Statistique pour les Sciences de la Vie et de la Terre

Alexander D. Rahm

Statistique
pour les Sciences de la Vie et de la Terre

Alexander D. Rahm

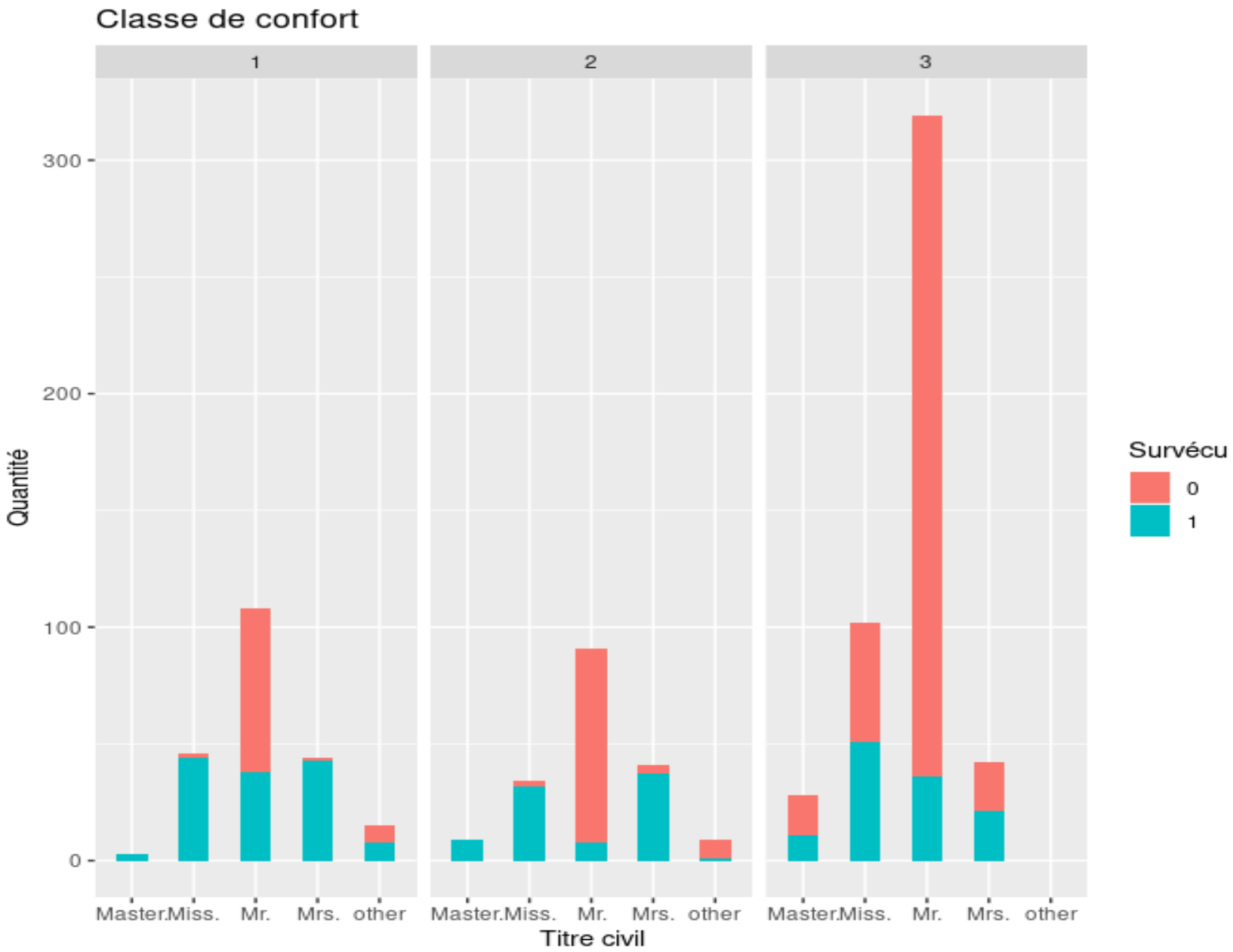
Semestre impair 2022

Exemple principal de base de données illustrant ce cours : ≪ La vie et la mort sur la mer ≫

drawing by Stöwer (public domain since deceased more than 70 years ago)

Donnée : Une partie de la liste des passagers du Titanic, avec l'information qui a survécu.
Notre but : Pronostiquer qui a survécu sur une autre partie de la liste de passagers, où cette information manque.
Notre stratégie : Construire un modèle statistique.

Observations pour construire notre modèle statistique



(diagram obtained with David Langer's source code)

Environnement de travail : La console *R*

Le langage de programmation *R* vient avec la console *R*, qui permet d'exécuter les commandes écrites dans ce langage, et donc d'effectuer des calculs statistiques.

Exemple : La jauge de carburant de ma voiture est cassée, et cette dernière est trop vielle pour justifier une réparation de la jauge. Plutôt, je note le kilométrage et la quantité de carburant chaque fois que je fais remplir à plein le réservoir de carburant.

Quelle est la distance maximale que je peux conduire avant de remettre du carburant, sans tomber en panne de carburant, ni transporter des bidons d'essence ?

J'aimerais trouver une réponse précise et fiable basée sur ma consommation documentée, sachant que la capacité du réservoir de carburant de ma voiture est spécifée à 35 litres.

Quelle est la distance maximale que je peux conduire avant de remettre du carburant, sans tomber en panne de carburant, ni transporter des bidons d'essence ?

Dans la console *R*, nous pouvons entrer mes valeurs notées lors des remises à plein du réservoir dans deux listes :

```
kilometrage = c(92627, 93086, 93398, 93864, 94198)
litres = c(26.59, 23.51, 29.37, 21.00)
```

où le volume de carburant noté avec le premier kilométrage a été omis intentionellement. Alors le code *R* suivant résout la question :

```
distance = NULL;
for (i in 1:length(litres)){
    gap = kilometrage[i+1] -kilometrage[i]
    distance = c(distance, gap)
} 35*distance/litres
```

Le retour affiché par la console *R* est

[1] 604.1745 464.4832 555.3286 556.6667

et la plus petite de ces valeurs (indiquant 464 kilomètres) est la réponse à ma question. Nous pouvons rajouter des mesurements dans les deux listes ci-dessous pour obtenir une réponse plus fiable.

Instructions d'installation

Sur votre propre ordinateur, installez le logiciel gratuit *R* en suivant les instructions au Comprehensive *R* Archive Network (CRAN) : `https://cran.r-project.org/`

Sur un client léger, démarrez la console *R* en cliquant Démarrer → Programmes → Statistique → *R*.

Il est préférable de vous servir d'un éditeur de texte qui colore la syntaxe *R*, comme `gedit` sous Linux ou `Notepad++` sous Windows. Vous pouvez aussi utiliser *R* `Studio` si vous voulez.

Instructions d'initialisation

 # Lancez la console R et ordonnez
≪ `dir.create("`*chemin*`/Titanic/")` ≫, **où vous nécrivez pas le mot ≪ chemin ≫, mais**

- sous UNIX / Linux / Apple / Mac :
  ≪ chemin ≫ = ≪ `~/Desktop` ≫,
- sur les clients legers :
  ≪ chemin ≫ = ≪ Z: ≫.
- sur les ordinateurs Windows personnels :
  ≪ chemin ≫ = ≪ C: ≫.

 # Téléchargez les fichiers "train.csv" et "test.csv" [publiés sur le site Kaggle],
`http://gaati.org/rahm/statistique/train.csv`
`http://gaati.org/rahm/statistique/test.csv`
 # et installez-les dans le répertoire "chemin/Titanic/". Ensuite dans la console,
`setwd("chemin/Titanic/")` # **où vous nécrivez pas le mot ≪ chemin ≫**

 # Importez les données :

```
train <- read.csv("train.csv", header = TRUE)
test <- read.csv("test.csv", header = TRUE)
```

 # Faites afficher leurs premières six lignes dans la console :

```
head(train)
```

Les six premières entrées de la liste

```
> head(train)
  PassengerId Survived Pclass
1           1        0      3
2           2        1      1
3           3        1      3
4           4        1      1
5           5        0      3
6           6        0      3
                                                 Name Gender Age SibSp Parch
1                             Braund, Mr. Owen Harris   male  22     1     0
2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
3                              Heikkinen, Miss. Laina female  26     0     0
4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
5                            Allen, Mr. William Henry   male  35     0     0
6                                    Moran, Mr. James   male  NA     0     0
            Ticket    Fare Cabin Embarked
1        A/5 21171  7.2500              S
2         PC 17599 71.2833   C85        C
3 STON/O2. 3101282  7.9250              S
4           113803 53.1000  C123        S
5           373450  8.0500              S
6           330877  8.4583              Q
>
```

Exercice 1

 # Regardez les six dernières entrées de la liste, en vous servant de la fonction

`tail()`

 # . Elle fonctionne comme la fonction

`head()`

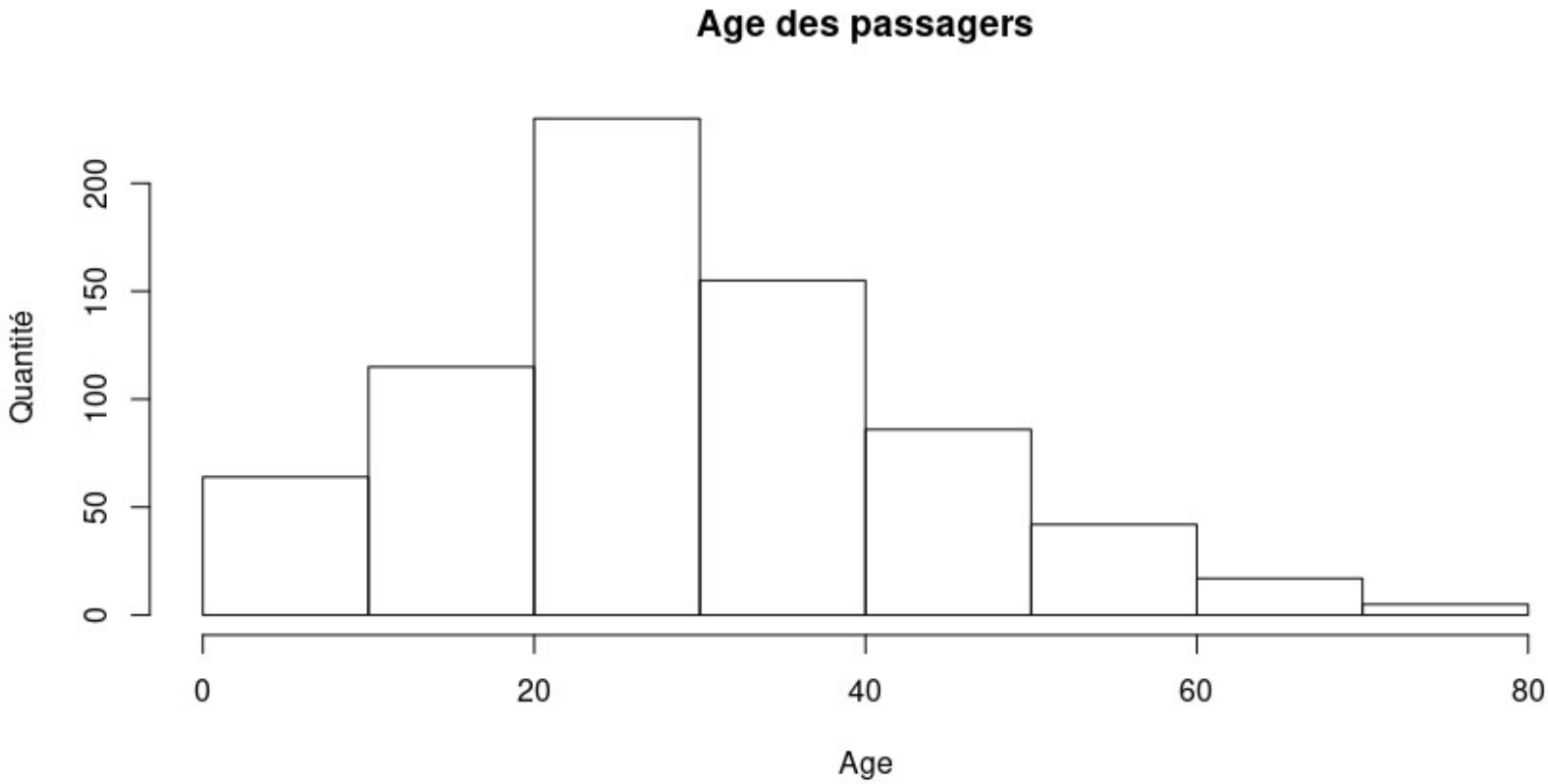
 # .

Histogrammes pour une visualisation rapide des données

 # La commande

```
hist(train$Age, xlab="Age", ylab="Quantité",
main="Age des passagers")
```

 # nous produit un histogramme pour une visualisation :



Note sur la console R

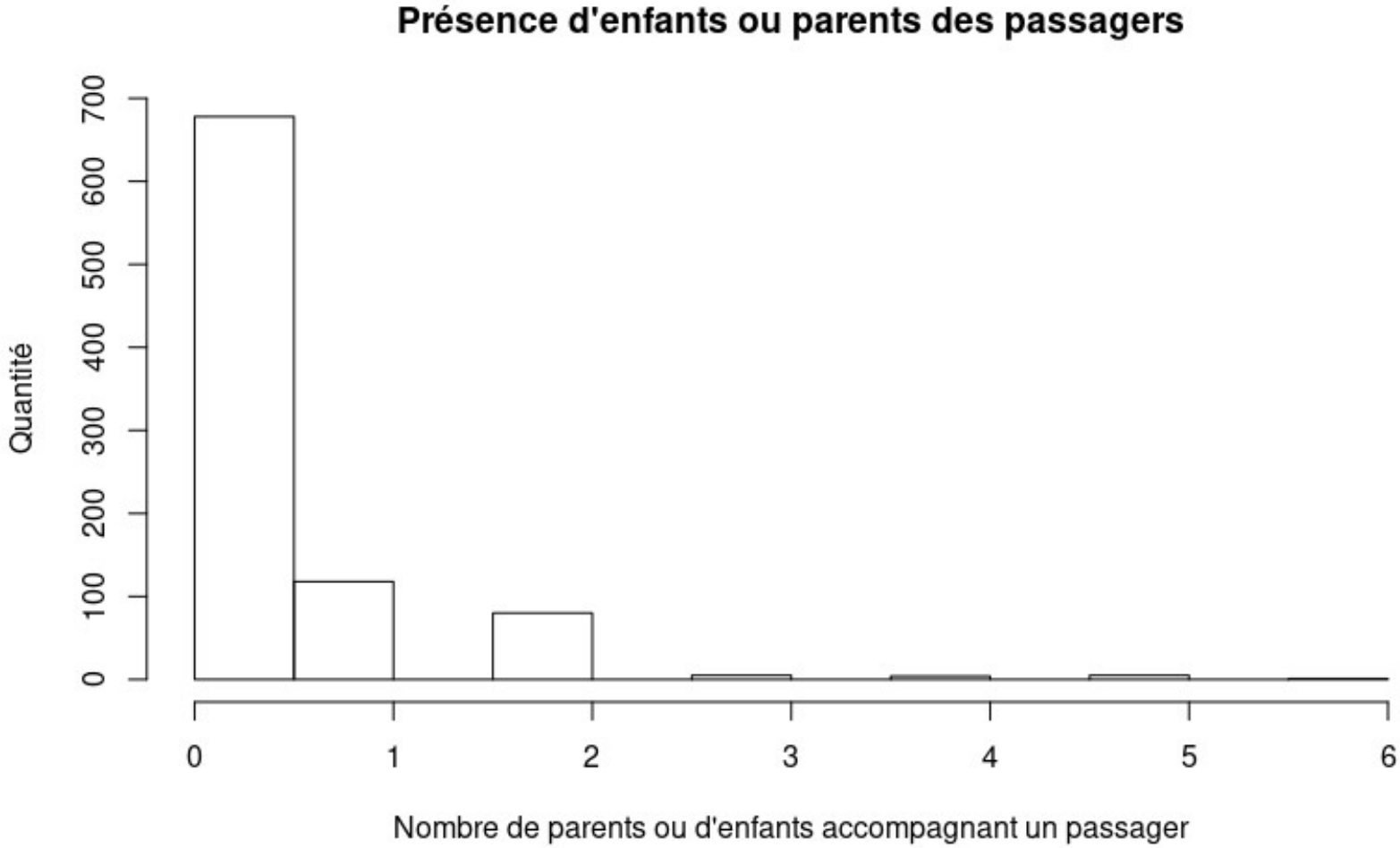
 # Dans la console R, la commande

```
hist(train$Age, xlab="Age", ylab="Quantité",
main="Age des passagers")
```

 # s'affiche comme :

```
> hist(train$Age, xlab="Age", ylab="Quantité",
+ main="Age des passagers")
>
```

 # N'entrez pas les symboles > et + !
 # Le symbole > signifie que la console R attend une nouvelle commande,
 # et le symbole + signifie que la console R attend que la commande continue parce qu'une paranthèse, ou un crochet, ou un guillement a été ouvert et pas encore fermé.




```
hist(train$Parch, xlab=
+ "Nombre de parents ou d'enfants accompagnant un passager",
+ ylab="Quantité",
+ main="Présence d'enfants ou parents des passagers")
```

Exercice 2

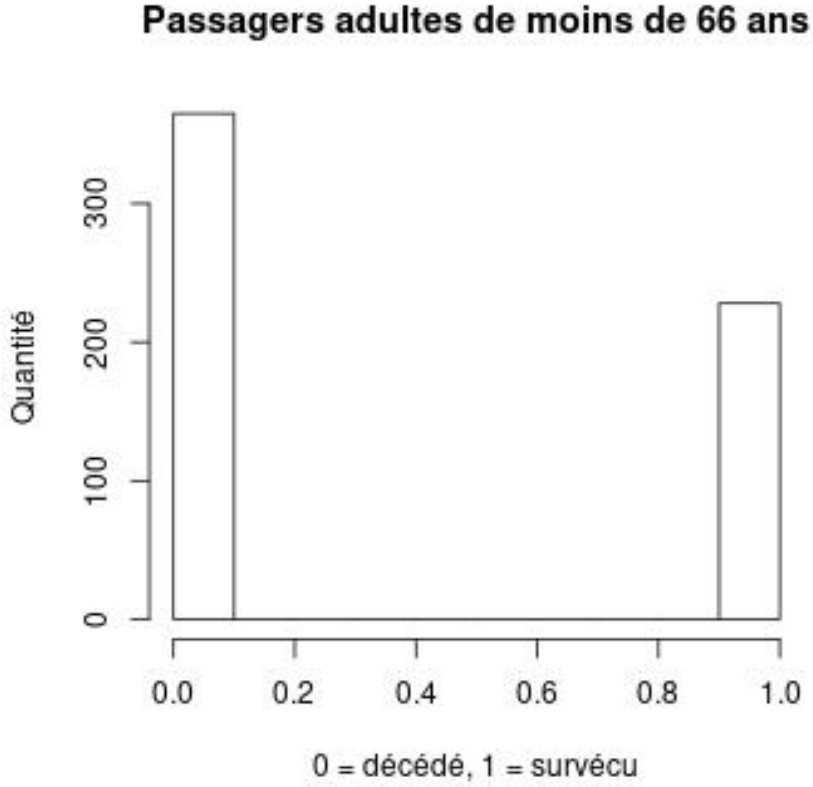
 # Faites dessiner un histogramme des classes de confort des passagers.

 # En regardant les six premières ou dernières entrées de la liste, vous pouvez voir que la classe de confort est enregistrée dans la colonne

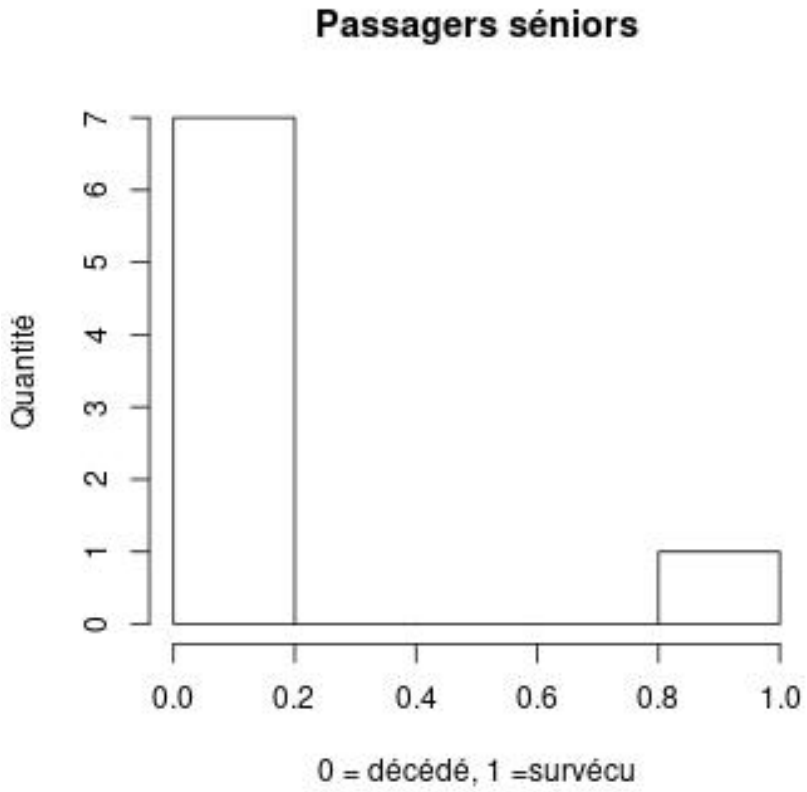
`Pclass`

 # du tableau.

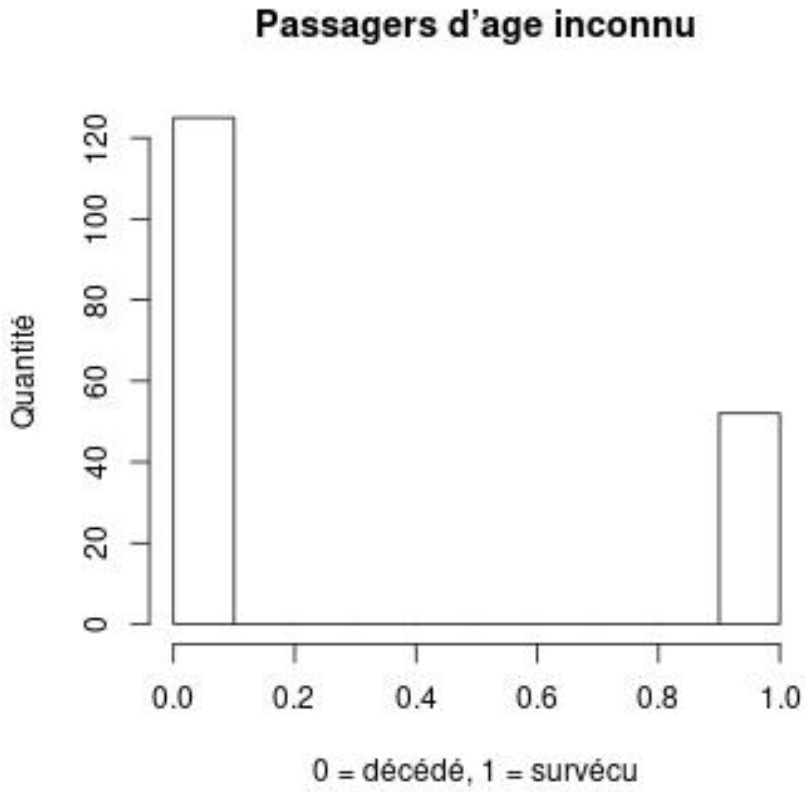
Pour cet exercice et tous qui suivront : Sauvegardez vos résultats (diagrammes, résultats affichés dans la console R, observations, etc.) avec les commandes avec lesquelles ils peuvent être reproduits dans un fichier bureautique, et exportez-le en ≪ portable document format ≫ (.pdf). Sauvegardez les commandes avec lesquelles vos résultats peuvent être reproduits aussi dans un fichier de texte brut (.txt).




```
# 1. Ouvrir le fichier .jpeg
jpeg("adultes_de_moins_de_66.jpg", width = 350, height = 350)
# 2. Créer l'histogramme
hist(train$Survived[train$Age > 17 & train$Age < 66],
+ xlab="0 = décédé, 1 = survécu", ylab="Quantité",
+ main="Passagers adultes de moins de 66 ans")
# 3. Fermer le fichier :  dev.off()
```




```
# 1. Ouvrir le fichier .jpeg
jpeg("seniors.jpg", width = 350, height = 350)
# 2. Créer l'histogramme
hist(train$Survived[train$Age > 65], xlab="0 = décédé, 1 =
survécu", ylab="Quantité", main="Passagers séniors")
# 3. Fermer le fichier :  dev.off()
```




```
# 1. Ouvrir le fichier .jpeg
jpeg("ageInconnu.jpg", width = 350, height = 350)
# 2. Créer l'histogramme hist(train$Survived[is.na(train$Age)],
xlab="0 = décédé, 1 = survécu", ylab="Quantité",
main="Passagers d'age inconnu")
# 3. Fermer le fichier : dev.off()
```

Exercice 3

 # Faites dessiner et sauvegarder sur le disque dur trois histogrammes de survie restreints à une classe de confort des passagers, respectivement distinguée comme suivant :

1. Un histogramme de survie des passagers de 1ère classe ;
2. Un histogramme de survie des passagers de 2ème classe ;
3. Un histogramme de survie des passagers de 3ème classe.

 # En regardant les six premières ou dernières entrées de la liste, vous pouvez voir que la classe de confort est enregistrée dans la colonne

`Pclass`

 # du tableau.

Produire des tableaux dans la console

```
table(train$Survived[train$Age < 18])
```

produit le petit tableau suivant dans la console :

$$\begin{pmatrix} 0 & 1 \\ 52 & 61 \end{pmatrix}$$

où la colonne du zéro contient les décédés, la colonne de l'un contient les survivants. De la même manière :

```
table(train$Survived[train$Age>17 & train$Age<66])
```

$$\begin{pmatrix} 0 & 1 \\ 365 & 228 \end{pmatrix}$$

```
table(train$Survived[train$Age > 65])
```

$$\begin{pmatrix} 0 & 1 \\ 7 & 1 \end{pmatrix}$$

```
table(train$Survived[is.na(train$Age)])
```

$$\begin{pmatrix} 0 & 1 \\ 125 & 52 \end{pmatrix}$$

Exercice 4

 # Regardez les six dernières entrées de la liste, en vous servant de la fonction `tail()` ,

 # et observez la dernière colonne, ≪ Embarked ≫.

 # Elle prend trois charactères comme valeurs dans un éventail, pour spécifier le port d'embarquement :

- ≪ "C" ≫ *# pour Cherbourg* ;
- ≪ "S" ≫ *# pour Southampton* ;
- ≪ "Q" ≫ *# pour Queenstown*.

 # Pour chacun de ces ports d'embarquement, faites construire un tableau dans la console qui affiche combien de passagers de ce port ont péri, et combien ont survécu.

Distinction des passagers par leur titres civil

 # La fonction extractTitle de David Langer nous permet d'extraire le titre civil d'un nom de passager :

```
extractTitle <- function(name) {
  name <- as.character(name)
  if (length(grep("Miss.", name)) > 0) {
        return("Miss.")
  } else if (length(grep("Mrs.", name)) > 0) {
        return("Mrs.")
  } else if (length(grep("Master.", name)) > 0) {
        return("Master.")
  } else if (length(grep("Mr.", name)) > 0) {
        return("Mr.")
  } else {
        return("other")
  }
}
```

Hiérarchisation des questions

Nous notons que les questions "if" sont hiérarchisées :

```
extractTitle("Mr.Miss.")
```

```
[1] "Miss."
```

```
extractTitle("Mrs.Miss.")
```

```
[1] "Miss."
```

```
extractTitle("Mrs.Mr.")
```

```
[1] "Mrs."
```

```
extractTitle("Master.Mr.")
```

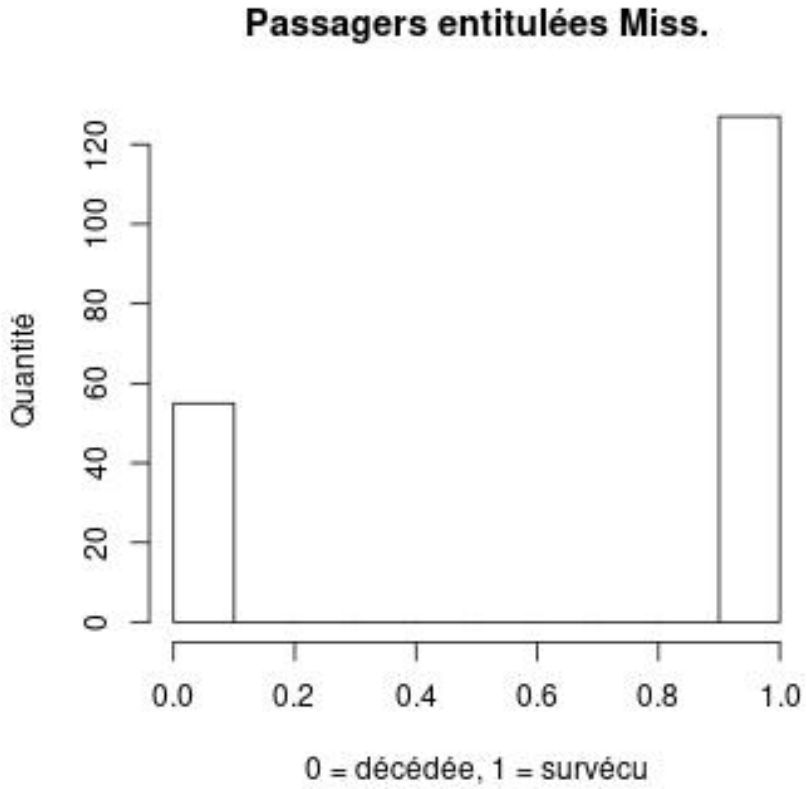
```
[1] "Master."
```

donc une fois qu'une d'elles est répondu de manière positive, nous n'entrons plus aux autres questions.

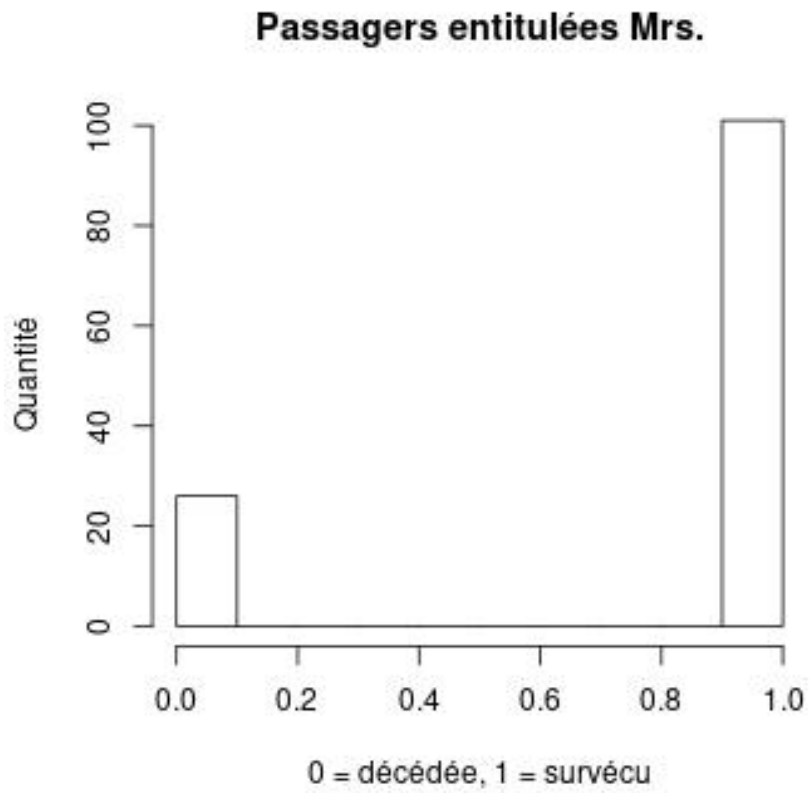
```
## Maintenant nous allons insérer les titres de tous les passagers
dans le fichier "train" :
titles <- NULL
for (i in 1:nrow(train)) {
    titles <- c(titles,
        extractTitle(train[i,"Name"]))
}

## et nous ajoutons cette liste de titres comme une colonne
supplémentaire à notre tableau :
train$Title <- titles
table(train$Title)
```

Master.	*Miss.*	*Mr.*	*Mrs.*	*other*
40	182	518	127	24

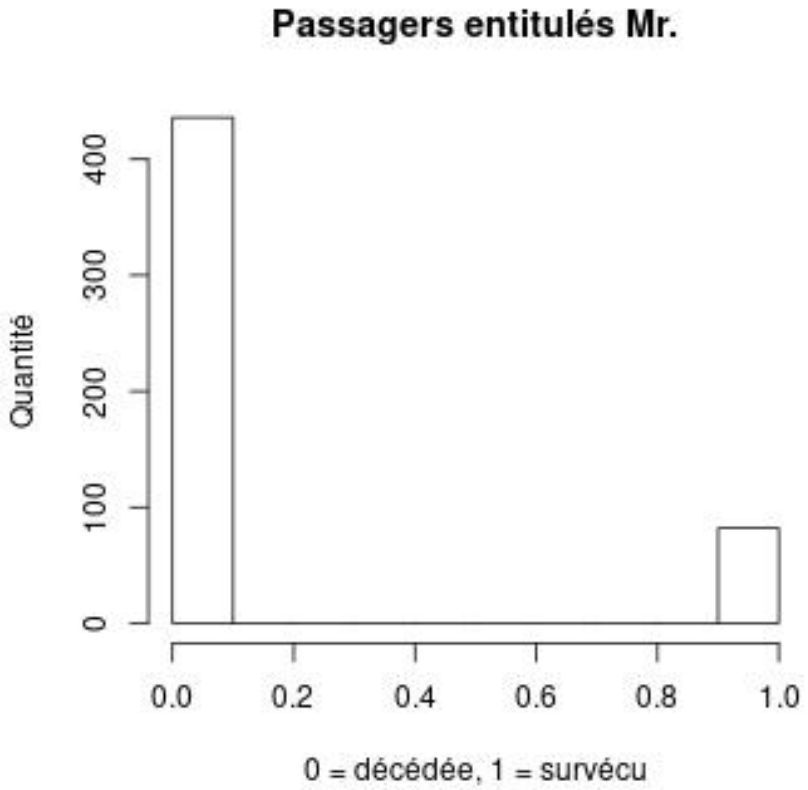

```
# 1° Ouvrir le fichier .jpeg
jpeg("Master.jpg", width = 350, height = 350)
# 2° Créer l'histogramme
hist(train$Survived[train$Title=="Master."], xlab="0
= décédée, 1 = survécu", ylab="Quantité",
main="Passagers entitulés Master.")
# 3° Fermer le fichier : dev.off()
```




```
# 1° Ouvrir le fichier .jpeg  jpeg("Miss.jpg", ...)
# 2° Créer l'histogramme
hist(train$Survived[train$Title=="Miss."], xlab="0 =
décédée, 1 = survécu", ylab="Quantité",
main="Passagers entitulées Miss.")
# 3° Fermer le fichier : dev.off()
```



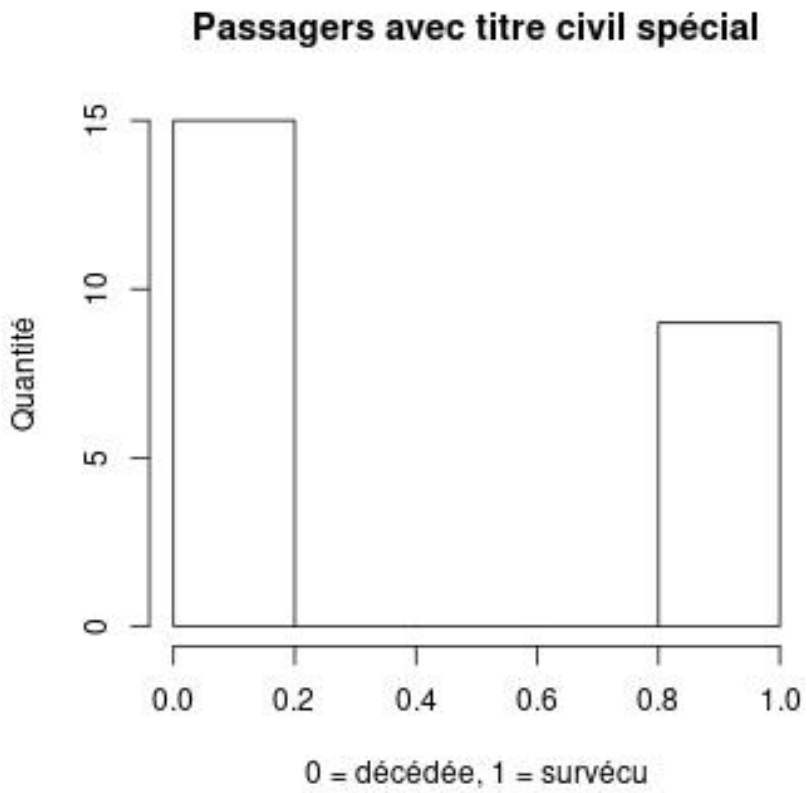

```
# 1° Ouvrir le fichier .jpeg : jpeg("Mrs.jpg", ...)
# 2° Créer l'histogramme
hist(train$Survived[train$Title=="Mrs."], xlab="0 =
décédée, 1 = survécu", ylab="Quantité",
main="Passagers entitulées Mrs.")
# 3° Fermer le fichier : dev.off()
```




```
# 1° Ouvrir le fichier .jpeg
jpeg("Mr.jpg", width = 350, height = 350)
# 2° Créer l'histogramme
hist(train$Survived[train$Title=="Mr."], xlab="0 =
décédée, 1 = survécu", ylab="Quantité",
main="Passagers entitulés Mr.")
# 3° Fermer le fichier : dev.off()
```

Afficher les autres titres civils

```
> hist(train$Age[train$Title=="other"], xlab="age", ylab="Quantité",
main="Passagers avec titre civil spécial")
> for (i in 1:nrow(train)) {
+   if (train[i,"Title"] == "other") {
+   print(train[i,2:6])
+   }
+ }
   Survived Pclass                      Name Gender Age
31        0      1 Uruchurtu, Don. Manuel E   male  40
    Survived Pclass                                 Name Gender Age
150        0      2 Byles, Rev. Thomas Roussel Davids   male  42
    Survived Pclass                           Name Gender Age
151        0      2 Bateman, Rev. Robert James   male  51
    Survived Pclass                            Name Gender Age
246        0      1 Minahan, Dr. William Edward   male  44
    Survived Pclass                               Name Gender Age
250        0      2 Carter, Rev. Ernest Courtenay   male  54
    Survived Pclass                     Name Gender Age
318        0      2 Moraweck, Dr. Ernest   male  54
    Survived Pclass                               Name Gender Age
370        1      1 Aubart, Mme. Leontine Pauline female  24
    Survived Pclass                Name Gender Age
399        0      2 Pain, Dr. Alfred   male  23
```




```
# 1° Ouvrir le fichier .jpeg : jpeg("other.jpg", ...)
# 2° Créer l'histogramme
hist(train$Survived[train$Title=="other"], xlab="0 =
décédée, 1 = survécu", ylab="Quantité",
main="Passagers avec titre civil spécial")
# 3° Fermer le fichier : dev.off()
```

Ages extrêmes et survie

```
min(train$Age[is.na(train$Age)==FALSE])
# [1] 0.42
max(train$Age[is.na(train$Age)==FALSE])
# [1] 80
min(train$Age[is.na(train$Age)==FALSE &
train$Survived == 1])
# [1] 0.42
max(train$Age[is.na(train$Age)==FALSE &
train$Survived == 1])
# [1] 80
min(train$Age[is.na(train$Age)==FALSE &
train$Survived == 0])
# [1] 1
max(train$Age[is.na(train$Age)==FALSE &
train$Survived == 0])
# [1] 74
```

Exercice 5

Rajoutez une colonne pour les passagers entitulés "Dr." au tableau ci-dessous,

$$\begin{pmatrix} \textit{Master}. & \textit{Miss}. & \textit{Mr}. & \textit{Mrs}. & \textit{other} \\ 40 & 182 & 518 & 127 & 24 \end{pmatrix}$$

qui peut être obtenu par la commande
`table(train$Title)`
une fois que la colonne `Title` a été rajoutée grâce à la fonction `extractTitle`. Modifiez donc la fonction `extractTitle` pour séparer les passagers entitulés "Dr." des autres passagers.

Préparation d'un diagramme avec ≪ ggplot ≫

```
install.packages("ggplot2")
```
Attention, ceci ne marche pas correctement sur les clients légers, donc en salle de cours, vous devrez abandonner la construction du diagramme suivant.
L'installation vous demandera de confirmer que vous voulez installer cette bibliothèque dans votre dossier personnel, et de choisir un serveur pas trop éloigné de la Polynésie pour télécharger la bibliothèque.

```
# Chargez les bibliothèques dans la console R :
library("ggplot2")
library("stringr")
```

```
## ggplot a besoin des entrées comme options dans un éventail (appelé ≪ factor ≫).
train$Survived <- as.factor(train$Survived)
train$Title <- as.factor(titles)
```
Attention : Maintenant vous ne pourrez plus faire dessiner des histogrammes basés sur ces deux colonnes. Si vous désireriez faire ceci, vous devriez reconvertir l'éventail en entiers, par exemple avec la fonction as.integer(), qui convertit le numéro de l'option en entier. Ensuite, il faudra le transformer en la valeur originale :

```
train$Survived <- as.integer(train$Survived) −1
```

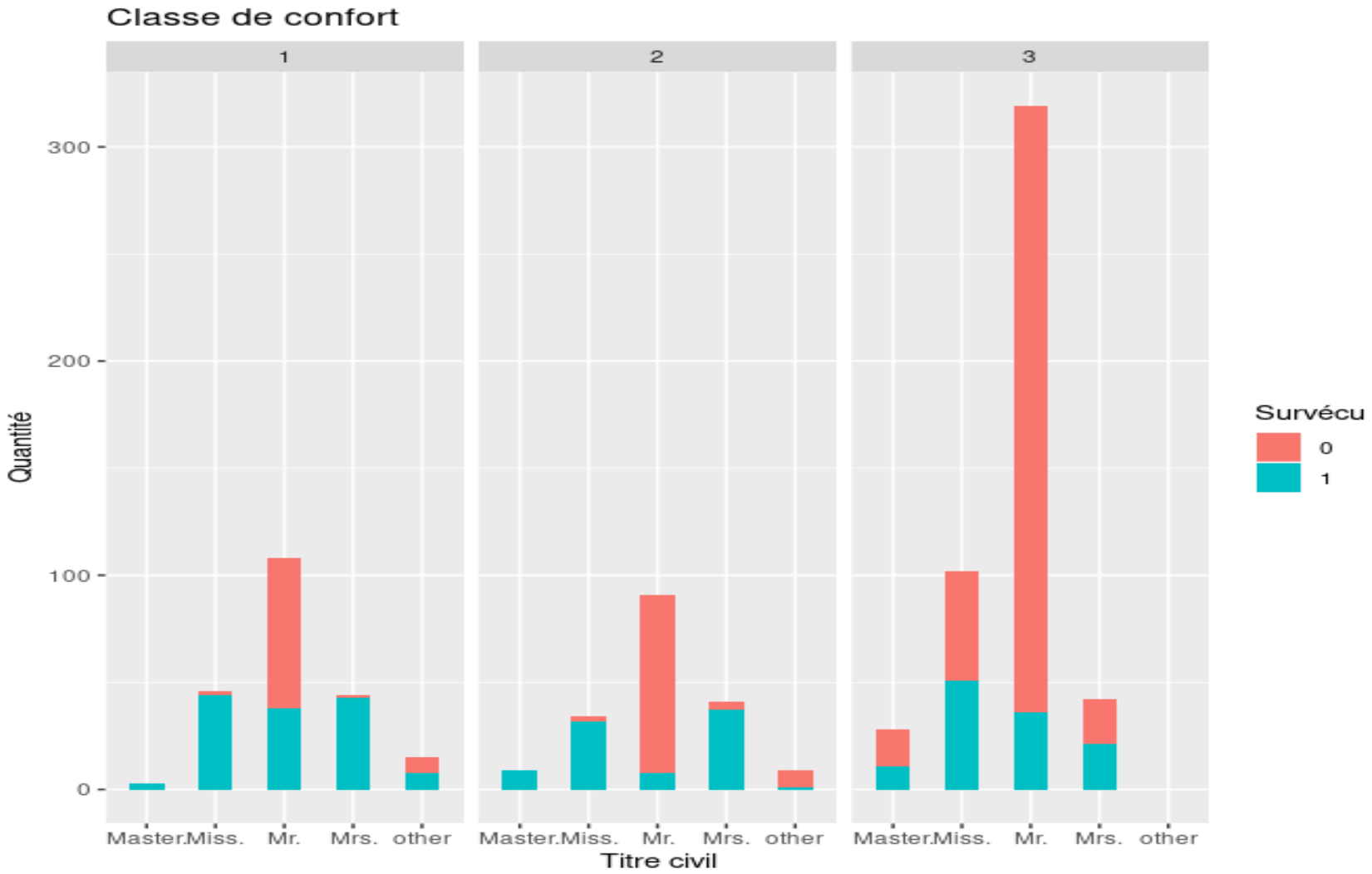


diagram obtained with David Langer's source code:

ggplot(train, aes(x = Title, fill = Survived)) stat_count(width = 0.5)
facet_wrap(~Pclass) ggtitle("Classe de confort")
xlab("Titre civil") ylab("Quantité") labs(fill = "Survécu")

Statistique descriptive univariée : quantile et médiane

Soit N la taille de notre échantillon :
`N = nrow(train); N`
nous retourne $\ll$ [1] 891 $\gg$.

Définition

Pour tout nombre $\alpha \in]0,1[$, *soient* m *et* d *les parties entière et décimale de* $\alpha \cdot (N+1)$. *Le* quantile d'ordre α, *noté* $Q_\alpha(x)$, *d'une série statistique est défini par :* $Q_\alpha(x) = x_{(m)} + d \cdot (x_{(m+1)} - x_{(m)})$.

La **médiane** est le quantile $Q_{\frac{1}{2}}(x)$. Pour l'age des passagers,

```
> quantile(train$Age[is.na(train$Age)==FALSE], 0.5, type=6)
50%
 28
```

et vous l'obtenez aussi par la commande
`median(train$Age[is.na(train$Age)==FALSE])`

Statistique descriptive univariée : quantile et quartiles

Le **quartile inférieur** est le quantile $Q_{\frac{1}{4}}(x)$; le **quartile supérieur** est le quantile $Q_{\frac{3}{4}}(x)$.

```
> quantile(train$Age[is.na(train$Age)==FALSE], 0.25, type=6)
25%
 20
> quantile(train$Age[is.na(train$Age)==FALSE], 0.75, type=6)
75%
 38
```

Sans spécification du paramètre α de $Q_\alpha(x)$, la commande `quantile` affiche les cinq quartiles :

```
> quantile(train$Age[is.na(train$Age)==FALSE], type=6)
   0%   25%   50%   75%  100%
 0.42 20.00 28.00 38.00 80.00
```

Nous pouvons distinguer les survivants (valeur 1) dans ce tableau :

```
> quantile(train$Age[is.na(train$Age)==FALSE & train$Survived==1], type=6)
   0%   25%   50%   75%  100%
 0.42 19.00 28.00 36.25 80.00
> quantile(train$Age[is.na(train$Age)==FALSE & train$Survived==0], type=6)
  0%  25%  50%  75% 100%
   1   21   28   39   74
```

Statistique descriptive univariée : Médian et moyenne

Nous notons la différence entre la médiane et la **moyenne arithmétique** que nous obtenons par la commande `mean`.

```
> mean(train$Fare)
[1] 32.20421
> median(train$Fare)
[1] 14.4542
> summary(train$Fare)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00    7.91   14.45   32.20   31.00  512.33
```

Dans notre échantillon, la moyenne arithmétique excède même le quartile supérieur.

Avec la **variance** $s^2(x) = \frac{1}{n}\sum_{j=1}^{n}(x_j - \text{mean}(x))^2$ (commande : `var`) et sa racine l'**écart-type** $\sqrt{s^2(x)}$ (commande : `sd`), nous mesurons combien notre échantillon dévie de la moyenne arithmétique. `sd(train$Fare)` nous retourne 49.69.

Exercice 6

Pour les prix des billets, calculez

1. le minimum (commande : `min`),
2. le maximum (commande : `max`),
3. la médiane,
4. le quartile inférieur,
5. le quartile supérieur,
6. la moyenne arithmétique et
7. l'écart-type

séparément

- **pour les passagers qui ont survécu,**
- **pour les passagers qui n'ont pas survécu.**

Un autre jeu de donnés, intégré dans la console *R*

```
# Utilisez les commands suivantes pour charger le jeu de données
# ≪ Seatbelts ≫ dans votre console R :

require(stats)
data(Seatbelts)
Seatbelts = data.frame( Year=floor(time(Seatbelts)),
                        Month=factor(cycle(Seatbelts), labels=month.abb),
                        Seatbelts)
attach(Seatbelts)

# Ce jeu de données contient pour chaque mois des années 1969 à 1984
# le nombre de conducteurs morts ou grièveme blessés au Royaume-Uni
# de la Grande Bretagne et l'Irlande du Nord. À partir du 31 janvier
# 1983, le port de la ceinture de sécurité était rendu obligatoire.
```

Le jeu de données ≪ Seatbelts ≫

Vous trouverez les données suivantes
dans les colonnes du jeu de données :

DriversKilled	conducteurs de voiture morts
drivers	conducteurs de véhicule morts ou grièveme blessés
front	passagers en front de voiture morts ou grièveme blessés
rear	passagers en arrière de voiture morts ou grièveme blessés
kms	distance parcourue
PetrolPrice	prix du carburant
VanKilled	conducteurs de camionette morts ou grièveme blessés
law	0 si le port de la ceinture de sécurité était facultatif, 1 si le port de la ceinture de sécurité était obligatoire pendant le mois concerné.

Exercice 6 bis

 # Calculez la moyenne
 # du nombre de conducteurs de voiture morts, séparément :

1. Pour les mois pendant lesquels le port de la ceinture de sécurité était facultatif,
2. pour les mois pendant lesquels le port de la ceinture de sécurité était obligatoire.

Note pratique

Si nous sommes uniquement intéressés à analyser un seul jeu de données, alors nous pouvons le spécifier par la commande `attach`. Après avoir tapé

```
attach(train)
```

nous pouvons donc nous débarasser de tous les préfixes

$$\ll \text{train\$} \gg$$

dans tous les transparents précédents.
C'est ce que nous allons faire pour la suite.

Statistique descriptive bivariée

La statistique bivariée nous permet de comparer deux variables du même jeu de données
(nous avons attaché le jeu de données ≪ train ≫).

```
> table(Pclass, Survived)
      Survived
Pclass   0   1
     1  80 136
     2  97  87
     3 372 119
>
```

```
> table(Embarked, Survived)
          Survived
Embarked   0   1
         0   2
       C  75  93
       Q  47  30
       S 427 217
```

```
> table(Embarked,Pclass)
          Pclass
Embarked   1   2   3
         2   0   0
       C  85  17  66
       Q   2   3  72
       S 127 164 353
```

Statistique descriptive bivariée : Le tableau de contingence

```
> addmargins(table(Pclass, Survived))
      Survived
Pclass   0   1 Sum
   1    80 136 216
   2    97  87 184
   3   372 119 491
   Sum 549 342 891
```

Nous pouvons sommer chaque ligne et colonne de notre dernier tableau, et obtenons un **tableau de contingence** :

	Survived $= 0$	Survived $= 1$	Total
Pclass $= 1$	$n_{1,1}$	$n_{1,2}$	$n_{1,\bullet}$
Pclass $= 2$	$n_{2,1}$	$n_{2,2}$	$n_{2,\bullet}$
Pclass $= 3$	$n_{3,1}$	$n_{3,2}$	$n_{3,\bullet}$
Total	$n_{\bullet,1}$	$n_{\bullet,2}$	N

En général, pour deux variables X, Y, le tableau de contingence est de la forme

	$Y = y_1$	$\dots$	$Y = y_q$	Total
$X = x_1$	$n_{1,1}$	$\dots$	$n_{1,2}$	$n_{1,\bullet}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$X = x_p$	$n_{p,1}$	$\dots$	$n_{p,q}$	$n_{p,\bullet}$
Total	$n_{\bullet,1}$	$\dots$	$n_{\bullet,q}$	N

Statistique descriptive bivariée : La covariance

Le tableau de contingence nous permet de calculer la **covariance** :

	$Y = y_1$	$\dots$	$Y = y_q$	Total
$X = x_1$	$n_{1,1}$	$\dots$	$n_{1,2}$	$n_{1,\bullet}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$X = x_p$	$n_{p,1}$	$\dots$	$n_{p,q}$	$n_{p,\bullet}$
Total	$n_{\bullet,1}$	$\dots$	$n_{\bullet,q}$	N

$$\mathbb{C}ov(X, Y) = \sum_{i,j=1}^{i=p,\, j=q} (x_i - \text{mean}(X))\,(y_j - \text{mean}(Y))\,\frac{n_{i,j}}{N}.$$

La commande

```
cov(Pclass, Survived)
```

nous calcule

$$\mathbb{C}ov(\text{Pclass, Survived}) = -0.1377029.$$

La covariance et l'indépendance

Divisons par N le tableau de contingence, afin d'obtenir un tableau de fréquences :

	$Y = y_1$	$\dots$	$Y = y_q$	Total
$X = x_1$	$n_{1,1}/N$	$\dots$	$n_{1,2}/N$	$n_{1,\bullet}/N$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$X = x_p$	$n_{p,1}/N$	$\dots$	$n_{p,q}/N$	$n_{p,\bullet}/N$
Total	$n_{\bullet,1}/N$	$\dots$	$n_{\bullet,q}/N$	1

Proposition

Supposons que les deux variables aléatoires X et Y soient indépendantes. Alors,

- *pour chaque $1 \leq i \leq p$ et $1 \leq j \leq q$, nous avons*

$$\frac{n_{i,j}}{N} = \frac{n_{i,\bullet}}{N} \cdot \frac{n_{\bullet,j}}{N}.$$

- *et la covariance $\mathbb{C}ov(X, Y)$ est zéro.*

Le coefficient de corrélation linéaire

Il est improbable d'obtenir un échantillon (non fabriqué, mais honnêtement mesuré) avec des variables X, Y telle que la covariance $\mathbb{C}ov(X, Y)$ est zéro. Donc il faudra plutôt regarder quand elle est $\ll$ quasiment zéro $\gg$. Et nous devons pouvoir comparer avec la covariance d'autres paires de variables. Ceci est possible avec **le coefficient de corrélation linéaire**,

$$r(X, Y) = \frac{\mathbb{C}ov(X, Y)}{\sqrt{s^2(X)}\sqrt{s^2(Y)}},$$

où $\sqrt{s^2(X)}$ est l'écart-type de X. Nous pouvons calculer $r(X, Y)$ par la commande `cor`. Et `cor(PassengerId, Survived)` nous donne -0.005006661, tandis que

```
> cov(Pclass, Survived)
[1] -0.1377029
> cov(Pclass, Survived)/(sd(Pclass)*sd(Survived))
[1] -0.338481
> cor(Pclass, Survived)
[1] -0.338481
```

Exercice 7

Calculez les rapports (≪ ratios ≫) de coefficients de corrélation

1. $\frac{r(\text{Pclass,Survived})}{r(\text{PassengerId,Survived})}$
2. $\frac{r(\text{Fare,Survived})}{r(\text{PassengerId,Survived})}$
3. $\frac{r(\text{Age,Survived})}{r(\text{PassengerId,Survived})}$
4. $\frac{r(\text{SibSp,Survived})}{r(\text{PassengerId,Survived})}$
5. $\frac{r(\text{Parch,Survived})}{r(\text{PassengerId,Survived})}$

Notons que le nom de la colonne ≪ Parch ≫ signifie ≪ Parents or children ≫, et cette colonne compte donc le nombre d'enfants ou de parents du passager qui étaient à bord.

Propriétés de la covariance

En passant de la variance $s^2(X)$ de l'échantillon à la variance $\mathbb{V}ar(X)$ de la population entière, nous arrivons aux propriétés suivantes :

Proposition

- $\mathbb{C}ov(X, Y) = \mathbb{C}ov(Y, X)$,
- $\mathbb{C}ov(X, X) = \mathbb{V}ar(X)$,
- $\mathbb{V}ar(X + Y) = \mathbb{V}ar(X) + 2\mathbb{C}ov(X, Y) + \mathbb{V}ar(Y)$,
- *Pour tous* $a, b, c, d \in \mathbb{R}$, *nous avons*

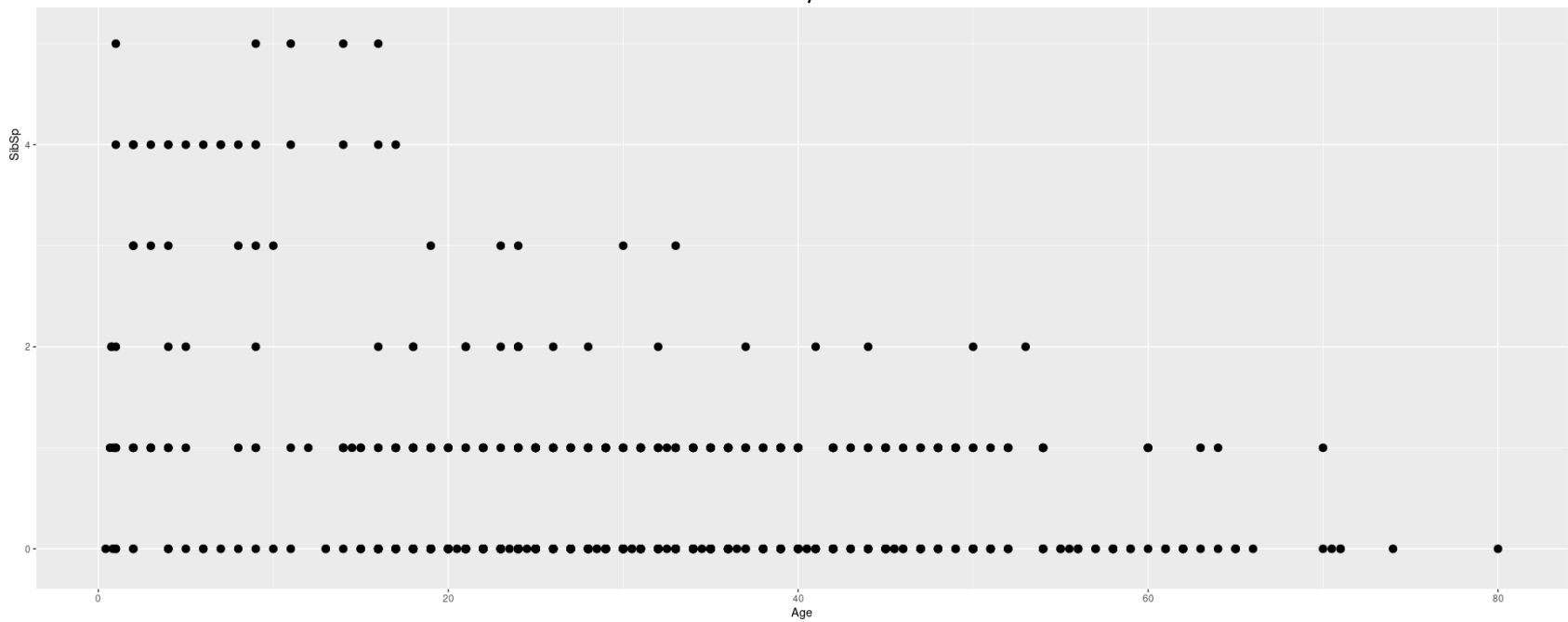
$$\mathbb{C}ov(aX + b, cY + d) = ac\mathbb{C}ov(X, Y)$$

- *et* $|\mathbb{C}ov(X, Y)| \leq \sqrt{\mathbb{V}ar(X)\mathbb{V}ar(Y)}$.

Nous déduisons de la dernière inéquation que $r(X, Y) = \frac{\mathbb{C}ov(X,Y)}{\sqrt{s^2(X)}\sqrt{s^2(Y)}}$ est contenu dans l'intervalle $]-1, 1[$.

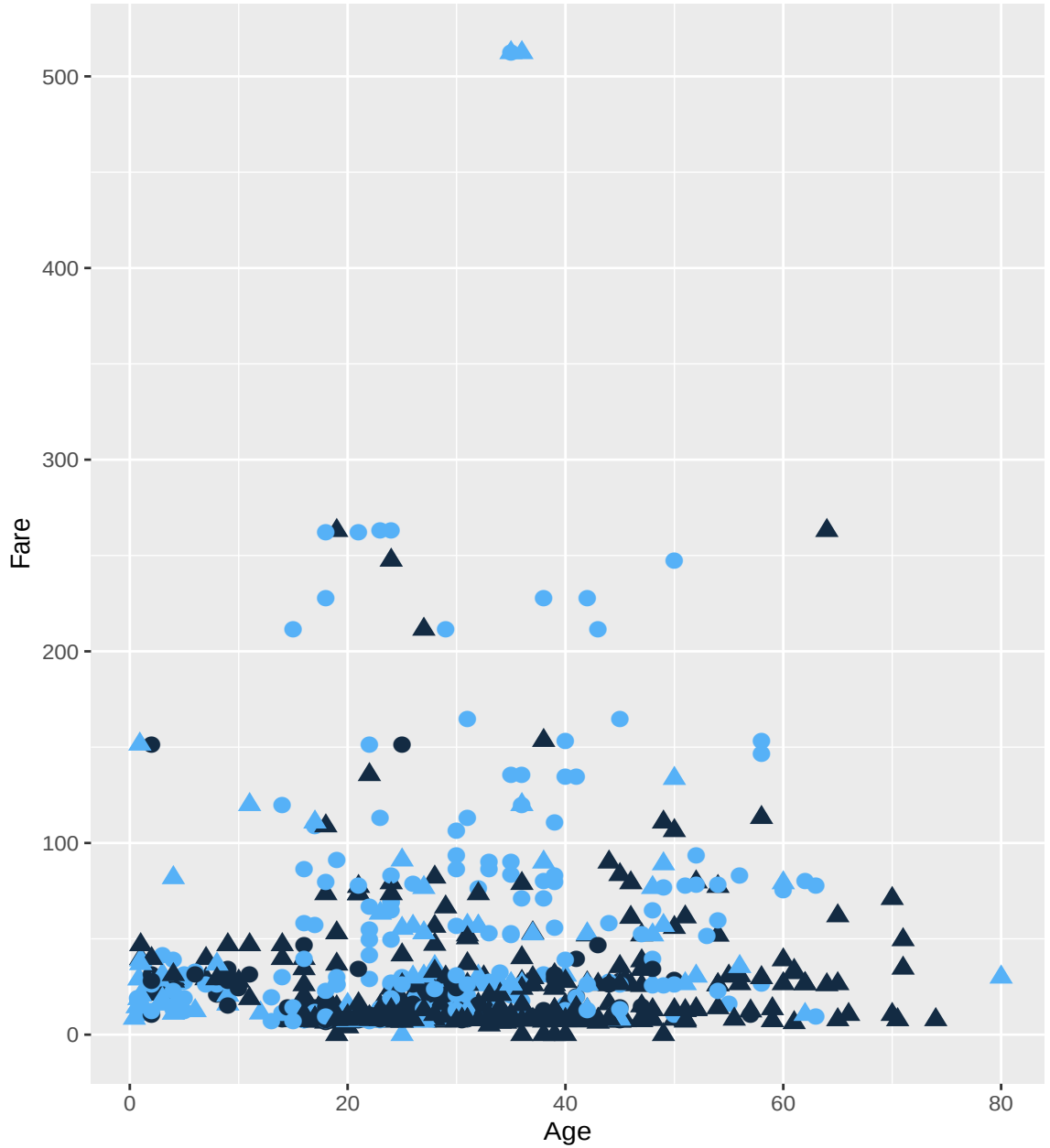
Visualisation bivariée

Par exemple :
Comme abscisse x, nous prenons la variable $\ll$ Age $\gg$, et comme ordonnée y, nous prenons la variable $\ll$ SibSp $\gg$, qui compte le nombre de soeurs, frères ou époux/épouses :




```
library(ggplot2)
ggplot(train, aes(x=Age,y=SibSp))+geom_point(size=3)
```

Visualisation multivariée




```
ggplot(train,
aes(
x=Age,
y=Fare,
color=Survived,
shape=Gender))
+geom_point
(size=3)
```

Exercice 8

Faites dessiner un diagramme analogue au diagramme ci-dessus, mais remplacez l'age par la taille de la famille du passager présente à bord. Elle se calcule par la somme `Parch+SibSp` selon la description de ces deux variables données dans les transparents précédents.

Une autre base de données : les fleurs iris

Iris setosa

Photo : Radomil Binek

Iris versicolor

Photo : D. Langlois

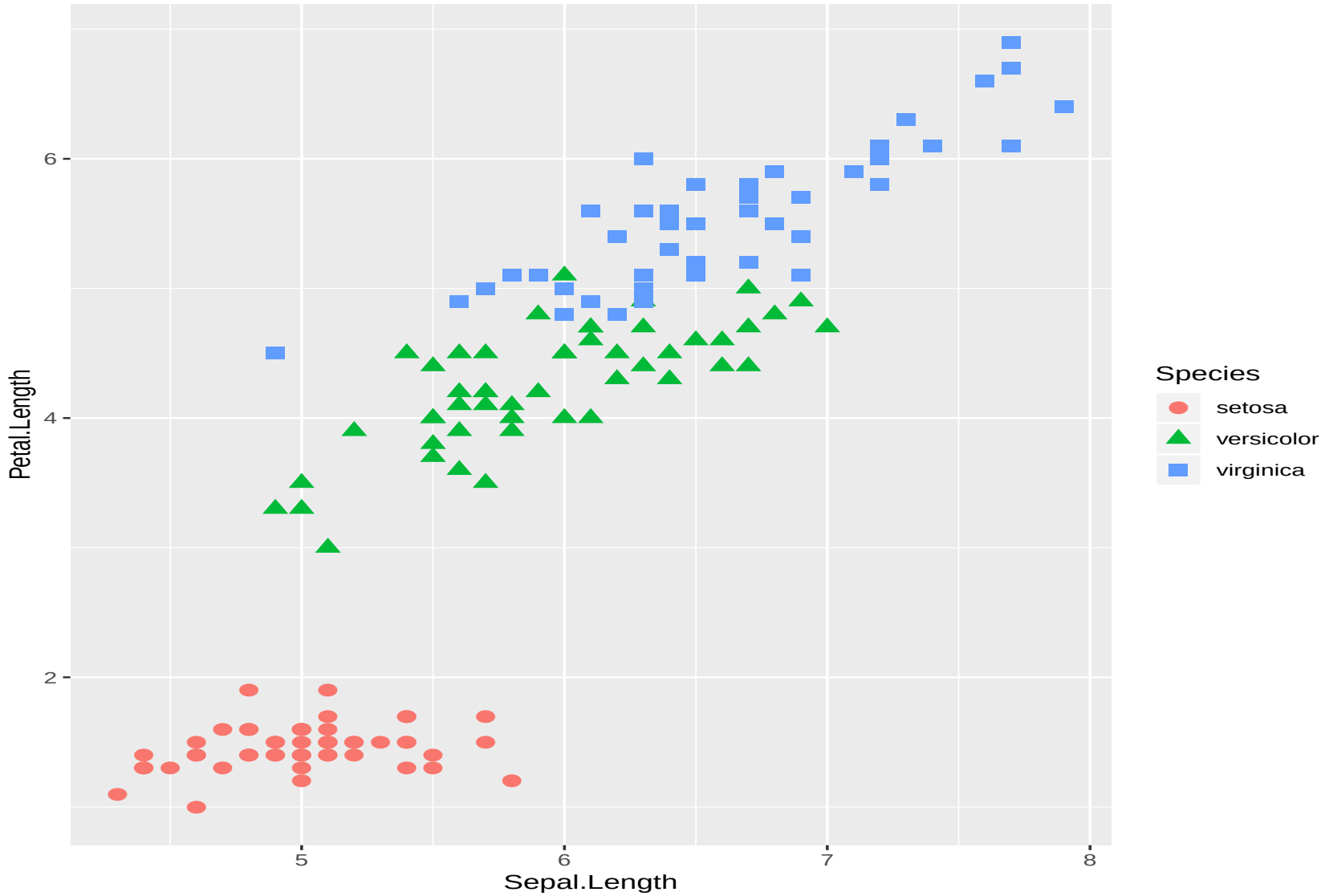
Iris virginica

Photo : Frank Mayfield

Comment distinguer ces trois espèces de la fleur, de manière automatisée ?

Le botaniste Edgar Anderson a mesuré 50 exemplaires de chaque espèce.

Longeur de sépales *versus* longueur de pétales




```
ggplot(iris, aes(x=Sepal.Length, y=Petal.Length,
color=Species, shape=Species)) +geom_point(size=3)
```

Regression linéaire

Dans le diagramme ci-dessus, nous avons vu que l'espèce $\ll$ virginica $\gg$ admet des coordonnées sépales-pétales organisés autour d'une ligne. Nous aimerions faire dessiner cette ligne. $X =$Sepal.Length ; $Y =$ Petal.Length ; alors pour

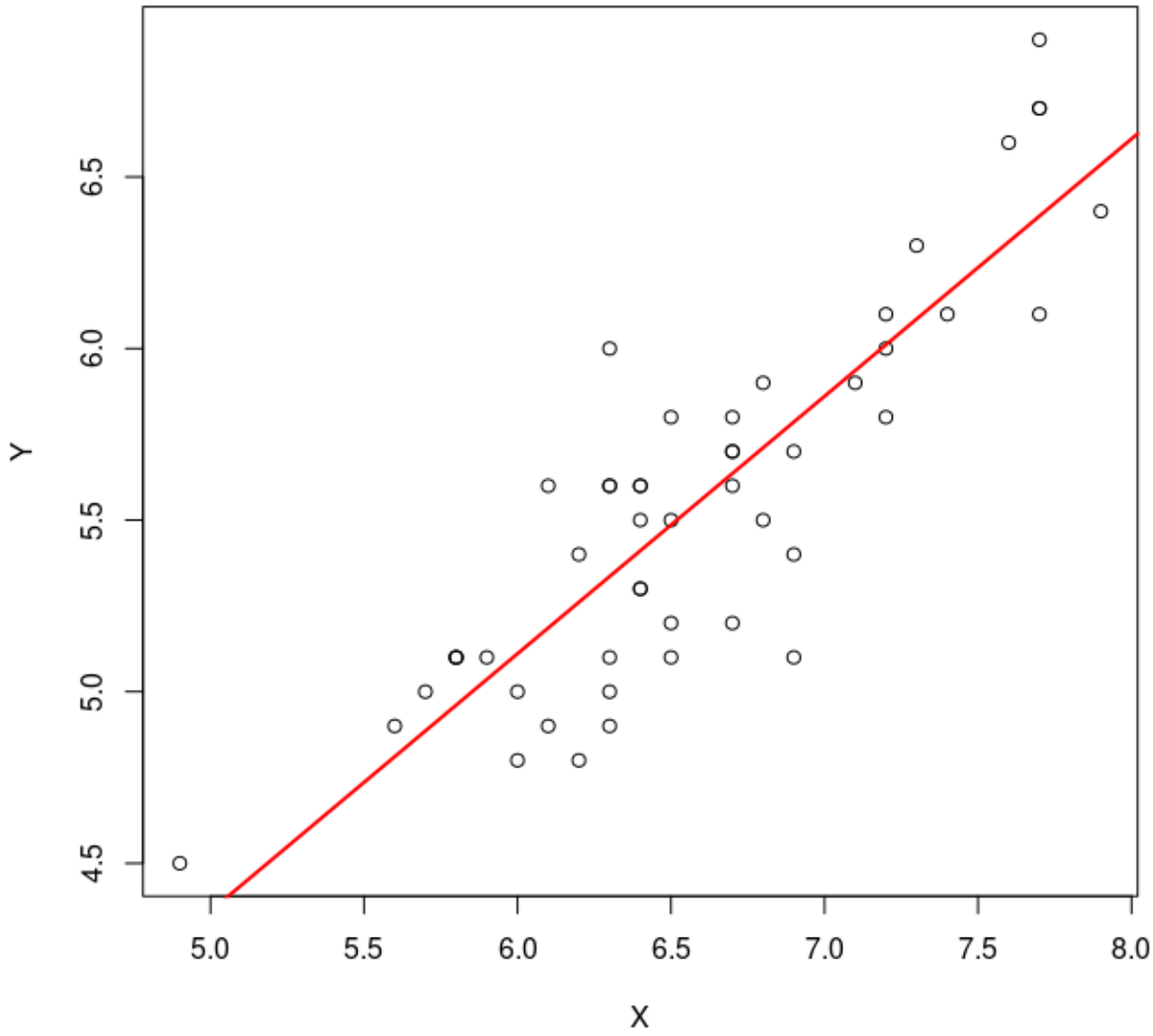
$$\beta_1 = \frac{\mathbb{C}ov(X,Y)}{\mathbb{V}ar(X)} \text{ et } \beta_0 = \mu(Y) - \beta_1\mu(X),$$

où μ est la moyenne arithmétique, obtenons une ligne $Y = \beta_0 + \beta_1 X$ qui minimise sa distance aux points du diagramme.

```
> X = iris$Sepal.Length[iris$Species=="virginica"]
> Y = iris$Petal.Length[iris$Species=="virginica"]
> beta1 = cov(X,Y)/var(X)
> beta0 = mean(Y)-beta1*mean(X)
> beta1
[1] 0.7500808
> beta0
[1] 0.610468
```

```
> droiteVirginica <- lm(Y~X, data=iris)
> coef(droiteVirginica)
(Intercept)           X
  0.6104680   0.7500808
```

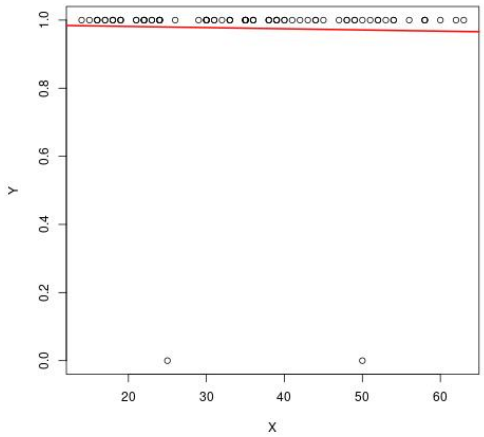
Regression linéaire pétale/sépale (virginica)



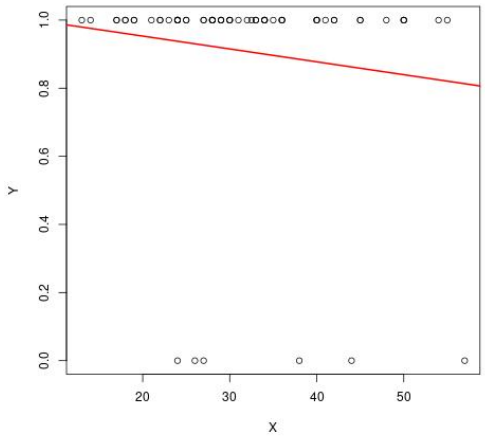

```
> plot.new()
> plot(X,Y)
> abline(coef(droiteVirginica),col="red",lwd=2)
```

Un modèle linéaire pour la survie des passagers

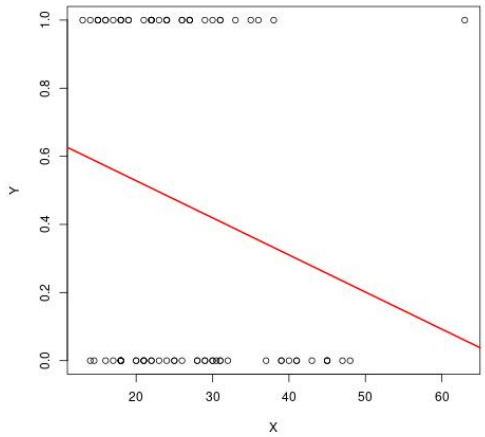
1ère classe



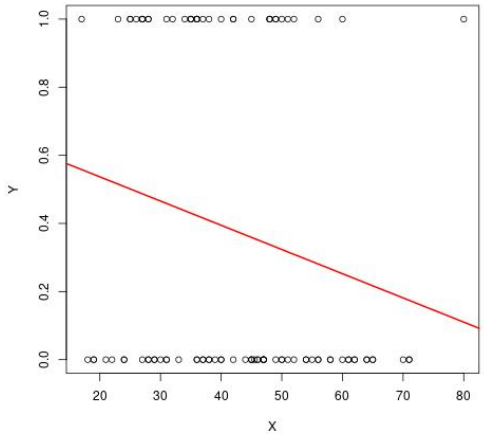
2nde classe

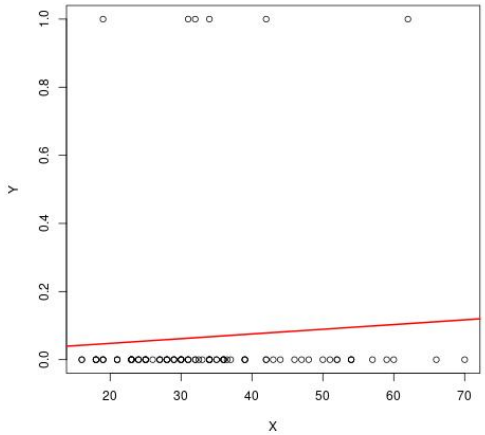


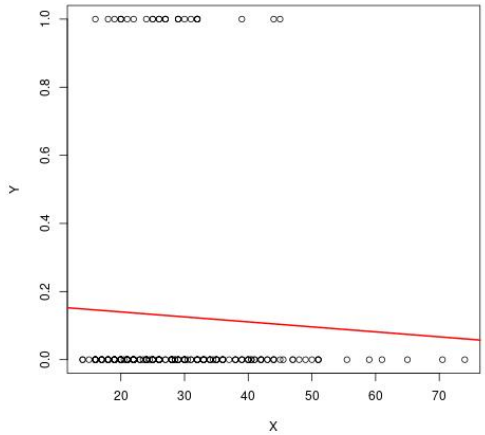
3ème classe



Passagères adolescentes et adultes







Passagers masculins adolescents et adultes

Exercice 9

Le premier des six diagrammes ci-dessus a été créé avec les commandes

```
 jpeg("1f.jpg")
Y=Survived[is.na(Age)==FALSE & Age > 12.9 &
Pclass == 1 & Gender == "female"]
X=Age[is.na(Age)==FALSE & Age > 12.9 & Pclass == 1 &
Gender == "female"]
plot.new()
plot(X,Y)
droite= lm(Y~X)
abline(coef(droite), col="red", lwd=2)
dev.off()
```

Faites dessiner de nouveau ces six diagrammes.

Exercice 9 bis

Dans le jeu de données ≪ Seatbelts ≫, calculez les coefficients de la droite de régression linéaire entre les passagers morts ou grièveme blessés en front de voiture contre ceux en arrière de voiture. Vérifiez ces coefficients en les calculant

- avec la fonction dédiée intégrée dans la console *R*,
- et explicitement avec la formule en termes de variance, covariance etc.

Faites dessiner cette droite de régression sur le fond des points de données qu'elle modélise (le nuage de points constitué par le nombre de passagers morts ou grièveme blessés en front de voiture contre le nombre de ceux en arrière de voiture).

Un modèle constant pour les passagers enfants

Pour les enfants parmis les passagers, nous pouvons simplement nous servir de la moyenne arithmétique :

```
enfants1 = mean(Survived[is.na(Age)==FALSE & Age < 13
& Pclass == 1])
enfants2 = mean(Survived[is.na(Age)==FALSE & Age < 13
& Pclass == 2])
enfants3 = mean(Survived[is.na(Age)==FALSE & Age < 13
& Pclass == 3])
```

```
enfants <- c(enfants1, enfants2, enfants3)
```

Ceci nous retourne la collection des trois valeurs suivantes :

```
[1]       0.7500000       1.0000000       0.4166667
```

Coefficients de la régression linéaire

Pour les passagères adolescentes et adultes en troisième classe,
Y=Survived[is.na(Age)==FALSE & Age $>$ 12.9 & Pclass $==$ 3 &
Gender $==$ "female"]
X=Age[is.na(Age)==FALSE & Age $>$ 12.9 & Pclass $==$ 3 &
Gender $==$ "female"]

les coefficients

$$\beta_1 = \frac{\mathbb{C}ov(X,Y)}{\mathbb{V}ar(X)} \text{ et } \beta_0 = \mu(Y) - \beta_1\mu(X),$$

où μ est la moyenne arithmétique, déterminent une ligne
$Y = \beta_0 + \beta_1 X$ qui minimise sa distance aux points de notre jeu de données.

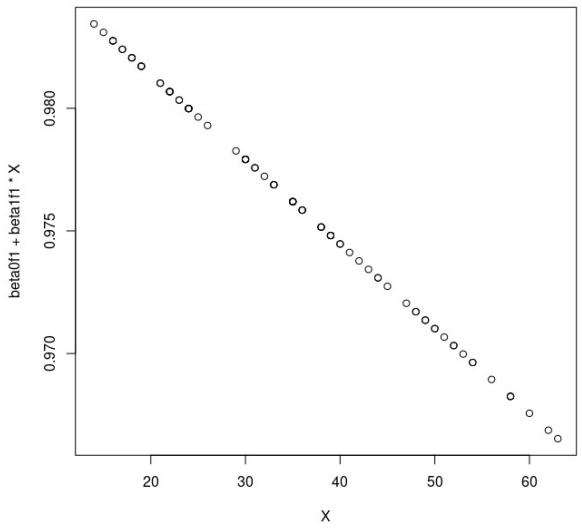
```
beta1f3 = cov(X,Y)/var(X)
beta0f3 = mean(Y) -beta1f3*mean(X)
```

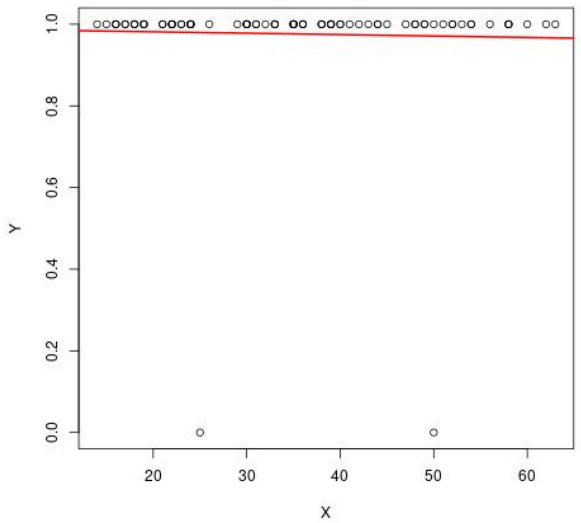
nous donne `beta0f3` = 0.7461679, `beta1f3` = -0.01089614.

Exercice 10

Avec les commandes `plot.new(); plot(X, beta0f1+beta1f1*X)` nous vérifions que les coéfficients calculés reproduisent la ligne de régression que vous avez fait visualiser dans l'exercice précédent :

`plot(X, beta0f1+beta1f1*X)` Passagères 1ère classe





Calculez les coefficients pour les lignes de régression des autres cinq diagrammes de l'exercice précédent, et faites la comparaison graphique analogue.

Un modèle constant pour les passagers d'age inconnu

Pour les passagers d'age inconnu, nous allons simplement nous servir de la moyenne arithmétique :

```
inconnuf1 = mean(Survived[is.na(Age)==TRUE & Pclass
== 1 & Gender =="female"])
inconnuf2 = mean(Survived[is.na(Age)==TRUE & Pclass
== 2 & Gender =="female"])
inconnuf3 = mean(Survived[is.na(Age)==TRUE & Pclass
== 3 & Gender =="female"])
inconnum1 = mean(Survived[is.na(Age)==TRUE & Pclass
== 1 & Gender =="male"])
inconnum2 = mean(Survived[is.na(Age)==TRUE & Pclass
== 2 & Gender =="male"])
inconnum3 = mean(Survived[is.na(Age)==TRUE & Pclass
== 3 & Gender =="male"])

inconnuf <- c(inconnuf1, inconnuf2, inconnuf3)
inconnum <- c(inconnum1, inconnum2, inconnum3)
```

Assemblage du modèle comme objet dans R

Nous pouvons maintenant assembler notre modèle :
Moyennes arithmétiques pour les enfants,
coefficients de régression linéaire pour les autres :

```
> beta1f <- c(beta1f1, beta1f2, beta1f3)
> beta0f <- c(beta0f1, beta0f2, beta0f3)
>
> beta1m <- c(beta1m1, beta1m2, beta1m3)
> beta0m <- c(beta0m1, beta0m2, beta0m3)
>
> pronostic <- data.frame(enfants, beta1f, beta0f, beta1m, beta0m, inconnuf, inconnum)
> pronostic
    enfants        beta1f    beta0f       beta1m     beta0m  inconnuf   inconnum
1 0.7500000 -0.0003451132 0.9882694 -0.007111145 0.67894023 1.0000000 0.23809524
2 1.0000000 -0.0037778753 1.0285233  0.001380862 0.02028505 1.0000000 0.22222222
3 0.4166667 -0.0108961381 0.7461679 -0.001464046 0.16936137 0.5952381 0.09574468
```

 # Nous pouvons ensuite enregistrer l'ensemble structuré de nos
 # paramètres trouvés comme jeu de données sur le disque dur :

```
write.csv(pronostic, file="pronostic.csv")
```

 # Ayant sauvegardé notre code source dans un fichier en format
 # texte brut (≪.txt≫), nous pouvons quitter la séance R*

```
quit() # et charger ce jeu de données dans une nouvelle séance :
pronostic = read.csv("pronostic.csv")
```

Mettre en oeuvre le pronostic

```
> pronostiquer <- function(age, classe, genre) {
+   if (is.na(age)){
+         if (genre == "female") {
+            return(inconnuf[[classe]])
+         } else {
+            return(inconnum[[classe]])
+         }
+   } else if (age < 13) {
+         return(enfants[[classe]])
+   } else if (genre == "female") {
+         regressionLineaire = beta1f[[classe]]*age+beta0f[[classe]]
+         return(max(regressionLineaire,0))
+     } else {
+         regressionLineaire = beta1m[[classe]]*age+beta0m[[classe]]
+         return(max(regressionLineaire,0))
+   }
+ }
> pronostiquer(NA,3,"female")
[1] 0.5952381
> pronostiquer(12,3,"female")
[1] 0.4166667
> pronostiquer(13,3,"female")
[1] 0.6045181
> pronostiquer(73,3,"female")
[1] 0
> pronostiquer(63,3,"female")
[1] 0.05971123
```

Tester le pronostic

```
for ( classe in 1:3 ){
    print("En classe "); print(classe);
    print(" nous pronostiquons les chances suivantes.");
    print("Passagères d'age inconnu : ");
    print(pronostiquer(NA,classe,"female"));
    print("Mineures : "); print(pronostiquer(12,classe,"female"));
    print("Passagères d'age 13 : "); print(pronostiquer(13,classe,"female"));
    print("Passagers masculins d'age inconnu : ");
    print(pronostiquer(NA,classe,"male"));
    print("Mineurs masculins : "); print(pronostiquer(12,classe,"male"));
    print("Passagers masculins d'age 13 : ");
    print(pronostiquer(13,classe,"male"));
}
```

```
"En classe "
1
" nous pronostiquons les chances suivantes."
"Passagères d'age inconnu : "
1
"Mineures : "
0.75
"Passagères d'age 13 : "
0.983783
"Passagers masculins d'age inconnu : "
0.2380952
"Mineurs masculins : "
0.75
"Passagers masculins d'age 13 : "
0.5864953
```

```
"En classe "
2
" nous pronostiquons les chances suivantes."
"Passagères d'age inconnu : "
1
"Mineures : "
1
"Passagères d'age 13 : "
0.9794109
"Passagers masculins d'age inconnu : "
0.2222222
"Mineurs masculins : "
1
"Passagers masculins d'age 13 : "
0.03823625
```

```
"En classe "
3
" nous pronostiquons les chances suivantes."
"Passagères d'age inconnu : "
0.5952381
"Mineures : "
0.4166667
"Passagères d'age 13 : "
0.6045181
"Passagers masculins d'age inconnu : "
0.09574468
"Mineurs masculins : "
0.4166667
"Passagers masculins d'age 13 : "
0.1503288
```

Comparer le pronostic à l'information disponible

```
survivalChance <- NULL
for (i in 1:nrow(train)) {
  + age = train[i, "Age"]
  + classe = train[i, "Pclass"]
  + genre = train[i, "Gender"]
  + survivalChance <- c(survivalChance,
    + pronostiquer(age, classe, genre))
+ }
```

Notre modèle est mieux corrélé à la survie que les colonnes du jeu de données :

```
cor(survivalChance, train$Survived)
[1] 0.6742172
cor(train$Pclass, train$Survived)
[1] -0.338481
```

Comparer notre modèle à la survie feminine

```
female <- NULL
for (i in 1:nrow(train)) {
  if ( train[i, "Gender"] == "female") {
    + female <- c(female, 1)
  + } else {
    + female <- c(female, 0)
  + }
+ }
```

Notre modèle est mieux corrélé avec la survie que le genre tout seul :

```
cor (female, train$Survived)
[1] 0.5433514
```

Toutefois, notre modèle est fortement corrélé avec les passagers féminins :

```
cor (female, survivalChance)
[1] 0.8011228
```

Exercice 11

Comparez notre modèle à la survie des passagers embarqués à Cherbourg, en calculant le coefficient de corrélation linéaire entre

- la chance de survie pronostiquée et
- une variable qui vaut 1 pour les passagers embarqués à Cherbourg, et 0 pour les autres passagers.

Appliquer le pronostic aux données test

Nous appliquons notre modèle au jeu de données test :

```
survivalChance <- NULL
for (i in 1:nrow(test)) {
  + age = test[i, "Age"]
  + classe = test[i, "Pclass"]
  + genre = test[i, "Gender"]
  + survivalChance <- c(survivalChance,
    + pronostiquer(age, classe, genre))
+ }
```

Nous rajoutons une colonne `test$SurvivalChance` au jeu de données test :

```
test$SurvivalChance <- survivalChance
```

Nous observons que notre prédiction donne une moyenne arithmétique très proche pour la survie quand nous comparons les deux jeux de données.

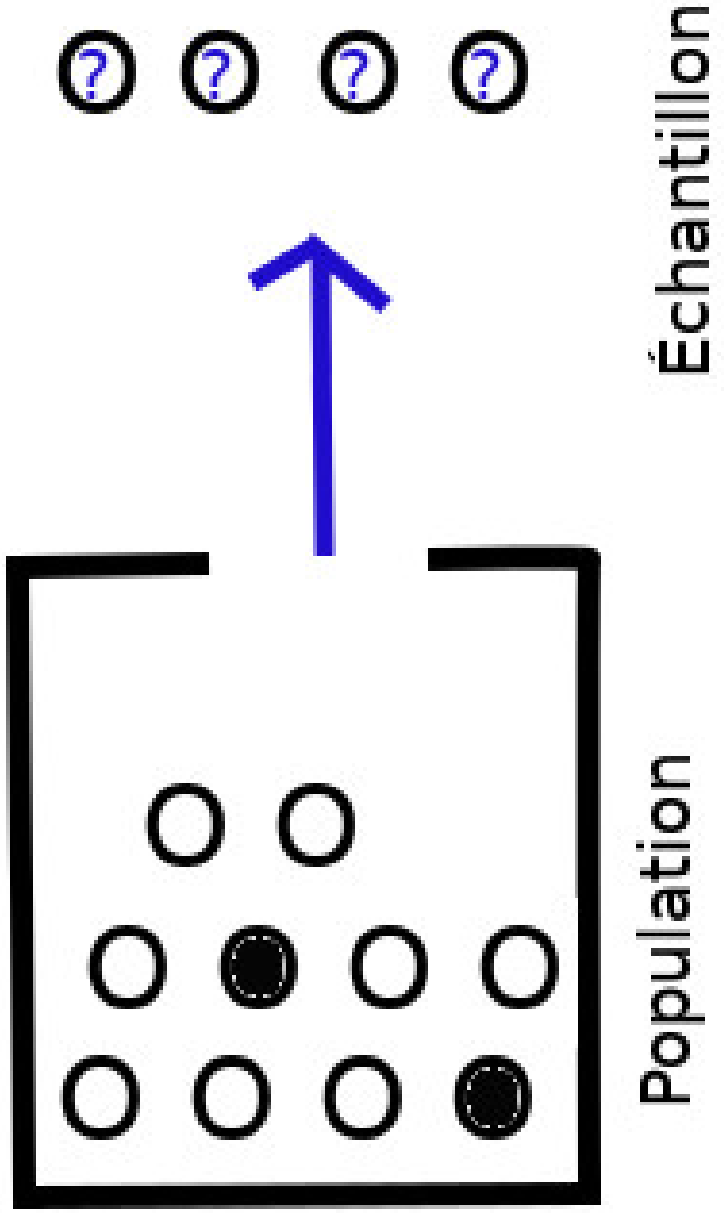
```
c( mean(test$SurvivalChance), mean(train$Survived))
[1] 0.3845285 0.3838384
```

Exercice 12

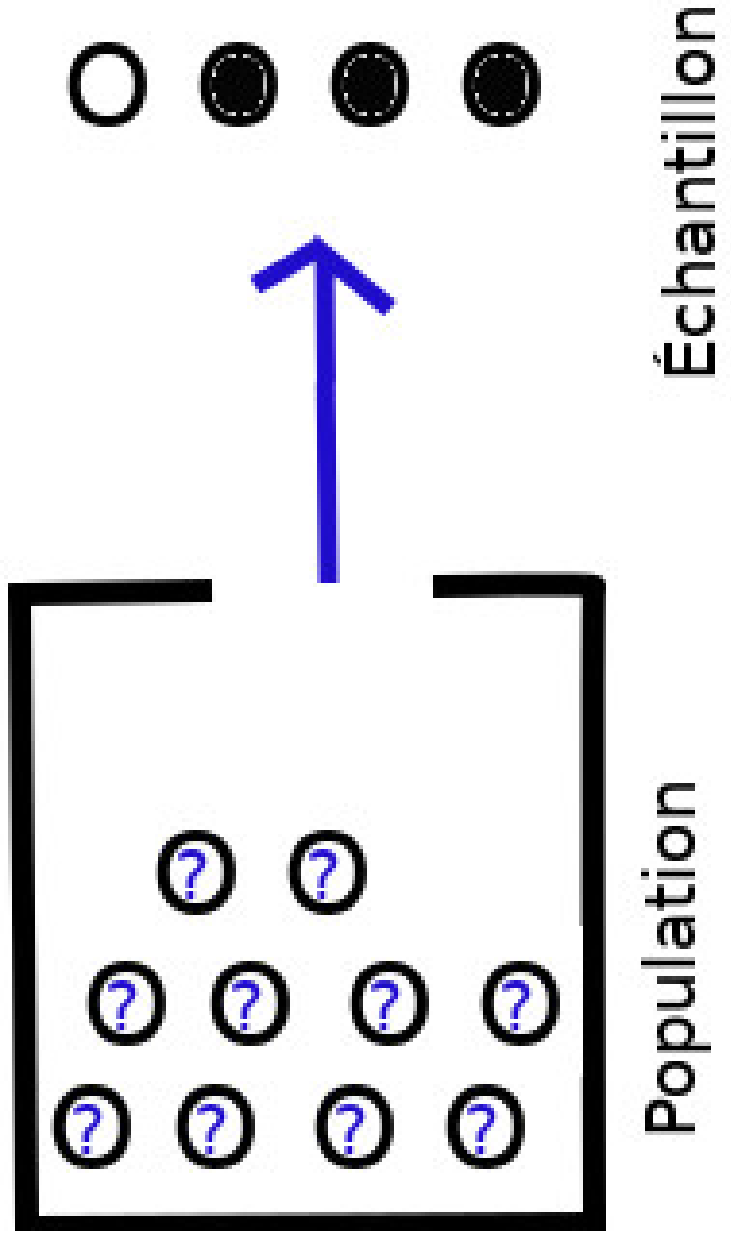
Visualisez les chances de survie pronostiquées en comparaison à quelques colonnes du jeu de données ≪ test ≫, de la manière que vous l'avez appris sur le jeu de données ≪ train ≫.

Statistiques inférentielles

Loi de probabilité :

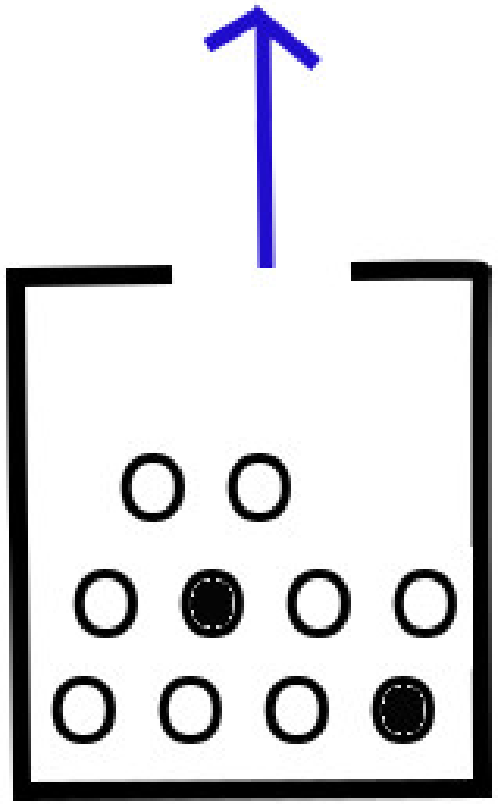


Statistiques inférentielles :

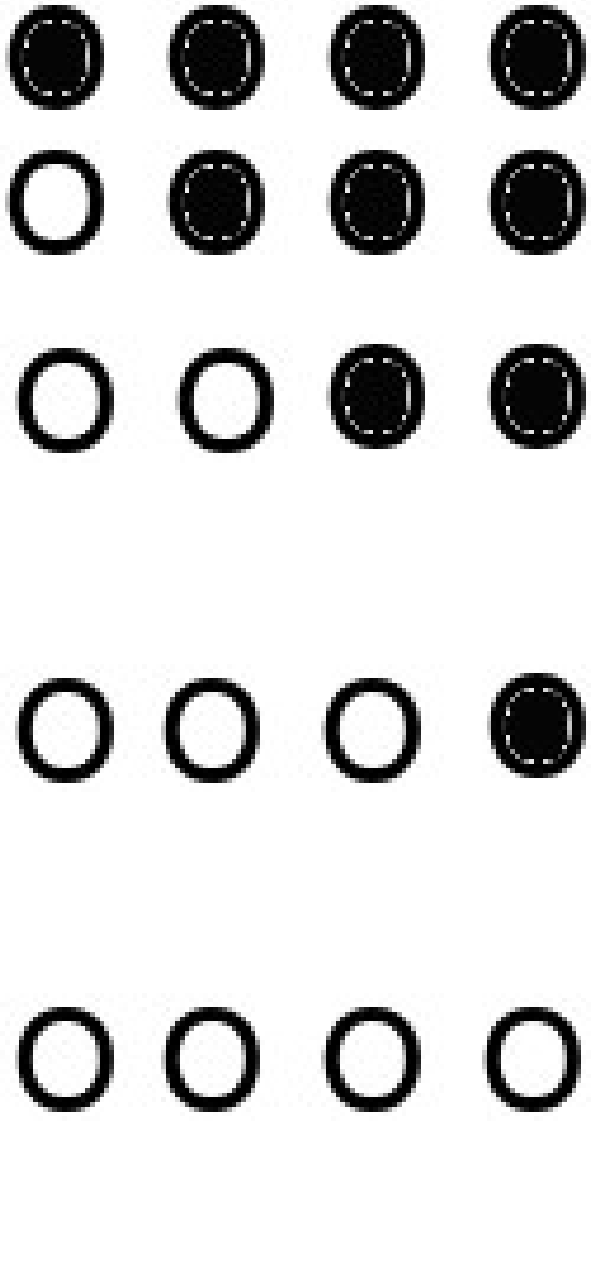


Tirage de 4 perles d'une population connue

Loi de probabilité :



Échantillon



impossible

impossible

$$\frac{\binom{8}{2}\cdot\binom{2}{2}}{\binom{10}{4}}$$

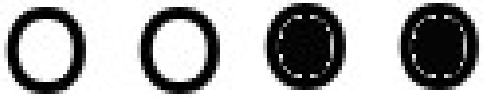
$$\frac{\binom{8}{3}\cdot\binom{2}{1}}{\binom{10}{4}}$$

$$\frac{\binom{8}{4}\cdot\binom{2}{0}}{\binom{10}{4}}$$

Tirage de 4 perles d'une population connue : Calcul

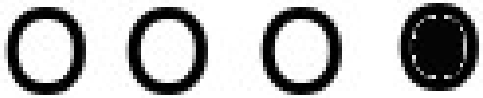
$$\binom{N}{K} = \frac{N!}{K! \cdot (N-K)!} = \frac{N \cdot (N-1) \cdot (N-2) \ldots \cdot 1}{K \cdot (K-1) \ldots \cdot 1 \cdot (N-K) \cdot (N-K-1) \ldots \cdot 1}$$

```
KparmiN <- function(N,K) {return
(factorial(N)/(factorial(K)*factorial(N-K)))}
```

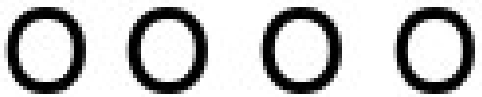


$$\binom{8}{2} \cdot \binom{2}{2} / \binom{10}{4} = 0.1333333$$

```
KparmiN(8,2)*KparmiN(2,2)/KparmiN(10,4)
```



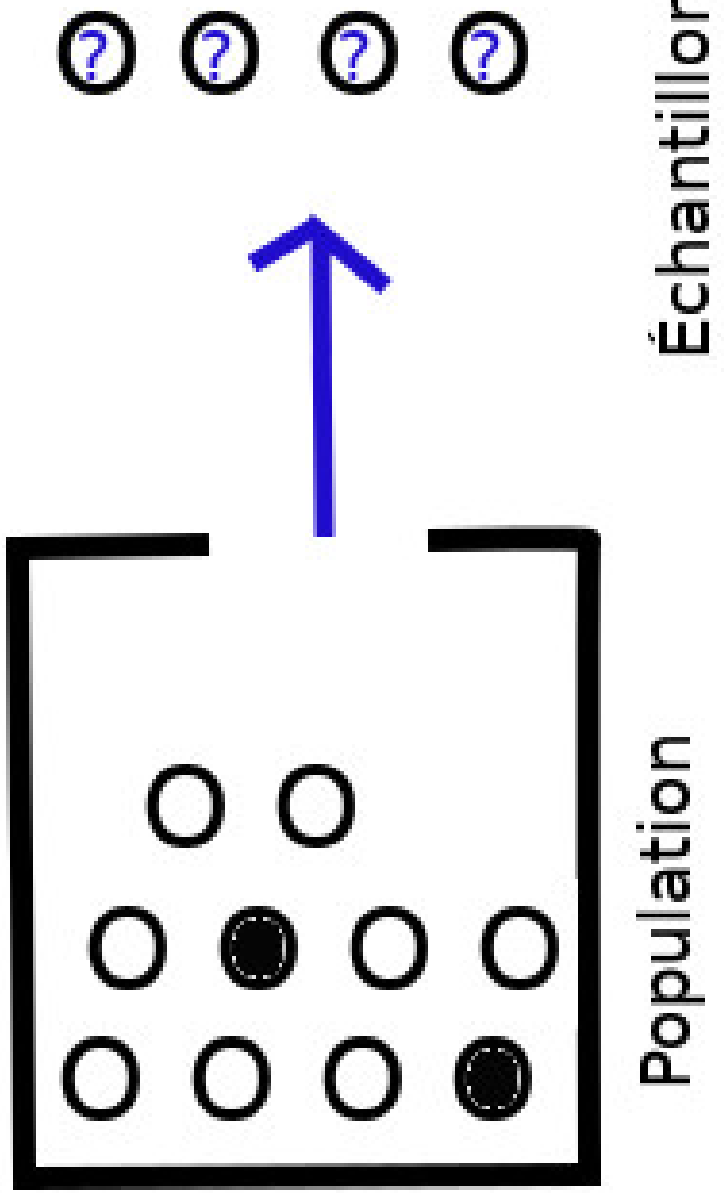
$$\frac{\binom{8}{3} \cdot \binom{2}{1}}{\binom{10}{4}}$$ nous donne 0.5333333



$$\frac{\binom{8}{4} \cdot \binom{2}{0}}{\binom{10}{4}}$$ nous donne 0.3333333

Tirage de 4 perles d'une population connue : Résumé

Loi de probabilité :



Si nous savons qu'il y a m perles blanches et n perles noires dans l'urne, alors la probabilité de recevoir x perles blanches dans un échantillon de k perles est

$$\frac{\binom{m}{x} \cdot \binom{n}{k-x}}{\binom{m+n}{k}}.$$

Cette formule s'évalue par la commande

```
dhyper(x, m, n, k)
```

Exercice 13

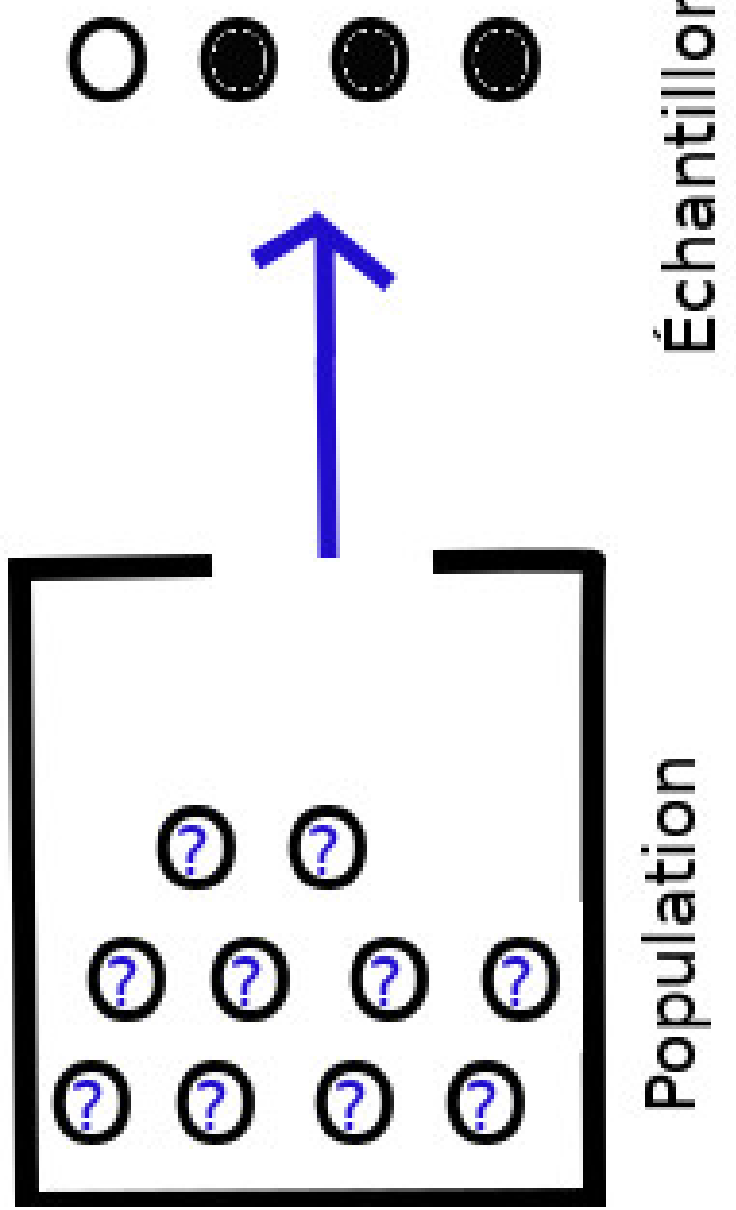
Calculez la probabilité pour recevoir x perles blanches dans un échantillon de 3 perles tirées d'une urne qui contient 4 perles blanches et 6 perles noires,

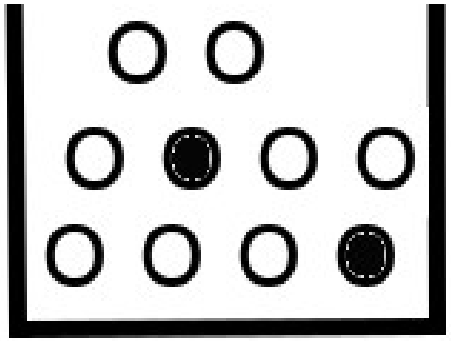
- en laissant x parcourir l'ensemble $\{0, 1, 2, 3\}$
- et en vous servant d'abord de l'implantation
  `KparmiN <- function(N,K) {return (factorial(N)/(factorial(K)*factorial(N-K)))}`
  pour évaluer $\begin{pmatrix} N \\ K \end{pmatrix}$ dans la formule $\dfrac{\begin{pmatrix} m \\ x \end{pmatrix} \cdot \begin{pmatrix} n \\ k-x \end{pmatrix}}{\begin{pmatrix} m+n \\ k \end{pmatrix}}$, et
  puis de la commande `dhyper(x, m, n, k)`
  afin de contrôler votre calcul.

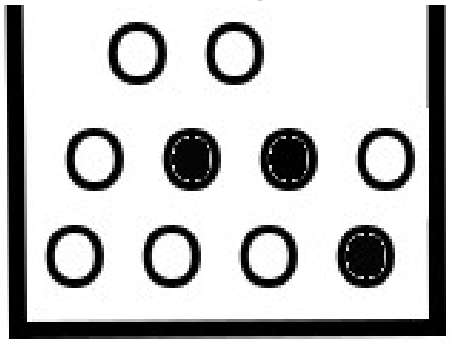
Vérifiez que la somme sur ces quatre probabilités, où x parcourt l'ensemble $\{0, 1, 2, 3\}$, vaut 1.

Tirage de 4 perles d'une population **inconnue**

Urne inconnue :



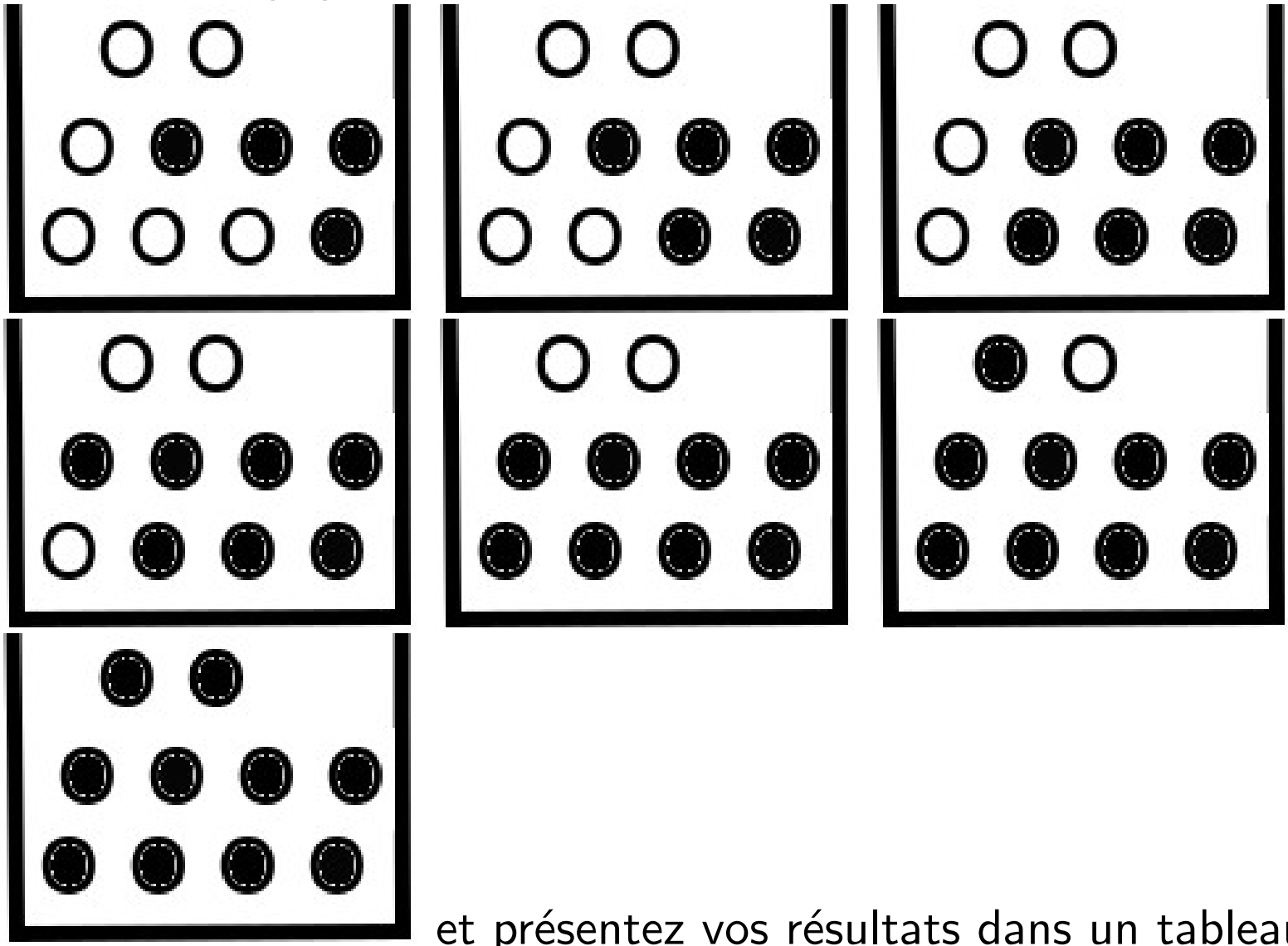
Pour  cet échantillon est impossible.

Pour  la probabilité de produire cet échantillon est

$$\frac{\binom{7}{1}\cdot\binom{3}{3}}{\binom{10}{4}} = \frac{1}{30}.$$

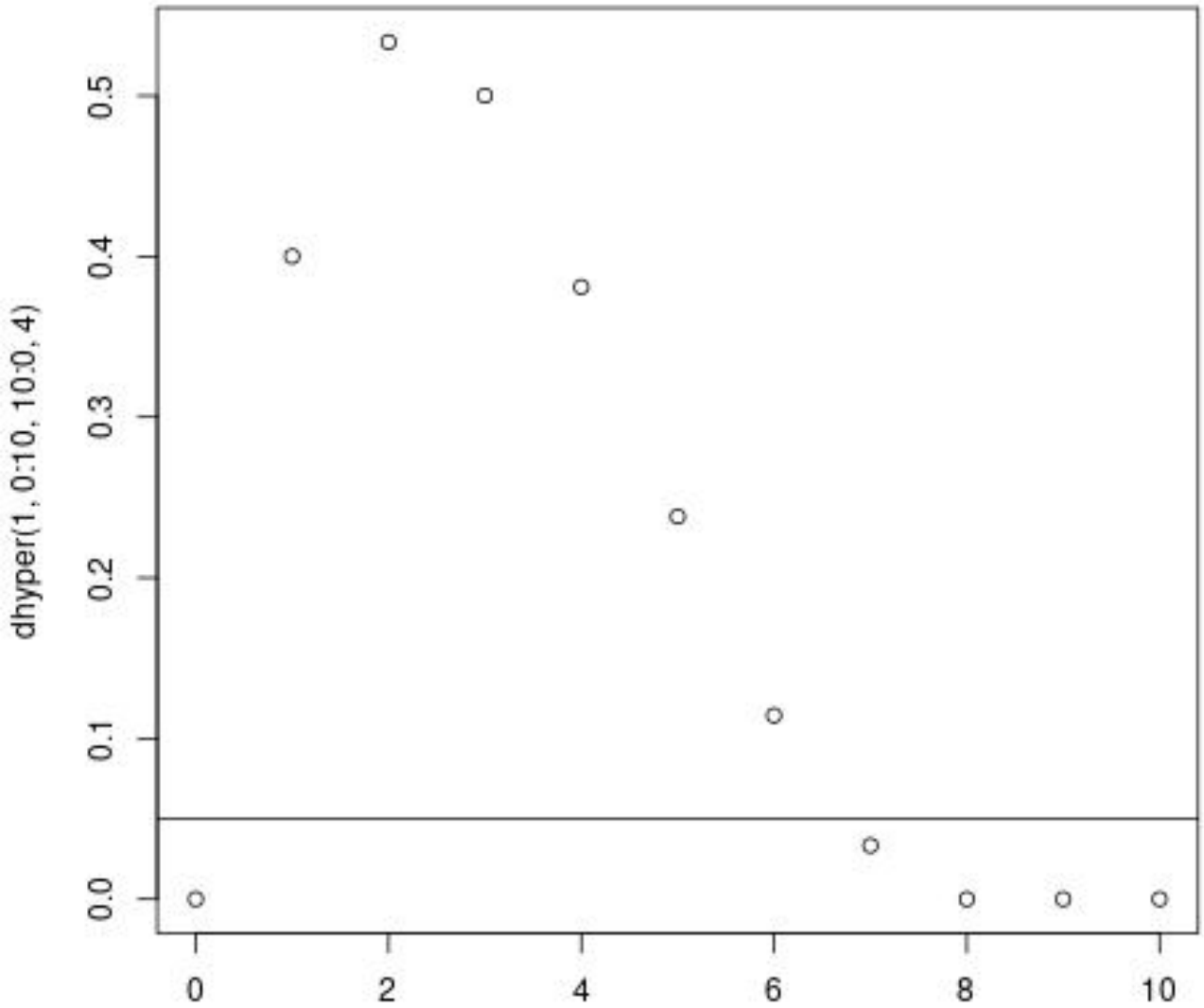
Exercice 14

Calculez la probabilité de recevoir l'échantillon  pour chacune des populations suivantes :



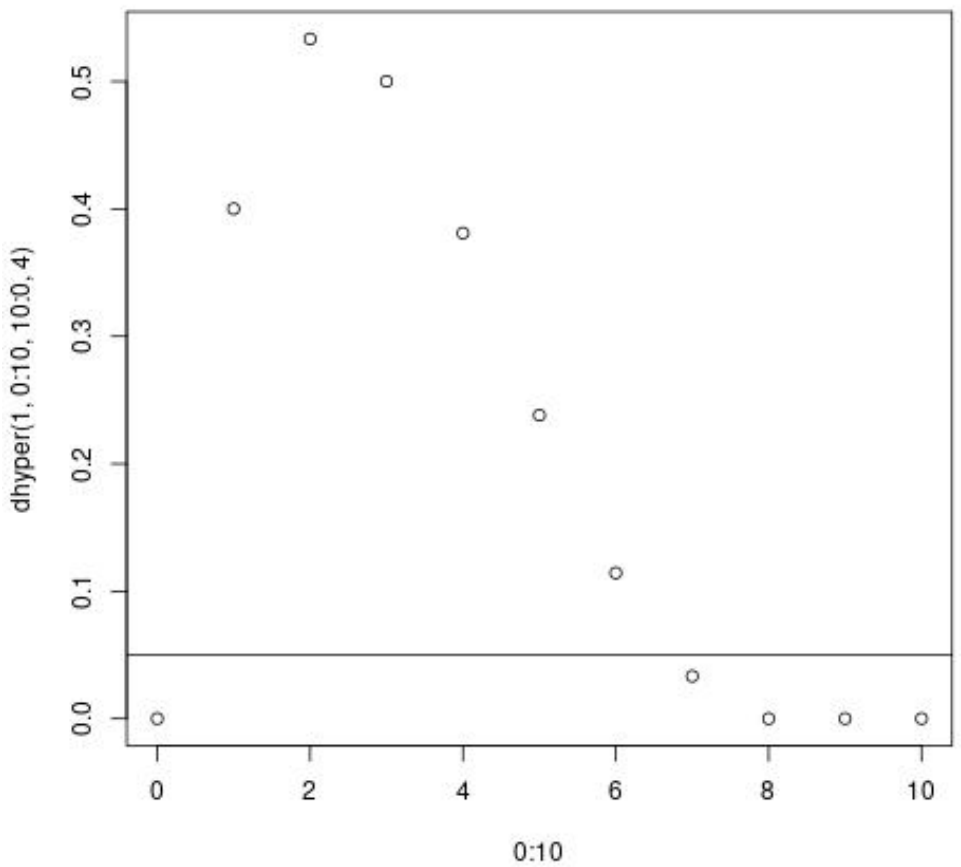
et présentez vos résultats dans un tableau.

Votre tableau se visualise par la commande
`plot(0:10,dhyper(1,0:10,10:0,4))`



Rejet d'hypothèses

Hypothèse H_m : Il y a m perles blanches dans la population.



Nous rejetons toutes les hypothèses qui donnent une probabilité au dessous de 5% : `abline(h=0.05)`

Nous **inférons** que seulement les autres hypothèses (avec 1, 2, 3, 4, 5 ou 6 perles blanches) sont probables.

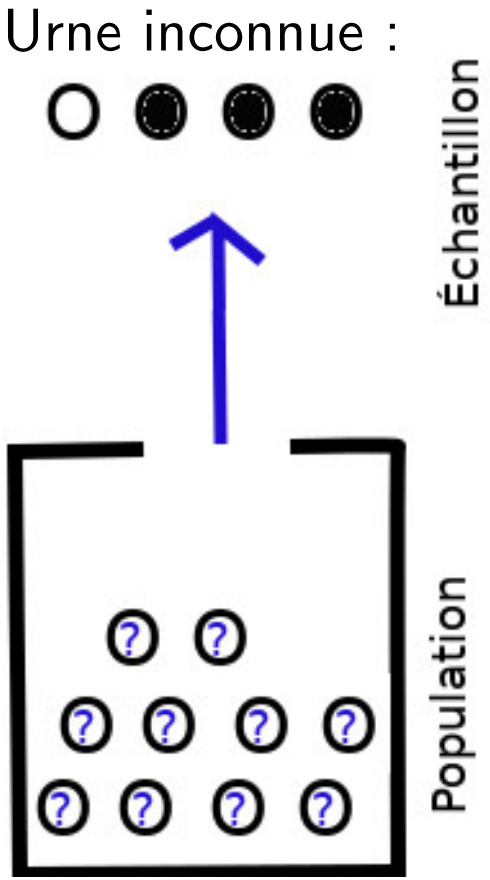
L'intervalle de confiance

Connaissant juste l'échantillon, nous voulons spécifier, avec un risque α (souvent $\alpha = 5\%$) de nous tromper, un intervalle qui contient la fréquence de perles blanches (ou dans d'autres situations, éléments avec une propriété détectée) dans la population. Cet intervalle est appelé *intervalle de confiance* de niveau $1 - \alpha$ ($= 95\%$), et calculé par la formule

$$]f - z_{\alpha/2}\sqrt{\frac{f \cdot (1-f)}{k}},\ f + z_{\alpha/2}\sqrt{\frac{f \cdot (1-f)}{k}}[,$$

où $f = \frac{x}{k}$ est la fréquence de la propriété détectée, donc le nombre x de perles blanches dans l'échantillon divisé par la taille k de ce dernier, et $z_{\alpha/2}$ est la valeur telle qu'une variable de loi normale centrée réduite peut dépasser $z_{\alpha/2}$ avec la probabilité $\alpha/2$. La commande `qnorm`$(1 - \alpha/2)$ calcule $z_{\alpha/2}$, donc pour $\alpha = 5\%$, `qnorm(0.975)` # nous donne $z_{5\%/2} = 1.959964$.

Intervalle de confiance dans notre exemple



Évaluons notre formule

$$]f - z_{\alpha/2}\sqrt{\frac{f\cdot(1-f)}{k}}, f + z_{\alpha/2}\sqrt{\frac{f\cdot(1-f)}{k}}[:$$

```
x = 1; k = 4; f = x/k
f-qnorm(0.975)*sqrt(f*(1-f)/k)
```

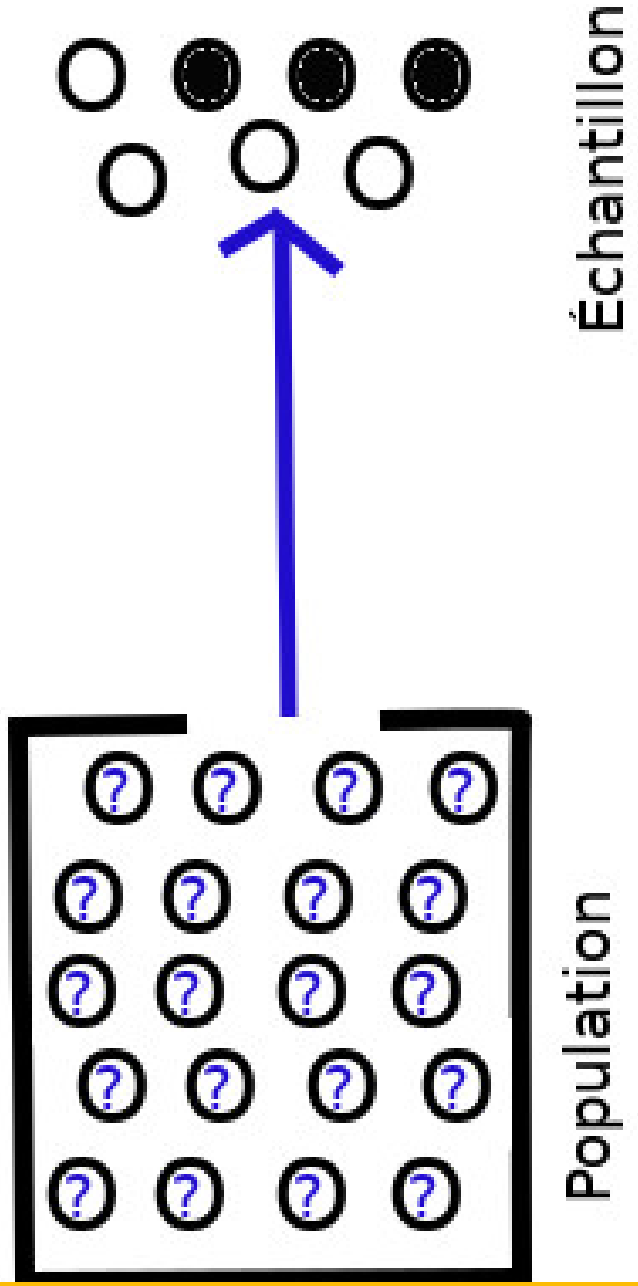
\# *donne* -0.1743447 ;

```
f+qnorm(0.975)*sqrt(f*(1-f)/k)
```

\# *donne* 0.6743447.

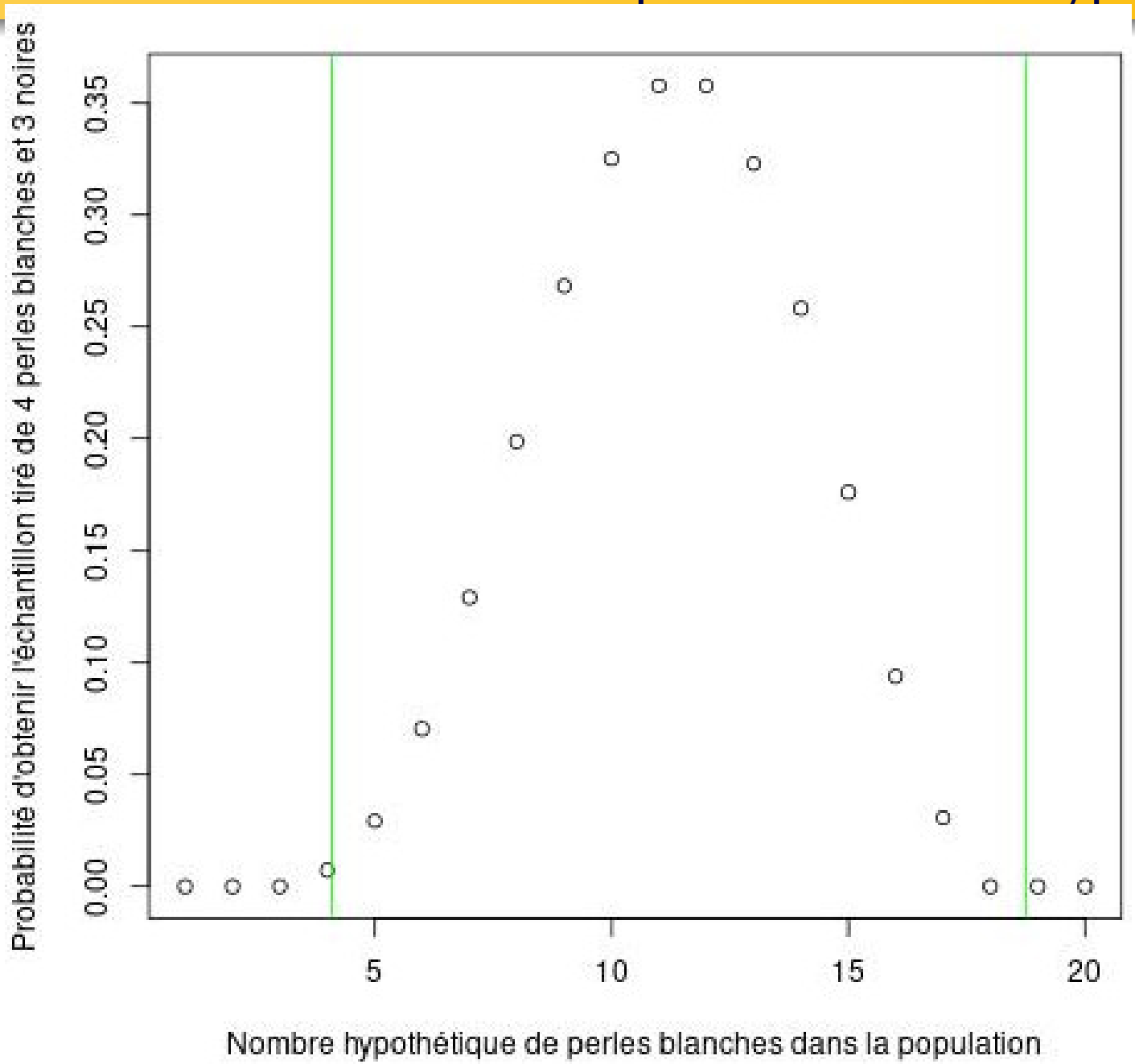
Afin de passer de la fréquence au nombre total de perles blanches, nous devons multiplier par la taille (ici $N = 10$) de la population. Donc notre intervalle de confiance prédit qu'il y a entre -1.743447 et 6.743447 perles blanches dans la population.

Exercice 15



1. Ayant tiré un échantillon de quatre perles blanches et de trois perles noires d'une population de vingt perles, quel est l'intervalle de confiance pour le nombre m de perles blanches dans la population ?
2. Ayant tiré un échantillon de trois perles blanches et onze perles noires d'une population de cent perles, quel est l'intervalle de confiance pour le nombre m de perles blanches dans la population ?

Intervalle de confiance vs. probabilité des hypothèses



Intervalle de confiance vs. probabilité des hypothèses

```
# Code source du diagramme ci-dessus :
collection = NULL
for (m in 0:20) {
      collection = c(collection, dhyper(4,m,20-m,7))
}
plot(collection,ylab="Probabilité d'obtenir l'échantillon tiré de 4
perles blanches et 3 noires", xlab="Nombre hypothétique de perles
blanches dans la population")

N = 20; x = 4; k = 7; f = x/k
abline(v=N*(f-qnorm(0.975)*sqrt(f*(1-f)/k)),col="green")
abline(v=N*(f+qnorm(0.975)*sqrt(f*(1-f)/k)),col="green")
```

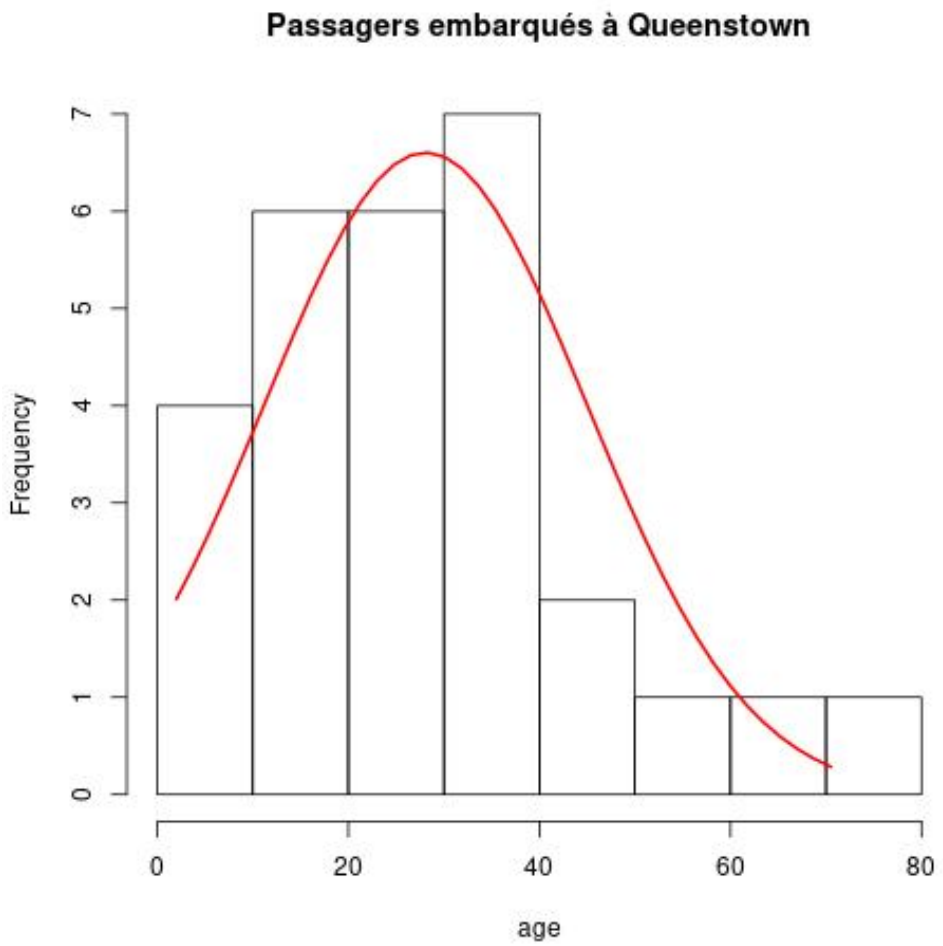
Estimation de l'intervalle de confiance

Le calcul de l'intervalle de confiance décrit ci-dessus est peu pratique quand nous étudions une grande population. Nous pouvons estimer l'intervalle de confiance si nous connaissons la distribution de notre variable.

Comment connaître la distribution de notre variable ?

Pour comparer notre variable à la distribution, nous avons des $\ll$ tests paramétriques $\gg$.

L'age des passagers est-il normalement distribué ?



Qu'allons nous faire pour les variables aléatoires desquelles nous ne savons pas si elles sont normalement distribuées ?

Pour le calcul de l'intervalle de confiance, nous nous sommes servis de ce que les éléments avec une propriété détectée sont normalement distribués dans la population, si nos échantillons sont *probabilistes*, donc issu d'une méthode d'échantillonnage où chaque unité statistique de la population a une chance d'être sélectionnée.

Le test de normalité de Shapiro–Wilk

Soit X une variable numérique que nous avons mesurée sur un échantillon. Le **test de normalité de Shapiro–Wilk** est un **test paramétrique** et essaie de rejeter l'hypothèse nulle

$$H_0 : X \textit{ est distribuée normalement sur la population.}$$

Exemple : Population $= \{$Passagers du Titanic sur notre liste$\}$,
échantillon $= \{$Passagers embarqués à Cherbourg$\}$.

Si la **p-valeur** calculée par ce test est

- $p \leq 0.01$, alors nous présumons très fortement que H_0 est fausse.
- $0.01 < p \leq 0.05$, alors nous présumons fortement que H_0 est fausse.
- $0.05 < p \leq 0.1$, alors nous présumons faiblement que H_0 soit fausse, avec une probabilité entre 5% et 10% de nous tromper.
- $p > 0.1$, alors nous pouvons rien dire sur H_0 étant vraie ou fausse.

Un exemple pour le test de normalité de Shapiro–Wilk

Exemple : Population = {Passagers du Titanic sur notre liste},
échantillon = {Passagers embarqués à Cherbourg}.

H_0 : *Le prix des billets est distribué normalement.*

```
Cherbourg = subset(train, subset=(train$Embarked == "C"))
```

`shapiro.test(Cherbourg$Fare)`
nous retourne

$$\text{p-value} < 2.2\text{e-}16$$

où 2.2e-16 signifie $2.2 \cdot 10^{-16}$.

Alors nous allons présumer très fortement que H_0 est fausse. Donc le contraire doit être le cas :

Le prix de billets n'*est* pas *distribué normalement.*

Mais attention : Notre échantillon n'est pas aléatoire. Le prix moyen sur le jeu de données est 32.20, tandis que le prix moyen est 59.95 pour les passagers de Cherbourg. En effet,

`table(Cherbourg$Pclass)` nous retourne

1	2	3
85	17	66

Un échantillon moins biaisé

Exemple : Population = {Passagers du Titanic sur notre liste},
échantillon = {100 premiers passagers sur notre liste}.
H_0 : *Le prix des billets est distribué normalement.*

```
centPremiers = subset(train, subset=(train$PassengerId < 101))
```

Le prix moyen sur le jeu de données est 32.20, donc `mean(centPremiers$Fare)` donnant 29.52 montre que cet échantillon est moins biaisé que le dernier.
`shapiro.test(centPremiers$Fare)`
nous retourne

p-value = 2.877e-16.

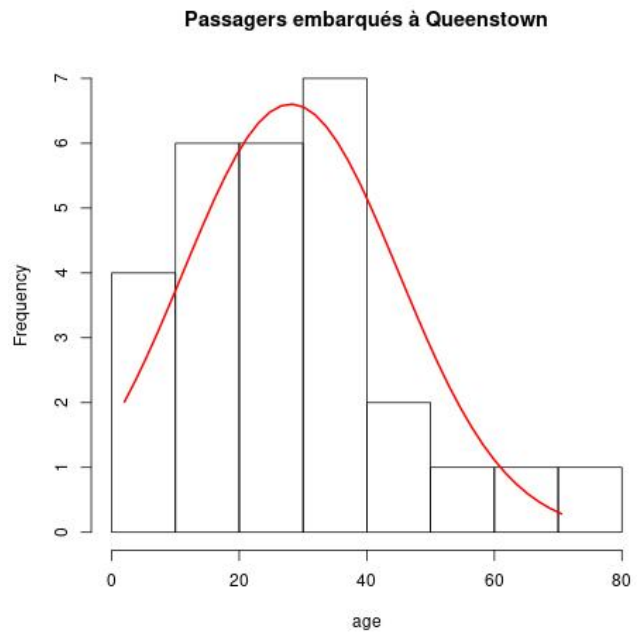
Alors nous pouvons vraiment présumer très fortement que H_0 est fausse. Donc le contraire doit être le cas :
Le prix de billets n'*est* pas *distribué normalement.*

Test de normalité de Shapiro–Wilk : Un autre exemple

Exemple : Population = {Passagers du Titanic sur notre liste},
échantillon = {Passagers embarqués à Queenstown}.

H_0 : *L'age des passagers est distribué normalement.*

```
Queenstown = subset(train,
subset=(train$Embarked == "Q"))
```



Le biais n'est pas trop grand :
`mean(Queenstown$Age[is.na(Queenstown$Age)==FALSE])`
vaut 28.08929, tandis que
`mean(train$Age[is.na(train$Age)==FALSE])`
vaut 29.69912. Le test
`shapiro.test(Queenstown$Age)`
nous retourne $\boxed{\text{p-value} = 0.1146}$.

Alors nous ne pouvons ni dire si H_0 est fausse, ni si H_0 est vraie.
Donc nous ne savons pas si l'age est distribué normalement.

Exercice 16

Testez l'hypothèse H_0 suivante :

Population = {Passagers du Titanic sur notre liste},
échantillon = {Ceux parmis les 100 passagers sur notre liste numérotés par `PassengerId` de 301 jusqu'à 400, pour lesquels l'age est connu (à vérifier avec `is.na(train$Age)==FALSE`)}.

H_0 : *L'age des passagers est distribué normalement.*

Tests non paramétriques

Le test de Shapiro–Wilk est exemple d'un test paramétrique. Un tel test vérifie si notre variable suit une distribution déterminée par des **paramètres** :

La distribution normale est déterminée par les deux paramètres *moyenne*, *variance*.

Quand nous avons rejeté les distributions déterminées par des paramètres pour notre variable, nous passons aux tests **non paramétriques**.

Un test non paramétrique : χ^2

Exemple d'un test non paramétrique :

le test de **khi carré** (χ^2) de Pearson.

Son hypothèse nulle : Deux variables X, Y admettent la propriété suivante, qu'elles doivent avoir si elles sont indépendantes :
Pour le tableau de fréquences
(où N est la taille du jeu de données)

	$Y = y_1$	$\dots$	$Y = y_q$	Total
$X = x_1$	$n_{1,1}/N$	$\dots$	$n_{1,2}/N$	$n_{1,\bullet}/N$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$X = x_p$	$n_{p,1}/N$	$\dots$	$n_{p,q}/N$	$n_{p,\bullet}/N$
Total	$n_{\bullet,1}/N$	$\dots$	$n_{\bullet,q}/N$	1

nous avons pour chaque $1 \leq i \leq p$ et $1 \leq j \leq q$ l'équation

$$\frac{n_{i,j}}{N} = \frac{n_{i,\bullet}}{N} \cdot \frac{n_{\bullet,j}}{N}.$$

Conséquence du test non paramétrique : χ^2

En conséquence, si nous partons de la hypothèse

H_0 : Les deux variables X, Y sont indépendantes,

alors la p-valeur du test de **khi carré** (χ^2) de Pearson appliqué sur X et Y nous permet de conclure :

- $p \leq 0.01$, alors nous présumons très fortement que H_0 est fausse.
- $0.01 < p \leq 0.05$, alors nous présumons fortement que H_0 est fausse.
- $0.05 < p \leq 0.1$, alors nous présumons faiblement que H_0 soit fausse, avec une probabilité entre 5% et 10% de nous tromper.
- $p > 0.1$, alors nous pouvons rien dire sur H_0 étant vraie ou fausse.

Application du test χ^2 aux données du Titanic

Revenons à notre jeu de données de passagers :

```
train <- read.csv("train.csv", header = TRUE)
attach(train);
```

et dressons le tableau de contingence entre classe de confort et survie :

```
XY = table(Pclass, Survived); addmargins(XY)
```

```
> addmargins(table(Pclass, Survived))
      Survived
Pclass   0   1 Sum
   1    80 136 216
   2    97  87 184
   3   372 119 491
   Sum 549 342 891
```

Nous pouvons appliquer le test χ^2 à ces deux variables :

```
( chisquare = chisq.test(XY) )
```

Il nous donne une p-valeur $< 2.2 \cdot 10^{-16}$, ce qui nous confirme que classe de confort et survie n'étaient pas indépendantes.

L'hypothèse nulle de χ^2

Ce qui nous prédit l'hypothèse nulle de χ^2 peut être affiché par les commandes

```
( chisquare = chisq.test(XY) )
chisquare$expected
```

qui produisent le tableau

	Survived	
Pclass	0	1
1	133.0909	82.90909
2	113.3737	70.62626
3	302.5354	188.46465

contrastant

```
> addmargins(table(Pclass, Survived))
      Survived
Pclass   0   1 Sum
   1    80 136 216
   2    97  87 184
   3   372 119 491
   Sum 549 342 891
```

Applicant le test de χ^2 à ce tableau obtenu de l'équation

$$\frac{n_{i,j}}{N} = \frac{n_{i,\bullet}}{N} \cdot \frac{n_{\bullet,j}}{N},$$

```
chisq.test(chisquare$expected)
```

nous donne la p-valeur 1, donc des variables indépendantes doivent nous donner une p-valeur proche de 1.

Exercice 17

Appliquez le test χ^2 pour comparer la survie

```
train$Survived
```

à l'arrondi de la chance de survie pronostiquée par notre modèle statistique,

```
round(survivalChance)
```

.

Faites afficher le tableau de contingence de ces deux variables,

ainsi que le tableau prédit par l'hypothèse nulle du test χ^2 pour ces deux variables.

Exercice 17 bis

Dans le jeu de données ≪ Seatbelts ≫, construisez la variable

$$\text{grandeMortalite} = (\text{DriversKilled} > \text{median}(\text{DriversKilled}))$$

qui enregistre si dans le mois observé,
le nombre de conducteurs de voiture morts excédait la médiane,
et

$$\text{petitPrix} = (\text{PetrolPrice} < \text{median}(\text{PetrolPrice}))$$

qui enregistre si dans le mois observé,
le prix du carburant était inférieur à la médiane.

Testez si on peut présumer (fortement) que
≪ grandeMortalite ≫ et ≪ petitPrix ≫ sont indépendantes,
ou si on peut présumer (fortement) le contraire.