

Data Science with R

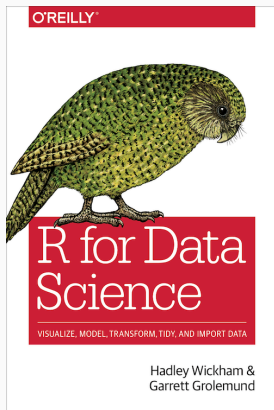
in the *tidyverse*

Matteo Sostero

May 18, 2018

Workshop material: <https://git.io/comos-r>

Sant'Anna School of Advanced Studies, Pisa

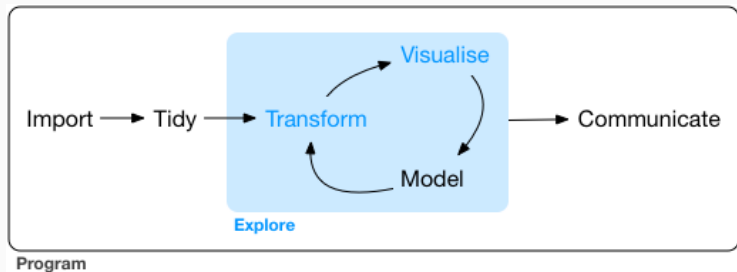


- Workshop material:
<https://git.io/comos-r>
- RStudio *Cheat sheets*:
<https://rstudio.com/resources/cheatsheets/>
- Book *R for Data Science*
by Garrett Grolemund, Hadley Wickham:
 - Online version: <http://r4ds.had.co.nz/>
 - Paperback: *R for Data Science*, O'Reilly Media, 2017.

Workshop outline

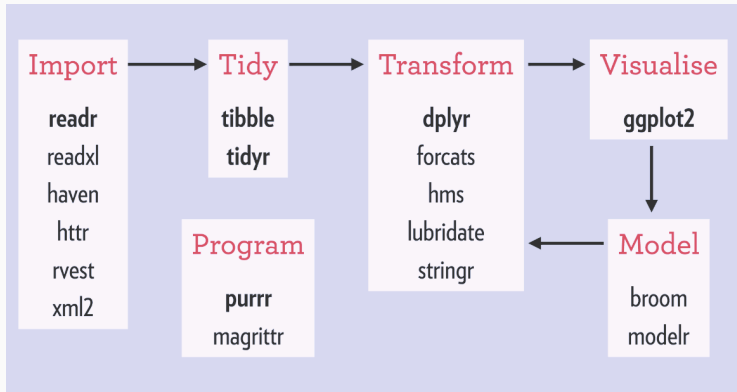
- Today:
 - Overview of Data Science workflow
 - *tidyverse* fundamentals
 - Data transformation with *dplyr*
 - Merging datasets with *join* operations
 - Data input strategies with *readr*
- Tomorrow:
 - String manipulation with *stringr*
 - Functional programming with *purrr*
 - Handling categorical variables with *forcats*
 - Data tidying with *tidyr*
- Friday:
 - Data visualization with *ggplot2*
 - Model estimation and selection with *broom*
 - Web scraping with *rvest*
 - ? More dataviz? (maps)

The Data Science Workflow



Credit: *R for Data Science*

The Tidyverse Ecosystem

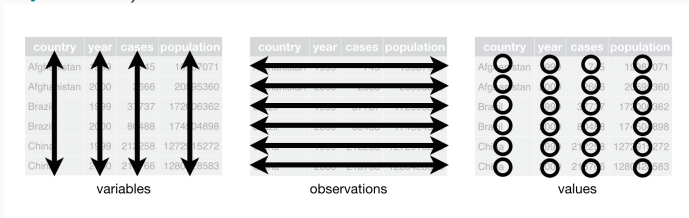


Credit: [Joseph Rickert](#)

Tidyverse Philosophy

The tidyverse is organised around a few principles:

- **Tidy data:** each variable is a column, each observation is a row, and each type of observational unit is a table (see also [article by Hadley on JSS](#));



Tidy data graphic from [R for Data Science](#)

- **Code readability:** consistent, expressive, verb-oriented syntax;
- **Functional programming:** functions applied to all elements of objects (no iterators); pass data along function pipeline `%>%`;
- **Compatibility:** classes and functions are mostly backward compatible with “Base R”.

Key differences with “Base R”

- `tibble` replaces `data.frame` class for rectangular datasets:
 - better preview printing;
 - “lazy and surly”: less type coercion (strings not converted to factors); doesn't change variable names;
 - doesn't use `row.names`
- style: function names are in `snake_case`
e.g.: `read_csv()` instead of `read.csv()`
- data is first argument of functions.

Building the pipeline

Tidyverse uses the *pipe* `%>%` to concatenate operations on data.

`%>%` builds chains of function by passing (“piping”) the **output** of one function (ie, data) as **input** of the next function.

Pseudocode: pipeline for baking a cake

```
ingredients %>%           {flour, water, eggs}  
  blend() %>%             dough  
  cook() %>%              whole cake  
  slice()                 slice of cake
```

The pipe composes functions: $x \%>\% f() \%>\% g() \equiv g(f(x))$

If a function `h(x,y)` has more than one argument (or data is not the first argument), `.` is an explicit placeholder: $y \%>\% h(x, .) \equiv h(x, y)$.

RStudio shortcut: `ctrl` + `⇧` + `M` or `⌘` + `⇧` + `M` on macOS.

Data manipulation with *dplyr*

In the tidyverse, *dplyr* provides a grammar of data manipulation, with consistent set of verbs that help you solve the most common data manipulation challenges:

- `mutate()` adds new variables that are functions of existing variables;
- `select()` picks variables based on their names;
- `filter()` picks cases based on their values;
- `summarise()` reduces multiple values down to a single summary;
- `arrange()` changes the ordering of the rows;

These all combine naturally with `group_by()` which allows to perform any operation by groups of values.

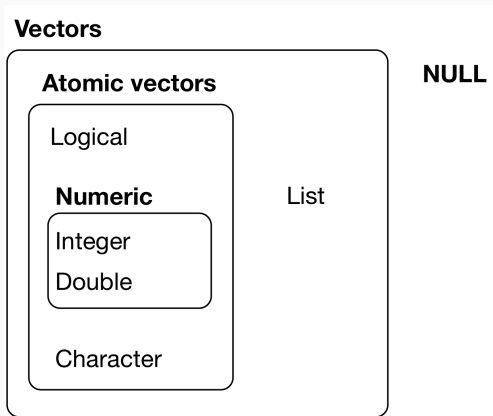
Vectors are a fundamental object class in R and come in two types:

- **Atomic vector** (homogeneous) of type:

1. *double*: most numbers
2. *integer*: integer numbers
3. *logical*: TRUE, FALSE, NA
4. *character*: strings
5. *complex*
6. *raw*

Integer and double vectors are also known as *numeric* vectors.

- **List** (heterogeneous) which are sometimes called *recursive vectors* because lists can contain other lists.



Functional programming with *purrr*

In *purrr*, the function `map(.x, .f)` to **maps** (ie, applies) a **function** `.f` to every element of an atomic vector or list `.x`

(loose) mathematical definition:

- Given a set (in R, a **vector**) $X = \{x_1, x_2, \dots, x_n\}$
- And a function $f(x) : X \mapsto ?$
- $\text{Map}(X, f) : \{f(x_1), f(x_2), \dots, f(x_n)\}$