Concepts de base

Jean-Michel Marin

Université de Montpellier

HAX815X Programmation R

Introduction
Le logiciel R

R est un logiciel de Statistique distribué librement par le CRAN, créé dans les années 90 par R. Ihaka et R. Gentleman dans l'esprit de S et de Lisp :

`http://cran.r-project.org/`

Il est dédié à l'analyse statistique et à la visualisation de données. Sachant qu'environ 80% du temps de l'analyse est dédié à la préparation des données, il sert aussi à manipuler des données

Introduction
Structure

R est disponible sous de nombreux systèmes d'exploitatio

R est composé d'un socle (≪ base ≫) et de bibliothèques de fonctions thématiques regroupées sous le nom de **package**

Il est possible de connecter R avec d'autres langages : C, Fortran, Java, Javascript, Python...

Il est possible d'appeler des fonctions R depuis Matlab, Excel, SAS, SPSS...

Des connectiques pour tous les types de bdd : RODBC, RMySQL, ROracle, RJDBC, RMongo...

Introduction
Les packages

R a été pensé comme un langage ouvert et modulaire. De nombreux chercheurs utilisent R donc les nouvelles méthodes sont souvent implémentées ; le passage recherche/industrie est de plus en plus rapide

Il est fort probable qu'une autre personne que vous ait déjà rencontré le même problème que le votre : possibilité d'utilisation de packages existants, de discussions R-bloggers....

Introduction
Installation

Pour installer R :

`https://ftp.igh.cnrs.fr/pub/CRAN/`

Choisissez votre plateforme....

Introduction
Différences

R est différent des autres logiciels donc n'essayez pas de rechercher des analogies : R a sa propre façon de travailler

Par exemple, vous n'avez pas besoin de classer (sort) les données pour les résumer, aggréger, splitter, merger... : des fonctions existent pour cela

Peu d'interfaces graphiques (GUI) natives

Premiers pas
Ouverture d'une session

```
R version 4.1.0 (2021-05-18) -- "Camp Pontanezen"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin17.0 (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

>
```

Premiers pas
Ouverture d'une session

R attend une instruction : ceci est indiqué par $>$ en début de ligne.
Cette instruction doit être validée par **Entrée** pour être **éxecutée**.

- instruction correcte, R éxecute et redonne la main $>$
- instruction incomplète R retourne $+$, il faut alors compléter l'instruction ou sortir avec **Echap** ou Ctrl+C en mode console.

Premiers pas
Répertoire courant et chemins

Endroit où R stocke ses objets et où R écrira/accèdera à des scripts et des fichiers

Windows	**Menu** : Fichier/Changer le répertoire courant
Linux	On lance R dans un répertoire (via une fenêtre de commande)
Mac	Icone divers, option Changer le répertoire courant
Rstudio	session/set working directory

Premiers pas
Objets et session

Fonctions utiles : **getwd(), setwd()**

```
setwd("C:\\Users") #notation typée windows
setwd("C:/Users")  #notation typée linux
getwd()
## [1] "C:/Users"
```

Pour obtenir de l'aide :

```
help(mean)
help.start()
```

Premiers pas
Packages

disponibles sur le CRAN : https ://cran.r-project.org/

- installation via l'onglet **Packages** dans RStudio
- sinon via la fonction **install.packages**

```
install.packages("ibr")
```

Une fois installé, il faut charger le package

```
library(ibr)
```

Premiers pas
Objets et session

- Pour sauvegarder un (ou plusieurs objets)

  ```
  > save(x,X,f,file="mesdonnees.RData")
  ```

- Pour sauvegarder tous les objets (en partant)

  ```
  > q()
  ```

  et la question est

  ```
  Save workspace image? [y/n/c]: y
  ```

- Pour sauvegarder tous les objets sans quitter

  ```
  > save.image()
  ```

Premiers pas
Editeur de texte

- Plus confortable (undo-redo, clavier ou souris etc.)
- Plus rapide : dès que l'on recommence !
- Plus clair : commentaire dans le texte

Rstudio s'impose comme le leader actuel

Premiers pas
Installation RStudio

RStudio est une application permettant de travailler en R dans un environnement de développement riche et complet

`https://www.rstudio.com/products/rstudio/download/`

Rstudio est divisé en 4 quadrants :

- Editeur de texte, de codes....
- Espace de travail, historique, importation...
- Visualisation, aide
- Console

Premiers pas
Auto formation

- énormément du supports sur le web
- MOOC (cf FUN)
- livre comme par exemple ≪ Statistiques avec R ≫ aux PUR
- package **swirl**
- ...

La session
Calculs

```
> 1+1
[1] 2
> pi
[1] 3.141593
> sin(pi/2)
[1] 1
```

La session
Commandes

- composées de fonctions et/ou opérateurs
- opèrent sur des objets

```
> help(mean)
> exp(2)+3
> q()
```

Les objets I

- Principaux types des données
  - booléen (logical) : `TRUE`, `FALSE`
  - numeric : integer `1L` ou double : `3.14`
  - caractères (character) : `'bonjour'`, `"hello"`
  - vide (null) : `NULL`
  - complexe (complex) : `2+0i`, `2i`
  - binaires (raw)
- Structuration des données
  - monotype, tous les éléments sont de même type (vecteur, matrice, tableaux)
  - de types différents : liste, data.frame
    Le tableau individus/variables est **LA** structure en Statistique : chaque colonne représente une variable. Tous les éléments d'une colonne sont donc de même mode mais les colonnes peuvent être de modes différents (variables qualitatives, quantitatives)

Les objets II

- Création : par affectation avec $=$ ou $->$ ou $<-$

```
> objects() # liste les objets en mémoire
character(0)
> x <- 2
> X=4
> 4 -> X
> objects() # les objets en mémoire
[1] "x" "X"
> print(X) # affichage
[1] 4
> x;X # on peut enchainer les instructions avec un ;
[1] 2
[1] 4
```

Les objets III

- Fonctions utiles

```
> rm(x)  # supprime l'objet
> is.vector(X)
[1] TRUE
> class(X)
[1] "numeric"
> length(X)
[1] 1
```

Les objets IV

- Attributs (caractéristiques additionnelles)

```
> attributes(X)
NULL
```

- Les grands types d'objets
  - Les vecteurs

```
> x <- c(1:3,4,5)
> x
[1] 1 2 3 4 5
> y <- c("M","F","F","M","F")
> y
[1] "M" "F" "F" "M" "F"
```

Les objets V

- Les facteurs (vecteur de caractères + attributs)
  Variables qualitatives

```
> y <- factor(y)
> factor(y)
[1] M F F M F
Levels: F M
```

- Les matrices : création avec **matrix()**

```
> X <- matrix(c("R","T","G","Y"),ncol=2,nrow=2)
> X
     [,1] [,2]
[1,] "R"  "G"
[2,] "T"  "Y"
```

Les objets VI

- Les listes : objets composites

```
> maliste=list(comp1=x,comp2=X)
> maliste
$comp1
[1] 1 2 3 4 5

$comp2
     [,1] [,2]
[1,] "R"  "G"
[2,] "T"  "Y"
```

Les objets VII

- Liste spéciale : data-frame
  Tableau de données : variables quantitatives + qualitatives

```
> data.frame(var1=x,var2=y)
  var1 var2
1    1    M
2    2    F
3    3    F
4    4    M
5    5    F
```

Typique des tableaux de données en statistiques. Après importation de données par **read.table()** (caractères $\rightarrow$ facteurs)

La création d'objets
Vecteurs I

- Concaténer/collecter

```
> x=c(TRUE,FALSE,TRUE)
> x
[1]  TRUE FALSE  TRUE

> c(x,FALSE)
[1]  TRUE FALSE  TRUE FALSE
```

La création d'objets
Vecteurs II

- Séquence

```
> seq(1,10,by=2)
[1] 1 3 5 7 9

> seq(1,10,length=4)
[1]  1  4  7 10

> 1:10
[1]  1  2  3  4  5  6  7  8  9 10
```

La création d'objets
Vecteurs III

- Répétition

```
> rep(c("hello","bye"),2)
[1] "hello" "bye"   "hello" "bye"

> rep(c("hello","bye"),times=2)
[1] "hello" "bye"   "hello" "bye"

> rep(c("hello","bye"),each=2)
[1] "hello" "hello" "bye"   "bye"
```

La création d'objets
Matrices I

- Créer matrices

```
> x <- matrix(1:6,nrow=2,ncol=3,byrow=TRUE)
> x
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6

> y <- matrix(1:2,ncol=1)
> y
     [,1]
[1,]    1
[2,]    2
```

La création d'objets
Matrices II

- Concaténer matrices

```
> rbind(x,z)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    3    2    1

> cbind(x,y)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    1
[2,]    4    5    6    2
```

La création d'objets
Liste I

- A partir d'une liste vide

```
> li <- list()
> li
list()
> li[[1]] <- 1:4
> li
[[1]]
[1] 1 2 3 4
```

La création d'objets
Liste II

- Ajout d'un élément

```
> li$nouv <- matrix(1:4,nrow=2)
> li
[[1]]
[1] 1 2 3 4

$nouv
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

La création d'objets
Facteurs I

- Conversion de vecteurs de caractères/numériques

```
> f <- factor(c("F","M","F","F"))
> f
[1] F M F F
Levels: F M
> levels(f)
[1] "F" "M"
> nlevels(f)
[1] 2
> table(f)
f
F M
3 1
```

La création d'objets
Facteurs II

- Découpage en classes

```
> x <- 1:10
> f <- cut(x,breaks=c(1,2,4,10),include.lowest=TRUE)
> f
 [1] [1,2]  [1,2]  (2,4]  (2,4]  (4,10] (4,10] (4,10]
 [8] (4,10] (4,10] (4,10]
Levels: [1,2] (2,4] (4,10]
```

L'importation

Il est possible d'importer tous les formats de fichiers car il existera un package pour vous aider

Il est possible d'importer directement des fichiers avec Rstudio et ce dernier vous propose directement dans **import Dataset** différents formats (Excel, SPSS, SAS, Stat) mais aussi à partir de Text avec 2 possibilités base ou readr

Comme son nom l'indique, base propose d'utiliser la version de base de R que nous présentons dans un petit exemple. L'objet obtenu est un data.frame et toutes les variables character sont considérées comme des facteurs

L'importation

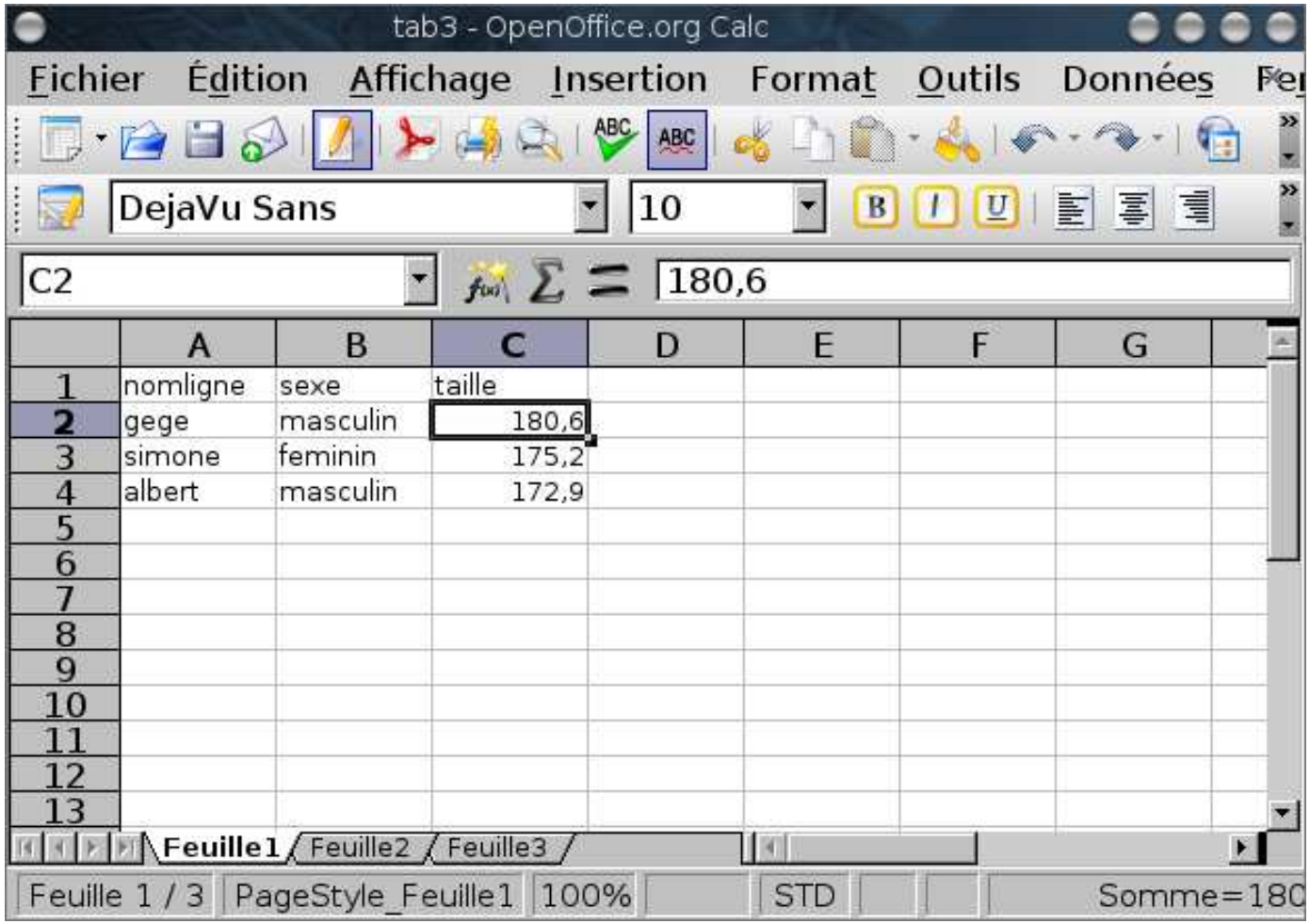


FIGURE – Contenu de `tab3.xls` ouvert grâce à un tableur

L'importation

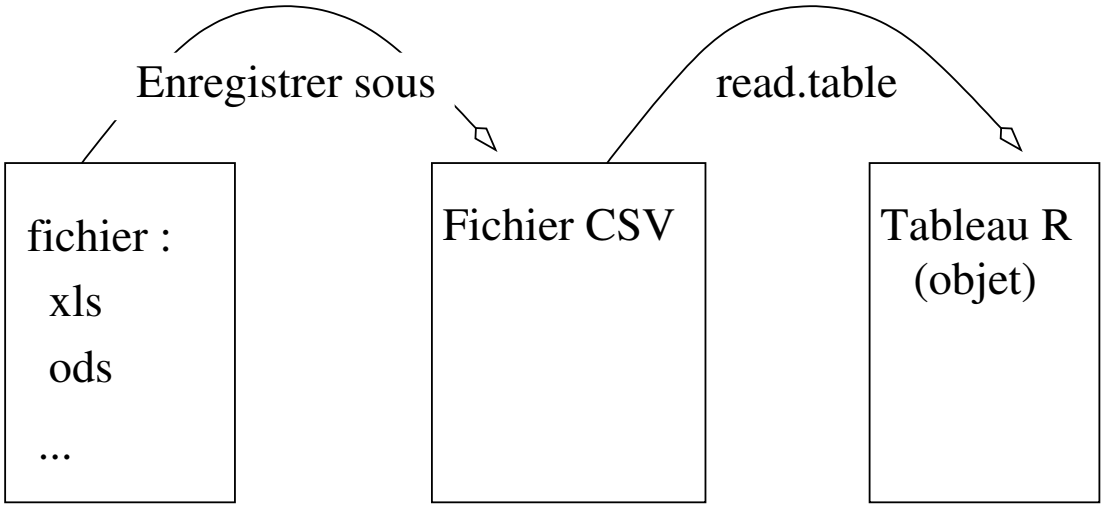


FIGURE – Importation de données depuis un tableur

L'importation

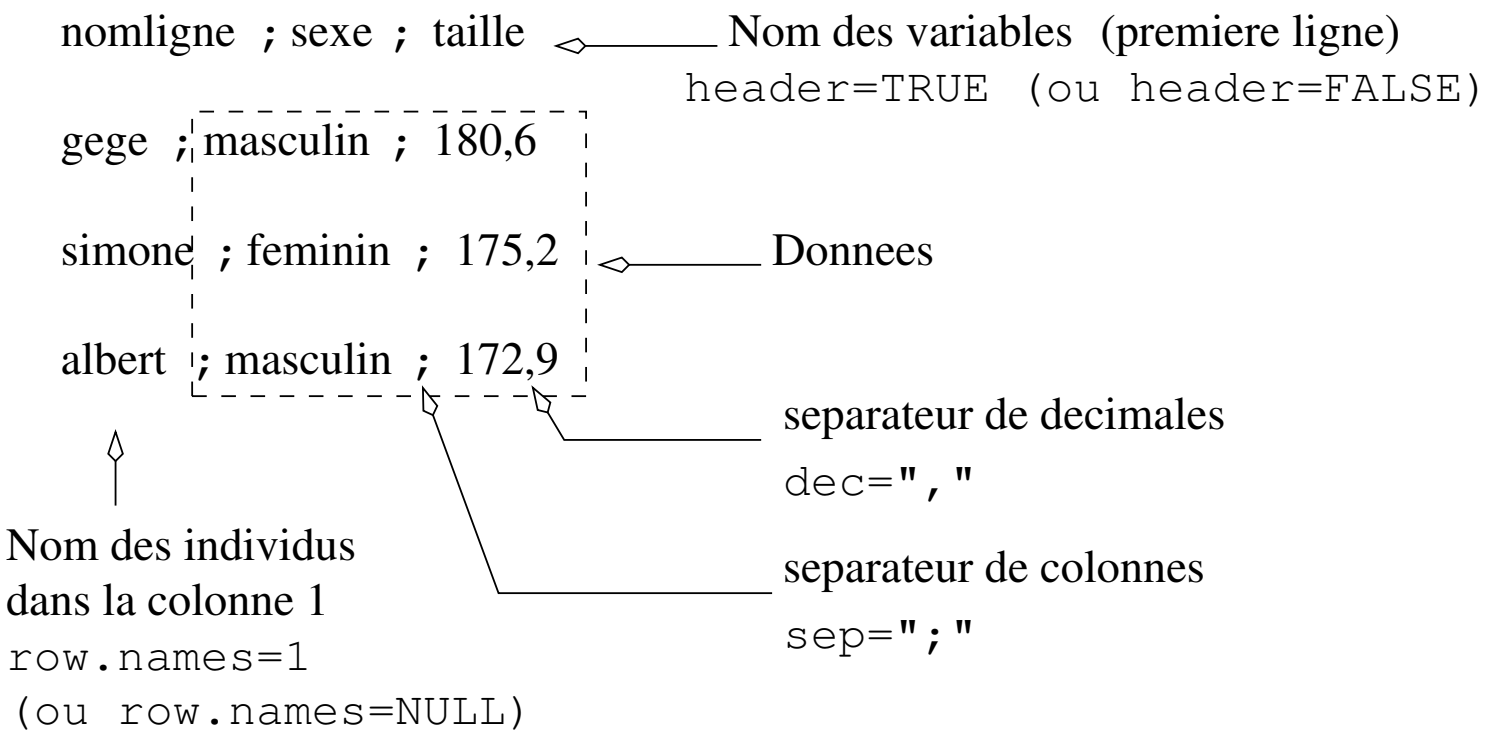


FIGURE – Contenu de `tab3.csv` ouvert grâce à un éditeur de texte

L'importation

```
> tab3 <- read.table("C:/monrep/tab3.csv",row.names=1,
    header=TRUE,sep=";",dec=",")
```

```
> write.table(tab3,"C:/temp/tab3.txt",quote=TRUE,sep=" "
    row.names=FALSE,col.names=TRUE)
```

L'importation

```
> x <- 1:10
> summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    3.25    5.50    5.50    7.75   10.00

> df <- data.frame(nombre = 1:3, lettres = LETTERS[1:3])
> summary(df)
     nombre    lettres
 Min.   :1.0   A:1
 1st Qu.:1.5   B:1
 Median :2.0   C:1
 Mean   :2.0
 3rd Qu.:2.5
 Max.   :3.0
```

Importation avec Rstudio

Rstudio propose un outil d'aide à l'importation dans le quadrant environnement avec l'onglet **Import Dataset**
Dans la dernière version de Rstudio, il est possible d'importer

- from Text (base) avec les fcts classiques, les caractères seront considérés comme des facteurs
- from Text (readr), les caractères sont conservés en caractères
- from Excel ...

La sélection dans les objets

- soit par la position : dans ce cas il faut indiquer un vecteur de positions (il peut être de longueur différente de l'objet) ;
- soit par des booléens : dans ce cas, le vecteur de booléens doit être de la longueur de l'objet à sélectionner et on ne conserve que les valeurs TRUE

La sélection dans les objets
Opérateurs de comparaison

négation (not)	`!`
et logique (and)	`&`
ou logique (or)	`|`
supérieur (strictement)	`>`
supérieur	`>=`
inférieur (strictement)	`<`
inférieur	`<=`
différent	`!=`
égal	`==`
dans un ensemble	`%in%` ou `match`

La sélection dans les objets
Sélection dans les vecteurs

- Par le numéro des coordonnées

```
> x <- c(2,-1,15)
> x[c(3,1,2,2,1)]
[1] 15  2 -1 -1  2
```

- Par des logiques

```
> x>0
[1]  TRUE FALSE  TRUE
> x[x>0]
[1]  2 15
```

- Par suppression de coordonnées

```
> x[-3]
[1]  2 -1
```

La sélection dans les objets
Sélection dans les matrices I

- Par le numéro des coordonnées, indiceligne et indicecolonne

```
> X[indiceligne,indicecolonne]
> X[indiceligne,]
> X[,indicecolonne]
```

- Par des logiques
- Par suppression des coordonnées

Attention, si on sélectionne une seule colonne, R renvoie un vecteur (possibilité de contrôler avec l'argument **drop = FALSE**)

La sélection dans les objets
Sélection dans les listes I

```
> maliste=list(comp1=x,comp2=X)
> maliste
$comp1
[1] 1 2 3 4 5

$comp2
     [,1] [,2]
[1,] "R"  "G"
[2,] "T"  "Y"
```

La sélection dans les objets
Sélection dans les listes II

- Par le numéro des coordonnées

```
 maliste[[2]]
     [,1] [,2]
[1,] "R"  "G"
[2,] "T"  "Y"
```

- Par des noms

```
> maliste$comp1
[1] 1 2 3 4 5
> maliste[["comp1"]]
[1] 1 2 3 4 5
```

- Par des logiques
- Par suppression des coordonnées

La sélection dans les objets
Sélection dans les data-frames I

```
> df <- data.frame(var1=x,var2=y)
> df
  var1 var2
1    1    M
2    2    F
3    3    F
4    4    M
5    5    F
```

- Par le numéro des coordonnées (cf. matrices)

La sélection dans les objets
Sélection dans les data-frames II

- Par des noms (cf. listes)

```
> df[,c("var2","var1")] # pour les matrices aussi
  var2 var1
1    M    1
2    F    2
3    F    3
4    M    4
5    F    5
> df$var1
[1] 1 2 3 4 5
```

- Par des logiques (cf. matrices)
- Par suppression des coordonnées (cf. matrices)

La sélection dans les objets
Ordonner les données

- **sort()** permet de trier un vecteur
- **order()** retourne les indices des données triées

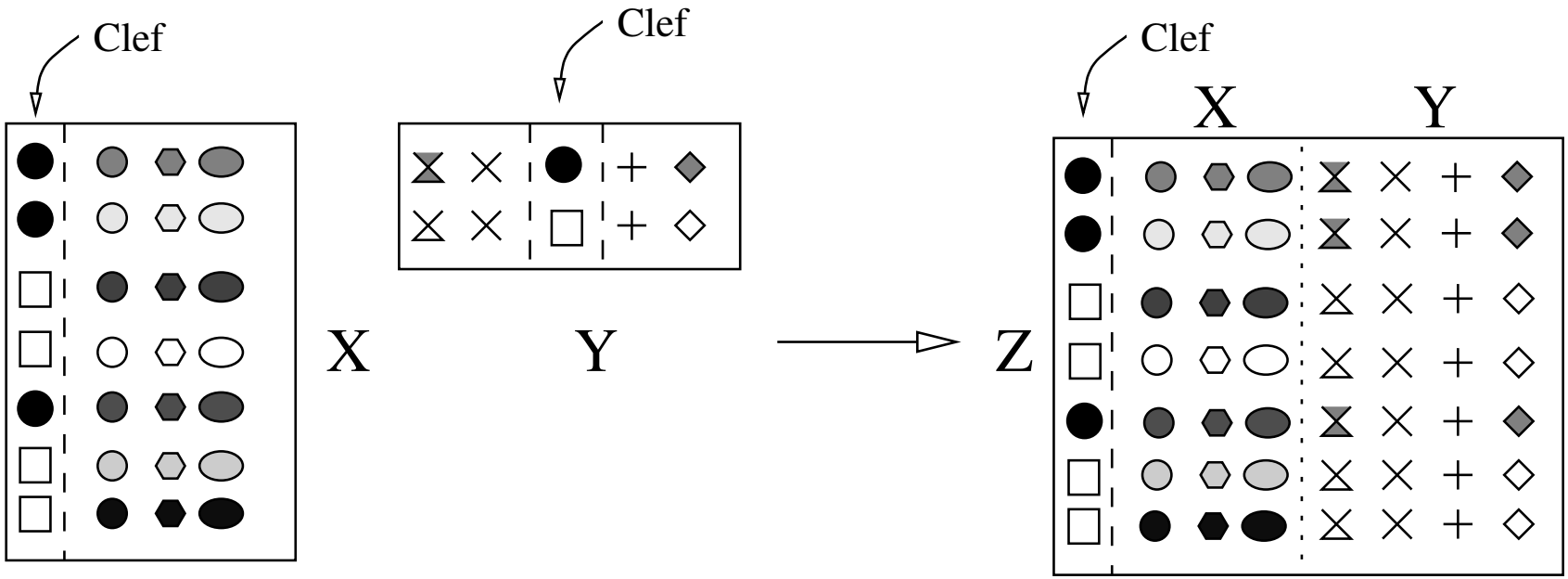
```
> x <- round(rnorm(8), digits =2)
> sort(x)
[1] -0.57 -0.24 -0.15  0.02  0.16  0.32  0.62  1.10
> sort(x,decreasing = TRUE)
[1]  1.10  0.62  0.32  0.16  0.02 -0.15 -0.24 -0.57
> order(x)
[1] 3 7 6 8 4 1 2 5
```

La sélection dans les objets
Manipulation

- **unique()** renvoie les valeurs uniques d'un objet Elle peut s'appliquer aux df et aux matrices
- **duplicated()** renvoie les valeurs en double
- **subset** choix d'un sous-ensemble via la condition
- **split** découpe le jeu de données en fonction d'un critère
- **aggregate** permet de faire des calculs par groupes

La sélection dans les objets
Fusionner

Il est possible de fusionner deux tableaux selon une clef (cf. fusion de 2 tables dans les bases de données), grâce à l'ordre classique **merge**



La sélection dans les objets
Importer/exporter

fichiers .csv / .txt avec **read.table(), write.table(), read.csv(), write.csv()** ; les principaux arguments sont

- header/col.names
- row.names
- sep
- dec
- skip et nrow

fichiers .xlsx / .xls avec le package readxl et **read_excel** ou avec le package XLConnect.

La sélection dans les objets
Sauvegarder/Restaurer

save() permet d'enregistrer sur le disque un ou plusieurs objets R. Le stockage est compressé et l'extension utilisée est **.RData**. **load()** permet de restaurer les objets qui conserveront leur nom

On peut aussi utiliser **saveRDS()** et affecter l'objet à la variable de son choix avec **readRDS()**

La sélection dans les objets
Pour aller plus loin : connexion à des bdd

Il existe toujours au moins un package qui permet de se connecter à chaque type de bdd

- **RMySQL** bdd MySQL
- **ROracle**
- **RPostgreSQL**
- **RSQLServer** bdd MS SQM server
- **mongolite** bdd NoSQL mongodb
- **RSQLite**
- **RJDBC** connexion à différents type de BDD via java
- **RODBC** via ODBC

La sélection dans les objets
Pour aller plus loin : connexion à des bdd

Syntaxe usuelle de connexion

- ouverture de la connexion avec**dbConnect()**
- requêtage avec **dbGetQuery()**
- fermeture de la connexion avec **dbDisconnect()**

```
> conn <- dbConnect("RMySQL", host = "myserver",
                    port = 123, dbnam = "database",
                    user = "eric", password = "aqw")
> res <- dbGetQuery (conn, "SELECT * FROM matable")
> dbDisconnect(conn)
```