ECE 1 - Année 2017-2018
Lycée français de Vienne
Mathématiques - F. Gaunard
http://frederic.gaunard.com

T.P. n°8

Simulation de v.a. finies

On a déjà vu que, sans davantage de précision, tout appel de la fonction `rand()` renvoyait un nombre réel au hasard entre 0 et 1 (on dit que la fonction simule la loi uniforme sur $[0;1]$, qui sera introduite avec plus de détails dans le chapitre sur les v.a.r à densité).

Exercice 1. (D'après **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, on note sa couleur, puis on la replace dans l'urne en ajoutant une boule de la couleur de celle qui vient d'être obtenue.

Pour tout $k \in \mathbb{N}^*$, on note B_k : "la boule tirée au $k-$ième tirage est bleue" et R_k l'évènement: "la boule tirée au $k-$ième tirage est rouge".

(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, n étant l'entier entré en argument.

```
function s=EML_17(n)
    b=1; //b représente le nombre de boules bleues dans l'urne
    r=2; //r représente le nombre de boules rouges dans l'urne
    s=0; //s représente le nombre de boules rouges obtenues en n tirages
    for k=1:n
        x=rand();
        if ............................. then
            ..................................
        else
            ..................................
        end
    end
endfunction
```

(2) L'exécution du programme ci-dessous renvoie `6.657`. Comment interpréter ce résultat?

```
n=0;
m=0;
for k=1:1000
    m=m+EML_17(n);
end
disp(m/1000)
```

Exercice 2. (Loi uniforme sur $[\![a;b]\!]$)

Écrire, à l'aide de la fonction `rand()` une fonction `unifZ()` prenant en arguments deux entiers relatifs a et b ($a < b$) et simulant une loi uniforme sur $[\![a;b]\!]$.

Exercice 3. Que font les suites d'instructions suivantes?

```
-->x=1:20; y=1+floor(6*rand(x)); bar(x,y)
-->clf; x=1:1000; y=1+floor(6*rand(x)); histplot(0:6,y)
```

Exercice 4. (Tirages dans une urne)

On considère une urne U contenant 3 boules bleues, 4 boules blanches et 5 boules rouges. On effectue n tirages dans cette urne et on note X_i la variable aléatoire égale à 1 si la boule tirée au $i-$ème coup est bleue, 2 si elle est blanche et 3 si elle est rouge.

(1) Créer un programme qui demande à l'utilisateur un nombre de tirages n et simule l'expérience en affichant les valeurs des X_i. (☞ On pourra utiliser une loi uniforme sur $[\![1;12]\!]$.)
(2) Déterminer la loi et calculer mathématiquement $E(X_i)$.
(3) Modifier le programme pour qu'il affiche la moyenne des X_i. Qu'observe-t-on pour n grand?

Exercice 5. On désigne par n un entier naturel supérieur ou égal à 2. On dispose de n urnes, numérotées de 1 à n, contenant chacune n boules. On répète n épreuves, chacune consistant à choisir une urne au hasard et à en extraire une boule au hasard (qu'on ne remet pas). On suppose que les choix des urnes sont indépendants les uns des autres.

On note X la variable aléatoire prenant la valeur 1 si l'urne numérotée 1 contient toujours n boules au bout de ces n épreuves, et qui prend la valeur 0 sinon.

(1) Écrire une fonction `y=hasard(n)` permettant de simuler la variable X avec le paramètre n.
(2) On note N la variable aléatoire correspondant au nombre de boules manquantes à l'issue des n épreuves. Modifier la fonction précédente pour qu'elle renvoie également la valeur de N.

Exercice 6. (Sauts de puce)

Une puce se déplace sur le dos d'un animal (assimilué à un axe gradué de 0 à N) de façon aléatoire. Excepté à partir des deux extrémités où le prochain mouvement est déterminé, elle avant de une unité vers la gauche ou vers la droite, de manière équiprobable à chaque instant. On note X_n la variable aléatoire correspondant à la position de la puce après n sauts. Écrire un programme permettant de simuler X_n.

Exercice 7. (Loi de Bernoulli et loi Binomiale)

(1) Écrire une fonction `Bern()` qui prend en argument un nombre réel $p \in]0;1[$ et retourne ensuite la valeur prise, lors d'une réalisation, par une variable aléatoire $X \hookrightarrow \mathcal{B}(1,p)$.
(2) Créer à la suite une fonction `Bino()` qui prend en argument un entier naturel $n \geq 1$, un réel $p \in]0;1[$ et renvoie la valeur prise par une v.a. $S \hookrightarrow \mathcal{B}(n,p)$.
(3) Ajouter à la suite du programme des instructions pour demander à l'utilisateur n et p et effectuer ensuite 10000 réalisations de S puis afficher ensuite l'histogramme de ces réalisations.

Avec `grand()`

☞ Les lois usuelles sont déjà définies sous `SciLab`.

☞ Compléter le texte ci-dessous

- `grand(1,N,      ,a,b)` génère 1 échantillon de N nombres aléatoires suivant la loi $\mathcal{U}([\![a;b]\!])$.
- `grand(                )` génère m échantillons de N nombres aléatoires suivant la loi $\mathcal{U}([\![a;b]\!])$.

- `grand(1,N,"bin",n,p)` génère 1 échantillon de N nombres aléatoires suivant la loi   .
- `grand(                )` génère m échantillon de N nombres aléatoires suivant la loi   .

Exercice 8. (D'après **ECRICOME 2017**) Soit n un entier naturel non nul.
On effectue une série illimité 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.

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
```