

# Importing Data Into R

L. Torgo

ltorgo@dcc.fc.up.pt

Departamento de Ciência de Computadores  
Faculdade de Ciências / Universidade do Porto

Oct, 2014



Introduction

## Data Sets

- The majority of data analysis tasks uses as source data sets stored in a tabular format
- A data set is a bi-dimensional structure (a table)
  - Rows represent observations of a phenomenon
  - Columns contain information obtained for each observation
- The R data structure used to store these tables is the *data frame*
- In the following slides we will see illustrations on how to import data stored in different formats/infra-structures into a R data frame

## Internal R Data Sets

- Any R installation includes many data sets. The list can be obtained doing:

```
data()
```

- Each new package we install may also add new data sets for illustration purposes

```
data(package = "DMwR") # data sets from a specific package
data(package = .packages(all.available = TRUE)) # all packages
```

- This type of data sets can be used/loaded using the function `data`

```
data(iris)
head(iris)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1         5.1         3.5         1.4         0.2   setosa
## 2         4.9         3.0         1.4         0.2   setosa
## 3         4.7         3.2         1.3         0.2   setosa
## 4         4.6         3.1         1.5         0.2   setosa
## 5         5.0         3.6         1.4         0.2   setosa
## 6         5.4         3.9         1.7         0.4   setosa
```

## The RData file format

- Any data frame (in reality any R object) may be saved in a RData file

```
dat <- data.frame(x = rnorm(10), y = rnorm(10))
save(dat, file = "exp.RData")
```

- These data may be loaded back into R later

```
rm(dat) # remove dat from the computer memory
dat # just checking that they are not there

## Error: object 'dat' not found

load("exp.RData") # load them back from the file
head(dat) # here they are again!

##           x           y
## 1 -0.5808 -0.4523
## 2 -0.2021  1.1401
## 3  0.2965 -0.5049
## 4 -0.2556 -1.2219
## 5  0.7552  1.0521
## 6  0.4378 -2.2693
```

## Importing data from text files

- Text files are a frequent way of storing and sharing data sets
  - Rows of the file usually correspond two rows of the data set
  - Values in the columns stored in a single row separated by some special character
- Text files are frequently the easiest way of importing data into R from other tools

## CSV files

The values on each line are separated by commas

File “x.csv”

ID, Name, Age
23424, Ana, 45
11234, Charles, 23
77654, Susanne, 76

To read the data into a data frame:

```
dat <- read.csv("x.csv",
                header=TRUE)

dat

##      ID      Nome  Idade
## 1 23424      Ana     45
## 2 11234   Carlos     23
## 3 77654   Susana     76
```

## Variations of the CSV format

On countries where the comma is used as a decimal separator it is common to use the semi-colon to separate values

File “y.csv”

```
ID; Name; Grade
23424; Ana; 4,6
11234; Charles; 12,3
77654; Susanne; 15,9
```

Reading this into a data frame:

```
dat <- read.csv2("y.csv")
dat

##           ID      Nome Nota
## 1 23424      Ana  4.6
## 2 11234   Carlos 12.3
## 3 77654   Susana 15.9
```

## Other text formats

File “z.txt”

```
ID Name Age Phone
23424 Ana 40 ???
11234 Charles 12 34567678
77654 Susanne 45 23435567
```

To read this data into a data frame we do:

```
dat <- read.table("z.txt", header = TRUE, na.strings = "???",
dat

##           ID      Name Age      Phone
## 1 23424      Ana  40      NA
## 2 11234   Charles  12 34567678
## 3 77654   Susanne  45 23435567
```

or

```
dat <- read.table("z.txt", header = TRUE, na.strings = "???",
colClasses = c("character", "character", "integer", "character"))
```

## Summary on text formats

- Family of functions with similar syntax and parameters
  - `read.table`, `read.csv`, `read.csv2`, `read.delim`, etc.
- Some of the more relevant parameters
  - `sep` indicates the character used as values separator
  - `dec` indicates the character used as decimal separator
  - `header` indicates whether the first line contains the column names
  - `na.strings` indicates the character of vector of characters that should be interpreted as unknown values
- The result of these functions is a data frame
- *character encodings* - potential source of problems
- Other functions : `readLines`, `scan`

### Importing from Spreadsheets

## Importing data from SpreadSheets

### Method 1 - ODBC connection (Windows)

- ODBC is a communication protocol between data bases
- Microsoft Excel is able to communicate using this protocol

### Installation

- Requires the installation of package `RODBC`

### Example

```
library(RODBC)
fc <- "c:\\Documents and Settings\\xpto\\My Documents\\calc.xls"
cn <- odbcConnectExcel(fc)
shs <- sqlTables(cn)
dat <- sqlQuery(cn, paste("SELECT * FROM", shs$TABLE_NAME[1]))
```

# Importing data from SpreadSheets

## Method 2 - package *gdata*

- Package *gdata* includes the function `read.xls`

## Installation

- Requires the installation of package *gdata*
- Requires the availability of Perl on the computer (standard on Mac OSx and Linux, but not on Windows)

## Example

```
library(gdata)
fc <- "c:\\Documents and Settings\\xpto\\My Documents\\calc.xls"
dat <- read.xls(fc, sheet = 1)
```

- The help page of function `read.xls` has several other examples

# Importing data from SpreadSheets

## Method 3 - using CSV format

- One way of sending some data into R consists in saving the wanted spreadsheet data on a text file, for instance in format CSV
- And then, inside R, use some of the previously explained methods to import the data into a data frame from the saved text file

# Importing data from SpreadSheets

Method 4 - through the *clipboard*

- When we want to import small data tables in a spreadsheet this is the easiest process

	A	B	C	D	E
1					
2					
3		ID	Nome	Nota	
4		45676	Ana	12.5	
5		65677	Carlos	3.75	
6		53455	Joana	17.3	
7					
8					
9					

Select the table and do  
Edit+Copy

```
dat <- read.table("clipboard",
  header = TRUE)
```

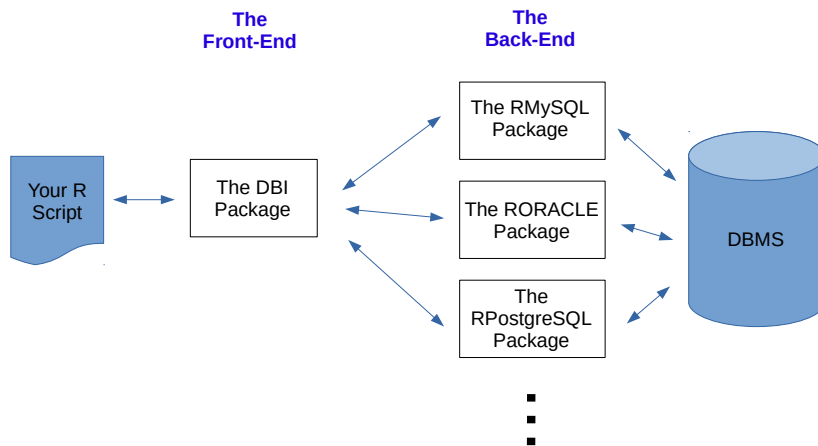
```
dat
##      ID  Nome  Nota
## 1 45676   Ana 12.50
## 2 65677 Carlos  3.75
## 3 53455   Joana 17.30
```

## Connecting with Data Bases

# Connecting with Data Bases

The *DBI* package

- *DBI* provides a front end interface to DBMS-specific drivers



## An Example with MySQL

```

library (DBI)
library (RMySQL)
drv <- dbDriver ("MySQL") # Loading the MySQL driver
con <- dbConnect (drv, dbname="transDB", # connecting to the DBM
                  username="myuser", password="mypasswd",
                  host="localhost")

# getting the results of a query as a data frame
data <- dbGetQuery (con, "SELECT * FROM clients")

dbDisconnect (con) # closing up stuff
dbUnloadDriver (drv)

```

## Another Example with Results in Chunks

```

library (DBI)
library (RMySQL)
drv <- dbDriver ("MySQL") # Loading the MySQL driver
con <- dbConnect (drv, dbname="transDB", # connecting to the DBM
                  username="myuser", password="mypasswd",
                  host="localhost")

res <- dbSendQuery (con, "SELECT * FROM transactions")
while (!dbHasCompleted (res)) {
  # get the next 50 records on a data frame
  someData <- fetch (res, n = 50)
  # call some function that handles the current chunk
  process (someData)
}
dbClearResult (res) # clear the results set

dbDisconnect (con) # closing up stuff
dbUnloadDriver (drv)

```



## Other forms of data importation

- Data bases
  - R has interfaces to all major DBMS (packages `DBI`, `RMySQL`, `ROracle`, etc.)
- Other statistical software
  - Importing from Minitab, S-Plus, SPSS, Stata, SAS, etc.
  - Packages `foreign`, `Hmisc`
- Several other formats / software.
- More details / information in the manual *R Data Import/Export* that comes with R

## Hands On Data Import - Audiology Data

- The site <https://archive.ics.uci.edu/ml/datasets/Audiology+%28Standardized%29> contains a data set of an Audiology problem
  - 1 Download the data set `audiology.standardized.data` to a local file and import that data into an R data frame. Do read the information on the web page, particularly the information on how unknown values are represented and make sure they are properly translated into R nomenclature.