

A Two Hour Tour of the Tidyverse



Aimee Gott

Senior Consultant

agott@mango-solutions.com

In This Workshop

- You will learn...
 - What the tidyverse is & why bother using it
 - What tools are available in the tidyverse
 - A brief overview of core functionality

Data for Today

- Funding of Olympic sports
- UK Sport World Class Performance Programme
- Data from Sydney (2000) to Rio de Janeiro (2016)

- Extracted from
<http://www.uk sport.gov.uk/our-work/investing-in-sport/historical-funding-figures>

Where to Get the Data

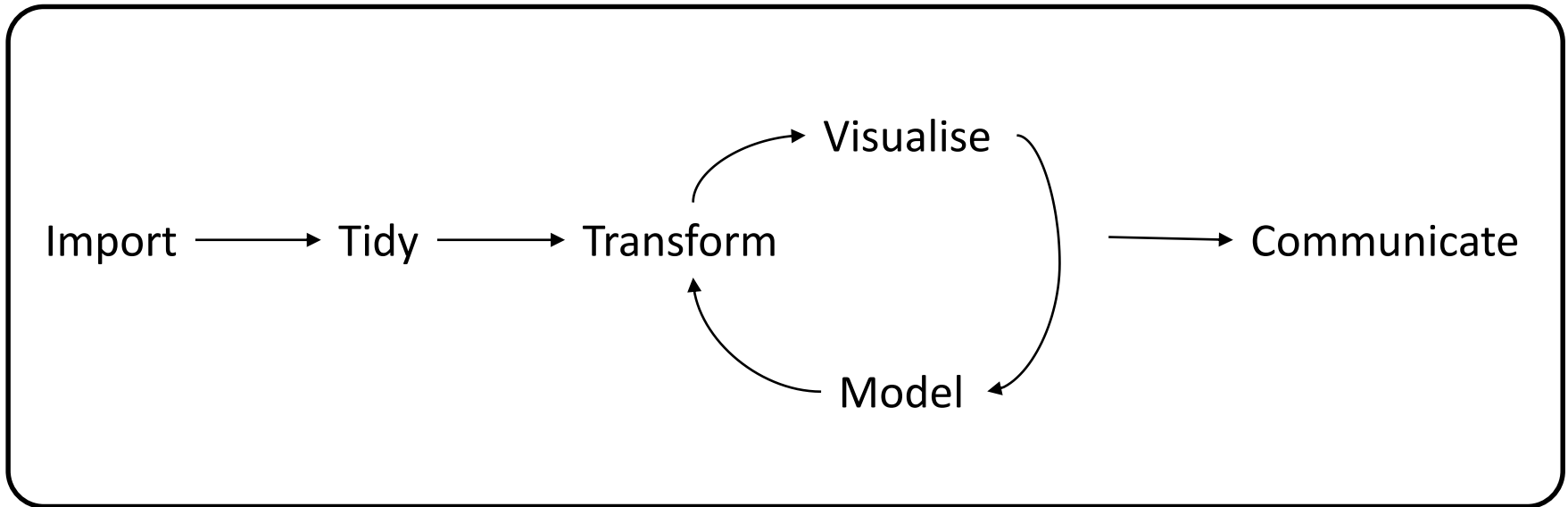
Files in multiple formats available on GitHub:

<https://github.com/MangoTheCat/Olympic-Funding-Data>

WHAT IS THE TIDYVERSE?

Unified Packages

- Set of unified packages covering the data analysis workflow
- Implementing a single philosophy
 - Makes it easy to learn and use!



Program

Useful Resources

RStudio Cheat Sheets

<https://www.rstudio.com/resources/cheatsheets/>

(My advice – print them all and keep them on your desk!)

Further Reading

R for Data Science (2017), *Hadley Wickham & Garrett Grolemund*, O'Reilly

Available online:

<http://r4ds.had.co.nz/>

WHAT'S IN THE TIDYVERSE?

What are the Packages?

- 18 Core Packages
 - Loads more dependencies
- 6 Packages loaded with Tidyverse
- All others installed and available

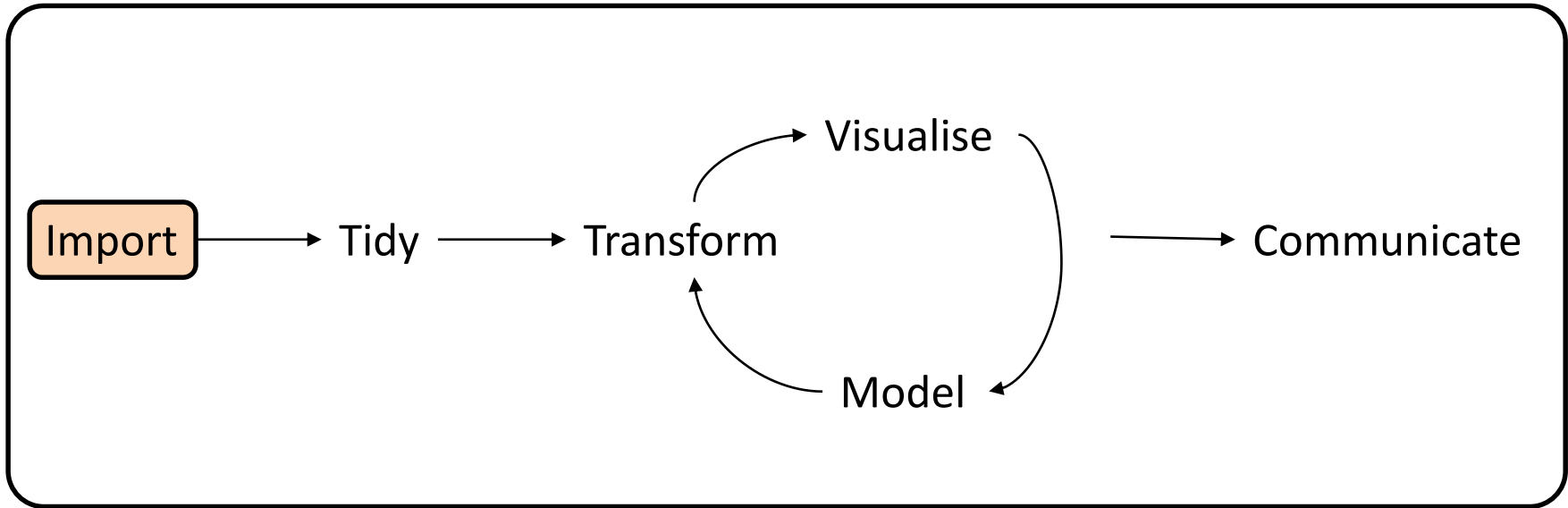
Packages Loaded By Default

- `ggplot2`
- `dplyr`
- `tidyr`
- `readr`
- `purrr`
- `tibble`

Other Installed Packages

- `hms`
- `stringr`
- `lubridate`
- `forcats`
- `haven`
- `httr`
- `jsonlite`
- `readxl`
- `rvest`
- `xml2`
- `modelr`
- `broom`

A DIVE INTO THE PACKAGES



Program

Import

- readr
- haven
- httr
- jsonlite
- readxl
- rvest
- xml2

Reading Web Data: rvest

```
read_html("www.myWebPage.co.uk")
```

- Includes lots of functions for extracting specific elements of a web page

Read Olympic Data

```
funding <-  
  read_html("http://www.ukSPORT.gov.uk/our  
            -work/investing-in-sport/  
            historical-funding-figures")  
  
funding <- html_table(funding, header = TRUE)
```

Reading Tabular Data: readr

```
read_csv("myFile.csv", na = c(" ", "NA"))
```

Other variants include:

- `read_tsv`
- `read_fwf`
- `read_delim`

Reading Summer Olympics Data

```
summer <-  
read_csv("SummerOlympicsFunding.csv",  
         na = c("n/a", "n/a**"),  
         col_types = cols(  
             .default = col_number(),  
             Sport = col_character()  
         )
```

Reading MS Excel Data: readxl

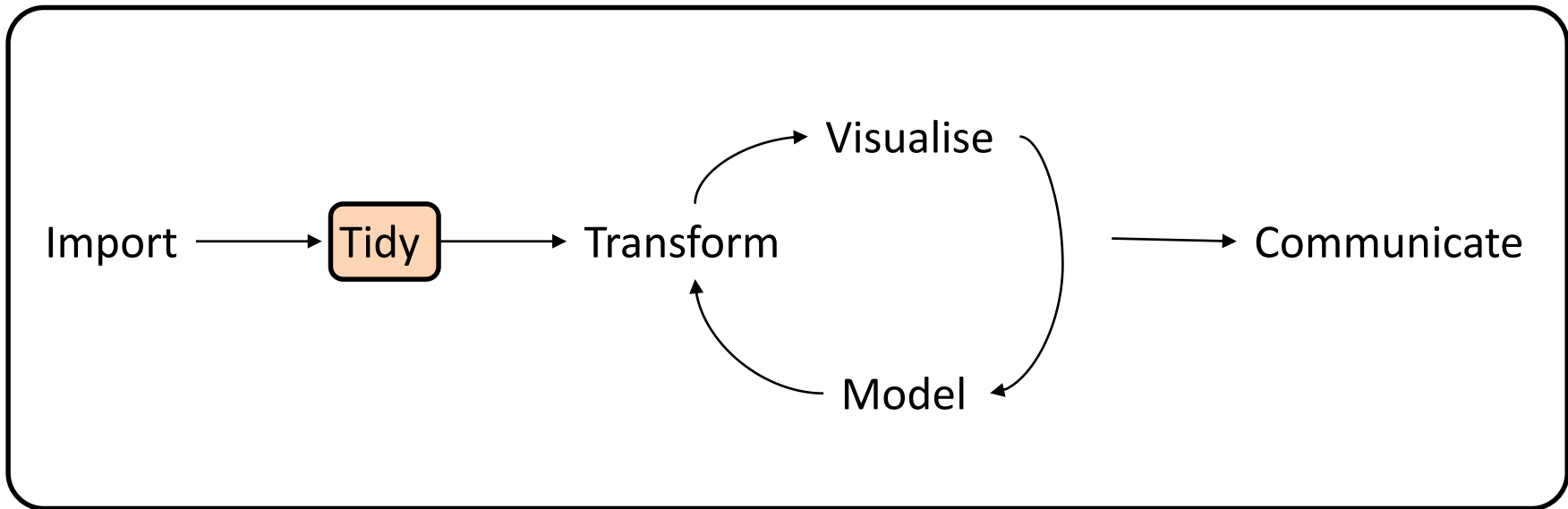
```
read_excel("myExcelFile.xlsx",  
           sheet = "Sheet 1")
```

Reading Proprietary Data: haven

```
read_sas("mySASData.sas7bdat")
```

Other formats include:

- SPSS Data
- Stata Data



Program

Tidy

- `tibble`
- `tidyr`

What is tibble?

- A tibble is an update to the data frame
- Generally works as a data frame but with extra features
 - Doesn't change column names
 - Doesn't convert strings to factors
 - Doesn't allow partial matching of names
 - Doesn't create row names

How Do You Create a tibble?

- Using the import functions in the tidyverse
- With the tibble or tribble functions

What Does the tibble Package Do?

- Provides utility functions for working with tibble structure
- Generally don't need to directly use the package much

Functions in tibble

- `glimpse` – useful overview of data
- `tibble/tribble` – creation of tibbles
- `add_row/add_column` – helpful functions for adding elements to an existing tibble
- There are also a number of functions for converting rownames if needed

Creating A Reference Tibble

```
years <- tribble(  
  ~Location, ~Year, ~Month, ~Day,  
  "Sydney", 2000, 9, 15,  
  "Athens", 2004, 8, 13,  
  "Beijing", 2008, 8, 8,  
  "London", 2012, 7, 27,  
  "Rio de Janeiro", 2016, 8, 5  
)
```

Tidy Data

- The tidyverse is designed to work with tidy data
- A single structure that is common to all of the packages
 - Makes it easy to move from manipulation to visualisation to modelling without changing the data

What is Tidy Data?

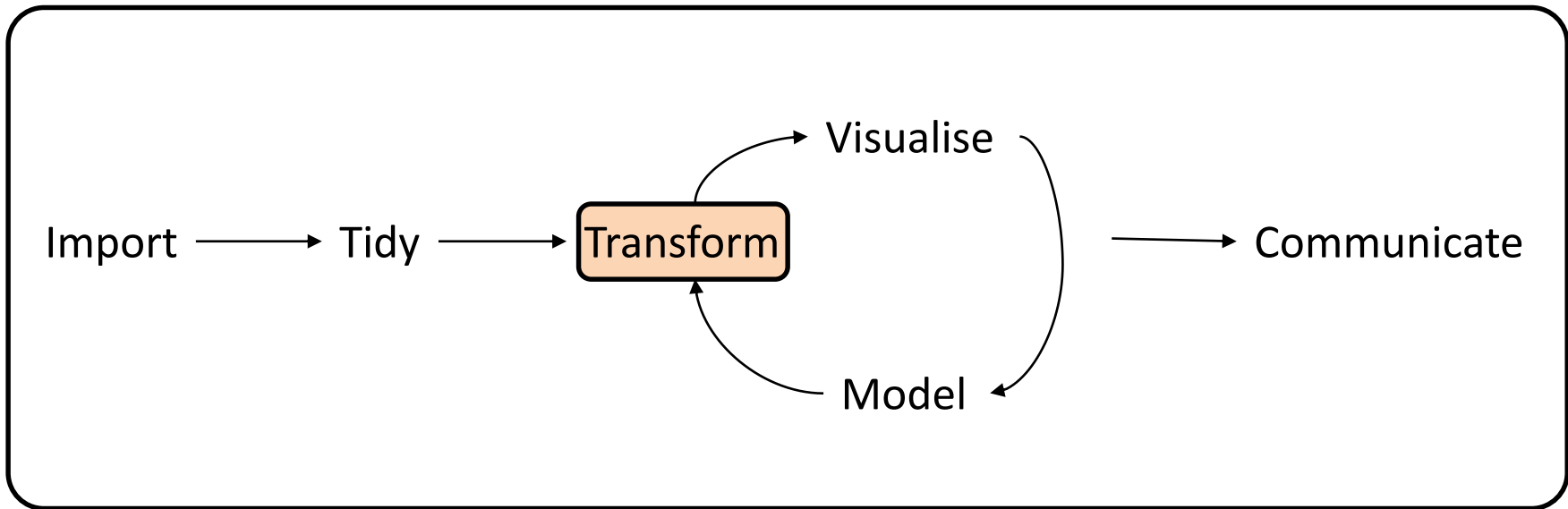
- Each variable has its own column
- Each observation has its own row
- Each value has its own cell

Tidying the Olympic Data

```
summer <- gather(summer,  
                  Location,  
                  Funding,  
                  -Sport)
```


Working with Missing Values

```
summer <- replace_na(summer,  
                      replace = list(Funding = 0))
```



Program

Transform

- `dplyr`
- `forcats`
- `stringr`
- `hms`
- `lubridate`

Data Manipulation: dplyr

- Standard manipulations
 - Filter
 - Select
 - Arrange
 - Mutate
 - Summarise
- Perform actions by group (`group_by`)

Join Related Data Sets

- Merge/Join two data sets
 - Full
 - Left/Right
 - Inner
 - Anti
 - Semi

Joining Olympic Data

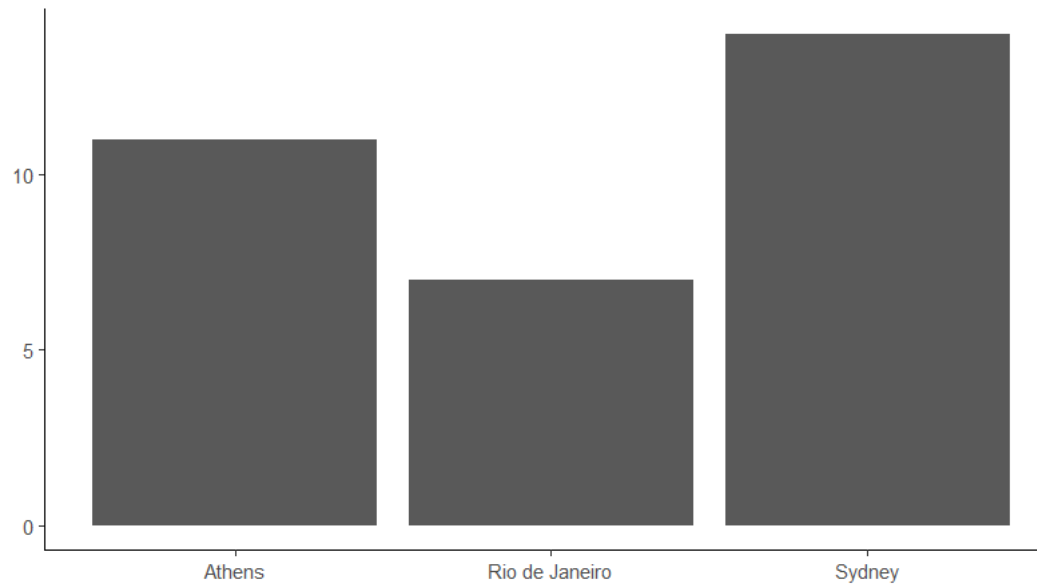
```
summer <- full_join(summer, years) %>%  
  filter(Sport != "Total")
```

Manipulating Factors: forcats

- Factors are a representation of categorical data
- Forcats allows us to easily manipulate factors
 - Change names
 - Group levels
 - Reorder levels

Without Manipulation

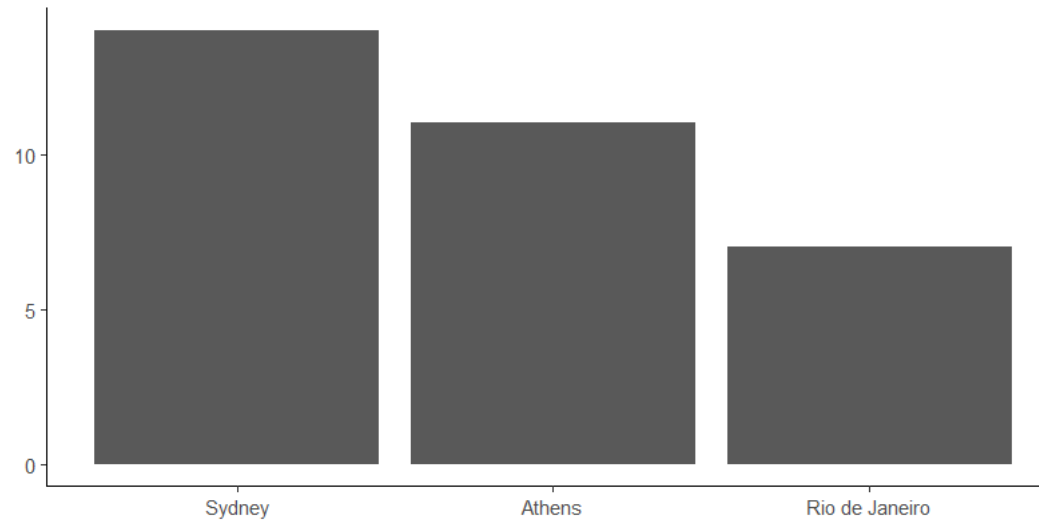
Changes in Number of Sports Not Provided UK Sport Funding
Funding Provided by UK Sport World Class Performance Programme



Data taken from uksport.gov.uk

With Manipulation

Changes in Number of Sports Not Provided UK Sport Funding
Funding Provided by UK Sport World Class Performance Programme



Data taken from uksport.gov.uk

Reordering Olympic Years

```
numberNoFund <- summer %>%  
  filter(Funding == 0)  
  count(Location) %>%  
  left_join(years) %>%  
  mutate(Location =  
           fct_reorder(Location, Year))
```

Manipulating Characters: stringr

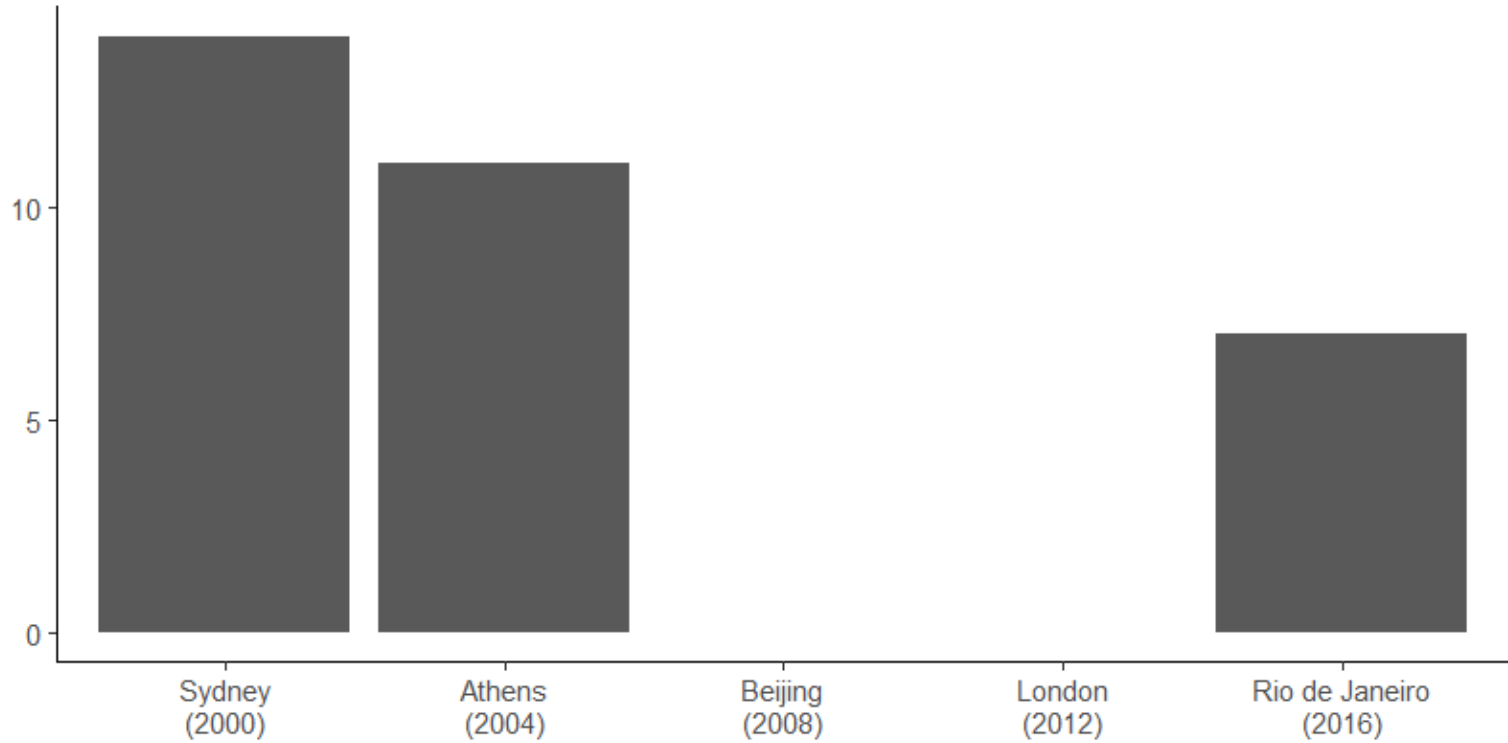
- The stringr package provides consistent functions for manipulating character strings
- Manipulations include:
 - Concatenation
 - Pattern search and replace
 - Subset with strings

Combining Location and Year

```
str_c(  
  magrittr::use_series(years, Location),  
  "\n",  
  magrittr::use_series(years, Year),  
  ") "  
)
```

Changes in Number of Sports Not Provided UK Sport Funding

Funding Provided by UK Sport World Class Performance Programme



Data taken from uksport.gov.uk

Exercise

- Import the winter Olympics data csv
- By performing the appropriate manipulations (using tidyr, dplyr and forcats)
 - Which sport has the most funding by Sochi?
 - Extract the data for just this sport
 - What is the difference between the maximum and minimum funding levels for this sport?
 - Can you create a simple graphic of this data?
 - Why is there some missing data for this sport?

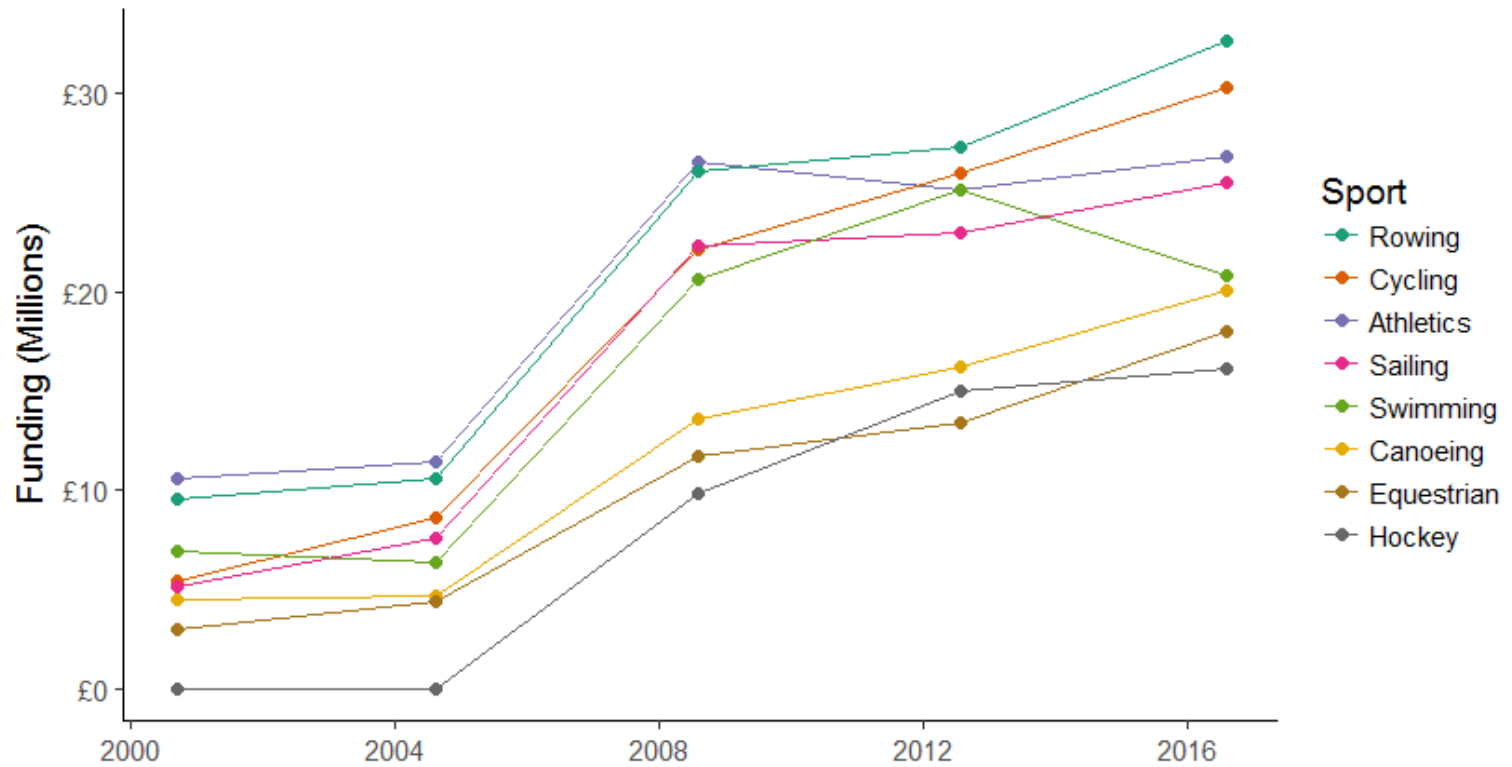
Manipulating Dates: lubridate

- Easy to use functions to convert characters to date format
 - ymd
 - mdy
 - dmy
- Manipulation functions to extract elements of dates
- Functions for accounting for addition of time periods to dates e.g. accounting for 30/31 day months

Creating Olympic Start Dates

```
summer <- mutate(summer,  
  Date = str_c(Year, Month, Day, sep = "-"),  
  Date = ymd(Date)  
)
```

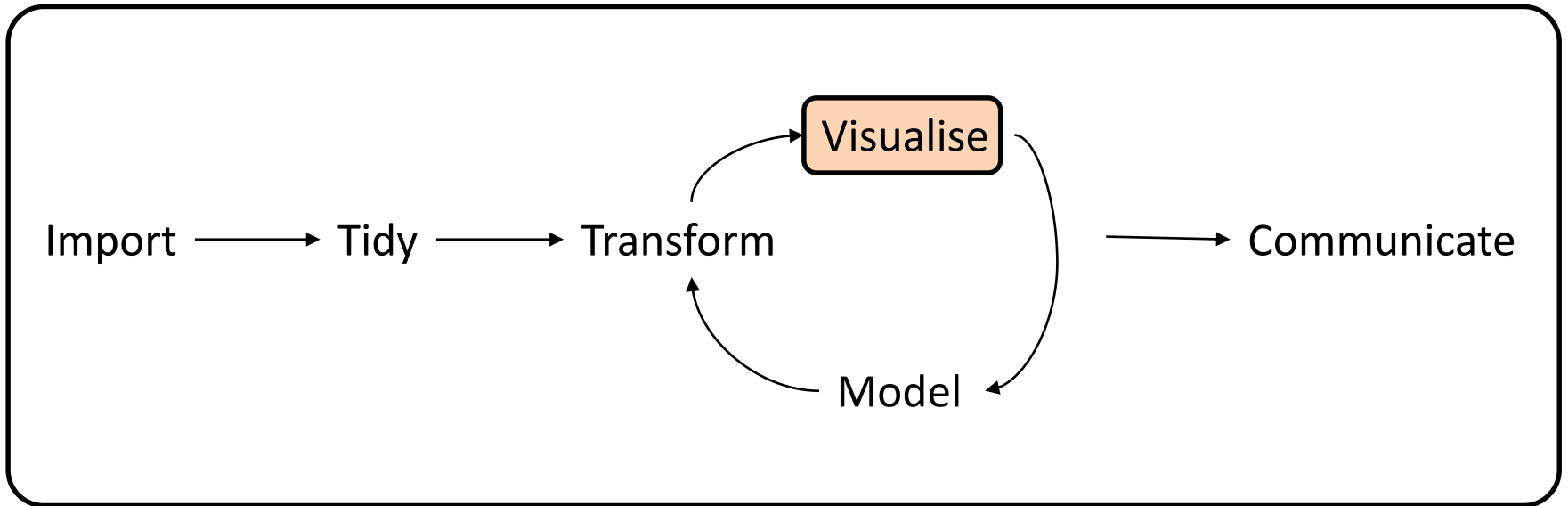

Overall Increasing Funding of Top Sports



Data taken from uk sport.gov.uk

Manipulating Times: HMS

- Allows working with just times
- Convert to just a time
 - hms



Program

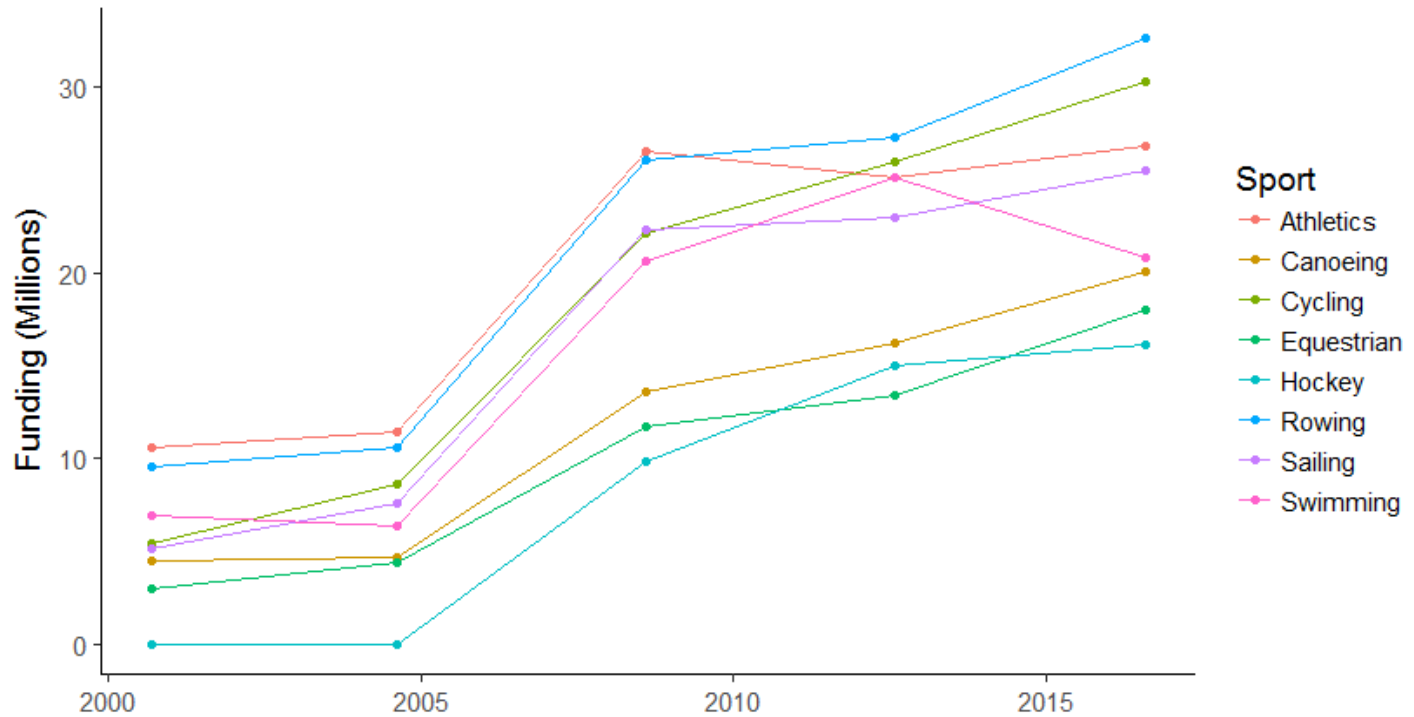
Visualise

- `ggplot2`

Creating Graphics: ggplot2

- A whole workshop on its own!
- Create quick plots with qplot
- Specify the type of plot using the geoms
- Add titles and labels with labs

Overall Increasing Funding of Top Sports



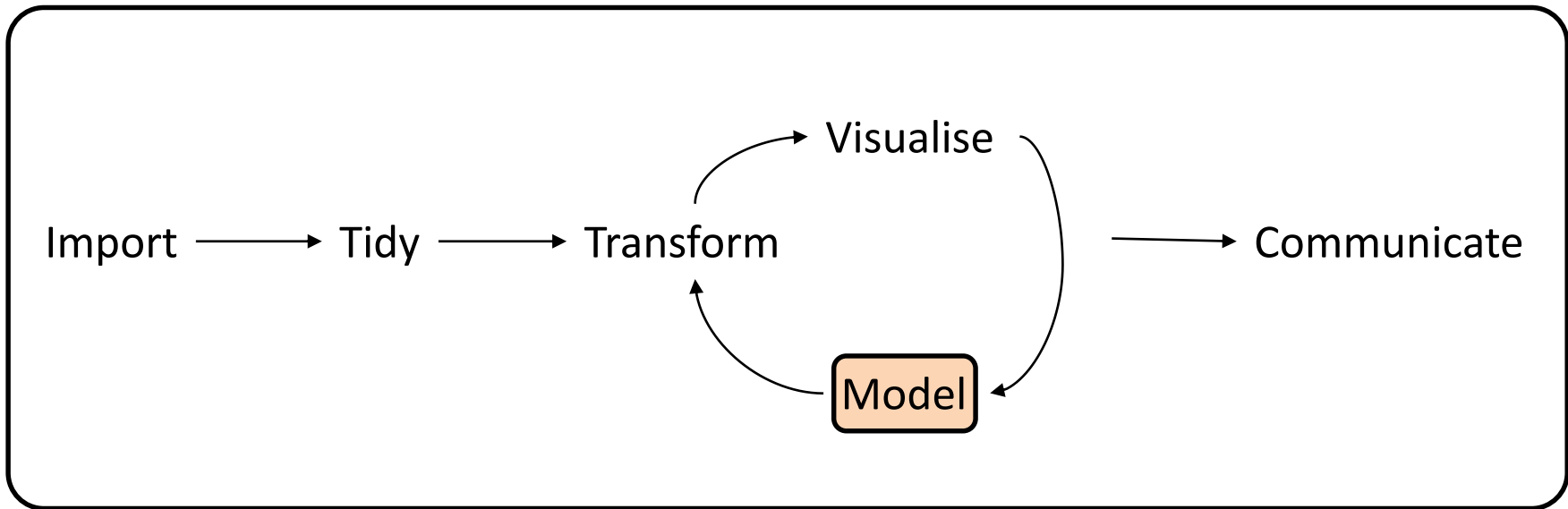
Data taken from uk sport.gov.uk

Graphic of Olympic Funding

```
qplot(data = fundingTime,  
      x = Date, y = Funding/1e6,  
      colour = Sport,  
      geom = c("line", "point")) +  
labs(title = "Increasing Funding ",  
      x = "", y = "Funding (Millions)",  
      colour = "Sport",  
      caption = "Data taken from ...")
```

Exercise

- Using the winter Olympics data create a graphic showing the change in funding over time for all of the winter Olympic sports



Program

Model

- `modelr`
- `broom`

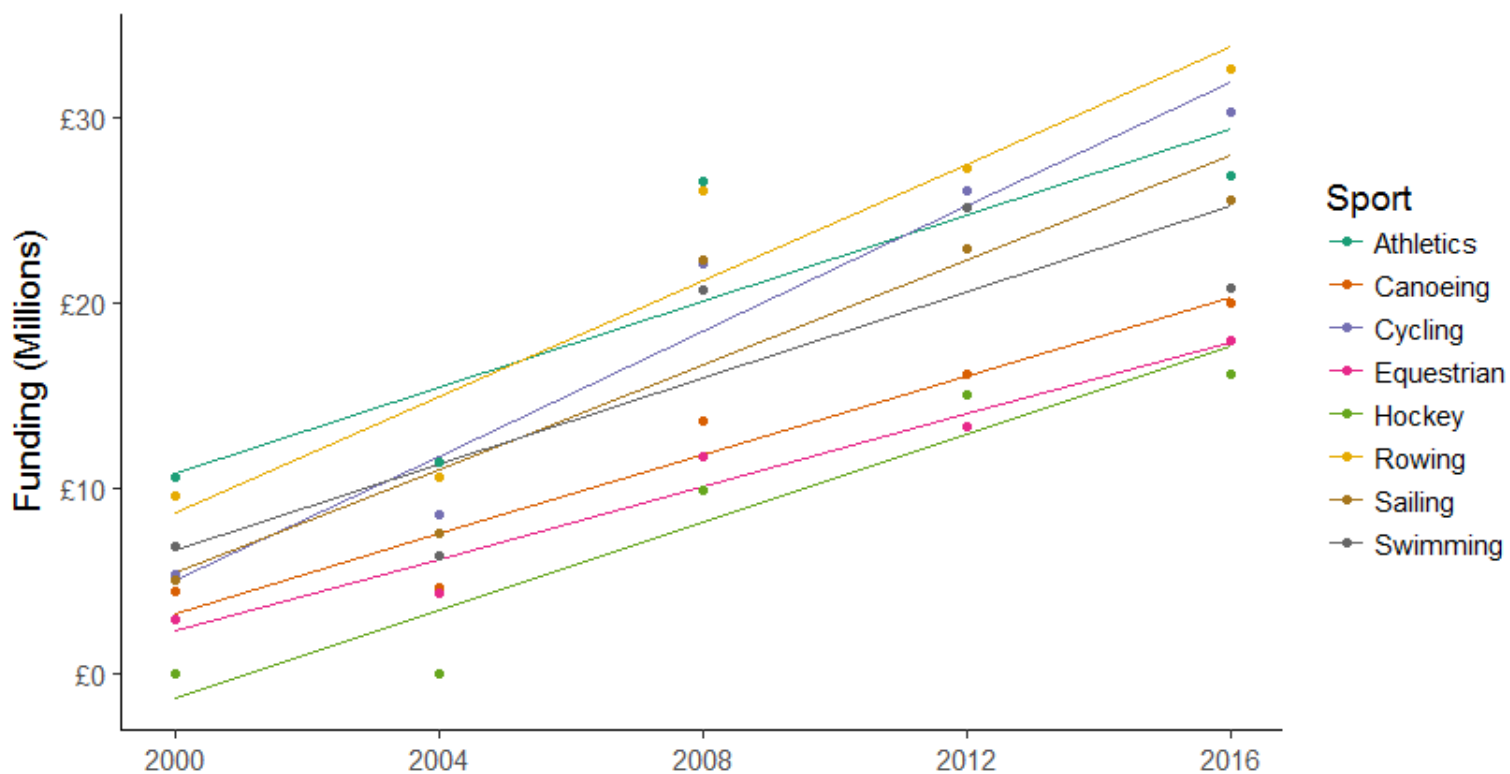
Fitting Models: modelr

- Functionality for bootstrapping/cross validation
- Model metrics – rsquare, rmse
- Extracting predictions and residuals

Predicting Olympic Funding

```
fundingModel <- lm(Funding ~ Sport*Year,  
                  data = fundingTime)  
  
modelGrid <- data_grid(fundingTime,  
                      Year, Sport)  
  
modelGrid <- modelGrid %>%  
             add_predictions(fundingModel)
```

Fitted Model for Top Funded Sports



Data taken from uksport.gov.uk

Investigating Model Residuals

```
fundingResid <- fundingTime %>%  
  add_residuals(fundingModel)
```

Residual Values Suggests Further Fitting Required



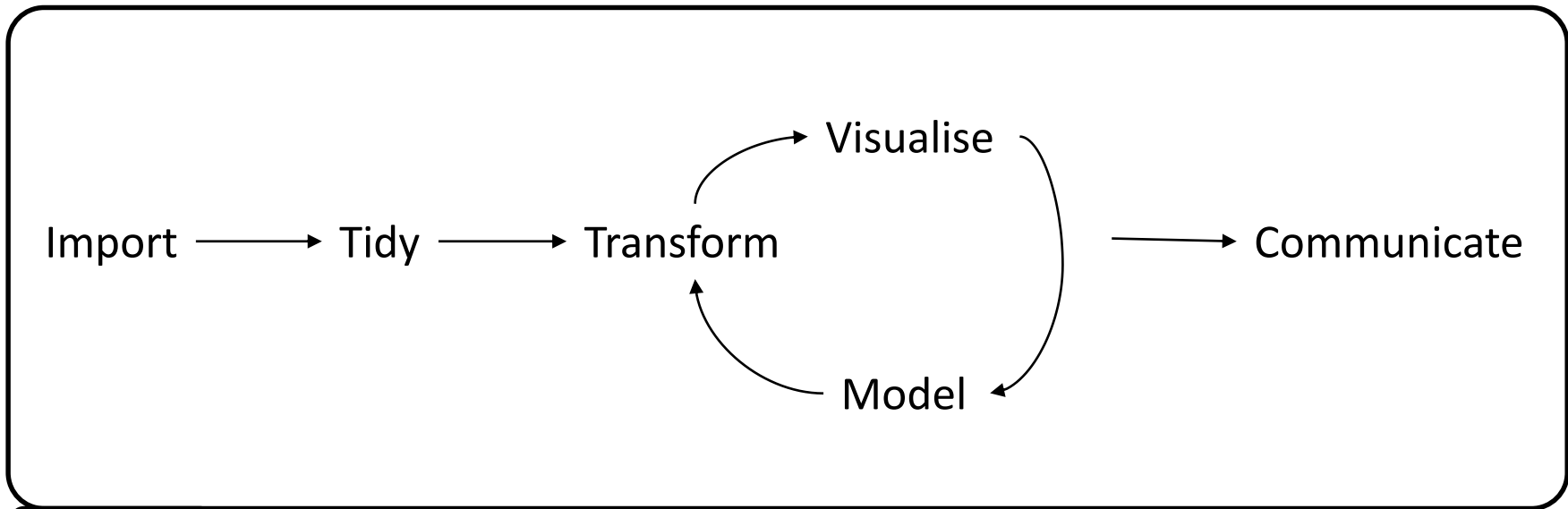
Data taken from uksport.gov.uk

Assessing Model Quality: broom

- Provides functions for extracting details on the model fit
 - Model coefficients – tidy
 - Model diagnostics – glance
- Useful for working with multiple models to compare fit

Fit of Olympics Models

```
tidy(fundingModel)  
glance(fundingModel)
```



Program

Program

- purrr

Iterating: purrr

- Iterate (over a vector of values) or apply to multiple set/subsets of data
- Output can be one of many types based on the function used:
 - map
 - map_df
 - map_dbl
 - ...

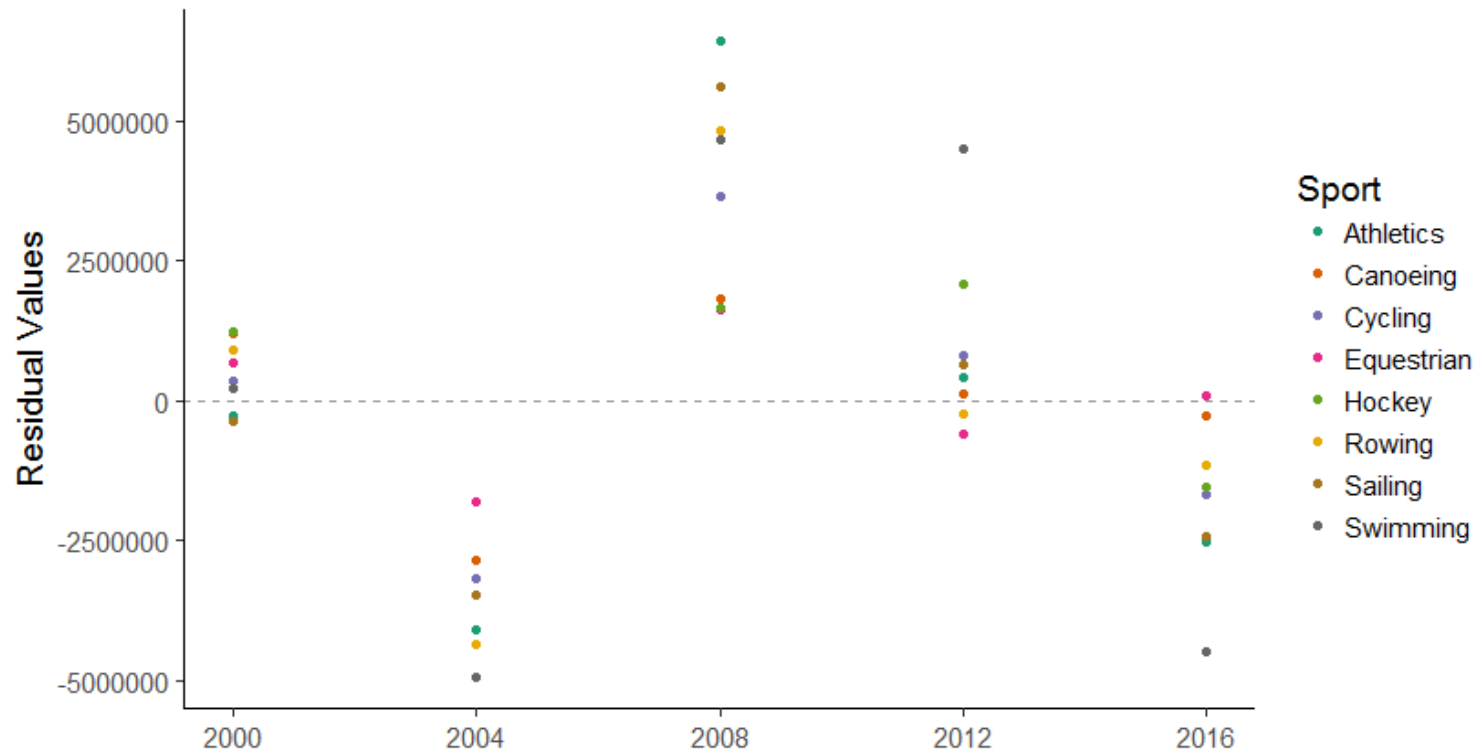
Model For Each Sport

```
sportData <- fundingTime %>%  
  group_by(Sport) %>%  
  nest()
```

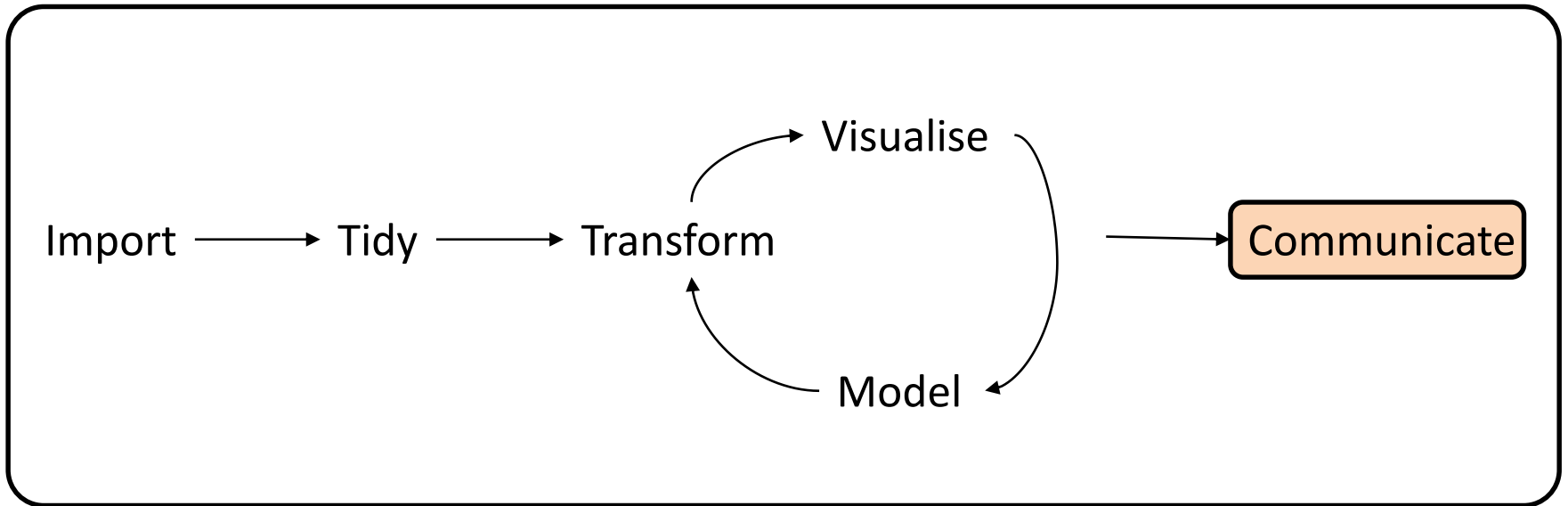
```
sportsModels <- map(sportData$data,  
  ~lm(Funding ~ Year, data = .))
```

```
sportResid <- map2_df(sportData$data,  
  sportsModels,  
  add_residuals, .id = "Sport")
```

Residual Values Suggests Further Fitting Required



Data taken from uksport.gov.uk



Program

Communicate

- There are no tidyverse packages directly aimed at communicating results (other than creating graphics)
- There are lots of packages that can be used to present results

Options for Communicating Results

- Rmarkdown
- Shiny
- Flexdashboard
- bookdown/blogdown

SUMMARY

Using the Tidyverse

- Single unified approach to manipulating and analysing data
- Provides packages for all stages of the analysis lifecycle

Learning More

- R for Data Science
- RStudio Cheat sheets
- Training

Upcoming Mango Training Courses

- An Introduction to R for Analytics
- Data Manipulation
- Data Visualisation

EARL London Workshops

- Spark and R with sparklyr
- Web Scraping and Text Analysis in R
- Writing R Functions for Fun and Profit
- Working with the MicrosoftML Package
- Working with GitHub
- Introduction to Shiny

Next LondonR Workshop

Matching Your Brand with ggplot2: Working with Scales, Themes and Guides