Data Analysis and Unsupervised Learning

Introduction to R

MAP573 – Julien Chiquet

École Polytechnique, Autumn semester, 2020-2021

https://jchiquet.github.io/MAP573

Outline

Outline

R ?

In a nutshell

`R` is a scientific software specialized in calculation and statistical analysis.

It is also

- a programming language,
- a environment/interpreter,
- an open-source project (`GNU`-R)
- a multi-plateform software (Linux, Mac, Windows)

A bit of history

Chronological approach

- 1970s: S-language developed at Bell labs (Chambers, Beckers)
- 1980s: S-PLUS developed at AT&T. Lab
- 1990s: R is developed as a GNU/GPL open-source counterpart to S by Gentleman and Ihaka (Auckland university)
- 1997: The R-core team now leads the development
- 2002: The R foundation is created and chaired by Gentleman and Ihaka
- 2011: first public beta version of R-studio (JJ Allaire and co)
- 2016: version 1.0 of Rstudio
- 2019: Rstudio lead scientist H. Wickham receives COPSS Award (statistician Nobel price)

R remarkable features

Scientific Computing

- linear algebra
- statistical models and data analysis

Data manipulation an visualization

- import, export, transformation
- great, versatile plotting system

Package manager

- easy to use, portable
- community driven

Interfacing is "easy"

- with most programming languages (C/C++, Python)
- with most database systems (SQL, postgrey)
- for distributed computing (Hadoop, H20, spark)

Why R ?

Accessibility

- easy to learn even for non-statistician/data scientist

Packages manager

- more than 13,000 community-based libraries
- cutting-edges statistical methods

Reproducibility

- Rmarkdown is not just a notebook
- Great for interfacing, plotting, making scientific reports
- R dev/package well integrated on github, binder

The community

- CRAN community https://cran.r-project.org/
- Rstudio community https://community.rstudio.com/
- SatRDay, R user groups, meet-up, conferences, Twitter

Why not R?

Some limitations

- Easy to write dirty (unreadable) code
- Easy to write inefficient code
- Many useless packages
- Not a typed language and no compilation by default: may be slow
- Many ways to do the same things
- Less well interfaced to GAFAM ML/Deep-learning libraries than Python

So why R again?

The Rstudio group

Even if it is a company...

- Rstudio IDE is a great all-in-one tool for data analysis and development
- Towards cleaner/consistent implementation (`tidyverse` and co)
- Many features from computer science (unitary test, github integration)
- Enhanced interfaces to ML/DL tools (Tensor Flow, Keras, Torch, etc.)
- Great interface with with Python (package `reticulate`)

The `R` community

- CRAN taskviews and `R` foundation
- Great interface with with `C++` (packages `Rcpp`, `RcppArmadillo`)
- Nice surrogate Oriented-Object programming with R6

$\rightsquigarrow$ These features basically saved `R` (for how long?)

Outline

Setup instructions I

R and **RStudio** are separate downloads and installations

- R is the underlying statistical computing environment
- RStudio is a graphical integrated development environment (IDE)

Windows

1. Download R from the CRAN website and
2. Run the `.exe` file that was just downloaded
3. Go to the RStudio download page
4. Under *Installers* select **RStudio x.yy.zzz - Windows Vista/7/8/10**

Setup instructions II

MacOS

1. Download R from the CRAN website.
2. Select the `.pkg` file for the latest R version and double click
3. Go to the RStudio download page
4. Under *Installers* select **RStudio x.yy.zzz - Mac OS X 10.6+ (64-bit)**

Linux

1. Follow the CRAN instructions, to update your /etc/sources.list
2. On Debian/Ubuntu, run `sudo apt-get install r-base`
3. Go to the RStudio download page
4. Under *Installers* select the version that matches your distribution

The R console

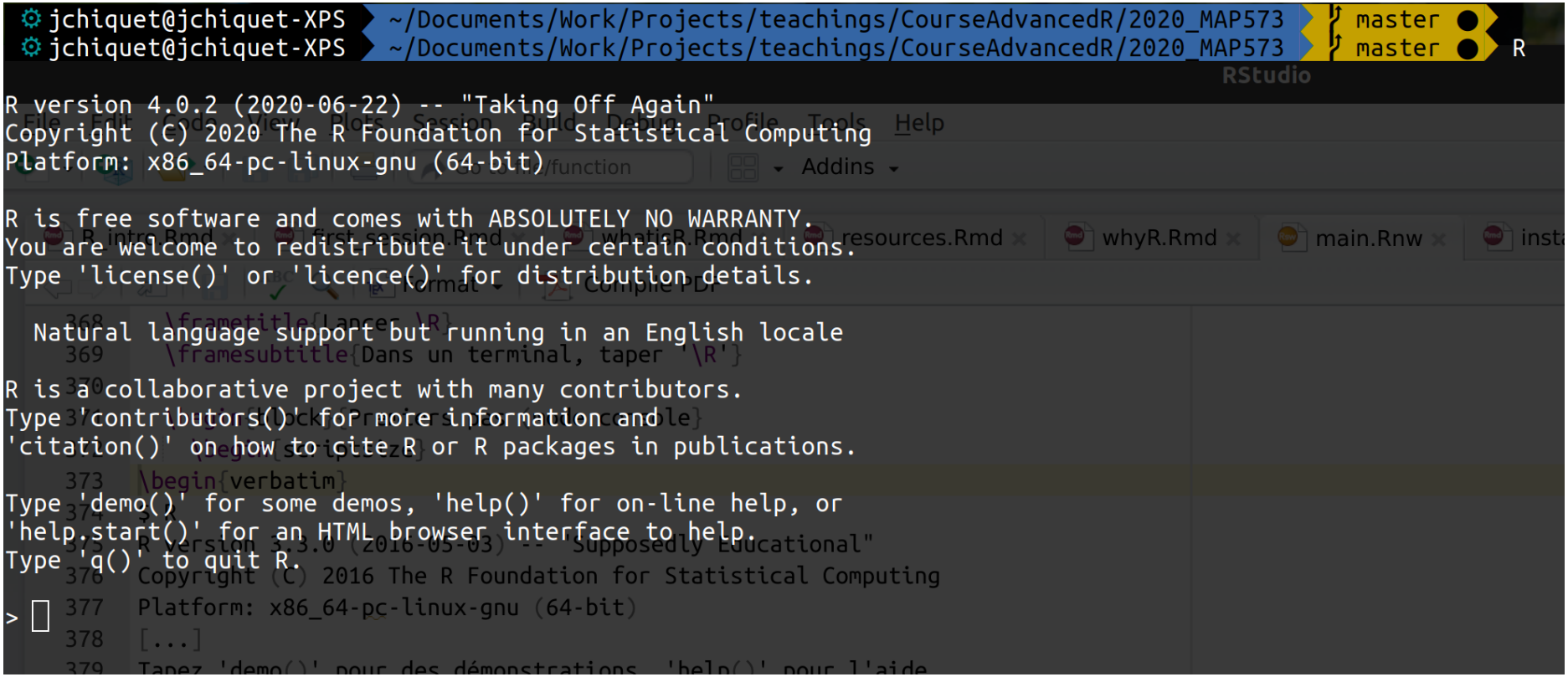


Figure 1: Screenshot of the R console

- `help(str)`, `?str`: launch dedicated help for command `str`,
- `help.search("factorial")`, `??factorial`: look for command with key word `factorial`,
- `help.start()`, `???`: launch the HTML help pages in a browser

The Rstudio IDE

- A full IDE with code, interpreter, workspace and plots
- Code completion (even for C++, Python)
- Debugger
- Package development and external code integration are easier
- Notebooks integration with Rmarkdown
- Interface with github

$\rightsquigarrow$ Rstudio is a state-of-the-art tool for efficient development in `R`

The Rstudio IDE

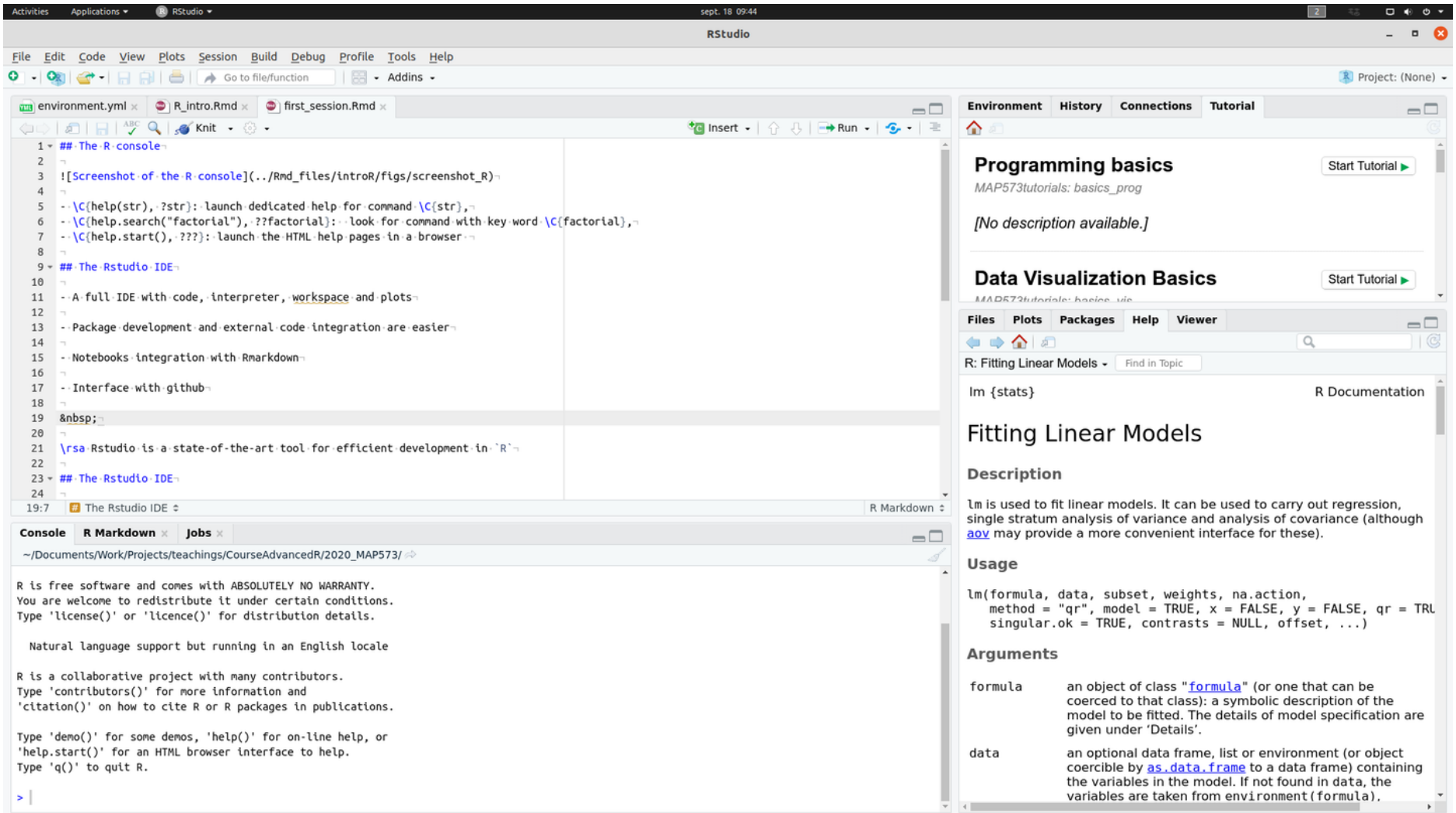


Figure 2: Screenshot of the Rstudio IDE 1.3

Rstudio hints

Shortcuts: see see 'Alt + Shift + K'

- `Ctrl + Return`: execute current selection in console
- `Ctrl + 1/2/3/4`: navigate between panels
- `Ctrl + Shift + K`: knit current document

Integrated Help

- "Cheatsheets" for data manipulation/plots
- Markdown reference
- Roxygen reference

Good practices

- Comment your code
- Create projects whenever possible
- Use relevant variables/files names
- Use notebooks to perform and comment a data analysis
- Write a function as soon as you repeat some code 3 times
- Coding style: https://google.github.io/styleguide/Rguide.html

Academic resources

Conferences

- UseR, annual conference of the R foundation conference
- SatRday, community-led, regional conferences
- Rstudio-conf, annual conference of the Rstudio community

Journals

- The R journal http://journal.r-project.org/
- The Journal of Statistical software https://www.jstatsoft.org/

Important web resources

Institutional

- R fondation web site: http://www.r-project.org/
- CRAN (Comprehensive R Arxiv Network): http://cran.r-project.org/
- Rstudio Community https://rstudio.com

Community

- https://ropensci.org/ promotes reproducible science
- `R` user groups, meet-up, conference
- Stackoverflow https://stackoverflow.com/

Blogs and plateforms

- `Swirl`, plateform + package to learn R in R https://swirlstats.com/
- `Datacamp`, online teaching plateform https://www.datacamp.com/
- Rstudio education program https://education.rstudio.com/
- Blogs community-driven http://www.inside-r.org/, http://www.r-statistics.com/, http://www.r-bloggers.com/
- Many books freely available at https://bookdown.org/

Outline

R markdown

Figure 3: an authoring framework for data science

R Markdown?

- Markdown is a **lightweight markup language** with plain text formatting syntax that can be converted to HTML. It is completely independent from R. The extension is typically `.md`.
- R Markdown is an *extension* of the markdown that **enables R code to be executed**. The extension is typically `.Rmd`.
- `rmarkdown` is a *package* which processes and **converts '.Rmd' files** into a number of different formats, including HTML or `.pdf`. The core function is `rmarkdown::render()`.
- `knitr` is a *package* which processes plain text document **with embedded code, executes the code** and 'knits' the results back into the document. The core function is `knitr::knit()`.

```
install.packages("rmarkdown")
install.packages("knitr")
```

How does it work?

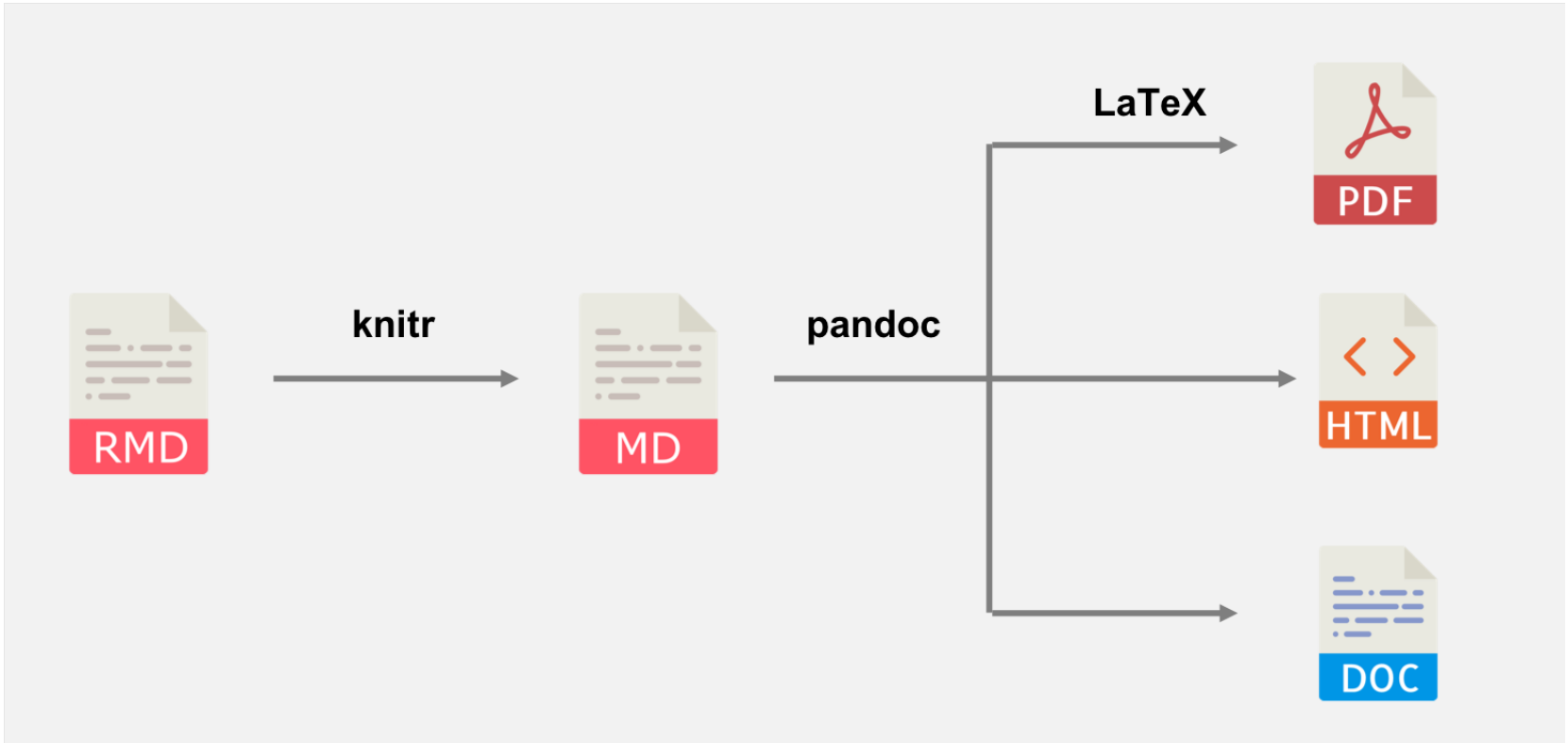
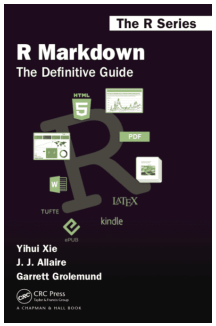


Figure 4: R Markdow workflow

References

Rmarkdown: Dynamic Documents for R (Allaire et al., 2020), https://bookdown.org/yihui/rmarkdown/



Knitr: A General-Purpose Package for Dynamic Report Generation in R (Xie, 2020), https://yihui.name/knitr/

Rstudio doc

See https://rmarkdown.rstudio.com/

R Markdown possibilities

See https://rmarkdown.rstudio.com/

Handle various inputs

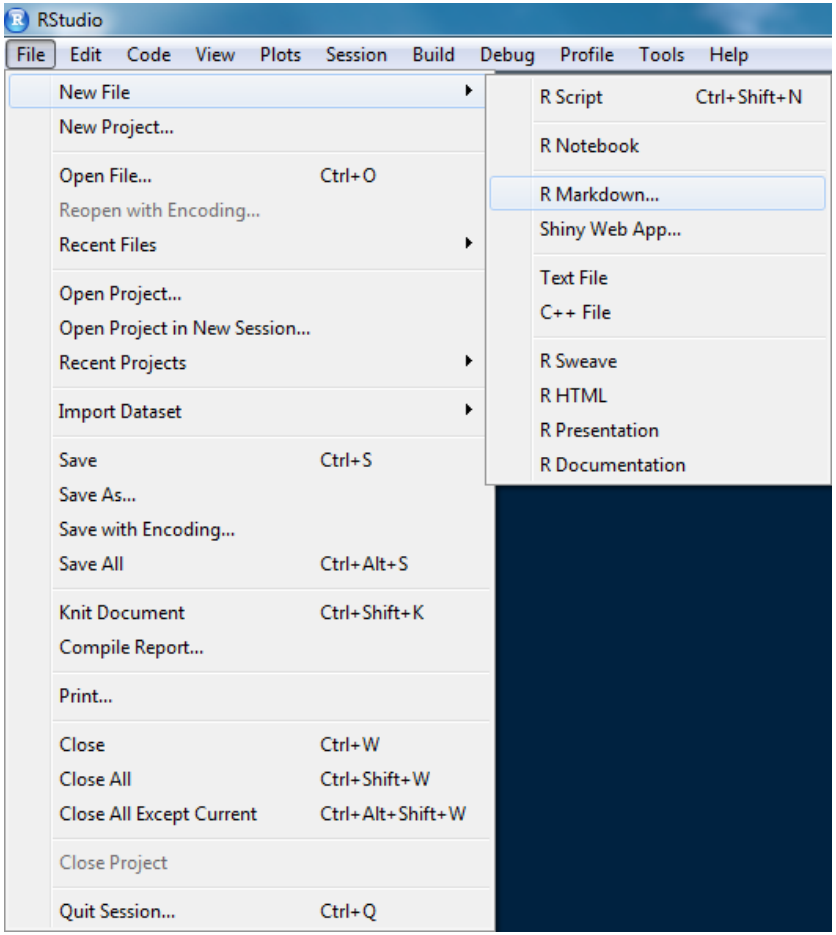
- Markdown Syntax (Markdown reference cheat sheet)
- LaTeX(Advanced mathematical expressions)
- HTML/javascript
- Code chunks (R, Python, Julia and more)

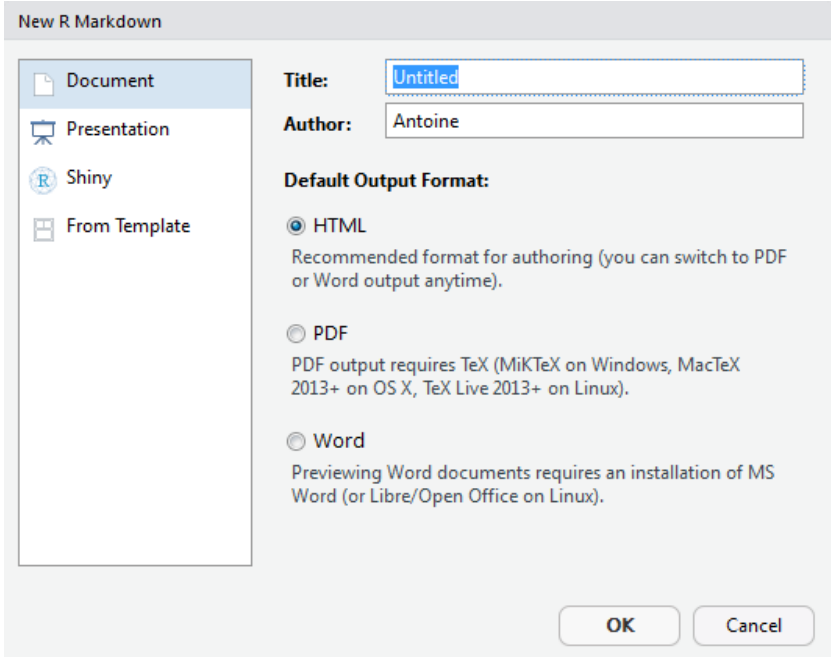
Handle various output

- Rstudio Notebook
- HTML report (static, dynamic)
- HTML website (static, dynamic)
- PDF document
- Doc documents

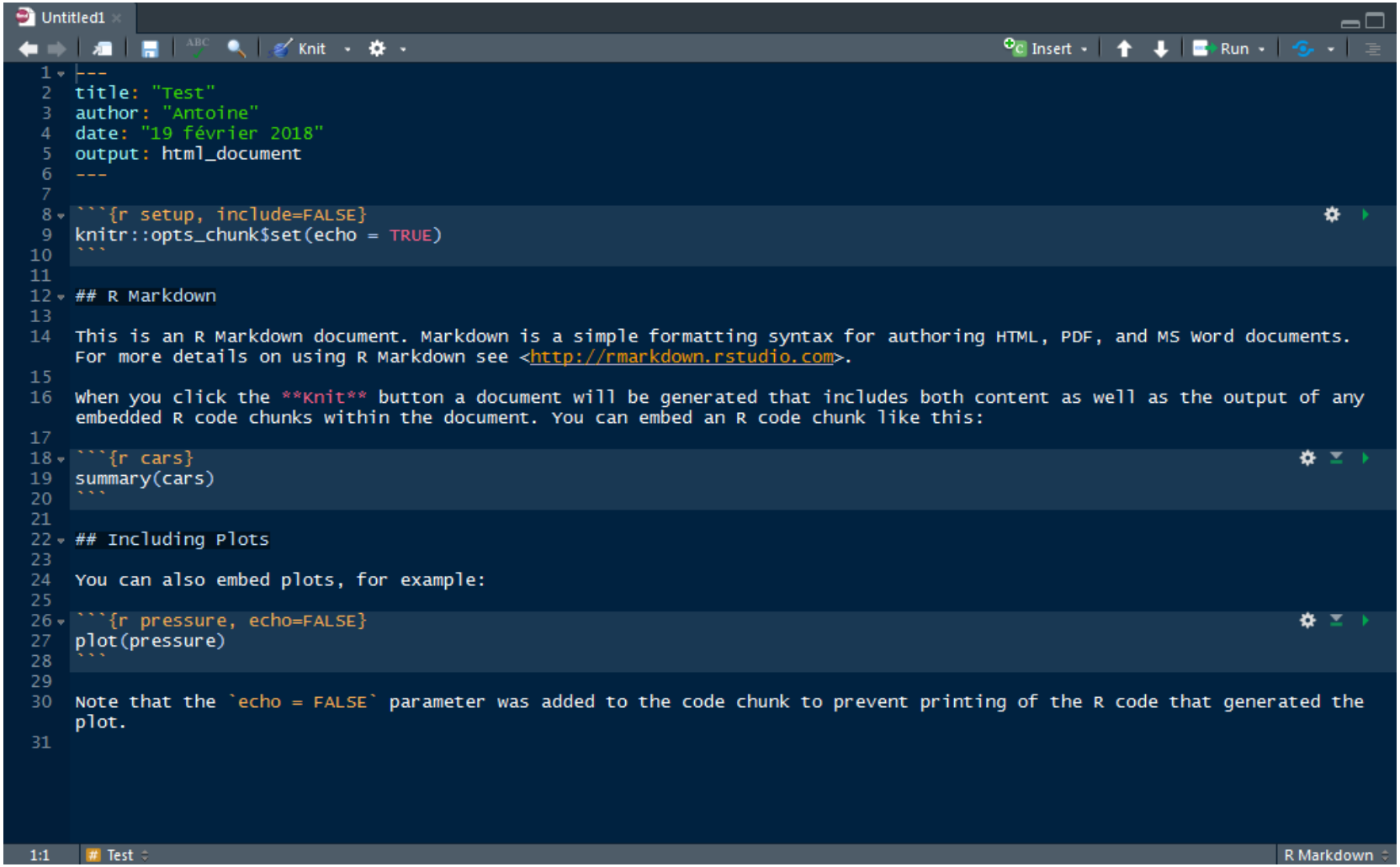
$\rightsquigarrow$ More than a Jupyter notebook

Create a new .Rmd





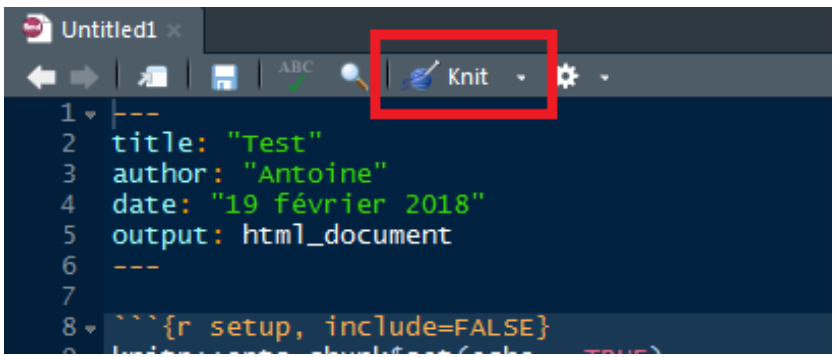
New .Rmd


Untitled1
Knit
Insert
Run
---
title: "Test"
author: "Antoine"
date: "19 février 2018"
output: html_document

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
R Markdown
This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
```{r cars}
summary(cars)
```
Including Plots
You can also embed plots, for example:
```{r pressure, echo=FALSE}
plot(pressure)
```
Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
1:1
Test
R Markdown

Compile `.Rmd`

Use the `Knit` button to produce a HTML file

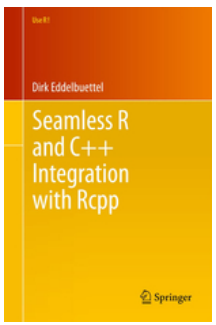


Shortcut: Ctrl + Maj + K

Outline

Rcpp

Seamless R and C++ integration with Rcpp (Eddelbuettel, 2013), for sale but see http://dirk.eddelbuettel.com



Interfacing C++ with R is *really* easy I

For a vector $\mathbf{x} = (x_1, \ldots, x_n)$, consider the simple task of computing

$$y_k = \sum_{i=1}^{k} \log(x_i), \quad k = 1, \ldots, n.$$

One can easily integrate some C++ version of this code with Rcpp.

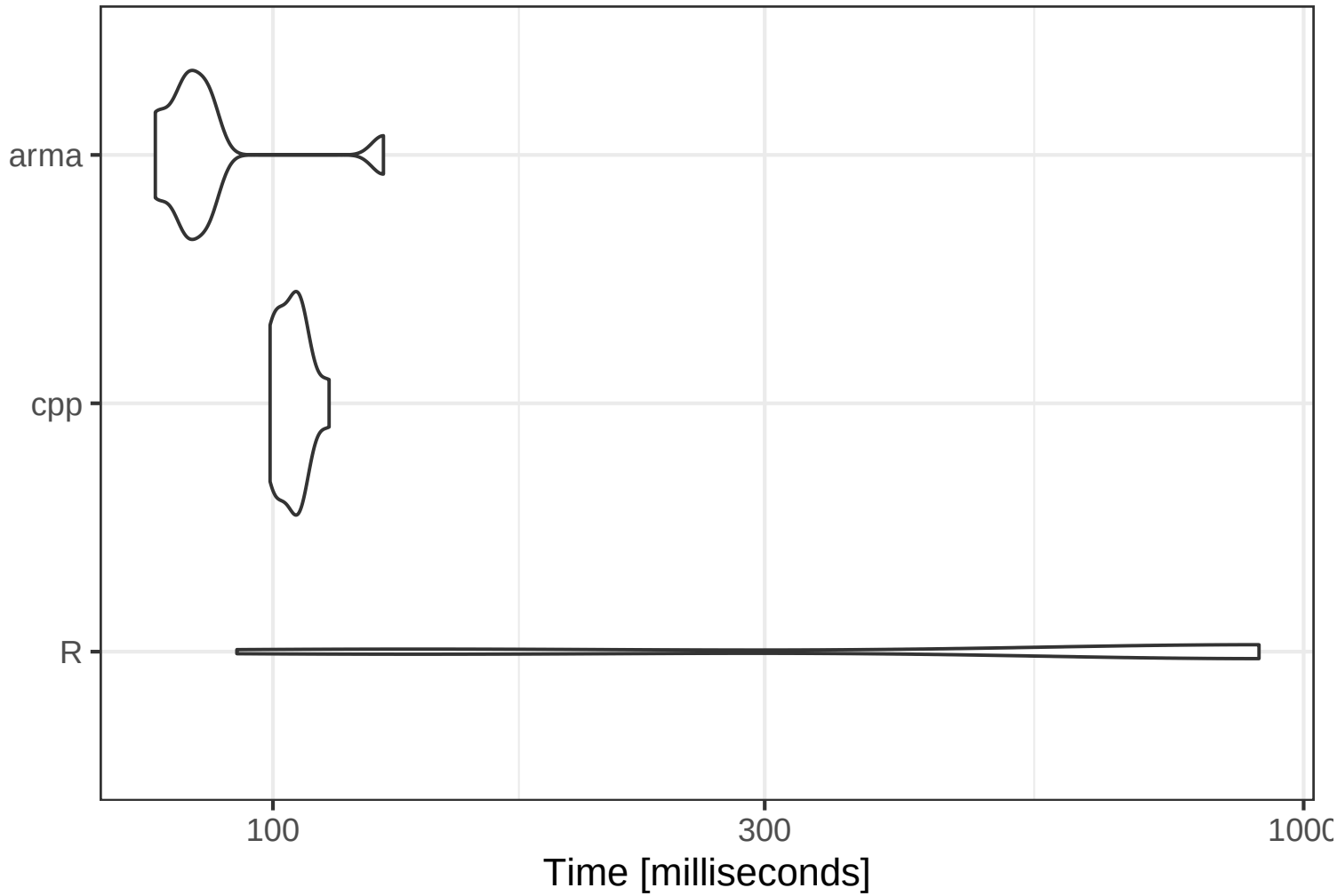
```
library(Rcpp)
rcpp <- cppFunction('NumericVector rcpp(NumericVector x) {
  using namespace Rcpp;

  int n = x.size() ;
  NumericVector res(x) ;
  res(0) = log(x(0));
  for (int i=1; i<n; i++) {
    res(i) = res(i-1) + log(x(i)) ;
  }
  return(wrap(res)) ;
}')
```

Interfacing C++ with R is *really* easy II

```
library(RcppArmadillo)
Arma <- cppFunction(depends = "RcppArmadillo", 'NumericVector Arma(NumericVector x) {
  using namespace Rcpp;
  using namespace arma;
  return(wrap(cumsum(log(as<vec>(x))))) ;
}')
```

```
x <-  runif(1e7, 1,2)
res <- microbenchmark(R = cumsum(log(x)), cpp = rcpp(x), arma = Arma(x), times = 10)
print(autoplot(res))
```

Interfacing C++ with R is *really* easy III



Reticulate

R interface to Python https://rstudio.github.io/reticulate/

Interfacing Python with R is *really* easy I

Configuring Python via reticulate

```
library(reticulate)
use_python("/usr/local/bin/python")
reticulate::py_config()
```

```
## python:         /home/jchiquet/.local/share/r-miniconda/envs/r-reticulate/bin/python
## libpython:      /home/jchiquet/.local/share/r-miniconda/envs/r-reticulate/lib/libpython3.6m.
## pythonhome:     /home/jchiquet/.local/share/r-miniconda/envs/r-reticulate:/home/jchiquet/.lo
## version:        3.6.11 | packaged by conda-forge | (default, Aug  5 2020, 20:09:42)  [GCC 7.
## numpy:          /home/jchiquet/.local/share/r-miniconda/envs/r-reticulate/lib/python3.6/site
## numpy_version:  1.19.1
```

First time, install

```
py_install("pandas")
```

Interfacing Python with R is *really* easy II

Performing analysis with Python

```
import pandas
flights = pandas.read_csv("flights.csv")
flights = flights[flights['dest'] == "ORD"]
flights = flights[['carrier', 'dep_delay', 'arr_delay']]
flights = flights.dropna()
```

Now use Python output with R

```
head(py$flights)
```

```
##    carrier dep_delay arr_delay
## 5       UA        -4        12
## 9       AA        -2         8
## 25      MQ         8        32
## 38      AA        -1        14
## 57      AA        -4         4
## 70      UA         9        20
```

Outline

Basic notions

R owns the same ingredients as most programming language

Functions

Functions have a special flavor in R (functional programming)

Variables, Objects

We manipulate object in R (as in OO programming)

Data structure

1. Atomic vector (integer, double, logical, character)
2. List (recursive vector)
3. Factor (vector with categories of values)
4. Matrix and array
5. Data Frame

Resources

- Advanced R, chapters I.2, I.3 (Wickham, 2014, http://adv-r.had.co.nz/)
- https://rbasics.netlify.app/

Your turn

I do not believe (anymore) in learning any programming language by reading slides

Basics programming

```
learnr::run_tutorial("basics_prog" , package = "MAP573tutorials")
```

Basics plotting

```
learnr::run_tutorial("basics_vis" , package = "MAP573tutorials")
```

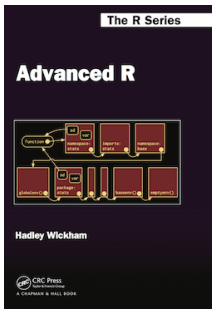
Swirl

```
library(swirl)
install_course_github("swirldev", "R_Programming_E")
swirl()
```

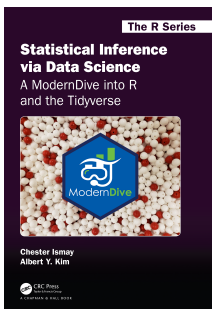
Going further

Great material available on the web

Advanced R (Wickham, 2014), http://adv-r.had.co.nz/



Modern dive into R (Ismay & Kim, 2020), https://moderndive.com/



Other personal links

- Base R: more comprehensive, faster
  https://github.com/jchiquet/CourseAdvancedR/
- Keep watch: workgroup https://stateofther.github.io/

Bibliographical references I

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., ... Chang, W. (2020). *Rmarkdown: Dynamic documents for R*. Retrieved from https://bookdown.org/yihui/rmarkdown

Burns, P. (2012). *The r inferno*. Lulu. com. Retrieved from http://www.burns-stat.com/documents/books/the-r-inferno/

Eddelbuettel, D. (2013). *Seamless R and C++ integration with Rcpp*. Springer. Retrieved from http://dirk.eddelbuettel.com

Gandrud, C. (2016). *Reproducible research with R and Rstudio*. Chapman; Hall/CRC. Retrieved from https://github.com/christophergandrud/Rep-Res-Book

Gillespie, C., & Lovelace, R. (2016). *Efficient R programming*. " O'Reilly Media, Inc.". Retrieved from https://bookdown.org/csgillespie/efficientR/

Ismay, C., & Kim, A. Y. (2020). *Statistical inference via data science: A moderndive into r and the tidyverse*. " CRC Press". Retrieved from https://moderndive.com/

Bibliographical references II

R Core Team. (2017). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Wickham, H. (2014). *Advanced r*. CRC Press. Retrieved from http://adv-r.had.co.nz/

Wickham, H. (2015). *R packages: Organize, test, document, and share your code*. " O'Reilly Media, Inc.". Retrieved from http://r-pkgs.had.co.nz/

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. Springer. Retrieved from http://ggplot2.tidyverse.org/reference/

Wickham, H., & Grolemund, G. (2016). *R for data science: Import, tidy, transform, visualize, and model data*. " O'Reilly Media, Inc.". Retrieved from http://r4ds.had.co.nz

Wilkinson, L. (2006). *The grammar of graphics*. Springer Science & Business Media.

Bibliographical references III

Xie, Y. (2020). *Knitr: A general-purpose package for dynamic report generation in R* (Vol. 29). CRC Press. Retrieved from https://yihui.name/knitr/