TP récapitulatif : Annales 2015 / 2016 / 2017 / 2018

I. Fonctions de deux variables

EDHEC – 2017

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

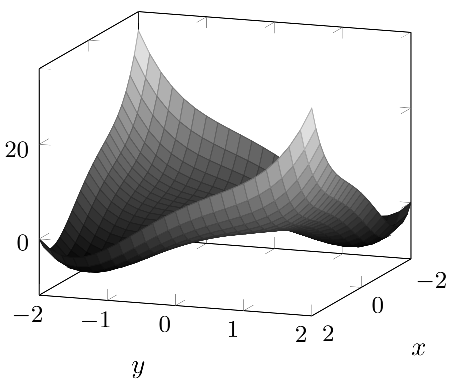
$$f(x, y) = x^4 + y^4 - 2\,(x - y)^2$$

- On montre que f admet des minima globaux en $(\sqrt{2}, -\sqrt{2})$ et $(-\sqrt{2}, \sqrt{2})$.

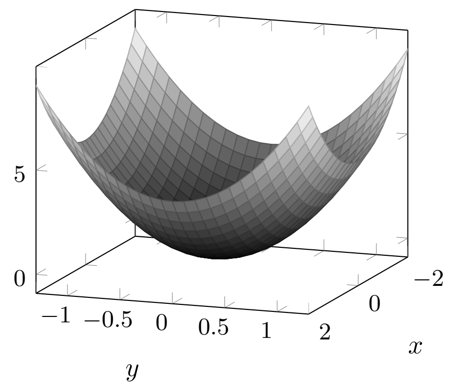
a) Compléter la deuxième ligne du script suivant afin de définir la fonction f.

```
1  function z = f(x,y)
2      z = ---
3  endfunction
4  x = linspace(-2,2,101)
5  y = x
6  fplotd3d(x,y,f)
```

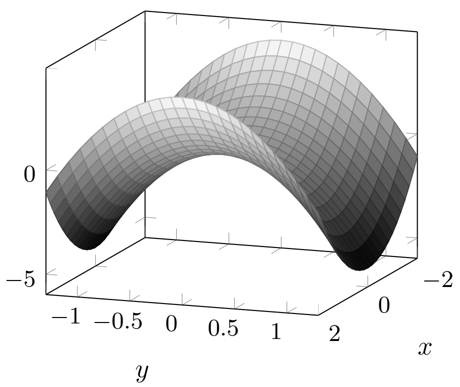
b) Le script précédent, une fois complété, renvoie l'une des trois nappes suivantes. Laquelle ? Justifier la réponse.



Nappe 1



Nappe 2



Nappe 3

II. Autour des matrices

EDHEC – 2017

- On note E l'espace vectoriel des fonctions polynomiales de degré inférieur ou égal à 2 et on rappelle que la famille (e_0, e_1, e_2) est une base de E, les fonctions e_0, e_1 e_2 étant définies par :

$$\forall t \in \mathbb{R} \quad e_0(t) = 1, \quad e_1(t) = t, \quad e_2(t) = t^2$$

On considère l'application φ qui, à toute fonction P de E, associe la fonction, notée $\varphi(P)$, définie par :

$$\forall x \in \mathbb{R}, \ (\varphi(P))\,(x) = \int_0^1 P(x+t)\ dt$$

- On montre que :

$$A = \mathrm{Mat}_{(e_0,e_1,e_2)}(\varphi) = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

a) Compléter les commandes `Scilab` suivantes pour que soit affichée la matrice A^n pour une valeur de n entrée par l'utilisateur :

```
n = input('entrez une valeur pour n : ')
A = [---]
disp(---)
```

HEC – 2017

- Pour tout $n \in \mathbb{N}^*$, on note $\mathscr{M}_n(\mathbb{R})$ l'ensemble des matrices carrés à n lignes et n colonnes à coefficients réels et B_n l'ensemble des matrices de $\mathscr{M}_n(\mathbb{R})$ dont tous les coefficients sont égaux à 0 ou à 1.

- *Exemple 2.* Soit B la matrice de B_3 définie par : $B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

On considère les instructions et la sortie (-→) **Scilab** suivantes :

```
B = [0,1,0;1,0,0;0,0,1]
P = [1,1,0;1,-1,0;0,0,1]
inv(P) * B * P
```

```
-->
   1.     0.     0.
   0.    -1.     0.
   0.     0.     1.
```

a) Déduire les valeurs propres de B de la séquence **Scilab** précédente.

b) Déterminer une base de chacun des sous-espaces propres de B.

HEC – 2018

À la fin de l'exercice 1 de l'épreuve HEC 18, on trouvait une question **Scilab** qui consistait à écrire une matrice $n \times n$ dont les coefficients $q(\ell, k)$ étaient donnés par la valeur d'une fonction $q : \mathbb{N}^2 \to \mathbb{N}$ étudiée dans le sujet. Plus pécisément, les questions précédant le **Scilab** permettait d'affirmer :

× $\forall k \in \mathbb{N}^*$, $q(1, k) = 1$.

× $\forall \ell \geqslant k$, $q(\ell, k) = p(k) = q(k, k)$.

× $\forall k < \ell$, $q(\ell, k) = q(k, k)$.

× $\forall \ell \in [\![1, n]\!]$, $q(\ell, \ell) = 1 + q(\ell - 1, \ell)$.

× $\forall k > \ell$, $q(\ell, k) = q(\ell - 1, k) + q(\ell, k - \ell)$.

- La fonction **Scilab** suivante dont le script est incomplet (lignes 5 et 6), calcule une matrice qmatrix(n) telle que pour chaque couple $(\ell, k) \in [\![1, n]\!]^2$, le coefficient situé à l'intersection de la ligne ℓ et de la colonne k est égal à $q(\ell, k)$.

```
1  function q = qmatrix(n)
2      q = ones(n, n)
3      for L = 2:n
4          for K = 2:n
5              if (K<L) then
6                  q(L,K) = ................
7              elseif (K==L) then
8                  q(L,K) = ..............
9              else
10                 q(L,K) = q(L-1,K) + q(L,K-L)
11             end
12         end
13     end
14 endfunction
```

L'application de la fonction qmatrix à l'entier $n = 9$ fournit la sortie suivante :

```
--> qmatrix(9)
    1.   1.   1.   1.   1.   1.   1.   1.   1.
    1.   2.   2.   3.   3.   4.   4.   5.   5.
    1.   2.   3.   4.   5.   7.   8.   10.  12.
    1.   2.   3.   5.   6.   9.   11.  15.  18.
    1.   2.   3.   5.   7.   10.  13.  18.  23.
    1.   2.   3.   5.   7.   11.  14.  20.  26.
    1.   2.   3.   5.   7.   11.  15.  21.  28.
    1.   2.   3.   5.   7.   11.  15.  22.  29.
    1.   2.   3.   5.   7.   11.  15.  22.  30.
```

a) Compléter les lignes 5 et 6 du script de la fonction qmatrix.

b) Donner un script **Scilab** permettant de calculer $p(n) = q(n, n)$ à partir d'une valeur de n entrée au clavier.

III. Méthodes itératives (suites récurrentes et calcul de sommes)

III.1. Suites de type $u_{n+1} = f(u_n)$

ECRICOME – 2015

- On considère la fonction F définie par :

$$F(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1 - e^{-x} & \text{si } x \geqslant 0 \end{cases}$$

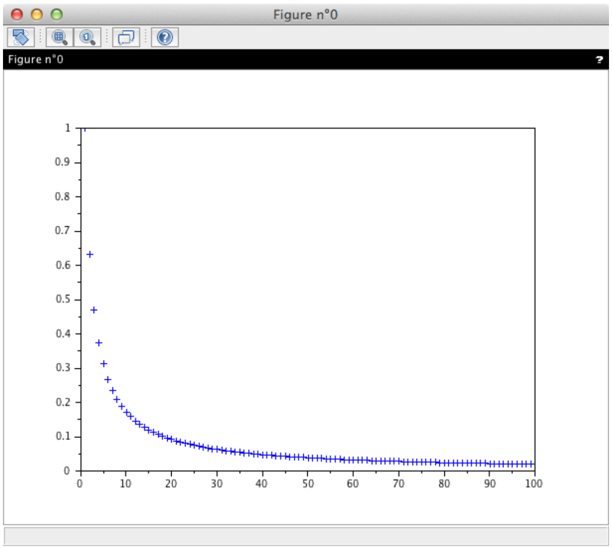
(fonction de répartition de d'une v.a.r. X telle que $X \hookrightarrow \mathcal{E}(1)$)
et on définit la suite $(u_n)_{n\in\mathbb{N}^*}$ définie par :

$$\begin{cases} u_1 = 1 \\ \forall n \in \mathbb{N}^*, \; u_{n+1} = F(u_n) \end{cases}$$

a) Recopier et compléter le programme **Scilab** suivant qui permet de représenter les cent premiers termes de la suite $(u_n)_{n\in\mathbb{N}^*}$:

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

b) Le programme complété permet d'obtenir la représentation graphique suivante.

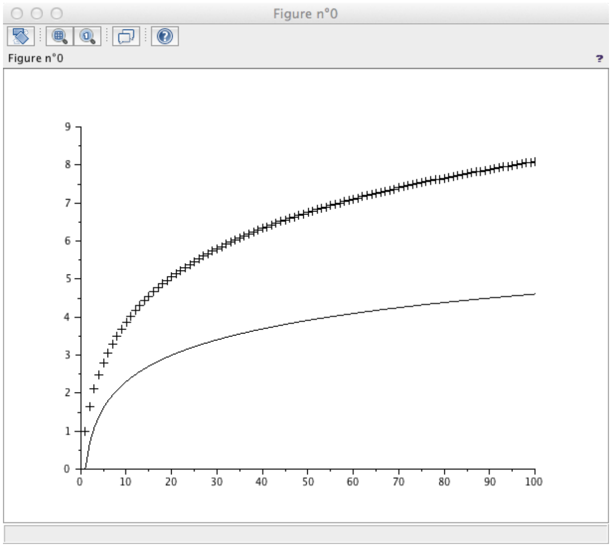


Quelle conjecture pouvez-vous émettre sur la monotonie et la limite de la suite $(u_n)_{n\in\mathbb{N}^*}$?

c) On modifie le programme écrit en question ***a)*** en remplaçant la dernière ligne par :

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

`ECRICOME` – 2018

- On considère les matrices : $A = \begin{pmatrix} 2 & 1 & -2 \\ 0 & 3 & 0 \\ 1 & -1 & 5 \end{pmatrix}$ et $B = \begin{pmatrix} 1 & -1 & -1 \\ -3 & 3 & -3 \\ -1 & 1 & 1 \end{pmatrix}$.

On pose $X_0 = \begin{pmatrix} 3 \\ 0 \\ -1 \end{pmatrix}$, $X_1 = \begin{pmatrix} 3 \\ 0 \\ -2 \end{pmatrix}$, et pour tout $n \in \mathbb{N}$: $X_{n+2} = \dfrac{1}{6}\ A\ X_{n+1} + \dfrac{1}{6}\ B\ X_n$.

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

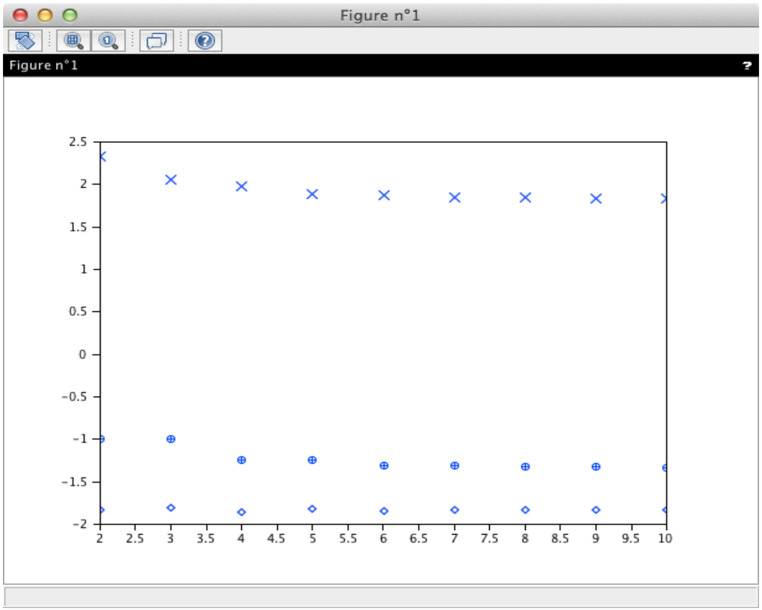
```
function res = X(n)
    Xold = [3; 0; -1]
    Xnew = [3; 0; -2]
    A = [2,1,-2; 0,3,0; 1,-1,5]
    B = [1,-1,-1; -3,3,-3; -1,1,1]
    for i = 2:n
        Aux = .........
        Xold = ........
        Xnew = ........
    end
    res = ........
endfunction
```

- Dans l'exercice, il était noté $X_n = \begin{pmatrix} \alpha_n \\ \beta_n \\ \gamma_n \end{pmatrix}$ et on démontrait, pour tout $n \in \mathbb{N}$:

$$\alpha_n = \frac{11}{6} + \frac{2}{3}\left(-\frac{1}{2}\right)^n + \left(\frac{1}{2}\right)^{n-1} - \frac{3}{2}\left(-\frac{1}{3}\right)^n$$

$$\beta_n = \left(\frac{1}{2}\right)^{n-1} - \frac{2}{3}\left(-\frac{1}{2}\right)^n - \frac{4}{3} \quad \text{et} \quad \gamma_n = -\frac{11}{6} - \frac{2}{3}\left(-\frac{1}{2}\right)^n + \frac{3}{2}\left(-\frac{1}{3}\right)^n$$

b) 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.



III.2. Calcul du premier entier n qui vérifie une condition donnée / valeur approchée d'une suite

`EDHEC – 2016`

- On considère une suite $(u_n)_{n\in\mathbb{N}}$ qui vérifie :

$$\forall n \in \mathbb{N},\ \mathrm{e}^{-\sqrt{u_n}} \leqslant u_n - n \leqslant \mathrm{e}^{-\sqrt{n}}$$

a) 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)
```

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

EML – 2015

- On considère l'application $f : \mathbb{R} \to \mathbb{R},\ x \mapsto f(x) = x^3 \mathrm{e}^x$.

a) Montrer que la série $\sum\limits_{n \geqslant 1} \dfrac{1}{f(n)}$ converge. On note $S = \sum\limits_{n=1}^{+\infty} \dfrac{1}{f(n)}$.

On démontre alors que : $\forall n \in \mathbb{N}^*,\ \left| S - \sum\limits_{k=1}^{n} \dfrac{1}{f(k)} \right| \leqslant \dfrac{1}{(\mathrm{e}-1)\ \mathrm{e}^n}$.

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

EML – 2016

- On considère la fonction f définie par : $f(x) = \begin{cases} x^2 - x\ln(x) & \text{si } x \neq 0 \\ 0 & \text{si } x = 0 \end{cases}$

et on définit la suite $(u_n)_{n \in \mathbb{N}^*}$ par : $\begin{cases} u_0 = \frac{1}{2} \\ \forall n \in \mathbb{N},\ u_{n+1} = f(u_n) \end{cases}$

(on démontre que (u_n) est croissante et qu'elle converge vers 1)

a) Écrire un programme en **Scilab** qui calcule et affiche un entier naturel N tel que :

$$1 - u_N < 10^{-4}$$

EML – 2017

- On considère la fonction $f :]0, +\infty[\to \mathbb{R}$ définie par $f : x \mapsto \mathrm{e}^x - \mathrm{e}\ln(x)$.
- On considère la suite réelle $(u_n)_{n \in \mathbb{N}}$ définie par : $\begin{cases} u_0 = 2 \\ \forall n \in \mathbb{N},\ u_{n+1} = f(u_n) \end{cases}$

a) Écrire un programme en **Scilab** qui, étant donné un réel A, renvoie un entier naturel N tel que $u_N \geqslant A$.

EML – 2018

- On pose : $u_0 = 4$ et $\forall n \in \mathbb{N},\ u_{n+1} = \ln(u_n) + 2$.

On devait démontrer les propriétés suivantes.

- $\forall n \in \mathbb{N},\ u_n \geqslant b$.
- $\forall n \in \mathbb{N},\ u_{n+1} - b \leqslant \dfrac{1}{2}\ (u_n - b)$.
- $\forall n \in \mathbb{N},\ 0 \leqslant u_n - b \leqslant \dfrac{1}{2^{n-1}}$.

a) Écrire une fonction **Scilab** d'en-tête `function u = suite(n)` qui, prenant en argument un entier n de $\mathbb{N}$, renvoie la valeur de u_n.

b) Recopier et compléter la ligne 3 de la fonction **Scilab** suivante afin que, prenant en argument un réel `epsilon` strictement positif, elle renvoie une valeur approchée de b à `epsilon` près.

```
1 function b = valeur_approchee(epsilon)
2     n = 0
3     while ..........
4         n = n + 1
5     end
6     b = suite(n)
7 endfunction
```

`ECRICOME – 2018`

- Pour tout entier naturel n non nul, on pose : $u_n = \sum\limits_{k=1}^{n} \dfrac{1}{k} - \ln(n)$.
- On démontrait que la suite (u_n) était convergente, vers une limite notée $\gamma \in \mathbb{R}$ puis :

$$\forall n \in \mathbb{N}^*, \ |u_n - \gamma| \leqslant \frac{1}{n}$$

a) 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.

b) On rappelle que l'instruction `floor(x)` renvoie la partie entière d'un réel x et on suppose que la fonction `u` 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))
```

IV. Autour des v.a.r.

IV.1. Simulation d'expériences aléatoires et de variables aléatoires

IV.1.a) Simulation de variables aléatoires

`EML – 2015`

- On considère une v.a.r. U tel que : $U \hookrightarrow \mathcal{U}([0,1[)$.
- On démontre que la v.a.r. $V = -\dfrac{1}{\lambda} \ln(1-U)$ est telle que $V \hookrightarrow \mathcal{E}(\lambda)$.

a) Écrire une fonction en **Scilab** qui, étant donné un réel $\lambda > 0$, simule la loi exponentielle de paramètre λ.

`EDHEC – 2016`

- 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 :

$$\mathbb{P}([Z=1]) = p \qquad \text{et} \qquad \mathbb{P}([Z=-1]) = 1-p$$

Enfin, on note X la v.a.r. définie par :

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

a) Vérifier que l'on a :

$$X = U \, \frac{1+Z}{2} + V \, \frac{1-Z}{2}$$

b) Soit T une variable aléatoire suivant la loi de Bernoulli de paramètre p.
Déterminer la loi de $2T-1$.

c) On rappelle que `grand(1,1,'unf',a,b)` et `grand(1,1,'bin',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.

EDHEC – 2016

- 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 $S_n = \sum\limits_{k=1}^{n} X_k$ pour tout $n \in \mathbb{N}^*$.

a) 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 supérieure à 1 : ')
S = ---
disp(S)
```

EDHEC – 2017

- Soit V une variable aléatoire suivant la loi exponentielle de paramètre 1, dont la fonction de répartition est la fonction F_V définie par : $F_V(x) = \begin{cases} 0 & \text{si } x \leqslant 0 \\ 1 - \mathrm{e}^{-x} & \text{si } x > 0 \end{cases}$.
On pose $W = -\ln(V)$ et on admet que W est aussi une variable aléatoire dont le fonction de répartition est notée F_W. On dit que W suit une loi de Gumbel.
- 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 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.

a) 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
```

b) Voici deux scripts :

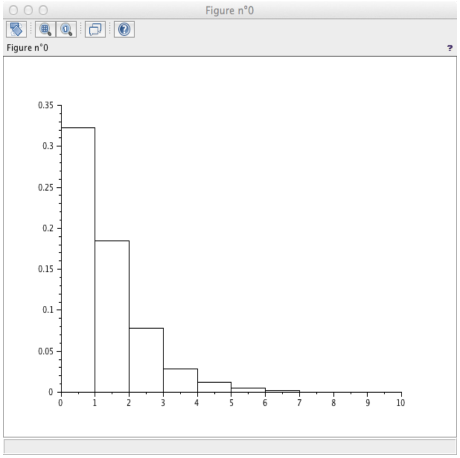
```
V = grand(1,10000,'exp',1)
W = -log(V)
s = linspace(0,10,11)
histplot(s,W)
```

Script (1)

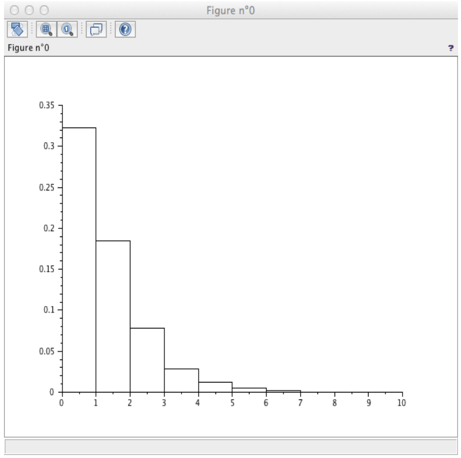
```
n = input('entrez la valeur de n : ')
Z = [] // la matrice-ligne Z est vide
for k = 1 :10000
    Z = [Z,f(n)]
end
s = linspace(0,10,11)
histplot(s,Z)
```

Script (2)

Chacun des scripts simule 10000 variables indépendantes, regroupe les valeurs renvoyées en 10 classes qui sont les intervalles $[0,1]$, $]1,2]$, $]2,3]$, ..., $]9,10]$ 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 Gumbel (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 v.a.r. (Z_n) ?

HEC – 2017

- On note :
  - × a et b deux réels strictement positifs ;
  - × $(\Omega, \mathcal{A}, \mathbb{P})$ un espace probabilisé sur lequel sont définies toutes les variables aléatoires du problème ;
  - × $G_{a,b}$ la fonction définie sur $\mathbb{R}_+$ par : $G_{a,b}(x) = \exp\left(-ax - \dfrac{b}{2}x^2\right)$.
- Pour tout $a > 0$ et pour tout $b > 0$, on pose : $f_{a,b}(x) = \begin{cases} (a+bx)\ \exp\left(-ax - \dfrac{b}{2}x^2\right) & \text{si } x \geqslant 0 \\ 0 & \text{si } x < 0 \end{cases}$.

  On dit qu'une variable aléatoire suit la loi exponentielle linéaire de paramètres a et b, notée $\mathcal{E}_\ell(a,b)$, si elle admet $f_{a,b}$ pour densité.
- Soit Y une variable aléatoire suivant la loi exponentielle de paramètre 1.

  On pose : $X = \dfrac{-a + \sqrt{a^2 + 2bY}}{b}$. On montre alors que X suit la loi $\mathcal{E}_\ell(a,b)$.
- On montre ensuite que, si U est une variable aléatoire suivant la loi uniforme sur $[0, 1[$, alors $Y = -\ln(1-U)$ suit la loi $\mathcal{E}(1)$.

a) 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
```

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.

c) 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
```

ESSECI – 2017

- 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 suppose que X suit la loi $\mathcal{L}(0, 1)$.
  Montrer que $\beta X + \alpha$ suit la loi $\mathcal{L}(\alpha, \beta)$.
- 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 $\dfrac{1}{2}$ et indépendante de U.
  On montre que $X = (2V - 1)U$ suit la loi $\mathcal{L}(0, 1)$.

a) 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
```

L'énoncé donnait l'aide ***Scilab*** *suivante en fin de sujet :*

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 $\mathcal{U}(\llbracket -1, 3 \rrbracket)$.

EDHEC – 2018

- Soit a un réel strictement positif et f la fonction définie par : $f(x) = \begin{cases} \frac{x}{a}\,\mathrm{e}^{-\frac{x^2}{2a}} & \text{si } x \geqslant 0 \\ 0 & \text{si } x < 0 \end{cases}$.

  1. Montrer que la fonction f est une densité.
  Dans la suite de l'exercice, on considère une variable aléatoire X de densité f.

  2. Déterminer la fonction de répartition F_X de X.

  3. On considère la variable aléatoire Y définie par : $Y = \dfrac{X^2}{2a}$.

  a) Montrer que Y suit la loi exponentielle de paramètre 1.

  b) 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 un script **Scilab** demandant la valeur de a à l'utilisateur et permettant de simuler la v.a.r. X.

ESSEC – I – 2018

On définit deux variables aléatoires Y_n et Z_n de la façon suivante.
Pour tout $\omega \in \Omega$:

- $Y_n(\omega) = \max(X_1(\omega), \ldots, X_n(\omega))$ est le plus grand des réels $X_1(\omega)$, ..., $X_n(\omega)$; on remarque que Y_n est définie également lorsque n vaut 1, de sorte que dans la suite du sujet on pourra considérer Y_{n-1}.
- $Z_n(\omega)$ est le « deuxième plus grand » des nombres $X_1(\omega)$, ..., $X_n(\omega)$, autrement dit, une fois que ces n réels sont ordonnés dans l'ordre croissant, Z_n est l'avant-dernière valeur. On note que lorsque la plus grande valeur est présente plusieurs fois, $Z_n(\omega)$ et $Y_n(\omega)$ sont égaux.

1. On suppose que l'on a défini une fonction **Scilab** d'entête `function x = simulX(n)` qui retourne une simulation d'un échantillon de taille n de la loi de X sous la forme d'un vecteur de longueur n. Compléter la fonction qui suit pour qu'elle retourne le couple $(Y_n(\omega), Z_n(\omega))$ associé à l'échantillon simulé par l'instruction `X = simulX(n)` :

```
function [y, z] = DeuxPlusGrands(n)
  X = simulX(n)
  if ...
    y = X(1) ; z = X(2)
  else
    ...
  end
  for k = 3:n
    if X(k) > y
      z = ... ; y = ...
    else
      if ...
        z = ...
      end
    end
  end
endfunction
```

IV.1.b) Simulation d'une expérience aléatoire et des v.a.r. associées

EDHEC – 2018

- 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 Face à coup sûr et une troisième pièce numérotée 2, donnant Pile 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 et la variable aléatoire Y, égale au rang d'apparition du premier Face. On convient de donner à X la valeur 0 si l'on n'obtient jamais Pile et de donner à Y la valeur 0 si l'on n'obtient jamais Face.

  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.

  a) 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)
```

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

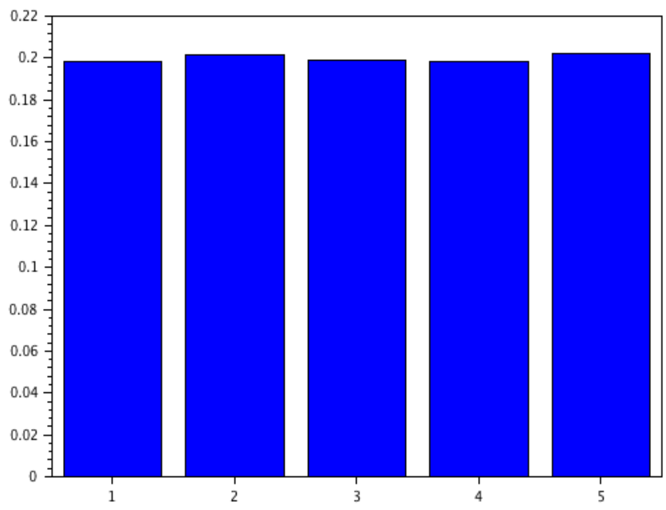
ECRICOME – 2015

- 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.
  On notera pour tout entier naturel i non nul :
  - × N_i l'événement « on tire une boule noire lors du i-ième tirage ».
  - × B_i l'événement « on tire une boule blanche lors du i-ième tirage ».

  a) 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)
```

  b) 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 ?

ECRICOME – 2016

- Dans tout l'exercice, X et Y sont deux variables aléatoires définies sur le même espace probabilisé et à valeurs dans $\mathbb{N}$. On dit que les deux variables X et Y sont **échangeables** si :

$$\forall (i,j) \in \mathbb{N}^2,\ \mathbb{P}([X=i] \cap [Y=j]) = \mathbb{P}([X=j] \cap [Y=i])$$

Résultats préliminaires

a) On suppose que X et Y sont deux variables indépendantes et de même loi.
Montrer que X et Y sont échangeables.

b) On suppose que X et Y sont échangeables. Montrer, à l'aide de la formule des probabilités totales, que :

$$\forall i \in \mathbb{N},\ \mathbb{P}([X=i]) = \mathbb{P}([Y=i])$$

Étude d'un exemple

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 de la boule 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.

c) (i) 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
```

(ii) 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,variante)
    x = tirage(b,n)
    if variante == 1 then
        if x == 1 then
            ...........
        else
            ...........
        end
    elseif variante == 2 then
        ...........
        ...........
        ...........
        ...........
        ...........
    end
    y = tirage(b,n)
endfunction
```

(iii) 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 :

– `loiX` : un tableau unidimensionnel à deux éléments qui estime $[\mathbb{P}([X = 1]), \mathbb{P}([X = 2])]$
– `loiY` : un tableau unidimensionnel à deux éléments qui estime $[\mathbb{P}([Y = 1]), \mathbb{P}([Y = 2])]$
– `loiXY` : un tableau bidimensionnel à deux lignes et deux colonnes qui estime :

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

```
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
```

(iv) On exécute notre fonction précédente avec $b = 1$, $n = 2$, $c = 1$, $N = 10000$ et dans chacune des variantes. On obtient :

```
--> [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
        0.25031     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
```

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 :

$$\begin{array}{c} 0.33 \times 0.33 \simeq 0.11 \\ 0.33 \times 0.41 \simeq 0.14 \\ 0.33 \times 0.58 \simeq 0.19 \\ 0.33 \times 0.66 \simeq 0.22 \\ 0.41 \times 0.66 \simeq 0.27 \\ 0.58 \times 0.66 \simeq 0.38 \\ 0.66 \times 0.66 \simeq 0.44 \end{array}$$

`EML – 2017`

- 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 et 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 $k^{\text{ème}}$ tirage »

  R_k l'événement : « on obtient une boule rouge au $k^{\text{ème}}$ tirage ».

a) 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
        s = rand()
        if ... then
            ...
        else
            ...
        end
    end
endfunction
```

b) 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 ?

IV.2. Illustration de la LfGN / méthode de Monte-Carlo

IV.2.a) Valeur approchée d'une probabilité

`ESSECI – 2015`

- On considère une v.a.r. D (de densité f nulle sur $]-\infty, 0[$) et on note R la fonction définie sur $[0, +\infty[$ par $R(x) = \mathbb{P}([D > x])$.
- On suppose que l'on a défini une fonction d'entête `function r = R(x)` qui renvoie la valeur de `R` au point `x`. On considère X une v.a.r. qui suit la loi exponentielle de paramètre 1.
- On cherche alors à calculer une valeur approchée du réel S défini par :

$$\mathbb{P}\left(\left[R(X) \leqslant \frac{k+c}{v}\right]\right) = \mathrm{e}^{-S}$$

  où k (coût de stockage), c (coût d'achat), et v (prix de vente) sont des constantes.

a) 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));
```

EDHEC – 2015

- 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 v.a.r. .
  On note T le temps total passé par C dans l'agence bancaire.
- On rappelle que, si `a` et `b` sont deux vecteurs lignes de taille n, les commandes `m = min(a,b)` et `M = max(a,b)` renvoient les vecteurs `m` et `M`, de même taille que `a` et `b`, et tels que, pour tout i de $[\![1, n]\!]$, 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 $[0, 1[$.

a) 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 $[T \geqslant V]$?

c) On souhaite déterminer une valeur approchée de la probabilité $\mathbb{P}([T \geqslant V])$, 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 dans le programme précédent, 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 ?

ECRICOME – 2017

- 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 v.a.r. 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 et 9, alors on obtient : $S_1 = 2$, $S_2 = 6$, $S_3 = 7$, $S_4 = 12$, $S_5 = 21$ et $T_{10} = 4$.
- On montre que la suite de v.a.r. $(T_n)_{n\geqslant 1}$ converge en loi vers la v.a.r. Y à valeurs dans $\mathbb{N}^*$ définie par :
$$\forall k \in \mathbb{N}^*,\ \mathbb{P}([Y = k]) = \frac{k-1}{k!}$$

a) On rappelle qu'en langage **Scilab**, l'instruction `grand(1,1,'uin',1,n)` renvoie un entier aléatoire de $[\![1, n]\!]$. 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
```

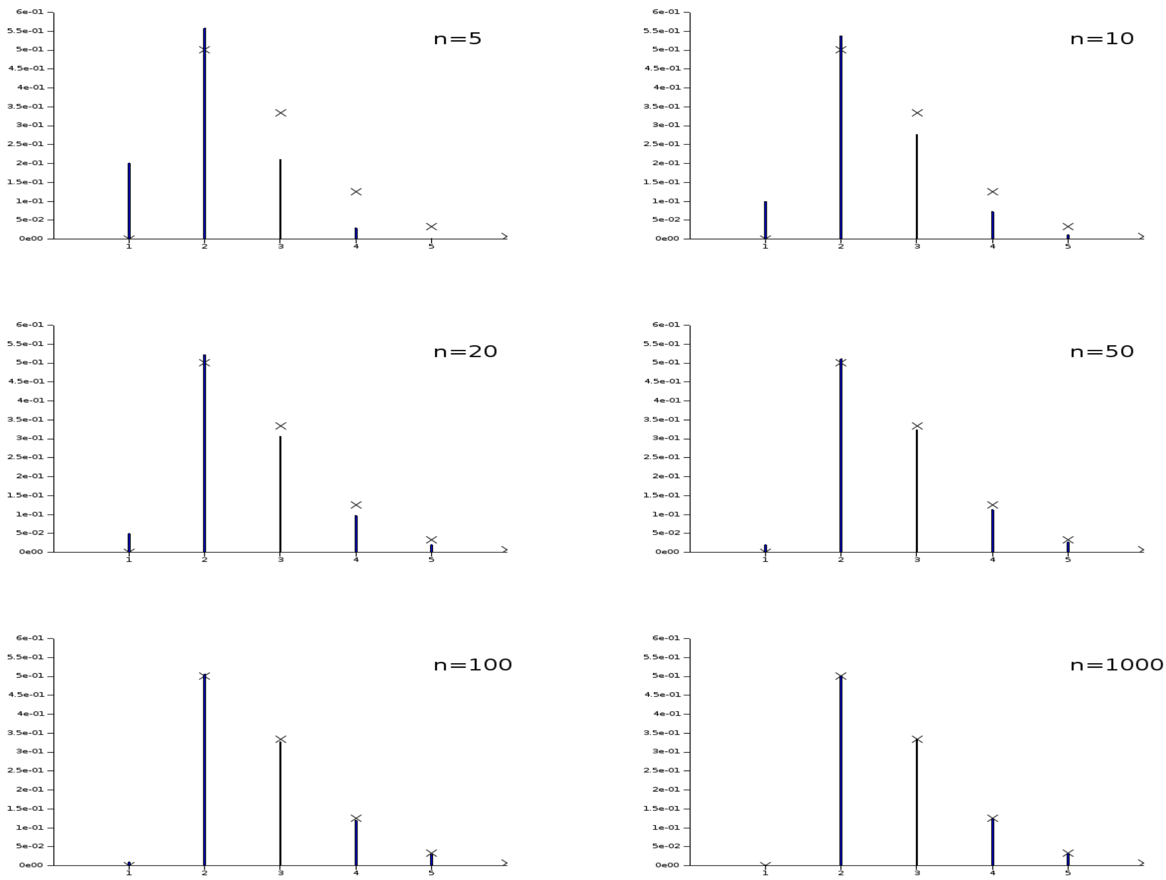
b) On suppose déclarée la fonction précédente et on écrit le script ci-dessous :

```
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 :



c) Expliquer ce que représente les vecteurs renvoyés par les fonctions `freqT` et `loitheoY`. Comment ces vecteurs sont-ils représentés graphiquement dans chaque graphique obtenu ?

d) Expliquer en quoi cette succession de graphiques permet d'illustrer la convergence en loi de (T_n) vers Y.

IV.2.b) Valeur approchée d'une espérance

HEC – 2015

- On considère une v.a.r. X telle que $X \hookrightarrow \mathcal{E}(\lambda)$.
- On considère la fonction F définie sur $\mathbb{R}$ à valeurs réelles telle que : $F(x) = \exp(-\mathrm{e}^{-\lambda x})$.
  F est la fonction de répartition d'une v.a.r. T. *(T suit la loi de Gumbel de paramètre λ)*
- On peut démontrer que $Z = -\dfrac{1}{\lambda}\ \ln(\lambda X)$ a même loi que T.
- On suppose alors que $\lambda = 1$.
  On démontre que F réalise une bijection de $\mathbb{R}$ dans $]0,1[$ et que sa bijection réciproque G est :

$$\begin{array}{cccc} G & : & \mathbb{R} & \rightarrow &]0,1[\\ & & x & \mapsto & -\ln(-\ln(x)) \end{array}$$

  On démontre alors que $G(U)$ a même loi que T (méthode d'inversion).

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

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

ESSECI – 2016

- On considère X une v.a.r. à densité. On suppose qu'il existe un unique intervalle I_X que lequel F_X (fonction de répartition de X) réalise une bijection de classe C^1 strictement croissante de I_X sur $]0,1[$. On note alors G_X la bijection réciproque de F_X, définie de $]0,1[$ sur I_X.
- On note $\beta \in]0,1[$ un niveau de confiance.
- Enfin, on définit $r_\beta(X)$ appelé « Value at Risk » au niveau de confiance β de X, par :

$$r_\beta(X) = G_X(\beta)$$

  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 considère une suite de v.a.r. $(X_k)_{k\geqslant 1}$ mutuellement indépendantes et de même loi que X.
  Pour tout $\omega \in \Omega$, et $n \in \mathbb{N}^*$, on ordonne $X_1(\omega), \ldots, X_n(\omega)$ dans l'ordre croissant.
  On note alors $X_{1,n}(\omega), \ldots, X_{n,n}(\omega)$ les valeurs obtenues.
  En particulier, $X_{1,n}(\omega)$ est la plus petite de ses valeurs et $X_{n,n}(\omega)$ la plus grande.
- Pour tout $n \in \mathbb{N}^*$ tel que $n\beta \geqslant 1$, on s'intéresse dans l'épreuve à $Y_n = X_{\lfloor n\beta \rfloor}, n$ qui est un estimateur de $r_\beta(X)$.
- On définit « l'Expected Shortfall » de X de niveau de confiance β par :

$$ES_\beta(X) = \frac{1}{1-\beta} \int_{r_\beta(X)}^{+\infty} x f_X(x)\ dx$$

  On démontre que :

$$ES_\beta(X) = r_\beta(X) + \frac{1}{1-\beta}\ \mathbb{E}(\max(X - r_\beta(X), 0))$$

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

```
--> triCroissant([0, -1, 0, 2, 4, 2, 3])
    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 $(X_1, \ldots, X_n)$ et `beta` la valeur de β.

b) En utilisant la méthode de Monte-Carlo, dont on supposera la validité, et la fonction `VaR` précédente, 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 `X` et de la valeur de β se trouvant dans la variable `beta`.

`ESSECI – 2018`

Soit $x \in \,]0, \alpha[$.

On considère la fonction φ_x définie sur $\mathbb{R}_+$ par : $\varphi_x(t) = \begin{cases} t & \text{si } t \leqslant x \\ 0 & \text{sinon} \end{cases}$

On démontrait alors :

$$\sigma(x) = \frac{\mathbb{E}(\varphi_x(Y_{n-1}))}{\mathbb{P}([Y_{n-1} \leqslant x])}$$

où X_1, ..., X_n est une famille de v.a.r. mutuellement indépendantes et de même loi que X (à valeurs dans $]0, \alpha[$) ; pour tout $\omega \in \Omega$, $Y_n(\omega) = \max(X_1(\omega), \ldots, X_n(\omega))$ est le plus grand des réels $X_1(\omega)$, ..., $X_n(\omega)$.

Enfin, on suppose que l'on a défini une fonction **Scilab** d'entête `function x = simulX(n)` qui retourne une simulation d'un échantillon de taille n de la loi de X sous la forme d'un vecteur de longueur n.

1. En déduire une fonction **Scilab** `function s = sigma(x,n)` qui retourne une valeur approchée de $\sigma(x)$ obtenue comme quotient d'une estimation de $\mathbb{E}(\varphi_x(Y_{n-1}))$ et de $\mathbb{P}([Y_{n-1} \leqslant x])$. On utilisera la fonction `simulX` pour simuler des échantillons de la loi de X, et on rappelle que si `v` est un vecteur, `max(v)` est égal au plus grand élément de `v`.

IV.2.c) Valeur approchée d'une intégrale

`EDHEC – 2018`

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

1. On rappelle qu'en **Scilab**, la commande `grand(1, 1, 'unf', a, b)` simule une variable aléatoire suivant la loi 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, 100 000, 'unf', 0, 1)
V = log(1 + U .^2)
f = -----
disp(f)
```

- On démontrait alors :

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

2. Modifier le script précédent pour donner un valeur approchée de $\sum\limits_{k=0}^{+\infty} u_k$.

IV.3. Chaînes de Markov

`EDHEC – 2017`

- 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.

  Pour tout $n \in \mathbb{N}$, on note X_n la variable aléatoire égale au numéro du sommet sur lequel se situe le mobile à l'instant n. D'après le premier des deux points précédents, on a donc $X_0 = 1$.
- Pour tout n de $\mathbb{N}$, on considère la matrice-ligne de $\mathcal{M}_{1,4}(\mathbb{R})$:

$$U_n = \big(\mathbb{P}([X_n = 1]) \quad \mathbb{P}([X_n = 2]) \quad \mathbb{P}([X_n = 3]) \quad \mathbb{P}([X_n = 4])\big)$$

- On montre que, si l'on pose $A = \dfrac{1}{3}\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$, on a :

$$\forall n \in \mathbb{N},\ U_{n+1} = U_n\ A$$

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 ?