Université Mohammed V
Faculté des Sciences de Rabat
Département de Mathématiques

R

M. Fihri
m.fihri@gmail.com

Introduction

- R est un logiciel de calcul statistique gratuit et peut être installé sous Windows, UNIX ou MacOS.
- Il est constitué d'un noyau de base et de multiples packages développés et mis à disposition de tous par des utilisateurs.
- Dans sa version de base il est utilisable en mode commande, certains packages fournissent une interface graphique.
  RStudio peut aussi être téléchargé gratuitement, il fournit également une interface graphique très ***conviviale***.
- De nombreuses introductions, documentations et tutoriels sont disponibles en français et en anglais sur internet.
- Il existe plusieurs liens pour télécharger R et RStudio avec les mises à jour nécessaires :

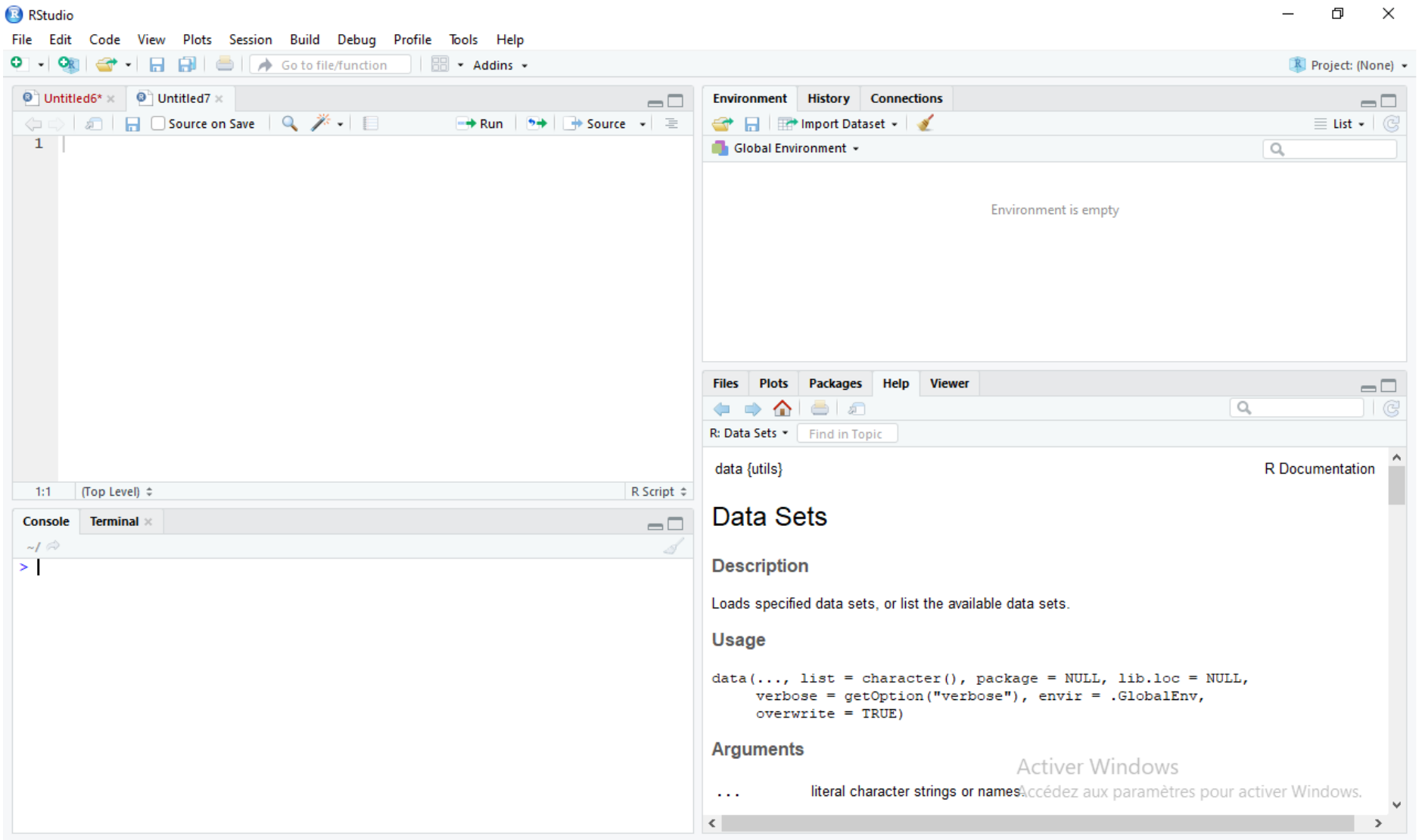
http://larmarange.github.io/analyse-R/installation-de-R-et-RStudio.html
http://www.r-project.org/
http://rstudio.org/

Aide

- `R` possède une aide très sophistiquée, accessible en mode commande ou au format HTML. Pour accéder à l'aide HTML il faut taper
  `help.start()`
  ou cliquer sur **Aide** dans le menu.
- On a alors accès à des manuels, à la documentation des packages, et à un moteur de recherche. L'aide en mode commande se fait par l'intermédiaire de la fonction
  `help(nomfonction)`
  ou bien
  `?nomfonction`.
  Exemple : `?read.table`

Les exemples proposés à la fin de la description des fonctions aident beaucoup en général à comprendre leur fonctionnement.

RStudio


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled6*
Untitled7
Source on Save
Run
Source
1
1:1
(Top Level)
R Script
Console
Terminal
>
Environment
History
Connections
Import Dataset
List
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
R: Data Sets
Find in Topic
data {utils}
R Documentation
Data Sets
Description
Loads specified data sets, or list the available data sets.
Usage
data(..., list = character(), package = NULL, lib.loc = NULL,
verbose = getOption("verbose"), envir = .GlobalEnv,
overwrite = TRUE)
Arguments
...
literal character strings or names
Activer Windows
Accédez aux paramètres pour activer Windows.

Script

- R marche en mode commande mais la plupart du temps pour des programmes un peu longs on utilise un fichier script.

- Pour créer un nouveau script ou en ouvrir un nouveau on peut utiliser l'éditeur par défaut de R sous Windows.

- Sous Linux, il faut utiliser en parallèle à R un éditeur de texte. Il est préférable d'utiliser des interfaces à télécharger telles que RStudio. Ces scripts peuvent être exécutés dans R en utilisant la commande source ("fichier.R")

R Markdown

Pour écrire un compte-rendu en Word ou Libre Office incluant automatiquement les sorties des commandes R et les graphiques, vous pouvez utiliser le package rmarkdown.

Il est en général installé dans les versions récentes de R sinon il faut l'installer en utilisant l'onglet "Packages".

Ensuite dans le menu déroulant "File/New File" il faut choisir : "**R Markdown**". Une fenêtre apparaît dans laquelle il faut remplir un certain nombre de champs : **"Titre"**, **"Auteur"**.
Il faut ensuite choisir le format de sortie **"Word"**. Un nouveau script pré-rempli s'ouvre alors. Dans les zones blanches on peut taper du texte comme dans du Word ou du Libre Office et dans les zones grisées, appelées **"Chunks"**, on tape les commandes R. Pour créer le document .docx il faut cliquer sur **"Knit to Word"**.

Importation des données

La fonction `read.table()` est très générale et permet l'importation de données contenues dans des fichiers avec des structures simples. Sa syntaxe est :

```
read.table(file, header = FALSE, sep = "", quote = "\"'", dec =
".", numerals = c("allow.loss", "warn.loss", "no.loss"),
row.names, col.names, as.is = !stringsAsFactors, na.strings =
"NA", colClasses = NA, nrows = -1, skip = 0, check.names = TRUE,
fill = !blank.lines.skip, strip.white = FALSE, blank.lines.skip =
TRUE, comment.char = "#", allowEscapes = FALSE, flush = FALSE,
stringsAsFactors = default.stringsAsFactors(), fileEncoding = "",
encoding = "unknown", text, skipNul = FALSE)
```

Voir aussi

`read.csv()`
`read.csv2()`
`read.delim()`
`read.delim2()`

Objet

R travaille avec des objets. Les données, les graphiques, les fonctions, les résultats d'analyse sont des objets. Les objets les plus courants sont :

`vector` vecteur composé d'une seule variable
`data.frame` tableau de données
`list` liste d'objets, généralement le résultat d'une analyse. Chaque élément de la liste est accessible en tapant le nom de la liste suivi du symbole `$` et du nom de l'élément.

La plupart des objets ont un attribut de classe. Les plus courants sont `numeric`, `logical`, `integer`, `factor`, `character`, `data.frame`, ...

Pour connaitre la nature d'on objet, on peut utiliser la fonction `class()`

Quelques commandes

- **Opérateurs mathématiques :** $+$, $-$, $\times$, $/$, $\hat{}$
- **Opérateurs logiques :** `&` (et), `|` (ou), `!` (négation), `==` (identité)
- **Fonctions mathématiques :** `sqrt()`, `log()`, `exp()`
- **Fonctions statistiques :** `min()`, `max()`, `sum()`, `mean()`, `var()`, `median()`
- **Opérations vectorielles :** `c()`
- **Concaténation éléments :** colonnes : `cbind()`, lignes : `rbind()`
- **Extraction :** `vec[i]`, `mat[i,j]`, `mat[,j]` où (`i` : ligne, `j` : colonne)
- **Multiplication matricielle :**`% * %`
- **Création de variables :** Au départ `R` utilisait les symboles $<-$ pour l'affectation. Mais les versions récentes admettent également le symbole $=$ pour l'affectation.

Programme

- **Boucles :**

```
for (indices in vecindices){ instructions }
while (condition) { instructions }
```

- **Tests :**

```
if (condition){ instructions } else { instructions }
```

- **Fonction :**

```
sortie = function(arg1=defaut1,arg2=defaut2,...)
{
instructions return(sortie)
}
```

Graphiques

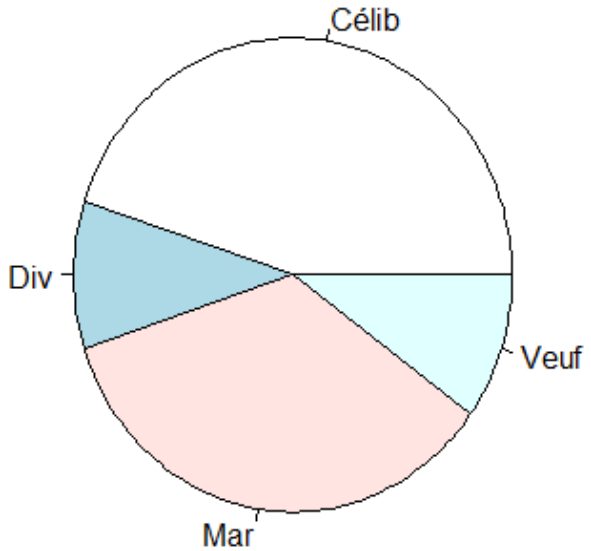
Le graphique de base est obtenu par la fonction `plot(x,y)` qui admet de multiples options.

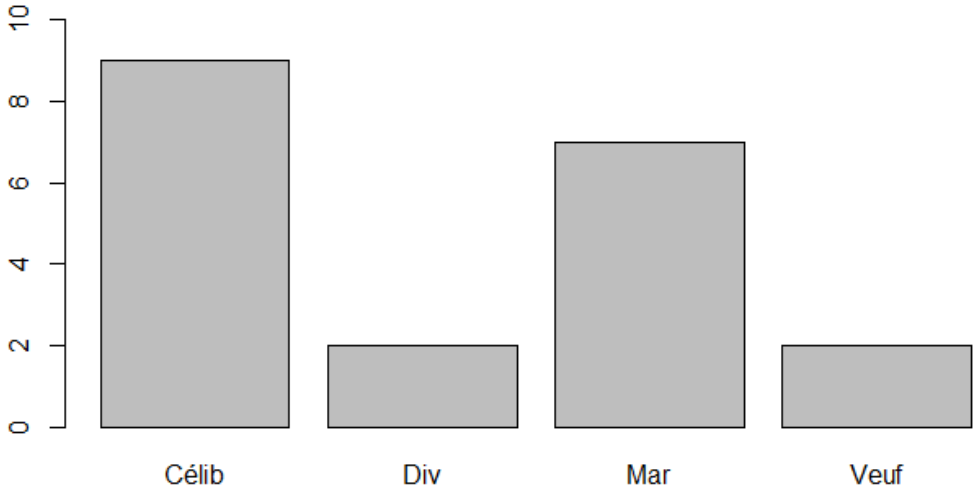
```
require(stats)
# for lowess, rpois, rnorm
plot(cars)
lines(lowess(cars))
plot(sin, -pi, 2*pi) # see ?plot.function
## Discrete Distribution Plot:
plot(table(rpois(100, 5)), type = "h", col = "red", lwd = 10,
main = "rpois(100, lambda = 5)")
## Simple quantiles/ECDF, see ecdf() library(stats) for a better
one:
plot(x <- sort(rnorm(47)), type = "s", main = "plot(x, type =
s̈)")
points(x, cex = .5, col = "dark red")
```

Variable qualitative (1)

```
X=c('Mar','Mar','Div','Célib','Célib','Mar','Célib','Célib',
'Célib','Mar','Célib','Mar','Veuf','Mar','Veuf','Div','Célib',
'Célib','Célib','Mar')
Tab1=table(X)
Var1=c(Tab1)
data.frame(Eff=Var1,Freq=Var1/sum(Var1))
pie(Tab1,radius=1.0)
m=max(Var1)
barplot(Tab1, ylim=c(0,m+1))
```

Variable qualitative (2)

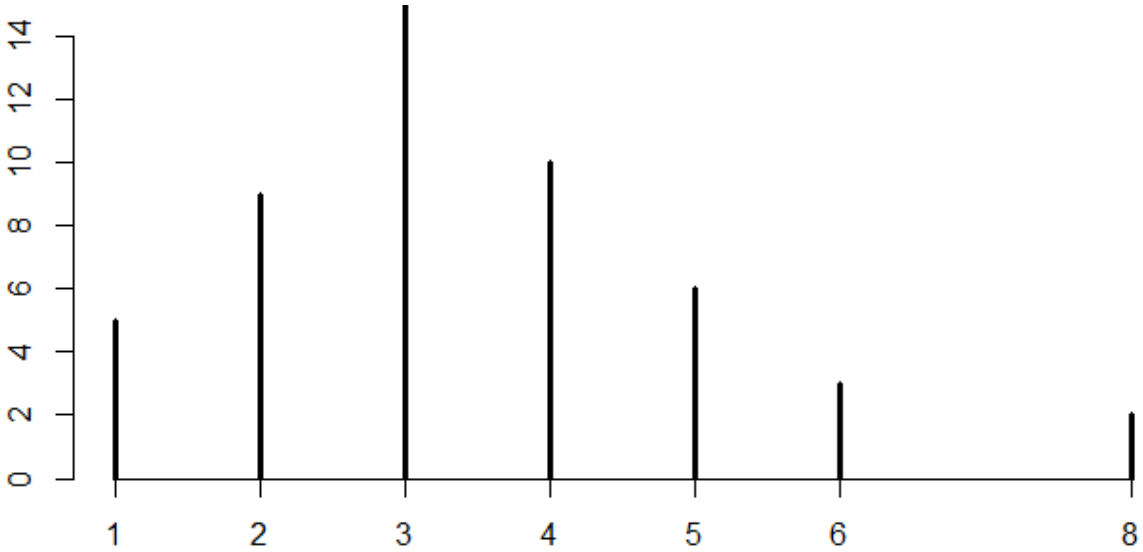


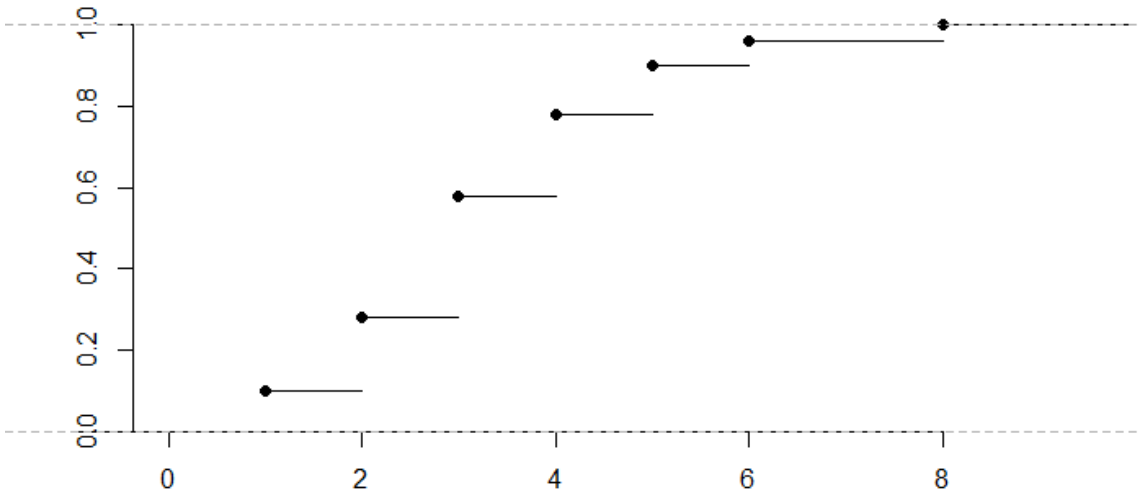


Variable quantitative discrète (1)

```
Y=c(1,1,1,1,1,2,2,2,2,2,2,2,2,2,3,
3,3,3,3,3,3,3,3,3,3,3,3,3,4,4,4,
4,4,4,4,4,4,4,5,5,5,5,5,5,6,6,6,8,8)
Tab2=table(Y)
Tab2c=c(Tab2)
data.frame(Eff=Tab2c,EffCum=cumsum(Tab2c),Freq=Tab2c/sum(Tab2c),
FreqCum=cumsum(Tab2c/sum(Tab2c)))
plot(Tab2,type="h",xlab="",ylab="",main="",frame=0,lwd=3)
plot(ecdf(Y),xlab="",ylab="",main="",frame=0)
```

Variable quantitative discrète (2)

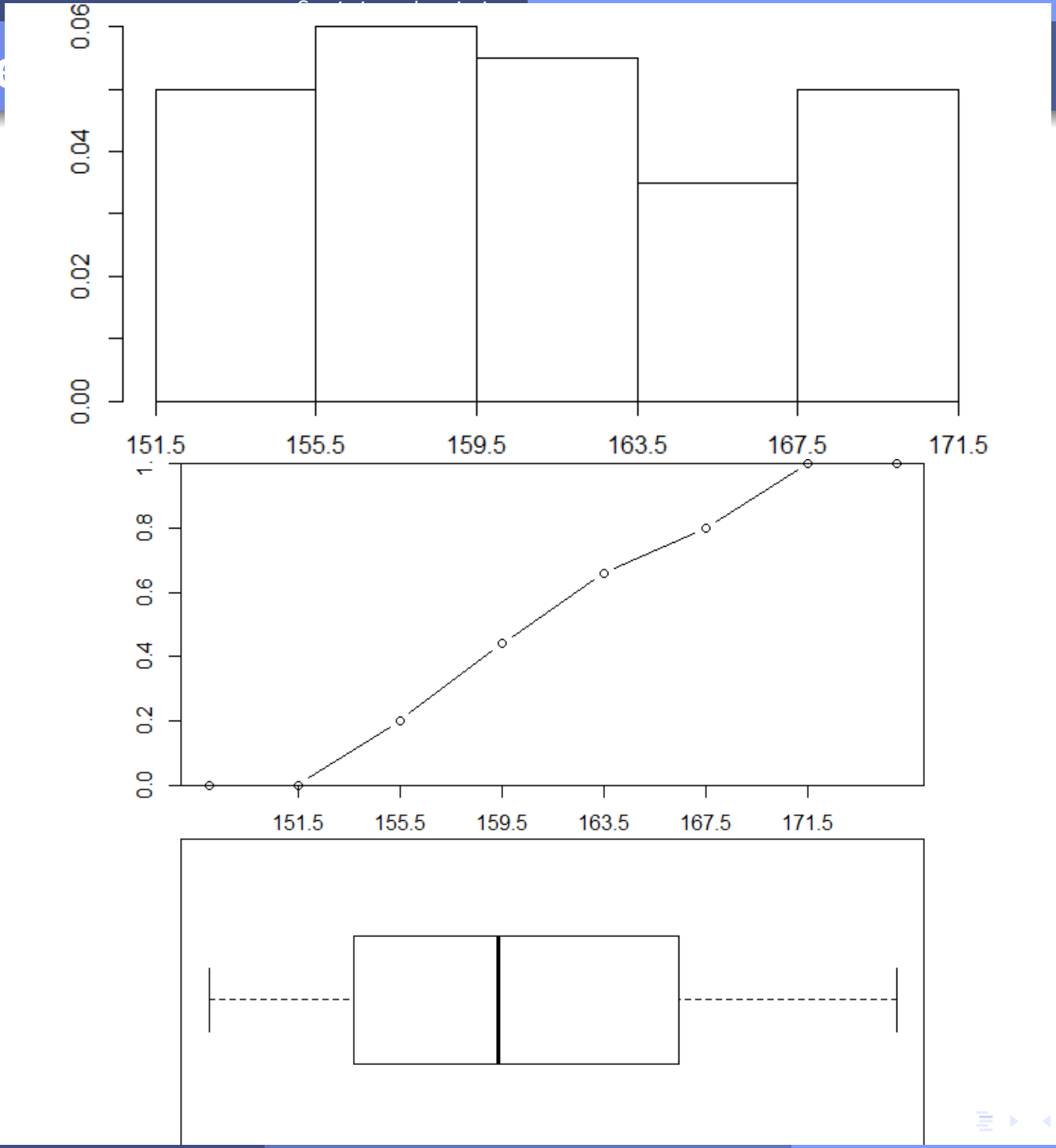




Variable quantitative continue (1)

```
Z=c(152,152,152,153,153,154,154,154,155,155,156,156,156,156,156,
157,157,157,158,158,159,159,160,160,160,161,160,160,161,162,
162,162,163,164,164,164,164,165,166,167,168,168,168,169,169,
170,171,171,171,171)
Tab3=table(cut(Z, breaks=c(151,155,159,163,167,171)))
Tab3c=c(Tab3)
data.frame(Eff=Tab3c,EffCum=cumsum(Tab3c),Freq=Tab3c/sum(Tab3c),
FreqCum=cumsum(Tab3c/sum(Tab3c)))
hist(Z,breaks=c(151.5,155.5,159.5,163.5,167.5,171.5), freq=FALSE,
xlab="",ylab="",main="",xaxt = "n")
axis(1, c(151.5,155.5,159.5,163.5,167.5,171.5))
y=c(0,0,cumsum(Tab3c/sum(Tab3c)),1)
x=c(148,151.5,155.5,159.5,163.5,167.5,171.5,175)
plot(x,y,type="b",xlab="",ylab="",xaxt = "n")
axis(1, c(151.5,155.5,159.5,163.5,167.5,171.5))
boxplot(Z,horizontal=TRUE)
```

Variable


0.00
0.02
0.04
0.06
151.5
155.5
159.5
163.5
167.5
171.5
0.0
0.2
0.4
0.6
0.8
1.
151.5
155.5
159.5
163.5
167.5
171.5

Deux Variables quantitatives (régression linéaire simple) (1)

- Pour ajuster un modèle linéaire simple gaussien $y_i = ax_i + b + \varepsilon_i$
  `modele <-lm(y ~ x)`
- Pour voir les principales statistiques du modèle
  `summary(modele)`
- Pour (aller plus loin) étudier la validité des hypothèses
  - **Adéquation et homoscédasticité :**
    `plot(fitted(modele),stdres(modele))`
  - **Indépendance :**
    `acf(res(modele))`
  - **Normalité :**
    `shapiro.test(resid(modele))` et `QQ-plot`
  - **Points aberrants :**
    `cooks.distance(modele)`

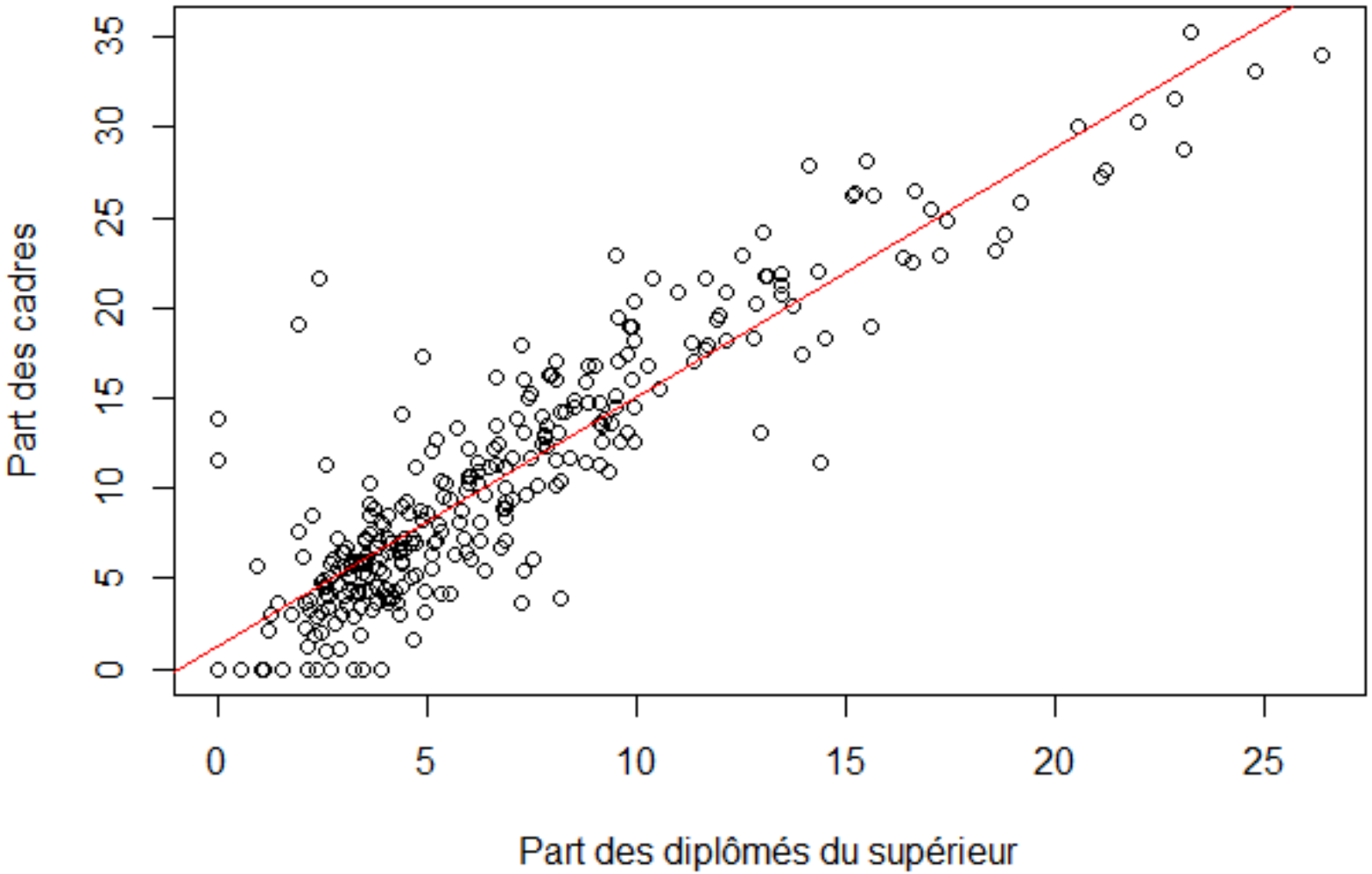
Deux Variables quantitatives (régression linéaire simple) (2)

Considérons le jeu de données `rp99` (fournit par l'extension `questionr`) qui est issu du recensement de la population de 1999 de l'INSEE (France). Il comporte une petite partie des résultats pour l'ensemble des communes du Rhône, soit 301 lignes et 21 colonnes, on s'intéresse aux deux variables suivantes : `dipl.sup` : Part des diplômés du supérieur et `cadres` : Part des cadres [1]

```
library(questionr)
data(rp99)
plot(rp99$dipl.sup, rp99$cadres, ylab = "Part des cadres", xlab =
"Part des diplomes du supérieur")
cor(rp99$dipl.sup, rp99$cadres)
reg <- lm(cadres ~ dipl.sup, data = rp99)
summary(reg)
plot(rp99$dipl.sup, rp99$cadres, ylab = "Part des cadres", xlab =
"Part des diplômés du supérieur")
abline(reg, col = "red")
```

1. https://r.developpez.com/tutoriels/r/introduction/?page=annexe-b-extensions

Deux Variables quantitatives (régression linéaire simple) (3)



Une variable quantitative et une variable qualitative (1)

Dans cette partie, on utilise des données dans "hdv2003" fournis dans R par l'extension questionr ; l'extrait est tiré du fichier détail mis à disposition librement (ainsi que de nombreux autres) par l'INSEE.

```
library(questionr)
data(hdv2003)
d <- hdv2003
d.hard <- subset(d, hard.rock == "Oui")
d.non.hard <- subset(d, hard.rock == "Non")
boxplot(d.hard$age, d.non.hard$age)

library(yarrr)
pirateplot(age ~ hard.rock, data = d, theme = 1, inf.method =
"ci", bar.f.o = 0.1, bar.f.col = "grey10")
```

Une variable quantitative et une variable qualitative (2)