SOLUTION TP n° 5

Solution 1. La fonction préimplémentée `sample` prélève successivement un nombre donné d'éléments d'un ensemble de cardinal fini, avec ou sans remise. Plus précisément :

`sample(x, size, replace = FALSE, prob = NULL)`

où:

- `x`: Décrit l'ensemble dans lequel on va échantillonner. Il s'agit soit d'un vecteur, soit d'un entier k. Dans ce dernier cas, l'ensemble considéré sera tous les entiers allant de 1 à k.
- `size`: Un entier positif ou nul donnant le nombre d'élément à tirer. Par défaut, il s'agit du cardinal de l'ensemble décrit par `x`.
- `replace` (ou `rep`): Un booléen : TRUE (ou T) signifie avec remise, et FALSE (ou F), sans remise.
- `prob`: Un vecteur de poids de probabilités d'obtenir les éléments de l'ensemble à partir duquel on échantillonne.

Considérons dans un premier temps un dé équilibré. On effectue 500 lancers, puis on regarde l'effectif des résultats :

```
resdé = sample(6, 500, replace = T)
table(resdé)
```

On réitère l'expérience, mais cette fois avec un dé pipé (une chance sur deux de tomber sur la valeur 1) :

```
resdépipé = sample(6, 500, replace = T, prob = c(0.5, 0.1, 0.1, 0.1, 0.1, 0.1))
table(resdépipé)
```

Pour comparer les deux résultats, on peut réaliser le graphique suivant :

```
titre1 = "Fréquences obtenues pour 500 lancers de dé équilibré"
titre2 = "Fréquences obtenues pour 500 lancers de dé pipé"
par(mfrow = c(1, 2))
barplot(table(resdé) / 500, main = titre1)
barplot(table(resdépipé) / 500, main = titre2)
```

Solution 2. Écrire une commande qui renvoie le résultat

1. du lancer d'un dé :

   ```
   sample(6, 1)
   ```

2. du lancer de 2 dés :

   ```
   sample(6, 2, replace = T)
   ```

3. de la somme des résultats du lancer de 2 dés :

   `sum(sample(6, 2, replace = T))`

4. du tirage du loto (5 numéros parmi 49) :

   `sample(49, 5)`

Solution 3. Soit X une *var* dont la loi est donnée par

$$\mathbb{P}(X = 0) = 0.2, \qquad \mathbb{P}(X = 2) = 0.5, \qquad \mathbb{P}(X = 5) = 0.3.$$

Simuler 1000 réalisations de X et préciser les effectifs associés aux valeurs de X :

```
x = sample(c(0, 2, 5), 1000, replace = T, prob = c(0.2, 0.5, 0.3))
table(x)
```

Solution 4. Une urne contient $p + q$ boules, dont p rouges et q noires. Créer une fonction `Urne` à 3 arguments (k, p, q) qui modélise le résultat de k tirages sans remise d'une boule de l'urne. Par exemple, la commande `Urne(6, 8, 5)` renvoie : `[1] "Rouge" "Noire" "Noire" "Rouge" "Rouge" "Rouge"` :

```
Urne = function(k, p, q)
{
contenu = rep(c("Rouge", "Noire"), c(p, q))
sample(contenu, k)
}
```

Solution 5. Lorsqu'on effectue n tirages indépendants d'une même expérience aléatoire, on appelle fréquence du résultat k le rapport entre le nombre de fois où k est tiré, et n.

Par exemple, si on jette 7 fois un dé cubique équilibré, avec pour résultats : 1; 1; 5; 2; 6; 5; 3, alors la fréquence de 5 est 2/7, celle de 4 est 0.

Écrire une fonction `Freq` à un paramètre n qui renvoie la fréquence de 5 lors de n tirages indépendants d'un dé cubique équilibré :

```
Freq = function(n) {
tirage = sample(1:6, n, replace = T)
sum(tirage == 5) / n
}
```

Comparer les fréquences pour $n \in \{10, 100, 1000\}$, avec la probabilité (théorique) d'obtenir un 5 lorsqu'on lance un dé :

`Freq(10)` renvoie (ici) : `[1] 0.4`
`Freq(100)` renvoie (ici) : `[1] 0.22`
`Freq(1000)` renvoie (ici) : `[1] 0.165`

On constate que cette dernière probabilité est proche de 1/6, la probabilité théorique d'obtenir un 5 lorsqu'on lance un dé.

Solution 6. Une urne contient 15 boules, dont 5 blanches et 10 noires. On considère les deux expériences suivantes :

- E1: On tire successivement 10 boules dans l'urne, avec remise,

- E2: On tire successivement 10 boules dans l'urne, sans remise.

1. Simuler une réalisation de chacune des deux expériences avec la fonction `sample`. On représentera une boule blanche par le chiffre 1 et une boule noire par le chiffre 0 :

   - Pour E1 :
     ```
     sample(c(0, 1), 10, replace = T, prob = c(2 / 3, 1 / 3))
     ```
   - Pour E2 :
     ```
     sample(c(rep(0, 10), rep(1, 5)), 10)
     ```

2. On s'intéresse à la *var* X égale au nombre de boules blanches tirées lors l'expérience E1, et à la *var* Y, l'analogue mais avec l'expérience E2.

   (a) Simuler 500 réalisations de chacune de ces *var* en utilisant la fonction `sample` :

   - Pour X :
     ```
     x = c()
     for(i in 1:500){
     x[i] = sum(sample(c(0, 1), 10, replace = T, prob = c(2 / 3, 1 / 3)))
     }
     ```
   - Pour Y:
     ```
     y = c()
     for(i in 1:500){
     y[i] = sum(sample(c(rep(0, 10), rep(1, 5)), 10))
     }
     ```

   (b) Comparer, suivant le type d'épreuve, le diagramme en barre des fréquences observées avec la distribution des lois binomiales et hypergéométriques correpondantes :
   ```
   titre1 = "Fréquences obtenues pour 500 tirages avec remise"
   titre2 = "B(10, 1/3)"
   titre3 = "Fréquences obtenues pour 500 tirages sans remise"
   titre4 = "H(10, 5, 10)"
   par(mfrow = c(2, 2))
   barplot(table(x) / 500, main = titre1)
   barplot(dbinom(0:10, 10, 1/3), names.arg = 0:10, main = titre2)
   barplot(table(y) / 500, main = titre3)
   barplot(dhyper(0:5, 5, 10, 10), names.arg = 0:5, main = titre4)
   ```

Solution 7. On considère la marche aléatoire suivante: un mobile est positionné à l'origine d'un axe. À chaque étape, il se déplace d'une distance de longueur 1 vers la droite ou la gauche avec la probabilité 0.5 pour chaque direction. Il effectue n étapes au total. Soit X_i la position du mobile à l'étape i (on pose X_0=0). Simuler une réalisation du vect de *var* $(X_0, X_1, \ldots, X_n)$ avec $n = 10000$ *(on pourra utiliser la fonction* `cumsum` *qui donne la somme cumulée d'un vecteur. Par exemple, la commande* `cumsum(c(2, 2, 2, 3))` *renvoie :* `[1] 2 4 6 9`*)* :

```
n = 10000
u = sample(c(-1,1), n, replace = T)
```

```
x = cumsum(u)
x = c(0, x)
```

Solution 8. On lance 5 dés cubiques équilibrés, puis on relance ceux qui n'ont pas fait 6, et ainsi de suite jusqu'à ce que les 5 dés affichent 6.

Simuler une réalisation de cette expérience aléatoire en affichant les chiffres obtenus après chaque lancers et le nombre de lancers total *(on pourra utiliser une boucle* `while` *et la commande* `sum(x != 6)` *qui calcule le nombre de composantes d'un vecteur* `x` *différentes de* 6*)* :

```
n = 0
x = rep(0, 5)
while(sum(x != 6) != 0)
{
x[x != 6] = sample(1:6, sum(x != 6), rep = T)
print(x)
n = n + 1
}
print(n)
```