Chapitre 8 : Simulation d'épreuves aléatoires avec la commande RAND

Scilab dispose de deux générateurs de nombres aléatoires : un générateur de base *rand* et un générateur plus sophistiqué *grand* que l'on verra dans le prochain chapitre.

I La fonction RAND

L'instruction ***rand(n,p)*** génère une matrice n lignes et p colonnes dont les coefficients sont les valeurs prises par des variables aléatoires indépendantes et suivant toutes la loi uniforme sur [0, 1] (au s.ens de le terminale S : c'est une loi continue que nous reverrons à la fin de l'année).

L'instruction ***rand()*** fait la même chose que *rand(1,1)*, c'est-à-dire génère un nombre qui est une réalisation de X où X est une variable aléatoire suivant la loi uniforme sur [0,1].

Exercices :

1) On tape l'instruction ***rand()*** en console. Quelle est la probabilité que le résultat soit entre 0.25 et 0.5 ?

2) On tape l'instruction ***3*rand()*** en console. Quelle est la probabilité que le résultat soit entre 1 et 2 ?

3) On tape l'instruction ***floor(3*rand())*** en console. Quels sont les différents résultats possibles ? Quelle est la probabilité d'obtenir 1 ?

4) On tape l'instruction ***floor(3*rand(1,20))*** en console. Quel type de résultat obtient-on ?

5) On tape la ligne d'instructions suivante : ***A=rand(1,500) ; B=find(A>0.75) ; C=size(B).*** Que prévoyez-vous comme résultat ?

II Histogrammes et diagrammes en bâtons

Soit x et y deux matrices ligne de même longueur. La commande ***bar(x,y)*** commande le tracé du diagramme en bâtons tel que l'abscisse de chaque bâton est x_i et sa hauteur y_i.

Ex : *x=[0,1,2];y=[0.25,0.5,0.25];bar(x,y)*

Donnons nous maintenant une matrice ligne x. On considère les x_i comme une série statistique que l'on souhaite représenter par un histogramme.
L'instruction ***c=linspace(min, max,n+1)*** crée n classes de même longueur entre min et max (n+1 est le nombre de réels qui délimitent ces n classes). Puis ***histplot(c,x)*** commande le tracé de l'histogramme correspondant.

Pour créer ces classes, on peut aussi utiliser ***c=min : pas : max***

Ex : Essayer *x=rand(1,100);c=0:0.1:1;histplot(c,x)* puis *x=rand(1,100);c=linspace(0,1,11);histplot(c,x)*
(commades identiques)

Exercice : On crée une matrice A par l'instruction A=floor(3*rand(1,1000)). Quelles classes proposez vous de créer pour « ranger » les valeurs contenues dans la matrice A ? Quels effectifs prévoit-on (approximativement) pour chacune de ces classes ? Donner une ligne de commandes qui réalise un histogramme adapté à cette matrice A.

REMARQUE : si on doit réaliser un histogramme pour une variable aléatoire à valeurs entières, les instructions c=0:3 ou de façon équivalente c=linspace(0,3,4) créent les classes [0,1],]1,2],]2,3] (première classe fermée gauche et droite, puis fermeture à droite, ouverture à gauche). C'est assez particulier et le plus sur me semble être l'utilisation de classes centrées en les valeurs entières.

III Simulation de variables aléatoires

Lorsqu'on s'intéresse à une variable aléatoire X, on peut vouloir faire réaliser à Scilab l'expérience aléatoire correspondante. Par exemple, si X est le numéro obtenu lors d'un lancer de dé à 6 faces, on va demander à Scilab de simuler ce lancer de dé. Pour cela, on va écrire des instructions faisant intervenir le hasard, c'est-à-dire pour l'instant la fonction rand.

1. Simulation d'une loi uniforme avec *rand*

Par exemple, la variable aléatoire X correspondant à un lancer de dé avec 6 faces équiprobables suit une loi uniforme.

Pour une loi uniforme sur {1,...,n}, la simulation se fait en une seule instruction :

```
x=
```

En effet, le résultat est un entier entre 1 et n et chacune de ces valeurs est équiprobable.

2. Simulation d'une loi de Bernoulli avec *rand*

Par exemple, si on pioche une boule dans une urne contenant deux boules noires et huit blanches et si on ote le résultat sous la forme : X=1 si la boule est noire, X=0 si la boule est blanche, la variable aléatoire X suit une loi de Bernoulli de paramètre p=0,2.

La simulation d'une loi de Bernoulli de paramètre p se fait avec les instructions :

```
if rand()<p then x=
        else x=
end
```

De cette manière, la probabilité que x prenne la valeur 1 est bien p et l'autre valeur possible pour x est 0 avec la probabilité 1-p.

3. Simulation d'une loi binomiale avec *rand*

Par exemple, si X est lenombre de boules blanches obtenues après six tirages successifs et avec remise dans l'urne précédente, X suit une loi binomiale de paramètres n=6 et p=0,2. Pour simuler X, on va simuler les six tirages dans une boucle for.

La simulation d'une loi binomiale de paramètres p et n se fait avec les instructions :

```
x=0
for k=1:n,
        if              then x=x+1
        end
end
```

La variable x compte le nombre de succès, chaque succèe étant obtenu avec la probabilité p.

Exercice : Décrire l'affichage en sortie du script suivant :

```
a=zeros(1,1000)
function[x]=bern()
if rand()<0.2 then x=1
else x=0
end
endfunction
for i=1:1000, a(i)=bern(),end
b=find(a==1) ; disp(size(b))
```

TP 8 :

Ex 1 :

1) L'instruction *x=floor(2*rand())* simule une variable aléatoire X suivant une loi usuelle. Laquelle ?

2) Compléter le script suivant pour qu'il fasse la même chose :

```
if rand(1,1)<... then x=...
        else x=...
end
```

3) Ecrire un script permettant de simuler une loi de Bernoulli de paramètre 0.75.

Ex 2 : On considère les instructions suivantes :

```
x=0
for k=1:12,  if rand(1,1)<0.7 then x=x+1;end
end
```

Elles simulent une variable aléatoire X suivant une loi usuelle. Laquelle ?

Ex 3 : Une urne contient une boule blanche et quatre boules noires.
Simuler un tirage, en codant par exemple blanc par 0 et noir par 1.

Ex 4 : Jeu de dé

1) Ecrire une fonction nommée « lancer » qui simule un lancer de dé aux 6 faces équiprobables.
2) Compléter le script suivant pour qu'il simule l'expérience aléatoire dont le résultat est le nombre de lancers jusqu'au premier 6 (celui-ci compris) :

```
x=lancer()
disp(x)
k= ...
while x<>6 then x=... ,
                disp(x)
                k=k+1
end
disp(k, 'nombre de lancers : ')
```

Ex 5 : Ecrire un script qui :

- Crée une matrice ligne de longueur 100 dont les coefficients sont les valeurs prises par des variables aléatoires indépendantes et suivant toutes la loi de Bernoulli de paramètre 0.2.
- Trace l'histogramme correspondant à cet échantillon avec deux classes : [-0.5,0.5] et [0.5,1,5].
- Trace sur la même figure le diagramme en bâtons théorique d'une loi de Bernoulli de paramètre 0.2.

Ex 6 : Jeu de pile ou face.
On joue à pile ou face n fois consécutivement avec une pièce dont les deux faces sont équiprobables. On choisit de coder pile par 1 et face par 0. Ecrire un script qui affiche les résultats des n lancers puis le nombre de pile obtenus au cours des n lancers. L'entier n est fourni par l'utilisateur.

Ex 7 : Un mobile se déplace sur les points à coordonnées entières d'un axe d'origine 0. Au départ, le mobile est à l'origine. Le mobile se déplace selon la règle suivante : s'il est sur le point d'abscisse k à l'instant n, alors, à l'instant (n +1) il sera sur le point d'abscisse (k+1) avec la probabilité 1/3 ou sur le point d'abscisse 0 avec la probabilité 2/3.
Pour tout n entier naturel, on note X_n l'abscisse de ce mobile à l'instant n et l'on a donc $X_0=0$.

1) Déterminer la loi de X_1 . Que vaut $X_n(\Omega)$?
2) Ecrire une fonction nommée « position » qui prend en entrée un entier n et qui donne la position du mobile à l'instant n.
3) Dans un script, écrire cette fonction « position » puis rajouter ce qui suit:

```
n=input('n= ? ')
A=zeros(1,1000)
for k=1:1000, A(k)=position(n);end
```

```
B=find(A==0)
C=size(B)
disp(C(2)/1000)
```

Faire tourner le programme pour plusieurs valeurs de n, et noter à chaque fois l'affichage de sortie. Interpréter ces résultats.

4) Montrer en utilisant le système complet d'événements associé à la variable aléatoire X_{n-1} que $P(X_n=0)=2/3$.

Ex 8 : *(extrait EDHEC S adapté pour Scilab)*

On désigne par n un entier naturel supérieur ou égal à 2.
On dispose de deux urnes U et V , l'urne U contenant une boule blanche et (n − 1) boules noires et l'urne V contenant une boule noire et (n − 1) boules blanches.
Un joueur choisit une urne au hasard puis il effectue des tirages d'une boule avec remise de cette boule dans l'urne obtenue.
On note U l'événement « l'urne tirée au sort est l'urne l'urne U ».
On note X le numéro du tirage où l'on obtient, pour la première fois, une boule noire.

On décide de coder l'événement U par 1 et l'événement contraire (tirage de l'urne V) par 0.

Compléter le programme suivant pour qu'il permette le calcul et l'affichage de la valeur prise par la variable aléatoire X lors de cette expérience.

```
n=input( 'entrez une valeur de n : ')
hasard=floor(2*rand())
x=0
if hasard==0      then y=0,while y<>1 then y=1+floor(n*rand()),x=x+1;end
                  else y= ... ; while ... then y= ... ,x= ... ;end
end
disp ...
```

Pour tester le fonctionnement du script, rajoutez l'affichage du tirage de l'urne. La valeur prise par X en dépend beaucoup si n est assez grand !

Chapitre 9 : Simulation d'épreuves aléatoires avec la commande GRAND
Variables discrètes

I La fonction GRAND

Cette fonction permet de simuler toutes les lois usuelles au programme.
Elle s'utilise sous la forme :

grand(n,m,'loi',paramètre 1,paramètre 2, … ,paramètre p)

Cette commande renvoie une matrice n lignes et m colonnes dont les coefficients sont les valeurs prises par des variables aléatoires indépendantes et suivant toutes la loi précisée dans les arguments.

LOI	COMMANDE
binomiale de paramètres N et p	grand (n , m , 'bin' , N , p)
uniforme sur $[\![a,b]\!]$ avec a et b entiers tels que a < b	grand (n , m , 'uin' , a , b)
géométrique de paramètre p	grand (n , m , 'geom' , p)
de Poisson de paramètre λ	grand (n , m , 'poi' , λ)

Ex : Tester les instructions suivantes sur la console une dizaine de fois et comprendre les résultats obtenus.
A=grand(1,1000,'bin',3,0.5) ; B=find((A==0)|(A==3)) ; C=size(B)
Continuer avec : (on réutilise la dernière matrice A obtenue) *c=-0.5:3.5 ; histplot(c,A)*

TP 9 :

Ex 1 : Ecrire un script qui :

- Crée une matrice ligne de longueur 100 dont les coefficients sont les valeurs prises par des variables aléatoires indépendantes et suivant toutes la loi uniforme sur $[\![1,5]\!]$.
- Trace l'histogramme correspondant à cet échantillon avec des classes adaptées (lesquelles?)
- Trace sur la même figure le diagramme en bâtons théorique d'une loi uniforme sur $[\![1,5]\!]$.

Le faire tourner quelques fois. Faire varier la taille de l'échantillon.

Ex 2 : On lance une pièce aux deux faces équiprobables et on note *Z* la variable aléatoire égale au rang du lancer où l'on obtient le premier "pile".
Après cette série de lancers, si *Z* a pris la valeur *k* ($k \in \mathbb{N}^*$), on remplit une urne de *k* boules numérotées 1, 2, ..., *k* puis on extrait au hasard une boule de cette urne.
On note *X* la variable aléatoire égale au numéro de la boule tirée après la procédure décrite ci-dessus. On décide de coder l'événement « obtenir un "pile" » par 1 et l'événement « obtenir un "face" » par 0.

1) Ecrire une fonction appelée « tirage » qui simule la variable aléatoire X.
2) Ecrire un script qui crée une matrice ligne de longueur 500 dont les coefficients sont 500 réalisations de X et qui calcule la moyenne de ces 500 valeurs.
   (pour juger la cohérence de vos résultats : on montre que X a une espérance égale à 1.5)

Ex 3 : Une urne contient une boule noire et une boule blanche. On tire une boule de l'urne. Si elle est blanche, on s'arrête. Si elle est noire, on la remet dans l'urne accompagnée d'une autre boule noire. On tire alors une boule de l'urne. Si elle est blanche, les tirages s'arrêtent. Si elle est noire, on la remet dans l'urne accompagnée d'une autre boule noire. On procède ainsi jusqu'à tirer la boule blanche.
On note alors X la variable aléatoire qui vaut 0 si la boule blanche ne sort jamais, et qui vaut le rang du tirage qui a amené pour la première fois la boule blanche sinon.

1) Pour tout entier k non nul, calculer P(X=k). En déduire P(X=0).
2) Compléter le script suivant pour qu'il simule l'expérience et affiche la valeur prise par X.

```
X=0
n=1
tirage=6
while tirage<>1 then tirage= ...
   n= ...
   X= ...
end
disp(X,'nombre de tirages : ' )
```

3) Est-on certain de sortir de la boucle *while* ? (une probabilité égale à 1 vaudra certitude)

Ex 4 : Approximation d'une loi binomiale par une loi de Poisson : écrire un script qui :

- Demande à l'utilisateur deux données : un réel $\lambda > 0$ et un entier non nul n.
- Crée une matrice ligne de longueur 5000 dont les coefficients sont les valeurs prises par des variables aléatoires indépendantes et suivant toutes la loi binomiale de paramètres n et $\frac{\lambda}{n}$.
- Crée une autre matrice ligne de longueur 5000 dont les coefficients sont les valeurs prises par des variables aléatoires indépendantes et suivant toutes la loi de Poisson de paramètre λ.
- Trace les deux histogrammes sur la même figure (correspondant à ces deux échantillons) avec les classes [-0.5,0.5] ... [9.5,10.5]. Utiliser un argument de couleur optionnel pour distinguer les deux histogrammes : on tape *histplot(c,x,style=2)* par exemple.

Faire tourner ce script avec les valeurs suivantes et observer :

λ	3	3	3
n	6	20	100

Ex 5 : On dispose d'une urne contenant 10 boules, numérotées de 1 à 10. On effectue deux tirages d'une boule au hasard, avec remise de la première boule tirée. Le script ci-dessous simule cette expérience et affiche en sortie une valeur proche de l'espérance d'une certaine variable aléatoire X. Laquelle ?

```
e=0
for k=1:5000, t=grand(1,2,'uin',1,10)
   if t(1)<t(2) then X=t(2)
      else X=t(1)
   end
   e=e+X
end
disp(e/5000,'une valeur approchée de E(X) est : ')
```

Rq : Nous avons calculé la valeur théorique de cette espérance dans un TD, c'est 7.15, à vous de retrouver l'exercice en question, ou de refaire les calculs.

Ex 6 : On considère une suite de lancers indépendants d'une pièce pour laquelle la probabilité d'apparition de Pile est 2/3 et celle de Face 1/3. On note X la variable aléatoire égale au nombre de lancers nécessaires pour l'obtention, pour la première fois, de deux piles consécutifs. On a montré (TD 25 ex.4) que X admet une espérance égale à E(X)=3.75.
Le script suivant fournit une simulation de cette expérience. Pile est codé par 1 et Face par 0. En sortie, on veut qu'il affiche la moyenne du rang d'apparition du premier double pile sur n expériences, l'entier n étant fourni par l'utilisateur.
A vous de compléter les instructions manquantes.

```
n=input('entrez le nombre d''expériences : ')

function[k]=attente()
   x=grand(1,1,'bin',1,2/3)
   y=grand(1,1,'bin',1,2/3)
   k=2
   while x*y==0 then ...
         ...
         ...
   end
endfunction

e=0
for i=1:n, ...
end
e=e/n
disp(e,'moyenne')
```

Ex 7 : Une urne contient 11 boules numérotées de 1 à 11. On tire successivement 2 boules sans remise dans cette urne.

1) Ecrire un script qui simule cette expérience. On veut qu'à l'exécution s'affichent les deux numéros sortis.
2) Ecrire une fonction qui simule la variable aléatoire Z égale au maximum des deux numéros.
3) Créer une matrice ligne de longueur 5000 dont les coefficients sont 5000 réalisations de Z puis tracer l'histogramme correspondant à cet échantillon avec des classes adaptées (lesquelles?). Comprendre les résultats en déterminant par le calcul la loi de X.