# Introduction to the Tidyverse

Author

Date

# Today's outline

- What you have learnt so far
- Taking it a step further – introducing the Tidyverse
- Data wrangling with `dplyr`
- Data visualisation with `ggplot2`
- Final exercise
- Feedback & concluding remarks

# R and R Studio

"It is important to note the differences between R and RStudio. *R is a programming language* used for statistical computing while RStudio uses the R language to develop statistical programs. In R, you can write a program and run the code independently of any other computer program. *RStudio however, must be used alongside R in order to properly function*. Often referred to as an IDE, or *integrated development environment*, RStudio allows users to develop and edit programs in R by supporting a large number of statistical packages, higher quality graphics, and the ability to manage your workspace.

*R and RStudio are not separate versions of the same program*, and cannot be substituted for one another. R may be used without RStudio, but RStudio may not be used without R (Source)."

# What you have learnt so far

- Familiarity with the RStudio interface
- Working with scripts
- Simple data structures
- First data wrangling and visualisations

# Tidy data – a philosophy of data

- Wickham (2014) aimed to standardise the data preparation process

**«Each variable is a column, each observation is a row, and each type of observational unit is a table.»**

```
head(table1, n = 4) %>%
  kable(format = "html")
```

| country | year | cases | population |
|---|---|---|---|
| Afghanistan | 1999 | 745 | 19987071 |
| Afghanistan | 2000 | 2666 | 20595360 |
| Brazil | 1999 | 37737 | 172006362 |
| Brazil | 2000 | 80488 | 174504898 |

# Another example

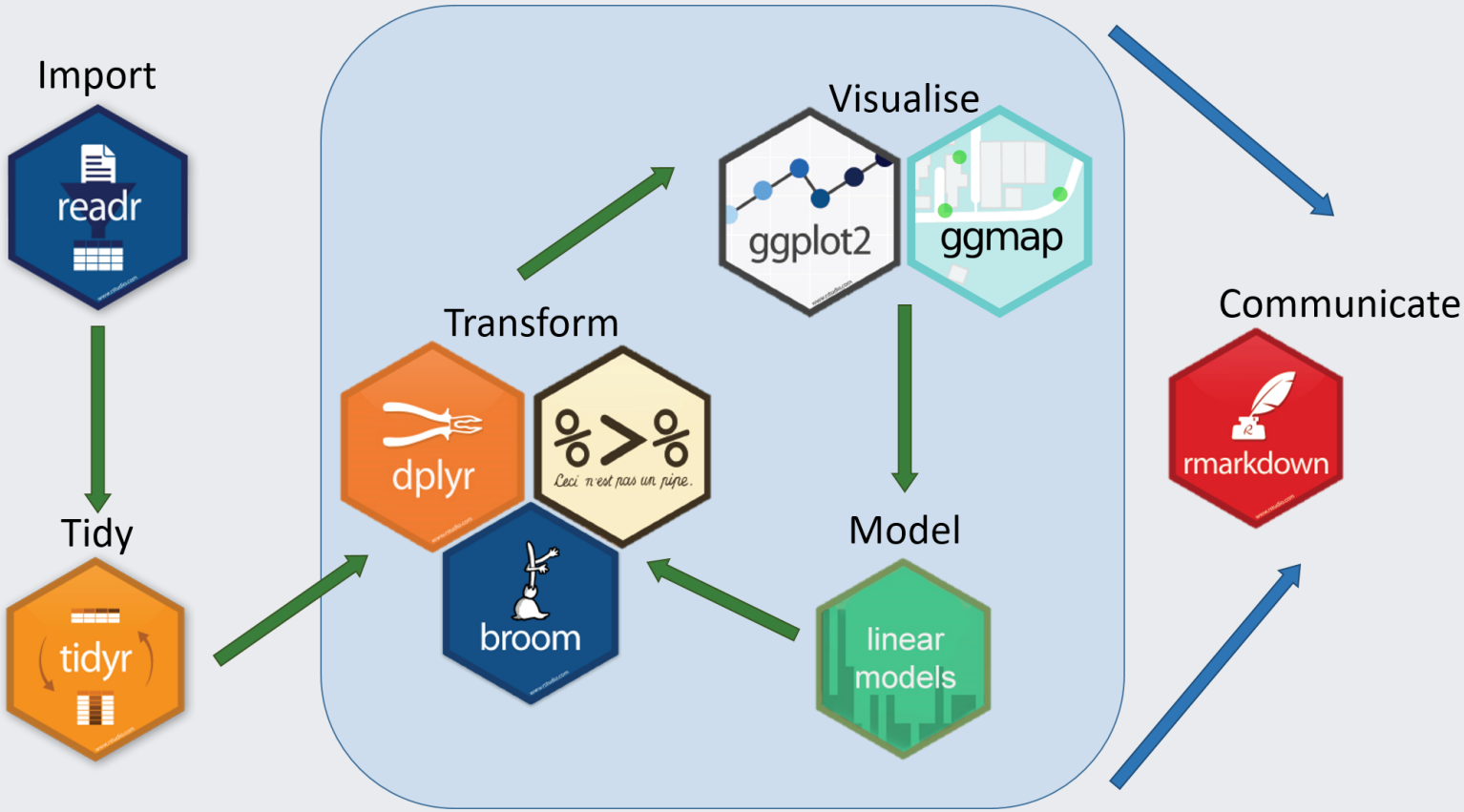
```
head(penguins, n = 5) %>%
  select(1:5) %>%
  kable(format = "html")
```

| species | island | bill_length_mm | bill_depth_mm | flipper_length_mm |
|---|---|---|---|---|
| Adelie | Torgersen | 39.1 | 18.7 | 181 |
| Adelie | Torgersen | 39.5 | 17.4 | 186 |
| Adelie | Torgersen | 40.3 | 18.0 | 195 |
| Adelie | Torgersen | NA | NA | NA |
| Adelie | Torgersen | 36.7 | 19.3 | 193 |

# From tidy data to the tidyverse

- The tidyverse is an opinionated collection of R packages designed to facilitate data science
- Its tools (read: packages) use/generate tidy data both as input/output
- `dplyr`: Helps to transform messy data to tidy data
- `ggplot2`: Uses tidy data to create graphics

# The whole workflow




Source: Teach Data Science

# Questions?

# How does Base R relate to the Tidyverse?

- Yesterday you learnt Base R commands, today Tidyverse commands
- Two different approaches to achieving the same goal
- **Treat them as friends!**
- A few examples

# Extract variables (columns)

### Base R

```
penguins[, c("island", "year")] # by name
penguins[, c(2, 8)]  # by column index
```

### Tidyverse

```
select(penguins, island, year) # by name
select(penguins, 2, 8)  # by column index
```

# Extract observations (rows)

### Base R

```
# Using [,]
penguins[penguins$body_mass_g > 3500 & penguins$island == "Torgersen",]
```

### Tidyverse

```
filter(penguins, body_mass_g > 3500 & island == "Torgersen")
```

# Filter rows with conditions evaluated within groups

In words: Penguin with maximum “bill_length_mm” for each “species”.

### Base R

```
# First operate in the data.frame by group (split-apply)
largest_bills <- by(penguins,
                    INDICES = penguins$species,
                    FUN = function(x){
                      x[x$bill_length_mm == max(x$bill_length_mm,
                                                na.rm = T), ]
                    })

# Then combine the results into a data.frame
do.call(rbind, largest_bills)
```

### Tidyverse

```
penguins %>%
  group_by(species) %>%
  filter(bill_length_mm == max(bill_length_mm, na.rm = TRUE))
```

# Questions?

# Introducing the pipe

- The pipe operator: `%>%`
- Takes one return value and feeds it in as an input to another function
- Think of it as: "and then"
- Helps you to write code that is easier to read and understand

```
penguins %>%
  select(species, island, sex, bill_length_mm, year)
```

```
penguins %>%
  select(species, island, sex, bill_length_mm, year) %>%
  filter(year == 2007)
```

```
penguins %>%
  select(species, island, sex, bill_length_mm, year) %>%
  filter(year == 2007) %>%
  group_by(island)
```

```r
penguins %>%
  select(species, island, bill_length_mm, year) %>%
  filter(year == 2007) %>%
  group_by(island) %>%
  summarise(bill_length = mean(bill_length_mm, na.rm = TRUE))
```

```
## # A tibble: 3 x 2
##   island    bill_length
##   <fct>           <dbl>
## 1 Biscoe           45.0
## 2 Dream            44.5
## 3 Torgersen        38.8
```

### Base R equivalent

```r
penguins <- penguins[,c("species", "island", "bill_length_mm", "year")]

penguins <- penguins[penguins$year == 2007, ]

penguins <- penguins[!(is.na(penguins$bill_length_mm)),]

tapply(penguins$bill_length_mm, penguins$island, mean)
```

```
##    Biscoe     Dream Torgersen
##  45.03864  44.53913  38.80000
```

# Questions?

# Recap & outlook

- Introduced to the tidy philosophy
- Tools of the Tidyverse
- Comparison of Base R and Tidyverse
- Next up: Data wrangling with `dplyr`

# Time to exercise!