TP 11 – Simulation de variables aléatoires finies

Avant-propos : la fonction **rand**

- **rand()** renvoie un nombre aléatoire réel de l'intervalle $[0,1]$, suivant la loi uniforme continue sur $[0,1]$.
- **rand(n,p)** renvoie une matrice taille (n,p) contenant les nombres aléatoires réels de l'intervalle $[0,1]$, suivant la loi uniforme continue sur $[0,1]$.

Application pratique

1. Exécutez plusieurs fois l'instruction `rand()` dans la console. Vérifiez que le nombre obtenu est toujours un nombre compris entre 0 et 1, aléatoire.
2. Exécutez plusieurs fois l'instruction `rand(2,5)` dans la console. Vérifiez que vous obtenez à chaque fois une matrice contenant de nombres compris entre 0 et 1, aléatoires.
3. Que contient la variable A après l'instruction **A=**`rand(1000,1000);` ?

1 Simulation de la loi uniforme finie.

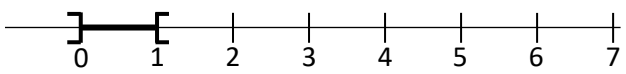
1.1 Simulation d'un lancer de dé.

On souhaite simuler une variable aléatoire prenant ses valeurs de façon équiprobable dans $[\![1,6]\!]$.

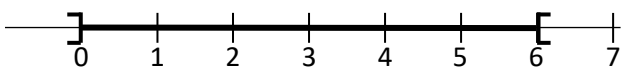
1.1.1 Avec la fonction rand()

```
X=6*rand()+1
```

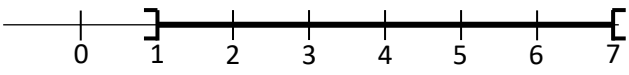
rand() renvoie un réel dans $]0,1[$



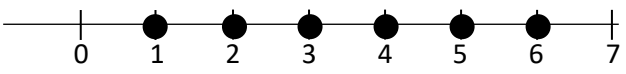
6*rand() renvoie un réel dans $]0,6[$



6*rand()+1 renvoie un réel dans $]1,7[$



floor(6*rand()+1) renvoie un entier dans $[\![1,6]\!]$



1.1.2 Avec la fonction `grand()` :

```
X=grand(1,1,"uin",1,6)
```

Application pratique : dans la console

1. **Simulons 20 lancers de dés :**

```
--> X=grand(1,20,"uin",1,6)
 X  =

   3.   1.   3.   6.   5.   2.   6.   6.   1.   2.   2.   6.   1.   1.   6.   1.   4.   5.
```

2. **Simulons 20 lancers de dés d'une autre façon :**

   Entourez les 6 obtenus

```
--> X=floor(6*rand(1,20)+1)
 X  =

   4.   2.   4.   6.   5.   1.   5.   3.   2.   6.   6.   2.   6.   5.   6.   1.   2.   5.
```

3. **Repérons les 6 :**

```
--> X==6
 ans  =

  F F F T F F F F F T T F T F T F F F F F
```

4. **Obtenons les emplacements des 6**

```
--> find(X==6)
 ans  =

   4.   10.   11.   13.   15.
```

5. **Comptons les 6 :**

```
--> sum(X==6)
 ans  =

   5.
```

6. **Comptons les 6 d'une autre façon :**

```
--> length(find(X==6))
 ans  =

   5.
```

7. **Calculons la moyenne des 20 résultats.**

```
-->
```

1.2 Simulation d'une variable aléatoire suivant la loi uniforme sur $[\![a, b]\!]$

Rappel : loi Uniforme Finie

On dit qu'une variable aléatoire X suit la loi uniforme sur $[\![a, b]\!]$ où a et b sont deux entiers, si elle peut prendre toutes les valeurs de $[\![a, b]\!]$, de façon équiprobable. On note :

$$X \hookrightarrow \mathcal{U}([\![a, b]\!])$$

Il y a …….. valeurs dans l'intervalle d'entiers $[\![a, b]\!]$, donc la probabilité que la variable X prenne chaque valeur $k \in [\![a, b]\!]$ vaut :

$$P(X = k) =$$

1.2.1 Avec la fonction rand()

```
X=floor((b-a+1)*rand()+a)
```

1.2.2 Avec la fonction grand()

```
X=grand(1,1,"uin",a,b)
```

Exemple 1 :

Simuler un tirage aléatoire dans une urne contenant 10 boules indiscernables numérotées de 1 à 10 :

Exemple 2 :

Simuler le choix de 100 nombres pris entre 10 et 20, de façon équiprobable.

Exemple 3 :

Ecrire un programme qui simule 10 tirages dans une urne qui contient 3 rouges et 2 vertes. On présentera le résultat dans un vecteur contenant un 1 si on a tiré une rouge et un 2 si on a tiré une verte.

Solution 1 :

```
//On crée un vecteur X :
X=floor(5*rand(1,10)+1)
// ou avec grand :
X=grand(1,10,"uin",1,5)

//On crée un vecteurs de 1
Y=ones(1,10)

//On parcours le vecteur X à l'aide d'une boucle for et on change le vecteur Y si besoin :
for k=1:10
    if X(k)>=4
        Y(k)=2
    end
end

//On affiche Y si besoin
disp(Y)


```

Solution 2 : sans boucle for (pour les experts)

```
//On crée un vecteur X :
X=floor(5*rand(1,10)+1)
// ou avec grand :
X=grand(1,10,"uin",1,5)

//On crée un vecteurs de 1
Y=ones(1,10)

//On change directement les valeurs de Y qu'il faut changer :

Y(find(X>=4))=2

//On affiche Y si besoin
disp(Y)
```

2 Simulation de la loi de Bernoulli

Rappel : loi de Bernoulli
On dit qu'une variable aléatoire X suit « la loi de Bernoulli de paramètre p » si :

$$\begin{cases} X(\Omega) = \{0,1\} \\ P(X=1) = p \\ P(X=0) = 1-p \end{cases}$$

On note :

$$X \hookrightarrow \mathcal{B}(p)$$

Exemple : si on lance un dé et qu'on note X la variable aléatoire égale à 1 si le dé tombe sur 6 et à 0 sinon, alors X suit la loi $\mathcal{B}\left(\frac{1}{6}\right)$

2.1 Simuler une loi de Bernoulli avec la fonction rand().

Exemple : on souhaite simuler une occurence d'une variable aléatoire X de loi $\mathcal{B}(1/4)$.

rand()]——————————[
0 1

```
r = rand()

if ______________ then

    X = 1

else

    X = 0

end
disp(X)
```

À retenir

Pour simuler une occurence d'une variable aléatoire X de loi $\mathcal{B}(p)$:

```
r = rand()

if ______________ then

    X = 1

else

    X = 0

end

disp(X)
```

2.2 Simuler une loi de Bernoulli avec la fonction grand().

À retenir

Pour simuler une occurence d'une variable aléatoire X de loi $\mathcal{B}(p)$:

```
grand( 1 , 1 , "bin" , 1 , p )
```

Pour simuler plusieurs occurences indépendantes d'une variable aléatoire X de loi $\mathcal{B}(p)$:

```
grand( r , s , "bin" , 1 , p )
```

3 Simulation de la loi Binomiale.

3.1 Rappel : la loi Binomiale.

Rappel : loi de Bernoulli
On dit qu'une variable aléatoire X suit « la loi de Binomiale de paramètres n et p » si elle correspond au nombre fois où se produit un évènement A (on dit le nombre de « succès »), de probabilité d'apparition p, lors de n répétitions indépendantes d'une même expérience.

On note :

$$X \hookrightarrow \mathcal{B}(n,p)$$

Exemple 1 : On lance 10 fois un dé et on note X le nombre de 6 obtenus. $X \hookrightarrow$ ……..

Exemple 2 : On lance 5 fois une pièce et on note Y le nombre de Pile obtenus. $Y \hookrightarrow$ ……..

Exemple 3 : On tire avec remises 10 boules dans une urne contenant 3 rouges et 2 vertes et on note Z le nombre de boules rouges obtenues : $Z \hookrightarrow$ ……..

Exemple 4 : Une famille a deux enfants et on note X le nombre de filles parmi ces deux enfants : $X \hookrightarrow$ ……..

On pourra remarquer que la loi de Bernoulli et la loi Binomiale avec 1 répétition.

3.2 Simuler une loi Binomiale avec rand()

Soit X une variable aléatoire de loi $\mathcal{B}(n,p)$. On souhaite simuler une occurence de X. On sait que X représente le **nombre de succès** quand on réalise n épreuves de Bernoulli indépendantes.

À retenir

Pour simuler une occurence d'une variable aléatoire X de loi $\mathcal{B}(n,p)$:

```
X = 0  // compteur du nombre de succès
for k = ___ : ___
    r = rand()
    if _____________ then  // succès
        X = ______          // on ajoute un succès
    end
end
disp(X)
```

Version plus compacte :

```
X = 0  // compteur du nombre de succès
for k = ___ : ___
    if _____________ then  // succès
        X = ______          // on ajoute un succès
    end
end
disp(X)
```

3.3 Simuler une loi Binomiale avec grand()

À retenir

Pour simuler une occurence d'une variable aléatoire X de loi $\mathcal{B}(n,p)$:

```
grand( 1 , 1 , "bin" , n , p )
```

Pour simuler plusieurs occurences indépendantes d'une variable aléatoire X de loi $\mathcal{B}(n,p)$:

```
grand( r , s , "bin" , n , p )
```

Exercices

Exercice 1

On considère une urne contenant 8 boules bleues et 5 jaunes.

1. On réalise n tirages avec remise et on note X le nombre de boules jaunes obtenues. Créer un programme qui demande le nombre n l'utilisateur, puis qui simule cette expérience et affiche la valeur de X.

2. Modifiez le programme précédent pour qu'il réalise non pas une mais 1000 simulations de cette expérience et qu'il affiche la valeur moyenne de X sur ces 1000 simulations.

Exercice 2 :

On considère une pièce qui tombe sur Pile avec la probabilité $p = 0{,}3$ sinon sur Face.

1. Ecrire une fonction « lancerPièce() » qui simule un lancer de cette pièce. La fonction renverra 1 si la pièce est tombée sur Pile et 0 sinon. On respectera la contrainte suivante :
   Contrainte : Pour cette fonction, on n'utilisera pas la fonction grand(), uniquement la fonction rand() !

2. Utiliser la fonction ci-dessus pour créer une fonction « lancerNPièces(N) » qui simule N lancers de cette pièce. La fonction renverra un vecteur ligne de taille N. Contenant des 1 et des 0 selon que la pièce est tombée sur Pile ou sur Face.
   On respectera ici encore la contrainte consistant à ne pas utiliser grand()

3. A l'aide de la fonction lancerNPièces(N), réaliser 100 000 lancers de cette pièce et compter le nombre de Piles obtenus.

4. Modifiez votre fonction « lancerNPièces(N) » ci-dessus en utilisant maintenant la fonction grand().
   Votre fonction doit tenir en une ligne et ne plus utiliser la fonction « lancerPiece() ».

   Réalisez de nouveau 100 000 simulations et remarquez que les calculs sont bien plus rapides (du coup vous pouvez effectuer 1000 000 simulations, voire plus.

Exercice 3 :
Ecrire une fonction « lancerDé(N) » qui réalise N lancers d'un dé équilibré et renvoie, dans une matrice ligne de taille 6, le nombre de 1 obtenus, puis le nombre de 2, puis le nombre de 3, etc…

Exercice 4 :

On considère une urne contenant 4 boules violettes et 7 boules oranges. On réalise 5 tirages **sans remise** dans cette urne.

Créer une fonction appelée « premiereOrange » qui simule cette expérience et renvoie le rang de la première boule orange obtenue. On complètera pour cela la fonction ci-dessous :

```
function R= premiereOrange()
    v=4
    o=7









endfunction
```

Exercice 5 (pour les experts) :

On considère une urne contenante v boules violettes et o boules oranges. On tire **sans remise** toutes les boules de cette urne.

Créer une fonction appelée « niemeOrange(v,o,n) » qui simule cette expérience et renvoie le rang de la n-ième boule orange obtenue. On complètera pour cela la fonction ci-dessous.