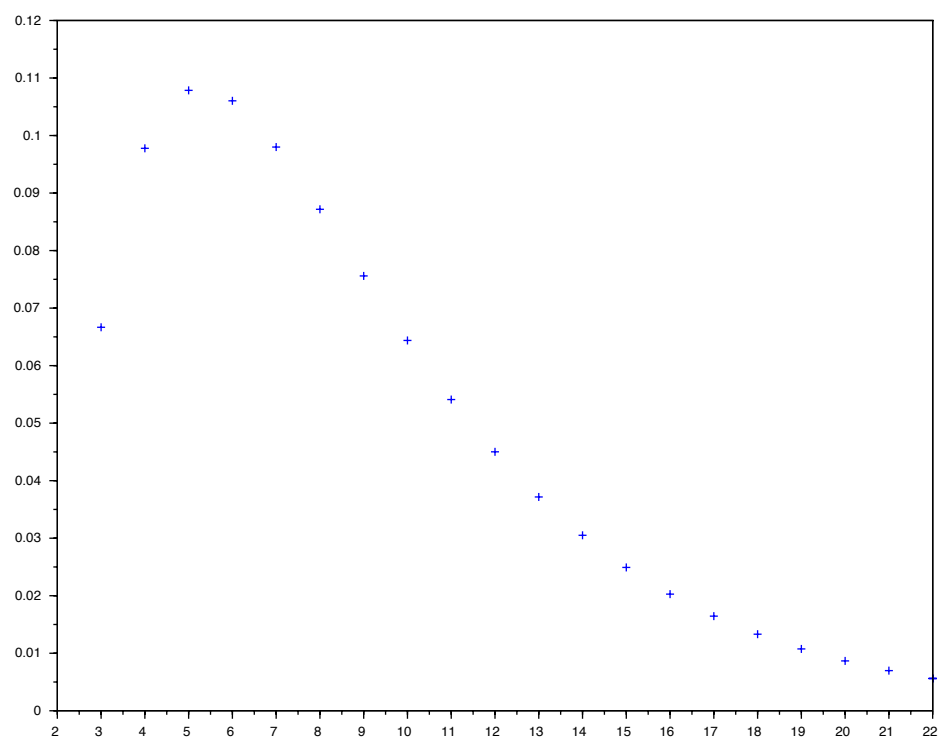


Figure 1

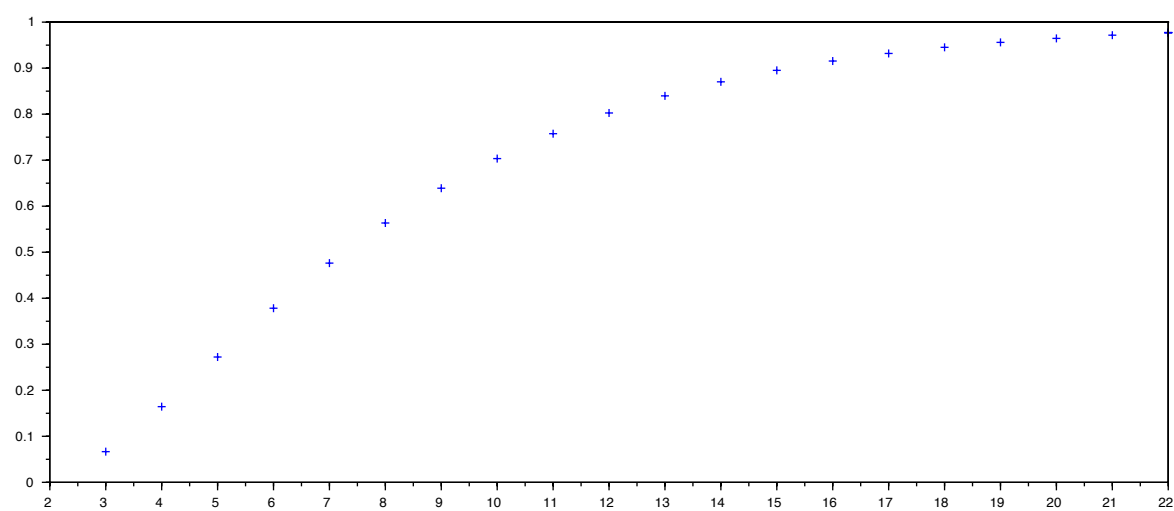


Figure 2

1.0.6 ECRICOME 2015 - Exercice 1 (Voie E)

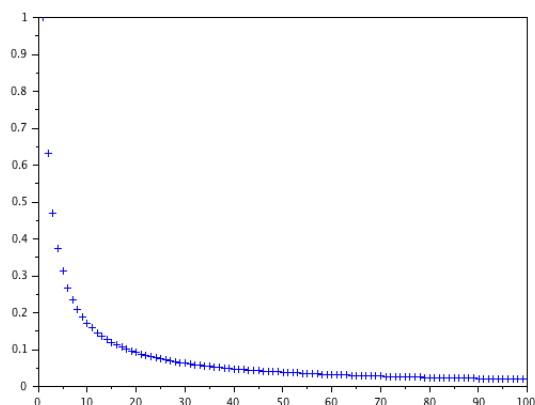
EXERCICE 1.1 On considère la suite $(u_n)_{n \geq 1}$ définie par $u_1 = 1$ et pour tout entier naturel non nul n par : $u_{n+1} = F(u_n)$. Avec

$$\forall x \in \mathbb{R}, \quad F(x) = \begin{cases} 0 & \text{si } x < 0, \\ 1 - e^{-x} & \text{si } x \geq 0. \end{cases}$$

2.c) Recopier et compléter le programme SCILAB suivant qui permet de représenter les cent premiers termes de la suite $(u_n)_{n \geq 1}$:

```
U = zeros(1,100)
U(1) = 1
for n = 1 : 99
    U(n+1) = -----
end
plot(U, "+")
```

2.d) Le programme précédent complété permet d'obtenir la représentation graphique suivante :

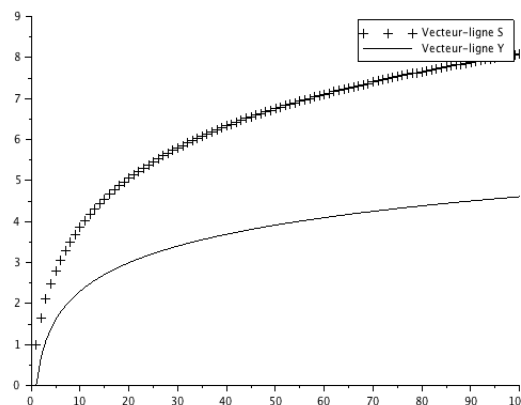


Quelle conjecture pouvez-vous émettre sur la monotonie et la limite de la suite $(u_n)_{n \geq 1}$?

2.i) On modifie le programme écrit en question 2(c) en remplaçant la dernière ligne par :

```
X = 1:100
S = cumsum(U)
Y = log(X)
plot2d(X,S)
plot2d(X,Y)
```

Le programme ci-dessus permet d'obtenir la représentation graphique suivante :



Que représente le vecteur-ligne S ?

Quelle conjecture pouvez-vous émettre sur la nature de la série de terme général u_n ?

1.0.7 ECRICOME 2015 (Voie S)

On définit pour tout entier naturel n , les suites $(a_n)_{n \in \mathbb{N}}$ et $(b_n)_{n \in \mathbb{N}}$ par $a_0 = \sqrt{\frac{3}{2}}, b_0 = \frac{1}{2}$ et :

$$a_{n+1} = \sqrt{\frac{1-b_n}{2}} \quad (*) \quad \text{et} \quad b_{n+1} = \sqrt{\frac{1+b_n}{2}} \quad (**)$$

On admet que pour tout $n \in \mathbb{N}$,

$$9 \times 2^n \frac{a_n}{2+b_n} < \pi < 2^n \left(2a_n + \frac{a_n}{b_n} \right).$$

On admet enfin que $a_n \sim \frac{\pi}{2^n}$ et $b_n \sim 1$

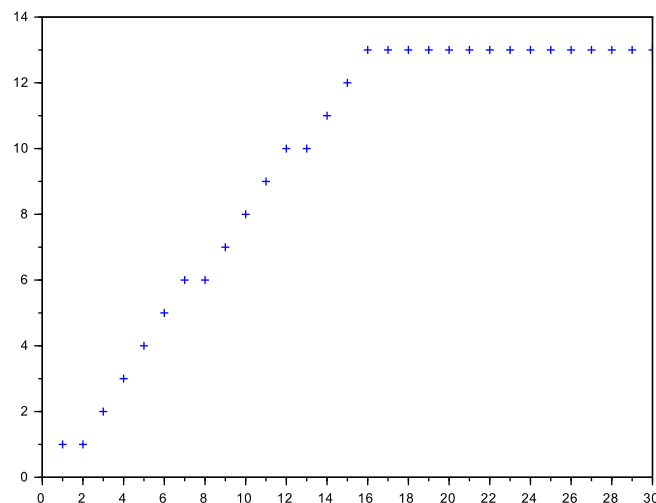
3.c) Justifier que les deux termes de l'encadrement précédent tendent vers π quand n tend vers l'infini.

3.d) Compléter la fonction Scilab suivante afin qu'elle retourne, à l'aide des relations (*) et (**) et de la question 3(b), une approximation x de π à e près, ainsi que le nombre k d'itérations qui ont été nécessaires.

```
function [x,k]=h(e)
    k = 0
    a = sqrt(3) / 2
    b = 1 / 2
    while -----
        a = -----
        b = -----
        k = -----
    end
    x = -----
endfunction
```

3.e) On souhaite étudier l'évolution du nombre d'itérations nécessaires en fonction de la précision souhaitée. Écrire une fonction Scilab qui prend comme paramètre d'entrée un entier p et qui retourne un vecteur de taille p qui contient les nombres d'itérations nécessaires pour les précisions 10^{-k} , pour $k \in \{1, 2, \dots, p\}$.

3.f) On utilise la fonction précédente avec $p = 30$ et on représente graphiquement les valeurs obtenues. On obtient le graphe suivant :



Commenter ce graphe.

1.0.8 ECRICOME 2017 - Exercice 1 (Voie S)

EXERCICE 1.2

1. Pour tout $n \in \mathbb{N}$, on pose :

$$u_n = \frac{(-1)^n}{(n+1)^{n+1}}$$

et :

$$S_n = \sum_{k=0}^n u_k.$$

- Montrer que la série de terme général u_n est convergente.
- Écrire une fonction Scilab d'en-tête `function S = somme(n)` qui prend en paramètre d'entrée un entier naturel n et qui produit en paramètre de sortie la valeur de S_n .

2. On note $I = \int_0^1 t^t dt$ et on admet que

$$\forall n \in \mathbb{N}, \quad |I - S_n| \leq \frac{1}{e^{n+1}(n+1)!}$$

Écrire une fonction d'en-tête `function I = estimation(eps)` qui prend comme paramètre d'entrée un réel flottant strictement positif ε et qui produit en paramètre de sortie une valeur approchée de I à ε près.

ECRICOME 2019 - Exercice 1 (Voie T)

Une entreprise de restauration collective propose trois formules de repas à n clients, où n est un entier naturel non nul.

Chacun des clients choisit exactement **une** de ces trois formules au hasard et de façon équiprobable puis envoie un bon de commande à l'entreprise. L'entreprise réceptionne alors pour chacun de ses n clients son bon de commande et lui livre la formule choisie.

On suppose que les choix de formule des clients sont mutuellement indépendants.

Pour tout entier naturel k tel que $1 \leq k \leq n$, on note :

- A_k l'événement : "après réception du $k^{\text{ième}}$ bon, une seule formule a été choisie" ;
- B_k l'événement : "après réception du $k^{\text{ième}}$ bon, exactement deux formules ont été choisies" ;
- C_k l'événement : "après réception du $k^{\text{ième}}$ bon, les trois formules ont été choisies".

Enfin, on pose pour tout entier k tel que $1 \leq k \leq n$: $a_k = P(A_k)$, $b_k = P(B_k)$ et $c_k = P(C_k)$.

et on admet pour tout entier naturel n non nul, les égalités :
$$\begin{cases} a_n = \frac{1}{3^{n-1}} \\ b_n = \frac{2^n - 2}{3^{n-1}} \\ c_n = 1 - \frac{2^n - 1}{3^{n-1}} \end{cases}$$
 On considère le script **Scilab**

suivant :

```
n = 1
c = 1 - (2^n - 1) / 3^(n-1)
while c < 0.95
    n = n + 1
    c = 1 - (2^n - 1) / 3^(n-1)
end
disp(n)
```

Après exécution, on obtient l'affichage suivant :

11.

Interpréter le résultat dans le contexte de l'énoncé.

ECRICOME 2019 - Exercice 2 (Voie T)

Soit g la fonction numérique définie sur l'intervalle $]0; +\infty[$ par :

$$g(x) = 2x - 1 + \ln\left(\frac{x}{x+1}\right).$$

On admet que que l'équation $g(x) = 0$ admet une unique solution α sur l'intervalle $]0; +\infty[$.

1. Recopier et compléter l'algorithme de dichotomie ci-dessous écrit à l'aide de **Scilab** afin qu'il affiche un encadrement de α à 10^{-2} près lorsque a et b sont choisis tels que $\alpha \in [a, b]$.

```

function y=g(x)
    y = .....
endfunction
a = input('Entrer la valeur de a : ')
b = input('Entrer la valeur de b : ')
while b - a .....
    m = .....
    if g(a)*g(m) <= 0 then
        b = .....
    else
        .....
    end
end
end
disp(.....)

```

2. On considère la suite $(u_n)_{n \geq 1}$ définie pour tout entier naturel n non nul par :

$$u_n = (2n - 1) - g(n).$$

Le script Scilab ci-dessous construit un vecteur ligne u contenant les 50 premiers termes de la suite $(u_n)_{n \geq 1}$.

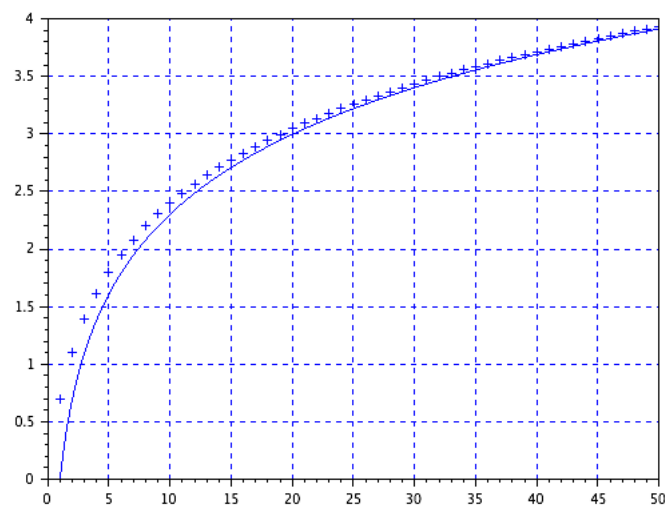
```

u = zeros(1,50)
for n =1:50
    u(n) = (2*n-1)-g(n)
end
S = cumsum(u)
plot(1:50,S,'+')

```

Dans ce script, g désigne la fonction g dont le code a été complété à la question 4(b).

On exécute le script précédent et on obtient le graphique ci-dessous. Sur ce graphique, on a aussi tracé la courbe représentative de la fonction logarithme népérien \ln en trait plein.



Interpréter le contenu du vecteur ligne S dans le contexte de l'énoncé. En notant pour tout

entier $n \geq 1$, $S_n = \sum_{k=1}^n u_k$, que peut-on conjecturer à l'aide du graphique précédent sur la limite de la suite $(S_n)_{n \geq 1}$?

ECRICOME 2018 - Exercice 1 (Voie E)

EXERCICE 1.3 Soit A et B les matrices de $\mathcal{M}_3(\mathbb{R})$ donnée par : $A = \begin{pmatrix} 2 & 1 & -2 \\ 0 & 3 & 0 \\ 1 & -1 & 5 \end{pmatrix}$ $B = \begin{pmatrix} 1 & -1 & -1 \\ -3 & 3 & -3 \\ -1 & 1 & 1 \end{pmatrix}$

On pose $X_0 = \begin{pmatrix} 3 \\ 0 \\ -1 \end{pmatrix}$, $X_1 = \begin{pmatrix} 3 \\ 0 \\ -2 \end{pmatrix}$, et pour tout entier naturel n : $X_{n+2} = \frac{1}{6}AX_{n+1} + \frac{1}{6}BX_n$.

On notera $X_n = \begin{pmatrix} \alpha_n \\ \beta_n \\ \gamma_n \end{pmatrix}$, et on admettra que :

$$\alpha_n = \frac{4}{3} + \frac{2}{3} \left(-\frac{1}{2}\right)^n + \left(\frac{1}{2}\right)^{n-1} + \left(-\frac{1}{2}\right) + \frac{3}{2} \left(-\frac{1}{3}\right)^n$$

$$\beta_n = \left(\frac{1}{2}\right)^{n-1} - \frac{2}{3} \left(-\frac{1}{2}\right)^n - \frac{4}{3}$$

1. Compléter la fonction ci-dessous qui prend en argument un entier n supérieur ou égal à 2 et qui renvoie la matrice X_n :

```
function res=X(n)
    Xold=[3;0;-1]
    Xnew=[3;0;-2]
    A=[2,1,-2;0,3,0;1,-1,5]
    B=[1,-1,-1;-3,3,-3;-1,1,1]
    for i=2:n
        Aux= .....
        Xold=.....
        Xnew=.....
    end
    res=.....
endfunction
```

2. La fonction précédente a été utilisée dans un script permettant d'obtenir graphiquement (voir figure 1) les valeurs de α_n , β_n et γ_n en fonction de n .
Associer chacune des trois représentations graphiques à chacune des suites $(\alpha_n)_{n \in \mathbb{N}}$, $(\beta_n)_{n \in \mathbb{N}}$, $(\gamma_n)_{n \in \mathbb{N}}$ en justifiant votre réponse.

1.0.9 ECRICOME 2018 - Exercice 2 (voie S)

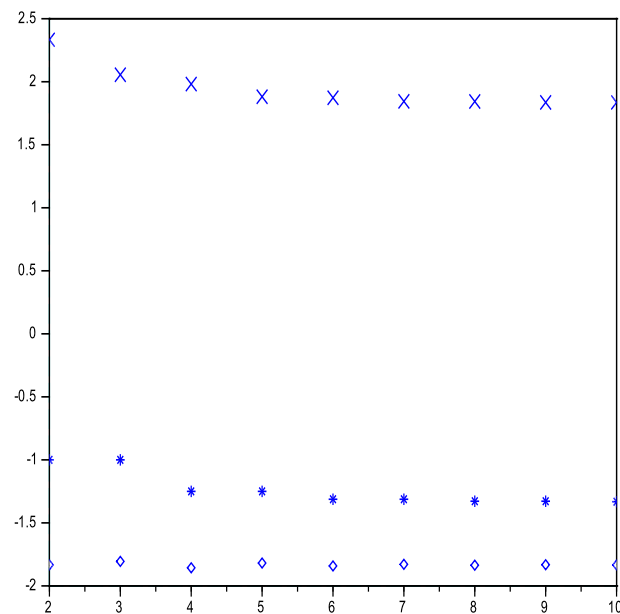
EXERCICE 1.4 On considère la suite $(u_n)_{n \geq 0}$ définie par :

$$u_0 = 0, \quad u_1 = 1 \quad \text{et} \quad \forall n \in \mathbb{N}, \quad u_{n+2} = u_{n+1} + u_n,$$

Recopier et compléter la fonction Scilab suivante afin que, prenant en argument un entier $n \geq 2$, elle calcule et renvoie la valeur du terme u_n de la suite $(u_n)_{n \geq 0}$.

```
function u=suite(n)
    v=0
    w=1
    for k=2:n
        .....
        .....
        .....
    end
    u=.....
endfunction
```

Figure 1.1:



ECRICOME 2018 - Exercice 2 (Voie E)

EXERCICE 1.5 Pour tout entier naturel n non nul, on pose : $u_n = \sum_{k=1}^n \frac{1}{k} - \ln(n)$ et $v_n = u_n - \frac{1}{n}$.

On admet que ces deux suites convergent vers la même limite γ et que $\forall n \in \mathbb{N}^* \quad v_n \leq \gamma \leq u_n$ puis que $\forall n \in \mathbb{N}^* \quad |u_n - \gamma| \leq \frac{1}{n}$

1. Écrire une fonction d'en-tête : `function y=u(n)` qui prend en argument un entier naturel n non nul et qui renvoie la valeur de u_n .
- 2.
3. On rappelle que l'instruction `floor(x)` renvoie la partie entière d'un réel x et on suppose que la fonction u de la question 1) e: a été correctement programmée. Expliquer l'intérêt et le fonctionnement du script ci-dessous :

```
eps=input('Entrer un réel strictement positif : ')
n=floor(1/eps)+1
disp(u(n))
```

ECRICOME 2019 - Exercice 1 (Voie S)

EXERCICE 1.6

On considère la suite $(I_n)_{n \geq 0}$ définie par :

$$\forall n \in \mathbb{N}, \quad I_n = \int_0^{\frac{\pi}{2}} (\cos t)^n dt$$

On admet que $\forall n \in \mathbb{N}, I_{n+2} = (n+1)(I_n - I_{n+2})$ et que :

$$\forall n \in \mathbb{N}, I_{2n} = \frac{(2n)!}{(2^n n!)^2} \frac{\pi}{2} \text{ et } I_{2n+1} = \frac{(2^n n!)^2}{(2n+1)!}$$

1. Compléter la fonction I suivante, qui prend en entrée un entier positif n , afin qu'elle retourne un vecteur y qui contient les $2n + 2$ premiers termes de la suite $(I_n)_{n \geq 0}$.

```
function y=I(n)
    u=zeros(1,.....)
    u(0)=.....
    u(1)=.....
    for k=1:n
        .....
    end
    y=u
endfunction
```

2. Ecrire une fonction en Scilab qui prend en entrée un entier naturel n et qui renvoie en sortie le terme de rang n de la suite des sommes partielles associée à la série $\sum_{n \geq 0} I_n$

1.0.10 EML 2015 - Exercice 2 (Voie E)

EXERCICE 1.7 On considère l'application $f : \mathbb{R} \rightarrow \mathbb{R}, x \mapsto f(x) = x^3 e^x$,

On admet que la série $\sum_{n \geq 1} \frac{1}{f(n)}$ converge. On note $S = \sum_{n=1}^{+\infty} \frac{1}{f(n)}$.

Et on admet que $\forall n \in \mathbb{N}^*, \left| S - \sum_{k=1}^n \frac{1}{f(k)} \right| \leq \frac{1}{(e-1)e^n}$.

En déduire une fonction en Scilab qui calcule une valeur approchée de S à 10^{-4} près.

1.0.11 EML 2016 - Exercice 3 (Voie E)

On considère l'application $f : [0; +\infty[\rightarrow \mathbb{R}$ définie, pour tout t de $[0; +\infty[$, par :

$$f(t) = \begin{cases} t^2 - t \ln(t) & \text{si } t \neq 0 \\ 0 & \text{si } t = 0 \end{cases}$$

On considère la suite $(u_n)_{n \in \mathbb{N}}$ définie par : $u_0 = \frac{1}{2}$ et $\forall n \in \mathbb{N}, u_{n+1} = f(u_n)$.

On admet que cette suite converge vers 1.

Écrire un programme en Scilab qui calcule et affiche un entier naturel N tel que $1 - u_N < 10^{-4}$.

1.0.12 EML 2017 Exercice 1 (Voie E)

On considère la fonction $f :]0; +\infty[\rightarrow \mathbb{R}$ définie, pour tout x de $]0; +\infty[$, par :

$$f(x) = e^x - e \ln(x)$$

On considère la suite réelle $(u_n)_{n \in \mathbb{N}}$ définie par :

$$u_0 = 2 \quad \text{et, pour tout } n \text{ de } \mathbb{N}, \quad u_{n+1} = f(u_n)$$

On admet que la suite $(u_n)_{n \in \mathbb{N}}$ admet $+\infty$ pour limite.

Ecrire un programme en Scilab qui, étant donné un réel A , renvoie un entier naturel N tel que $u_N \geq A$.

1.0.13 EML 2016 Problème 2 (Voie S)

PARTIE I : Étude d'une fonction définie par la somme d'une série

On s'intéresse dans cette partie, pour tout $x \in \mathbb{R}$, à la série $\sum_{n \in \mathbb{N}^*} \frac{(-1)^{n+1}}{n^x}$.

On admet qu'elle converge et que l'on a : $S(x) = \sum_{n=1}^{+\infty} \frac{(-1)^{n+1}}{n^x}$.

et on admet enfin que $\forall n \in \mathbb{N}^*, |S(x) - u_n| \leq \frac{1}{(n+1)^x}$.

2.f) En déduire une fonction Scilab qui, étant donnés deux réels $x > 0$ et $\varepsilon > 0$, renvoie une valeur approchée de $S(x)$ à ε près.

1.0.14 EML 2017 - Problème 2 (voie S)

EXERCICE 1.8 On définit la fonction réelle H d'une variable réelle x par : $H(x) = \int_0^{+\infty} \frac{1}{(1+t^2)^x} dt$.

Dans tout le problème, I désigne l'intervalle $]\frac{1}{2}; +\infty[$. On admet que H est définie sur I

1. Calculer $H(1)$.
2. Soit $n \in \mathbb{N}^*$. Montrer, à l'aide d'une intégration par parties : $H(n) = 2n(H(n) - H(n+1))$.
En déduire une expression de $H(n+1)$ en fonction de n et de $H(n)$.
3. Écrire un programme Scilab qui, étant donné un entier n de \mathbb{N}^* , renvoie la valeur de $H(n)$.

1.0.15 EDHEC 2016 - Exercice 2 (Voie E)

Soit $(u_n)_{n \in \mathbb{N}}$ une suite de réels dont on sait que $\lim_{n \rightarrow +\infty} u_n = +\infty$. et $\forall n \in \mathbb{N}, e^{-\sqrt{u_n}} \leq u_n - n \leq e^{-\sqrt{n}}$.

Compléter les commandes Scilab suivantes afin qu'elles permettent d'afficher un entier n pour lequel $u_n - n$ est inférieur ou égal à 10^{-4} .

```
n=0
while -----
    n= -----
end
disp(n)
```

Le script ci-dessus affiche l'une des trois valeurs $n = 55$, $n = 70$ et $n = 85$. Préciser laquelle en prenant 2,3 comme valeur approchée de $\ln 10$.

1.0.16 EDHEC 2015 - Exercice 1 (Voie S)

Étant donné deux suites réels $(I_n)_{n \in \mathbb{N}}$ et $(J_n)_{n \in \mathbb{N}}$ vérifiant :

$$I_n + I_{n+1} = \frac{1}{n+2} \quad J_n + J_{n+1} = I_{n+1} \quad J_0 = \frac{1}{2} \quad I_1 = \ln(2)$$

compléter les commandes Scilab suivantes afin qu'elles permettent le calcul de I_n et J_n pour une valeur de n , supérieure ou égale à 2, entrée par l'utilisateur.

```
n = input('entrer une valeur de n superieure ou egale a 2 : ')
I = Log(2) ; J=1/2 ; J = -----
for k = 2 : n
    I = ----- ;
    J = ----- ;
end
disp(I, 'la valeur de I est : ')
disp(J, 'la valeur de J est : ')
```

1.0.17 EDHEC 2016 - Exercice 1 (Voie S)

On considère la fonction f définie sur \mathbb{R}_+^* par : $\forall x \in \mathbb{R}_+^*, f(x) = \frac{e^{-x}}{x}$.

On considère également la suite $(u_n)_{n \in \mathbb{N}}$ définie par $u_0 = 1$ et par la relation $u_{n+1} = f(u_n)$, valable pour tout entier naturel n .

- Dresser le tableau de variation de f , limites comprises.
 - Vérifier que chaque terme de la suite $(u_n)_{n \in \mathbb{N}}$ est parfaitement défini et strictement positif.
- Les scripts suivants renvoient, pour celui de gauche, la valeur 5, et celui de droite, la valeur 6. Que sait-on de u_5 et u_6 ? Quelle conjecture peut-on émettre sur le comportement de la suite $(u_n)_{n \in \mathbb{N}}$?

```
u=1
n=0
while u>0.00001
    u = exp(-u)/u
    n=n+1
end
disp(n)
```

```
u=1
n=0
while u<100000
    u = exp(-u)/u
    n=n+1
end
disp(n)
```


1.0.18 EDHEC 2017 - Exercice 1 (Voie S)

EXERCICE 1.9 Soit n un entier naturel supérieur ou égal à 1. On considère la fonction f_n définie par :

$$\forall x \in [0, 1], f_n(x) = \sum_{k=1}^n x^k$$

1. (a) Compléter la fonction `Scilab` suivante pour qu'elle renvoie la valeur de $f_n(x)$ à l'appel de `f(x, n)`, où x et n sont donnés par l'utilisateur.

```
function y=f(x,n)
    y=sum(-----)
endfunction
```

- (b) Transformer, pour $x \neq 1$, l'expression de $f_n(x)$ puis en déduire une deuxième façon de déclarer `f`, en complétant la déclaration suivante où la fonction est toujours nommée `f`.

```
function y=f(x,n)
    if x==1 then y=-----
    else y=-----
    end
endfunction
```

2. On admet que l'équation $f_n(x) = 1$, d'inconnue $x \in [0, 1]$, possède une unique solution α_n dans $[0, 1]$.

On suppose que f_n a été déclarée (cf question 1). On considère les commandes supplémentaires suivantes :

```
n=input('entrer la valeur de n : ')
x=0
while f(x,n)<1
    x=x+0.001
end
disp(x)
```

Quel est le lien entre le résultat affiché et α_n ?

1.0.19 ECRICOME 2019 - Problème (Voie S)

EXERCICE 1.10

Une urne contient initialement une boule, blanche et une boule noire. On effectue une succession de tirages d'une boule dans cette urne. Après chaque tirage, on remet la boule tirée dans l'urne, et on rajoute dans l'urne une boule de couleur opposée à celle qui vient d'être tirée.

On suppose que cette expérience est modélisée par un espace probabilisé $(\Omega, \mathcal{T}, \mathbb{P})$

Pour tout $k \in \mathbb{N}$, on note X_k le nombre de boules blanches présentes dans l'urne juste avant le $(k+1)$ -ième tirage. En particulier, on a $X_0 = 1$. On admet que pour tout entier k , X_k est une variable aléatoire de $(\Omega, \mathcal{T}, \mathbb{P})$.

On admet que :

$$\forall k \in \mathbb{N}, \forall i \in \mathbb{N}^*, P([X_{k+1} = i]) = \frac{i}{k+2} \mathbb{P}([X_k = i]) + \frac{3+k-i}{k+2} P([X_k = i-1]) \quad (*)$$

et on admet que pour tout $k \in \mathbb{N}$:

$$P([X_k = 1]) = P([X_k = k+1]) = \frac{1}{(k+1)!}$$

1. Que renvoie la fonction Scilab suivante pour un entier k non nul ?

Détailler le fonctionnement de la ligne 5.

```
1 function x=mystere(k)
2   n=1;
3   b=1;
4   for i=1:k
5       r=floor(rand()*(n+b)+1)
6       if r>n then
7           n=n+1
8       else
9           b=b+1
10      end
11  end
12  x=b
13 endfunction}
```

9. Écrire une fonction Scilab d'en-tête `function LE = loi-exp(k,N)` qui prend en entrée un entier strictement positif N , qui effectue N simulations de k tirages successifs dans l'urne et qui retourne un vecteur `LE` qui contient une estimation de la loi de X_k (c'est-à-dire que pour chaque $i \in \llbracket 1, k+1 \rrbracket$, `LE(i)` contient la fréquence d'apparition de l'événement $[X_k = i]$ au cours des N simulations).

On pourra utiliser la fonction `mystere`.

10. Recopier et compléter la fonction `loi-theo` suivante, qui prend en entrée un entier strictement positif n , afin qu'elle retourne un vecteur `LT` qui contient la loi théorique de X_n .

```
function LT=loi-theo(n)
2   M=zeros(n,n+1)
3   M(1,1)=1/2
4   M(1,2)=1/2
5   for k=1:n-1
6       M(k+1,1)=.....
7       for i=2:k+1
8           M(k+1,i)=.....
9       end
10      M(k+1,k+2)=.....
11  end
12  LT=.....
13 endfunction
```

11. Un étudiant nous propose comme loi de X_5 le résultat suivant :

k	1	2	3	4	5	6
$P(X_5 = k)$	0.001368	0.079365	0.419434	0.418999	0.079454	0.00138

A-t-il utilisé `loi-exp` ou bien `loi-theo` ?

1.0.20 EDHEC 2019 - Exercice 2 (Voie E)

EXERCICE 1.11 Soit n un entier naturel supérieur ou égal à 3.

Une urne contient une boule noire non numérotée et $n-1$ boules blanches, dont $n-2$ portent le numéro 0 et une porte le numéro 1. On extrait ces boules au hasard, une à une, sans remise, jusqu'à l'apparition de la boule noire. Pour chaque i de $\llbracket 1, n-1 \rrbracket$, on note B_i l'événement : "le i -ème tirage donne une boule blanche", on pose $\overline{B_i} = N_i$ et on note X la variable aléatoire égale au rang d'apparition de la boule noire.

On rappelle qu'en Scilab, la commande `grand(1,1,'uin',a,b)` simule une variable aléatoire suivant la loi uniforme sur $\llbracket a, b \rrbracket$.

1. Compléter le script `Scilab` suivant afin qu'il simule l'expérience aléatoire décrite dans cet exercice et affiche la valeur prise par la variable aléatoire X .
On admettra que la boule noire est codée tout au long de ce script par le nombre $nB+1$, où nB désigne le nombre de boules blanches.

```

1  n=input('entrez une valeur pour n : ')
2  nB=n-1
3  X=1
4  u=rand(1,1,'uin',1,nB+1)
5  while u<nB+1
6      nB=-----
7      u=rand(1,1,'uin',1,-----)
8      X=-----
9  end
10 disp(X,'la boule noire est apparue au tirage numéro')
```

2. Compléter les lignes 4 et 8 ajoutées au script précédent afin que le script qui suit renvoie et affiche, en plus de celle prise par X , la valeur prise par Y .

```

1  n=input('entrez une valeur pour n : ')
2  nB=n-1
3  X=1
4  Y=-----
5  u=rand(1,1,'uin',1,nB+1)
6  while u<nB+1
7      nB=-----
8      if u==1 then Y=-----
9      end
10     u=rand(1,1,'uin',1,-----)
11     X=-----
12 end
13 disp(X,'la boule noire est apparue au tirage numéro')
14 disp(Y,'la valeur de Y est')
```

1.0.21 ESSEC II 2016 (Voie E)

Soit $(u_n)_{n \in \mathbb{N}}$ et $(p_n)_{n \in \mathbb{N}^*}$ deux suites vérifiant Pour tout entier naturel n non nul :

$$u_n = u_{n-1}p_1 + \dots + u_0p_n \quad \text{et} \quad u_0 = 1$$

En Scilab, soit $P = [p_1, p_2, \dots, p_n]$ le vecteur ligne tel que $P(j) = p_j$ pour j dans $\llbracket 1, n \rrbracket$. Écrire un programme en Scilab qui calcule u_n à partir de P .

1.0.22 Oral HEC 2015 - Exercice principal E 77

1. Question de cours : Donner les critères de convergence des séries à termes positifs.

Soit f la fonction définie sur \mathbb{R}_+^* par $f(x) = \ln\left(\frac{e}{2}\left(x + \frac{1}{x}\right)\right)$.

On note \mathcal{C} la courbe représentative de f dans le plan rapporté à un repère orthonormé.

2. Dresser le tableau de variation de la fonction f .
3. (a) Montrer que la courbe (Γ) d'équation $y = \ln\left(\frac{e}{2}x\right)$ est asymptote à (\mathcal{C}) .
 (b) Tracer (\mathcal{C}) et (Γ) dans le même repère.
4. Établir pour tout réel $x \geq 1$ l'encadrement : $0 \leq f'(x) < 1$.
 En déduire le signe de $f(x) - x$ pour tout $x \geq 1$ ainsi que la position de (\mathcal{C}) par rapport à la droite (\mathcal{D}) d'équation $y = x$.

5. Soit le programme Scilab suivant :

```
function y=f(x)
    y=log(%e*(x+x.^(-1))/2)
endfunction

x=[0.01:0.1:5]
plot2d(x,f(x),rect=[0,0,5,5])

x=[0,5]
plot2d(x,x)

u=input('u0=')
x=[u];y=[0]
for k=1:10
    z=f(u)
    x=[x,u]
    x=[x,z]
    y=[y,z,z]
    u=z
end
plot2d(x,y)
```

Expliquer ce que fait ce programme et ce qu'il illustre.

Dans `plot2d, rect[0,0,5,5]` signifie que seule la partie de la courbe contenue dans le rectangle $\{(x,y)/0 \leq x \leq 5 \text{ et } 0 \leq y \leq 5\}$ sera tracé.

6. Étudier la suite $(u_n)_{n \in \mathbb{N}}$ définie par : $u_0 \in [1, +\infty[$ et $\forall n \in \mathbb{N}, u_{n+1} = f(u_n)$.
7. (a) Justifier l'existence d'un réel $a > 1$ tel que $x \in [1, a] \iff f'(x) \leq \frac{1}{2}$.
 (b) On pose $\forall n \in \mathbb{N}, v_n = u_n - 1$. Quelle est la nature de la série de terme général v_n ?

STATISTIQUES UNIVARIÉES

2.0.1 EDHEC 2017 - Exercice 3 (Voie E)

Soit V une variable aléatoire suivant la loi exponentielle de paramètre 1.

On désigne par n un entier naturel non nul et par X_1, \dots, X_n des variables aléatoires définies sur le même espace probabilisé, indépendantes et suivant la même loi que V , c'est à dire la loi $\mathcal{E}(1)$.

On considère la variable aléatoire Y_n définie par $Y_n = \max(X_1, X_2, \dots, X_n)$, c'est à dire que pour tout ω de Ω , on a : $Y_n(\omega) = \max(X_1(\omega), X_2(\omega), \dots, X_n(\omega))$. On admet que Y_n est une variable aléatoire à densité.

Enfin, on pose $Z_n = Y_n - \ln(n)$.

- On rappelle que `grand(1, n, 'exp', 1)` simule n variables aléatoires indépendantes et suivant toutes la loi exponentielle de paramètre 1.

Compléter la déclaration de fonction `Scilab` suivante afin qu'elle simule la variable aléatoire Z_n .

```
function Z=f(n)
x=grand(1,n,'exp',1)
Z=...
endfunction
```

- Voici deux scripts :

- script (1)

```
V=grand(1,10000,'exp',1)
W=-log(V)
s=linspace(0,10,11)
histplot(s,W)
```

- script (2)

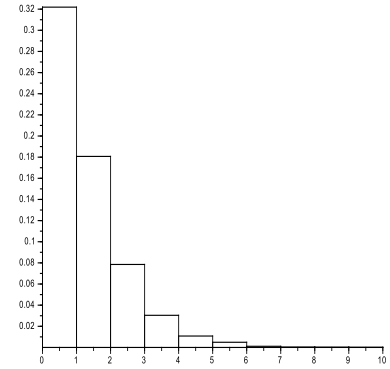
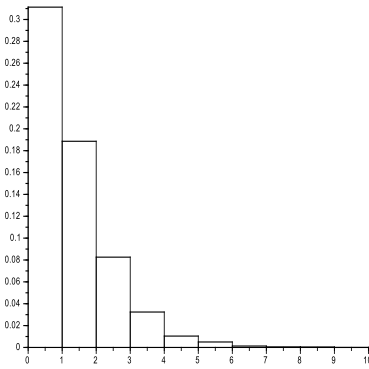
```
n=input('entrez la valeur de n : ')
Z=[] // la matrice-ligne Z est vide
for k=1:10000
    Z=[Z,f(n)]
end
s=linspace(0,10,11)
histplot(s,Z)
```

Chacun des scripts simule 10000 variables indépendantes, regroupe les valeurs renvoyées en 10 classes qui sont les intervalles $[0, 1]$, $[1, 2]$, $[2, 3]$, \dots , $[9, 10]$ et trace l'histogramme correspondant (la largeur de chaque rectangle est égale à 1 et leur hauteur est proportionnelle à l'effectif de chaque classe).

Le script (1) dans lequel les variables aléatoires suivent la loi de Grumbel (loi suivie par W), renvoie l'histogramme (1) ci-dessous, alors que le script (2) dans lequel les variables aléatoires suivent la même loi que Z_n , renvoie l'histogramme (2) ci-dessous, pour lequel on a choisi $n = 1000$.

Histogramme (1)

Histogramme(2) pour $n = 1000$



Quelle conjecture peut-on émettre quant au comportement de la suite des variables aléatoires (Z_n) ?

2.0.2 HECII 2017 - Partie III (voie S)

On admet que la fonction $f_X : x \mapsto \frac{1}{\pi} \times \frac{1}{1 + (x - \theta)^2}$ est une densité de probabilité sur \mathbb{R} .

Dans toute la suite du problème, on note X une variable aléatoire, à valeurs réelles, de densité f_X . On dit que X suit une loi de Cauchy de paramètre θ ;

10. On pose : $Y = X - \theta$. Pour n entier de \mathbb{N}^* , soit (Y_1, Y_2, \dots, Y_n) un n —échantillon de variables aléatoires indépendantes et de même loi que Y .

Pour tout $n \in \mathbb{N}^*$, on pose : $S_n = \sum_{i=1}^n Y_i$ et $\bar{Y}_n = \frac{S_n}{n}$. (moyenne empirique de l'échantillon (Y_1, Y_2, \dots, Y_n))

10.c) On admettra que Y et \bar{Y}_n suivent toutes deux une loi de Cauchy de paramètre θ .

Soit $(N, n) \in \mathbb{N}^{*2}$. On veut simuler N réalisations de la moyenne empirique \bar{Y}_n .

On suppose que l'on connaît une fonction Scilab `cauchy` telle que la commande `A=cauchy(N,m)` retourne une matrice $A \in \mathcal{M}_{N,n}(\mathbb{R})$, réalisation d'une famille $(Y_{i,j})_{1 \leq i \leq N; 1 \leq j \leq n}$ de variables aléatoires indépendantes de loi de Cauchy de paramètre 0.

Soit M une matrice de $\mathcal{M}_{N,n}(\mathbb{R})$ avec $(N, n) \in \mathbb{N}^{*2}$. On rappelle que dans le langage Scilab :

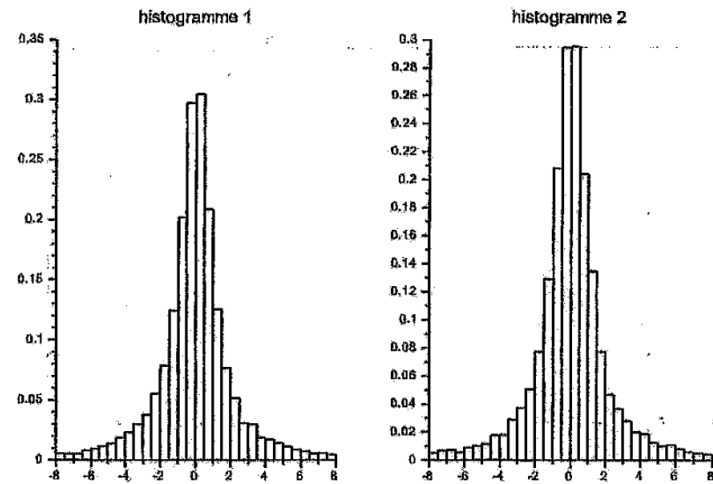
- la commande `sum(M)` retourne une matrice de $\mathcal{M}_{1,1}(\mathbb{R})$ contenant la somme de tous les éléments de M ;
- la commande `sum(M, 'r')` retourne un vecteur ligne de $\mathcal{M}_{1,n}(\mathbb{R})$ contenant les sommes des éléments de M calculées colonne par colonne;
- la commande `sum(M, 'c')` retourne un vecteur colonne de $\mathcal{M}_{N,1}(\mathbb{R})$ contenant les sommes des éléments de M calculées ligne par ligne;
- la commande `linspace(a,b,m)` retourne un vecteur ligne de m valeurs régulièrement espacées entre a et b et l'on obtient le même vecteur avec la commande `(a:l:b)` en prenant $\ell = \frac{b-a}{m-1}$;
- la commande `histplot(y,data)` permet de représenter les éléments du vecteur `data` sous la forme d'un histogramme; les classes de l'histogramme sont définies par le vecteur strictement croissant `y` : si ce vecteur contient m éléments `y(1), y(2), \dots, y(m)` tels que `y(1) < y(2) < \dots < y(m)`, alors la première classe de l'histogramme est l'intervalle `[y(1), y(2)]` et les autres classes sont les intervalles `]y(i), y(i+1)]` pour $2 \leq i \leq m$.

1. Compléter le programme suivant afin que la matrice `MoyEmp` contienne 12000 réalisations de la moyenne empirique \bar{Y}_{200} .

```

N=12000 ;n=200
A=cauchy(N, n)
MoyEmp=.....
x=(-8: 0.5 : 8)
histplot(x,MoyEmp) \\ histogramme 1
histplot(x,A(:,1)) \\ histogramme 2

```



Les histogrammes 1 et 2 ont été obtenus à l'aide de ce programme. Expliquer en quoi ce couple d'histogrammes illustre le résultat de la question 10.c)

2. On rappelle que X suit une loi de Cauchy de paramètre θ supposé ici inconnu. Pour n entier de \mathbb{N}^* , on note $(X_1, X_2, \dots, X_{2n+1})$ un $(2n+1)$ —échantillon de variables aléatoire indépendantes et de même loi que X . On admet l'existence de $(2n+1)$ fonctions $g_1, g_2, \dots, g_{2n+1}$ continues sur \mathbb{R}^{2n+1} à valeurs réelles, telles que les variables aléatoires réelles $\hat{X}_1, \hat{X}_2, \dots, \hat{X}_{2n+1}$ définies par : $\forall k \llbracket 1, 2n+1 \rrbracket, \hat{X}_k = g_k(X_1, X_2, \dots, X_{2n+1})$ soient des variables aléatoires à densité et que pour tout $\omega \in \Omega$, les réels $\hat{X}_1(\omega), \hat{X}_2(\omega), \dots, \hat{X}_{2n+1}(\omega)$ soient un réarrangement par ordre croissant de $X_1(\omega), X_2(\omega), \dots, X_{2n+1}(\omega)$: $\forall \omega \in \Omega, \hat{X}_1(\omega) \leq \hat{X}_2(\omega) \leq \dots \leq \hat{X}_{2n+1}(\omega)$. En particulier, la variable aléatoire \hat{X}_{n+1} est la médiane empirique de l'échantillon $(X_1, X_2, \dots, X_{2n+1})$.

On pose enfin pour tout entier $n \geq 2$: $W_{n+1} = \frac{2\sqrt{2n+1}}{\pi} (\hat{X}_{n+1} - \theta)$.

Dans le langage Scilab, la fonction `gsort` permet de trier les éléments d'une matrice réelle A :

- la commande `gsort(A, 'r')` renvoie une copie de A triée colonne par colonne par ordre décroissant (chaque colonne est triée indépendamment des autres)
- la commande `gsort(A, 'c')` renvoie une copie de A triée ligne par ligne, par ordre décroissant (chaque ligne est triée indépendamment des autres).

On suppose que $\theta = 0$ et on considère p réalisations ($p \geq 10^4$) de $(2n+1)$ —échantillon $(X_1, X_2, \dots, X_{2n+1})$. Recopier et compléter le code suivant afin que son exécution retourne un vecteur `MedianeEmp` de p réalisations de la médiane empirique \hat{X}_{n+1} , puis un vecteur W de p réalisations de W_{n+1}

```

A=cauchy(2, 2*n+1)
S=gsort .....
MedianeEmp=.....
W=.....

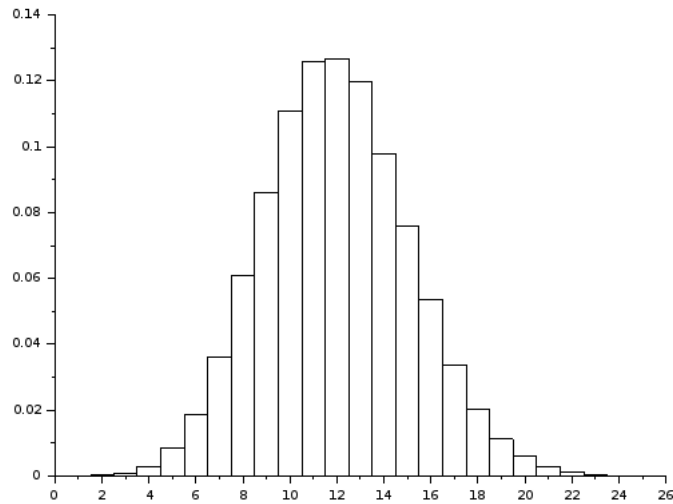
```

2.0.3 Oral HEC 2015 Exercice sans préparation E 69

Après une alerte incendie, les 60 élèves d'une école se répartissent au hasard dans cinq salles de classe. Afin de savoir comment se répartissent les élèves, on exécute le programme Scilab suivant :

```
Y=grand(100000,1,"bin",60,1/5)
histplot(0.5:25,Y)
```

Qui donne la représentation ci-dessous :



Que représente la valeur maximale prise par cet histogramme ? Prouver le résultat concernant cette valeur.

SIMULATION DE VARIABLE ALÉATOIRES DISCRÈTES

3.0.1 ESCP 2016 - Exercice 4 (Voie T)

Une puce se déplace sur un axe gradué. À l'instant 0, la puce se trouve sur le point d'abscisse 0.

À partir de l'instant 0, la puce effectue à chaque instant, un saut vers la droite selon le protocole suivant :

- elle effectue un saut d'une unité vers la droite avec la probabilité $\frac{1}{2}$;
- elle effectue un saut de deux unités vers la droite avec la probabilité $\frac{1}{4}$;
- elle effectue un saut de trois unités vers la droite avec la probabilité $\frac{1}{4}$.

Les différents sauts sont supposés indépendants.

4. On rappelle qu'en *Scilab* l'instruction `grand(1,1,'uin',1,4)` simule une variable aléatoire suivant la loi discrète uniforme sur $[[1,4]]$.

Compléter le programme suivant pour qu'il simule les 100 premiers déplacements de la puce.

```
A=zeros(1,100)
for k=1:100
    t=grand(1,1,'uin',1,4)
    if t<=..... then A(k) =1
    end
    if t==..... then A(k) =2
    end
    if t==..... then A(k) =3
    end
end
disp(A)
```

Quelle sortie graphique obtient-on ?

8. On rappelle que si $x = (x_1, x_2, \dots, x_n)$ et $y = (y_1, y_2, \dots, y_n)$ sont deux vecteurs de même taille, la commande `plot2d(x,y)` permet de tracer la ligne brisée joignant les points $M_1(x_1, y_1)$, $M_2(x_2, y_2)$, ..., $M_n(x_n, y_n)$.

On complète le programme *Scilab* de la question 4 en y ajoutant les trois commandes suivantes :

```
x=1:100
y=cumsum(A)
plot2d(x,y)
```

3.0.2 ECRICOME 2017 - Exercice 2 (voie T)

L'urne U contient 1 boule noire et 3 boules blanches indiscernables au toucher. L'urne V contient 2 boules noires et 2 boules blanches indiscernables au toucher.

On lance une pièce équilibrée. Si elle retombe sur le côté Pile, on tire deux boules successivement et avec remise dans U , et si on obtient Face, on tire deux boules successivement et avec remise dans V .

On note T la variable aléatoire égale au nombre de fois où l'on a pioché une boule noire.

On rappelle qu'en langage Scilab l'instruction `grand(1,1,"uin",n1,n2)` renvoie un entier au hasard et uniformément compris entre $n1$ et $n2$. Compléter, sur votre copie, le programme Scilab suivant afin qu'il affiche une simulation de la variable aléatoire T .

```

T = .....
if grand(1,1,"uin",1,2) == 1 then
    for k = 1 : 2
        if grand(1,1,"uin",1,4)<2 then
            T = T+1
        end
    end
else
    .....
    .....
    .....
    .....
    .....
end
disp(T,"Une simulation de T donne :")

```

3.0.3 ECRICOME 2015 - Exercice 3 (Voie E)

Dans tout cet exercice, N désigne un entier naturel supérieur ou égal à 3.

On dispose de deux urnes opaques U_1 et U_2 , d'apparence identique et contenant chacune N boules indiscernables au toucher.

L'urne U_1 contient $(N - 1)$ boules blanches et une boule noire.

L'urne U_2 contient N boules blanches.

I - Une première expérience aléatoire

On effectue des tirages **sans remise** dans l'urne U_1 , jusqu'à l'obtention de la boule noire.

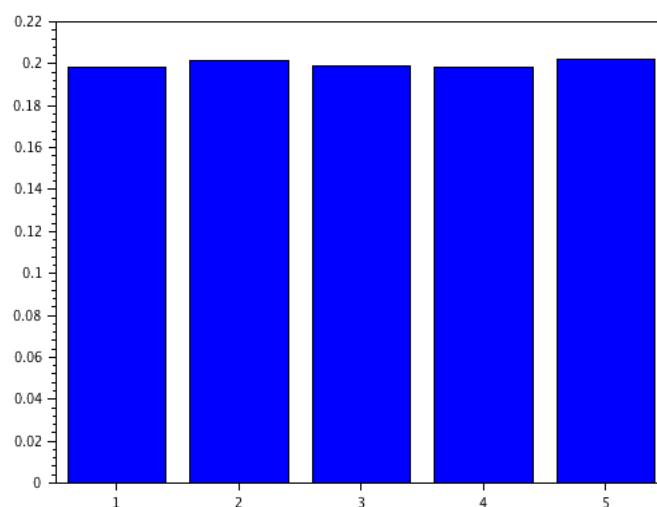
On note X la variable aléatoire qui prend pour valeur le nombre de tirages nécessaires pour l'obtention de la boule noire.

1. On simule 10000 fois cette expérience aléatoire.

Recopier et compléter le programme SCILAB suivant pour qu'il affiche l'histogramme donnant la fréquence d'apparition du rang d'obtention de la boule noire :

```
N = input(' Donner un entier naturel non nul' ) ;
S = zeros(1,N);
for k = 1 : 10000
    i = 1 ;
    M = N ;
    while -----
        i = i + 1 ;
        M = ----- ;
    end
    S(i) = S(i) + 1 ;
end
disp(S / 10000)
bar(S / 10000)
```

2. On exécute le programme complété ci-dessus. On entre 5 au clavier et on obtient l'histogramme suivant :



Quelle conjecture pouvez-vous émettre sur la loi de la variable aléatoire X ?

3.0.4 ECRICOME 2016 - Exercice 3 (Voie E)

Dans tout l'exercice, X et Y sont deux variables aléatoires définies sur le même espace probabilisé et à valeurs dans \mathbb{N} .

Soient n , b et c trois entiers strictement positifs.

Une urne contient initialement n boules noires et b boules blanches. On effectue l'expérience suivante, en distinguant trois variantes.

- On pioche une boule dans l'urne.
On définit X la variable aléatoire qui vaut 1 si cette boule est noire et 2 si elle est blanche.
- On replace la boule dans l'urne et :
 - ★ Variante 1 : on ajoute dans l'urne c boules de la même couleur que la boule qui vient d'être piochée.
 - ★ Variante 2 : on ajoute dans l'urne c boules de la couleur opposée à celle qui vient d'être piochée.
 - ★ Variante 3 : on n'ajoute pas de boule supplémentaire dans l'urne.
- On pioche à nouveau une boule dans l'urne.
On définit Y la variable aléatoire qui vaut 1 si cette seconde boule piochée est noire et 2 si elle est blanche.

3. (a) Compléter la fonction Scilab suivante, qui simule le tirage d'une boule dans une urne contenant b boules blanches et n boules noires et qui retourne 1 si la boule tirée est noire, et 2 si la boule tirée est blanche.

```
function res = tirage(b,n)
    r=rand()
    if ..... then
        res = 2
    else
        res = 1
    end
endfunction
```

Compléter la fonction suivante, qui simule l'expérience N fois (avec $N \in \mathbb{N}^*$), et qui estime la loi de X , la loi de Y et la loi du couple (X, Y) .
Les paramètres de sortie sont :

- (b) Compléter la fonction suivante, qui effectue l'expérience étudiée avec une urne contenant initialement b boules blanches, n boules noires et qui ajoute éventuellement c boules après le premier tirage, selon le choix de la variante dont le numéro est variante .
Les paramètres de sortie sont :

- x : une simulation de la variable aléatoire X
- y : une simulation de la variable aléatoire Y .

```
function [x,y] = experience (b,n,c,va
x=tirage(b,n)
if variante == 1 then
    if x == 1 then
        .....
    else
        .....
    end
else if variante == 2 then
    .....
    .....
    .....
    .....
end
y = tirage(b,n)
endfunction
```

- loiX : un tableau unidimensionnel à deux éléments qui estime $[P(X = 1), P(X = 2)]$
- loiY : un tableau unidimensionnel à deux éléments qui estime $[P(Y = 1), P(Y = 2)]$

- `loiXY` : un tableau bidimensionnel à deux lignes et deux colonnes qui estime
- On exécute notre fonction précédente avec $b=1, n=2, c=1, N=10\,000$ et dans chacune des variantes. On obtient :

$$\begin{bmatrix} P([X=1] \cap [Y=1]) & P([X=1] \cap [Y=2]) \\ P([X=2] \cap [Y=1]) & P([X=2] \cap [Y=2]) \end{bmatrix}.$$

```

-->[loiX,loiY,loiXY]=estimation (1,2,1,1,10000)
loiXY =
    0.49837    0.16785
    0.16697    0.16681
loiY =
    0.66534    0.33466
loiX =
    0.66622    0.33378

-->[loiX,loiY,loiXY]=estimation (1,2,1,2,10000)
loiXY =
    0.33258    0.33286
    0.25931    0.08425
loiY =
    0.58289    0.41711
loiX =
    0.66544    0.33456

-->[loiX,loiY,loiXY]=estimation (1,2,1,3,10000)
loiXY =
    0.44466    0.22098
    0.22312    0.11124
loiY =
    0.66778    0.33222
loiX =
    0.66564    0.33436

function[loiX,loiY,loiXY]=estimation(b,n,c,variante,N)
loiX = [0,0]
loiY=[0,0]
loiXY=[0,0;0,0]
for k=1:N
    [x,y]=experience(b,n,c,variante)
    loiX(x)=loiX(x)+1
    .....
    .....
end
loiX = loiX / N
loiY = loiY / N
loiXY = loiXY / N
endfunction

```

En étudiant ces résultats, émettre des conjectures quant à l'indépendance et l'échangeabilité de X et Y dans chacune des variantes.

On donne les valeurs numériques approchées suivantes :

$$0.33 \times 0.33 \simeq 0.11$$

$$0.33 \times 0.41 \simeq 0.14$$

$$0.33 \times 0.58 \simeq 0.19$$

$$0.33 \times 0.66 \simeq 0.22$$

$$0.41 \times 0.66 \simeq 0.27$$

$$0.58 \times 0.66 \simeq 0.38$$

$$0.66 \times 0.66 \simeq 0.44$$

3.0.5 ECRICOME 2017 - Exercice 3 (Voie E)

Soit n un entier naturel non nul.

On effectue une série illimitée de tirages d'une boule avec remise dans une urne contenant n boules numérotées de 1 à n . Pour tout entier naturel k non nul, on note X_k la variable aléatoire égale au numéro de la boule obtenue au k -ième tirage.

Pour tout entier naturel k non nul, on note S_k la somme des numéros des boules obtenues lors des k premiers tirages :

$$S_k = \sum_{i=1}^k X_i.$$

On considère enfin la variable aléatoire T_n égale au nombre de tirages nécessaires pour que, pour la première fois, la somme des numéros des boules obtenues soit supérieure ou égale à n .

Exemple : avec $n = 10$, si les numéros obtenus aux cinq premiers tirages sont dans cet ordre 2, 4, 1, 5, 9, alors on obtient : $S_1 = 2$, $S_2 = 6$, $S_3 = 7$, $S_4 = 12$, $S_5 = 21$ et $T_{10} = 4$.

1. On rappelle qu'en langage Scilab, l'instruction `grand(1,1,'uin',1,n)` renvoie un entier aléatoire de $\llbracket 1, n \rrbracket$.¹ Compléter la fonction ci-dessous, qui prend en argument le nombre n de boules contenues dans l'urne, afin qu'elle simule la variable aléatoire T_n :

```
function y=T(n)
    S = .....
    y = .....
    while .....
        tirage = grand(1,1,'uin',1,n)
        S = S + tirage
        y = .....
    end
endfunction
```

2. On suppose déclarée la fonction précédente et on écrit le script ci-dessous :

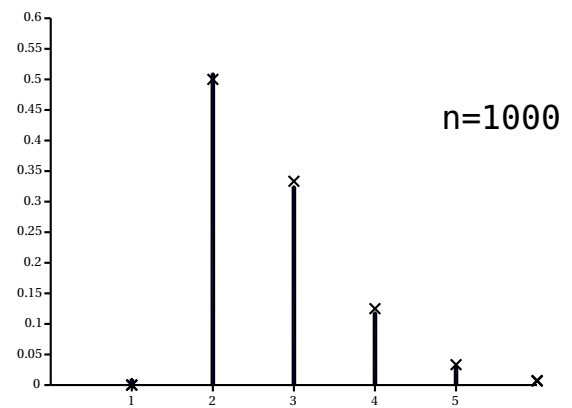
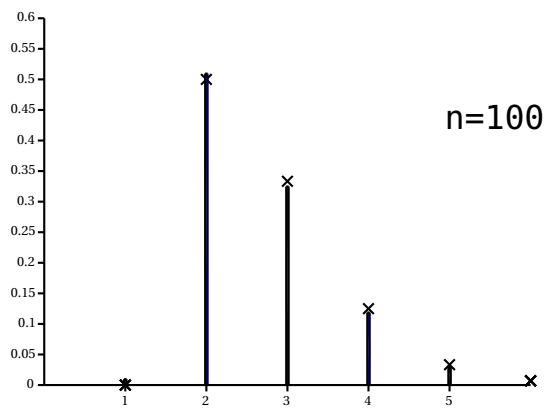
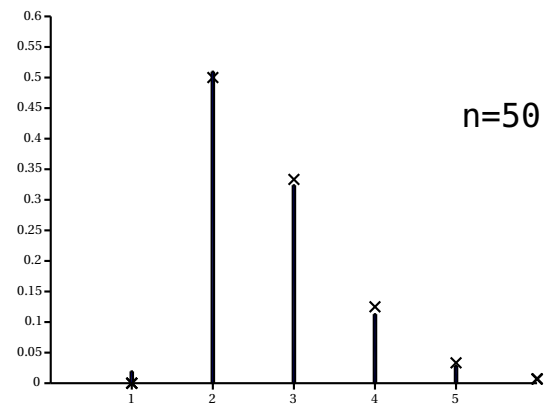
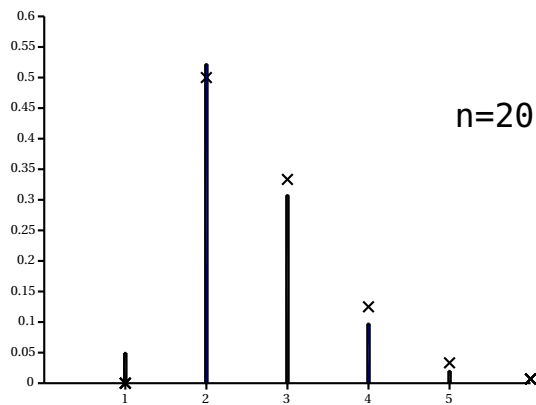
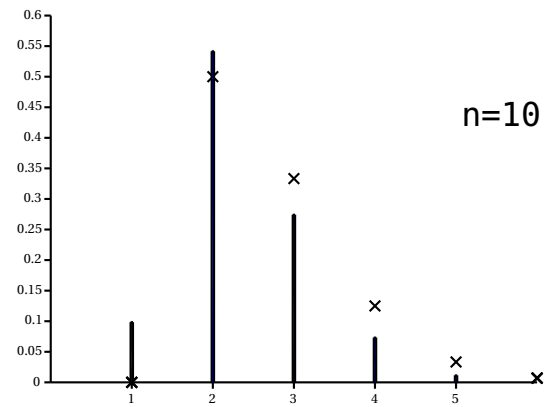
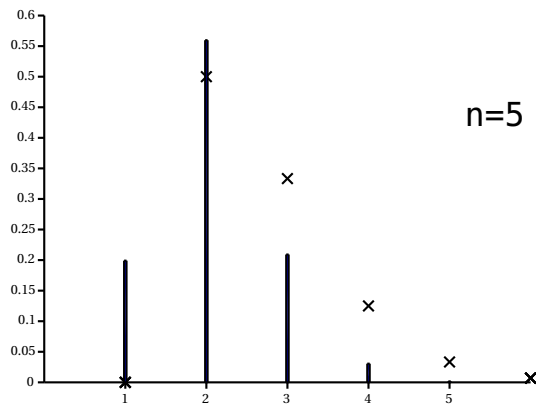
```
function y=freqT(n)
    y = zeros(1,n)
    for i=1:100000
        k = T(n)
        y(k) = y(k)+1
    end
    y = y/100000
endfunction

function y=loitheoY(n)
    y = zeros(1,n)
    for k=1:n
        y(k) = (k-1)/prod(1:k)
    end
endfunction

clf
n = input('n=?')
plot2d(loitheoY(6),style=-2)
x = freqT(n)
bar(x(1:5))
```

L'exécution de ce script pour les valeurs de n indiquées a permis d'obtenir les graphes ci-dessous :

¹L'énoncé Ecricome oublie de préciser "selon la loi uniforme".



- (a) Expliquer ce que représentent les vecteurs renvoyés par les fonctions `freqT` et `loiTheoY`. Comment ces vecteurs sont-ils représentés graphiquement dans chaque graphique obtenu ?
- (b) Expliquer en quoi cette succession de graphiques permet d'illustrer le fait que $(T_n)_{n \geq 1}$ converge en loi vers la variable aléatoire Y .

3.0.6 ECRICOME 2016 (Voie S)

Soient a, b deux entiers strictement positifs. Une urne contient initialement a boules rouges et b boules blanches. On effectue une succession d'épreuves, chaque épreuve étant constituée des trois étapes suivantes :

- on pioche une boule au hasard dans l'urne,
- on remplace la boule tirée dans l'urne,
- on rajoute dans l'urne une boule de la même couleur que celle qui vient d'être piochée

Après n épreuves, l'urne contient donc $a + b + n$ boules.

Pour tout $n \in \mathbb{N}^*$, on note X_n le nombre de boules rouges qui ont été **ajoutées** dans l'urne (par rapport à la composition initiale) à l'issue des n premières épreuves.

Pour tout $n \in \mathbb{N}^*$, on notera R_n l'événement "on pioche une boule rouge au n -ième tirage".

3. Donner l'ensemble $X_n(\Omega)$ des valeurs prises par la variable aléatoire X_n en fonction de n .

4. On souhaite simuler l'expérience grâce à Scilab.

- (a) Compléter la fonction suivante, qui simule le tirage d'une boule dans une urne contenant x boules rouges et y boules blanches et qui retourne la valeur 0 si la boule est rouge et 1 si elle est blanche.

```
function res = tirage(x,y)
    r = rand()
    if ..... then
        res = 0
    else
        res = 1
    end
endfunction
```

- (b) Compléter la fonction suivante, qui simule n tirages successifs dans une urne contenant initialement a boules rouges et b boules blanches (selon le protocole décrit ci-dessus) et qui retourne la valeur de X_n :

```
function Xn = experience(a,b,n)
    x = a
    y = b
    for k=1:n
        r = tirage(x,y)
        if r == 0 then
            x = .....
        else
            .....
        end
    end
    Xn = .....
endfunction
```

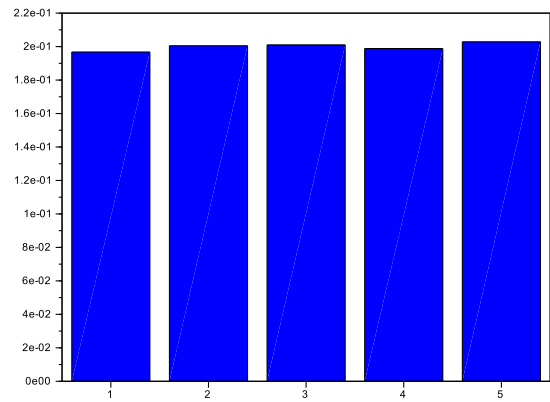
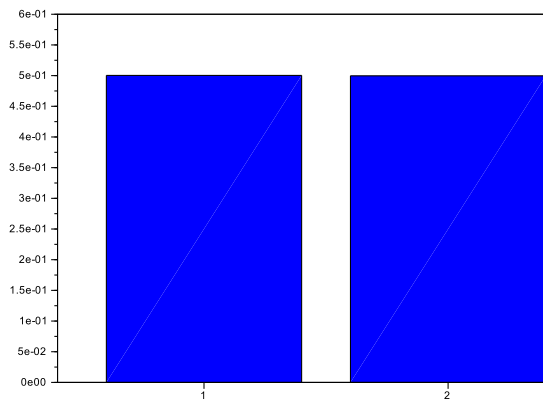
- (c) Écrire une fonction Scilab d'en tête :

```
function loi = simulation(a,b,n,m)
```

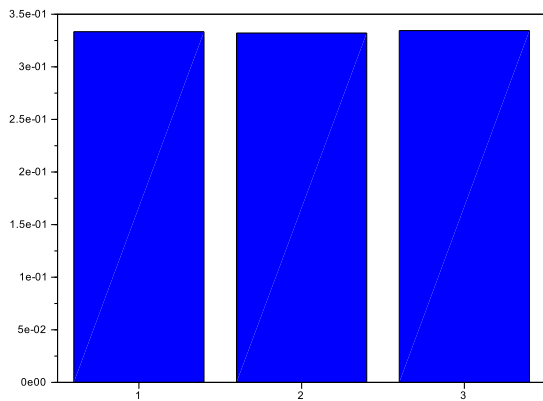
qui fait appel m fois à la fonction précédente pour estimer la loi de X_n . Le paramètre de sortie sera un vecteur contenant les approximations de $P(X_n = 0)$, $P(X_n = 1)$, ..., $P(X_n = n)$.

5. On s'intéresse ici au cas où $a = b = 1$. On utilise la fonction `simulation` avec des valeurs de n entre 1 et 5 et on affiche à chaque fois l'estimation de la loi de X_n sous forme d'un diagramme en "bâtons".

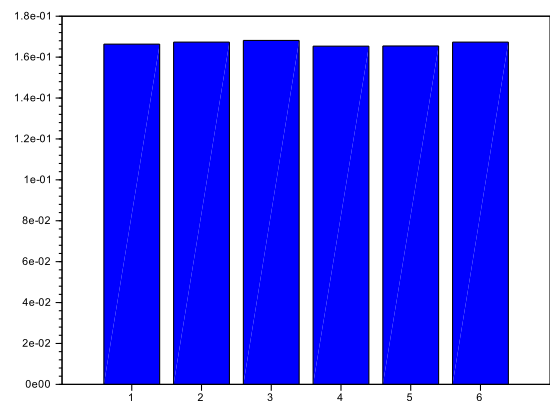
```
--> bar( simulation(1,1,1,100000))    --> bar( simulation(1,1,4,100000))
```



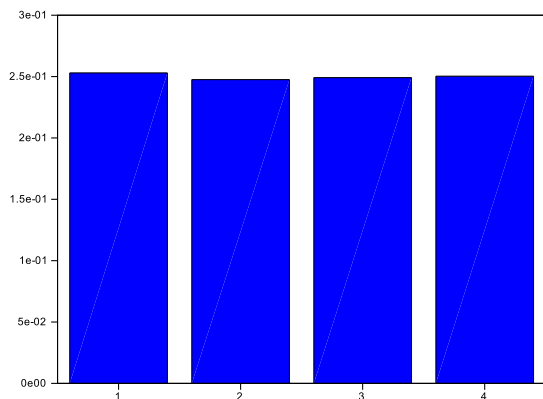
```
--> bar( simulation(1,1,2,100000))
```



```
--> bar( simulation(1,1,5,100000))
```



```
--> bar( simulation(1,1,3,100000))
```



6. À l'aide de ces résultats, conjecturer la loi de X_n .

3.0.7 ECRICOME 2017 - Problème (Voie S)

Pour tout entier n strictement positif, on considère l'expérience suivante : on dispose de n urnes initialement vides, numérotées de 1 à n et on dispose d'un grand stock de boules que l'on dépose une à une dans ces urnes. Pour chaque boule, on choisit au hasard, de façon équiprobable, l'urne dans laquelle la boule est déposée. On note X_n le rang du premier tirage pour lequel une des urnes contiendra deux boules.

1. Compléter la fonction Scilab suivante pour qu'elle simule une réalisation de la variable aléatoire X_n :

```
function X = tirage(n)
    urnes = zeros(1,n)
    X = 1
    choix = floor((rand()*n))+1
    while .....
        urnes(choix) = urnes(choix)+1
        choix = floor((rand()*n)+1)
        X = .....
    end
endfunction
```

2. On admet que $X_n(\Omega) = \llbracket 2, n+1 \rrbracket$, que $\forall k \in \llbracket 2, n+1 \rrbracket$, $P(X_n = k) = \frac{n!(k-1)}{n^k(n-k+1)!}$. et que X_n admet une espérance.

On souhaite écrire une fonction Scilab qui calcule $E(X_n)$ en fonction de n .

Compléter la fonction suivante à cet effet :

```
function E = esperance(n)
    facto = prod([1:n])
    fac = facto
    somme = 0
    for k = 2 : (n+1)
        puissance = .....
        fac = .....
        somme = somme + k*(k-1)/(puissance*fac)
    end
    E = facto * somme
endfunction
```

EXERCICE 3.1 On considère une suite $(X_n)_{n \in \mathbb{N}^*}$ de variables aléatoires de même loi, à valeurs dans \mathbb{N} , mutuellement indépendantes et indépendantes de la variable N étudiée dans la question 4 de la partie 1.

On considère alors la variable aléatoire S définie par :

$$S = \begin{cases} 0 & \text{si } N = 0 \\ \sum_{k=1}^N X_k & \text{si } N \geq 1 \end{cases}$$

autrement dit :

$$\forall \omega \in \Omega, \quad S(\omega) = 0 \text{ si } N(\omega) = 0 \quad \text{et} \quad S(\omega) = \sum_{k=1}^{N(\omega)} X_k(\omega) \text{ sinon.}$$

a) On considère la fonction Scilab suivante, où n est un paramètre dont dépend la loi commune des X_k :

```
function y=simuX(n)
    y=0;
    for i=1:n
        if rand()<1/2
            y=y+1;
        end
    end
endfunction
```

Quelle loi de probabilité est simulée par la fonction `simuX` ? Préciser ses paramètres.

b) On rappelle qu'en Scilab l'instruction `grand(1,1,"poi",lambda)` renvoie une réalisation d'une loi de Poisson de paramètre `lambda`.

On suppose que N suit une loi de Poisson de paramètre λ , et que la loi des variables X_k est celle simulée à la question précédente par la fonction `simuX`.

Recopier et compléter la fonction Scilab suivante, afin qu'elle renvoie une simulation de la variable aléatoire S :

```
function s=simulS(lambda,n)
    N=grand(1,1,"poi",lambda)
    .....
    .....
    .....
    .....
    .....
endfunction
```

3.0.8 ECRICOME 2019 - Problème (Voie S)

EXERCICE 3.2

Une urne contient initialement une boule, blanche et une boule noire. On effectue une succession de tirages d'une boule dans cette urne. Après chaque tirage, on remet la boule tirée dans l'urne, et on rajoute dans l'urne une boule de couleur opposée à celle qui vient d'être tirée.

On suppose que cette expérience est modélisée par un espace probabilisé $(\Omega, \mathcal{T}, \mathbb{P})$

Pour tout $k \in \mathbb{N}$, on note X_k le nombre de boules blanches présentes dans l'urne juste avant le $(k+1)$ -ième tirage. En particulier, on a $X_0 = 1$. On admet que pour tout entier k , X_k est une variable aléatoire de $(\Omega, \mathcal{T}, \mathbb{P})$.

On admet que :

$$\forall k \in \mathbb{N}, \forall i \in \mathbb{N}^*, \quad P([X_{k+1} = i]) = \frac{i}{k+2} P([X_k = i]) + \frac{3+k-i}{k+2} P([X_k = i-1]) \quad (*)$$

et on admet que pour tout $k \in \mathbb{N}$:

$$P([X_k = 1]) = P([X_k = k+1]) = \frac{1}{(k+1)!}$$

1. Que renvoie la fonction Scilab suivante pour un entier k non nul ?

Détailler le fonctionnement de la ligne 5.

```

1 function x=mystere(k)
2   n=1;
3   b=1;
4   for i=1:k
5     r=floor(rand()*(n+b)+1)
6     if r>n then
7       n=n+1
8     else
9       b=b+1
10    end
11  end
12  x=b
13 endfunction}

```

9. Écrire une fonction Scilab d'en-tête `function LE = loi-exp(k,N)` qui prend en entrée un entier strictement positif N , qui effectue N simulations de k tirages successifs dans l'urne et qui retourne un vecteur `LE` qui contient une estimation de la loi de X_k (c'est-à-dire que pour chaque $i \in [1, k+1]$, `LE(i)` contient la fréquence d'apparition de l'événement $[X_k = i]$ au cours des N simulations).

On pourra utiliser la fonction `mystere`.

10. Recopier et compléter la fonction `loi-theo` suivante, qui prend en entrée un entier strictement positif n , afin qu'elle retourne un vecteur `LT` qui contient la loi théorique de X_n .

```

function LT=loi-theo(n)
2   M=zeros(n,n+1)
3   M(1,1)=1/2
4   M(1,2)=1/2
5   for k=1:n-1
6     M(k+1,1)=.....
7     for i=2:k+1
8       M(k+1,i)=.....
9     end
10    M(k+1,k+2)=.....
11  end
12  LT=.....
13 endfunction

```

11. Un étudiant nous propose comme loi de X_5 le résultat suivant :

k	1	2	3	4	5	6
$P(X_5 = k)$	0.001368	0.079365	0.419434	0.418999	0.079454	0.00138

A-t-il utilisé `loi-exp` ou bien `loi-theo` ?

3.0.9 EDHEC 2015 - Problème (Voie E)

Un joueur réalise une suite de lancers indépendants d'une pièce. Cette pièce donne pile avec la probabilité p ($0 < p < 1$) et face avec la probabilité $q = 1 - p$.

On note N la variable aléatoire égale au rang d'apparition du premier pile.

Si N prend la valeur n , le joueur place n boules numérotées de 1 à n dans une urne, puis il extrait une boule au hasard de cette urne. On dit que le joueur a gagné si le numéro porté par la boule tirée est impair et on désigne par A l'événement : "le joueur gagne". On appelle X la variable aléatoire égale au numéro porté par la boule extraite de l'urne.

1. Reconnaître la loi de N .
2. (a) Montrer que, si m est un entier naturel, la commande `2*floor(m/2)` renvoie la valeur m si et seulement si m est pair.
- (b) Compléter les commandes `Scilab` suivantes pour qu'elles simules N et X puis renvoient l'un des deux messages "le joueur a gagné" ou "le joueur a perdu".

```
p=input('donner la valeur de p')
N=grand(1,1,'geom', ---) // 'geom' designe une loi geometrique
X=grand(1,1,'uin',---) // 'uin' designe une loi uniforme discrete
if -----
    then disp('-----')
    else disp('-----')
end
```

3.0.10 EDHEC 2016 - Problème (Voie E)

On considère une suite $(X_n)_{n \in \mathbb{N}^*}$ de variables aléatoires, mutuellement indépendantes, suivant toutes la loi géométrique de paramètre x , et on pose $S_n = \sum_{k=1}^n X_k$.

On rappelle que la commande `grand(1, n, 'geom', p)` permet à `Scilab` de simuler n variables aléatoires indépendantes suivant toutes la loi géométrique de paramètre p .

Compléter les commandes `Scilab` suivantes pour qu'elles simulent la variable aléatoire S_n .

```
n=input('entrez une valeur de n superieure a 1: ')
S=-----
disp(S)
```

3.0.11 EDHEC 2018 - Exercice 2 (Voie E)

EXERCICE 3.3

On dispose de trois pièces : une pièce numérotée 0, pour laquelle la probabilité d'obtenir "pile" vaut $\frac{1}{2}$ et celle d'obtenir "face" vaut également $\frac{1}{2}$, une pièce numérotée 1, donnant "pile" à coup sûr et une troisième pièce, numérotée 2, donnant "face" à coup sûr.

On choisit l'une de ces pièces au hasard et on la lance indéfiniment.

Pour tout i de $\{0, 1, 2\}$, on note A_i l'événement : "on choisit la pièce numérotée i ".

Pour tout entier naturel k non nul, on note P_k l'événement : "on obtient "pile" au lancer numéro k " et on pose $F_k = \overline{P_k}$.

On considère la variable aléatoire X , égale au rang d'apparition du premier "pile" On convient de donner à X la valeur 0 si l'on n'obtient jamais "pile"

On rappelle que , pour tout entier naturel m , l'instruction `grand(1, 1, 'uin', 0, m)` renvoie un entier aléatoire compris entre 0 et m (ceci de façon équiprobable).

On décide de coder "pile" par 1 et "face" par 0 .

1. Compléter le script `Scilab` suivant pour qu'il permette le calcul et l'affichage de la valeur prise par la variable aléatoire X lors de l'expérience réalisée dans cet exercice.

```
piece = grand(1,1,'uin',- - -, - - -)
x=1
if piece==0
then lancer=grand(1,1,'uin',- - -, - - -)
    while lancer==0
        lancer=- - -
        x=- - -
    end
else
    if piece==1 then x=- - -
    end
end
disp(x)
```

2. Justifier que le cas où l'on joue avec la pièce numérotée 2 ne soit pas pris en compte dans le script précédent.

3.0.12 EDHEC 2015 - Problème (Voie S)

On désigne par α et p deux réels de $]0, 1[$. Un joueur participe à un jeu constitué d'une suite de manches.

Avant chaque manche y compris la première, le joueur a une probabilité α de ne pas être autorisé à jouer la manche en question, (on dit qu'il est disqualifié, et c'est définitif), et une probabilité $1 - \alpha$ d'y être autorisé, et ceci indépendamment du fait qu'il ait gagné ou perdu la manche précédente s'il y en a eu une. A chaque manche jouée, le joueur gagne un euro avec la probabilité p et perd un euro avec la probabilité $1 - p$.

Si le jeu a commencé, le joueur joue jusqu'à ce qu'il soit disqualifié, et on suppose que les manches sont jouées de façon indépendantes. On note :

- X le nombre de manches jouées par le joueur avant d'être disqualifié.
- Y le nombre de manches gagnées par le joueur.
- G le gain du joueur à la fin du jeu.

On admet que X , Y et G sont des variables aléatoires définies toutes les trois sur le même espace probabilisé $(\Omega, \mathcal{A}, \mathcal{P})$.

1. Compléter, en utilisant la fonction **grand** les commandes **Scilab** suivantes pour qu'elles simulent l'expérience aléatoire étudiée et affichent les valeurs prises par X et Y .

```
alpha=input('entrer la valeur de alpha : ')
p=input('entrer la valeur de p : ')
X=-----
Y=-----
disp(X)
disp(Y)
```

2. Quelles commandes faut-il ajouter aux précédentes pour calculer et afficher la valeur prise par G ?

3.0.13 EDHEC 2018 (épreuve annulée) - Exercice 3 (Voie S)

Un mobile se déplace aléatoirement sur un axe dont l'origine est le point O d'abscisse 0.

Au départ (instant 0), le mobile est situé sur le point O .

Le mobile se déplace selon la règle suivante : à l'instant n ($n \in \mathbb{N}^*$), il se place de façon équiprobable, sur l'un des points d'abscisse $0, 1, \dots, n$.

Pour tout entier naturel n , on note X_n l'abscisse de ce point à l'instant n (on a donc $X_0 = 0$).

On admet que, pour tout entier naturel n , X_n est une variable aléatoire définie un espace probabilisé (Ω, \mathcal{A}, P) que l'on ne cherchera pas à déterminer. On admet aussi que $(X_n)_{n \in \mathbb{N}}$ est une suite de variables aléatoires mutuellement indépendantes. On rappelle qu'en **Scilab**, l'instruction **grand(1,1,'uin',a,b)** permet de simuler une variable aléatoire suivant la loi uniforme à valeurs dans $[[a, b]]$.

1. Écrire des commandes **Scilab** calculant et affichant la valeur de l'abscisse du mobile après son n^e déplacement lorsque la valeur de n est entrée au clavier par l'utilisateur.
2. Compléter le script **Scilab** suivant pour qu'il permette d'afficher dans cet ordre les valeurs prises par les variables aléatoires Y et Z .

```
n = 0
a = 0
while a < 2
    n = n+1
    if grand(1,1,'uin',0,n) == 0 then
        a = a+1
        if a == 1 then y=n,end
    end
end
disp(...,'y=')
disp(...,'z=')
```

3.0.14 EDHEC 2018 (sujet Clermont) - Problème (Voie S)

EXERCICE 3.4 On effectue des lancers d'une pièce donnant pile avec la probabilité p , élément de $]0, 1[$, et donnant face avec la probabilité $q = 1 - p$, les différents lancers étant supposés indépendants. Pour tout entier naturel k non nul, on note P_k (resp. F_k) l'événement : "la pièce donne pile (resp. face) au $k^{\text{ème}}$ lancer", on note également S_k le rang du $k^{\text{ème}}$ pile et T_k le rang d'apparition du dernier pile de la première série de k piles consécutifs. On suppose que S_k et T_k sont deux variables aléatoires toutes deux définies sur le même espace probabilisé (Ω, \mathcal{A}, P) . Par exemple, si les lancers donnent $F_1, P_2, F_3, P_4, F_5, P_6, P_7, P_8$, alors S_1 et T_1 prennent la valeur 2, S_2 prend la valeur 4, T_2 prend la valeur 7, S_3 prend la valeur 6, T_3 prend la valeur 8, S_4 prend la valeur 7 et S_5 prend la valeur 8.

Partie 1 : simulations de S_k et T_k .

1. Compléter les lignes 7 et 10 du script Scilab suivant pour qu'il affiche la valeur prise par S_k lorsque k et p sont entrés par l'utilisateur :

```
k = input('donnez une valeur pour k :')
p = input('donnez une valeur pour p :')
n = 0
c = 0
while c < k
    n = n + 1
    if - - - then c = c + 1
end
end
disp(- - -)
```

2. On souhaite que le script précédent affiche la valeur prise par T_k . Remplacer la ligne 7 par la suivante, dûment complétée :

```
if - - - then c = c + 1, else - - -
```


3.0.15 EDHEC 2019 - Exercice 2 (Voie E)

EXERCICE 3.5 Soit n un entier naturel supérieur ou égal à 3.

Une urne contient une boule noire non numérotée et $n - 1$ boules blanches, dont $n - 2$ portent le numéro 0 et une porte le numéro 1. On extrait ces boules au hasard, une à une, sans remise, jusqu'à l'apparition de la boule noire. Pour chaque i de $\llbracket 1, n - 1 \rrbracket$, on note B_i l'événement : "le i -ème tirage donne une boule blanche", on pose $\overline{B_i} = N_i$ et on note X la variable aléatoire égale au rang d'apparition de la boule noire.

On rappelle qu'en Scilab, la commande `grand(1,1,'uin',a,b)` simule une variable aléatoire suivant la loi uniforme sur $\llbracket a, b \rrbracket$.

1. Compléter le script Scilab suivant afin qu'il simule l'expérience aléatoire décrite dans cet exercice et affiche la valeur prise par la variable aléatoire X .

On admettra que la boule noire est codée tout au long de ce script par le nombre $nB+1$, où nB désigne le nombre de boules blanches.

```

1  n=input('entrez une valeur pour n : ')
2  nB=n-1
3  X=1
4  u=grand(1,1,'uin',1,nB+1)
5  while u<nB+1
6      nB=-----
7      u=grand(1,1,'uin',1,-----)
8      X=-----
9  end
10 disp(X,'la boule noire est apparue au tirage numéro')
```

2. Compléter les lignes 4 et 8 ajoutées au script précédent afin que le script qui suit renvoie et affiche, en plus de celle prise par X , la valeur prise par Y .

```

1  n=input('entrez une valeur pour n : ')
2  nB=n-1
3  X=1
4  Y=-----
5  u=grand(1,1,'uin',1,nB+1)
6  while u<nB+1
7      nB=-----
8      if u==1 then Y=-----
9      end
10     u=grand(1,1,'uin',1,-----)
11     X=-----
12 end
13 disp(X,'la boule noire est apparue au tirage numéro')
14 disp(Y,'la valeur de Y est')
```

3.0.16 Oral HEC 2015 Exercice sans préparation S 132

On considère l'expérience consistant à lancer un dés équilibré n fois de manière consécutives. L'expérience en question est décrite par le programme Scilab ci-dessous pour la valeur $n = 10$ (et répétée ici 1000 fois) :

```

Nexp=1000;
n=10;
u=floor(2*rand(n,Nexp))+1;
test=0;
for k=1:Nexp
    ok=1;
```

```

for i=2:n
    if (u(i-1,k)==1) & (u(i,k)==1) then
        ok=0;
    end
end
test=test+ok;
end
P_n=test/Nexp;
dips(Pn)

```

1. Que représente la variable P_n affichée par le programme ?
2. En notant P_n le résultat de l'expérience précédent pour une valeur de n quelconque, montrer que

$$P_n = \frac{1}{2}P_{n-1} + \frac{1}{4}P_{n-2}$$

Puis trouver la limite de P_n quand n tend vers l'infini.

3.0.17 Oral HEC 2015 Exercice sans préparation S 143

Dans une classe de 30 élèves, on considère une expérience consistant d'abord à demander à chaque élève sa date d'anniversaire. La suite de l'expérience (simuler 1000 sur ordinateur) est décrite par le programme ci-dessous :

```

Nexp=1000;Neleve=30;test=0;
for n=1:Nexp
    anniv=zeros(Neleve,1)
    for i=1:Neleve
        anniv(i)=floor(365*rand())+1;
    end
    anniv=gsort(anniv);ok=0; //gsort=tri par ordre croissant
    for j=1:Neleve-1
        if anniv(j)==anniv(j+1) then
            ok=1;
        end
    end
    test=test+ok;
end
disp(test/Nexp);

```

1. Le code retourne une valeur (à chaque fois différente) autour de 0.71. Que représente cette valeur ?
2. Calculer la valeur exacte de la probabilité simulée par ce programme.
3. Écrire un programme `Scilab` permettant de déterminer le nombre d'élèves à partir duquel cette probabilité dépasse 0.5.

3.0.18 Oral HEC 2016 Exercice sans préparation S 179

On tire (avec remise) une boule d'une urne contenant n boules numérotées.

1. On note T la variable aléatoire égale au numéro du tirage où pour la première fois deux boules différentes ont été tirées.
Déterminer l'espérance de T .
2. Quelle est la variable aléatoire V_n dont la fonction `scilab` suivante calcule une simulation ?

```
function compt=V(n)
    A=[];compt=0;
while length(A) <n
    u=floor(n*rand()+1);
    i=find(A==u); // renvoie de u dans le vecteur A
    if length(i)==0 then
        A=[A,u];
    end;
    compt=compt+1;
end;
endfunction
```


CHAÎNES DE MARKOV

4.0.1 EDHEC 2017 - Problème (Voie E)

Les sommets d'un carré sont numérotés 1, 2, 3, et 4 de telle façon que les côtés du carré relient le sommet 1 au sommet 2, le sommet 2 au sommet 3, le sommet 3 au sommet 4 et le sommet 4 au sommet 1.

Un mobile se déplace aléatoirement sur les sommets de ce carré selon le protocole suivant :

- Au départ, c'est à dire à l'instant 0, le mobile est sur le sommet 1.
- Lorsque le mobile est à un instant donné sur un sommet, il se déplace à l'instant suivant sur l'un quelconque des trois autres sommets, et ceci de façon équiprobable.

3. (a) Compléter le script `Scilab` suivant pour qu'il affiche les 100 premières positions autres que celle d'origine, du mobile dont le voyage est étudié dans ce problème, ainsi que le nombre n de fois où il est revenu sur le sommet numéroté 1 au cours de ses 100 premiers déplacements (on pourra utiliser la commande `sum`).

```
A=[ ..... ] /3
x=grand(100, 'markov', A, 1)
n=.....
disp(x)
disp(n)
```

- (b) Après avoir exécuté cinq fois ce script, les réponses concernant le nombre de fois où le mobile est revenu sur le sommet 1 sont : $n = 23$, $n = 28$, $n = 23$, $n = 25$, $n = 26$. En quoi est-ce normal ?

4.0.2 EDHEC 2016 - Problème (Voie S)

On considère deux urnes U et V contenant chacune 3 boules. Au départ, l'urne U contient 3 boules blanches et l'urne V contient 3 boules noires.

On effectue une suite de tirages dans ces urnes de la façon suivante : chaque tirage consiste à tirer au hasard une boule de chaque urne et à la mettre dans l'autre urne (un tirage est un échange de 2 boules).

Pour tout entier naturel n , on note X_n la variable aléatoire égale au nombre de boules blanches que contient U avant le $(n+1)^{\text{ème}}$ tirage (c'est à dire après le $n^{\text{ème}}$ échange) et on a donc $X_0 = 3$.

- 5.a) Soient n un entier supérieur ou égal à 3 et M la matrice de $\mathcal{M}_4(\mathbb{R})$ dont l'élément de la $(i+1)^{\text{ème}}$ ligne et de la $(j+1)^{\text{ème}}$ colonne est égal à $P_{(X_n=i)}(X_{n+1}=j)$. On admet que M est la matrice donnée à la question 12).

12. On rappelle que la commande `X=grand(n, 'markov', M, X0)` renvoie les n premiers états suivant l'état initial X_0 , d'une chaîne de Markov de matrice M et on rappelle également que `Scilab` assimile un booléen vrai au nombre 1 et un booléen faux au nombre 0.

On considère le script suivant :

```
n=input('entrer la valeur de n :')
M=[0,1,0,0 ; 1/9,4/9,4/9,0 ; 0,4/9,4/9,1/9 ; 0,0,1,0]
X=grand(n, 'markov', M, 4)-1
f=sum(X==0)/n
disp(f)
```

De quelle valeur exacte le contenu de `f` est-il proche lorsque n est assez grand ?

4.0.3 Oral HEC 2015 Exercice sans préparation S 140

Un signal binaire (de valeur -1 ou 1) doit transiter par n relais au passage desquels il est susceptible de changer de valeur avec une probabilité τ . L'expérience est simulée 1000 fois par le programme Scilab ci-dessous :

```
Nexp=1000;n=10;test=0;proba=0.3;
for i=1:Nexp
    ok=0;
    p0=-2*floor(2*rand())+1;
    p=p0;
    for i=1:n
        u=rand()
        if(u<proba) then
            p=-p;
        end
    end
    if(p=p0) then
        ok=1;
    end
    test=test+ok;
end
dips(test/Nexp)
```

Préciser le cas choisi et donner la valeur limite que retournera (approximativement) le programme pour n grand.

STATISTIQUES BIVARIÉES

5.0.1 HEC 2016 (voie E)

13. On rappelle qu'en **Scilab**, les commandes **variance** et **corr** permettent de calculer respectivement la variance d'une série statistique et la covariance d'une série statistique double.

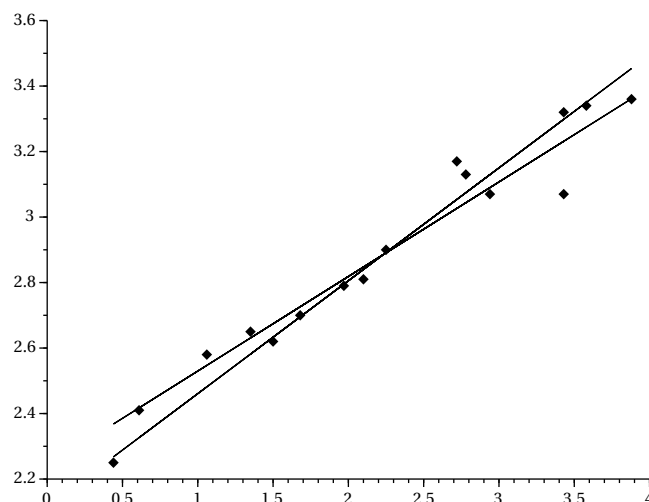
Si $(v_i)_{1 \leq i \leq n}$ et $(w_i)_{1 \leq i \leq n}$ sont deux séries statistiques, alors la variance de $(v_i)_{1 \leq i \leq n}$ est calculable par **variance(v)** et la covariance de $(v_i, w_i)_{1 \leq i \leq n}$ est calculable par **corr(v,w,1)**.

On a relevé pour $n = 16$ entreprises qui produisent le bien considéré à l'époque donnée, les deux séries statistiques $(u_i)_{1 \leq i \leq n}$ et $(t_i)_{1 \leq i \leq n}$ reproduites dans les lignes (1) et (2) du code **Scilab** suivant dont la ligne (5) est incomplète :

```
(1) u=[1.06,0.44,2.25,3.88,0.61,1.97,3.43,2.10,1.50,1.68,2.72,1.35,2.94,2.78,3.4
(2) t=[2.58,2.25,2.90,3.36,2.41,2.79,3.32,2.81,2.62,2.70,3.17,2.65,3.07,3.13,3.0
(3) plot2d(u,t,-4) // -4 signifie que les points sont representes par des losang
(4) plot2d(u,corr(u,t,1)/variance(u)*u+mean(t)-corr(u,t,1)/variance(u)*mean(u))
// equation de la droite de regression de t en u.

(5) plot2d(u,.....)
// equation de la droite de regression de u en t.
```

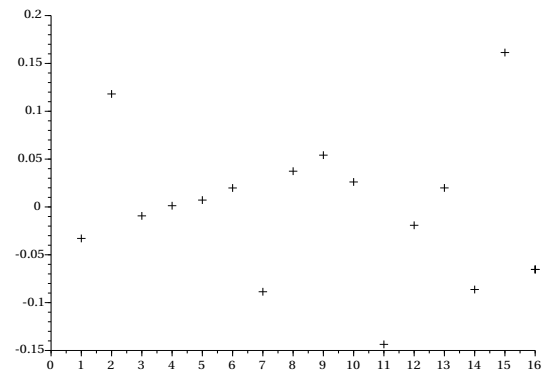
Le code précédent complété par la ligne (5) donne alors la figure suivante :



- Compléter la ligne (5) du code permettant d'obtenir la figure précédente (on reportera sur sa copie, uniquement la ligne (5) complétée).
- Interpréter le point d'intersection des deux droites de régression.

- (c) Estimer graphiquement les moyennes empiriques \bar{u} et \bar{t} .
- (d) Le coefficient de corrélation empirique de la série statistique double $(u_i, t_i)_{1 \leq i \leq 16}$ est-il plus proche de -1, de 1 ou de 0 ?
- (e) On reprend les lignes (1) et (2) du code précédent que l'on complète par les instructions (6) à (11) qui suivent et on obtient le graphique ci-dessous :

```
(6) a0=corr(u,t,1)/variance(u)
(7) b0=mean(t) - corr(u,t,1)/variance(u)
(8) t0=a0*u+b0
(9) e=t0-t
(10) p=1:16
(11) plot2d(p,e,-1)
// -1 signifie que les points sont representes par des symboles d'addition
```



Que représente ce graphique ? Quelle valeur peut-on conjecturer pour la moyenne des ordonnées des 16 points obtenus sur le graphique ? Déterminer mathématiquement la valeur de cette moyenne.

SIMULATION DE VARIABLE ALÉATOIRES CONTINUES

6.0.1 EML 2015 - Exercice 1 (Voie E)

Soit U une variable aléatoire suivant la loi uniforme sur $[0, 1[$. On admet que la variable aléatoire $V = -\frac{1}{\lambda} \ln(1 - U)$ simule une loi exponentielle de paramètre λ

Écrire une fonction en Scilab qui, étant donné un réel λ strictement positif, simule la loi exponentielle de paramètre λ .

6.0.2 EDHEC 2016 - Exercice 3 (voie E)

Dans cet exercice, toutes les variables aléatoires sont supposées définies sur un même espace probabilisé (Ω, A, P) . On désigne par p un réel de $]0, 1[$.

On considère deux variables aléatoires indépendantes U et V , telles que U suit la loi uniforme sur $[-3, 1]$, et V suit la loi uniforme sur $[-1, 3]$.

On considère également une variable aléatoire Z , indépendante de U et V , dont la loi est donnée par:

$$P(Z = 1) = p \text{ et } P(Z = -1) = 1 - p.$$

Enfin, on note X la variable aléatoire, définie par:

$$\forall \omega \in \Omega, X(\omega) = \begin{cases} U(\omega) & \text{si } Z(\omega) = 1 \\ V(\omega) & \text{si } Z(\omega) = -1 \end{cases}$$

On admet enfin que $X = U \frac{1+Z}{2} + V \frac{1-Z}{2}$.

On rappelle que `grand(1,1,'unf',a,b)` et `grand(1,1,'bin',1,p)` sont des commandes Scilab permettant de simuler respectivement une variable aléatoire à densité suivant la loi uniforme sur $[a, b]$ et une variable aléatoire suivant la loi de Bernoulli de paramètre p .

Écrire des commandes Scilab permettant de simuler U , V , Z , puis X .

6.0.3 EDHEC 2018 (épreuve annulée) - Problème - Partie 2 (Voie S)

Soit X une variable aléatoire à densité défini par la densité : $f(x) = \begin{cases} \sin x & \text{si } 0 \leq x \leq \pi/2 \\ 0 & \text{sinon} \end{cases}$

Soit h la restriction de la fonction cosinus à $[0, \frac{\pi}{2}]$. On admet que h réalise une bijection de $[0, \frac{\pi}{2}]$ sur $[0, 1]$. On admet que $Y = h(X)$ est une variable aléatoire, elle aussi définie sur (Ω, \mathcal{A}, P) et que Y suit une loi uniforme.

On rappelle que la commande `grand(1, 1, 'unf', a, b)` renvoie une simulation `Scilab` d'une variable aléatoire à densité suivant une loi uniforme sur $[a, b]$ et on admet que la fonction h^{-1} s'obtient par l'instruction `acos`. Compléter les commandes `Scilab` suivantes afin qu'elles permettent de simuler la variable aléatoire X .

```
Y = grand(1, 1, 'unf', ..., ...)
X = ...
```

6.0.4 EDHEC 2018 (sujet final) - Exercice 3 (Voie S)

EXERCICE 6.1 On admet que toutes les variables aléatoires considérées dans cet exercice sont définies sur le même espace probabilisé que l'on ne cherchera pas à déterminer.

On considère une variable aléatoire X suivant la loi exponentielle de paramètre $\frac{1}{2}$ et on pose $Y = \sqrt{X}$.

1. On rappelle qu'en **Scilab** la commande `grand(1,1,'exp',1/lambda)` simule une variable aléatoire suivant la loi exponentielle de paramètre λ .
Ecrire une (ou des) commande(s) **Scilab** utilisant `grand` et permettant de simuler Y .
2. On pose $U = 1 - e^{-X/2}$, et on admet que U suit une loi uniforme sur $[0, 1]$.
3. Exprimer X en fonction de U , puis en déduire une simulation **Scilab** de Y utilisant uniquement la fonction `rand`.

6.0.5 EDHEC 2018 - Exercice 3 (Voie E)

EXERCICE 6.2 On considère une variable aléatoire à densité X dont une densité est donnée par par : $f(x) = \begin{cases} \frac{x}{a} e^{-x^2/2a} & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases}$

On considère la variable aléatoire Y définie par : $Y = \frac{X^2}{2a}$ et on admet que Y suit une exponentielle de paramètre 1.

On rappelle qu'en **Scilab**, la commande `grand(1,1,'exp',1/lambda)` simule une variable aléatoire suivant la loi exponentielle de paramètre λ . Ecrire un script **Scilab** demandant la valeur de a à l'utilisateur et permettant de simuler la variable aléatoire X .

6.0.6 EDHEC 2019 - Problème (Voie E)

EXERCICE 6.3 Dans cette exercice, θ désigne un réel élément de $\left]0, \frac{1}{2}\right[$.

On considère une variable aléatoire X de densité f définie par : $f(x) = \begin{cases} \frac{1}{\theta x^{1+\frac{1}{\theta}}} & \text{si } x \geq 1 \\ 0 & \text{si } x < 1 \end{cases}$.

On pose $Y = \ln(X)$ et on admet que Y est une variable aléatoire définie sur le même espace probabilisé que X et que Y suit une loi exponentielle de paramètre $\frac{1}{\theta}$.

On rappelle qu'en **Scilab**, la commande `grand(1,1,'exp',1/lambda)` simule une variable aléatoire suivant la loi exponentielle de paramètre λ . Écrire des commandes **Scilab** utilisant `grand` et permettant de simuler X .

6.0.7 ESSEC I 2015 (Voie E)

Dans cette partie, on suppose que la variable aléatoire D représentant la demande admet une densité f qui est nulle sur $]-\infty, 0]$, et continue et strictement positive sur $]0, +\infty[$.

On note R la fonction définie sur $[0, +\infty[$ par $R(x) = P(D > x)$. On admet que R réalise une bijection de $]0, +\infty[$ sur $]0, 1[$ et on pose : $S = R^{-1}\left(\frac{k+c}{v}\right)$

On suppose que l'on a défini une fonction d'entête `function r=R(x)` qui renvoie la valeur de R au point x . Soit X une variable aléatoire qui suit la loi exponentielle de paramètre 1. On admet que $P\left(R(X) \leq \frac{k+c}{v}\right) = e^{-S}$. Compléter le script Scilab qui suit, puis expliquer pourquoi il affiche une valeur approchée de S :

```
k=input('k=') ; c=input('c=') ; v=input('v=') ;

compt = 0;

for i=1:1000 do
    X=grand(1,1,"exp",1)
    if ...
        compt = compt+1;
    end
end

disp('S='); disp(-log(compt/1000));
```

Dans cette partie, on suppose que la variable aléatoire D qui représente la demande est à valeurs dans \mathbb{N} . Sa loi est définie par la donnée de la suite de nombres $(p_n)_{n \in \mathbb{N}}$ avec $p_n = P(D = n)$.

On suppose que pour tout entier naturel n , $p_n > 0$.

On pose $R_n = P(D \geq n)$.

On définit la suite $(\varphi_n)_{n \in \mathbb{N}}$ par $\varphi_n = v \sum_{k=1}^n R_k - (k+c)n$ si $n \geq 1$ et $\varphi_0 = 0$.

On admet que $R_n = p_n + R_{n+1}$, que $\varphi_n - \varphi_{n-1} = vR_n - (k+c)$ et enfin que la suite $(\varphi_n)_{n \in \mathbb{N}}$ est d'abord croissante puis décroissante. Elle admet donc un maximum pour un certain entier S . Ce maximum vaut φ_S .

On suppose que l'on a défini une fonction d'entête `function r=p(n)` qui renvoie la valeur de p_n .

Compléter le script Scilab qui suit pour qu'il affiche $\varphi_0, \dots, \varphi_S$ puis la valeur de S :

```
k=input('k=') ; c=input('c=') ; v=input('v=') ;
n=0 ; phi=0 ; R=1-p(0) ; disp(phi);

while R >= ... do
    n = n+1;
    phi = phi+...;
    disp(phi)
    R = R-...;
end

disp('S='); disp(n);
```

6.0.8 ESSECI 2017 (Voie E)

Est-il possible que le marketing digital pose des problèmes de sécurité des données personnelles? De récents travaux¹, mettant en cause les outils de mesure de performance en temps réel des différentes campagnes de publicité sur internet, démontrent que certaines données très sensibles (préférences religieuses, sexuelles, etc.) peuvent être obtenues par des segmentations précises des audiences et sans aucune action de la part de l'utilisateur.

Dans ce problème, nous nous intéressons à une méthode proposée pour protéger ces données, méthode baptisée **confidentialité différentielle**.

Les parties I et II sont totalement indépendantes. Vous trouverez une aide Scilab en fin de sujet.

On considère un espace probabilisé (Ω, \mathcal{A}, P) sur lequel sont définies les variables aléatoires qui apparaissent dans l'énoncé.

Partie I - Lois de Laplace - propriétés et simulation

Soit $\alpha \in \mathbb{R}$ et $\beta > 0$. On dit qu'une variable aléatoire réelle à densité suit une loi de Laplace de paramètre (α, β) , notée $\mathcal{L}(\alpha, \beta)$, si elle admet comme densité la fonction f donnée par:

$$\forall t \in \mathbb{R}, \quad f(t) = \frac{1}{2\beta} \exp\left(-\frac{|t - \alpha|}{\beta}\right)$$

On admet f est bien une densité et que si X suit la loi $\mathcal{L}(0, 1)$, $\beta X + \alpha$ suit la loi $\mathcal{L}(\alpha, \beta)$.

Simulation à partir d'une loi exponentielle. Soit U une variable aléatoire qui suit la loi exponentielle de paramètre 1 et V une variable aléatoire qui suit la loi de Bernoulli de paramètre $\frac{1}{2}$ et indépendante de U .

1. En utilisant le système complet naturellement associé à V , montrer que $X = (2V - 1)U$ suit la loi $\mathcal{L}(0, 1)$.
2. Compléter la définition Scilab ci-dessous pour que la fonction ainsi définie réalise la simulation d'une variable aléatoire qui suit la loi $\mathcal{L}(\alpha, \beta)$:

```
function r = Laplace(alpha,beta)
if ... <= 1/2
    V = 1
else
    V = 0
end
X = (2*V - 1) * grand(1, 1, "exp", 1)
r = ...
endfunction
```

Aide Scilab. La fonction Scilab `grand` permet de simuler, en particulier, les lois exponentielles et uniformes discrètes. Par exemple:

- `grand(3,2,"exp",0.5)` renvoie une matrice aléatoire (3,2) dont les coefficients sont des variables indépendantes qui suivent la loi exponentielle d'espérance 0,5.
- `grand(1,2,"uin",-1,3)` renvoie une matrice aléatoire (1,2) dont les coefficients sont des variables indépendantes qui suivent la loi uniforme discrètes sur $\llbracket -1, 3 \rrbracket$.

¹Par exemple, A. Korolova. Privacy violations using microtargeted ads: A case study (2010)

6.0.9 HEC 2015 (Voie E)

- On considère une variable aléatoire X à valeurs strictement positives suivant la loi exponentielle de paramètre λ
 - Soit (X_1, X_2, \dots, X_n) un n -échantillon de variables aléatoires à valeurs strictement positives, indépendantes et de même loi que X .

On pose pour tout $n \in \mathbb{N}^*$: $M_n = \max(X_1, X_2, \dots, X_n)$. $T_n = M_n - \frac{1}{\lambda} \ln n$

On admet que la suite de variables aléatoire $(T_n)_{n \in \mathbb{N}^*}$ converge en loi vers une variable aléatoire T dont la fonction de répartition est donnée par $F(x) = \exp(-e^{-\lambda x})$, pour tout $x \in \mathbb{R}$.

11. On suppose dans cette question que $\lambda = 1$.

a) Expliciter la bijection réciproque G de la fonction F .

b) On considère le programme Scilab suivant :

```
x=linspace(-2,2,400);
y=(exp(-exp(-x)));
plot(x,y)
plot(y,x)
```

(i) Le réel 0 fait-il partie des nombres renvoyés par la commande `x=linspace(-2,2,400)`?

(ii) Quel sera le résultat de l'exécution de ce programme?

c) Soit U une variable aléatoire suivant la loi uniforme sur l'intervalle $]0, 1[$.

Quelle est la loi de la variable aléatoire $G(U)$?

d) Établir l'inégalité: $E(T) \leq 1$.

e) Par une méthode de votre choix, écrire en *Scilab* les commandes qui permettent de simuler la loi de T .

f) Écrire en *Scilab* les commandes qui permettent de renvoyer une valeur numérique approchée de $E(T)$ en utilisant la méthode de Monte-Carlo.

6.0.10 Oral HEC 2016 Exercice sans préparation S 89

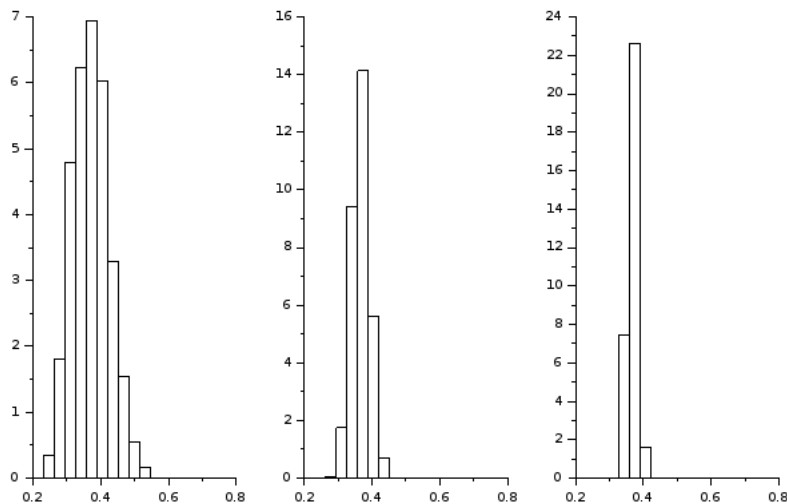
Soient X_1, X_2, \dots, X_n des variables aléatoires indépendantes et identiquement distribuées selon la loi uniforme $\mathcal{U}[0,1]$.

Pour tout $n \in \mathbb{N}^*$, on pose :

$$Y_n = \left(\prod_{i=1}^n X_i \right)^{1/n}$$

1. Expliquez pourquoi la figure ci-dessous, obtenue à partir du code **Scilab** suivant suggère la convergence en probabilité de la suite $(Y_n)_{n \in \mathbb{N}}$

```
x=linspace(0.2,0.8,20);
n=[50 200 800];
for i=1:3
N=1000
LY=sum(log(rand(N,n(i))), "r");//sommation des colonnes
C=exp(LY/n(i));
subplot(1,3,i);//place trois graphiques côte à côte
histplot(x,C);
end
```



2. Justifier la convergence en probabilité de la suite $(Y_n)_{n \in \mathbb{N}}$

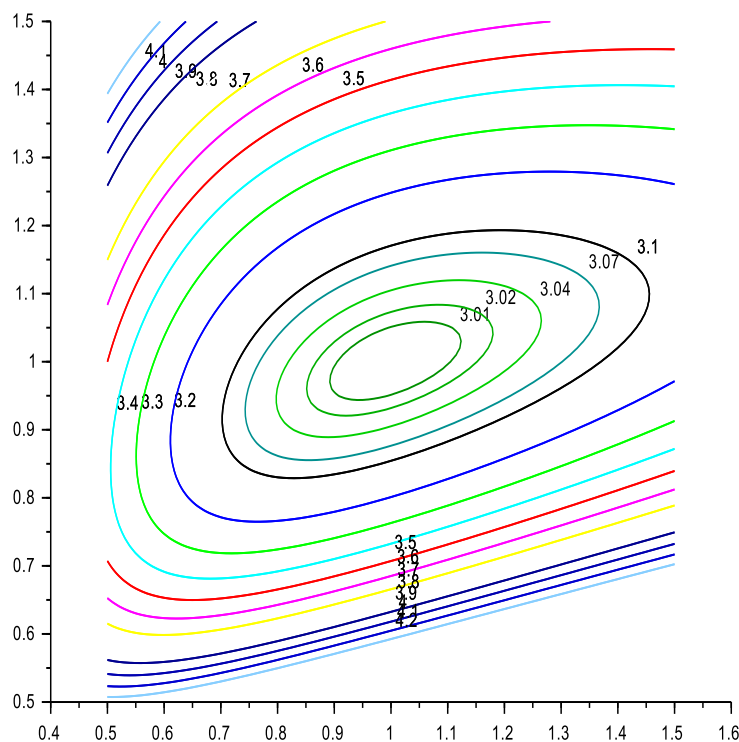
FONCTIONS À DEUX VARIABLES

ECRICOME 2019 - Exercice 2 (Voie E)

EXERCICE 7.1 On considère la fonction f définie sur l'ouvert de $\mathbb{R}^{+*} \times \mathbb{R}^{+*}$ par :

$$\forall (x, y) \in \mathbb{R}^{+*} \times \mathbb{R}^{+*} \quad f(x, y) = \frac{x}{y^2} + y^2 + \frac{1}{x}$$

1. On utilise **Scilab** pour tracer les lignes de niveau de la fonction f . On obtient le graphe suivant :



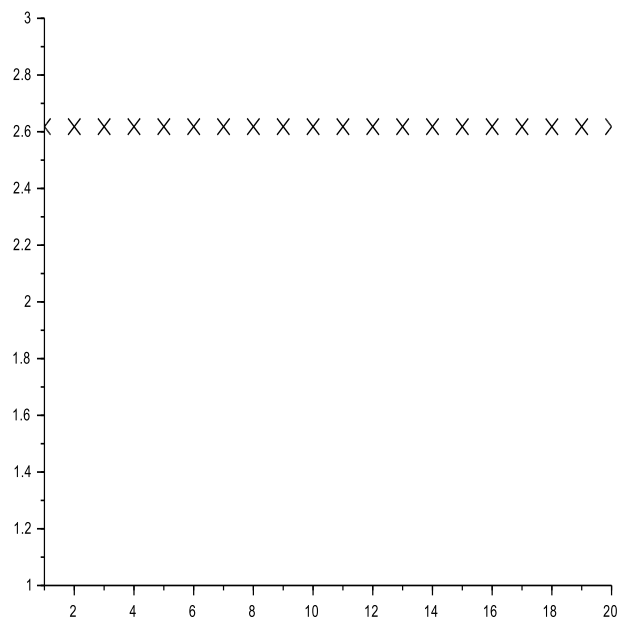
- (b) Compléter la fonction suivante pour qu'elle renvoie une valeur approchée à 10^{-5} près de v_n par la méthode de dichotomie lorsqu'on lui fournit un entier $n \geq 1$ en entrée :

```
function res=v(n)
    a=1
    b=3
    while (b-a)>10^(-5)
        c=(a+b)/2
        if h(n,c) <4 then .....
            else .....
        end
    end
    .....
endfunction
```

- (c) À la suite de la fonction v , on écrit le code suivant :

```
X=1:20
Y=zeros(1,20)
for k=1:20
    Y(k)=v(k)^k
end
plot2d(X,Y,style=-2,rect=[1,1,20,3])
```

À l'exécution du programme, on obtient la sortie graphique suivante :



Expliquer ce qui est affiché sur le graphique ci-dessus.
Que peut-on conjecturer ?

7.0.1 EDHEC 2017 - Exercice 1 (Voie E)

On considère la fonction f qui à tout couple (x, y) de \mathbb{R}^2 associe le réel :

$$f(x, y) = x^4 + y^4 - 2(x - y)^2$$

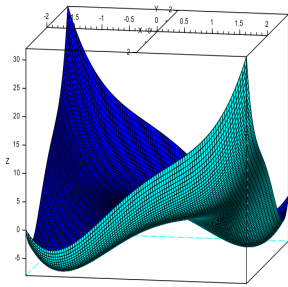
On admet que f possède trois points critiques : $(0, 0)$, $(\sqrt{2}, -\sqrt{2})$, $(-\sqrt{2}, \sqrt{2})$ et qu'elle admet un minimum global sur ces deux derniers points.

1. Compléter la deuxième ligne du script suivant afin de définir la fonction f .

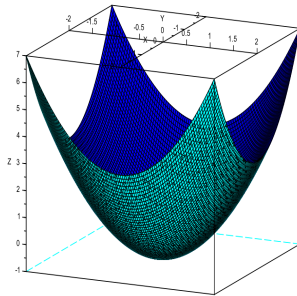
```
function z=f(x,y)
z= ...
endfunction

x=linspace(-2,2,101)
y=x
fplotd3d(x,y,f)
```

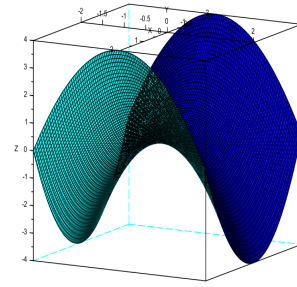
2. Le script précédent, une fois complété, renvoie l'une des trois nappes suivantes. Laquelle ? Justifier la réponse.



Nappe 1



Nappe 2



Nappe 3

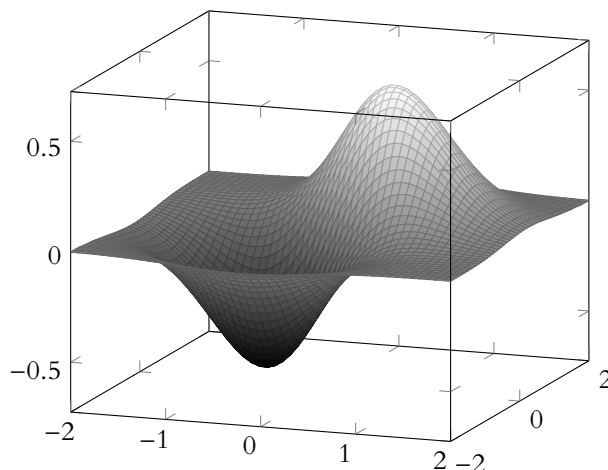
7.0.2 EDHEC 2018 (sujet Clermont) - Exercice 2 (Voie S)

EXERCICE 7.2 Dans cet exercice, n désigne un entier naturel supérieur ou égal à 2. On note I_n la matrice identité de $M_n(\mathbb{R})$ et J_n la matrice de $M_n(\mathbb{R})$ dont tous les éléments valent 1. on considère la fonction f_n définie sur \mathbb{R}^n par :

$$\forall x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n, \quad f_n(x) = \left(\sum_{k=1}^n x_k \right) \exp \left(- \sum_{k=1}^n x_k^2 \right).$$

On admet que f_n admet que deux points critiques $a = \frac{1}{\sqrt{2n}}(1, 1, \dots, 1)$ et $b = -a$, que f_n présente un maximum global en a , un minimum global en b et la hessienne de f_n en a est $H_n(a) = \frac{-2}{\sqrt{2ne}}(nI_n + J_n)$.

1. Ecrire des commandes **Sci lab** permettant de calculer et d'afficher $H_n(a)$ pour une valeur de n entrée par l'utilisateur.
2. Dans le cas $n = 2$, la nappe suivante est-elle acceptable en tant que représentation graphique de la fonction f_2 ? Justifier.



ESTIMATION

8.0.1 ESCP 2017 - Exercice 4 (voie T)

Soit α un réel strictement supérieur à 1. On considère la fonction f définie par :

$$f(t) = 0 \text{ si } t < 1 \text{ et } f(t) = \frac{\alpha}{t^{\alpha+1}} \text{ si } t \geq 1.$$

On admet que f est une densité de probabilité.

Un apiculteur cherche à estimer la durée de vie de ses bougies à la cire d'abeille, c'est-à-dire le temps en heures qu'elles restent allumées en continu avant de s'éteindre. Dans la suite de l'exercice on considère que la durée de vie d'une bougie prise au hasard est une variable aléatoire X ayant pour densité f . On pose $Y = \ln(X)$. On admet que Y est une variable aléatoire à densité

Le paramètre α de la variable aléatoire X est pour l'instant inconnu. On cherche à l'estimer. Pour cela, on allume n bougies au hasard dans des conditions identiques et indépendantes et on mesure les moments où elles s'éteignent. On note X_1, X_2, \dots, X_n les durées de vie respectives de chaque bougie. On pose enfin $Z_n = \frac{1}{n} \sum_{i=1}^n \ln(X_i)$.

On suppose que $n = 100$ et que les valeurs de l'échantillon ont été entrées dans une variable Scilab $X = [X(1), X(2), \dots, X(100)]$ (matrice ligne à 100 colonnes). L'instruction `mean(log(X))` amène le résultat 0,33.

À combien estimez-vous la valeur de α ? À combien estimez-vous la durée de vie moyenne d'une bougie ?

ECRICOME 2019 - Exercice 3 (Voie T)

Soit a un réel strictement positif. On considère la fonction f définie sur \mathbb{R} par :

$$f(x) = \begin{cases} 0 & \text{si } x < a, \\ e^{a-x} & \text{si } x \geq a. \end{cases}$$

et on admet que f est la densité d'une variable aléatoire X .

On considère ensuite la variable aléatoire $Y = X - a$ et on admet que Y suit une loi exponentielle de paramètre 1.

Soit enfin n un entier naturel non nul et X_1, X_2, \dots, X_n des variables aléatoires mutuellement indépendantes, et de même loi que X .

On cherche à estimer le réel a à l'aide de la variable aléatoire $S_n = \frac{1}{n} \sum_{k=1}^n (X_k - 1)$.

On souhaite simuler une réalisation de la variable aléatoire S_n à l'aide de **Scilab**. L'instruction `grand(1, n, 'exp', alpha)` construit un vecteur ligne contenant n coefficients donnant chacun une réalisation d'une variable aléatoire de loi exponentielle de paramètre $\frac{1}{\alpha}$.

Recopier et compléter le code ci-dessous pour qu'il fournisse une réalisation de la variable aléatoire S_{50} pour une valeur de a entrée par l'utilisateur.

```
a = .....
Y = grand(1, ....., 'exp', ..... )
X = Y+a
S = .....
```

ECRICOME 2019 - Exercice 2 (Voie E)

EXERCICE 8.1 On considère la fonction f définie sur l'ouvert de $\mathbb{R}^{+*} \times \mathbb{R}^{+*}$ par :

$$\forall (x, y) \in \mathbb{R}^{+*} \times \mathbb{R}^{+*} \quad f(x, y) = \frac{x}{y^2} + y^2 + \frac{1}{x}$$

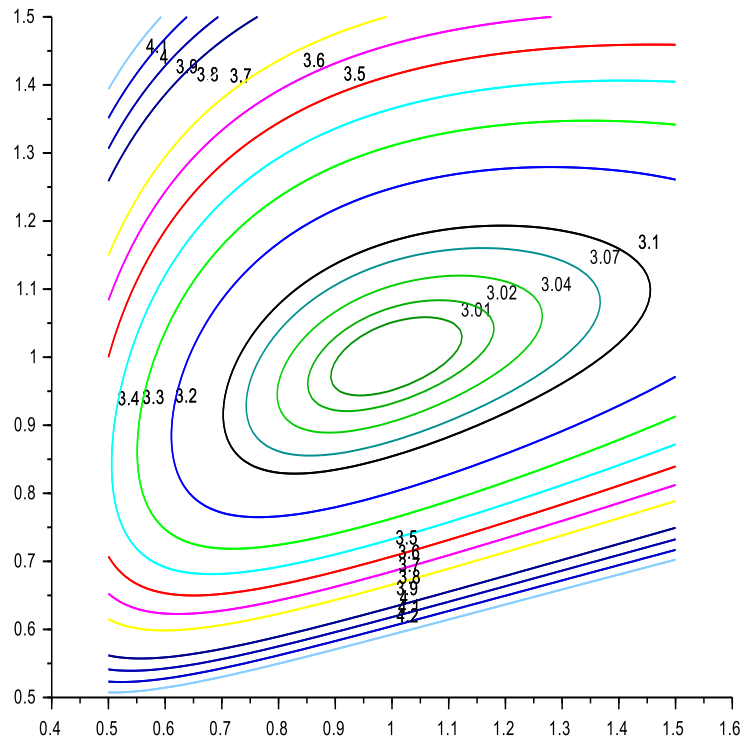
1. On utilise **Scilab** pour tracer les lignes de niveau de la fonction f . On obtient le graphe suivant :
Établir une conjecture à partir du graphique quant à l'existence d'un extremum local pour f , dont on donnera la nature, la valeur approximative et les coordonnées du point en lequel il semble être atteint.
2. Pour tout entier n non nul, on note h_n la fonction définie sur \mathbb{R}^{+*} par :

$$\forall x > 0 \quad h_n(x) = f(x^n, 1) = x^n + 1 + \frac{1}{x^n}$$

On admet que l'équation : $h_n(x) = 4$ admet exactement deux solutions, notées u_n et v_n et vérifiant : $0 < u_n < 1 < v_n$.

- (a) Écrire une fonction **Scilab** d'en-tête `function y=h(n,x)` qui renvoie la valeur de $h_n(x)$ lorsqu'on lui fournit un entier naturel n non nul et un réel $x \in \mathbb{R}^{+*}$ en entrée.
- (b) Compléter la fonction suivante pour qu'elle renvoie une valeur approchée à 10^{-5} près de v_n par la méthode de dichotomie lorsqu'on lui fournit un entier $n \geq 1$ en entrée :

```
function res=v(n)
    a=1
    b=3
    while (b-a)>10^(-5)
        c=(a+b)/2
        if h(n,c) < 4 then .....
            else .....
        end
    end
    .....
endfunction
```



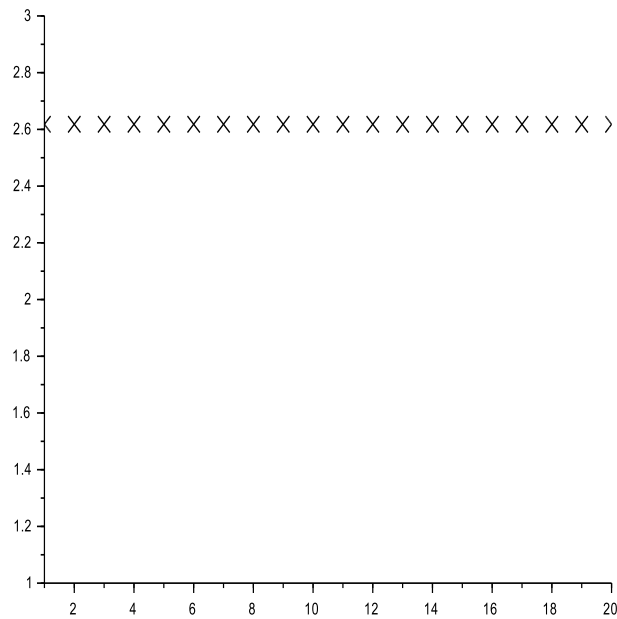
(c) À la suite de la fonction v , on écrit le code suivant :

```
X=1:20
Y=zeros(1,20)
for k=1:20
    Y(k)=v(k)^k
end
plot2d(X,Y,style=-2,rect=[1,1,20,3])
```

À l'exécution du programme, on obtient la sortie graphique suivante :

Expliquer ce qui est affiché sur le graphique ci-dessus.

Que peut-on conjecturer ?



8.0.2 EML 2017 - Exercice 3 (voie E)

On considère une urne contenant initialement une boule bleue et deux boules rouges.

On effectue, dans cette urne, des tirages successifs de la façon suivante : on pioche une boule au hasard, on note sa couleur, puis on la replace dans l'urne en ajoutant une boule de la même couleur que celle qui vient d'être obtenue.

Pour tout k de \mathbb{N}^* , on note :

B_k l'événement : "on obtient une boule bleue au k -ième tirage"

R_k l'événement : "on obtient une boule rouge au k -ième tirage"

Partie I : Simulation informatique

1. Recopier et compléter la fonction suivante afin qu'elle simule l'expérience étudiée et renvoie le nombre de boules rouges obtenues lors des n premiers tirages, l'entier n étant entré en argument.

```
function s=EML(n)
    b=1 // b désigne le nombre de boules bleues présentes dans l'urne
    r=2 // r désigne le nombre de boules rouges présentes dans l'urne
    s=0 // s désigne le nombre de boules rouges obtenues lors des n tirages
    for k=1:n
        x=rand()
        if ... then
            ...
        else
            ...
        end
    end
end
endfunction
```

2. On exécute le programme suivant :


```
n=10  
m=0  
for i=1:1000  
    m=m+EML(n)  
end  
disp(m/1000)
```

On obtient 6.657. Comment interpréter ce résultat?

8.0.3 EDHEC 2015 - Exercice 2 (voie E)

Trois personnes, notées A , B et C entrent simultanément dans une agence bancaire disposant de deux guichets. Les clients A et B occupent simultanément à l'instant 0 les deux guichets tandis que C attend que l'un des deux guichets se libère pour se faire servir.

On suppose que :

- Les durées de passage au guichet des trois personnes A , B et C sont mesurées en heures et on suppose que ce sont des variables aléatoires indépendantes, notées respectivement X , Y et Z , et suivant toutes la loi uniforme sur $[0, 1[$.
- La durée du changement de personne à un guichet est négligeable.

On pose $U = \min(X, Y)$ et $V = \max(X, Y)$ et on admet que U et V sont des variables aléatoires.

On note T le temps total passé par C dans l'agence bancaire.

3. (a) On rappelle que, si \mathbf{a} et \mathbf{b} sont deux vecteurs lignes de taille n , les commandes $\mathbf{m} = \min(\mathbf{a}, \mathbf{b})$ et $\mathbf{M} = \max(\mathbf{a}, \mathbf{b})$ renvoient les vecteurs \mathbf{m} et \mathbf{M} , de même taille que \mathbf{a} et \mathbf{b} , et tels que, pour tout i de $\llbracket 1, n \rrbracket$, on ait : $m(i) = \min(a(i), b(i))$ et $M(i) = \max(a(i), b(i))$.

On rappelle également que `grand(1, n, 'unf', 0, 1)` simule n variables aléatoires indépendantes suivant la loi uniforme sur $[0, 1[$.

Compléter les commandes `Scilab` suivantes pour qu'elles permettent de simuler n fois les variables aléatoires U , V et T , pour n entré par l'utilisateur :

```
n=input('entrez la valeur de n :')
x=grand(1,n,'unf',0,1)
y=grand(1,n,'unf',0,1)
z=grand(1,n,'unf',0,1)
u= ----- ; disp(u, 'u=')
v= ----- ; disp(v, 'v=')
t= ----- ; disp(t, 't=')
```

- (b) Que représente l'événement $(T \geq V)$?
- (c) On souhaite déterminer une valeur approchée de la probabilité $P(T \geq V)$, notée p , en simulant un grand nombre de fois le passage des clients A , B et C aux guichets.
Compléter les commandes `p= ----- ; disp(p, 'p=')` pour que, placées sous les commandes écrites à la question 3.(a), elles permettent d'obtenir une valeur approchée de p .
- (d) Lors de plusieurs essais des commandes ci-dessus, avec $n = 10000$, la réponse donnée par `Scilab` est comprise entre 0.66 et 0.67.
Que peut-on conjecturer quant à la valeur exacte de p ?

8.0.4 EDHEC 2015 - Exercice 2 (Voie S)

On considère une variable aléatoire X suivant une loi normale centrée réduite (espérance nulle et variance égale à 1) et on note Φ la fonction de répartition de X .

On considère 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 que c'est une densité d'une variable aléatoire Z . On admet enfin que

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

1. Ecrire une commande **Scilab** permettant de simuler la variable aléatoire Z .
2. On considère les commandes **Scilab** suivantes :

```
n=input('Entrer la valeur de n : ')
w= grand(1,n,'exp',1)
s=sum(w.*sqrt(w))/n / sqrt( \%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

8.0.5 EDHEC 2017 - Problème (Voie S)

Dans ce problème, n est un entier naturel non nul et $\mathbb{R}_n[X]$ est l'espace vectoriel des polynômes à coefficients réels de degré inférieur ou égal à n .

On note $\mathcal{B} = (e_0, e_1, \dots, e_n)$ la base canonique de $\mathbb{R}_n[X]$. On rappelle que $e_0 = 1$ et que $\forall k \in \llbracket 1, n \rrbracket$, $e_k = X^k$.

Partie 1 : étude d'une application définie sur $\mathbb{R}_n[X]$.

On considère l'application φ qui à tout polynôme P de $\mathbb{R}_n[X]$ associe $\varphi(P) = \sum_{k=0}^n P^{(k)}$, où $P^{(k)}$ désigne la dérivée d'ordre k de P , avec la convention $P^{(0)} = P$.
On admettra que φ est un automorphisme de $\mathbb{R}_n[X]$.

1. Pour tout polynôme P de $\mathbb{R}_n[X]$, calculer $\varphi(P - P')$.
2. Déterminer φ^{-1} , puis écrire la matrice de φ^{-1} dans la base \mathcal{B} .
3. On donne le script **Scilab** suivant :

```
n=input('entrer la valeur de n : ')
M=eye(n+1,n+1)
for k=1:n
    M(k,k+1)=-k
end
A=-----
disp(A)
```

Compléter la sixième ligne de ce script pour qu'il affiche la matrice A de φ dans la base \mathcal{B} lorsque la valeur de n est entrée par l'utilisateur.

Partie 2 : étude d'une autre application définie sur $\mathbb{R}_n[X]$.

On désigne par x un réel quelconque.

On admet que pour tout entier naturel k , $\int_x^{+\infty} t^k \mathbf{E}^{-t} dt$ est convergente et que $\int_x^{+\infty} t^k \mathbf{E}^{-t} dt = k! \sum_{i=0}^k \frac{x^i}{i!} \mathbf{E}^{-x}$.

- On admet que si u est un vecteur, la commande **Scilab** `prod(u)` renvoie le produit des éléments de u et la commande `cumprod(u)` renvoie un vecteur de même format que u dont la coordonnée $n^{\circ}k$ est le produit des k premiers éléments de u . Utiliser l'égalité obtenue à la question 5b) pour compléter le script **Scilab** suivant afin qu'il calcule et qu'il affiche la variable S contenant la valeur de l'intégrale $\int_x^{+\infty} t^k \mathbf{E}^{-t} dt$, les valeurs de x et de k étant entrées par l'utilisateur.

```
k=input('entrer la valeur de k : ')
x=input('entrer la valeur de x : ')
p=prod(1:k)
u=-----
s=p*----- *exp(-x)
disp(s)
```

- Montrer, grâce à un changement de variable simple, que $\int_x^{+\infty} t^k \mathbf{E}^{-t} dt = \mathbf{E}^x \int_0^{+\infty} (u+x)^k \mathbf{E}^{-u} du$
En déduire la commande manquante du script **Scilab** suivant afin qu'il permette de calculer et d'afficher une valeur approchée de $\int_x^{+\infty} t^k \mathbf{E}^{-t} dt$ grâce à la méthode de Monte Carlo.

```
k=input('entrer la valeur de k : ')
x=input('entrer la valeur de x : ')
Z=grand(1,100000,'exp',1)
s=exp(-x)* mean(-----)
disp(s)
```

8.0.6 EDHEC 2018 - Exercice 2 (Voie E)

EXERCICE 8.2 On considère la fonction f qui à tout réel x associe : $f(x) = \int_0^x \ln(1+t^2) dt$.

- On rappelle qu'en **Scilab**, la commande `grand(1,1,'unf',a,b)` simule une variable aléatoire uniforme sur $[a;b]$. Compléter le script **Scilab** suivant pour qu'il calcule et affiche, à l'aide de la méthode de Monte-Carlo, une valeur approchée de $f(1)$:

```
U=grand(1,100000,'unf',0,1)
V=log(1+U.^2)
f=-----
disp(f)
```

- On pose $u_0 = 1$, et pour tout entier naturel n non nul, $u_n = \int_0^1 (\ln(1+t^2))^n dt$.
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 $\sum_{k=0}^{+\infty} u_k$

8.0.7 HEC 2015 (Voie E)

- On considère une variable aléatoire X à valeurs strictement positives suivant la loi exponentielle de paramètre λ
 - Soit (X_1, X_2, \dots, X_n) un n -échantillon de variables aléatoires à valeurs strictement positives, indépendantes et de même loi que X .

On pose pour tout $n \in \mathbb{N}^*$: $M_n = \max(X_1, X_2, \dots, X_n)$. $T_n = M_n - \frac{1}{\lambda} \ln n$

On admet que la suite de variables aléatoire $(T_n)_{n \in \mathbb{N}^*}$ converge en loi vers une variable aléatoire T dont la fonction de répartition est donnée par $F(x) = \exp(-e^{-\lambda x})$, pour tout $x \in \mathbb{R}$.

11. On suppose dans cette question que $\lambda = 1$.

a) Expliciter la bijection réciproque G de la fonction F .

b) On considère le programme Scilab suivant :

```
x=linspace(-2,2,400);
y=(exp(-exp(-x)));
plot(x,y)
plot(y,x)
```

(i) Le réel 0 fait-il partie des nombres renvoyés par la commande `x=linspace(-2,2,400)`?

(ii) Quel sera le résultat de l'exécution de ce programme?

c) Soit U une variable aléatoire suivant la loi uniforme sur l'intervalle $]0, 1[$.

Quelle est la loi de la variable aléatoire $G(U)$?

d) Établir l'inégalité: $E(T) \leq 1$.

e) Par une méthode de votre choix, écrire en *Scilab* les commandes qui permettent de simuler la loi de T .

f) Écrire en *Scilab* les commandes qui permettent de renvoyer une valeur numérique approchée de $E(T)$ en utilisant la méthode de Monte-Carlo.

8.0.8 HEC 2017 - Problème (voie E)

$G_{a,b}$ la fonction définie sur \mathbb{R}^+ par : $G_{a,b}(x) = \exp\left(-ax - \frac{b}{2}x^2\right)$.

On admet que $G_{a,b}$ réalise une bijection de \mathbb{R}^+ sur l'intervalle $]0, 1]$ et que pour tout u dans $[0, 1[$, $G_{a,b}^{-1}(1-u) = \frac{-a + \sqrt{a^2 - 2b \ln(1-u)}}{b}$

On dit qu'une variable aléatoire suit la loi exponentielle linéaire de paramètres a et b , notée $\mathcal{E}_\ell(a, b)$ si pour tout réel $x \in \mathbb{R}^+$, on a :
$$P([X \leq x]) = \begin{cases} 1 - G_{a,b}(x) & \text{Si } x \in \mathbb{R}^+ \\ P([X \leq x]) = 0 & \text{Sinon} \end{cases}.$$

On admet enfin que si Y est une variable aléatoire suivant la loi exponentielle de paramètre 1, alors $X = \frac{-a + \sqrt{a^2 + 2bY}}{b}$ suit une loi $\mathcal{E}_\ell(a, b)$.

1. On note U une variable aléatoire suivant la loi uniforme sur $[0, 1]$. Déterminer la loi de la variable aléatoire $G_{a,b}^{-1}(1-U)$.
2. La fonction `Scilab` suivante génère des simulations de la loi exponentielle linéaire.

```
(1)  function x=grandlinexp(a,b,n)
(2)      u=rand(n,1)
(3)      y= ....
(4)      x=(-a+sqrt(a^2+2*b*y))/b
(5)  endfunction
```

- (a) Quelle est la signification de la ligne de code (2) ?
 - (b) Compléter la ligne de code (3) pour que la fonction `grandlinexp` génère les simulations désirées.
3. De quel nombre réel peut-on penser que les six valeurs générées par la boucle `Scilab` suivante fourniront des valeurs approchées de plus en plus précises et pourquoi ?

```
for k=1:6
    mean(grandlinexp(0,1,10^k))
end
```

8.0.9 ESSEC I 2016 (Voie E)

On s'intéresse dans ce problème à deux mesures du risque utilisées par les marchés financiers.

Dans tout le problème β est un réel appartenant à $]0, 1[$ et représentant un niveau de confiance.

On définit alors $r_\beta(X)$ appelé la « Value at Risk » au niveau de confiance β de X . C'est une grandeur qui permet d'évaluer le risque pris par l'acteur qui détient l'actif dont les pertes sont modélisées par X . On remarque que $r_\beta(X)$ est égal au capital minimal qu'il faut détenir pour être en mesure de couvrir les pertes de l'actif associé à X avec une probabilité égale à β

Soient $(X_k)_{k \geq 1}$ est une suite de variables aléatoires, mutuellement indépendantes, de même loi que X . Pour tout $\omega \in \Omega$ et $n \in \mathbb{N}^*$, on ordonne $X_1(\omega), \dots, X_n(\omega)$ dans l'ordre croissant et on note alors $X_{1,n}(\omega), \dots, X_{n,n}(\omega)$ les valeurs obtenues. En particulier, $X_{1,n}(\omega)$ est la plus petite des valeurs $X_1(\omega), \dots, X_n(\omega)$ et $X_{n,n}(\omega)$ la plus grande.

On définit pour tout $n \in \mathbb{N}^*$ tel que $n\beta > 1$, la variable aléatoire Y_n sur (Ω, \mathcal{A}, P) par $Y_n = X_{[n\beta],n}$ où $[n\beta]$ désigne la partie entière de $n\beta$. Et on admet que Y_n est un estimateur convergent de $r_\beta(X)$

11. On suppose que l'on a défini une fonction d'en-tête `function R=triCroissant(T)` qui renvoie le tableau des valeurs se trouvant dans `T` rangées dans l'ordre croissant.

Par exemple, si `T=[0 -1 0 2 4 2 3]` alors `disp(triCroissant(T))` affiche :

`ans =`

`-1. 0. 0. 2. 2. 3. 4.`

Écrire une fonction Scilab d'en-tête `function r=VaR(X,beta)` qui renvoie la valeur de l'estimation obtenue avec l'estimateur Y_n pour $r_\beta(X)$ si le tableau `X` contient la réalisation de l'échantillon (X_1, \dots, X_n) et `beta` la valeur de β .

On admet que $E(\max(X - r_\beta(X), 0))$ est bien défini et on appelle l'« Expected Shortfall » de X de niveau de confiance β la quantité : $ES_\beta(X) = r_\beta(X) + \frac{1}{1-\beta} E(\max(X - r_\beta(X), 0))$.

19. En utilisant la méthode de Monte-Carlo, dont on supposera la validité, et la fonction `VaR` définie dans la question 11, écrire une fonction Scilab qui calcule une valeur approchée de $ES_\beta(X)$ à partir de la réalisation d'un échantillon de taille n de la loi de X dont les valeurs se trouvent dans le tableau Scilab `X` et de la valeur de β se trouvant dans la variable Scilab `teta`.

8.0.10 HEC I 2017 (voie S)

Soit f une fonction continue sur $[0, 1]$. Soit $z \in [0, 1]$. Pour tout $n \in \mathbb{N}^*$, soit Z_n une variable aléatoire définie sur cet espace, et suivant la loi binomiale de paramètres n et z . Pour tout $n \in \mathbb{N}^*$, on pose $\bar{Z}_n = \frac{Z_n}{n}$. On admet que la suite de variables aléatoires $(\bar{Z}_n)_{n \in \mathbb{N}^*}$ converge en probabilité vers le réel z et que $\lim_{n \rightarrow +\infty} E(f(\bar{Z}_n)) = f(z)$.

1. Compléter le code Scilab suivant afin qu'un appel à la fonction `binom(n,z)` renvoie une réalisation d'une loi binomiale de paramètres n et z .

```
function Z=binom(n,z)
Z=.....
endfunction
```

2. Soit une fonction Scilab `f` et une variable `Z` définies par :

```
function y=f(x)
if x==0 then y=0 else y= -x*log(x) ; end
endfunction
z=0.4
```

On considère le code Scilab suivant :

```
n=100 ; N=1000
S=0
for k=1:N
S=S+f(binom(n,z)/n)
end
disp(S/n)
```

Ce code affiche une valeur approchée d'une certaine quantité, laquelle ?

Cette valeur approchée est le résultat de la mise en oeuvre de certaines méthodes, lesquelles ?

8.0.11 Oral HEC 2016 Exercice sans préparation S 173

Donner la finalité du programme suivant :

```
N=100000; S=0;
for i=1:N
    u=rand();
    S=S+4/N*1/(1+u^2);
end
disp(S)
```

On pourra penser à la loi des grands nombres.

8.0.12 Oral HEC 2016 Exercice principal S 171

Soit $G_1, G_2, \dots, G_n, \dots$ une suite de variables aléatoires indépendantes et identiquement distribuées selon la loi normale centrée réduite $N(0, 1)$.

Pour tout $N \in \mathbb{N}^*$, on note $Z_N = \subset 1 \leq n \leq NG_n$ et F_N la fonction de répartition de Z_N .

1. Justifier l'existence de l'espérance de Z_N .
2. Soit le code Scilab

```
Y=zeros(1,5);
for k=[1:5]
    n=1000;
    N=10^k;
    X=grand(N,n,"nor",0,1);
    Xmax=max(X,'r');//vecteur-ligne des maxima des colonne de X
    Y(k)=mean(Xmax);
end;
disp(Y./sqrt(2*[1:5]*log(10)));
```

qui retourne l'affichage :

```
0.7093236 0.8216955 0.8755261 0.8996736 0.9138259
```

Expliquer comment il fonctionne et ce qu'il calcule.

AUTRE

9.0.1 ECRICOME 2017 - Exercice 3 (Voie T)

Soit h la fonction définie sur \mathbb{R} par :

$$\forall x \in \mathbb{R}, \quad h(x) = \begin{cases} \frac{4 \ln x}{x^3} & \text{si } x \geq 1 \\ 0 & \text{si } x < 1 \end{cases}$$

On admet que h est une densité de probabilité et on considère dans la suite de cet exercice une variable aléatoire X dont h est une densité.

1. Calculer la fonction de répartition F de la variable aléatoire X .
2. Compléter sur votre copie les lignes 3 à 5 du programme Scilab suivant pour que la fonction F prenne en entrée un réel x et calcule la valeur de $F(x)$.

```

1  function calcul=F(x)
2      if x < 1 then
3          .....
4      else
5          .....
6      end
7  endfunction
8
9  for i=1:300, a(i)=-2+7*i/300; b(i)=F(a(i));
10 end
11 plot(a,b)

```

3. Qu'obtient-on lors de l'exécution des lignes 9 à 11 du programme précédent ?

9.0.2 EDHEC 2015 - Exercice 1 (Voie E)

On considère la matrice $C = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$ Compléter, à l'aide de matrices de type `zeros` et `ones`, les deux espaces laissés libres dans la commande Scilab suivante pour qu'elle permette de construire la matrice C .

`C = [ones(1,5) ; ----- , ----- ; ones(1,5)]`

9.0.3 EDHEC 2017 - Exercice 1 (Voie S)

EXERCICE 9.1 Soit n un entier naturel supérieur ou égal à 1. On considère la fonction f_n définie par :

$$\forall x \in [0, 1], f_n(x) = \sum_{k=1}^n x^k$$

1. (a) Compléter la fonction **Scilab** suivante pour qu'elle renvoie la valeur de $f_n(x)$ à l'appel de $f(x, n)$, où x et n sont donnés par l'utilisateur.

```
function y=f(x,n)
    y=sum(-----)
endfunction
```

- (b) Transformer, pour $x \neq 1$, l'expression de $f_n(x)$ puis en déduire une deuxième façon de déclarer f , en complétant la déclaration suivante où la fonction est toujours nommée f .

```
function y=f(x,n)
    if x==1 then y=-----
    else y=-----
    end
endfunction
```

2. On admet que l'équation $f_n(x) = 1$, d'inconnue $x \in [0, 1]$, possède une unique solution α_n dans $[0, 1]$.

On suppose que f_n a été déclarée (cf question 1). On considère les commandes supplémentaires suivantes :

```
n=input('entrer la valeur de n : ')
x=0
while f(x,n)<1
    x=x+0.001
end
disp(x)
```

Quel est le lien entre le résultat affiché et α_n ?

9.0.4 EDHEC 2019 - Exercice 1 (Voie S)

EXERCICE 9.2 On note Id l'endomorphisme identité de \mathbb{R}^3 et on considère l'endomorphisme f de \mathbb{R}^3 dont la matrice dans la base canonique est :

$$A = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 3 & -2 \\ -1 & 1 & 0 \end{pmatrix}$$

En **Scilab**, la commande $r=\text{rank}(M)$ renvoie dans la variable r le rang de la matrice M . On a saisi :

```
A=[1,0,0;-2,3,-2;-1,1,0]
r1=rank(A-eye(3,3))
r2=rank(A-2*eye(3,3))
disp(r1,'r1=')
disp(r2,'r2=')
```

Scilab a renvoyé :

```
r1 =
    1.
r2 =
    2.
```

Que peut-on conjecturer quant aux valeurs propres de f et à la dimension des sous-espaces propres associés ?

9.0.5 EDHEC 2019 - Problème (Voie S)

EXERCICE 9.3 Dans cette partie, n désigne toujours un entier naturel non nul.

6. On admet que, si a et b sont des entiers tels que $a < b$, la commande `grand(1,1,'uin',a,b)` permet à `Scilab` de simuler une variable aléatoire suivant la loi uniforme discrète sur $\llbracket a, b \rrbracket$.

Compléter le script suivant pour que les lignes (5), (6), (7) et (8) permettent d'échanger les contenus des variables $A(j)$ et $A(p)$.

```
(1) n=input('entrez une valeur pour n :')
(2) A=1:n
(3) p=n
(4) for k=1:n
(5)   j=grand(1,1,'uin',1,p)
(6)   aux=----
(7)   A(j)=----
(8)   A(p)=----
(9)   p=p-1
(10) end
(11) disp(A)
```

7. On suppose dorénavant qu'après exécution du script précédent correctement complété, le vecteur A est rempli de façon aléatoire par les entiers de $\llbracket 1, n \rrbracket$ de telle sorte que les $n!$ permutations soient équiprobables. On considère alors les commandes `Scilab` suivantes (exécutées à la suite du script précédent) :

```
m=A(1)
c=1
for k=2:n
    if A(k)>m then m=A(k)
                  c=k
    end
end
disp(c)
```

- Expliquer pourquoi, à la fin de la boucle `for`, la variable m contient la valeur n .
- Quel est le contenu de la variable c affiché à la fin de ces commandes?
- On rappelle qu'en `Scilab`, l'instruction `find(test)` permet de trouver à quelle(s) place(s) se trouvent les éléments d'une matrice satisfaisant au test proposé.
Compléter le script `Scilab` ci-dessous afin qu'il renvoie et affiche le contenu de la variable C étudiée plus haut :

```
c=find(---)
disp(c)
```

9.0.6 HEC II-ESCP 2015 (voie S)

Soit Φ la fonction de répartition de la loi normale centrée réduite dont la dérivée, notée φ , est telle que: $\forall t \in \mathbb{R}, \varphi(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}$.

$$\mathbb{R}, \varphi(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}.$$

On admet que Φ est une bijection de \mathbb{R} sur $]0, 1[$. On note Ψ la bijection réciproque de Φ .

Soit $\alpha \in]0, \frac{1}{2}[$ et w_α la fonction définie sur $[0, 1]$ telle que: $w_\alpha(x) = \begin{cases} 0 & \text{si } x=0 \\ \Phi(\Psi(x) - \Psi(\alpha)) & \text{si } 0 < x < 1 \\ 1 & \text{si } x=1 \end{cases}$.

Pour faire tracer par *Scilab* la courbe représentative de w_α , on utilise la fonction `cdfnor` qui permet de calculer

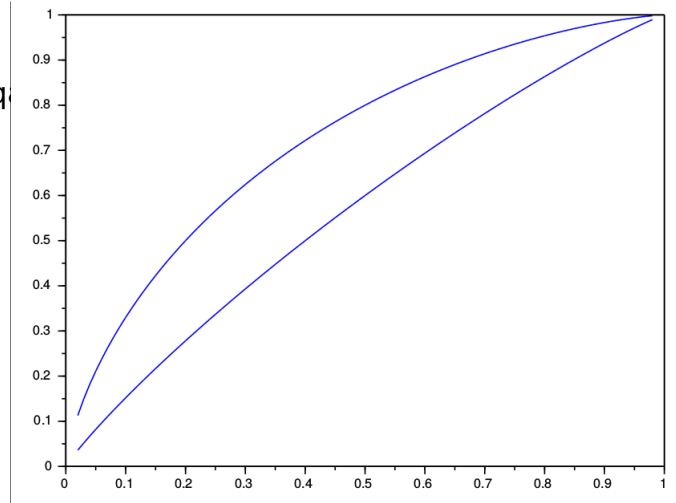
les valeurs de la fonction de répartition de variables aléatoires de loi normale.

Si une variable aléatoire Z suit la loi normale centrée réduite et si x et p sont deux réels reliés par l'égalité $P([Z \leq x]) = p$, alors:

- p est calculable en *Scilab* par `cdfnor("PQ", x, 0, 1);`
- x est calculable en *Scilab* par `cdfnor("X", 0, 1, p, 1-p).`

Le graphique ci-dessous a été obtenu en affectant successivement à la variable `alpha` les valeurs 0.2 et 0.4, et en exécutant les cinq instructions codées comme suit, la quatrième étant incomplète.

```
(1) qa=cdfnor("X", 0, 1, alpha, 1-alpha)
(2) p=[0.02:0.01:0.98]
(3) q=cdfnor("X", zeros(p), ones(p), p, 1-p) - qa
(4) wa=cdfnor("PQ", ?, ?, ?)
(5) plot(p, wa)
```



- Quelles sont les valeurs affectées aux variables p et q par les instructions (2) et (3) (on en précisera le format matriciel)?
- Compléter la quatrième ligne de code.

CORRECTION 1.1

```

2.c)  U=zeros(1,100)}
      U(1)=1
      for n=1:99
          U(n+1)=1-exp(-U(n))
      end
      plot(U,"+")

```

2.d) On peut conjecturer que la suite (u_n) est décroissante et tend vers 0.

2.i) $S=\text{cumsum}(U)$ donc les coordonnées de S sont $(u_1, u_1 + u_2, u_1 + u_2 + u_3, \dots, \sum_{k=1}^{100} u_k)$ autrement dit les sommes partielles de la série de terme général u_n .
Ainsi on peut conjecturer que la suite des sommes partielles tend vers $+\infty$ autrement dit que la série de terme général u_n diverge.

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.2

2.e) Pour tout $n \geq 1$, on a $(n+1)^{n+1} \geq (n+1)^2 \geq n^2$ et donc

$$\forall n \geq 1, |u_n| \leq \frac{1}{n^2}$$

Or la série de terme général $\frac{1}{n^2}$ est convergente, donc la série de terme général $|u_n|$ est convergente.
Donc la série de terme général u_n est absolument convergente et donc convergente.

```

2.f) function S=somme(n)
      S=1
      for k=1:n
          S=S+(-1)**k/(k+1)**(k+1)
      end
      endfunction

```

3.d) Il suffit de prendre n tel que $\frac{1}{e^{n+1}(n+1)!} \leq \varepsilon$ pour assurer que S_n donne une valeur approchée de I à ε près :

```

function I=estimation (eps)
    n=0
    facto=1
    while exp(n+1)*facto < 1/eps
        n=n+1
        facto=facto*(n+1)
    end
    I=somme(n)
endfunction

```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.3

1. Le programme demandé reprend la structure classique d'un programme permettant le calcul des termes d'une suite à récurrence linéaire d'ordre 2.

```
function res=X(n)
    Xold=[3;0;-1]
    Xnew=[3;0;-2]
    A=[2,2,-2;0, 3,0; 1, -1, 5]
    B=[1,-1,-1; -3,3,-3;-1, 1, 1]
    for i=2:n
        Aux=1/6*A*Xold+1/6*B*Xnew
        Xold=Xnew
        Xnew=Aux
    end
    res=Xnew
endfunction
```

2. Pour chaque suite, le dernier terme représenté correspond à $n = 10$. Pour cette valeur de n , les termes de chaque suite sont proches de leurs limites. On voit que

$$\lim_{n \rightarrow +\infty} \alpha_n = \frac{4}{3} + \frac{1}{2} = \frac{11}{6} \sim 1.8, \quad \lim_{n \rightarrow +\infty} \beta_n = -\frac{4}{3} \sim -1.3$$

Ainsi, la suite avec des croix \times correspond à (α_n) , celle avec des croix entourées à (β_n) et celle avec des losanges à (γ_n) .

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.4

```
function u=suite(n)
    v=0
    w=1
    for k=2:n // ici v vaut u(k-2) et w vaut u(k-1)
        temporaire = v+w // temporaire vaut u(k-2) + u(k-1) = u(k)
        v = w // v vaut u(k-1)
        w = temporaire // w vaut u(k)
    end // en sortie de boucle k=n et w vaut u(n)
    u= w
endfunction
```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.5

1. C'est un petit programme sans réelle difficulté que l'on peut faire avec une boucle `for` ou avec une opération pointée. On propose les deux versions.

```
function y=u(n)
    y=0;
    for k=1:n
        y=y+1/k;
    end
    y=y-log(n)
endfunction
```

ou bien

```
function y=u(n)
    y=sum([1:n].^(-1))-log(n)
endfunction
```

2. Le programme proposé permet de calculer une approximation de γ à la précision `eps` près (rentrée par l'utilisateur). En effet, une telle approximation sera réalisée par un terme u_n tel que $|u_n - \gamma| < \text{eps}$, ce qui, d'après la question précédente a lieu dès que $1/n < \text{eps}$. Il suffit de prendre le premier entier n tel que $n > 1/\text{eps}$, donné par $\lceil 1/\text{eps} \rceil + 1$.

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.6

1. Je prends l'initiative de ne pas placer I_0 dans le tableau.

```
function y=I(n)
    u=zeros(1,2*n+2)
    //u(0)= %pi/2 ce terme n'existe pas dans le tableau
    u(1) = 1
    u(2) = %pi/4
    for k=1:n
        u(2*k+1) = (2*k)*u(2*k-1)/(2*k+1)
        u(2*k+2) = (2*k+1)*u(2*k)/(2*k+2)
    end
    y = u
endfunction
```

2. function S=somme(n)
 S = %pi/2
 tableau = I(floor(n/2))// sinon on laisse I(n), tant pis pour les termes su
 for j=1:n
 S=S+tableau(j)
 end
 endfunction

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.7

On cherche une valeur approchée de la somme infinie $S = \sum_{k=1}^{+\infty} \frac{1}{f(k)}$. Or sur scilab, nous ne pouvons calculer que des sommes finies, donc nous allons faire calculer à scilab une bonne approximation de cette somme qui est la somme partielle $S_n = \sum_{k=1}^n \frac{1}{f(k)}$.

Regardons à présent : $|S - S_n|$ mesure l'écart qu'il y a entre S et S_n . Puisqu'on souhaite une approximation de S à 10^{-4} près, il faut que cet écart soit inférieur à 10^{-4} .

D'après l'inégalité de la question 7, Il suffit donc de prendre un n tel que $\frac{1}{(e-1)e^n}$ soit plus petit que 10^{-4} pour que l'écart entre S_n et S soit plus petit que 10^{-4} .

Nous allons donc faire calculer à scilab S_1 puis S_2 puis S_3 et lui faire continuer à calculer le terme suivant tant que la quantité $\frac{1}{(e-1)e^n}$ n'est pas plus petite que 10^{-4}

```
function y=approx()
    n=1 // n representera le notre compteur. On l'initialise a 1
    S=1/( exp(1) ) // S represente La somme Sn. On l'initialise a S1=1/f(1)
    // tant que la precision n'est pas atteinte, on continue de calculer
    while (1/((%e-1)*exp(n)) > 0.0001)
        //Sn n'etait donc pas assez precis, calculons S(n+1)
        S=S + 1/( n^3 * exp(n+1) ) // On utilise S(n+1)=Sn + 1/f(n+1)
        n=n+1 // on incremente notre compteur
    end
    y=S // on a atteint la precision suffisante, la fonction renvoie la valeur S
endfunction
```

REMARQUES 1.1. • En sortie du `while`, S contient $\sum_{k=1}^n \frac{1}{f(k)}$ et n contient $n+1$.

- Comme, on n'est pas rentré dans le `while` pour la 1ère valeur de n_0 qui vérifie $1/((\text{il faut donc calculer ici})$
 $S_{n_0} = \sum_{k=1}^{n_0} \frac{1}{f(k)}$

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.8

- Soit $A > 0$, $\int_0^A \frac{1}{(1+t^2)^1} dt = [\text{Arctan}(t)]_0^A = \text{Arctan}(A) - \text{Arctan}(0) = \text{Arctan}(A) \xrightarrow{A \rightarrow +\infty} \frac{\pi}{2}$.

Bilan : $H(1) = \frac{\pi}{2}$.

- Soit $n \in \mathbb{N}^*$. Soit $A > 0$. On pose $u'(t) = 1$, $u(t) = t$, $v(t) = \frac{1}{(1+t^2)^n}$, $v'(t) = \frac{-n(2t)}{(1+t^2)^{n+1}}$. Les fonctions u, v sont C^1 sur \mathbb{R}^+ .

$$\int_0^A \frac{1}{(1+t^2)^n} dt = \left[\frac{t}{(1+t^2)^n} \right]_0^A - \int_0^A \frac{-2nt^2}{(1+t^2)^{n+1}} dt = \frac{A}{(1+A^2)^n} + 2n \int_0^A \frac{(t^2+1)-1}{(1+t^2)^{n+1}} dt = \frac{A}{(1+A^2)^n} + 2n \int_0^A \frac{1}{(1+t^2)^{n+1}} dt - 2n \int_0^A \frac{1}{(1+t^2)^{n+1}} dt.$$

On a $\frac{A}{(1+A^2)^n} \underset{A \rightarrow +\infty}{\sim} \frac{1}{A^{2n-1}} \underset{A \rightarrow +\infty}{\longrightarrow} 0$ car $2n-1 > 0$.

De sorte que, par passage à la limite, $\int_0^A \frac{1}{(1+t^2)^n} dt \xrightarrow{A \rightarrow +\infty} 2nH(n) - 2nH(n+1)$.

Bilan : $H(n) = 2n(H(n) - H(n+1))$.

On développe et $H(n) = 2nH(n) - 2nH(n+1)$ donc $2nH(n+1) = (2n-1)H(n)$ donc

$$H(n+1) = \frac{2n-1}{2n} H(n).$$

3.

```

function H=emlyon17(n)
    H=%pi/2
    for k=2:n // H vaut H(k-1)
        H=(2*(k-1)-1)*H/(2*(k-1))
        // H vaut (2*(k-1)-1)*H(k-1)/(2*(k-1))=H(k)
    end
    // en sortie H vaut H(n)
    // car la dernière valeur de k est n
endfunction

```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.9

1. (a) On peut compléter le programme comme suit :

```

function y=f(x,n)
    y=sum(x.^(1:n))
endfunction

```

- (b) Pour $x \neq 1$, on a : $f_n(x) = \frac{1-x^{n+1}}{1-x} - 1$ et pour $x = 1$, $f_n(x) = n$. Le programme peut alors s'écrire :

```

function y=f(x,n)
    if x==1 then
        y=n
    else
        y=(1-x^(n+1))/(1-x)-1
    end
endfunction

```

2. Fixons $n \in \mathbb{N}^*$. On sait déjà que f_n est croissante sur $[0, 1]$ et $f_n(0) = 0$. Le programme va alors tester toutes les valeurs de $f_n(x)$ à partir de $x = 0$ et avec un pas de 0,001 tant que $f_n(x) < 1$. La valeur affichée par le programme est alors le premier de ces nombres pour lequel $f_n(x) \geq 1$. Le résultat affiché est donc le plus petit nombre du type $k \times 0,001$ ($k \in \mathbb{N}$) supérieur ou égal à α_n .

Conclusion :

Le résultat affiché est donc une valeur approchée par excès à 0,001 près de α_n .

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.10

1. La ligne 5 simule une loi uniforme sur $[[1, n + b]]$ en effet `rand()` simule une loi uniforme sur $]0, 1[$ donc `rand() * (n+b)` simule une loi uniforme sur $]0, n + b[$ donc `1+ rand() * (n+b)` simule une loi uniforme sur $]1, 1 + n + b[$ en prenant la partie entière `floor(1+ rand() * (n+b))` simule une loi uniforme sur $[[1, n + b]]$. Donc r est un nombre entier choisi au hasard dans $[[1, n + b]]$, on peut considérer que les boules N ont les numéros 1 à n et que les blanches ont les numéros $n + 1$ à $n + b$. Ainsi n est le nombre de N et b le nombre de B . Donc si $r > n$, on a sorti une B et n est augmenté de 1 sinon on a sorti une N et b augmenté de 1.

n et b sont initialisés à 1 ensuite on simule k tirages et on retourne b le nombre de B à la fin des ces tirages. On a donc simulé X_k .

2. On calcule les effectifs de chaque valeur entre 1 et $k + 1$ puis on divise par N pour avoir les fréquences. On passe sur le tiret du nom de la fonction qui n'est correct syntaxiquement.

```
function LE = loiExpo(k,N)
    LE=zeros(1,k+1)
    for j=1:N
        valeur = mystere(k)
        LE(valeur) = LE(valeur) + 1
    end
    LE = LE/N
endfunction
```

3. La matrice M est telle que $(M)_{k,i} = \mathbb{P}(X_k = i)$, on retourne donc la n^e ligne de M .

```
function LT=loiTheo(n)
    M=zeros(n,n+1)
    M(1,1)=1/2
    M(1,2)=1/2
    for k=1:n-1
        M(k+1,1)=M(k,1)/(k+2)
        for i=2:k+1
            M(k+1,i)=(i/(k+2))*M(k,i)+(3+k-i)/(k+2)*M(k,i-1)
        end
        M(k+1,k+2)=M(k+1,1)
    end
    LT=M(n,:)
endfunction
```

4. En théorie $\mathbb{P}(X_5 = 1) = \mathbb{P}(X_5 = 6) = \frac{1}{6!}$ or dans le tableau les valeurs sont distinctes donc c'est un calcul expérimental avec `LoiExp`.

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 1.11

1. En ligne 6 : $Nb = Nb - 1$ (une boule blanche a été tirée)
 En ligne 7 : $u = \text{grand}(1, 1, 'uin', 1, Nb+1)$ (il reste $Nb + 1$ boule dans l'urne)
 En ligne 8 : $X = X + 1$ (un tirage de plus)
2. En ligne 4 : $Y = 0$
 En ligne 8 : $Y = 1$
 On peut répondre aux deux question a: et b: avec le script complet suivant :

```

1  n=input('entrez une valeur pour n : ')
2  nB=n-1
3  X=1
4  Y=0
5  u=grand(1,1,'uin',1,nB+1)
6  while u<nB+1
7      nB=nB-1
8      if u==1 then Y=1
9      end
10     u=grand(1,1,'uin',1,nB+1)
11     X=X+1
12 end
13 disp(X,'la boule noire est apparue au tirage numéro : ')
14 disp(Y,'la valeur de Y est : ')

```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 3.1

1. La fonction `simulX(n)` simule une répétition de n expériences identiques et indépendantes qui a deux issues et elle compte le nombre de succès, où un succès est de probabilité $1/2$. Par conséquent, `simulX(n)` simule une loi binomiale de paramètres $(n, 1/2)$.
2. Le programme est le suivant :

```

function s=simulS(lambda,n)
    N=grand(1,1,"poi",lambda)
    s=0;
    if N>=1 then
        for k=1:N
            s=s+simulX(n)
        end
    end
endfunction

```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 3.2

1. La ligne 5 simule une loi uniforme sur $[[1, n + b]]$ en effet `rand()` simule une loi uniforme sur $]0, 1[$ donc `rand() * (n+b)` simule une loi uniforme sur $]0, n + b[$ donc `1+ rand() * (n+b)` simule une loi uniforme sur $]1, 1 + n + b[$ en prenant la partie entière `floor(1+ rand() * (n+b))` simule une loi uniforme sur $[[1, n + b]]$. Donc r est un nombre entier choisi au hasard dans $[[1, n + b]]$, on peut considérer que les boules N ont les numéros 1 à n et que les blanches ont les numéros $n + 1$ à $n + b$. Ainsi n est le nombre de N et b le nombre de B . Donc si $r > n$, on a sorti une B et n est augmenté de 1 sinon on a sorti une N et b augmenté de 1 .

n et b sont initialisés à 1 ensuite on simule k tirages et on retourne b le nombre de B à la fin des ces tirages. On a donc simulé X_k .

2. On calcule les effectifs de chaque valeur entre 1 et $k+1$ puis on divise par N pour avoir les fréquences. On passe sur le tiret du nom de la fonction qui n'est correct syntaxiquement.

```
function LE = loiExpo(k,N)
    LE=zeros(1,k+1)
    for j=1:N
        valeur = mystere(k)
        LE(valeur) = LE(valeur) + 1
    end
    LE = LE/N
endfunction
```

3. La matrice M est telle que $(M)_{k,i} = \mathbb{P}(X_k = i)$, on retourne donc la n^e ligne de M .

```
function LT=loiTheo(n)
    M=zeros(n,n+1)
    M(1,1)=1/2
    M(1,2)=1/2
    for k=1:n-1
        M(k+1,1)=M(k,1)/(k+2)
        for i=2:k+1
            M(k+1,i)=(i/(k+2))*M(k,i)+(3+k-i)/(k+2)*M(k,i-1)
        end
        M(k+1,k+2)=M(k+1,1)
    end
    LT=M(n,:)
endfunction
```

4. En théorie $\mathbb{P}(X_5 = 1) = \mathbb{P}(X_5 = 6) = \frac{1}{6!}$ or dans le tableau les valeurs sont distinctes donc c'est un calcul expérimental avec LoiExp.

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 3.3

1. Compléter le script Scilab suivant pour qu'il permette le calcul et l'affichage de la valeur prise par la variable aléatoire X lors de l'expérience réalisée dans cet exercice.

```
piece = grand(1,1,'uin',0,2) // c'est le choix de la pièce utilisée
x=1 //au début X vaut 1
if piece==0 // on lancera la pièce 0 et alors P("pile")=P("face")=1/2
then lancer=grand(1,1,'uin',0,1) //0 pour "face" et 1 pour "pile"
    while lancer==0
        lancer=grand(1,1,'uin',0,1) //on relance la pièce
        x=x+1 // on incrémente le compteur des lancers
    end
else
    if piece==1 then x=0 // on lance la pièce 1 qui ne donne jamais "pile"
    end
end
disp(x)
```

2. Le cas où l'on joue avec la pièce numérotée 2 n'est pas pris en compte car si pièce ne vaut ni 0, ni 1, c'est qu'il vaut 2 et alors on a "pile" du premier coup et X reste égal à 1, ce qui est bien la valeur correcte dans ce cas .

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 3.4

1. Complétons les lignes 7 et 10 du script Scilab suivant pour qu'il affiche la valeur prise par S_k lorsque k et p sont entrés par l'utilisateur :

```
k = input('donnez une valeur pour k :')
p = input('donnez une valeur pour p :')
n = 0
c = 0
while c < k
    n = n + 1
    if - - - then c = c + 1
end
end
disp(- - -)
```

Dans ce script, n correspond au nombre de lancers effectués et c au nombre de piles obtenus au cours des n premiers lancers. Dans ce cas, pour que l'on passe de c à $c + 1$ dans la boucle "while", il faut et il suffit que le résultat du n -ème lancer soit un pile, ce qui peut se simuler par l'instruction " $\text{rand} < p$ " (vu que la pièce a une probabilité p de donner pile à chaque fois). Quant à la valeur de S_k , elle correspond au nombre de fois que la boucle "while" aura tourné, c'est-à-dire à n . Dès lors, on pourra compléter le script ci-dessus comme suit :

```
k = input('donnez une valeur pour k :')
p = input('donnez une valeur pour p :')
n = 0
c = 0
while c < k
    n = n + 1
    if rand < p then c = c + 1
end
end
disp(n)
```

2. On souhaite que le script précédent affiche la valeur prise par T_k . Il s'agit alors de remplacer la ligne 7 du script précédent par la suivante, dûment complétée :

if - - - then c = c+1, else - - -

Dans ce nouveau script, l'entier n correspond toujours au nombre de lancers effectués. Par contre, l'entier c est égal au nombre de piles de la dernière série de piles obtenus si le n -ème lancer donne un pile, et à 0 sinon. Dans ce cas, pour que l'on passe de c à $c+1$ dans la boucle "while", il faut et il suffit que le résultat du n -ème lancer soit un pile, ce qui peut se simuler par l'instruction " $\text{rand} < p$ " (vu que la pièce a une probabilité p de donner pile à chaque fois). En revanche, si le n -ème lancer donne un face, alors c est réinitialisé à 0, ce qui signifie que les n premiers lancers ne se terminent pas par une série de piles. Dans ces conditions, la boucle "while" s'interrompra si $c = k$, c'est-à-dire si les n lancers se terminent par une série de k piles consécutifs. Quant à la valeur de T_k , elle correspondra au nombre de fois que la boucle "while" aura tourné, c'est-à-dire à n . Dès lors, pour obtenir une simulation de T_k , il suffit de compléter la ligne 7 ci-dessus comme suit :

if rand<p then c = c+1, else c=0

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 3.5

1. En ligne 6 : $Nb = Nb - 1$ (une boule blanche a été tirée)
 En ligne 7 : $u = \text{grand}(1, 1, 'uin', 1, Nb+1)$ (il reste $Nb + 1$ boule dans l'urne)
 En ligne 8 : $X = X + 1$ (un tirage de plus)

2. En ligne 4 : $Y = 0$
 En ligne 8 : $Y = 1$

On peut répondre aux deux question a: et b: avec le script complet suivant :

```

1  n=input('entrez une valeur pour n : ')
2  nB=n-1
3  X=1
4  Y=0
5  u=grand(1,1,'uin',1,nB+1)
6  while u<nB+1
7      nB=nB-1
8      if u==1 then Y=1
9      end
10     u=grand(1,1,'uin',1,nB+1)
11     X=X+1
12 end
13 disp(X,'la boule noire est apparue au tirage numéro : ')
14 disp(Y,'la valeur de Y est : ')

```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 6.1

1. $Y = \text{sqrt}(\text{grand}(1, 1, 'exp', 2))$.
2. On a $U = 1 - e^{-X/2} \iff e^{-X/2} = 1 - U \iff X = -2\ln(1 - U)$. Cette dernière expression a du sens car U est presque sûrement à valeurs dans $[0, 1[$. On propose $X = -2 * \log(1 - \text{rand}())$.

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 6.2

Remarquons que $X = \sqrt{2aY}$ où Y suit une loi exponentielle de paramètre 1 donc un script Scilab demandant la valeur de a à l'utilisateur et permettant de simuler la variable aléatoire X est :

```
a=input('entrer la valeur de a')
Y=grand(1,1,'exp',1)
X=sqrt(2*a*Y)
disp(X)
```

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 6.3

Sans problème : `Y=grand(1,1,'exp',theta);X=exp(Y)`

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 7.1

1. Par observation du graphique représentant les lignes de niveaux de f (sur $[0.4, 1.6] \times [0.5, 1.5]$), il semble que f présente un minimum local en $(1, 1)$ dont la valeur approximative serait de 3.01.
2. (a) On propose le programme ci-dessous

```
function y=h(n,x)
    y=x^n+1+x^(-n) //une variante s'obtient avec 1/(x^n)
```

- (b) On complète le programme de dichotomie. On sait que v_n est entre 1 et 3 et que $h_n(v_n) = 4$. Tant que la taille de l'intervalle de recherche est supérieure à la précision voulue (ici 10^{-5}), on continue le processus. Si l'évaluation au milieu de l'intervalle de recherche est strictement inférieure à 4, alors on décale l'intervalle de recherche à droite (h_n est croissante et dans ce cas on est pas encore passé par 4), sinon on est allé trop loin et il faut décaler à gauche. Il vient

```
function res=v(n)
    a = 1
    b = 3
    while (b-a)>10^(-5)
        c=(a+b)/2
        if h(n,c)< 4 then a=c
            else b=c
        end
    end
    res=c
endfunction
```

- (c) Le code proposé en complément permet de représenter les termes $v_1, v_2^2, v_3^3, \dots, v_{20}^{20}$ (soit les vingts premiers termes de la suite (v_n^n)). On peut conjecturer que cette suite est constante

$$\forall n \in \mathbb{N}^*, \quad v_n^n \simeq 2,6.$$

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 7.2

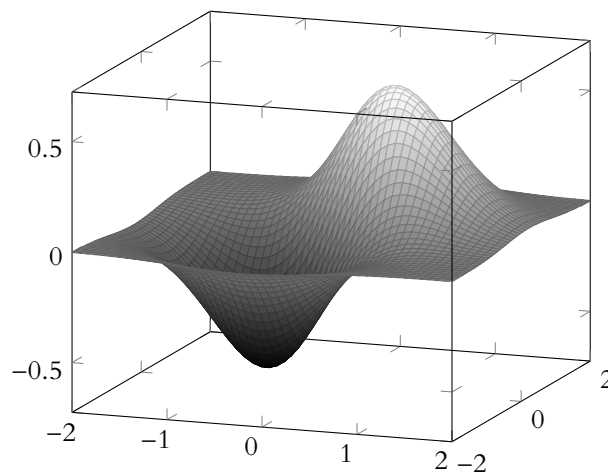
1. Ecrivons des commandes **Scilab** permettant de calculer et d'afficher $H_n(a)$ pour une valeur de n entrée par l'utilisateur. D'après les questions précédentes, on sait que :

$$H_n(a) = \frac{-2}{\sqrt{2ne}}(nI_n + J_n).$$

Dès lors, on pourra utiliser le script suivant :

```
n = input('entrez un entier n :');
A = (-2/sqrt(2 * n * exp(1))) * (ones(n, n) + n * eye(n));
disp(A)
```

2. Dans le cas $n = 2$, la nappe suivante est acceptable en tant que représentation graphique de la fonction f_2 . En effet, comme cela a été montré aux questions précédentes, le graphe de f_2 (comme c'est le cas pour cette nappe) présente un maximum global et un minimum global, lesquels sont atteints en des points opposés par rapport à l'origine de \mathbb{R}^2 .



Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 8.1

1. Par observation du graphique représentant les lignes de niveaux de f (sur $[0.4, 1.6] \times [0.5, 1.5]$), il semble que f présente un minimum local en $(1, 1)$ dont la valeur approximative serait de 3.01.
2. (a) On propose le programme ci-dessous

```
function y=h(n,x)
y=x^n+1+x^(-n) //une variante s'obtient avec 1/(x^n)
```


- (b) On complète le programme de dichotomie. On sait que v_n est entre 1 et 3 et que $h_n(v_n) = 4$. Tant que la taille de l'intervalle de recherche est supérieure à la précision voulue (ici 10^{-5}), on continue le processus. Si l'évaluation au milieu de l'intervalle de recherche est strictement inférieure à 4, alors on décale l'intervalle de recherche à droite (h_n est croissante et dans ce cas on est pas encore passé par 4), sinon on est allé trop loin et il faut décaler à gauche. Il vient

```
function res=v(n)
    a = 1
    b = 3
    while (b-a)>10^(-5)
        c=(a+b)/2
        if h(n,c)< 4 then a=c
            else b=c
        end
    end
    res=c
endfunction
```

- (c) Le code proposé en complément permet de représenter les termes $v_1, v_2^2, v_3^3, \dots, v_{20}^{20}$ (soit les vingts premiers termes de la suite (v_n^n)). On peut conjecturer que cette suite est constante

$$\forall n \in \mathbb{N}^*, \quad v_n^n \simeq 2,6.$$

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 8.2

1. $f(1) = \int_0^1 \ln(1+t^2) dt = E[\ln(1+U^2)]$ où U suit une loi uniforme sur $[0;1]$ d'après le théorème de transfert. On peut donc approcher la valeur de $f(1)$ en considérant la moyenne de la réalisation d'un échantillon de variables aléatoires suivant la même loi que $\ln(1+U^2)$.

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

2. Cette fois-ci la fonction intégrande n'est pas f mais $t \mapsto \frac{1}{1-f(t)}$ d'où le script suivant :

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

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 9.1

1. (a) On peut compléter le programme comme suit :

```
function y=f(x,n)
    y=sum(x.^(1:n))
endfunction
```

- (b) Pour $x \neq 1$, on a : $f_n(x) = \frac{1-x^{n+1}}{1-x} - 1$ et pour $x = 1$, $f_n(x) = n$. Le programme peut alors s'écrire :

```
function y=f(x,n)
    if x==1 then
        y=n
    else
        y=(1-x^(n+1))/(1-x)-1
    end
endfunction
```

2. Fixons $n \in \mathbb{N}^*$. On sait déjà que f_n est croissante sur $[0, 1]$ et $f_n(0) = 0$. Le programme va alors tester toutes les valeurs de $f_n(x)$ à partir de $x = 0$ et avec un pas de 0,001 tant que $f_n(x) < 1$. La valeur affichée par le programme est alors le premier de ces nombres pour lequel $f_n(x) \geq 1$. Le résultat affiché est donc le plus petit nombre du type $k \times 0,001$ ($k \in \mathbb{N}$) supérieur ou égal à α_n .

Conclusion :

Le résultat affiché est donc une valeur approchée par excès à 0,001 près de α_n .

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 9.2

Les valeurs propres de f sont égales à celles de A .

Pour λ réel, on note $E_\lambda = \ker(f - \lambda id)$.

Avec le théorème du rang, $\dim E_\lambda = 3 - \text{rg}(A - \lambda I_3)$. On peut donc conjecturer que 1 et 2 sont effectivement valeurs propres de f et que $\dim E_1 = 2$ et $\dim E_2 = 1$.

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).

CORRECTION 9.3

- On peut compléter les lignes 6, 7 et 8 par : `aux = A(j) ; A(j) = A(p) ; A(p) = aux.`
- La boucle `for` programmée a pour effet d'attribuer à `m` la plus grande valeur apparaissant dans `A`. Comme `A` contient les entiers 1 à `n`, `m` vaut bien `n`.
 - `c` est la "position" du maximum, indice de `A` en lequel figure la valeur `n`.
 - `c = find(A==n)`

Merci de m'aider à améliorer ce corrigé en me signalant toute anomalie ou tout passage peu compréhensible : [en cliquant ici](#).