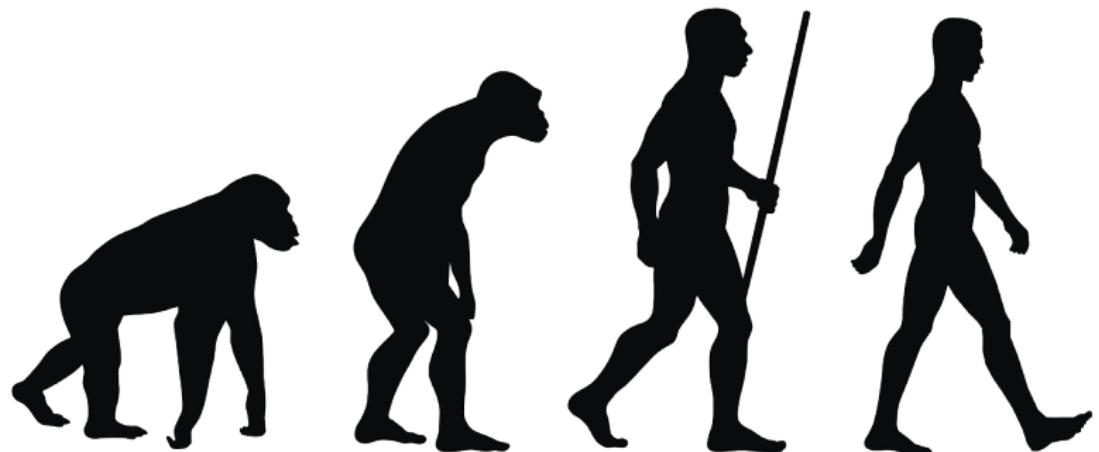


# R4H2O: R for Water Professionals

Dr Peter Prevos

# My Data Science Evolution



# Day 1 Program

- ▶ Data science principles
- ▶ Basics of R and Tidyverse
- ▶ Water quality case study
- ▶ Lunch
- ▶ Data visualisation
- ▶ Data products



R for Water Professionals workshop  
(Melbourne, 2019).

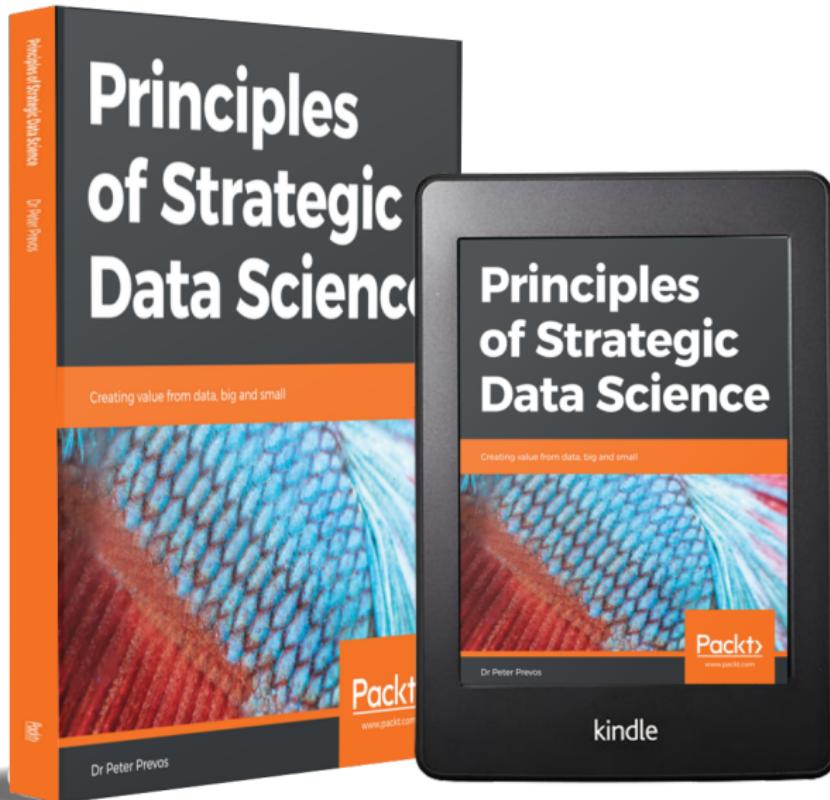
# Resources



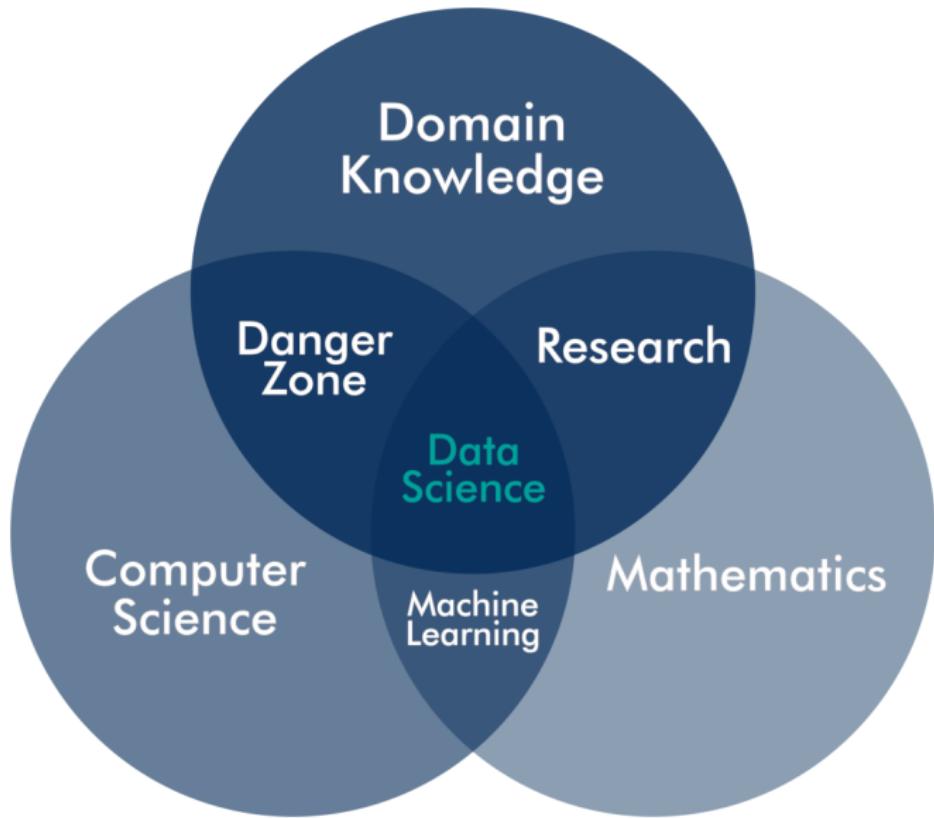
PETER PREVOS

Register to get access to the on-line syllabus:  
<https://leanpub.com/c/R4H2O/c/waterra>

# Principles of Data Science

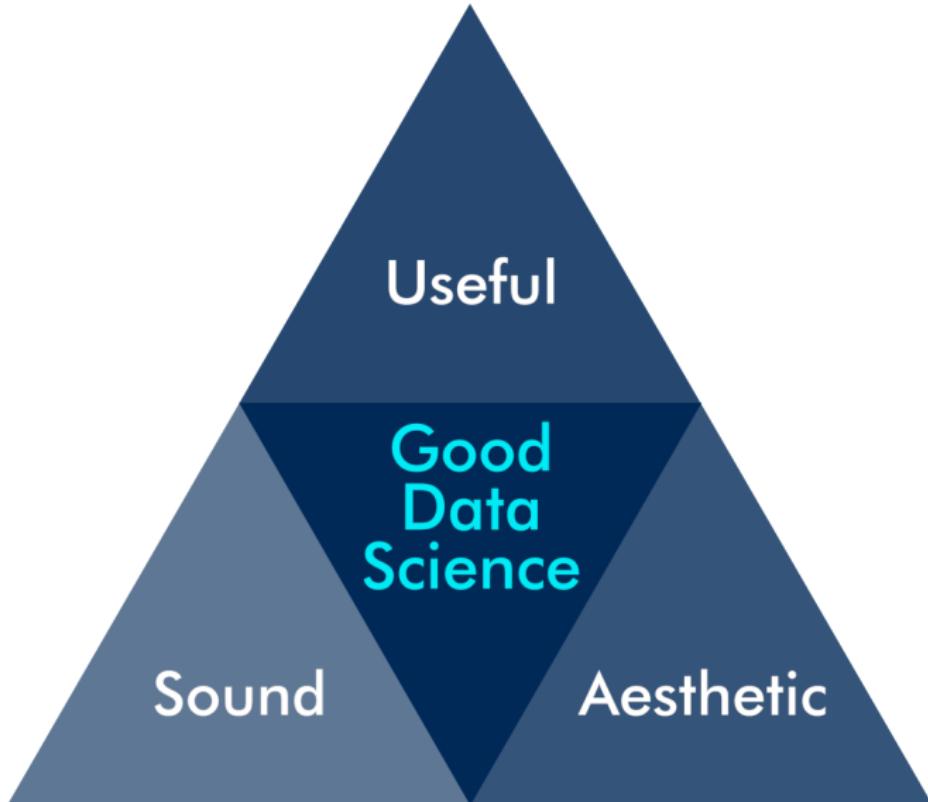


# What is Data Science?



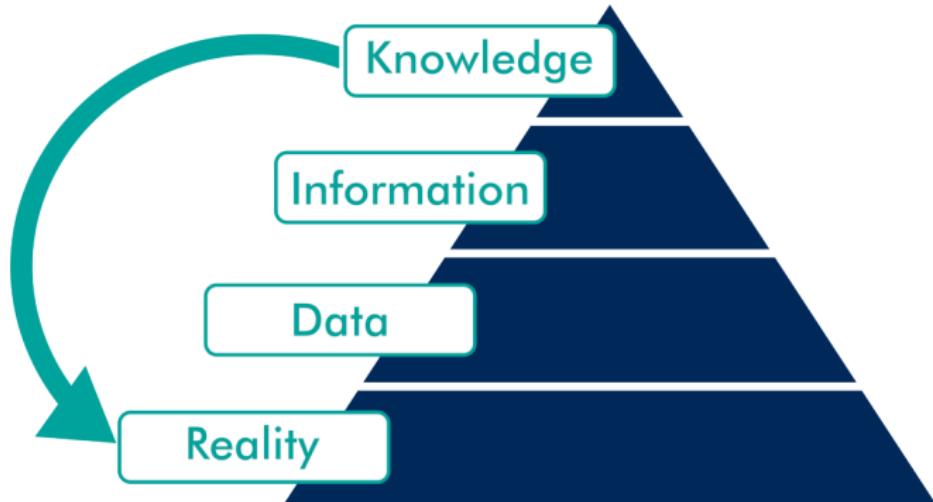
The Conway Venn Diagram (Drew Conway, 2013).

# What is good data science?



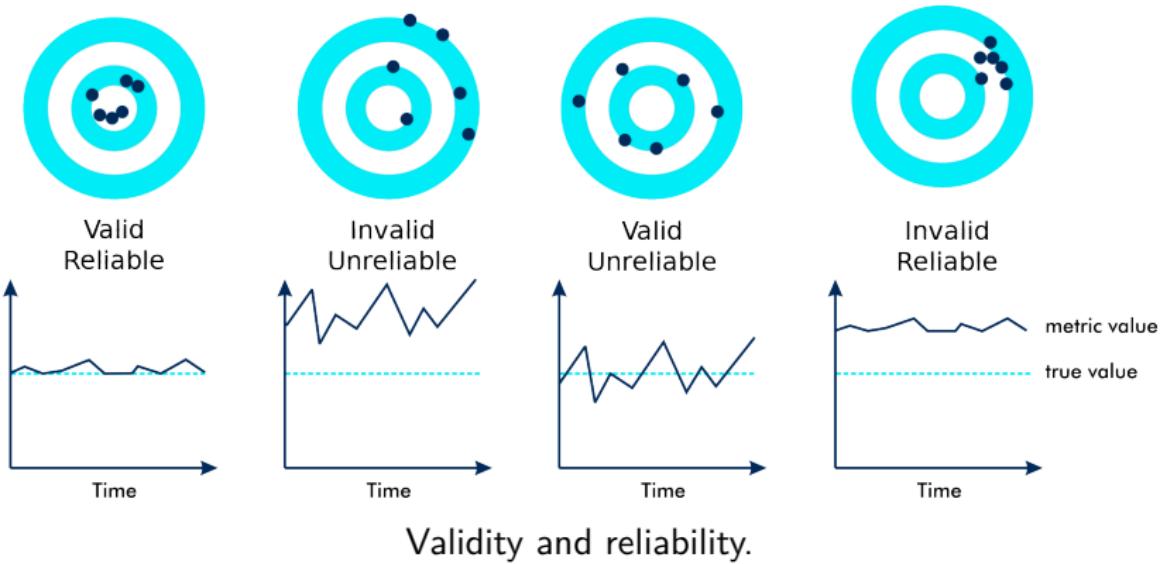
The Vitruvian triangle of good data science.

# What is useful data science?

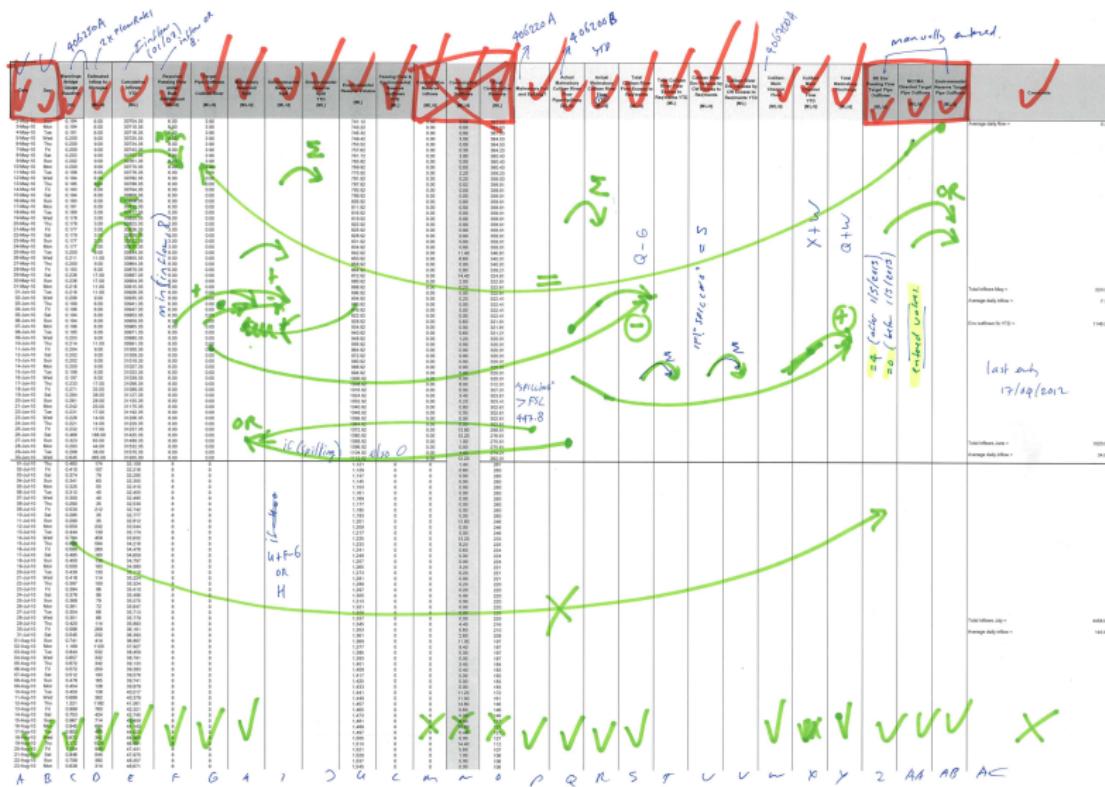


Modified version of the DIKW model.

# What is sound data science?



## What is sound data science?



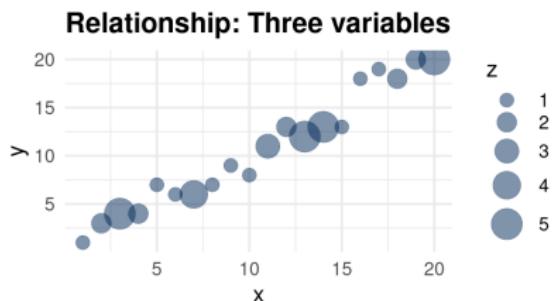
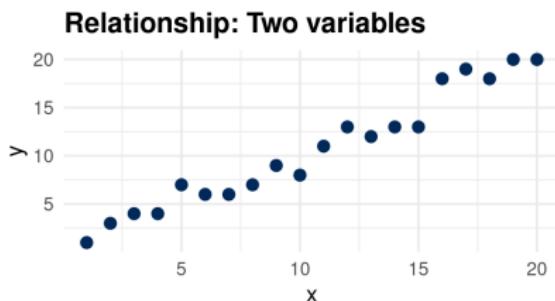
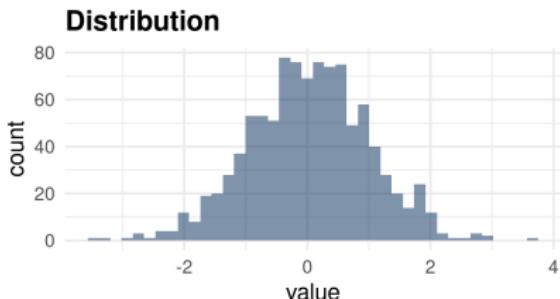
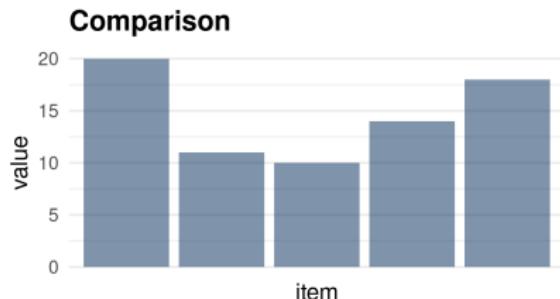
## Reverse-engineering a spreadsheet.

# What is sound data science?

Reproducible code:

```
reserve %>%
  select(Date, River_Flow, Natural_Flow, ERV) %>%
  mutate(Date = as.Date(Date, format = "%d %m %Y")) %>%
  gather(Source, Value, -Date) %>%
  mutate(type = factor(Source == "ERV"),
         type = fct_recode(type, Flow = "FALSE",
                           Volume = "TRUE")) %>%
  ggplot(aes(Date, Value, col = Source)) +
  geom_line() +
  facet_grid(type~., scales = "free_y")
```

# What is aesthetic data science?



Data visualisation is about telling stories.

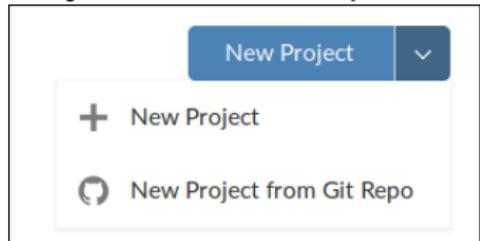
# Configure R Studio

## Desktop

- ▶ Install R and RStudio
- ▶ Download and unzip materials from <https://github.com/pprevos/r4h2o>
- ▶ *File > Open Project*
- ▶ Open the `r4h2o.Rproj` file in the downloaded folder

## Cloud

- ▶ Sign-up at: [rstudio.cloud](https://rstudio.cloud)
- ▶ *New Project > New Project from Git Repo*



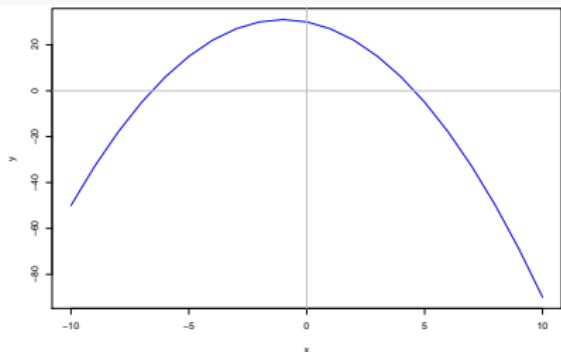
- ▶ Enter GitHub URL: <https://github.com/pprevos/r4h2o>

# Console exercise

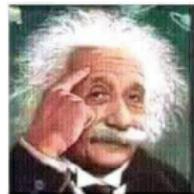
1. Enter sample code into the console (see syllabus for examples)
2. Observe the output in the console
3. Observe the environment
4. Use ↑↓ to scroll history
5. Use TAB for completion
6. Play with variations

```
x <- -10:10
y <- -x^2 - 2 * x + 30

plot(x, y, type = "l",
      col = "blue")
abline(h = 0, col = "grey")
abline(v = 0, col = "grey")
```



# R is Meme-Proof



Only for  
genius ??

$$3-3\times 6+2=??$$

VIRAL MATH PROBLEM

$$6 \div 2(1+2) =$$

Aritmetic memes.

# Finding Help

- ▶ Built-in *help()* function
- ▶ Cheat sheets (RStudio and Tidyverse websites)
- ▶ Twitter #rstats
- ▶ Reddit rstats, rlanguage
- ▶ stackoverflow.com
- ▶ Google the problem

MathFun (base) R Documentation

## Miscellaneous Mathematical Functions

**Description**

`abs(x)` computes the absolute value of  $x$ , `sqrt(x)` computes the (principal) square root of  $x$ ,  $\sqrt{x}$ .

The naming follows the standard for computer languages such as C or Fortran.

**Usage**

```
abs(x)
sqrt(x)
```

**Arguments**

`x` a numeric or [complex](#) vector or array.

**Details**

These are [internal generic primitive](#) functions: methods can be defined for them individually or via the [Math](#) group generic. For complex arguments (and the default method),  $z$ , `abs(z) == Mod(z)` and `sqrt(z) == z^0.5`. `abs(x)` returns an [integer](#) vector when `x` is [integer](#) or [logical](#).

**S4 methods**

Both are S4 generic and members of the [Math](#) group generic.

**References**

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

**See Also**

[Arithmetic](#) for simple, [log](#) for logarithmic, [sin](#) for trigonometric, and [Special](#) for special mathematical functions.

['plotmath'](#) for the use of `sqrt` in plot annotation.

**Examples**

```
require(stats) # for spline
require(graphics)
xx <- -9:9
plot(xx, sqrt(abs(xx)), col = "red")
lines(spline(xx, sqrt(abs(xx)), n=101), col = "pink")
```

screenshot of help window

# Exercise: Calculate Channel Flows

Determine the flow in a channel.  
Go to exercise 1 and answer the  
questions.

$$q = \frac{2}{3} C_d \sqrt{2g} b h^{3/2}$$

- ▶  $q$ : Flow [ $m^3/s$ ].
- ▶  $C_d \approx 0.6$ : Constant.
- ▶  $g = 9.81 m/s^2$
- ▶  $b$ : Width of the weir [m]
- ▶  $h$ : Water depth over weir [m]



Photo: Coliban Water

## Scripts versus Console

- ▶ Store all code in a text file with .R extension
- ▶ Output in console, plots and viewer
- ▶ Use comments (start with #) to explain the code
- ▶ *File > New File > R Script*
- ▶ Open the channel\_flow.R script in introduction folder.
- ▶ Reverse-Engineer the code

# Reproducible Code

- ▶ Give meaningful names
- ▶ Use a consistent method,  
e.g.:
  - ▶ Only lower case:  
`channelflow`
  - ▶ Underscore for spaces:  
`channel_flow`
  - ▶ Camel case:  
`ChannelFlow`
- ▶ Use comments to explain  
the process
- ▶ Add links to documentation
- ▶ Automate as much as  
possible

## Case Study 1: Water Quality

Safe Drinking Water Regulations  
2015:

*"the 95th percentile of results for samples in any 12 months must be less than or equal to 5.0 Nephelometric Turbidity Units."*

Guidance document:  
*"The method recommended by the department is described as the Weibull method and is the method adopted by the National Institute of Standards and Technology (NIST)."*

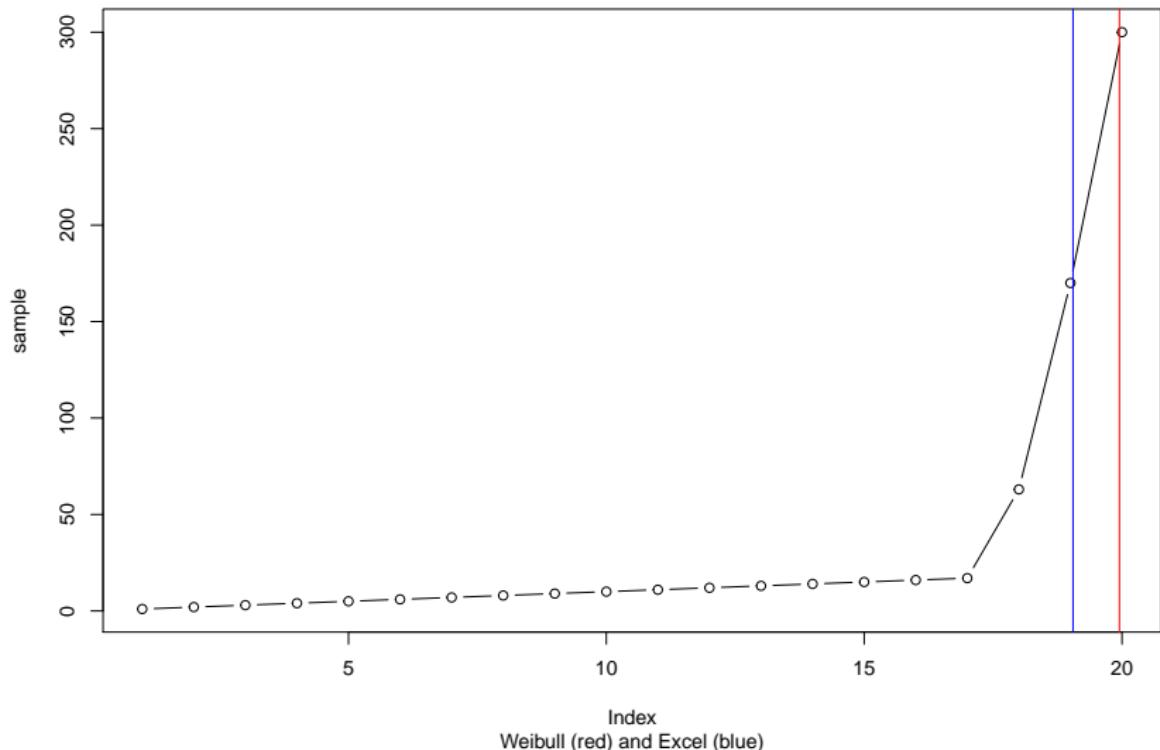
## Percentiles

1. The data are placed in ascending order:  $y_1, y_2, \dots, y_n$ .
2. Calculate the rank of the required percentile
  - ▶ Weibull:  $r = p(n + 1)$
  - ▶ Excel:  $r = 1 + p(n - 1)$
3. Interpolate between adjacent numbers:

$$X_p = (1 - r_{frac}) Y_{r_{int}} + r_{frac} Y_{r_{int+1}}$$

# Percentile example

Percentile example



Explore the percentiles.R script in the casestudy1 folder.

# The Tidyverse

An opinionated collection of R packages optimised for data science. All packages share an underlying design philosophy, grammar, and data structures.

```
install.packages("tidyverse")
library(tidyverse)
```

Load the casestudy1.R script in the casestudy1 folder.



# Data frames or 'tibbles'

- ▶ Rectangular data
- ▶ Variables in columns
- ▶ Observations in rows
- ▶ One variable in R environment
- ▶ Tidy data
- ▶ Read data:

```
dataframe <- read_csv(filename)
```

group	var	val
1	B	12
2	B	34
1	C	43
2	C	76
1	D	5
2	D	12

Data frame structure.

## Filter a data frame

```
filter(df, var == "B")
```

group	var	val
1	B	12
2	B	34
1	C	43
2	C	76
1	D	5
2	D	12

group	var	val
1	B	12
2	B	34

# Grouping

```
group_by(df, var)
```

group	var	val
1	B	12
2	B	34
1	C	43
2	C	76
1	D	5
2	D	12

group	var	val
1	B	12
2	B	34

group	var	val
1	C	43
2	C	76

group	var	val
1	D	5
2	D	12

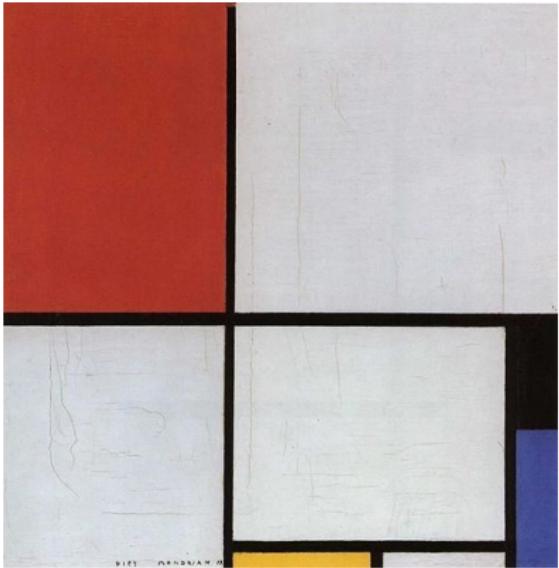
## Exercise

- ▶ Load the CSV file for the Gormsey system in the casestudy1 folder.
- ▶ Explore the data.
- ▶ Answer the questions in Exercise 2 in your syllabus.
- ▶ You can cheat by opening the gormsey\_quiz.R script.

# Data Visualisation



Jackson Pollock, *Blue Poles* (1973).

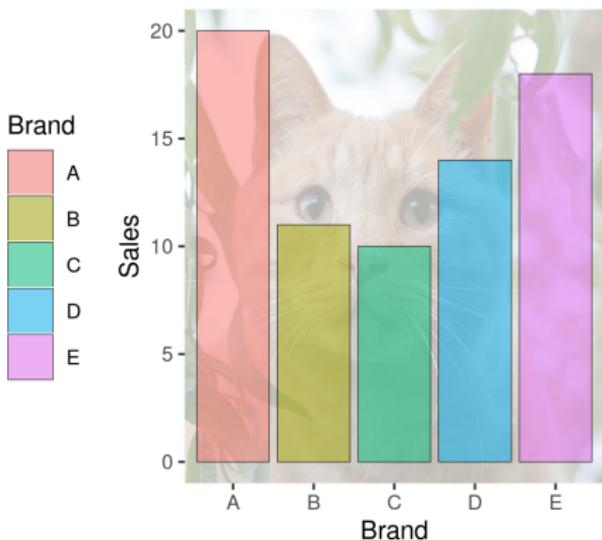


Piet Mondrian, *Composition in Red, Yellow and Blue* (1928)

# Data-to-Pixel Ratio

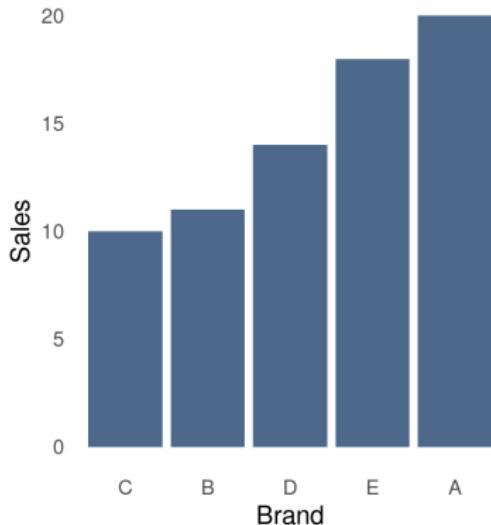
**Cat food sales**

Low Data-Pixel Ratio



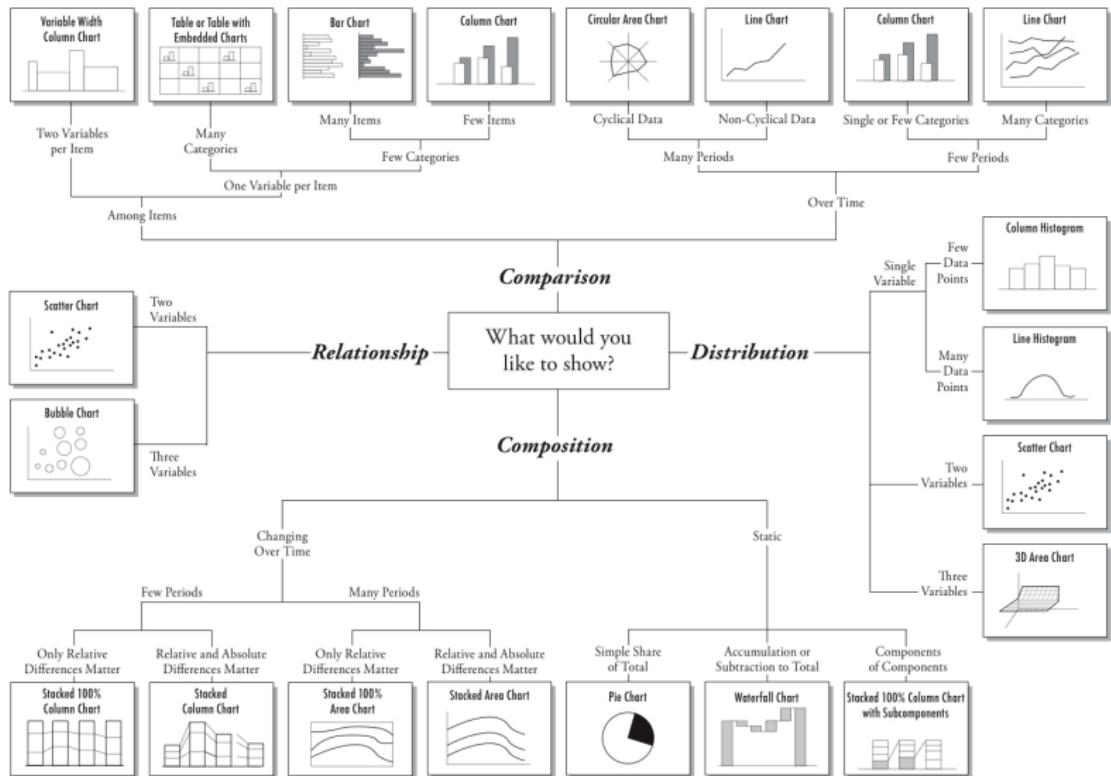
**Cat food sales**

High Data-Pixel Ratio



Maximise the data to pixel ratio for aesthetic visualisations.

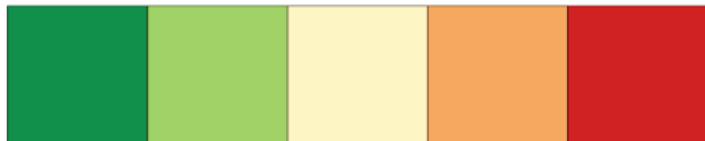
# Chart Chooser



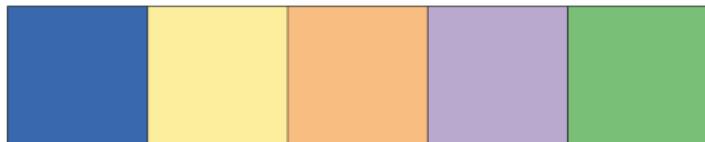
## Use Colours Sparingly



Sequential



Diverging



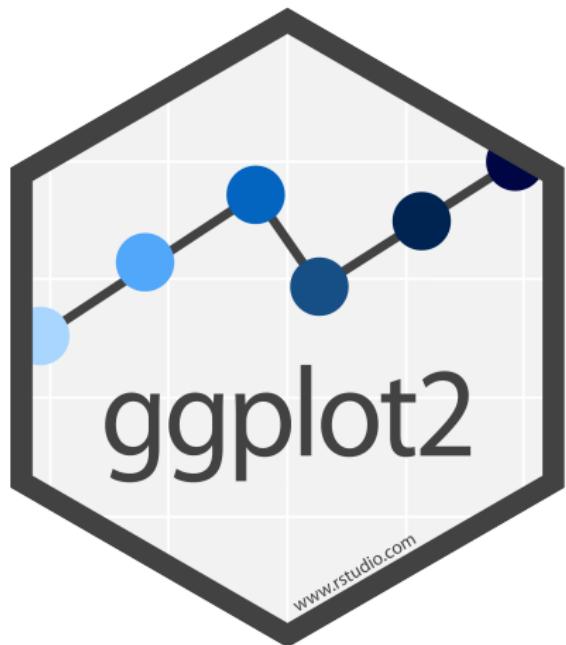
Qualitative

Types of colour pallets. Go to [colorbrewer2.org](http://colorbrewer2.org) for details.

# ggplot2

- ▶ System for creating graphics, based on *The Grammar of Graphics*.
- ▶ Go to [ggplot2.tidyverse.org](https://ggplot2.tidyverse.org) for documentation.
- ▶ Included in the Tidyverse.  
You can call it separately with:

```
library(ggplot2)
```



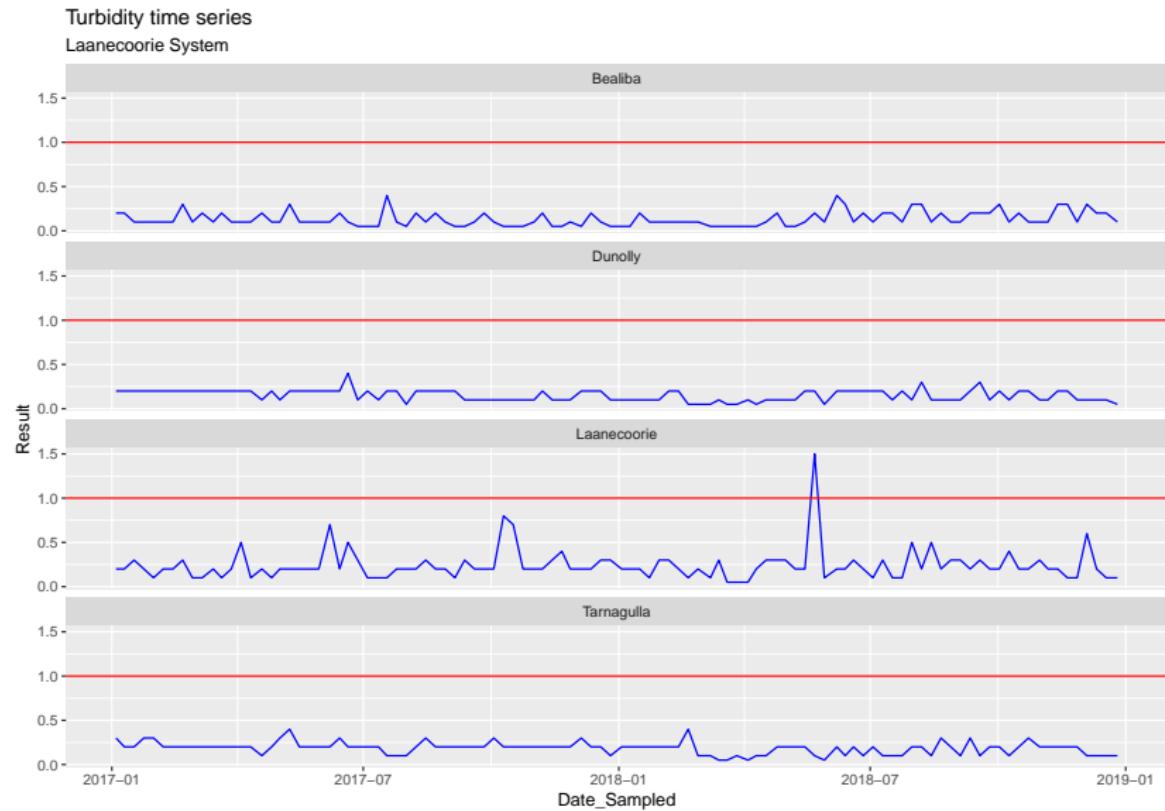
# Grammar of Graphics

*Theme*  
*Coordinates*  
*Statistics*  
*Facets*  
*Geometries*  
*Aesthetics*  
*Data*



Leland Wilkinson, *Grammar of Graphics* (2005).

# ggplot2 Example



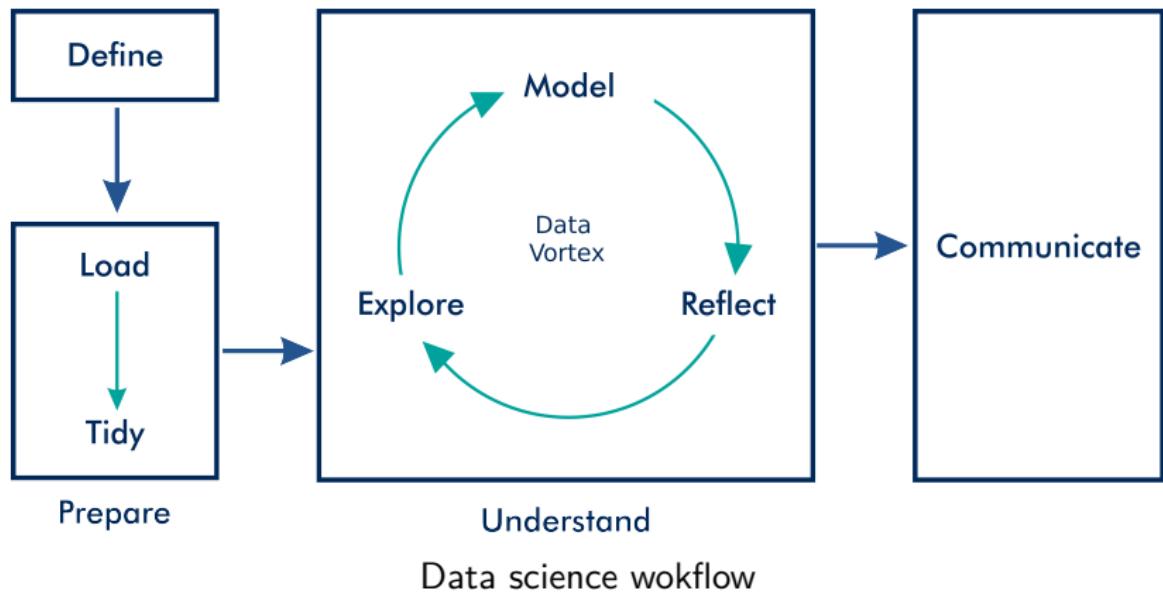
# Visualisation Exercise

Use your knowledge of the Gormsey data to create two visual data stories. Use the following four steps:

1. Explore the data and define the story you want to tell.
2. Decide on the best way to visualise the story.
3. Develop the basic visualisation.
4. Select a theme and annotate the graph.



# Data Science Workflow



# RMarkdown

Literate programming:

- ▶ Combine prose with code
- ▶ Link the code to dynamic data
- ▶ Generate shareable output from code

Data products:

- ▶ Reports
- ▶ Web sites
- ▶ Presentations
- ▶ Applications (dashboards)

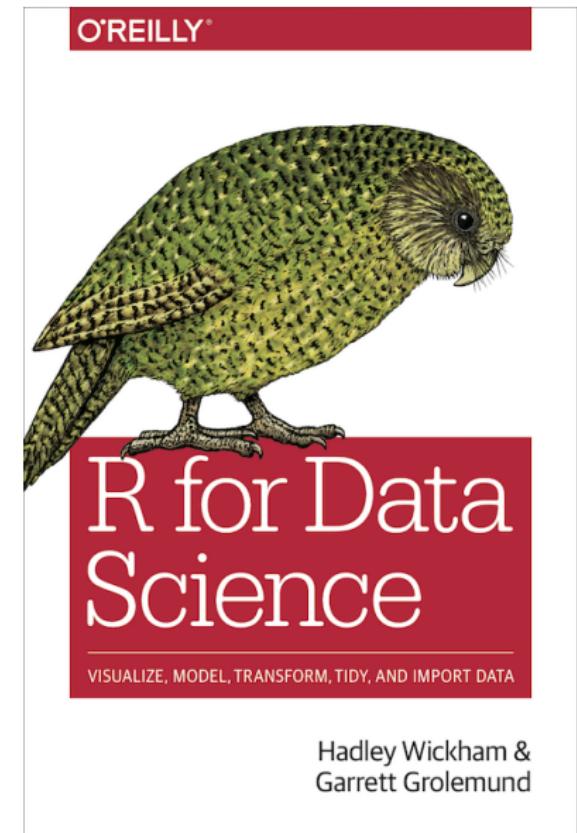
## Mini Hackathon

To close this day, we will do a mini hackathon.

1. Create a script that results in a PowerPoint presentation about the Gormsey data.
2. Pick a story you like to tell about this data.
3. Create a RMarkdown script that results in a Powerpoint presentation.
4. Add an introduction.
5. Explore the data.
6. Share the story.

## Further Study

- ▶ Tidyverse website
- ▶ RStudio website (webinars and cheat sheets)



[r4ds.had.co.nz](http://r4ds.had.co.nz)