Travaux pratiques n° 1

Initiation à R

Une note tenant compte du déroulement de votre TP (implication, autonomie, ...) pourra être attribuée dans le cadre du contrôle continu. Une absence injustifiée ou un oubli de vos identifiants donne lieu à la note 0.

1 Variables aléatoires

En `R`, les noms des lois usuelles sont : `binom` pour la loi binomiale, `geom` pour la loi géométrique, `pois` pour la loi de Poisson, `unif` pour la loi uniforme, `exp` pour la loi exponentielle, `norm` pour la loi normale, `gamma` pour la loi Gamma, ...

Soit X une variable aléatoire réelle suivant la loi notée `loi`. Il existe plusieurs fonctions associées à ces lois, notées `*loi`, où `*` est remplacé par une lettre :

- `dloi` est la densité de `loi` si X est continue ou alors $\mathbb{P}(X = x)$ si X est discrète (`d` comme `densité`).,
- `ploi` est la fonction de répartition de `loi` (`p` comme `probabilité`).,
- `qloi` est le quantile de `loi` (`q` comme `quantile`).,
- `rloi` renvoie une ou des simulation(s) de `loi` (`r` comme `random`).

1.1 La densité

Le nombre d'arguments pour l'appel de la fonction `dloi` dépend du nombre de paramètres de chacune des lois. Reprenons les mêmes notations que celles du cours.

Le premier argument est `x` la variable aléatoire ou le vecteur aléatoire. Les arguments suivants sont les paramètres de la loi :

- si X est discrète : `dbinom(x,n,p)`, `dgeom(x,p)`, `dpois(x,lambda)`, ...
- si X est continue : `dunif(x,a,b)`, `dexp(x,lambda)`, `dnorm(x,m,sigma)`, ...

Attention, pour la loi normale, si $X \sim \mathcal{N}(m, \sigma^2)$ (où m est l'espérance de X et σ^2 sa variance), il faut effectuer l'appel `dnorm(x,m,sigma)` où `sigma` est l'écart-type.

Exercice 1: Pour compléter les lois ci-dessus, taper `help("dgamma")`, `help("dchisq")`, `help("dt")`, `help("df")`.

RÉPONSE :

Exercice 2: Supposons que $X \sim \mathcal{B}(8, 0.3)$, alors

$$\mathbb{P}(X = 4) = \binom{8}{4} 0.3^4 (1 - 0.3)^{8-4}.$$

Comparer `dbinom(4,8,0.3)` et `choose(8,4)*0.3^4*(1-0.3)^(8-4)`.

RÉPONSE :

Exercice 3: Supposons que $X \sim \mathcal{N}(2, 0.12^2)$, alors

$$f_X(1.7) = \frac{1}{0.12\sqrt{2\pi}} \exp\left(-\frac{(2-1.7)^2}{2 * 0.12^2}\right).$$

Comparer `dnorm(1.7,2,0.12)` et `1/sqrt(2*pi*0.12^2)*exp(-(2-1.7)^2/(2*0.12^2))`.

RÉPONSE :

Exercice 4: `x` et les paramètres des lois peuvent aussi être des vecteurs. Tester les commandes :

```
c(4,6)
dbinom(c(4,6),8,0.3)
c(1,2,3)
dexp(2,c(1,2,3))
vec=dexp(2,c(1,2,3))
vec
```

RÉPONSE :

Exercice 5: Pour représenter graphiquement les valeurs $\mathbb{P}(X = x)$ d'une variable aléatoire discrète, on utilise `plot`. Que trace la commande suivante ?

```
plot(0:5,dbinom(0:5,5,0.2),type="h",ylab="P(X=x)")
```

RÉPONSE :

Exercice 6: Pour représenter graphiquement la densité d'une variable aléatoire continue, on utilise `curve`. Que trace la commande suivante ?

```
curve(dnorm(x,5,1.5),0.5,9.5,ylab="fX(x)")
```

RÉPONSE :

1.2 La fonction de répartition

La fonction `ploi` renvoie la fonction de répartition de `loi`. Le nombre d'arguments dépend du nombre de paramètres de chacune des lois. Reprenons les mêmes notations que celles du cours.

Exercice 7: Si X est discrète, alors

$$\forall x \in \mathbb{N}, \quad F_X(x) = \mathbb{P}(X \leq x) = \sum_{k=0}^{x} \mathbb{P}(X = k).$$

Comparer `pbinom(4,8,0.3)` et `sum(dbinom(0:4,8,0.3))`

RÉPONSE :

Exercice 8: Si X est continue, alors

$$\forall x \in \mathbb{R}, \quad F_X(x) = \mathbb{P}(X \leq x) = \int_{-\infty}^{x} f_X(x)\ dx.$$

Comparer `pnorm(12,9,2)` et `integrate(function(x) dnorm(x,9,2),-100,12)`.

RÉPONSE :

Exercice 9: Si X est continue, alors on définit la fonction de survie par

$$\forall x \in \mathbb{R}, \quad S_X(x) = \mathbb{P}(X > x) = \int_x^{+\infty} f_X(x)\, dx.$$

Comparer `pexp(2,3,lower.tail=FALSE)`, `1-pexp(2,3)` et `exp(-6)`

RÉPONSE :

Exercice 10: Si X est discrète, alors on représente graphiquement la fonction de répartition de X avec la fonction `stepfun`. Que représente le tracé suivant ?

```
plot(stepfun(0:15,c(0, pbinom(0:15, 15, 0.6))),ylab="FX(x)",main="Fonction de répartition")
```

RÉPONSE :

Exercice 11: Si X est continu, alors on représente graphiquement la fonction de répartition de X avec la fonction `curve`. Que représente le tracé suivant ?

```
curve(pnorm(x,5,1.5),0.5,9.5,ylab="FX(x)")
```

RÉPONSE :

1.3 Les quantiles

La fonction `qloi` renvoie les quantiles de `loi`. Le nombre d'arguments dépend du nombre de paramètres de chacune des lois. Reprenons les mêmes notations que celles du cours.

Exercice 12: Si X est discrète et $p \in]0,1[$, le p-ième quantile de X est l'entier

$$x_p = \inf\{k \in \mathbb{Z} : \quad F_X(k) \geq p\}.$$

Que renvoie `qbinom(0.25,5,0.6)` ?
Vérifier avec `pbinom(0:5,5,0.6)`.

RÉPONSE :

Exercice 13: Si X est continu et $p \in]0,1[$, le p-ième quantile de X est le réel x_p tel que

$$F_X(x_p) = p.$$

Que renvoie `qnorm(0.975,0,1)` ?
Vérifier avec `pnorm(1.96,0,1)`.

RÉPONSE :

Remarque : $x_{1/2}$ est appelé **médiane** de X. La médiane vérifie les deux égalités

$$P(X \leq x_{1/2}) = 1/2 = P(X > x_{1/2}).$$

2 Simulation

2.1 Simulation de variables aléatoires

Pour simuler des réalisations de `n` variables aléatoires indépendantes, on utilise la fonction `rloi` dont le premier argument sera `n`, suivi des arguments de `loi` en reprenant les mêmes notations que celles du cours.

Exercice 14: Tester et commenter :
`rpois(10,2)`, `sum(rbinom(80,1,0.02))`, `rbinom(1,80,0.02)`, `x=rnorm(15,22,2)` et `x`.

RÉPONSE :

2.2 Simulation de dé, pile ou face, urne

Exercice 15: On crée une urne contenant des boules numérotées de 1 à 10 avec la commande `urne<-1:10`.
On vérifie le contenu de l'urne en tapant `urne`.
Si on veut faire un tirage sans remise, on tape `sample(urne,1)`.
On vérifie de nouveau le contenu de l'urne en tapant `urne`.
On peut réaliser une nouvelle expérience (indépendante de la précédente) de 8 tirages sans remise avec la commande `sample(urne,8)`.
On vérifie qu'il est impossible de faire 11 tirages sans remise avec la commande `sample(urne,11)`.

RÉPONSE :

<u>Exercice 16:</u> On crée une urne contenant des boules numérotées de 1 à 10 avec la commande `urne<-1:10`.
On vérifie le contenu de l'urne en tapant `urne`.
Si on veut faire un tirage avec remise, on tape `sample(urne,1,replace=TRUE)`.
On vérifie le contenu de l'urne avec la commande `urne`.
On peut réaliser une nouvelle expérience (indépendante de la précédente) de 8 tirages avec remise avec la commande `sample(urne,8,replace=TRUE)`.
Puis, une autre de 12 tirages avec remise avec la commande `sample(urne,12,replace=TRUE)`.

RÉPONSE :

<u>Exercice 17:</u> On crée un dé cubique avec la commande `dé<-1:6`.
On vérifie le contenu du dé en tapant `dé`.
Si on veut faire un lancer de dé, on tape `sample(dé,1)`.
On vérifie que le dé a bien ses six faces avec la commande `dé`.
On peut de nouveau simuler 8 lancers indépendants avec la commande `sample(dé,8,replace=TRUE)`.
On vérifie de nouveau que le dé a encore ses six faces avec la commande `dé`.

RÉPONSE :

<u>Exercice 18:</u> On crée une piéce non truquée avec la commande `pièce<-c("pile","face")`.
On vérifie le contenu de la pièce en tapant `pièce`.
Si on veut faire un lancer de la pièce, on tape `sample(pièce,1)`.
On peut de nouveau faire 4 lancers indépendants avec la commande `sample(pièce,4,replace=TRUE)`.

RÉPONSE :

<u>Exercice 19:</u> Nous venons de simuler des jeux de hasard avec un ordinateur qui est déterministe. Vérifions que le générateur d'aléatoire est raisonnable.
Pour initialiser le générateur d'aléatoire, on fait appel à la fonction `set.seed`.
Nous allons ensuite effectuer 600 000 lancers d'un dé et observer combien de fois chaque face a été obtenue avec la fonction `table` (ce qui nous permettra d'estimer la probabilité de tomber sur chaque face).

```
set.seed(1)
dé<-1:6
obs<-table(sample(dé,600000,replace=TRUE))
barplot(obs, main = "Résultat de l'expérience", xlab = "Face du dé",
           ylab ="Nombre de tirages obtenus")
```

RÉPONSE :