R-eproducible workflows

1-day workshop
Morning practical session

Brendan Palmer,
Statistics & Data Analysis Unit,
Clinical Research Facility - Cork

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

CRF-C
HRB Clinical Research Facility Cork

HRB
Health Research Board

A: Tibbles

- The tidyverse equivalent of data.frames

4 main points of difference:

1. Printing in the console
2. Subsetting (The use of a placeholder (“.”)
3. Interacting with older code
4. Tibbles don’t change the input

- Open the script 01_am_tibbles.R

A: readr and more

- fast way to read rectangular data (like csv, tsv)
- read_csv(): comma separated (CSV) files
- read_tsv(): tab separated files
- read_delim(): general delimited files

- readxl supports both the legacy .xls format and the modern xml-based .xlsx format
- Need to load explicitly

- read_sas(): SAS files
- read_sav(): SPSS files
- read_dta(): Stata files
- Also need to load explicitly

- Open the script 02_am_readr.R

5481 Datasets

102 Publishers

Search Datasets...

Advanced Search

IRELAND'S **OPEN DATA** PORTAL

Promoting innovation and transparency through the publication of Irish Public Sector data in open, free and reusable formats.

Explore Datasets

- Agriculture, Fisheries, Forestry & Food
- Arts, Culture and Heritage
- Justice, Legal System, and Public Safety
- Economy and Finance
- Education and Sport
- Energy
- Environment
- Government and Public Sector
- Health
- Housing and Zoning
- Population and Society
- Science and Technology
- Regions and Cities
- Transport

Data structures

#vectors:
These come in two forms
- A: Atomic vectors contain exactly one type of data

```
all_numbers          <- c(1, 2, 0.5, -0.5, 3.4)

all_characters       <- c("One", "too", "3")

all_logical          <- c(TRUE, FALSE)
```

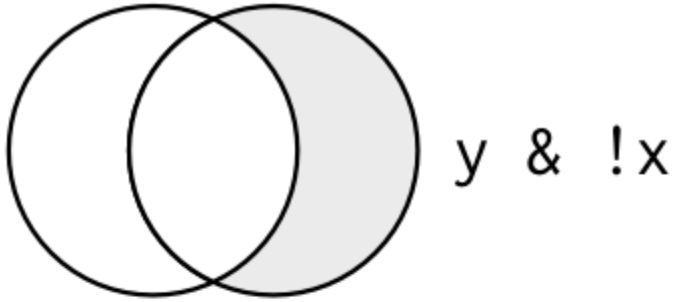
- B: Lists allow combinations of different types of data

```
this_is_a_list       <- list(1, TRUE, "Three")
```

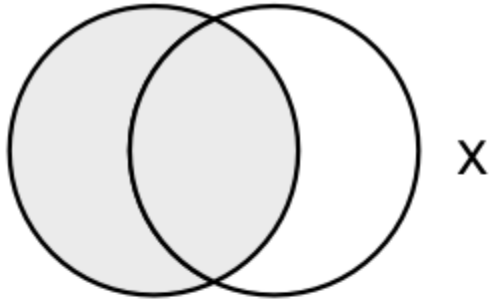
- If you try to create a vector with more than one data type, then it will undergo coercion to the least common denominator

- The coercion rule goes:
  logical -> integer -> numeric -> complex -> character

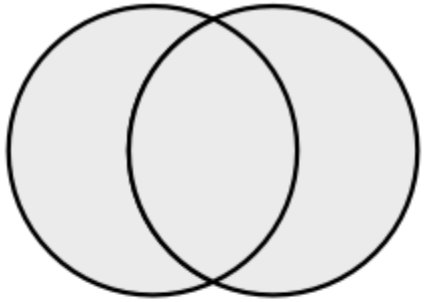
Logical operators and conditional subsetting



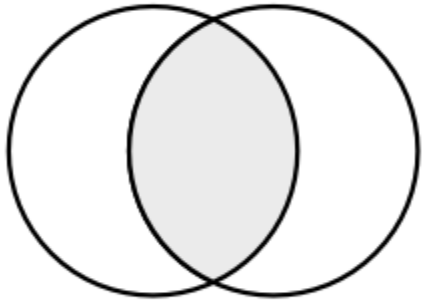
y & !x



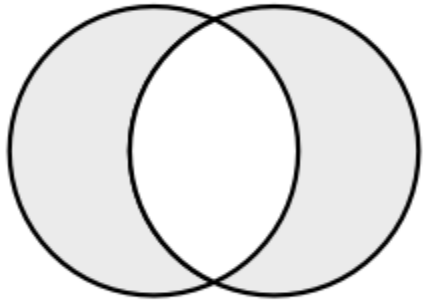
x



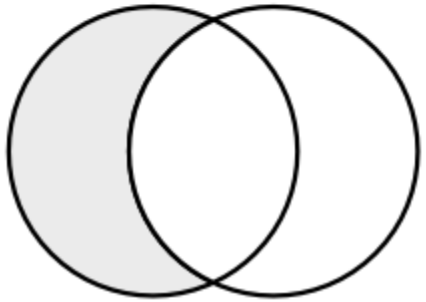
x | y



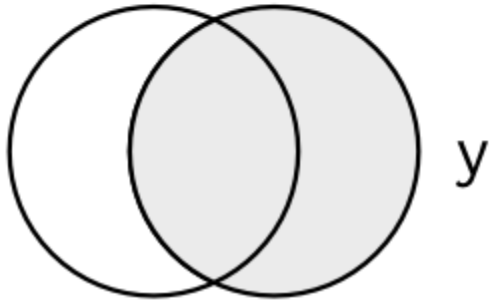
x & y



xor(x, y)



x & !y

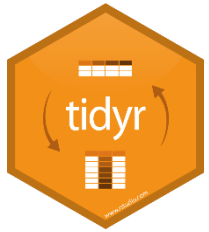


y

- & -> AND
- | -> OR (inclusive)
- ! -> NOT
- == -> EQUAL (identity)
- != -> NOT EQUAL

Image: Hadley Wickham R for Data Science Chapter 3

A: tidyr



- The goal is to create tidy data
    1. Each variable a column
    2. Each observation a row
    3. Each value is a cell

Main functions:
- gather()
- separate()

- Open the script 03_am_tidyr.R

Journal of Statistical Software
MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

Tidy Data

Hadley Wickham
RStudio

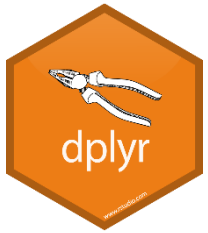
Abstract

A huge amount of effort is spent cleaning data to get it ready for analysis, but there has been little research on how to make data cleaning as easy and effective as possible. This paper tackles a small, but important, component of data cleaning: data tidying. Tidy datasets are easy to manipulate, model and visualise, and have a specific structure: each variable is a column, each observation is a row, and each type of observational unit is a table. This framework makes it easy to tidy messy datasets because only a small set of tools are needed to deal with a wide range of un-tidy datasets. This structure also makes it easier to develop tidy tools for data analysis, tools that both input and output tidy datasets. The advantages of a consistent data structure and matching tools are demonstrated with a case study free from mundane data manipulation chores.

Keywords: data cleaning, data tidying, relational databases, R.

DOI: 10.18637/jss.v059.i10

B: dplyr for data transformation



- Solves the most common data manipulation challenges

Main functions:
- select()
- filter()
- mutate()
- group_by()
- summarise()
- …… and many many more

- Open the script 04_am_dplyr.R

Time for some hands on application

- Open the script 05_am_practise.R

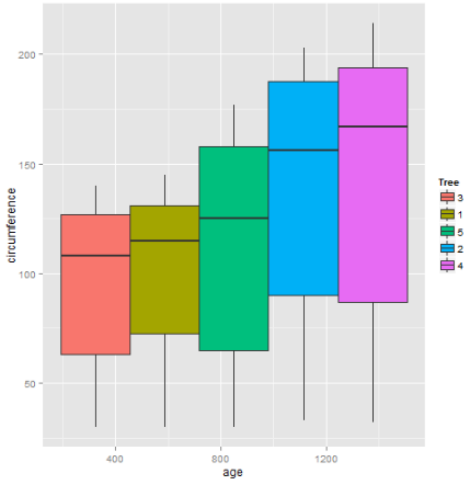
C: ggplot2

- Data visualisation based on "The Grammar of Graphics"

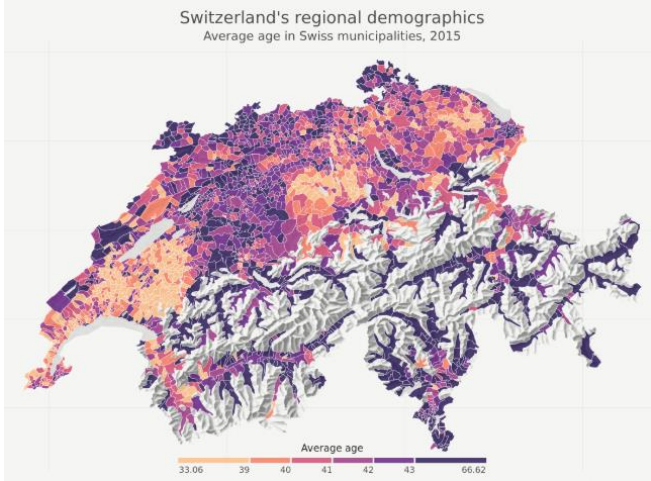
```
ggplot(data = <DATA>) +

    <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))    +

     linear model  +

     axes formatting  +

     legend formatting  +

     title  + etc. etc.
```

C: ggplot2

- Very versatile
- Allows you to go from

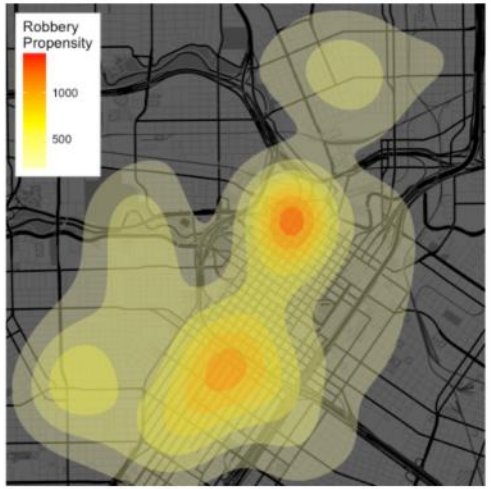


to

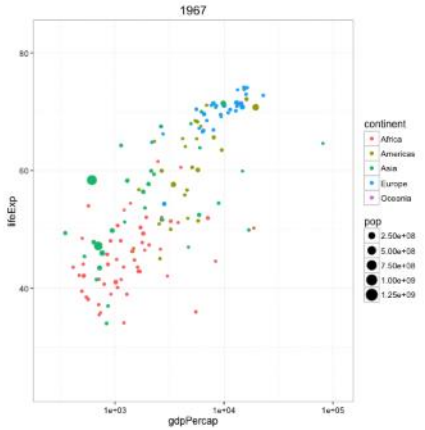


- Lots of add-on packages

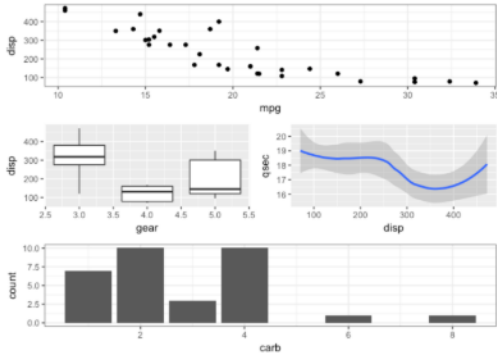
ggmap



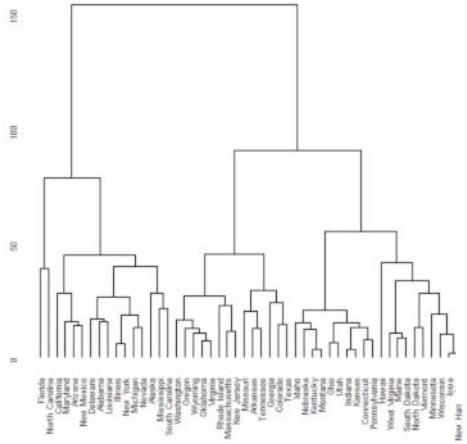
gganimate



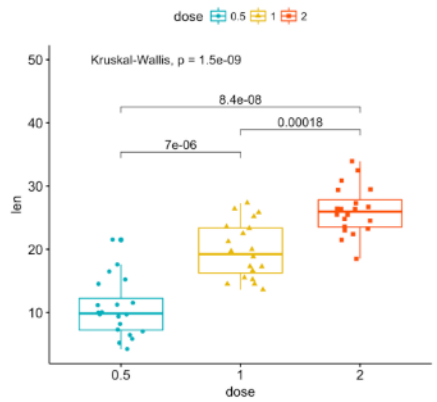
patchwork



ggdendro



ggpubr

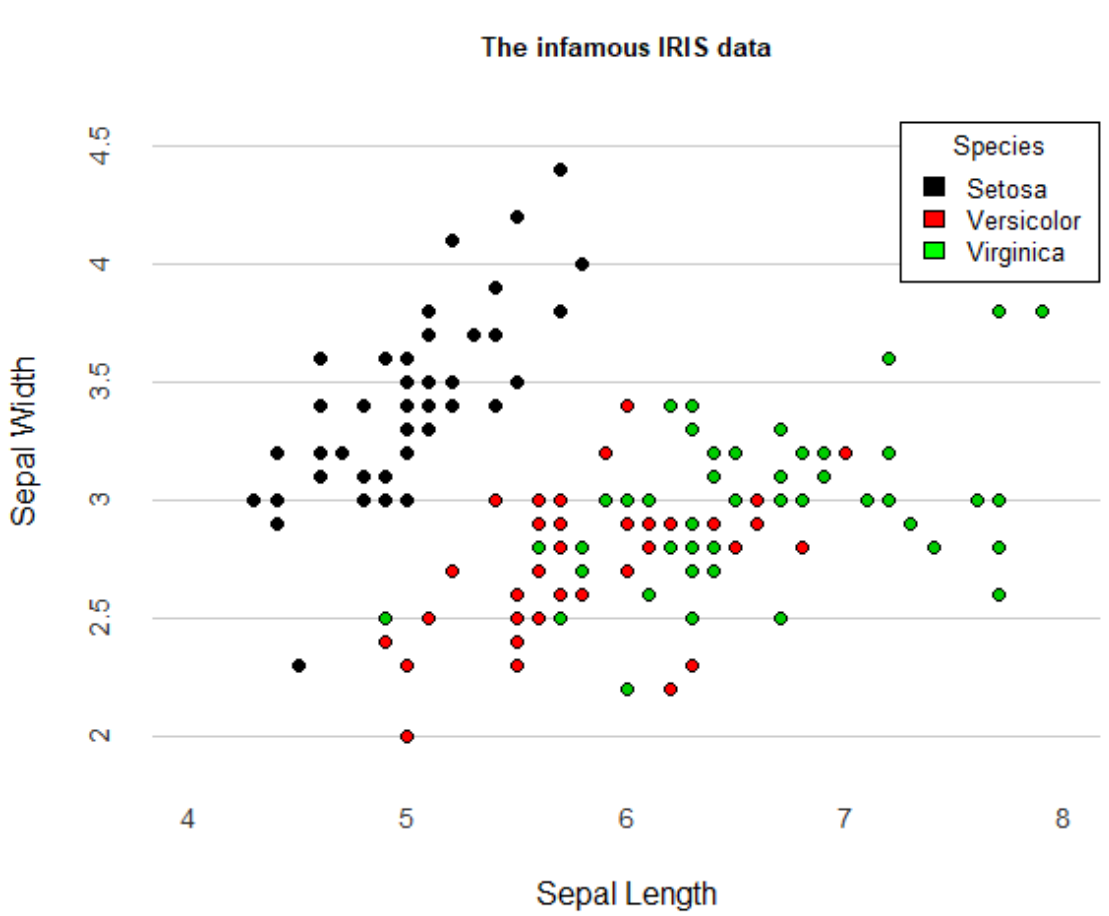


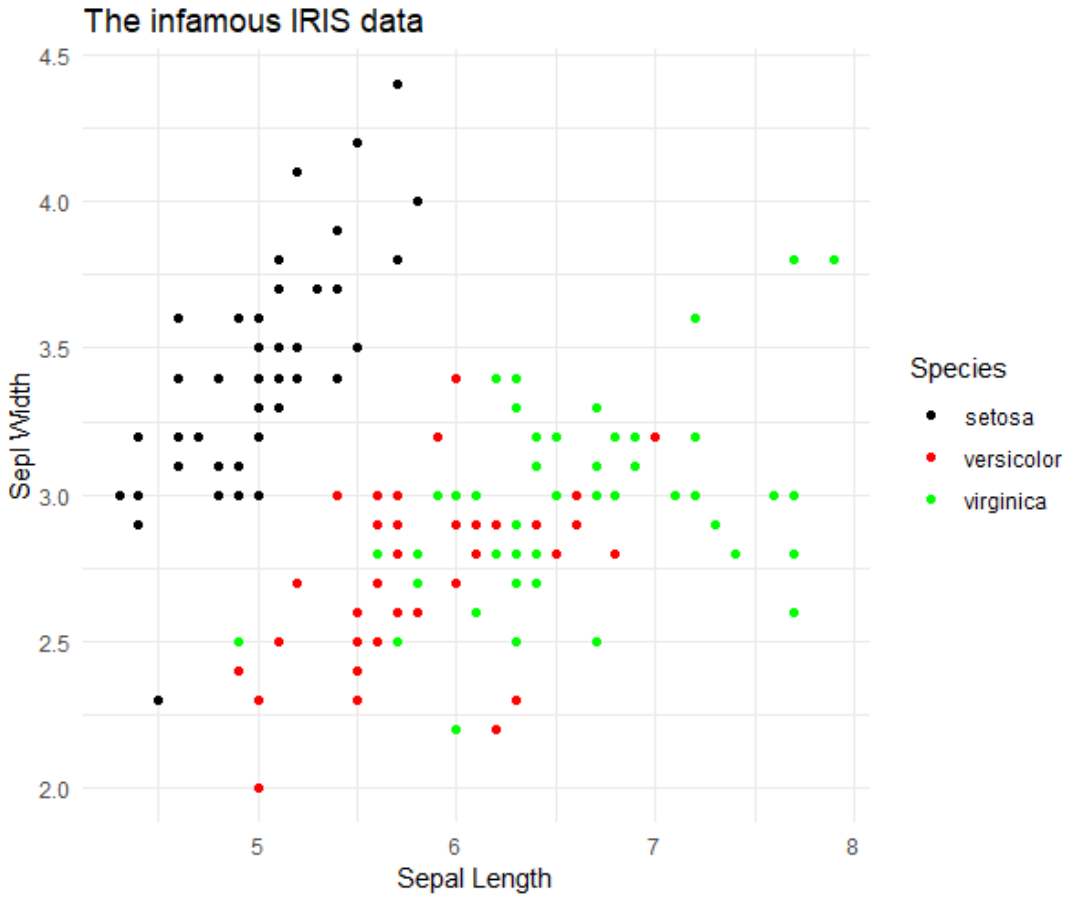
C: Plotting using base R graphics vs ggplot2

```r
# Here's an example using the graphics packages that comes with base R
plot(iris$Sepal.Length, iris$Sepal.Width,
     bg = iris$Species, # Fill colour
     pch = 21, # Shape: circles that can filed
     xlab = "Sepal Length", ylab = "Sepal Width", # Labels
     axes = FALSE, # Don't plot the axes
     frame.plot = FALSE, # Remove the frame
     xlim = c(4, 8), ylim = c(2, 4.5), # Limits
     panel.first = abline(h = seq(2, 4.5, 0.5), col = "grey80"))

at = pretty(iris$Sepal.Length)
mtext(side = 1, text = at, at = at,
      col = "grey20", line = 1, cex = 0.9)

at = pretty(iris$Sepal.Width)
mtext(side = 2, text = at, at = at, col = "grey20", line = 1, cex = 0.9)

legend("topright", legend = c("Setosa", "Versicolor", "Virginica"),
       title = "Species", fill=c("black", "red", "green"), cex=0.8)

title("The infamous IRIS data",
      cex.main = 0.8, font.main = 2, col.main = "black")

```

```r
# Now lets view the ggplot2 version
# library(tidyverse)
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, colour = Species)) +
  geom_point() +
  scale_colour_manual(values = c("black", "red", "green")) +
  theme_minimal() +
  labs(title = "The infamous IRIS data",
       x = "Sepal Length",
       y = "Sepl Width")

```

C: Plotting using base R graphics vs ggplot2





- Open the script 06_am_graphics_example.R to see for yourself

C: Whistle-stop tour of ggplot2

Main features:
1. The data
2. The geoms
3. The mappings (x, y, colour, shape etc.)
4. Legends
5. Labels
6. Themes

and many many more

- Open the script 07_am_ggplot2.R

- Open the script 08_am_practise_plots.R