Introduction aux lois de probabilité avec R

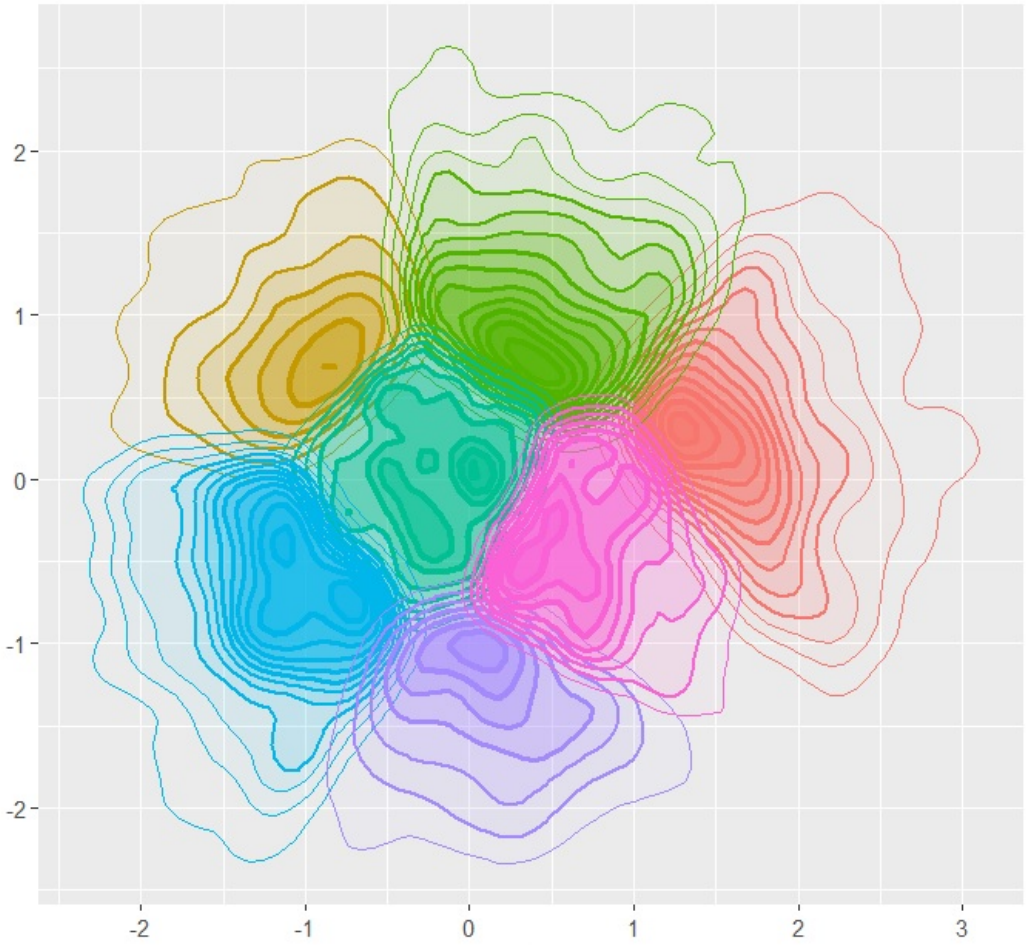
Christophe Chesneau

Université de Caen

Introduction aux lois de probabilité avec R

Christophe Chesneau

http://www.math.unicaen.fr/~chesneau/



Caen, le 10 Août 2016

Table des matières

~ Note ~

L'objectif de ce document est de présenter les principales commandes R associées aux lois de probabilités et simulations de variables aléatoires réelles (*var*). Quelques éléments théoriques sont consultables ici : `http://www.math.unicaen.fr/~chesneau/form-prob-MATHS.pdf`

Contact : `christophe.chesneau@gmail.com`

Bonne lecture !

1 Syntaxe générale

Pour une *var* X suivant une loi notée `loi` dans R, la syntaxe générale est la suivante :

- pour obtenir "la densité" de X, la commande est : `dloi` ; on ajoute la lettre d devant `loi`,
- pour obtenir la fonction de répartition de X, la commande est : `ploi` ; on ajoute la lettre p devant `loi`,
- pour obtenir le quantile de X, la commande est : `qloi` ; on ajoute la lettre q devant `loi`,
- pour simuler des réalisations de *var* suivant la même loi que X, la commande est : `rloi` ; on ajoute la lettre r devant `loi`.

Ci-dessous, un tableau grossier récapitulatif :

Loi : `loi`	densité	fonction de répartition	quantile	simulation
notations	$f(x)$; $\mathbb{P}(X = x)$	$F(x)$	valeur liée à $F(x)$	$x_1, \ldots, x_n$
commandes	`dloi`	`ploi`	`qloi`	`rloi`

Les noms `loi` les plus célèbres sont : `norm` (pour la loi normale), `binom` (pour la loi binomiale), `unif` (pour la loi uniforme), `geom` (pour la loi géométrique), `pois` (pour la loi de Poisson), `t` (pour la loi de Student), `chisq` (pour la loi du Chi-deux), `exp` (pour la loi exponentielle), `f` (pour la loi de Fisher)...

La suite de ce document apporte des détails, des exemples et des illustrations à ces commandes.

2 Densité

Définition

- Pour une *var* X discrète, on appelle "la densité" de X en x la probabilité $\mathbb{P}(X = x)$.
- Pour une *var* X à densité de densité f_X, on appelle "la densité" de X en x la fonction $f_X(x)$.

Dans les deux cas, la densité ainsi définie caractérise la loi de X.

Commandes

Si la loi de X dépend d'un ou de plusieurs paramètres, disons `par1` et `par2`, alors la densité de X en x est donnée par les commandes :

```
dloi(x, par1, par2)
```

Quelques exemples sont décrits ci-dessous :

Loi	binomiale	géométrique	Poisson
paramètres	$n \in \mathbb{N}^*,\ p \in]0,1[$	$p \in]0,1[$	$\lambda > 0$
$X \sim$	$\mathcal{B}(n,p)$	$\mathcal{G}(p)$	$\mathcal{P}(\lambda)$
$X(\Omega)$	$\{0,\dots,n\}$	$\mathbb{N}$	$\mathbb{N}$
$\mathbb{P}(X = x)$	$\binom{n}{x} p^x (1-p)^{n-x}$	$p(1-p)^x$	$e^{-\lambda}\frac{\lambda^x}{x!}$
commandes	`dbinom(x, n, p)`	`dgeom(x, p)`	`dpois(x, lambda)`

Loi	uniforme	exponentielle	normale
paramètres	$(a,b) \in \mathbb{R}^2,\ a<b$	$\lambda > 0$	$\mu \in \mathbb{R},\ \sigma > 0$
$X \sim$	$\mathcal{U}([a,b])$	$\mathcal{E}(\lambda)$	$\mathcal{N}(\mu, \sigma^2)$
$X(\Omega)$	$[a,b]$	$[0,\infty[$	$\mathbb{R}$
$f_X(x)$	$\frac{1}{b-a}$	$\lambda e^{-\lambda x}$	$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
commandes	`dunif(x, a, b)`	`dexp(x, lambda)`	`dnorm(x, mu, sigma)`

Pour compléter, voir : `help("dgamma")`, `help("dt")`, `help("dchisq")` et `help("df")`.

Calculs

On fait :

```
dbinom(4, 8, 0.3)
```

Cela renvoie : `[1] 0.1361`

Ainsi, on a calculé la densité d'une *var* $X \sim \mathcal{B}(8, 0.3)$ en $x = 4$:

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

On peut vérifier cela en faisant :

```
choose(8, 4) * 0.3^4 *(1- 0.3)^(8 - 4)
```

Cela renvoie : `[1] 0.1361`

On fait :

```
dnorm(1.7, 2, 0.12)
```

Cela renvoie : `[1] 0.1460692`

Ainsi, on a calculé la densité d'une *var* $X \sim \mathcal{N}(2, 0.12^2)$ en $x = 1.7$:

$$f_X(1.7) = \frac{1}{\sqrt{2\pi \times 0.12^2}} e^{-\frac{(1.7-2)^2}{2\times 0.12^2}}$$

On peut vérifier cela en faisant :

```
(1 / sqrt(2 * pi * 0.12^2)) * exp(- (1.7 - 2)^2 / (2 * 0.12^2))
```

Cela renvoie : `[1] 0.1460692`

Pour calculer la densité en plusieurs valeurs, on prend pour `x` le vecteur ayant pour éléments ces valeurs. On peut faire de même avec un ensemble de paramètres et les arguments correspondants.

On fait :

```
dbinom(c(4, 6), 8, 0.3)
```

Cela renvoie : `[1] 0.13613670 0.01000188`

On a ainsi calculé la densité d'une *var* $X \sim \mathcal{B}(8, 0.3)$ pour $x \in \{4, 6\}$.

On fait :

```
dexp(2, c(1, 2, 3))
```

Cela renvoie : `[1] 0.135335283 0.036631278 0.007436257`

Ainsi, on a calculé la densité d'une *var* $X \sim \mathcal{E}(\lambda)$ en $x = 2$, avec $\lambda = 1$, $\lambda = 2$ et $\lambda = 3$.

On peut aussi mettre ces résultats dans un vecteur pour le réutiliser ultérieurement.

On fait :

```
vec = dexp(2, c(1, 2, 3))

vec
```

Cela renvoie : `[1] 0.135335283 0.036631278 0.007436257`

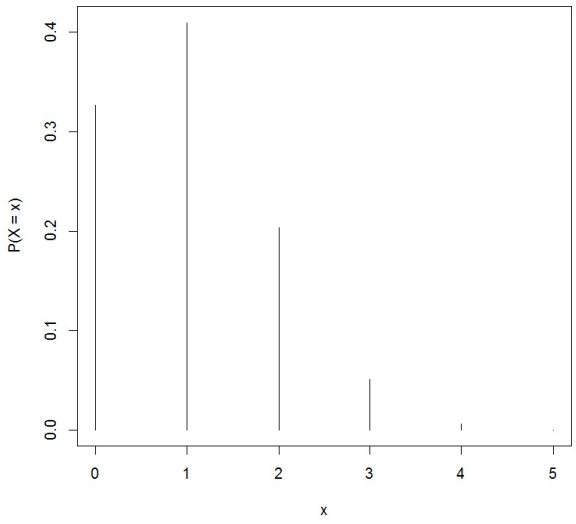
Représentation graphique

On peut représenter le graphe de la densité d'une *var* X discrète avec la commande `plot` et l'option `type = h`.

On fait :

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

Cela renvoie :



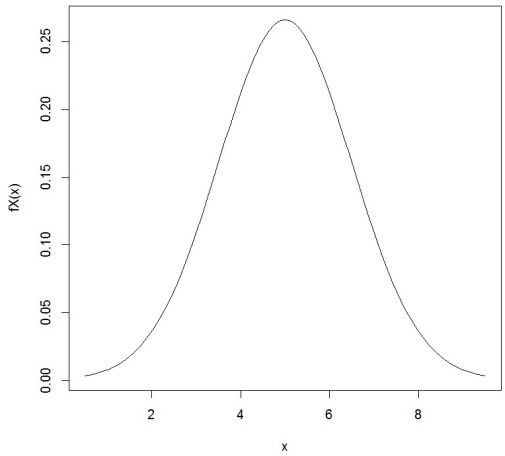
On a ainsi représenté le graphe de la densité d'une *var* $X \sim \mathcal{B}(5, 0.2)$.

On peut représenter le graphe de la densité d'une *var* X à densité avec la commande `curve`.

On fait :

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

Cela renvoie :



On a ainsi représenté le graphe de la densité d'une *var* $X \sim \mathcal{N}(5, 1.5^2)$.

3 Fonction de répartition

Définition

On appelle fonction de répartition d'une *var* X en x la fonction $F_X(x) = \mathbb{P}(X \leq x)$.

- Si X est discrète, on a

$$F_X(x) = \sum_{k \in X(\Omega) \cap]-\infty, x]} \mathbb{P}(X = k).$$

- Si X est à densité, on a

$$F_X(x) = \int_{-\infty}^{x} f_X(t)dt.$$

Commandes

Si la loi de X dépend d'un ou de plusieurs paramètres, disons `par1` et `par2`, alors la fonction de répartition de X en x est donnée par les commandes :

```
ploi(x, par1, par2)
```

On peut calculer : $\mathbb{P}(X > x) = 1 - F_X(x)$, en faisant :

```
ploi(x, par1, par2, lower.tail = FALSE)
```

Calculs

On fait :

```
pbinom(4, 8, 0.3)
```

Cela renvoie : `[1] 0.9420324`

Ainsi, on a calculé la fonction de répartition d'une *var* $X \sim \mathcal{B}(8, 0.3)$ en $x = 4$:

$$F_X(4) = \mathbb{P}(X \leq 4) = \sum_{k=0}^{4} \mathbb{P}(X = k).$$

On peut vérifier le calcul précédent en faisant :

```
sum(dbinom(0:4, 8, 0.3))
```

Cela renvoie : `[1] 0.9420324`

On fait :

```
pnorm(12, 9, 2)
```

Cela renvoie : `[1] 0.9331928`

Ainsi, on a calculé la fonction de répartition d'une *var* $X \sim \mathcal{N}(9, 2^2)$ en $x = 12$:

$$F_X(12) = \mathbb{P}(X \leq 12) = \int_{-\infty}^{12} \frac{1}{\sqrt{2\pi \times 2^2}} e^{-\frac{(t-9)^2}{2\times 2^2}} dt.$$

On fait :

```
pexp(2, 3, lower.tail = FALSE)
```

Cela renvoie : `[1] 0.002478752`

Ainsi, on a calculé : $\mathbb{P}(X > 2)$, $X \sim \mathcal{E}(3)$:

$$\mathbb{P}(X > 2) = \int_2^{\infty} f_X(x)dx = \int_2^{\infty} 3e^{-3x}dx = \left[-e^{-3x}\right]_2^{\infty} = e^{-3\times 2} = e^{-6}.$$

On peut vérifier le calcul précédent en faisant :

```
exp(-6)
```

Cela renvoie : `[1] 0.002478752`

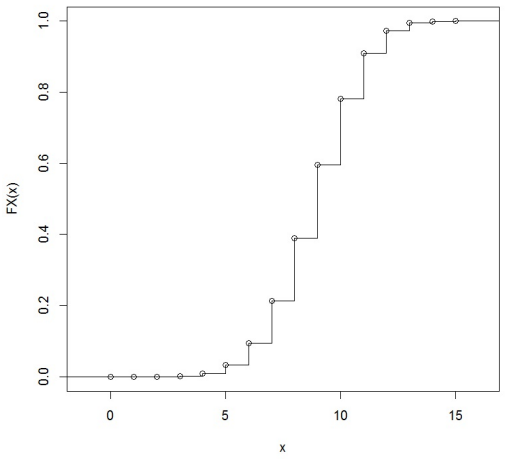
Représentation graphique

On peut représenter le graphe de la fonction de répartition d'une *var* X discrète avec la commande `stepfun`.

On fait :

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

Cela renvoie :



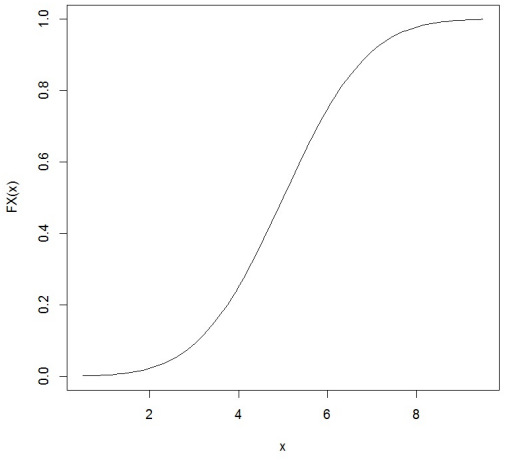
On a ainsi représenté le graphe de la fonction de répartition d'une *var* $X \sim \mathcal{B}(15, 0.6)$.

On peut représenter le graphe de la fonction de répartition d'une *var* X à densité avec la commande `curve`.

On fait :

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

Cela renvoie :



On a ainsi représenté le graphe de la fonction de répartition d'une *var* $X \sim \mathcal{N}(5, 1.5^2)$.

4 Quantiles

Définition

Soit $p \in]0,1[$ et X une *var*.

- Si X est discrète, on appelle p-ème quantile de X l'entier x_p défini par

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

- Si X est à densité, on appelle p-ème quantile de X le réel x_p tel que $F_X(x_p) = p$.

Commandes

Si la loi de X dépend d'un ou de plusieurs paramètres, disons `par1` et `par2`, alors le p-ème quantile de X est donné par les commandes :

```
qloi(p, par1, par2)
```

Calculs

On fait :

```
qbinom(0.25, 5, 0.6)
```

Cela renvoie : `[1] 2`

Ainsi, on a calcule le p-ème quantile avec $p = 0.25$ d'une *var* $X \sim \mathcal{B}(5, 0.6)$:

$x_{0.25} = \inf\{k \in \mathbb{Z};\ F_X(k) \geq 0.25\}$.

On peut vérifier ce résultat en cherchant toutes les valeurs de $F_X(x)$ pour $x \in \{0,1,2,3,4,5\}$:

```
pbinom(0:5, 5, 0.6)
```

Cela renvoie : `[1] 0.01024 0.08704 0.31744 0.66304 0.92224 1.00000`

On constate alors que $F_X(1) = 0.08704 < 0.25 \leq 0.31744 = F_X(2)$, donc $x_{0.25} = 2$.

On fait :

```
qnorm(0.975, 0, 1)
```

Cela renvoie : `[1] 1.96`

Ainsi, on a calcule le p-ème quantile avec $p = 0.975$ d'une *var* $X \sim \mathcal{N}(0, 1)$: le réel $x_{0.975}$ tel que $F_X(x_{0.975}) = 0.975$.

On peut vérifier cela en faisant :

```
pnorm(1.96, 0, 1)
```

Cela renvoie : `[1] 0.975`

On aurait aussi pu considérer les commandes : `pnorm(1.96)`, sans spécifier $\mu = 0$ et $\sigma = 1$; la commande prend par défaut les paramètres de la loi centrée et réduite.

5 Simulation

5.1 La commande `sample`

Utilisation de base

L'utilisation de base est : `sample(x)`, où `x` est un vecteur numérique, logique ou chaîne de caractères.

Cela renvoie un vecteur dont les éléments correspondent au tirage d'un élément de `x` sans remise. Cela revient à permuter aléatoirement les éléments de `x`.

On fait :

```
x = 1:7

sample(x)
```

Cela renvoie : `[1] 2 4 1 7 5 6 3`

On refait la même commande :

```
sample(x)
```

Cela renvoie : `[1] 1 7 6 2 3 4 5`

On obtient des résultats différents ; l'ordre de la permutation est due au hasard à chaque fois.

On peut conserver le résultat d'une permutation aléatoire dans un vecteur pour le réutiliser ultérieurement :

```
y = sample(c("rouge", "vert", "bleu", "blanc", "noir"))

y
```

Cela renvoie : `[1] "noir" "blanc" "vert" "rouge" "bleu"`

Options

Il existe des options dans `sample` permettant de modifier le type de tirage. On les active en rajoutant une ou plusieurs commandes dans `sample`.

Par exemple, on fait :

```
sample(1:3, size = 2, replace = TRUE, prob = c(25 / 100, 20 / 100,
55 / 100))
```

Quelques options sont présentées ci-dessous.

Option : `size`. Les commandes : `size = "n"`, où `n` est un entier, précise le nombre de fois `n` que l'on effectuera le tirage. Par défaut, le nombre de tirage est égal à la longueur de `x`.

On fait :

```
sample(1:10, size = 3)
```

Cela renvoie : `[1] 10 7 3`

Dans la syntaxe de `sample`, l'argument `size` est en deuxième position. On peut donc se contenter de : `sample(1:10, 3)`.

Option : `replace`. Les commandes : `replace = L`, où `L` est `TRUE` ou `FALSE`, dont que les tirages sont avec remise si `L = TRUE` et sans remise sinon.

On fait :

```
sample(1:5, size = 9, replace = TRUE)
```

Cela renvoie : `[1] 5 1 4 1 3 1 2 1 3`

Option : `prob`. La commande `prob` permet la simulation de *var* dont le support est l'ensemble des éléments de `x`, que l'on suppose au nombre de k. Ainsi, la loi commune de ces *var* est :

$\mathbb{P}(X = j\text{-ème élément de } \texttt{x}) = p_j$, avec $j \in \{1, \dots, k\}$, $p_j \in [0, 1]$ et $\sum_{j=1}^{k} p_j = 1$.

On ajoute donc `prob = vec`, où `vec` est le vecteur des probabilités $(p_1, \dots, p_k)$.

Si `prob` n'est pas indiqué, par défaut $p_j = 1/k$ (tirage équiprobable).

On fait :

```
y = c("rouge", "vert", "bleu", "blanc", "noir")
sample(y, 2, prob = c(10 / 100, 30 / 100, 10 / 100, 30 / 100, 20 / 100))
```

Cela renvoie : `[1] "rouge" "vert"`

5.2 Simulation par des lois préprogrammées

Commandes

Si la loi de X dépend d'un ou de plusieurs paramètres, disons `par1` et `par2`, alors on simule des réalisations de `n` *var* indépendantes suivant la même que X par les commandes :

```
rloi(n, par1, par2)
```

Calculs

On fait :

```
rpois(10, 2)
```

Cela renvoie : `[1] 0 2 2 3 3 5 1 3 1 2`

Ainsi, on a simulé des réalisations de 10 *var* indépendantes suivant chacune la loi de Poisson $\mathcal{P}(2)$.

On fait :

```
sum(rbinom(80, 1, 0.02))
```

Cela renvoie : `[1] 3`

On a alors simulé une réalisation de la *var* $\sum_{i=1}^{80} X_i$, où $X_1, \ldots, X_{80}$ sont des *var* indépendantes suivant chacune la loi binomiale $\mathcal{B}(1, 0.02)$, (ainsi, la loi commune est la loi de Bernoulli $\mathcal{B}(0.02)$: $\mathbb{P}(X_1 = 0) = 1 - 0.02 = 0.98$, $\mathbb{P}(X_1 = 1) = 0.02$). Comme $\sum_{i=1}^{80} X_i \sim \mathcal{B}(80, 0.02)$, cela revient à simulé une réalisation d'une *var* $Y \sim \mathcal{B}(80, 0.02)$: `rbinom(1, 80, 0.02)`.

On fait :

```
x = rnorm(15, 22, 2)

x
```

Cela renvoie :

```
[1] 16.29346 20.03768 23.64521 19.80074 21.79431 23.40846 20.32446 25.50194
[9] 23.25963 21.64157 21.97450 23.91137 21.44780 23.14619 22.82958
```

Ainsi, on a construit un vecteur dont les éléments sont des réalisations de 15 *var* indépendantes suivant chacune la loi normale $\mathcal{N}(22, 2^2)$.

6 Exercices

Exercice 1. Représenter le graphe de la densité d'une *var* $X \sim \mathcal{P}(1)$ pour $x \in \{0, \ldots, 8\}$.

Exercice 2. Représenter le graphe de la densité d'une *var* $X \sim \chi^2(3)$ pour $x \in [0, 10]$.

Exercice 3. Diviser la fenêtre de l'écran en deux fenêtres lignes.

- Représenter, dans la première fenêtre, le graphe de la densité d'une *var* $X \sim \mathcal{B}(50, 0.08)$ en prenant `ylim = c(0, 0.25)`.
- Représenter dans la deuxième fenêtre, au-dessous de la première, le graphe de la densité d'une *var* $Y \sim \mathcal{P}(4)$ pour $x \in \{0, \ldots, 50\}$ avec la même option : `ylim = c(0, 0.25)`.

(Ceci illustrera le fait que lorsque $n \geq 31$ *et* $np \leq 10$ *on peut approcher la loi binomiale* $\mathcal{B}(n, p)$ *par la loi de Poisson* $\mathcal{P}(np)$*).*

Exercice 4. Séparer l'écran graphique en 3 (1 ligne, 3 colonnes).

- Dans la première fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(4, 1)$, puis ajouter dans la même fenêtre, avec une autre couleur, celui de la densité d'une *var* $Y \sim \mathcal{N}(5, 1)$.
- Dans la deuxième fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(4, 1)$ puis ajouter dans la même fenêtre, avec une autre couleur, celui de la densité d'une *var* $Y \sim \mathcal{N}(4, 4)$.
- Dans la troisième fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(4, 1)$ puis ajouter dans la même fenêtre, avec une autre couleur, celui de la densité d'une *var* $Y \sim \mathcal{N}(5, 4)$.

Exercice 5. Représenter le graphe de la densité d'une *var* $X \sim \mathcal{B}(50, 0.4)$, puis ajouter par dessus ce graphe celui de la densité d'une *var* $Y \sim \mathcal{N}(20, 12)$ *(cela illustrera le fait que, lorsque* $n \geq 31$*,* $np \geq 5$ *et* $n(1-p) \geq 5$*, on peut approcher la loi binomiale* $\mathcal{B}(n, p)$ *par la loi normale* $\mathcal{N}(np, np(1-p))$*).*

Exercice 6. Représenter le graphe de la densité d'une *var* $X \sim \mathcal{B}(n,p)$ pour $x \in \{0,\dots,10\}$ avec $n = 100$ et $p = 0.01$, puis ajouter dans la même fenêtre le graphe de la densité d'une *var* $Y \sim \mathcal{N}(np, np(1-p))$. Que constatez-vous ?

Exercice 7. Soit $X \sim \mathcal{N}(0,1)$. Calculer les probabilités : $\mathbb{P}(X < -0.5)$, $\mathbb{P}(X > 1.5)$, $\mathbb{P}(X > -1)$, $\mathbb{P}(|X| \leq 1.96)$, $\mathbb{P}(|X| \leq 2.58)$ et $\mathbb{P}(|X| \geq 3)$.

Exercice 8. Soit $X \sim \mathcal{N}(15,9)$.

1. Calculer les probabilités : $\mathbb{P}(16 \leq X \leq 20)$, $\mathbb{P}(X > 18)$, $\mathbb{P}(X < 6)$ et $\mathbb{P}(|X - 15| > 5.88)$.
2. Représenter le graphe de la fonction de répartition de X pour $x \in [6,24]$.

Exercice 9. Représenter, sur une même figure, la fonction de répartition d'une *var* $X \sim \mathcal{B}(50, 0.4)$ et celle d'une v̂ar $Y \sim \mathcal{N}(20,12)$.

Exercice 10.

1. Soit $X \sim \mathcal{N}(0,1)$. Calculer les quantiles : $x_{0.00135}$, $x_{0.025}$, $x_{0.95}$, $x_{0.999}$, $x_{0.995}$ et $x_{0.99865}$.
2. Soit $Y \sim \mathcal{N}(19,3)$. Calculer les quantiles : $y_{0.975}$ et $y_{0.025}$. Vérifier que $y_{0.975} = 19 + \sqrt{3}\, x_{0.975}$.

Exercice 11. Construire un vecteur `simul1` composé de 1000 éléments dont les valeurs sont des réalisations d'une *var* $X \sim \mathcal{N}(15,3)$. Faire l'histogramme des fréquences de `simul1` et superposer la densité de X. Faire le boxplot de `simul1`.

Exercice 12.

1. Construire un vecteur `simul2` qui est composé de 1000 éléments dont les valeurs sont des réalisations d'une *var* $X \sim \mathcal{P}(1)$.
2. Construire un vecteur dont le premier élément sera le nombre de réalisations qui vaudront 0, le deuxième élément sera le nombre de réalisations qui vaudront 1... *(On pourra utiliser la fonction* `compter` *décrite ci-dessous.*

```
compter = function(a, b) {
  d = numeric()
  for(i in 1:(length(a))) {
    d[i] = sum(b == a[i])
  }
  names(d) = as.character(a)
  d
}
```

3. Faire un `barplot` et superposer la densité de X.

Exercice 13. Construire dans R une fonction `péage` qui a 4 arguments : `mu1`, `mu2`, `sigma` et `n`. À l'intérieur de cette fonction :

- on définit un vecteur numérique `attente` de longueur n,
- on construit un objet `cabines` constitué des cabines `C1` et `C2`,
- on ouvre une boucle qui va de 1 à n. Pour chaque i avec $i \in \{1, \ldots, n\}$, à l'intérieur de cette boucle, on tire au hasard une cabine dans l'objet `cabines`.
  - Si la cabine `C1` a été choisie, on simule le tirage d'une *var* d'espérance `mu1` et d'écart-type `sigma` et on affecte le résultat à la i-ème composante de `attente`.
  - Si la cabine `C2` a été choisie, on simule le tirage d'une $\widehat{\text{var}}$ de moyenne `mu2` et d'écart-type `sigma` et on affecte le résultat à la i-ème composante de `attente`,
- après avoir constitué le vecteur `attente`, on construit l'histogramme des fréquences du vecteur `attente`,
- on calcule `moy` qui est la moyenne du vecteur `attente`,
- on calcule `ecart` qui est l'écart-type corrigé des valeurs des éléments du vecteur `attente` en utilisant la fonction **sd**,
- on ajoute à l'histogramme la courbe de la densité associée à la loi normale de moyenne `moy` et d'écart-type `ecart`,
- on ajoute à la figure un texte comportant l'indication de la valeur de `moy` et l'indication de la valeur de `ecart` (garder deux décimales seulement),

- on ajoute à l'histogramme le graphe de la fonction suivante :

$$f(x) = 0.5(f_1(x) + f_2(x)),$$

où $f_1(x)$ est la densité associée à la loi $\mathcal{N}(\mu_1, \sigma^2)$ et $f_2(x)$ celle associée à la loi $\mathcal{N}(\mu_2, \sigma^2)$.

On exécute ensuite cette fonction sur deux fenêtres :

- pour la première prendre `mu1 = 50`, `mu2 = 51`, `sigma = 2` et `n = 1000`,
- pour la seconde prendre `mu1 = 50`, `mu2 = 60`, `sigma = 2` et `n = 1000`.

L'objectif est de montrer qu'un mélange de lois normales n'est pas obligatoirement une loi normale.

Exercice 14. Diviser l'écran en 4 fenêtres.

- Dans la première fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(0, 1)$ pour $x \in [-3, 3]$, puis ajouter celui de la densité d'une *var* $Y \sim \mathcal{T}(2)$.
- Dans la deuxième fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(0, 1)$ pour $x \in [-3, 3]$, puis ajouter celui de la densité d'une *var* $Y \sim \mathcal{T}(3)$.
- Dans la troisième fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(0, 1)$ pour $x \in [-3, 3]$, puis ajouter celui de la densité d'une *var* $Y \sim \mathcal{T}(10)$.
- Dans la quatrième fenêtre, représenter le graphe de la densité d'une *var* $X \sim \mathcal{N}(0, 1)$ pour $x \in [-3, 3]$, puis ajouter celui de la densité d'une *var* $Y \sim \mathcal{T}(30)$.

Exercice 15. Construire dans R une fonction `stu1` qui a 4 arguments : `n`, `nb`, `mu` et `sigma`.

À l'intérieur de cette fonction, on définit un vecteur `vec` de longueur `nb`.

On construit les composantes du vecteur par une boucle allant de 1 à `nb`.

La i-ème composante de `vec` est calculée ainsi :

- On simule un vecteur `simul` qui est composé de `n` réalisations d'une *var* $X \sim \mathcal{N}(\texttt{mu}, \texttt{sigma}^2)$.
- On calcule la moyenne $\overline{x}$ et l'écart-type corrigé s des valeurs des éléments de `simul`.
  La i-ème composante de `vec` est égale à $\sqrt{\texttt{n}}\left(\dfrac{\overline{x} - \texttt{mu}}{s}\right)$.
- Diviser l'écran en 2 fenêtres superposées.

- Dans la première fenêtre, construire l'histogramme des fréquences de `vec`. Superposer le graphe de la densité associée à loi $\mathcal{N}(0,1)$ et, avec une autre couleur, le graphe de la densité associée à la loi $\mathcal{T}(\mathtt{n}-1)$.
- Dans la deuxième fenêtre, représenter la fonction de répartition empirique de `vec` avec les commandes : `plot(ecdf(vec))`. Ajouter en rouge le graphe de la fonction de répartition associée à la loi $\mathcal{T}(\mathtt{n}-1)$. Ajouter en jaune le graphe de la fonction de répartition associée à la loi $\mathcal{N}(0,1)$.

Essayer cette fonction avec `n=3`, `nb = 1000`, `mu = 10` et `sigma = 2`, puis avec `n=30`, `nb = 1000`, `mu = 10` et `sigma = 2`.

Rappel : La fonction de répartition empirique associée à un n-échantillon $(X_1,\dots,X_n)$ d'une var X de fonction de répartition F_X est $\widehat{F}(x)=\dfrac{1}{n}\displaystyle\sum_{i=1}^{n}\mathbf{1}_{\{X_i\leq x\}}$. Le théorème de Glivenko-Cantelli nous assure que : $\max\limits_{-\infty<x<+\infty}|\widehat{F}(x)-F_X(x)|\to 0$ (presque sûrement) quand $n\to+\infty$.

Exercice 16. Diviser l'écran en 6 fenêtres.

- Dans la fenêtre 1, représenter le graphe de la densité d'une *var* $X\sim\chi^2(1)$ pour $x\in[0.01,10]$.
- Dans la fenêtre 2, représenter le graphe de la densité d'une *var* $X\sim\chi^2(2)$ pour $x\in[0.01,10]$.
- Dans la fenêtre 3, représenter le graphe de la densité d'une *var* $X\sim\chi^2(3)$ pour $x\in[0.01,10]$.
- Dans la fenêtre 4, représenter le graphe de la densité d'une *var* $X\sim\chi^2(5)$ pour $x\in[0.01,10]$.
- Dans la fenêtre 5, représenter le graphe de la densité d'une *var* $X\sim\chi^2(10)$ pour $x\in[0.01,20]$.
- Dans la fenêtre 6, représenter le graphe de la densité d'une *var* $X\sim\chi^2(20)$ pour $x\in[0.01,60]$.

Exercice 17. Construire dans R une fonction `khi2` qui a 4 arguments : `n`, `nb`, `mu` et `sigma`.

À l'intérieur de cette fonction, on définit un vecteur `vec` de longueur `nb`.

On construit les composantes de ce vecteur par une boucle allant de 1 à `nb`.

La i-ème composante de `vec` est calculée ainsi :

- On simule un vecteur `simul` qui est composé de `n` réalisations d'une *var* $X\sim\mathcal{N}(\mathtt{mu},\mathtt{sigma}^2)$.
- On calcule la variance corrigée s^2 de la série des `n` composantes de `simul`.

La i-ème composante de `vec` est égale à $\dfrac{(\mathtt{n}-1)s^2}{\mathtt{sigma}^2}$.

Une fois ce vecteur construit, on ne garde que les composantes comprises entre 0.01 et 10 et on construit l'histogramme des fréquences.

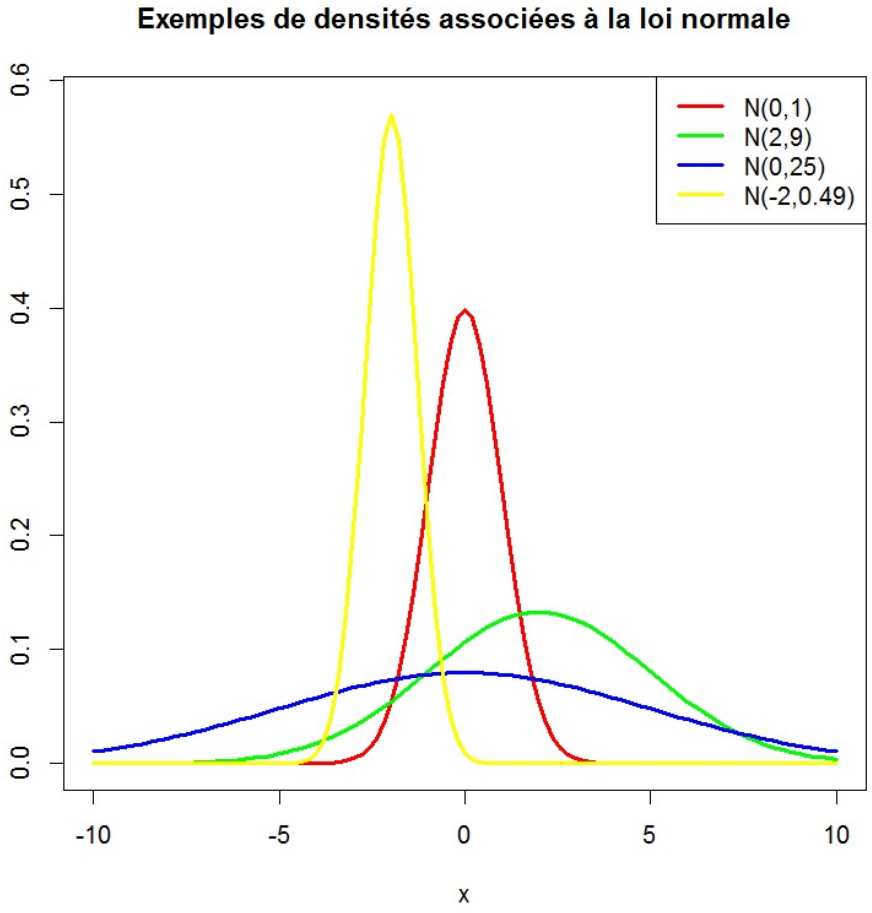
On superpose le graphe de la densité associée à la loi $\chi^2(n-1)$.

Essayer cette fonction avec `n = 4`, `nb = 10000`, `mu = 10` et `sigma = 2`.

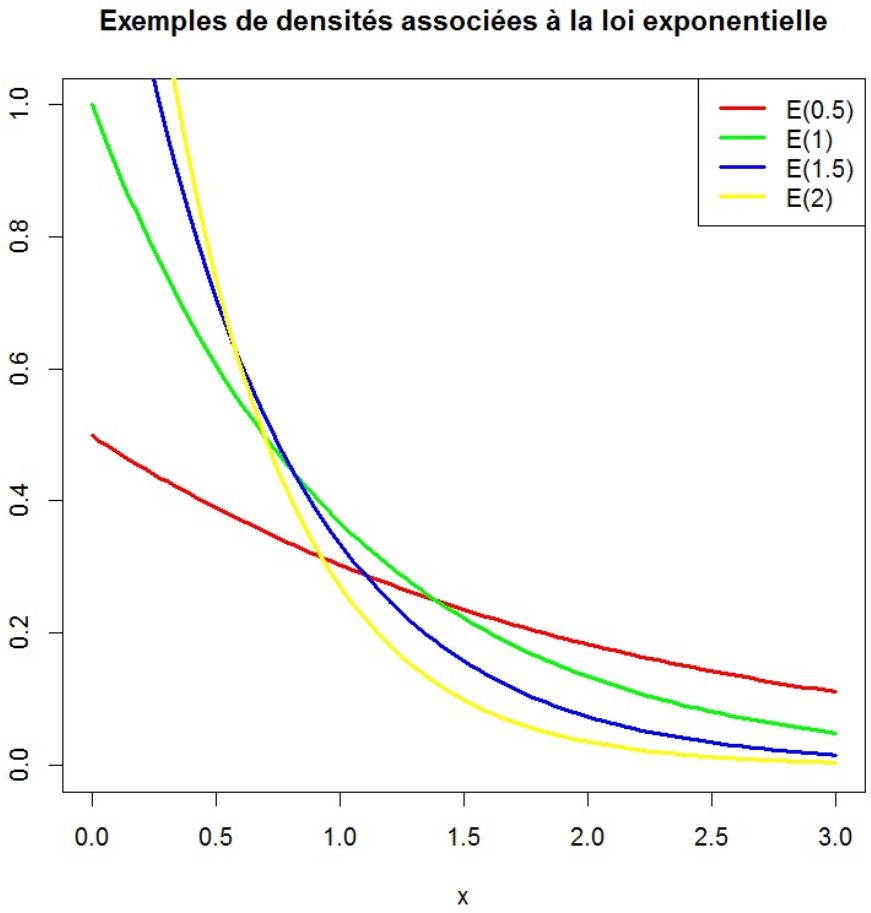
Exercice 18. Diviser l'écran en 4 fenêtres.

- Dans la fenêtre 1, représenter le graphe de la densité d'une *var* $X \sim \mathcal{E}(1)$ pour $x \in [0, 3]$.
- Dans la fenêtre 2, , représenter le graphe de la densité d'une *var* $X \sim \mathcal{E}(2)$ pour $x \in [0, 3]$.
- Dans la fenêtre 3, représenter le graphe de la densité d'une *var* $X \sim \mathcal{E}(0.5)$ pour $x \in [0, 20]$.
- Dans la fenêtre 4, représenter le graphe de la densité d'une *var* $X \sim \mathcal{E}(0.1)$ pour $x \in [0, 60]$.

Exercice 19. Proposer des commandes R permettant d'obtenir le graphique suivant :



Exercice 20. Proposer des commandes R permettant d'obtenir le graphique suivant :



Exercice 21. On s'interesse à la durée de vie d'un certain type de voiture. Soit X la *var* égale à la durée de vie en années d'une de ces voitures. On suppose que X suit la loi exponentielle $\mathcal{E}(\frac{1}{10})$, *i.e.* de densité

$$f(x) = \begin{cases} \frac{1}{10} e^{-\frac{x}{10}} & \text{si } x \geq 0, \\ 0 & \text{sinon.} \end{cases}$$

1. Écrire une nouvelle fonction R équivalente à `dexp` *(on pourra utiliser une structure de contrôle* `if ...else` *pour que celle-ci valle* 0 *quand* $x < 0$*).*
2. Vérifier numériquement que $\int_0^{100} f(x)dx \simeq 1$ avec la commande `integrate`.

3. Séparer l'écran graphique en 2 : dans la fenêtre 1, représenter le graphe de la densité f et, dans la fenêtre 2, celui de la fonction de répartition de X.
4. Calculer la probabilité que la durée de vie d'une voiture dépasse 10.2 ans.

Exercice 22. Soit X une *var* suivant la loi gamma $\Gamma(5,1)$, *i.e.* de densité

$$f(x) = \begin{cases} \frac{1}{24}x^4 e^{-x} & \text{si } x \geq 0, \\ 0 & \text{sinon.} \end{cases}$$

1. Écrire une nouvelle fonction R équivalente à `dgamma`.
2. Vérifier numériquement que $\int_0^{100} f(x)dx \simeq 1$.
3. Évaluer la fonction de répartition de X pour $x \in \{2, \ldots, 10\}$.
4. Déterminer le réel x vérifiant $\mathbb{P}(X \leq x) = 0.92$.

Exercice 23. Soit X une *var* suivant la loi normale $\mathcal{N}(0,1)$, *i.e.* de densité

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \qquad x \in \mathbb{R}.$$

1. Écrire une nouvelle fonction R équivalente à la fonction `dnorm`.
2. Vérifier numériquement que $\int_{-100}^{100} f(x)dx \simeq 1$.
3. Séparer l'écran graphique en 2 : dans la fenêtre 1, représenter le graphe de la densité f et, dans la fenêtre 2, celui de la fonction de répartition de X.
4. Calculer $\mathbb{P}(X \leq 2.2)$, $\mathbb{P}(X \geq 1.7)$, $\mathbb{P}(0.2 \leq X < 1.4)$ et $\mathbb{P}(|X| \leq 1.96)$.
5. Déterminer le réel x vérifiant $\mathbb{P}(X \leq x) = 0.98$.

Exercice 24. On suppose que le poids d'un foie gras peut être modélisé par une *var* X suivant la loi normale $\mathcal{N}(550, 100^2)$, l'unité étant le gramme. Quelle est la probabilité qu'un foie gras pèse

- moins de 650 grammes ?
- plus de 746 grammes ?
- entre 550 grammes et 600 grammes ?

Exercice 25. Soit X une *var* suivant la loi normale $\mathcal{N}(0,1)$. La fonction de répartition de X, *i.e.* $F_X(x) = \mathbb{P}(X \leq x)$, avec $x \geq 0$, peut s'écrire comme

$$F_X(x) = \frac{1}{2} + \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \sum_{n=0}^{\infty} \frac{x^{2n+1}}{\prod_{m=0}^{n}(2m+1)}.$$

Utiliser cette expression pour écrire une nouvelle fonction R équivalente à la fonction `pnorm` *(on tronquera la somme infinie à la valeur* 50*)*.

Exercice 26. 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.

Exercice 27. Une urne contient $p + q$ boules, dont p rouges et q noires. Construire dans R une fonction `Urne` qui a 3 arguments : `k`, `p` et `q`. L'enjeu de celle-ci est de modéliser 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" "Noire" "Rouge"`.

Exercice 28. 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.

Construire dans R une fonction `Freq` qui a un argument : `n`. L'enjeu de celle-ci est de renvoyer la fréquence de 5 lors de `n` tirages indépendants d'un dé cubique équilibré. Comparer les fréquences pour `n` $\in \{10, 100, 1000\}$, avec la probabilité (théorique) d'obtenir un 5 lorsqu'on lance un dé.

Exercice 29. 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.
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`.
   (b) Comparer, suivant le type d'épreuve, le diagramme en barre des fréquences observées avec la distribution des lois binomiales et hypergéométriques correspondantes.

Exercice 30. 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 vecteur de *var* $(X_0, X_1, \ldots, X_n)$ *(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`*).*

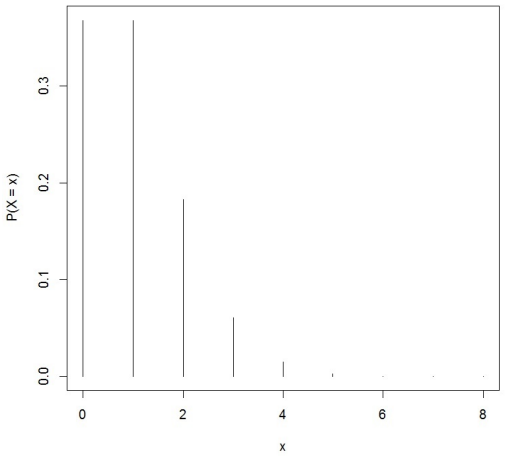
Exercice 31. On lance 5 dés cubiques équilibrés, puis on relance ceux qui n'ont pas fait 6, et ainsi de suite jusqu'à ce que les 5 dés affichent 6.

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*).*

7 Solutions

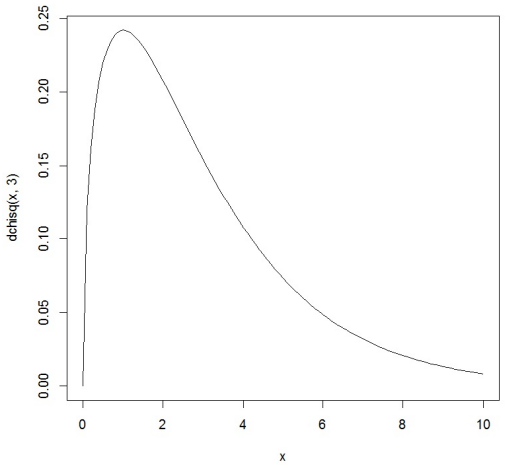
Solution 1. On fait : `plot(0:8, dpois(0:8, 1), type = "h", xlab = "x", ylab = "P(X = x)")`

Cela renvoie :



Solution 2. On considère les commandes : `curve(dchisq(x, 3), 0, 10, xlab = "x")`

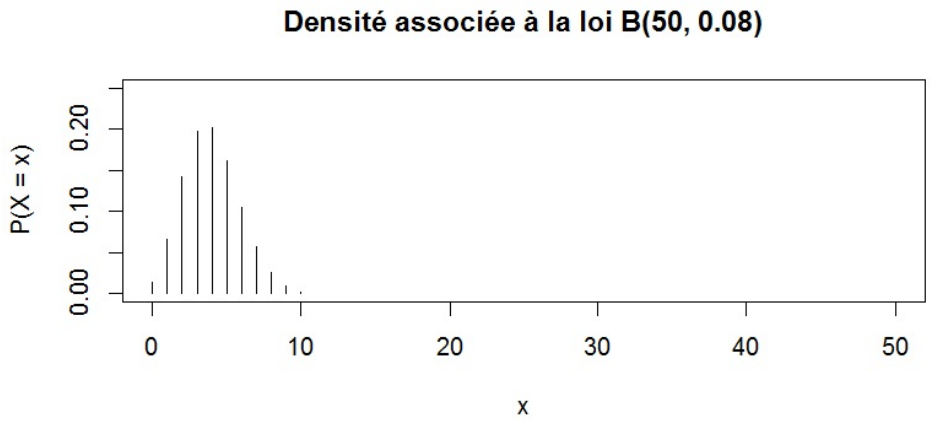
Cela renvoie :

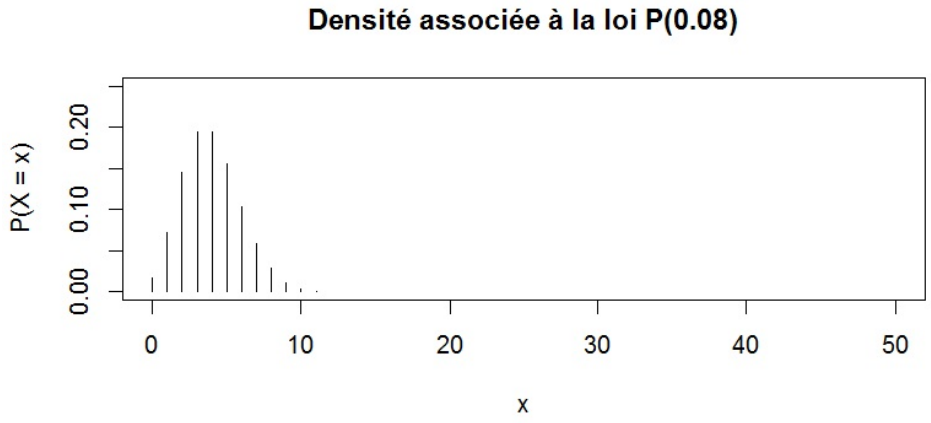


Solution 3. On fait :

```
par(mfrow = c(2,1))
plot(0:50, dbinom(0:50, 50, 0.08), type = "h", xlab = "x", ylab = "P(X = x)",
     ylim=c(0, 0.25), main = "Densité associée à la loi B(50, 0.08)")
plot(0:50, dpois(0:50, 4), type = "h", xlab = "x", ylab = "P(X = x)",
     ylim = c(0, 0.25), main = "Densité associée à la loi P(0.08)")
       par(mfrow = c(1,1))
```

Cela renvoie :

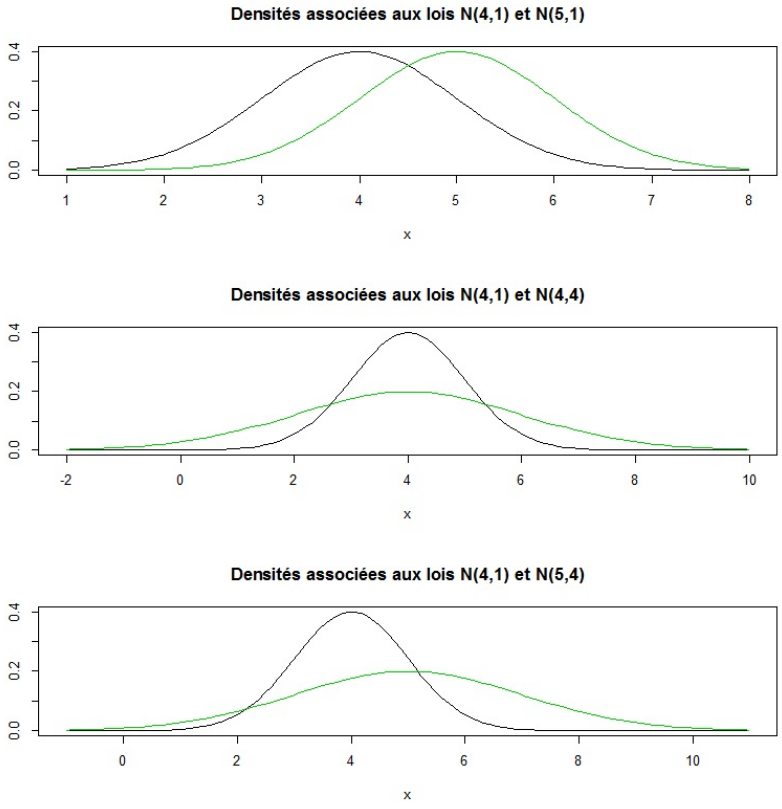




Solution 4. On fait :

```
par(mfrow = c(3, 1))
curve(dnorm(x, 4, 1), 1, 8, ylab = "",
main = "Densités associées aux lois N(4,1) et N(5,1)")
curve(dnorm(x, 5, 1), 1, 8, col = 3, ylab = "", add = TRUE)
curve(dnorm(x, 4, 1), -2, 10, ylab = "",
main = "Densités associées aux lois N(4,1) et N(4,4)")
curve(dnorm(x, 4, 2), -2, 10, col = 3,  ylab = "", add = TRUE)
curve(dnorm(x, 4, 1), -1, 11, ylab = "",
main = "Densités associées aux lois N(4,1) et N(5,4)")
curve(dnorm(x, 5, 2),-1, 11, col = 3, ylab = "", add = TRUE)
  par(mfrow = c(1,1))
```

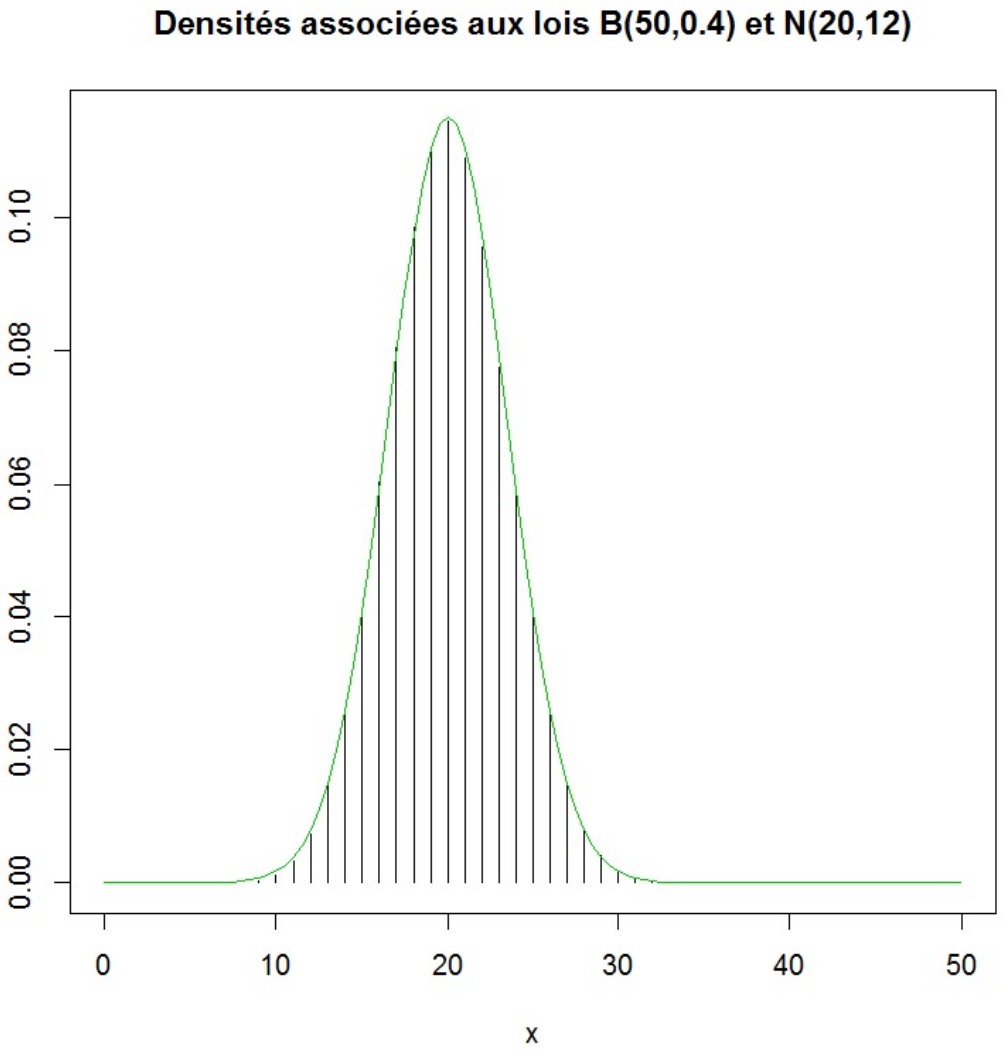
Cela renvoie :



Solution 5. On fait :

```
plot(0:50, dbinom(0:50, 50, 0.4), type = "h", xlab = "x", ylab = "",
main = "Densités associées aux lois B(50,0.4) et N(20,12)")
curve(dnorm(x, 20, sqrt(12)), 0, 50, col = 3,  ylab = "", add = TRUE)
```

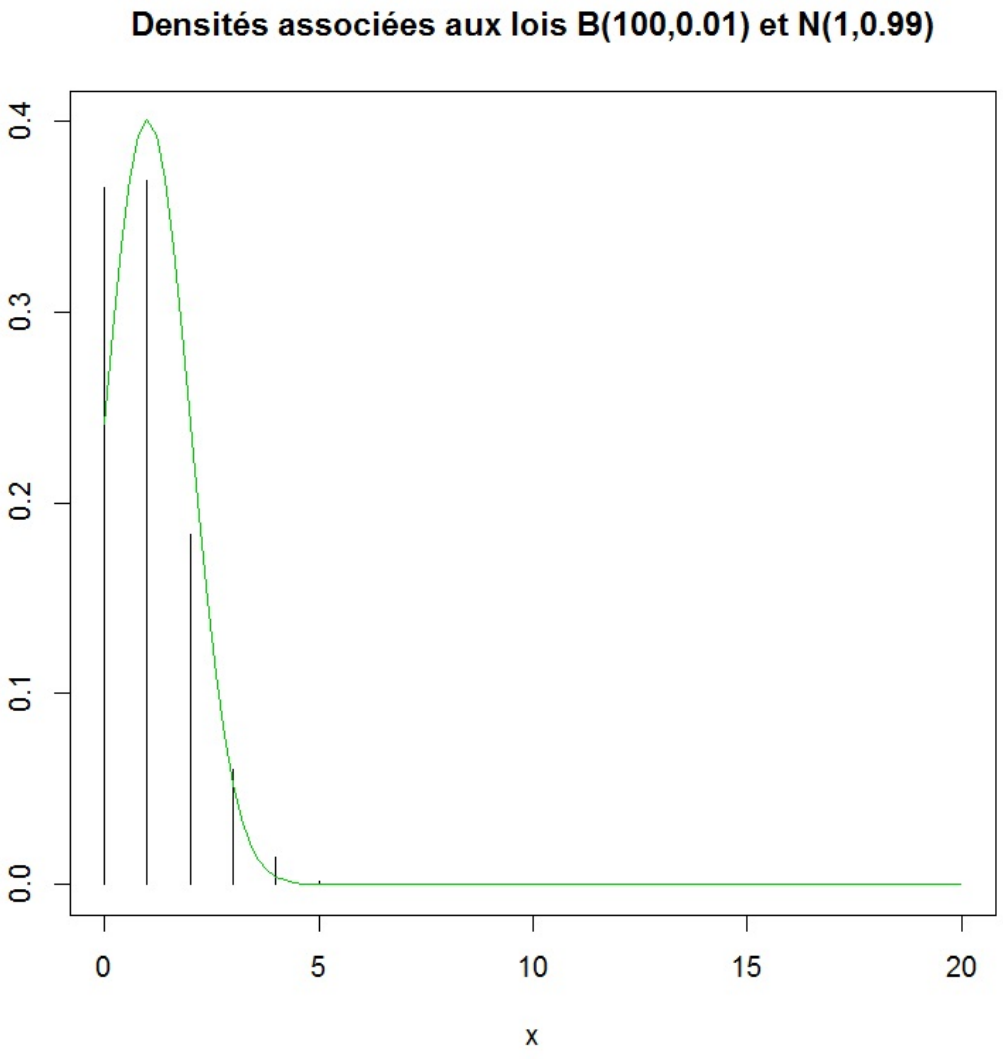
Cela renvoie :



Solution 6. On considère les commandes :

```
plot(0:20, dbinom(0:20, 100, 0.01), type = "h", xlab = "x", ylab = "",
ylim = c(0, 0.4), main = "Densités associées aux lois B(100,0.01) et N(1,0.99)")
curve(dnorm(x, 1, sqrt(0.99)), 0, 20, col = 3,  ylab = "", add = TRUE)
```

Cela renvoie :



Solution 7. Pour $X \sim \mathcal{N}(0,1)$, on a :

$\mathbb{P}(X < -0.5)$	$\mathbb{P}(X > 1.5)$	$\mathbb{P}(X > -1)$
`pnorm(-0.5, 0, 1)`	`1 - pnorm(1.5, 0, 1)`	`1 - pnorm(-1, 0, 1)`
`[1] 0.3085375`	`[1] 0.0668072`	`[1] 0.8413447`

$\mathbb{P}(\lvert X\rvert \leq 1.96)$	$\mathbb{P}(\lvert X\rvert \leq 2.58)$
`pnorm(1.96, 0, 1) - pnorm(-1.96, 0, 1)`	`pnorm(2.58, 0, 1) - pnorm(-2.58, 0, 1)`
`[1] 0.9500042`	`[1] 0.99012`

$\mathbb{P}(\lvert X\rvert \geq 3)$
`1 - (pnorm(3, 0, 1) - pnorm(-3, 0, 1))`
`[1] 0.002699796`

On aurait pu utiliser d'autres commandes. Par exemple, pour $\mathbb{P}(X > 1.5)$:

```
pnorm(1.5, 0, 1, lower.tail = FALSE)
```

Solution 8.

1. Pour $X \sim \mathcal{N}(15,9)$, on a :

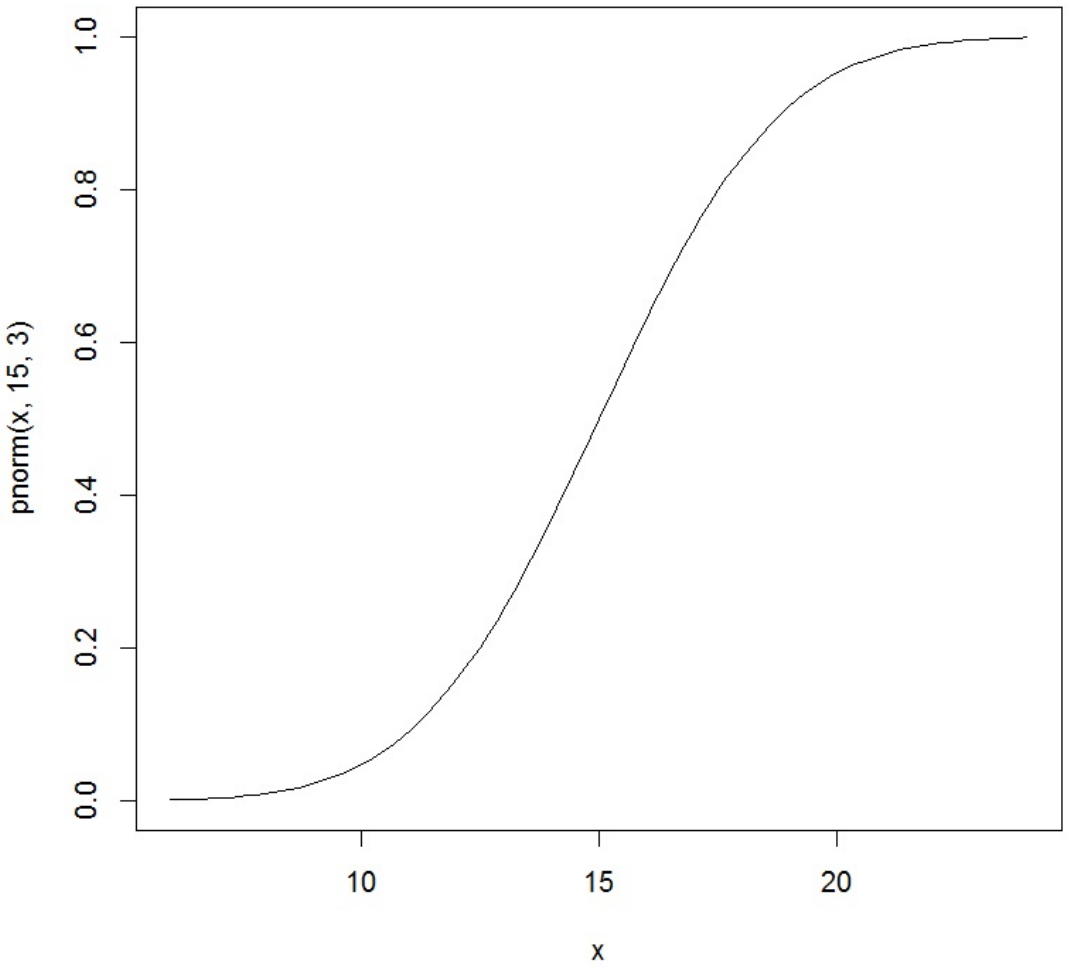
$\mathbb{P}(16 < X < 20)$	$\mathbb{P}(X > 18)$	$\mathbb{P}(X < 6)$
`pnorm(20, 15, 3) - pnorm(16, 15, 3)`	`1 - pnorm(18, 15, 3)`	`pnorm(6, 15, 3)`
`[1] 0.321651`	`[1] 0.1586553`	`[1] 0.001349898`

$\mathbb{P}(\lvert X - 15 \rvert > 5.88)$
`1 - (pnorm(15 + 5.88, 15, 3) - pnorm(15 - 5.88, 15, 3))`
`[1] 0.04999579`

2. On fait :

```
curve(pnorm(x, 15, 3), 6, 24)
```

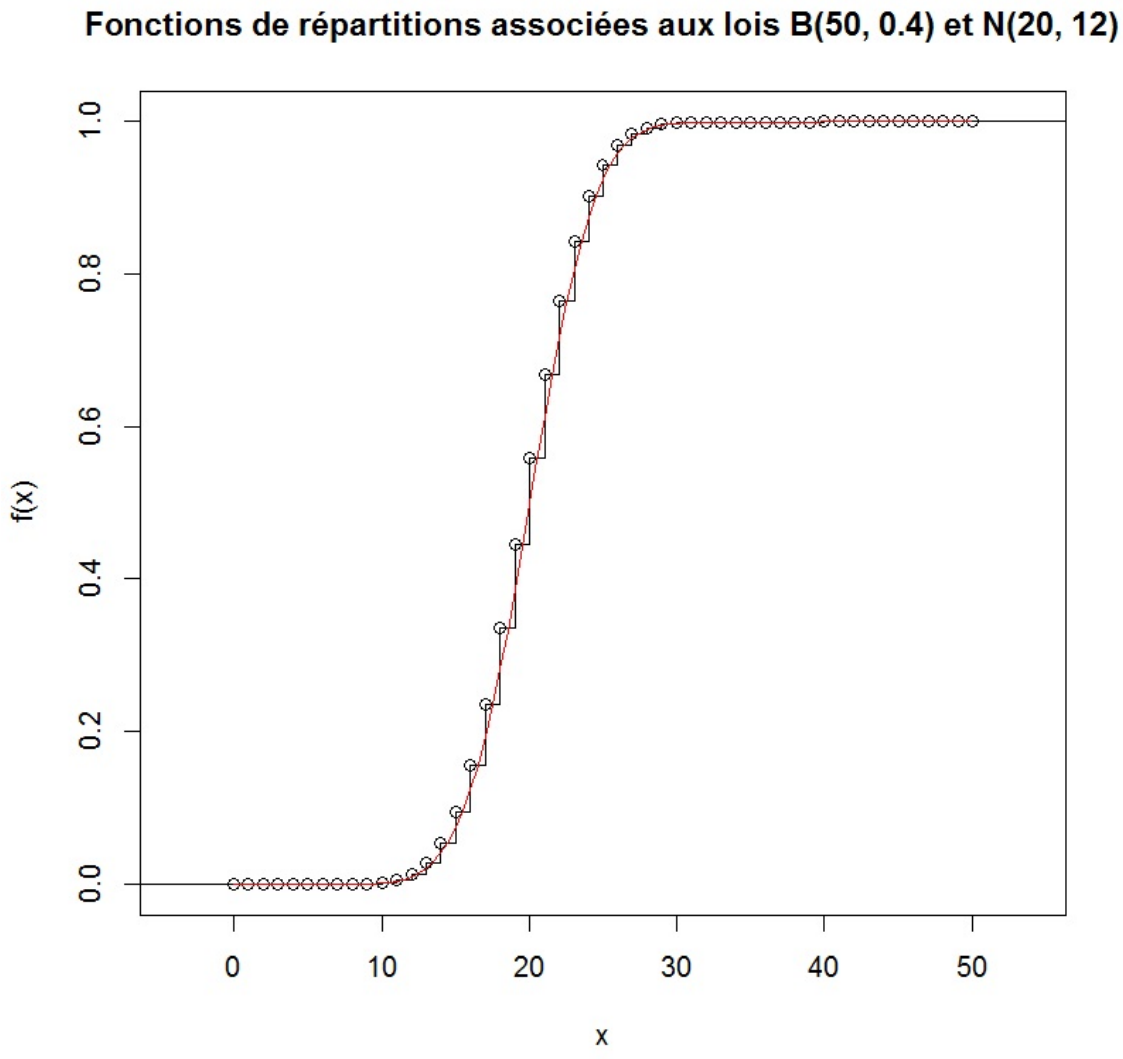
Cela renvoie :



Solution 9. On fait :

```
plot(stepfun(0:50, c(0, pbinom(0:50, 50, 0.4))),
main = "Fonctions de répartitions associées aux lois B(50, 0.4) et N(20, 12)")
curve(pnorm(x, 20, sqrt(12)), 0, 50, col = "red", add = TRUE)
```

Cela renvoie :



Solution 10.

1. On fait :

```
p = c(0.00135, 0.025, 0.95, 0.999, 0.995, 0.99865)
x = qnorm(p)
cbind(p, x)
```

On obtient les quantiles demandés dans la colonne `x` :

```
           p         x
[1,] 0.00135 -2.999977
[2,] 0.02500 -1.959964
[3,] 0.95000  1.644854
[4,] 0.99900  3.090232
[5,] 0.99500  2.575829
[6,] 0.99865  2.999977
```

2. On fait :

```
p = c(0.975, 0.025)
y = qnorm(p, 19, sqrt(3))
cbind(p, y)
```

On obtient les quantiles demandés dans la colonne `y` :

```
         p        y
[1,] 0.975 22.39476
[2,] 0.025 15.60524
```

On a

```
y[1] - 19 - sqrt(3) * qnorm(0.975)
```

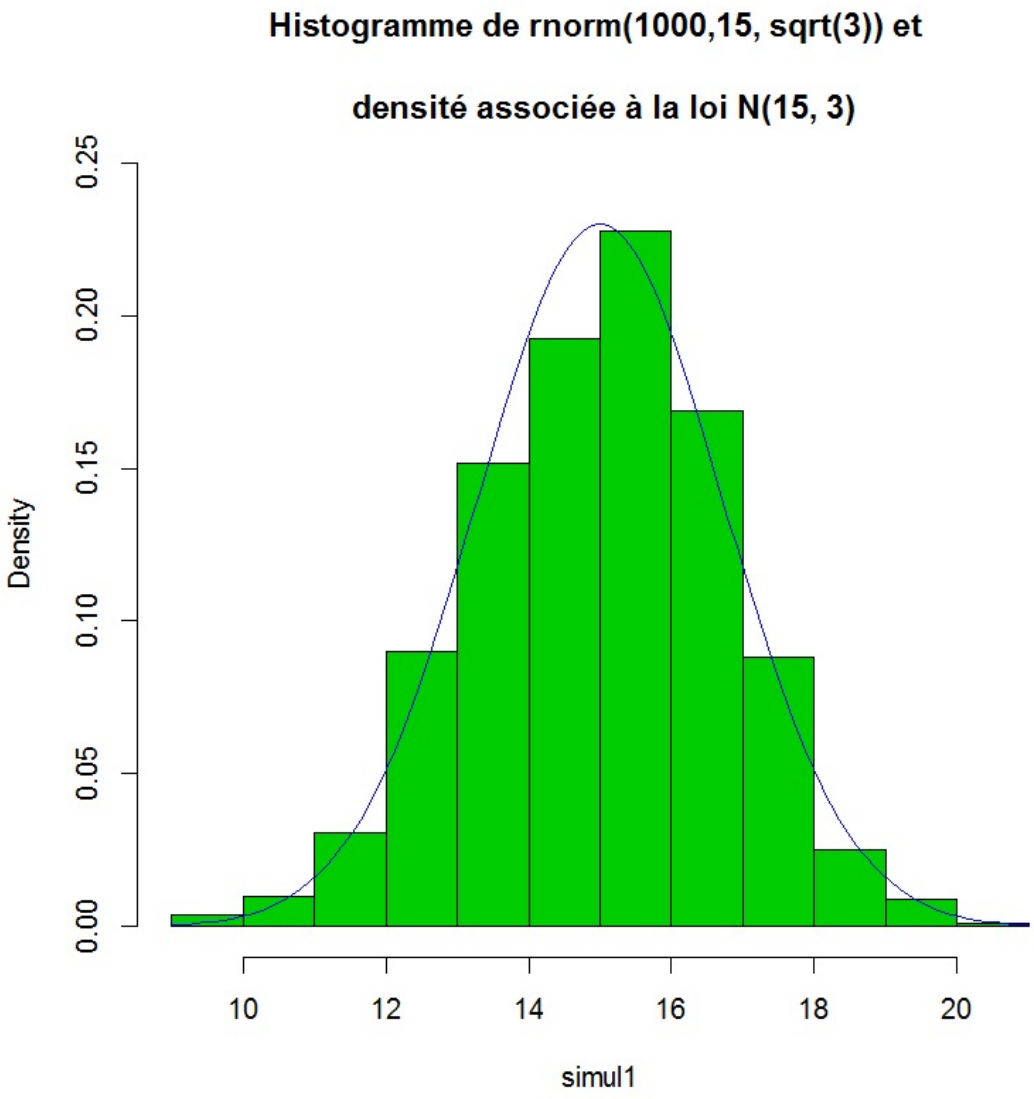
Cela renvoie : `[1] 8.881784e-16`

On arrondit cela à 0.

Solution 11. On fait :

```
simul1 = rnorm(1000, 15, sqrt(3))
hist(simul1, col = 3,  prob = TRUE, ylim = c(0, 0.25),
     main = "Histogramme de rnorm(1000,15, sqrt(3)) et
     \n densité associée à la loi N(15, 3)")
curve(dnorm(x, 15, sqrt(3)), col = 4, add = TRUE)
boxplot(simul1, main = "Boxplot de rnorm(1000, 15, sqrt(3))")
```

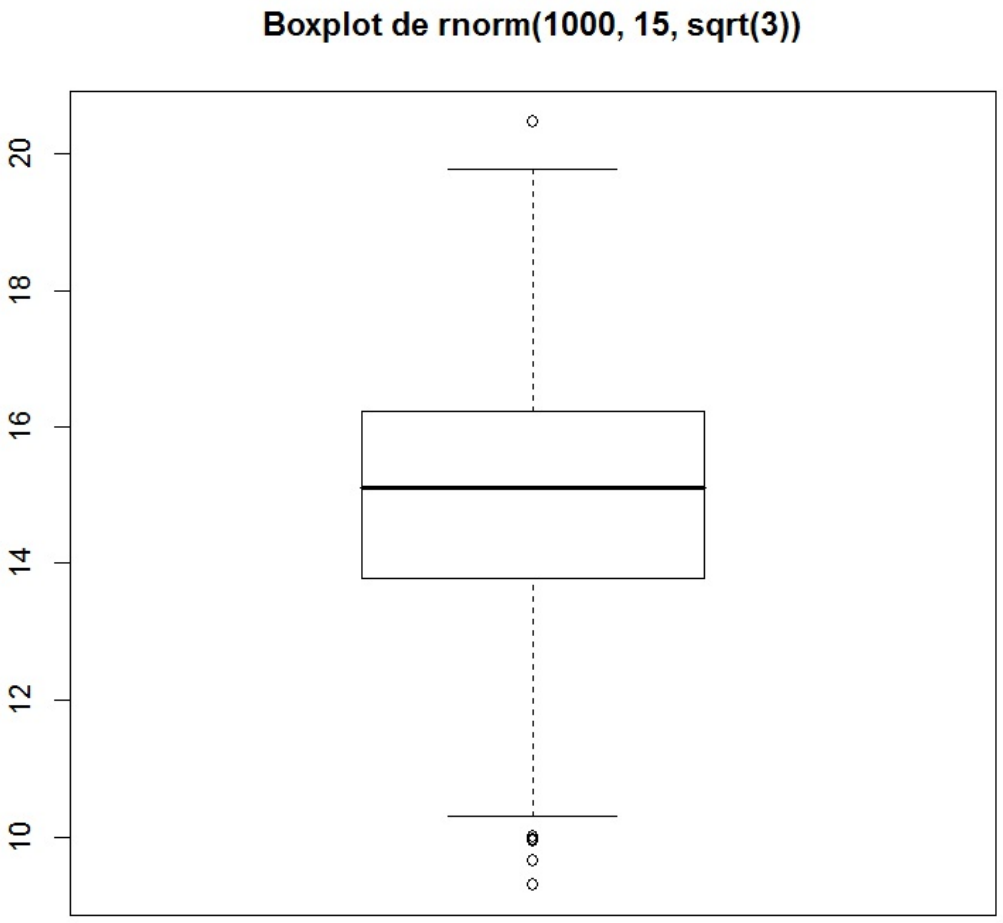
Cela renvoie :



On fait :

```
boxplot(simul1, main = "Boxplot de rnorm(1000, 15, sqrt(3))")
```

Cela renvoie :



Solution 12.

1. On fait :

   ```
   simul2 = rpois(1000, 1)
   ```

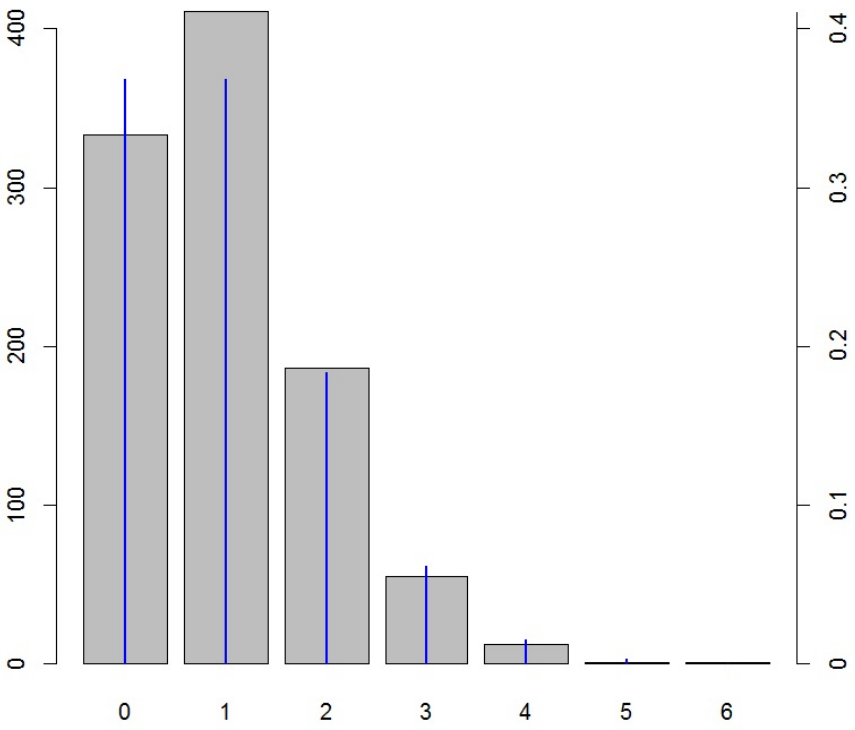
2. En utilisant la fonction `compter`, on fait :

```
maxi = max(simul2)
nbsimul2 = compter(0:maxi, simul2)
```

3. On fait :

```
bppois = barplot(nbsimul2)
points(bppois, dpois(0:maxi,1) * 1000, type = "h",
       col = "blue", lwd = 2)
axis(4, at = seq(0, 1000, by = 100), labels = seq(0, 1, by = 0.1))
```

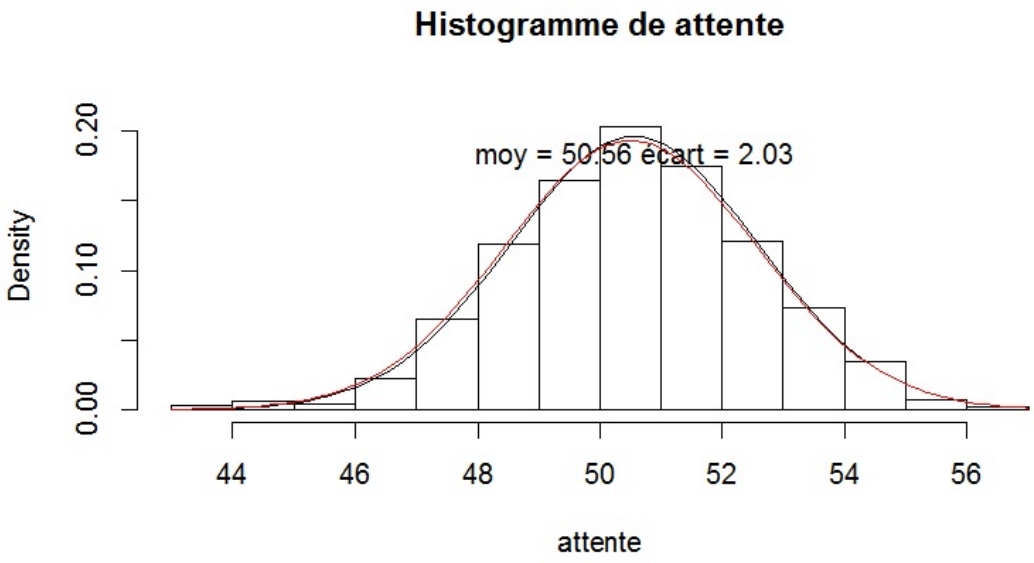
Cela renvoie :

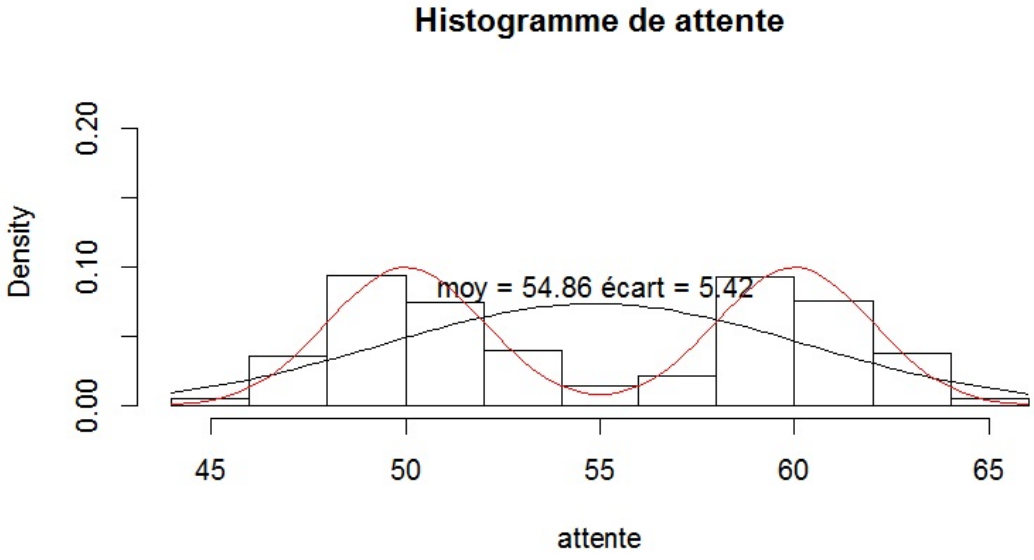


Solution 13. On considère les commandes :

```
peage = function(mu1, mu2, sigma, n) {
attente = numeric(n)
cabines = c("C1", "C2")
for(i in 1 : n) {
tirage = sample(cabines, 1)
if(tirage == "C1") {
attente[i] = rnorm(1, mu1, sigma)
} else {
attente[i] = rnorm(1, mu2, sigma)
}
}
a = 0.9 * max(hist(attente, prob = TRUE, ylim = c(0, 0.22),  xlab = "attente",
main = "Histogramme de attente")$density)
moy = mean(attente)
ecart = sd(attente)
curve(dnorm(x, moy, ecart), add = TRUE)
text(moy, a, paste("moy =", round(moy, 2), "écart =", round(ecart, 2)))
curve(0.5 * (dnorm(x, mu1, sigma) + dnorm(x, mu2, sigma)), col = "red", add = TRUE)
}
par(mfrow = c(2, 1))
peage(50, 51, 2, 1000)
peage(50, 60, 2, 1000)
par(mfrow = c(1, 1))
```

Cela renvoie :

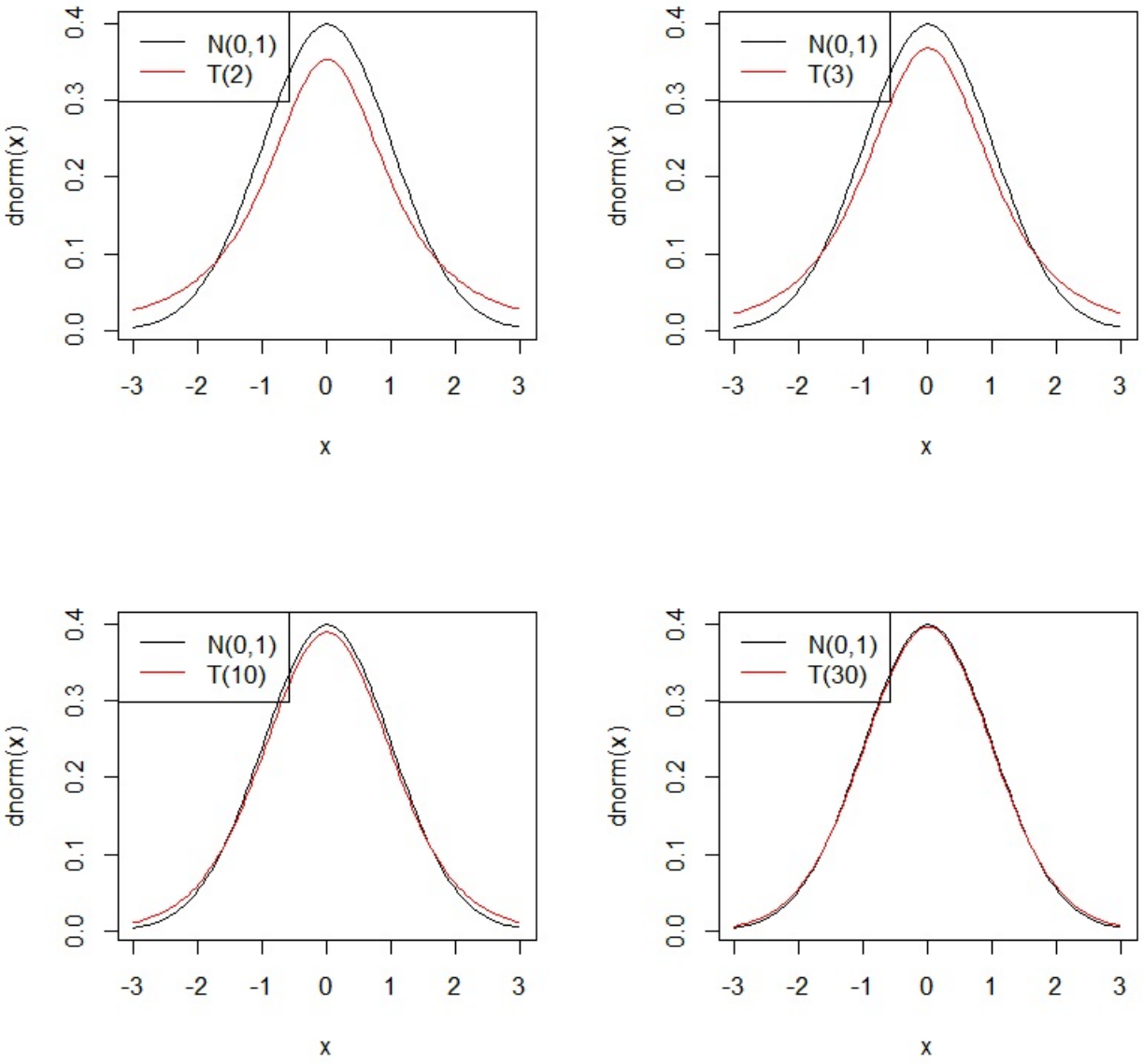




Solution 14. On fait :

```
par(mfrow = c(2, 2))
curve(dnorm(x), -3, 3)
curve(dt(x, 2), -3, 3, col = "red", add = TRUE)
legend("topleft", legend = c("N(0,1)", "T(2)"), col = c("black", "red"), lwd=1)
curve(dnorm(x), -3, 3)
curve(dt(x,3), -3, 3, col = "red", add = TRUE)
legend("topleft", legend = c("N(0,1)", "T(3)"), col = c("black", "red"), lwd = 1)
curve(dnorm(x), -3, 3)
curve(dt(x, 10), -3, 3, col = "red", add = TRUE)
legend("topleft", legend = c("N(0,1)", "T(10)"), col = c("black", "red"), lwd = 1)
curve(dnorm(x), -3, 3)
curve(dt(x, 30), -3, 3, col = "red", add = TRUE)
legend("topleft", legend = c("N(0,1)", "T(30)"), col = c("black", "red"), lwd = 1)
par(mfrow = c(1, 1))
```

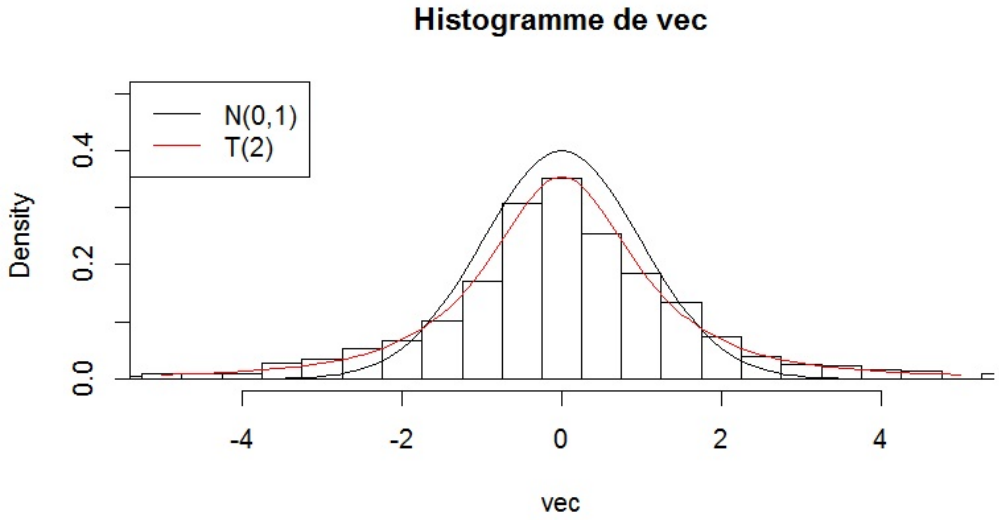
Cela renvoie :

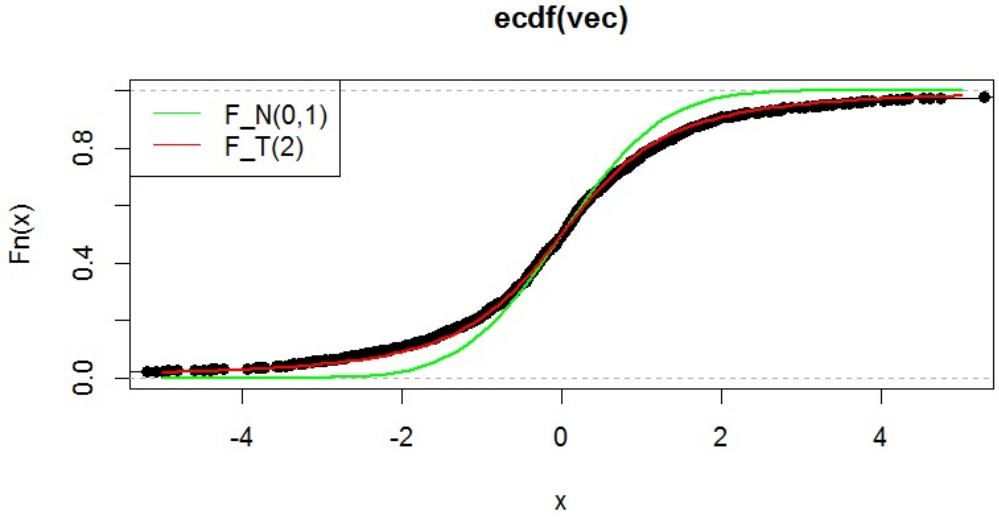


Solution 15. On fait :

```
stu = function(n,nb,mu,sigma) {
vec = numeric(nb)
for (i in 1:nb) {
simul = rnorm(n,mu,sigma)
xbar = mean(simul)
sech = sd(simul)
vec[i] = sqrt(n) * (xbar - mu) / sech
}
par(mfrow = c(2, 1))
hist(vec, prob = TRUE,
breaks = seq(-1000.25, 1000.25, by = 0.5), xlim = c(-5, 5), ylim = c(0, 0.5),
main = "Histogramme de vec")
curve(dnorm(x), -5, 5, add = TRUE)
curve(dt(x, n - 1), -5, 5, col = "red", add = TRUE)
legend("topleft",
legend = c("N(0,1)", paste("T(", n - 1, ")", sep = "")), col = c("black", "red"),
lwd = 1)
plot(ecdf(vec), xlim = c(-5, 5))
curve(pnorm(x), -5, 5, col = "green", lwd = 2, add = TRUE)
curve(pt(x, n-1), -5, 5, col = "red", lwd = 2,  add = TRUE)
legend("topleft", legend = c("F_N(0,1)", paste("F_T(", n - 1, ")", sep = "")),
col = c("green", "red"), lwd = 1)
par(mfrow = c(1,1))
}
stu(3, 1000, 10, 2)
```

Cela renvoie :

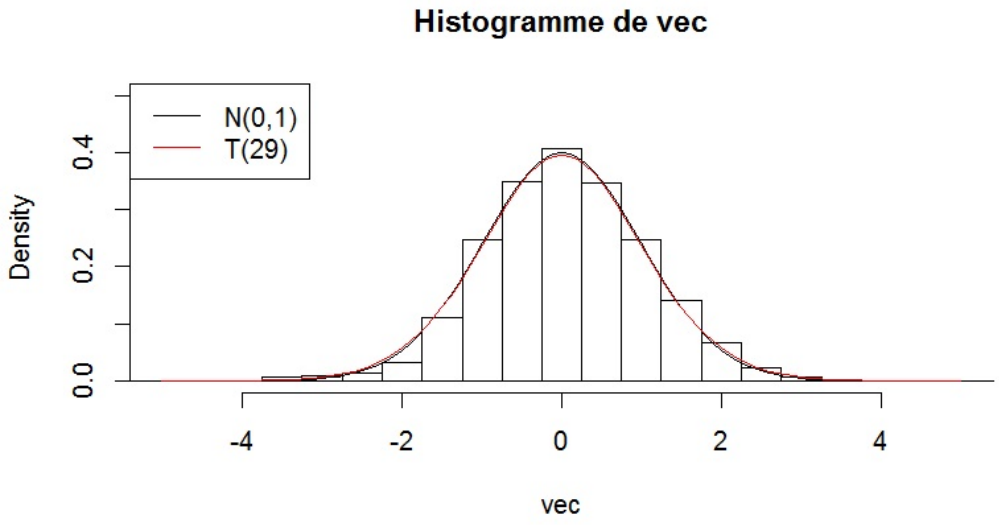


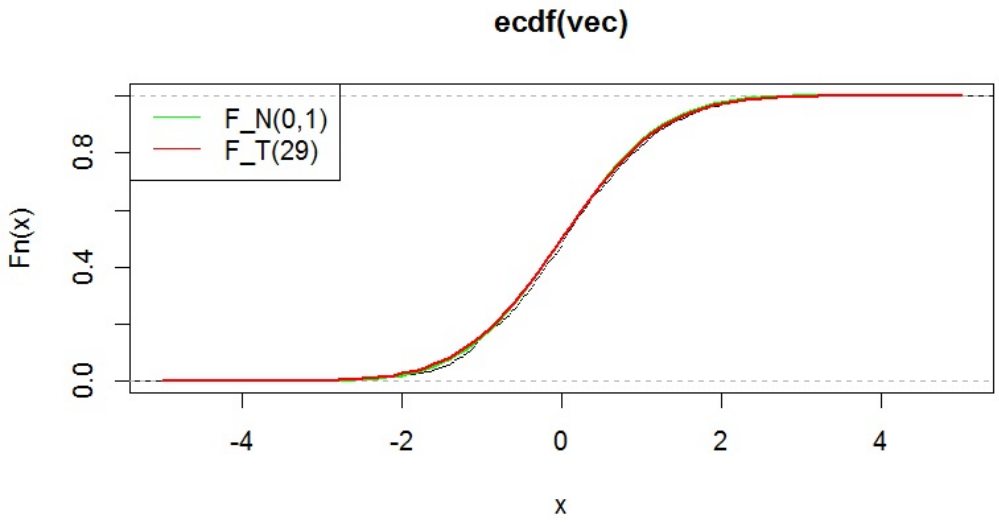


Puis :

```
stu(30, 1000, 10, 2)
```

Cela renvoie :

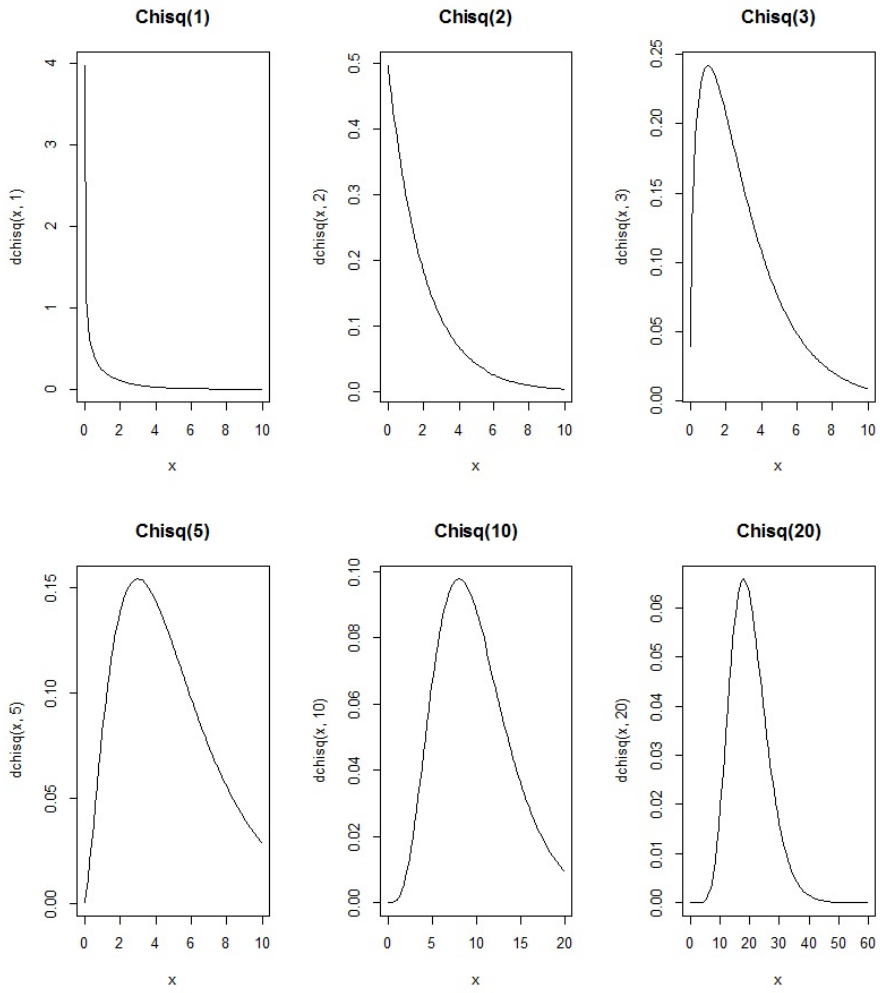




Solution 16. On considère les commandes :

```
par(mfrow = c(2, 3))
curve(dchisq(x, 1), 0.01, 10, main = "Chisq(1)")
curve(dchisq(x, 2), 0.01, 10, main = "Chisq(2)")
curve(dchisq(x, 3), 0.01, 10, main = "Chisq(3)")
curve(dchisq(x, 5), 0.01, 10, main = "Chisq(5)")
curve(dchisq(x, 10), 0.01, 20, main = "Chisq(10)")
curve(dchisq(x, 20), 0.01, 60, main = "Chisq(20)")
par(mfrow = c(1, 1))
```

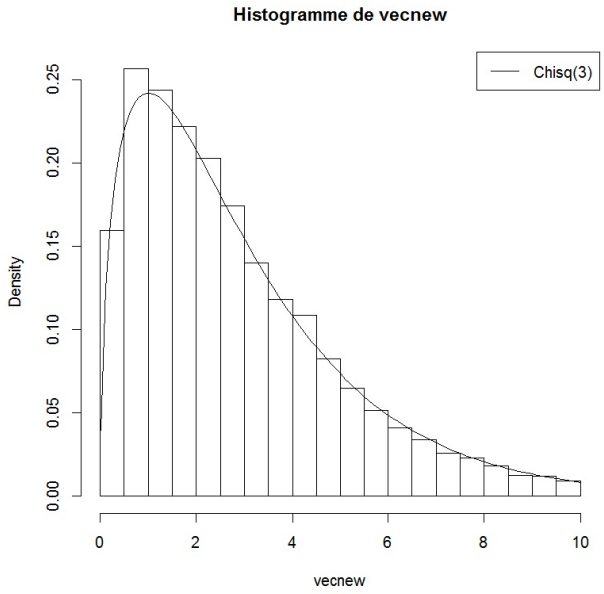
Cela renvoie :



Solution 17. On fait :

```
khi2 = function(n, nb, mu, sigma) {
vec = numeric(nb)
for (i in 1:nb) {
simul = rnorm(n, mu, sigma)
sech2 = var(simul)
vec[i] = (n-1) * sech2 / sigma^2
}
vecnew = vec[vec >= 0.01 & vec <= 10]
hist(vecnew, prob  = TRUE, main = "Histogramme de vecnew")
curve(dchisq(x, n-1), 0.01, 10, add = TRUE)
legend("topright", legend = paste("Chisq(",n-1,")", sep = ""), col = "black",
lwd = 1)
}
par(mfrow = c(1, 1))
khi2(4, 10000, 10, 2)
```

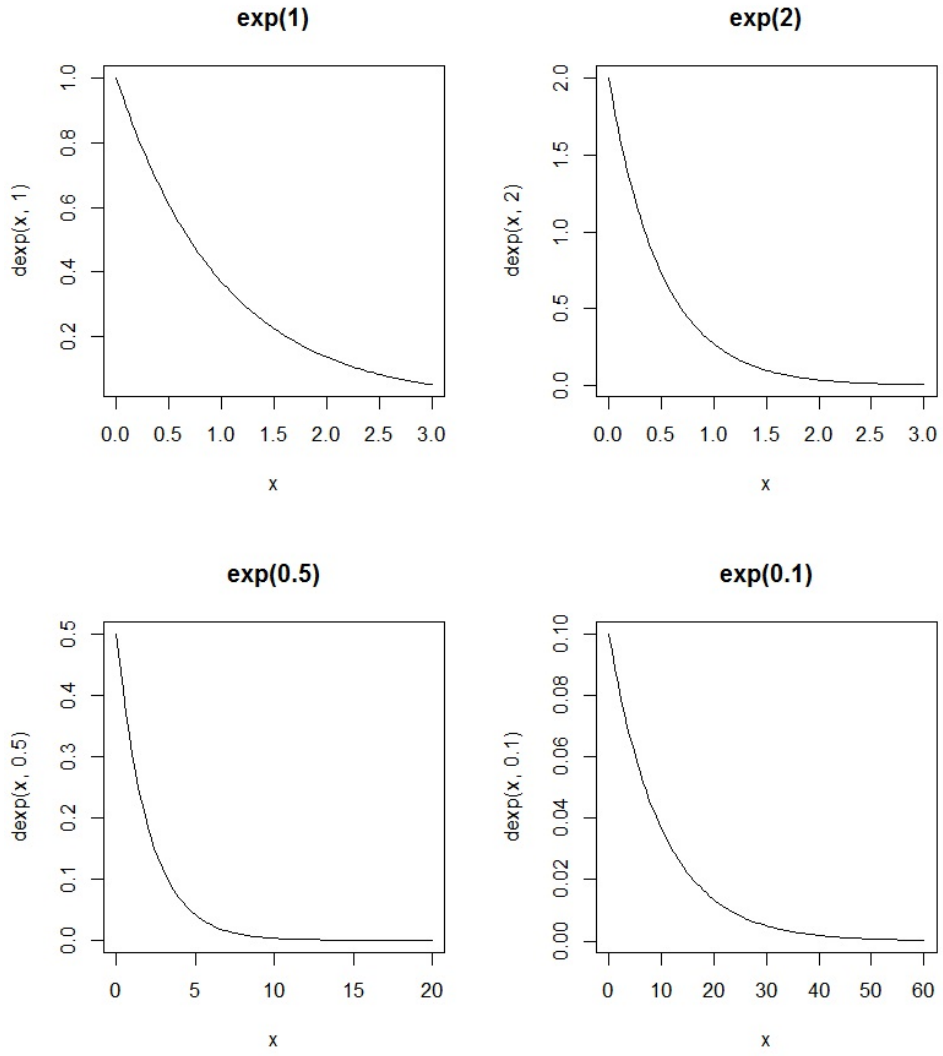
Cela renvoie :



Solution 18. On fait :

```
par(mfrow = c(2, 2))
curve(dexp(x, 1), 0, 3, main = "exp(1)")
curve(dexp(x, 2), 0, 3, main = "exp(2)")
curve(dexp(x, 0.5), 0,20, main = "exp(0.5)")
curve(dexp(x, 0.1), 0,60, main = "exp(0.1)")
par(mfrow=c(1, 1))
```

Cela renvoie :



Solution 19. On considère les commandes :

```
curve(dnorm(x, 0, 1), xlim = c(-10, 10), ylim = c(0, 0.58), col = "red", lwd = 3,
ylab = "", main = "Exemples de densités associées à la loi normale")
curve(dnorm(x, 2, 3), col = "green", lwd = 3, add = TRUE)
curve(dnorm(x, 0, 5), col = "blue", lwd = 3, add = TRUE)
curve(dnorm(x, -2, 0.7), col = "yellow", lwd = 3, add = TRUE)
legend("topright", legend = c("N(0,1)", "N(2,9)", "N(0,25)", "N(-2,0.49)"),
lty = 1, lwd = 3, col = c("red", "green", "blue", "yellow"))
```

Solution 20. On considère les commandes :

```
curve(dexp(x, 0.5), xlim = c(0, 3), ylim = c(0, 1), col = "red", lwd = 3,
ylab = "", main = "Exemples de densités associées à la loi exponentielle")
curve(dexp(x, 1), col = "green", lwd = 3, add = TRUE)
curve(dexp(x, 1.5), col = "blue", lwd = 3, add = TRUE)
curve(dexp(x, 2), col = "yellow", lwd = 3, add = TRUE)
legend("topright", legend = c("E(0.5)", "E(1)", "E(1.5)", "E(2)"),
lty = 1, lwd = 3, col = c("red", "green", "blue", "yellow"))
```

Solution 21.

1. On fait :

   ```
   densexp = function(x, lambda) {
   ifelse (x >= 0,  lambda * exp(-lambda * x), 0)
   }
   ```

2. On fait :

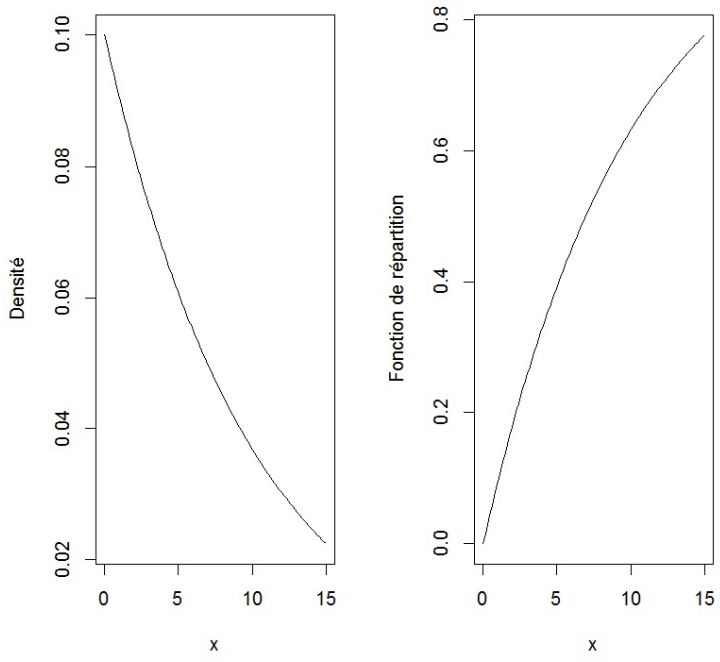
   ```
   integrate(function(x) dexp(x, 1 / 10), 0, 100)
   ```

   Cela renvoie : `0.9999546 with absolute error < 2.8e-14`

3. On considère les commandes :

```
par(mfrow = c(1, 2))
curve(dexp(x, 1 / 10), 0, 15, ylab = "Densité")
curve(pexp(x, 1 / 10), 0, 15, ylab = "Fonction de répartition")
```

Cela renvoie :



4. La probabilité que la durée de vie d'une voiture dépasse 10.2 ans est donnée par :

```
1 - pexp(10.2, 1 / 10)
```

Cela renvoie : `[1] 0.3605949`

Solution 22.

1. On fait :

```
densgamma = function(x, m, lambda) {
ifelse (x >= 0, (1/factorial(m - 1)) * lambda^m * x^(m - 1) *
exp(-lambda * x), 0)
}
```

2. On fait :

```
integrate(function(x) dgamma(x, 5, 1), 0, 100)
```

Cela renvoie : `1 with absolute error < 2.7e-07`

3. On considère les commandes :

```
x = 2:10
pgamma(x, 5, 1)
```

Cela renvoie :

```
[1] 0.05265302 0.18473676 0.37116306 0.55950671 0.71494350 0.82700839 0.90036760
[8] 0.94503636 0.97074731
```

Ces valeurs correspondent à $F_X(2), \ldots, F_X(10)$ respectivement.

4. On fait :

```
qgamma(0.92, 5, 1)
```

Cela renvoie : `[1] 8.376739`

Solution 23.

1. On fait :

```
densnorm = function(x) {
(1 / sqrt(2 * pi)) * exp(-x^2 / 2)
}
```

2. On fait :

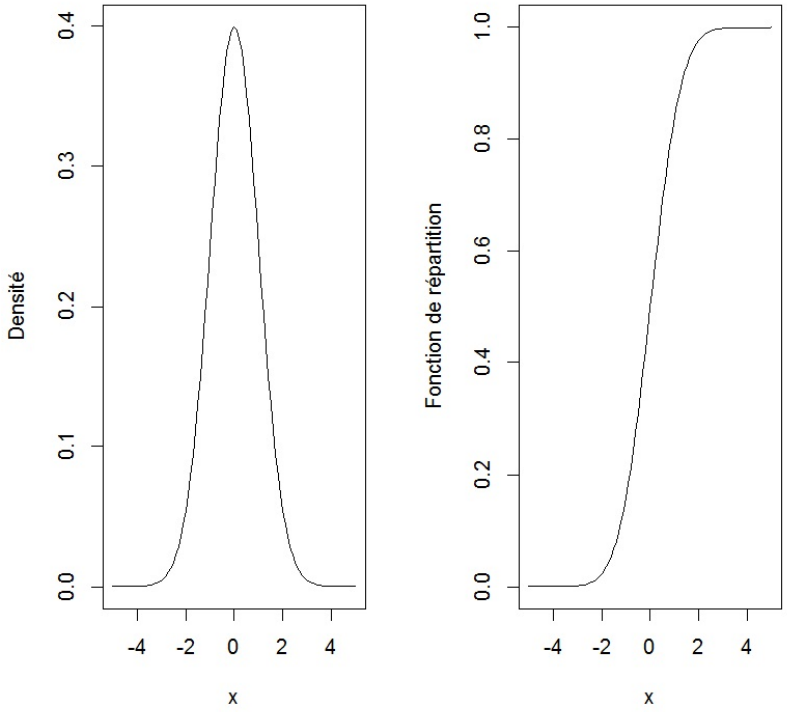
```
integrate(function(x) dnorm(x), -100, 100)
```

Cela renvoie : `1 with absolute error < 3.2e-07`

3. On considère les commandes :

```
par(mfrow = c(1, 2))
curve(dnorm(x), -5, 5, ylab = "Densité")
curve(pnorm(x), -5, 5, ylab = "Fonction de répartition")
```

Cela renvoie :



4. – La probabilité $\mathbb{P}(X \leq 2.2)$ est donnée par :

   `pnorm(2.2)`

   Cela renvoie : `[1] 0.9860966`

   – $\mathbb{P}(X \geq 1.7)$ est donnée par :

   `1 - pnorm(1.7)`

   (ou `pnorm(1.7, lower.tail = FALSE)`)

   Cela renvoie : `[1] 0.04456546`

   – La probabilité $\mathbb{P}(0.2 \leq X < 1.4)$ est donnée par :

   `pnorm(1.4) - pnorm(0.2)`

   Cela renvoie : `[1] 0.3399836`

   – La probabilité $\mathbb{P}(|X| \leq 1.96) = 2\mathbb{P}(X \leq 1.96) - 1$ est donnée par :

   `2 * pnorm(1.96) - 1`

   Cela renvoie : `[1] 0.9500042`

5. On fait :

   `qnorm(0.98)`

   Cela renvoie : `[1] 2.053749`

Solution 24. La probabilité qu'un foie gras pèse

– moins de 650 grammes est donnée par :

`pnorm(650, 550, 100)`

Cela renvoie : `[1] 0.8413447`

– plus de 746 grammes est donnée par :

`1 - pnorm(746, 550, 100)`

(ou `pnorm(746, 550, 100, lower.tail = FALSE)`)

Cela renvoie : `[1] 0.0249979`

– entre 550 grammes et 600 grammes est donnée par :

`pnorm(600, 550, 100) - pnorm(550, 550, 100)`

Cela renvoie : `[1] 0.1914625`

Solution 25. On fait :

```
foncrep = function(x) {
n = 1 + 2 * 0:50
0.5 + (1 / sqrt(2 * pi)) * exp(-x^2 / 2) * sum(x^n / cumprod(n))
}
```

Par exemple, on fait :

```
foncrep(1.2)
```

Cela renvoie : `[1] 0.8849303`

Cela correspond exactement au résultat de `pnorm(1.2)`.

Solution 26.

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

Solution 27. On considère les commandes :

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

Solution 28. On fait :

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

On compare 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 29.

1. On fait,
   - 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. (a) On fait,
   - pour X :

```
x = numeric()
for(i in 1:500){
x[i] = sum(sample(c(0, 1), 10, replace = T, prob = c(2 / 3, 1 / 3)))
}
```

   - pour Y :

```
y = numeric()
for(i in 1:500){
y[i] = sum(sample(c(rep(0, 10), rep(1, 5)), 10))
}
```

   (b) On propose les commandes :

```
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 30. On considère les commandes :

```
n = 10000
u = sample(c(-1,1), n, replace = T)
x = cumsum(u)
x = c(0, x)
```

Solution 31. On fait :

```
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)
```