Programming for Cognitive Science

Lecture 3 – R for data analysis

Joanna Tobiasz, PhD
Anna Papiez, PhD

Department of Data Science and Engineering

Plan for today

01 Data import/export

02 Data cleaning

03 High-dimensional data analysis

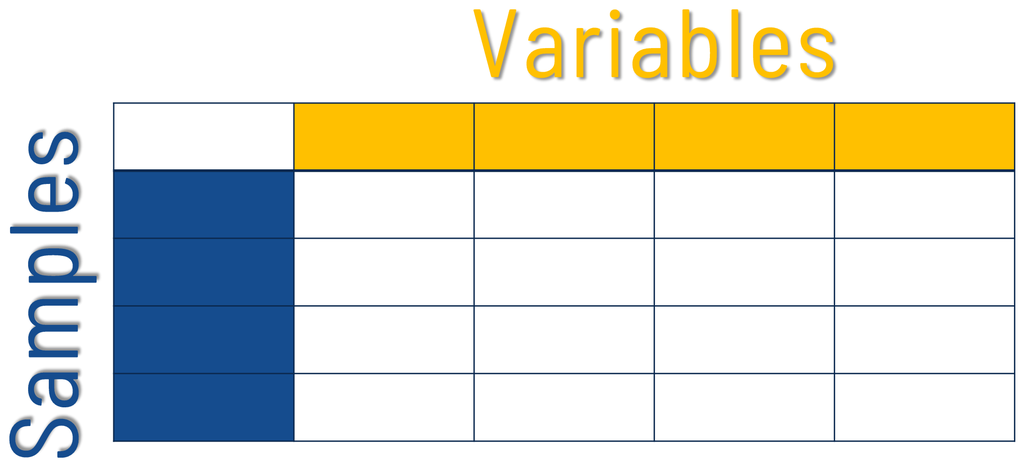
04 Parallelization

Part 1

Data import/export

Data import/export

Usually, the data have the form of a matrix with observations/samples in rows and features/variables in columns.



We usually import the data from:

- packages
- text files
- Excel sheets

Importing data from packages

To use an installed package:

```
library(package_name)
```

To use a dataset from an installed and loaded package:

```
data("dataset_name")
attach(dataset_name)
```

Importing data from text files

```
read.table(file, header = FALSE, sep = "", dec = ".", …)
```

- Name of the file
- Does the first row contain names of the variables?
- The character used for field separation
- The character used for decimal points

Example:

```
Species,Weight,Length1,Length2,Length3,Height,Width
Bream,242,23.2,25.4,30,11.52,4.02
Bream,290,24,26.3,31.2,12.48,4.3056
Bream,340,23.9,26.5,31.1,12.3778,4.6961
Bream,363,26.3,29,33.5,12.73,4.4555
```

```
read.table(file, header = TRUE, sep = ",")
```

Importing data from text files

Other functions:

- `read.csv(file, header = TRUE, sep = ",", dec = ".",...)`
- `read.csv2(file, header = TRUE, sep = ";", dec = ",",...)`
- `read.delim(file, header = TRUE, sep = "\t", dec = ".",...)`
- `read.delim2(file, header = TRUE, sep = "\t", dec = ",",...)`

Tabulator

Exporting data to text files

```
write.table(x, file = “file_name", append = FALSE,
    quote = TRUE, sep = " ", eol = "\n",
    na = "NA", dec = ".",
    row.names = TRUE, col.names =   TRUE, …)
```

Name of object to save

Name of the file

Should we modify the existing file or create a new one?

Should values be in quotes?

The character for field separation

The character to mark the end of each row: “\n” – new line/ENTER

What should be printed instead of missing values?

The character used for decimal points

Should the row and column names be saved?

Exporting data to text files

Other functions:

- `write.csv(...)`
- `write.csv2(...)`

Importing data from Excel sheets

- ```
  library(xlsx)
  read.xlsx(file, sheetIndex, sheetName = NULL, header = TRUE,
      ...)
  ```

Name of the file

Which sheet to import?

Does the first row contain names of the variables?

- ```
  library(openxlsx)
  read.xlsx(xlsxFile, sheet, colNames = TRUE, rowNames = FALSE,
      ...)
  ```

Name of the file

Which sheet to import?

Does the first row contain names of the variables?

Does the first column contain names of the sample?

Exporting data to Excel sheets

- ```
  library(xlsx)
  write.xlsx(x, file, sheetName = "Sheet1", append = FALSE,
  ```

Name of object to save

Name of the file

Name of the sheet

Should we modify the existing file or create a new one?

```
col.names = TRUE, row.names = TRUE, …)
```

Should the row and column names be saved?

- ```
  library(openxlsx)
  write.xlsx(x, file, ...)
  ```

Name of object to save

Name of the file

Silesian
University
of Technology

RESEARCH
UNIVERSITY
EXCELLENCE INITIATIVE
Ministry of Science
and Higher Education

Part 2

Data cleaning

Missing data

- `is.na(x)` – returns `TRUE` if the value is missing and `FALSE` otherwise

- `complete.cases(x)` – returns `FALSE` if the value is missing and `TRUE` otherwise

Let’s move on to coding...

Part 3

Statistical analysis

Probability distributions

Normal distribution:

- `dnorm` - density function
- `pnorm` - cumulative distribution function
- `qnorm` - quantile function (inverse cumulative distribution function)
- `rnorm` - random generator

Other available distributions:

- `unif` - uniform
- `binom` - binomial
- `chisq` - Chi-square
- `pois` - Poisson
- etc.

Statistical hypothesis testing

1) Is there a statistically significant difference in horsepower between cars with 8 cylinders and cars with 6 cylinders?
2) A lady claimed that she can tell only by the taste if milk or tea was poured into her cup first. Can she really?

Let’s move on to coding...

Part 4

Parallelization

Parallel computing

- When many multiple calculations are needed for different parameters, we can make them simultaneously (in parallel) at different cores instead of one after another in a loop.
- This can significantly reduce time of calculations.

mclapply {parallel}

Parallel version of `lapply`. Applies a function to each list element, returns list.

```
mclapply(L, FUN)
```

`L`	list
`FUN`	function

mclapply {parallel}

```
library(parallel)
L <- list(a = 1, b = 1:3, c = 10:100)
mclapply(L, length)
    $a
    [1] 1
    $b
    [1] 3
    $c
    [1] 91
```

%dopar% {doParallel}

One can use the %dopar% function to parallelize for loops. The result returned is a list:

```
library(doParallel)
cl <- makeCluster(2)
registerDoParallel(cl)
foreach(i=1:3) %dopar% sqrt(i)
stopCluster(cl)
```

Let’s move on to coding...

Silesian
University
of Technology

RESEARCH
UNIVERSITY
EXCELLENCE INITIATIVE
Ministry of Science
and Higher Education

I APPRECIATE YOUR ATTENTION