Getting started with Quantitative Research Methods and R

Lukas Wallrich

Last updated: 2021-02-01

Contents

Overview **7**
0.1 Further sources 7
0.2 Why R? 8

Using R and describing data **11**

1 Getting started with R **11**
1.1 Install R on your computer 11
1.2 Some key concepts 11

2 Importing and describing data **17**
2.1 Importing data into R 17
2.2 View data 18
2.3 Manipulating data and using dplyr 20
2.4 Summarise data (descriptive statistics) 21

3 Data visualisation **23**
3.1 ggplot2 23
3.2 Opinionated visualisation 25
3.3 Further resources 25

Quantitative Research **29**

4 Introduction **29**
4.1 Key concepts 29

Inferential statistics **33**

5 Simple and multiple linear regression **33**
5.1 Simple linear regression 33
5.2 Bivariate correlation 37

5.3 Multiple linear regression 39

6 Comparing means between conditions or groups 45

7 Interaction terms in linear models 53
7.1 Example 1: Memory and chess 53
7.2 Example 2: link between obesity and negative emotions in the European Social Survey 56
7.3 Example 3: link between working hours, income and life satisfaction 59

8 Chi-squared tests for associations between categorical variables 65
8.1 Tables of frequencies and proportions 65
8.2 The null hypothesis and the χ^2 statistic 66
8.3 The chisq.test() function 67
8.4 Post-hoc tests 69
8.5 Effect sizes 71

9 Reporting the results of statistical tests 73
9.1 Essential pieces of information 73

10 Survey research - testing scales 77
10.1 Creating scales 77
10.2 Further resources 79

Supplementary topics 83

11 Accessing online data sources 83
11.1 World Bank data 83
11.2 Wikidata 85
11.3 Other data sources 87

12 Dealing with missing data 89
12.1 Example 89
12.2 Further resources 93

13 Probability in statistics and real life 95
13.1 What are probabilities? 95
13.2 Combining multiple events together 96
13.3 Looking at independent events 97
13.4 Updating our beliefs - Bayes' theorem 99
13.5 Further resources 101

A Packages and references 103

B Two more pipes 105
B.1 Expanding the tidyverse: the exposition pipe (%$%) 105

B.2 A somewhat risky time-saver: the assignment pipe (`%<>%`) 106

Overview

This guide supports the Core Quantitative Methods Course offered by the Goldsmiths' Graduate School. **It is a living document and currently focuses on R at the expense of broader considerations of quantitative research, but will grow over time - please raise any issues and suggestions here.** It does not follow the order of sessions in the course; instead, it is ordered in a way that might allow you to see connections and hopefully helps to look things up more easily.

0.1 Further sources

This guide does not aim to be comprehensive, but just to provide sufficient orientation. There are many fantastic free online resources that go further.

0.1.1 Free online books

- Hadley Wickham, the leading brain behind the `tidyverse` packages, has co-authored *R for Data Science* (with Garrett Grolemund). This book does not cover inferential statistics, but explains how to use R to process, describe and visualise data in line with the aproach taken in this course.
- The *Learning statistics with R* book by Danielle Navarro focuses on explaining the different statistical tests and their application in quite a lot of detail, including the underlying maths. It is written by a psychologist, and might therefore be particularly helpful for people from that discipline.
- Statistical thinking for the 21st century is a great modern book on statistics, which also offers this handy R companion that covers most of what is needed for this course (though it is still a work-in-progress document).

0.1.2 Paper/library books

- Charles Wheelan's *Naked Statistics: Stripping the Dread from the Data* is an easy-to-read and entertaining New York Times bestseller that introduces statistical thinking and some key concepts without distracting details.
- Neil Burdess' *Starting Statistics: A short clear guide* covers the basic ground with more practical and technical detail.
- Also have a look at the **Module Guide** on the learn.gold Module Page that contains further recommendations, especially with regard to research methods.

0.1.3 Other key resources

- The RStudio team created and collected a very helpful set of Cheat Sheets that cover the key elements of various R packages - have a look here. For this course, the sheets on `dplyr` and `ggplot2` are the most important.
- Stackoverflow is an online forum with a knewledgeable and welcoming support community. However, make sure to use Google and their search function first to check that your question hasn't already been answered and show what you already understand - if the same questions keep on getting asked or if the questions are very unclear, people volunteering their time to help can get a bit testy.

0.2 Why R?

R is not the easiest statistical software to learn, but we are confident that it is the most useful. This article on why SPSS is dying provides some arguments for why that is the case, while Andy Fields (a leading stats teacher and textbook author) offers his case for learning R here.

Using R and describing data

Chapter 1

Getting started with R

1.1 Install R on your computer

R is highly customizable, so that your life is a lot easier if you use your own copy rather than a shared copy on a university computer. To install it on your computer, follow the detailed instructions in Section 1.4 of the R for Data Science book

Then **start exploring** - using R is a skill that comes with practice, not with reading about it. The examples from these notes can be copy-pasted or typed into R (when you move the mouse pointer over a code box you should see a copy-button in the top-right corner).

1.2 Some key concepts

If you understand the following concepts, using R will make a lot more sense.

1.2.1 R, RStudio and R packages

You can think about the different parts of R in terms of a computer or smartphone. In that analogy:

- **R is the processor and operating system** that carries out everything under the hood. It also contains many basic functions (i.e. commands) for everyday tasks
- **RStudio is the user interface.** The following parts are most relevant:

- **Console** to type and run commands (usually to try things out or do things that need to be done once only)
- **File editor** to develop production code (i.e. code needed to reproduce your analyses and create reports). Here you edit .Rmd files (or plain .R scripts, which are not used in this course)
- **Environment** shows a list of all variables you have created. For dataframes, it shows the number of observations (rows) and variables (columns). Clicking on the name opens the data, clicking on the arrow shows some details
- **History** is a searchable list of all commands you have run - helpful if you want to reuse something you have done a while back.

- **R packages are the apps**
  - Installed once, using `install.packages("name")`
  - Loaded in every session, using `library(name)`

For a more comprehensive introduction to R, you can watch this video:

1.2.2 Data and variables in R

Lines starting with a # are comments that are ignored by R. I use them below to explain parts of the code further.

```r
# <- saves a value (on the right) into an object (on the left)
score <- 10

# "" need to be used around text that is not an object name
month <- "January"

# c() combines several values into one vector that
# can then be saved as a single object
weekend <- c("Saturday", "Sunday")
sleep_hours <- c(4, 9)

#Variables can then be combined into dataframes (R's version of tables)
sleep_data <- data.frame(day = weekend, hours = sleep_hours)

#Variables within dataframes are accessed using the $ operator
sum(sleep_data$hours)
```

```
## [1] 13
```

Variable classes indicate what kind of data a variable contains. There are four main types

- numeric, integer or double (i.e. numbers, the differences are not relevant to us)

- character, i.e. any text
- logical, i.e. TRUE/FALSE values
- and factors, i.e. categorical variables, for example weekdays.

The `class()` function shows the class of a single variable, the `str()` and `glimpse()` functions include the classes of all variables in a dataframe.

Usually, R gets the classes right by itself and converts as required. If you need to change variable classes, you can use the `as.numeric()`, `as.character()`, `as.logical()`, `as.factor()` etc. functions.

```
class(sleep_data$hours)
str(sleep_data)
```

```
## [1] "numeric"
## 'data.frame':	2 obs. of  2 variables:
##  $ day  : chr  "Saturday" "Sunday"
##  $ hours: num  4 9
```

1.2.3 Functions in R

All work in R is based on calling functions that do something. A function is called by its name with () behind. If you just type the name, the function is printed out - you usually don't want this.

```
#Some functions work without any additional arguments
timestamp() #This shows the time this code segment was run

#Functions that work with given data are more helpful -
# that data is given as an 'argument' in brackets
print("Hello")
mean(c(1,2,3))

#Additionally, arguments can contain instructions to the function
mean(c(1,2,3, NA), na.rm = TRUE)
# NA is a missing value, na.rm tells the mean function to remove it
# before calculating the mean

#To save the results of a function into a variable, use the <- operator again
variableName <- mean(c(1,2,3, NA), na.rm = TRUE)
```

```
## ##------ Mon Feb  1 12:42:41 2021 ------##
## [1] "Hello"
## [1] 2
## [1] 2
```

To learn more about the arguments accepted by a specific function, use the `?` command to open the help page, e.g., by typing `?mean`. Arguments are matched by their name, as with `na.rm =` above; if no name is given, they are matched by their position in the function call. Usually, only very common arguments, typically the data, should be left unnamed.

1.2.4 Key R packages

The most essential packages for this course are included in the `tidyverse` - a whole set of packages to make data manipulation and visualisation in R efficient and consistent. They include:

- `dplyr`: a package to efficiently manipulate data and create summary statistics
- `ggplot2`: a package to create all kinds of graphs
- `readr`: a package to load table data into R that is formatted as text

All these packages are loaded with the `library(tidyverse)` command. There are some additional packages in the tidyverse that need to be loaded separately, using `library(packagename)`. The most relevant for us are:

- `readxl` and `haven`: package to load data from Excel and statistical software
- `stringr`: a package to manipulate strings
- `lubridate`: a package to convert, filter and analyse times and dates

1.2.4.1 Using the `pacman` package

As an alternative to `install.packages()` and `library()`, you can also use the `p_load` function in the `pacman` package. That function checks first whether the package is installed, installs it if needed, and then loads it. I am using it in the code here so that you easily copy-paste parts of it; in your own code, the choice is up to you. If you want to use `pacman`, just run `install.packages("pacman")` and then use `pacman::p_load(package1, package2)` to load your packages.

1.2.4.2 Using a single function in a package

If you just need to use one function from a package, you do not need to load the whole package. You can also give R the whole address and use `package::function()`. I will only do that for `pacman::p_load()` since that is the only function from that package we will ever use in this course.

1.2.5 RMarkdown

A key strength of R are **reproducible analyses.** For that, we don't want to type commands and just write down the results, but rather create scripts that can produce results again and again. That also makes changes much less painful, for instance if you've made a mistake at the start.

RMarkdown (.Rmd) files are such scripts that additionally allow to add text and split the code into separate chunks. Within them, *grey chunks* are code, anything else is not executed. It is just formatted (e.g., with headings and highlights) when you create the report. Markup symbols allow for easy formatting of the text, as per the instructions here. (Most of that is not required in this course, a couple of key symbols such as # to indicate headings will be introduced later.)

Within *.Rmd* files, you need to **Insert** new code blocks for the major parts of the analysis. For that, use the insert button at the top of the File Editor window. When you want to test your code, you can also **Run** it there or by pressing *Ctrl+Enter* in the line you want to use (or after you have selected multiple lines to run). Finally, to create the report you need to **Knit** it, again using the button in the File Editor Without installing anything else, you can only knit to *HTML files*, but those can be opened in any internet browser, so are good to share with anyone.

1.2.6 HELP! What to do when problems appear?

When **red text** appears in the console, don't panick. Some messages that are often entirely irrelevant are displayed in red, for example when packages are loaded. **Warnings** are more important, but they just alert you that something might be going wrong - read them, but then decide whether you need to worry. For instance, ggplot prints a warning when data is missing, which might or might not be a problem for you. Only **Errors** always need to be dealt with as they stop your commands from being executed and the .Rmd file from being knitted. Unfortunately, many R functions give rather cryptic error messages - if you are stuck, first check your code for typos, then Google the error message.

Many errors occur due to **common mistakes**, that you need to pay attention to:

- **Missing brackets** - make sure each `(` is matched by a `)` in the right place. When your cursor is at a closing bracket, R highlights the opening bracket it is matched with, which helps to check your syntax
- **Misspellings** - variable names and functions need to be spelled correctly, which includes capitalisation, otherwise you might inadvertently create duplicates or cause other issues. Some functions are available in both American and British spelling (e.g., `summarise()`and `summarize()` in dplyr), but this is not always the case so that it might be sensible to always stick to American spelling
- **Missing library() calls** to load packages - if a function is not found, run `library(packagename)`

- **Function masking** - many packages have functions of the same name, so that packages that are loaded later mask (i.e. "over-write") the names of functions in packages loaded earlier (for example, dplyr's `filter()` function masks the `filter()` function in base R). Often, that is exactly what we want, but if a function does not do what you expect it to do, it's worth specifying the package it comes from to make sure you are not actually calling a different function that has masked it. You can do that by adding the package name and `::`, for example, by calling `dplyr::filter()` *Note*: If you only want to use a single function from a large package, this is an alternative to loading the full package.

As you progress in R, **do not try to remember everything** - instead, look things up as needed, in particular things like argument names:

- Use `?functionName` to get help in R
- Refer to the RStudio Cheatsheets
- Google your questions and pay particular attention to results from r-bloggers.com and stackoverflow.com

There is also a very active R user community that is willing to help - stackoverflow.com is a good place to ask questions about R Code, while stats.stackexchange.com is the place to go with questions that are more generally about statistics. In both cases, it pays off to first google and then post questions that are as specific as possible.

Chapter 2

Importing and describing data

2.1 Importing data into R

Generally, R is a poor choice for data entry, so you will usually import data from spreadsheet software or other sources. That also helps to keep data and analyses apart, and thereby increases transparency.

The most common ways to import data into R are to

- read tabular data saved as text (such as the common *.csv* files) with the `readr` package
- read Excel files with the `readxl` package
- read data files from SPSS or SAS with `haven`.

In these packages, the functions to read data are always `read_x()`, with x replaced by the type of file you want to open. You always need to specify the name of the datafile, with its path (unless it is in the same folder as your .Rmd file), as the first argument of the `read_x()` function. For the path, it can be helpful to start with the dot (`.`), which means: start from the current folder. For example, I often have a data folder within the folder where the .Rmd file is, so that the path would then be `"./data/file.csv"`. Always remember to assign the result of the function (i.e. the data) to a variable using `<-`, otherwise the data is just printed and not saved in R.

The functions that load data from tables need to establish the variable class for each variable. They can either guess (which you will then need to check), or you can provide the classes in the `col_types`-argument. Check `?readr::read_csv` for details.

```
#readr is loaded with the tidyverse, but can be loaded separately
#as well if you don't need the other packages
pacman::p_load(readr)
gapminder <- read_csv("./data/gapminder.csv", col_types = "finnnnnff")

pacman::p_load(readxl)
gapminder <- read_xlsx("./data/gapminder.xlsx")
# Here col_types are not defined so that R will guess and print
# out the results - make sure to check whether the classes are
# what you would expect

pacman::p_load(haven)
gapminder <- read_sav("./data/gapminder.sav")
#SPSS files contain information on the class of each variable,
#so that col_types do not need to be defined
```

There are shortcuts to quickly get some data into R, which can be helpful if you just want to try something out - all of them only make sense in the Console, not in a .Rmd script. Using the `edit()` function, you can open up a simple spreadsheet to edit and input data, using `read.table(file="clipboard", sep="\t")`, you can get data you copied in Excel. As with all ways of importing data, remember to assign the result to a variable.

```
#If you want to edit an existing dataset
gapminder_corrected <- edit(gapminder)

#If you want to type or copy in a new dataset
yourdata <- edit(data.frame())
#The data.frame() function creates an empty dataframe for you to then edit

#If you want to read data from the clipboard
pasted <- read.table(file="clipboard", sep="\t")

#If that data contains variable names in the first row:
pasted <- read.table(file="clipboard", sep="\t", header = TRUE)

#Note that reading data from the clipboard does not always work -
#if it doesn't, try pasting your data into the window opened
#with the edit() function
```

2.2 View data

When you start with a new dataset, there are some helpful functions to get a first look at the data: `glimpse()`gives a good overview of the variables contained in the dataset, while `head()` and `tail()` print the first and last lines. In the **Console** (but not really within scripts such as *.Rmd* files), you can also use `View()` to open up the whole dataset. Finally, you can have a look at data in the **Environment** pane in RStudio.

```
pacman::p_load(dslabs) #Load the gapminder teaching dataset
pacman::p_load(dplyr) #Load the dplyr package that offers glimpse()
glimpse(gapminder)
```

```
## Rows: 10,545
## Columns: 9
## $ country          <fct> Albania, Algeria, Angola, Antigua and Barbuda, Arg...
## $ year             <int> 1960, 1960, 1960, 1960, 1960, 1960, 1960, 1960, 19...
## $ infant_mortality <dbl> 115.40, 148.20, 208.00, NA, 59.87, NA, NA, 20.30, ...
## $ life_expectancy  <dbl> 62.87, 47.50, 35.98, 62.97, 65.39, 66.86, 65.66, 7...
## $ fertility        <dbl> 6.19, 7.65, 7.32, 4.43, 3.11, 4.55, 4.82, 3.45, 2....
## $ population       <dbl> 1636054, 11124892, 5270844, 54681, 20619075, 18673...
## $ gdp              <dbl> NA, 13828152297, NA, NA, 108322326649, NA, NA, 966...
## $ continent        <fct> Europe, Africa, Africa, Americas, Americas, Asia, ...
## $ region           <fct> Southern Europe, Northern Africa, Middle Africa, C...
```

```
head(gapminder, n=5) #n defines number of rows shown
```

```
##               country year infant_mortality life_expectancy fertility
## 1             Albania 1960           115.40           62.87      6.19
## 2             Algeria 1960           148.20           47.50      7.65
## 3              Angola 1960           208.00           35.98      7.32
## 4 Antigua and Barbuda 1960               NA           62.97      4.43
## 5           Argentina 1960            59.87           65.39      3.11
##   population          gdp continent          region
## 1    1636054           NA    Europe Southern Europe
## 2   11124892  13828152297    Africa Northern Africa
## 3    5270844           NA    Africa   Middle Africa
## 4      54681           NA  Americas       Caribbean
## 5   20619075 108322326649  Americas   South America
```

```
tail(gapminder, n=5)
```

```
##                   country year infant_mortality life_expectancy fertility
## 10541 West Bank and Gaza 2016               NA           74.70        NA
## 10542            Vietnam 2016               NA           75.60        NA
## 10543              Yemen 2016               NA           64.92        NA
## 10544             Zambia 2016               NA           57.10        NA
## 10545           Zimbabwe 2016               NA           61.69        NA
##       population gdp continent             region
## 10541         NA  NA      Asia       Western Asia
## 10542         NA  NA      Asia South-Eastern Asia
## 10543         NA  NA      Asia       Western Asia
## 10545          NA   NA     Africa     Eastern Africa
## 10545          NA   NA     Africa     Eastern Africa
```

2.3 Manipulating data and using dplyr

Once you have imported your data into R, you might need to filter it, sort it, add some new variables, and calculate summary statistics. The `dplyr` package (part of the `tidyverse`) is designed to make all these steps easier, so we use it extensively during this course. `dplyr` is based on two main ideas:

- Develop a chain of data manipulation steps that get you towards the desired data, with the steps connected by the `%>%` operator ("pipe"). That removes the need for nested functions or for saving lots of intermediate results.
- Use functions names after natural language verbs to develop code that can be easily understood.

`%>%` takes the argument on the left and places it as the first argument into the function on the right. For example:

```
## [1] 2
## [1] 2
```

The most important `dplyr` functions are:

- `select()`: select specific variables
- `filter()`: filter rows based on condition
- `arrange()`: sort data ascendingly (arrange(desc()) to reverse)
- `mutate()`: create/change variables
- `summarise()`: calculate summary statistics
- `group_by()`: separate data into groups, usually to summarise by group

All functions use a dataframe as their first argument, usually from `%>%`. After that, variables in the dataframe are accessed just with their name (no `$`).

If we want to calculate the average income per capita in 2010 on each continent from the gapminder dataset, we need most of these functions - if you **read the `%>%`-operator as 'then'**, and mentally add 'take' at the very start, you should be able to follow along quite naturally.

One thing to note: you can split R code into multiple lines as long as each line is incomplete. Since the lines here end with `%>%`, R includes the next line into the same command. If that operator was moved to the start of the next line, the code would no longer work.

```r
pacman::p_load(dplyr)
gapminder %>%
  filter(year == 2010) %>%
    mutate(gdpPerCap = gdp/population) %>%
      filter(!is.na(gdpPerCap)) %>%
      #This filter() removes countries where either gdp or population is missing
        group_by(continent) %>%
        #group_by() allows for summary statistics to be calculated
        #for each continent separately
          summarise(AvgGdpPerCap_Nations = mean(gdpPerCap),
                    AvgGdpPerCap_People = sum(gdp)/sum(population)) %>%
              mutate_if(is.numeric, round, 0) %>%
              #This rounds all numeric variables to 0 decimal places
              #(beyond expectations for this course)
                arrange(desc(AvgGdpPerCap_Nations))
```

```
## # A tibble: 5 x 3
##   continent AvgGdpPerCap_Nations AvgGdpPerCap_People
##   <fct>                    <dbl>               <dbl>
## 1 Europe                   15214               14643
## 2 Asia                      8163                3277
## 3 Americas                  6797               16305
## 4 Oceania                   5281               17942
## 5 Africa                    1303                 867
```

Just as an aside: consider the differences between the two ways of calculating averages per continent. One focuses on the average of country values, and thus yields the average national income per capita. The other takes population sizes into account and thus yields the average personal income per capita. For most continents, the difference in results is huge. Both approaches are used in the media - so it's always worth looking a little closer at summary statistics.

For an introduction to `dplyr`, you can watch this video:

2.4 Summarise data (descriptive statistics)

Next you might want to use summary functions - either in a dplyr `summarise()` function or on their own. For most of them, you can use the `na.rm = TRUE` argument to tell R to ignore missing values (only do that when you have reason to assume that missing values can safely be ignored).

```
gapminder2010 <- gapminder %>% filter(year==2010)

#Global population
sum(gapminder2010$population, na.rm = TRUE)
```

```
## [1] 6778331427
```

```
#Number of countries included
nrow(gapminder2010)
```

```
## [1] 185
```

```
#To get the number inside dplyr's summarise function, it would have to be
gapminder2010 %>% summarise(n_rows = n())
```

```
##   n_rows
## 1    185
```

```
#Average (mean) number of children per woman
mean(gapminder2010$fertility, na.rm = TRUE)
```

```
## [1] 2.885297
```

```
#Average (median) number of children per woman
median(gapminder2010$fertility, na.rm = TRUE)
```

```
## [1] 2.38
```

```
#Highest and lowest fertility rates
max(gapminder2010$fertility)
```

```
## [1] 7.58
```

```
min(gapminder2010$fertility)
```

```
## [1] 1
```

```
#Countries with highest and lowest fertility.
#There are many ways to look this up - here is a simple dplyr pipe
## Note that | means 'or' in the context of logical comparisons
## and == is needed to test for equality because = just assigns a value.

gapminder2010 %>%
  filter(fertility == max(fertility) | fertility == min(fertility)) %>%
    select(country, fertility)
```

```
##         country fertility
## 1 Macao, China      1.00
## 2        Niger      7.58
```

Chapter 3

Data visualisation

Being able to look at the data is a key step in data exploration, during the analysis and in the communication of results. R has a range of powerful tools to create graphs quickly, and then to develop them into a publication-ready format where helpful.

For an introduction to the Grammar of Graphics and ggplot2, the package that we will be using for visualisation, watch this video:

Click here for the code used in the video

3.1 ggplot2

We use the `ggplot2`-package because it offers a consistent way to create anything from simple exploratory plots to complex data visualisation. Each graph command needs certain parts:

- a call to the `ggplot`-function and the **data** as the first argument: `ggplot(gapminder,`
- a mapping of the **aesthetics**, i.e. of variables to visual elements. This uses the `aes`-function that is given to `ggplot` as the second argument: `aes(x=infant_mortality, y=fertility, col=continent))` (Note that two closing brackets are needed as this also completes the ggplot function call)
- a **geometry**, i.e. a type of chart, that is added with a plus-symbol and the function call, e.g., `+ geom_point()`
- optional elements such as labels that are included again with plus-symbols and function calls, e.g., `+ labs(title = "Association of infant mortality and fertility", subtitle="2010 data from gapminder.org")`

Multiple geometries can be layered on top of each other, for example to add trend lines to scatterplots. In that case, `aes()` functions can be included into the `geom_x()`-functions to make some of the mappings specific to certain geometries. This is done in the example below to color the points by continent without applying that aestethic to the line - if it was included in the main `aes()`-function, the plot would contain a separate coloured line for each continent.

Note: Line breaks are completely up to you and can be used to make the code readable as long as the command does not appear complete too early - to keep it simple, there should always be an open bracket, a comma or a + before a line break within the command to create a ggplot-chart

```
pacman::p_load(dslabs, dplyr, ggplot2)
gapminder2010 <- gapminder %>% filter(year==2010)

ggplot(gapminder2010, aes(x=infant_mortality, y=fertility)) +
  geom_point(aes(col=continent)) + geom_smooth() +
    ggtitle("Association of infant mortality and fertility",
            subtitle="2010 data from gapminder.org")
```

```
## Warning: Removed 7 rows containing non-finite values (stat_smooth).

## Warning: Removed 7 rows containing missing values (geom_point).
```

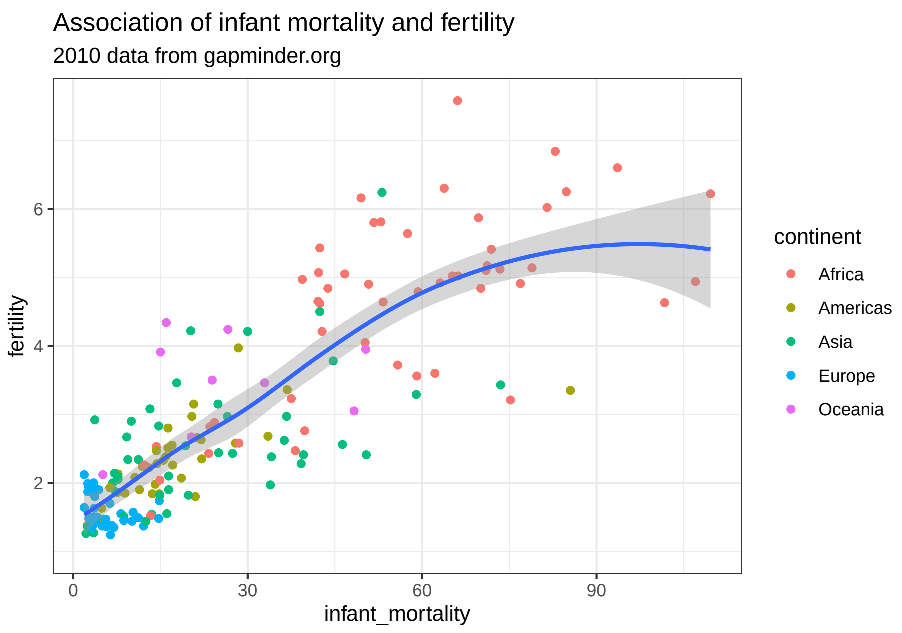


Figure 3.1: A simple ggplot example

3.2 Opinionated visualisation

It is easy to find examples of misleading charts that should never have been published. However, even when all the elements of a chart are legitimately presented, the same data can still be used to suggest radically different conclusions. See the following example and follow the source link to read more about "opinionated" data visualisations. Apart from being critical when seeing charts, the take-away message from this is to be conscious of the power of small design choices - that is a power you should wield consciously when creating charts.

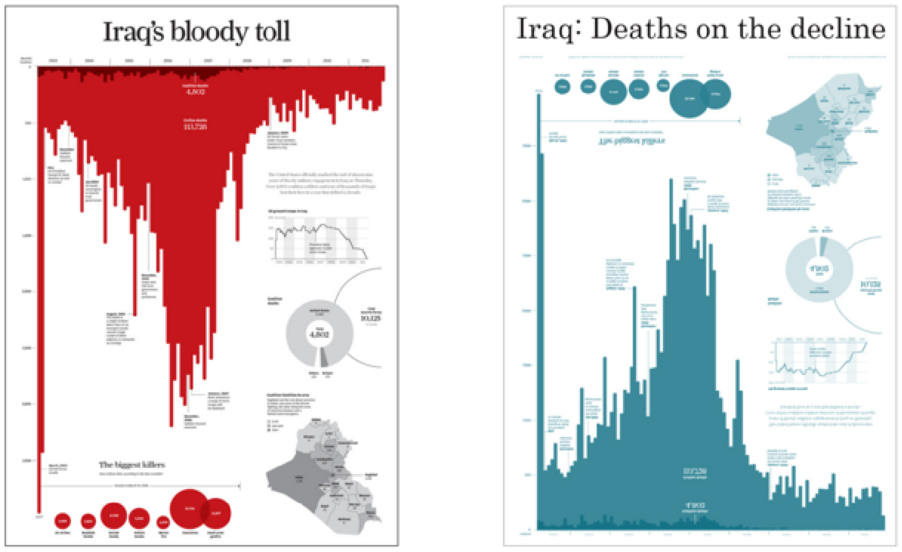


Figure 3.2: Same data, two messages (Source: <a href="https://www.infoworld.com/article/3088166/why-how-to-lie-with-statistics-did-us-a-disservice.html">infoworld.com</a>)

3.3 Further resources

- The R Graphics Cookbook by Winston Chang is available online with 150 "recipes" that cover everything from basic exploratory charts to colour-coded maps.
- The BBC graphics team has published their own R Cookbook with many tips for making charts that convey a clear message, as well as some custom functions for making clean publication-ready charts.
- Irizarry's *Introduction to Data Science* has a good chapter on data visualisation principles

Quantitative Research

Chapter 4

Introduction

For an introduction to conducting and evaluating quantitative research, you can watch this video:

4.1 Key concepts

If you understand the following concepts, you will be in a good place to understand and assess quantitative research:

- **Hypothesis testing:** quantitative research aims to develop hypotheses and then check whether new data is consistent with these hypotheses. This sequence matters particularly for inferential statistics, i.e. for knowing how to determine whether a certain pattern in the data is likely to arise due to random variation between samples we look at, or due to a real difference in the population of interest.
- **Experiments with randomisation and blinding**. In essence, the idea is that if participants are randomly allocated to groups, the only difference between the groups is what happens during the experiment. If there is adequate blinding (i.e. experimenters and participants do not know what condition they are assigned to), the only difference is that of interest (e.g., the content of a pill). Otherwise, experimenter and participant expectations will also differ, which might change results. You might want to have a look at this detailed but brief discussion of blinding
- **Correlational/observational studies** and their inability to show causality: observational studies can be great to describe reality and to highlight associations, but without experiments (or more complex designs that aim to approximate experiments), we must be careful not to

confuse correlation (co-occurrence of two variable) with causation (causal dependence of one on the other).

- **Validity**
  - **Internal validity**: degree to which a study can rule out alternative explanations for the effects observed, apart from those highlighted by the researchers. This is most concerning in experimental research. A list of things that might reduce internal validity can be found here
  - **External and ecological validity:** all about generalizability of results. External validity concerns generalisation to other participants, while ecological validity concerns the generalisation to real situations of interest. Detailed notes here

Inferential statistics

Chapter 5

Simple and multiple linear regression

This document recaps how we assess relationships between variables when at least the outcome variable is continuous.

5.1 Simple linear regression

For an introduction to simple linear regression and correlation, watch this video:

When considering the relationship between two continuous variables, we should always start with a scatterplot - for example, of the relationship between social trust and life satisfaction at the national level in European countries (data from the European Social Survey 2014).

```r
pacman::p_load(tidyverse)
#Data preparation
ess <- read_rds(url("http://empower-training.de/Gold/round7.RDS"))
ess <- ess %>% mutate(soctrust = (ppltrst + pplfair + pplhlp)/3)
nat_avgs <- ess %>% group_by(cntry) %>%
  summarize(nat_soctrust = mean(soctrust, na.rm=T),
            nat_stflife = mean(stflife, na.rm=T))

ggplot(nat_avgs, aes(x=nat_soctrust, y=nat_stflife)) + geom_point()
```

The scatterplot can now help us to think about what kind of model would help us to explain or predict one variable based on the other. If a straight-line relationship seems reasonable, we can use a linear model.

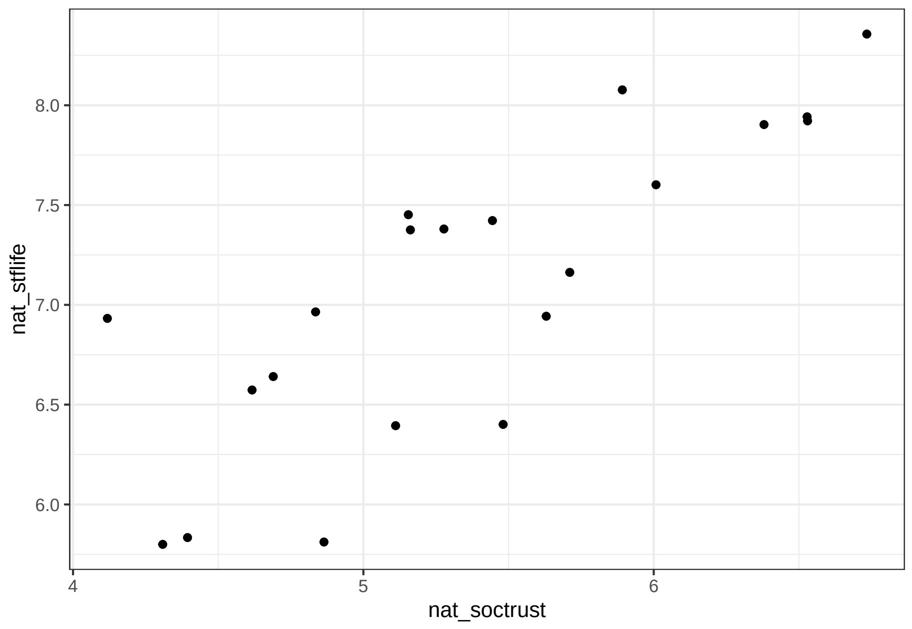


Figure 5.1: Scatterplot relating social trust to life satisfaction

5.1.1 Plotting a simple linear regression model

Geometrically, a simple linear regression model is just a straight line through the scatterplot, fitted in a way that minimises the distances from the points to the line. It can be plotted with the `geom_smooth(method="lm")` function.

```r
ggplot(nat_avgs, aes(x=nat_soctrust, y=nat_stflife)) +
  geom_point() + geom_smooth(method="lm", se = FALSE)
```

```r
  #se=FALSE hides confidence bands that are often just visual clutter
```

For each value of one variable, this line now allows us to find the corresponding expected value of the other variable.

5.1.2 Mathematical representation of a simple linear regression model

The idea of a simple linear regression model is to link one predictor variable, x, to an outcome variable, y. Their relationship can be expressed in a simple formula:

$$y = \beta_0 + \beta_1 * x$$

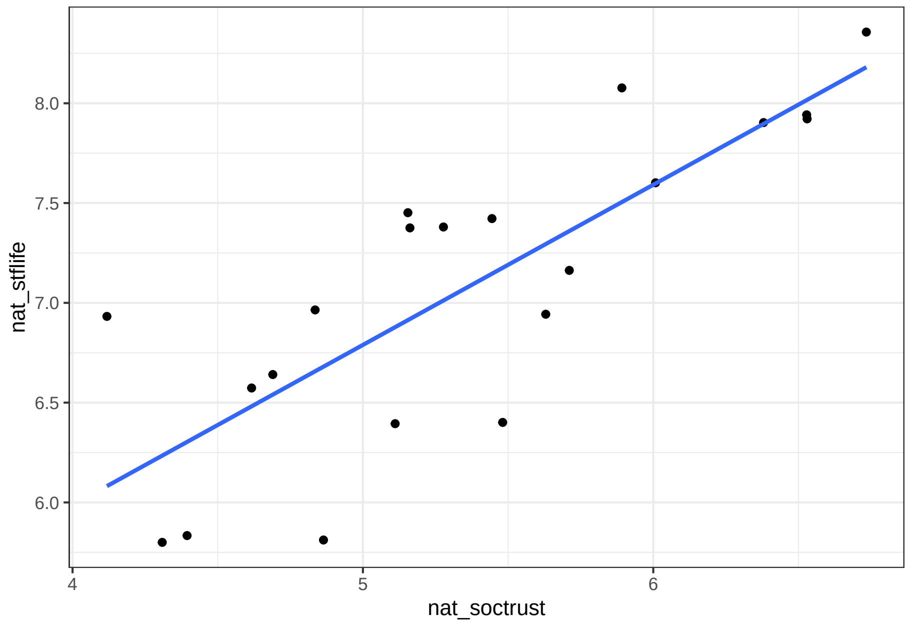


Figure 5.2: Scatterplot with added regression line (”line of best fit”)

Here, β_1 is the most important parameter - it tells us, how much a change of 1 in the x-variable affects the y-variable. Geometrically, it is the slope of the regression line. β_0 is the intercept of that line, i.e. the value of y when x is 0 - since x can often not actually be zero, that parameter tends to be of little interest on its own.

5.1.3 Fitting a simple linear regression model in R

Linear models are fitted in R using the `lm()` function. The model itself is specified as a formula with the notation `y ~ x` where the `~` means ‘is predicted by’ and the intercept is added automatically.

```
lm(nat_stflife ~ nat_soctrust, data = nat_avgs) %>% summary()
```

```
##
## Call:
## lm(formula = nat_stflife ~ nat_soctrust, data = nat_avgs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.86854 -0.35172  0.00667  0.30745  0.85047
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.7767     0.7311   3.798  0.00122 **
## nat_soctrust   0.8025     0.1347   5.957 9.84e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4633 on 19 degrees of freedom
## Multiple R-squared:  0.6513,Adjusted R-squared:  0.6329
## F-statistic: 35.49 on 1 and 19 DF,  p-value: 9.839e-06
```

In the summary, the first column of the coefficients table gives the estimates for β_0 (in the intercept row) and β_1 (in the nat_soctrust row). Thus in this case, the regression formula could be written as:

$$LifeSatisfaction = 2.76 + 0.80 * SocialTrust$$

Clearly this formula does not allow us to perfectly predict one variable from the other; we have already seen in the graph that not all points fall exactly on the line. The distances between each point and the line are the **residuals** - their distribution is described at the top of the `summary()`-function output.

5.1.3.1 Interpreting R^2

The model output allows us to say how well the line fits by considering the R^2 value. It is based on the sum of the squares of the deviations of each data point from either the mean (SS_{total}) or the model ($SS_{residual}$).

If I was considering a model that predicted people's height, I might have a person who is 1.6m tall. If the mean of the data was 1.8m, their squared total deviation would be $0.2 * 0.2 = 0.04$. If the model then predicted their height at 1.55m, their squared residual deviation would be $-0.05 * -0.05 = 0.025$. Once this is summed up for all data points, R^2 is calculated with the following formula:

$$R^2 = 1 - \frac{SS_{residual}}{SS_{total}}$$

Given that the sum of squared residuals can never be less than 0 or more than the sum of the total residuals, R^2 can only take up values between 0 and 1. It can be understood as the share of total variance explained by the model (or to link it to the formula: as the total variance minus the share *not* explained by the model).

5.1.3.2 Interpreting Pr(>|t|), the *p*-value

Each coefficient comes with an associated *p*-value in the summary that is shown at the end of the row. As indicated by the column title, it indicates the probability that the test statistic would be this large or larger *if* the null hypothesis was true and there was no association in the underlying population.

As per usual, we would typically report coefficients with a *p*-value below .05 as statistically significant. The significance level for the intercept is often not of interest - it simply indicates whether the value of y is significantly different from 0 when x is 0.

There is also a *p*-value for the overall model that essentially tests whether R^2 is significantly greater than 0. This is reported at the very bottom of the `summary()`-output. If there is only one predictor variable, this will the same as the *p*-value for β_1, but it will be of greater interest when there are several predictors.

5.2 Bivariate correlation

Simple linear regression can tell us whether two variables are linearly related to each other, whether this finding is statistically significant (i.e. clearer than what we might expect due to chance), and how strong the relationship is *in terms of the units of the two variables.* Quite often, however, we would prefer a standardised measure of the strength of a relationship, and a quicker test. That is where correlation comes in.

The **Pearson's correlation coefficient** is the slope of the regression line when both variables are standardised, i.e. expressed in units of their standard deviations (as so-called z-scores). It is again bounded, between -1 and 1. As it is unit-free, it can give quick information as to which relationships matter more and which matter less.

5.2.1 Calculating the correlation coefficient

Before calculating a correlation coefficient, **you need to look at the scatterplot and make sure that a straight-line relationship looks reasonable.** If that is the case, you can use the `cor.test()` function to calculate the correlation coefficient.

```
cor.test(nat_avgs$nat_soctrust, nat_avgs$nat_stflife)
#If you have missing data, use the na.rm = TRUE argument
#to have it excluded before the calculation
```

```
##
```

```
##  Pearson's product-moment correlation
## 
## data:  nat_avgs$nat_soctrust and nat_avgs$nat_stflife
## t = 5.957, df = 19, p-value = 9.839e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5760041 0.9186640
## sample estimates:
##       cor 
## 0.8070221
```

The estimated `cor` at the very bottom of the output is the correlation coefficient, usually reported as $r = .81$. This shows that there is a strong positive relationship between the two variables. Check the p-value to see whether the relationship is statistically significant and the 95%-confidence interval to see how precise the estimate is likely to be.

5.2.2 Equivalence to linear regression

Just to show that this is indeed equivalent to simple linear regression on the standardised variables, which can be calculated using the `scale()`-function.

```
ggplot(nat_avgs, aes(x=scale(nat_soctrust), y=scale(nat_stflife))) +
  geom_point() + geom_smooth(method="lm", se=FALSE) + labs(x="National Social Trust (s
```

```
lm(scale(nat_stflife) ~ scale(nat_soctrust), data = nat_avgs) %>% summary()
```

```
## 
## Call:
## lm(formula = scale(nat_stflife) ~ scale(nat_soctrust), data = nat_avgs)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -1.13572 -0.45992  0.00872  0.40203  1.11209 
## 
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)    
## (Intercept)         1.910e-16  1.322e-01   0.000        1    
## scale(nat_soctrust) 8.070e-01  1.355e-01   5.957 9.84e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.6059 on 19 degrees of freedom
## Multiple R-squared:  0.6513,Adjusted R-squared:  0.6329 
## F-statistic: 35.49 on 1 and 19 DF,  p-value: 9.839e-06
```

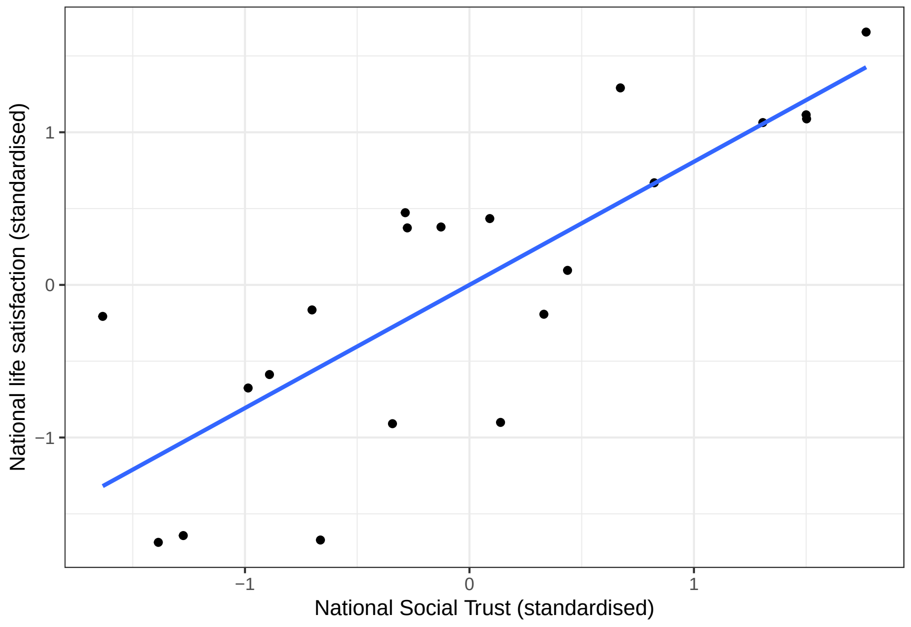


Figure 5.3: Scatterplot with scaled variables (z-scores)

Note that the regression coefficient for the scaled variable is identical to the correlation coefficient shown above.

5.3 Multiple linear regression

Linear models can easily be extended to multiple predictor variables. Here I will primarily focus on linear models that include categorical predictor variables.

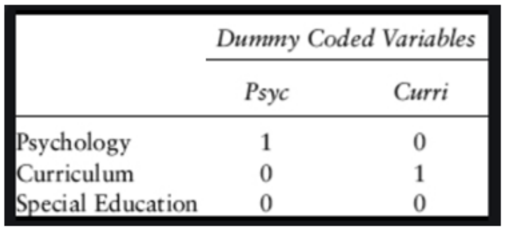
You can watch this video to take you through the process of extending simple linear regression to include multiple variables step-by-step:

5.3.1 Dummy coding revisited

Multiple linear regression models often use categorical predictors. However, they need to be turned into numbers. This is done through automatic dummy coding, which creates variables coded as 0 and 1 (so-called dummy variables).

Note that dummy coding always results in **one fewer dummy variable than the number of levels of the categorical variable**. For example, a *gender* variable with two levels (male/female) would be recoded into a single dummy variable *female* (0=no, 1=yes). Note that there is no equivalent variable *male* as that would be redundant. Given that the hypothetical gender variable here is defined as either male or female, not female necessarily implies male.

The same applies to larger number of levels - see below how three possible values for the department variable would be recoded into two dummy variables.



	Dummy Coded Variables	
	Psyc	*Curri*
Psychology	1	0
Curriculum	0	1
Special Education	0	0

Figure 5.4: Example for dummy-coding

The level that is not explicitly mentioned anymore is the **reference level** - in the case above, that is Special Education. In a linear model, that is what all other effects are compared to, so it is important to keep in mind which that is.

5.3.2 Mathematical structure of multiple linear models

Linear models are characterised by intercepts and slopes. Geometrically, intercepts shift a line or plane along an axis, while a slope changes its orientation in space. Conceptually, intercepts are like on/off switches, while slopes indicate what happens when variables are dialed up or down.

For example, we might want to predict life expectancy based on a person's gender and their level of physical activity. Then the model would be

$$LifeExpectancy = \beta_0 + \beta_1 * female + \beta_2 * activity$$

β_0 here would describe the notional life expectancy of males without any activity, $\beta_0 + \beta_1$ would describe that of females without any exercise - so both of these are intercepts, the $female$ variable simply determines which intercept we choose. β_2 is now the effect of each unit of activity, i.e. the slope of the regression line.

5.3.3 Running multiple linear regression models in R

In the `lm()`-function, it is very easy to keep on adding predictors to the formula by just using the `+`-symbol. For example, we can consider the UK General Election Results in 2019 and see what helps explain the vote share of the Conservatives.

```
#Load and prep data
constituencies <- read_csv(url("http://empower-training.de/Gold/ConstituencyData2019.csv"), col_t
constituencies <- constituencies %>% filter(RegionName != "Northern Ireland") %>%
          mutate(nation = case_when(RegionName == "Scotland" ~ "Scotland",
                                    RegionName == "Wales" ~ "Wales",
                                    T ~ "England"))  %>%
          filter(ConstituencyName != "Chorley")

#Simple linear regression model:
lm(ElectionConShare ~ MedianAge, data = constituencies) %>% summary()
```

```
##
## Call:
## lm(formula = ElectionConShare ~ MedianAge, data = constituencies)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41131 -0.09130  0.01502  0.10007  0.30207
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.252020   0.040956  -6.153 1.35e-09 ***
## MedianAge    0.017104   0.001003  17.054  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1384 on 629 degrees of freedom
## Multiple R-squared:  0.3162,Adjusted R-squared:  0.3151
## F-statistic: 290.8 on 1 and 629 DF,  p-value: < 2.2e-16
```

We now know that the Conservative vote share seems to be strongly related to the median age of the constituency, and can find and interpret the coefficient estimate (1.7 pct-points per year of median age), significance value and R^2 as above. What happens when we add nation, to account for differences between Scotland, England and Wales?

```
#Declare nation as factor and then set the reference level
#for categorical predictors
constituencies$nation <- factor(constituencies$nation) %>% relevel(ref = "England")
lm(ElectionConShare ~ MedianAge + nation, data = constituencies) %>% summary()
```

```
##
## Call:
## lm(formula = ElectionConShare ~ MedianAge + nation, data = constituencies)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.37287 -0.06969  0.00687  0.07799  0.27545
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -0.2777133  0.0337157  -8.237 1.03e-15 ***
## MedianAge       0.0185574  0.0008295  22.373  < 2e-16 ***
## nationScotland -0.2527752  0.0156649 -16.136  < 2e-16 ***
## nationWales    -0.1493447  0.0187193  -7.978 7.08e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1138 on 627 degrees of freedom
## Multiple R-squared:  0.5392,Adjusted R-squared:  0.537
## F-statistic: 244.6 on 3 and 627 DF,  p-value: < 2.2e-16
```

Now we have a more complex model. Based on the coefficient estimates, we would now estimate the vote share as follows

$$VoteShare = -0.27 + 0.018 * age + -0.25 * Scotland + -0.15 * Wales$$

The figures for Scotland and Wales need to be compared to the unnamed reference level, i.e. to England - so we would expect a Scottish constituency to have a 25 percentage points lower Conservative vote share *when keeping median age constant.* Likewise, we now would expect an increase of the median age by 1 year to increase the Conservative vote share by 1.8 percentage points, *keeping the effect of nation constant.*

```
#The moderndive is needed to plot parallel regression lines
pacman::p_load("moderndive")

ggplot(constituencies, aes(x=MedianAge, y=ElectionConShare, color = nation)) + geom_po
```

5.3.3.1 Significance testing and reporting in multiple regression models

With more than one predictor, we have two questions:

- is the **overall model** significant, i.e. can we confidently believe that its estimates are better than if we just took the overall mean as our estimate for each constituency? That is shown in the last line of the summary. Here we would report that the regression model predicting the conservative vote share per constituency based on its median age and the nation it was located in was significant, with $F(3, 627) = 244.6$, $p < .001$, and explained a substantial share of the variance, $R^2 = .54$.
- does **each predictor** explain a significant share of unique variance? That is shown at the end of each line in the coefficients table. With many

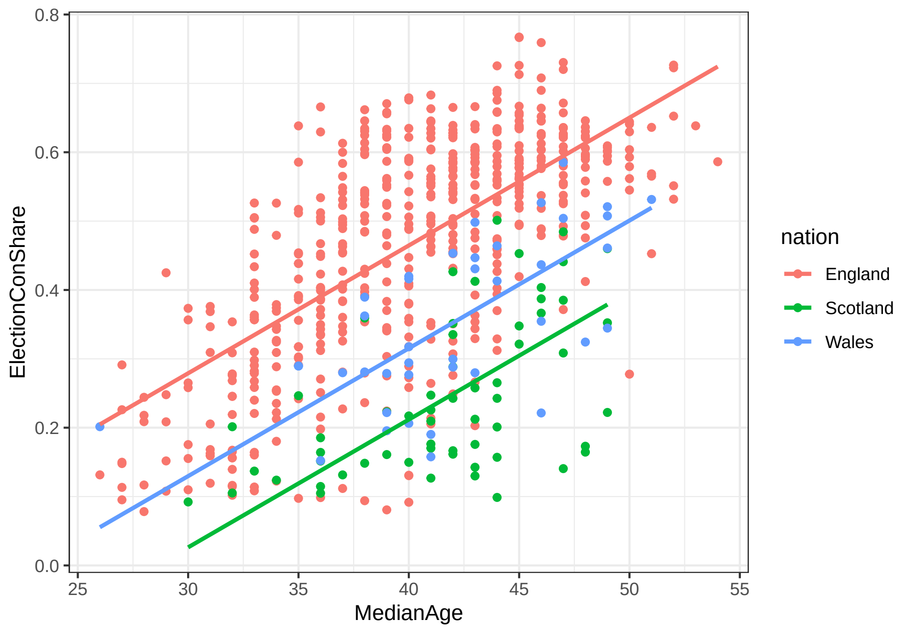


Figure 5.5: Multiple regression model to predict Tory 2019 vote share

predictors, coefficients and significance levels would usually be reported in a table.

Chapter 6

Comparing means between conditions or groups

Watch this video for an introduction to testing for differences between two means and the differences between repeated measures and independent samples:

Watch this video for an introduction to testing for differences between *more than* two means:

6.0.1 Repeated measures or independent samples

The first question always needs to be whether each participant contributes one or several data points to the dependent variable that is compared. If they contribute only one, we have an *independent samples* or *between-participants* design; if they contribute several, the design is *repeated measures* or *within participants.* In the latter case, we need to account for the relationships between some of the measurements in our analysis.

6.0.2 Two independent means

Going back to the European Social Survey 2014 data, we might be curious whether social trust differs between men and women in the UK. For that, we should always first calculate the descriptive statistics:

```r
pacman::p_load(tidyverse)
ess <- read_rds(url("http://empower-training.de/Gold/round7.RDS"))
ess <- ess %>% mutate(soctrust = (ppltrst + pplfair + pplhlp)/3)
essUK <- ess %>% filter(cntry=="GB")
essUK %>% group_by(gndr) %>% summarise(mean(soctrust, na.rm = TRUE))
```

```
## # A tibble: 2 x 2
##   gndr   `mean(soctrust, na.rm = TRUE)`
## * <fct>                           <dbl>
## 1 Male                             5.70
## 2 Female                           5.72
```

Those means look very close. Nevertheless, we might want to know how likely we would have been to see a difference this large under the null-hypothesis, i.e. if social trust did not differ between men and women. For that, we can run a t-test.

```
t.test(soctrust ~ gndr, data=essUK, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  soctrust by gndr
## t = -0.2116, df = 2248, p-value = 0.8324
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1601569  0.1289597
## sample estimates:
##   mean in group Male mean in group Female
##             5.702288             5.717886
```

This is just a special case of a linear model, so we could also use the `lm()` function:

```
lm(soctrust ~ gndr, data=essUK) %>% summary()
```

```
##
## Call:
## lm(formula = soctrust ~ gndr, data = essUK)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.7179 -1.0512  0.2821  1.2821  4.2977
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.70229    0.05450 104.623   <2e-16 ***
## gndrFemale   0.01560    0.07372   0.212    0.832
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.741 on 2248 degrees of freedom
##   (14 observations deleted due to missingness)
## Multiple R-squared:  1.992e-05,Adjusted R-squared:  -0.0004249
## F-statistic: 0.04478 on 1 and 2248 DF,  p-value: 0.8324
```

Both show, as expected, that the difference we observed could easily have been due to chance, and show the same p and t-values. You might report this by saying: there was no significant difference in social trust between men and women, $t(2248) = -0.21$, $p = .83$.

6.0.3 Two dependent means

In the ESS data, participants were asked how much they drank when they last drank during a weekday and during a weekend. Here the same participants provided two alcohol measures, so that these data points represent repeated measures. If we ignore that in the analysis, we violate a key assumption of linear models - namely the independence of observations. Therefore, we need to use a paired t-test. But as always, first descriptive statistics.

```
essUK %>%
  summarise(weekday = mean(alcwkdy, na.rm=T), weekend=mean(alcwknd, na.rm = T)) %>%
  mutate(diff = weekend - weekday)
```

```
## # A tibble: 1 x 3
##   weekday weekend  diff
##     <dbl>   <dbl> <dbl>
## 1    36.3    62.4  26.1
```

So there seems to be a large difference in the average amount people drink during a single session on weekdays and weekends. Is the difference statistically significant?

```
t.test(essUK$alcwknd, essUK$alcwkdy, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  essUK$alcwknd and essUK$alcwkdy
## t = 13.119, df = 1800, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  22.23332 30.04986
## sample estimates:
## mean of the differences
##                26.14159
```

The paired t-test - as it says in the output - tests whether the mean of the differences between the two variables is significantly different from 0. We could also specify that condition directly to receive identical results :

```
t.test(essUK$alcwknd - essUK$alcwkdy, mu = 0)
```

```
##
##  One Sample t-test
##
## data:  essUK$alcwknd - essUK$alcwkdy
## t = 13.119, df = 1800, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  22.23332 30.04986
## sample estimates:
## mean of x
##  26.14159
```

6.0.4 More than two independent means

We might be interested whether levels of life satisfaction differ between the European countries we could reach by ferry from the UK. First, let's look at the descriptive statistics.

```
essF <- ess %>% filter(cntry %in% c("FR", "ES", "IE", "BE", "NL"))
essF %>% group_by(cntry) %>% summarise(life_satisfaction = mean(stflife, na.rm = T))
```

```
## # A tibble: 5 x 2
##   cntry life_satisfaction
## * <fct>             <dbl>
## 1 BE                 7.45
## 2 ES                 6.96
## 3 FR                 6.39
## 4 IE                 6.94
## 5 NL                 7.60
```

There seem to be some differences - but are they statistically significant? Here we actually have two questions:

- do the countries together explain a significant share of the variance in life satisfaction? (*omnibus test*)
- are the levels of life satisfaction in any two countries significantly different from each other? (*pairwise comparisons*)

6.0.4.1 Omnibus test (ANOVA)

```
#Set the reference level, so that we know what the coefficients mean
essF$cntry <- factor(essF$cntry) %>% relevel(ref = "FR")
lm(stflife ~ cntry, data = essF) %>% summary()
```

```
##
## Call:
## lm(formula = stflife ~ cntry, data = essF)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6017 -0.9646  0.3983  1.3983  3.6054
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.39456    0.04612 138.663   <2e-16 ***
## cntryBE      1.05711    0.06651  15.893   <2e-16 ***
## cntryES      0.57006    0.06512   8.753   <2e-16 ***
## cntryIE      0.54832    0.06192   8.856   <2e-16 ***
## cntryNL      1.20711    0.06516  18.526   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.016 on 9896 degrees of freedom
##   (19 observations deleted due to missingness)
## Multiple R-squared:  0.04125,Adjusted R-squared:  0.04086
## F-statistic: 106.4 on 4 and 9896 DF,  p-value: < 2.2e-16
```

Here we just want to see whether the country variable explains a significant share of the variance. That is shown by the last line, so that we would report: There was an overall effects of country on life satisfaction, $F(4, 9869) = 106.4$, p<.001.

If you are so inclined, note that this is identical to a one-way ANOVA. To see that, you can replace summary() by car::Anova() - however, this is only truly essential if you are a psychologist, most other disciplines prefer the plain linear models when they suffice.

```r
pacman::p_load(car)
```

```
## Installing package into '/home/runner/work/_temp/Library'
## (as 'lib' is unspecified)

## also installing the dependencies 'matrixStats', 'RcppArmadillo', 'zip', 'numDeriv', 'SparseM',

##
## car installed
```

```r
lm(stflife ~ cntry, data = essF) %>% car::Anova()
```

```
## Anova Table (Type II tests)
##
## Response: stflife
##          Sum Sq   Df F value    Pr(>F)
## cntry       1730    4  106.45 < 2.2e-16 ***
## Residuals  40218 9896
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.0.4.2 Pairwise comparisons

Now that we know that some of the countries are different, we will want to locate the differences. That is where pairwise t-tests come in.

```
pairwise.t.test(essF$stflife, essF$cntry, p.adjust.method = "bonferroni")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  essF$stflife and essF$cntry
##
##    FR      BE      ES      IE
## BE < 2e-16 -       -       -
## ES < 2e-16 2.4e-12 -       -
## IE < 2e-16 1.0e-14 1.00    -
## NL < 2e-16 0.24    < 2e-16 < 2e-16
##
## P value adjustment method: bonferroni
```

This gives us *p*-values for all tests, that are adjusted for the fact that we are doing many (i.e. 10) comparisons and thus running a greater risk of getting a false positive. The Bonferroni adjustment, selected here, multiplies each *p*-value by the number of comparisons, unless the resulting value would exceed 1 and thus be an impossible probability.

Combining this with the descriptive statistics, we can say, for instance, that the people in the Netherlands and Belgium are more satisfied with life than those in any of the other countries, but that their satisfaction levels do not differ significantly from each other.

6.0.5 More than two means from repeated measures

Here I will revert to the simulated example from the video linked above. In that, the effect of four conditions during studying on participants' test scores was assessed, namely whether participants were exposed to instrumental music, vocal music, white noise or silence.

Note that the data for repeated measures analysis in R generally needs to be formatted in a way that each row shows one observation rather than multiple observations from one participant (“long” format). If you have data in a “wide” format, you can reshape it with the `gather()` or `pivot_longer()` functions.

To analyse whether there are differences between the conditions, as always, we start with descriptive statistics.

```
noiseData %>% group_by(condition) %>% summarise(mean(score))
```

```
## # A tibble: 4 x 2
##   condition    `mean(score)`
## * <chr>                <dbl>
## 1 instrumental          11.6
## 2 silence               13.0
## 3 vocals                10.8
## 4 whiteNoise            13.5
```

It looks like there are some differences, but to be able to judge statistical significance, we would again be interested in omnibus tests and then pairwise comparisons.

6.0.5.1 Omnibus test

Testing whether the conditions make a difference is a little bit harder with repeated measures because the observations are not independent. Therefore, we need to run a model that takes into account the relationships between the observations taken from a single participant. This does not work with `lm()`; instead we need to use an additional package that allows for multi-level modeling where some observations are clustered together, `lme4` is the most frequently used such package for this purpose.

In the case of independent samples, we were comparing our model with the group variable as a predictor implicitly to the model that predicts the overall mean for everyone (that is what the `lm()` F-test is doing). Here, the null model uses each participant’s own overall mean as the prediction for their performance in any one condition. We need to set up that null model explicitly and then compare it to the model that considers groups.

```
pacman::p_load(lme4)

#Set reference level explicitly
noiseData$condition <- noiseData$condition %>% factor() %>% relevel("silence")

#Run null model - predicting only an individual intercept per participant
model0 <- lmer((score ~ (1 | participantID)),        data = noiseData)

#Run hypothesized model - adding the groups as a predictor
model1 <- lmer((score ~ condition + (1 | participantID)),        data = noiseData)
```

```
#Comparing the two models
anova(model0, model1)
```

```
## refitting model(s) with ML (instead of REML)
```

```
## Data: noiseData
## Models:
## model0: score ~ (1 | participantID)
## model1: score ~ condition + (1 | participantID)
##        npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## model0    3 480.29 488.10 -237.14   474.29
## model1    6 447.31 462.94 -217.66   435.31 38.978  3  1.754e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we can see that the hypothesised model showed a significantly better fit. This is tested with a χ^2-test, which we will look at further later in the course, but the key criterion again is that p (here Pr(>Chisq)) is smaller than .05.

6.0.5.2 Pairwise comparisons

Now that we know that there is a difference between some of the conditions, we will want to know which are different. For that, we can again run pairwise t-tests; we just need to specify that they are run on paired data by setting `paired = TRUE`.

```
pairwise.t.test(noiseData$score, noiseData$condition,        p.adj = "bonferroni", pai
```

```
##
##  Pairwise comparisons using paired t tests
##
## data:  noiseData$score and noiseData$condition
##
##              silence instrumental vocals
## instrumental 0.045   -            -
## vocals       8.2e-06 0.423        -
## whiteNoise   1.000   0.001        2.0e-05
##
## P value adjustment method: bonferroni
```

This indicates that the scores of participants in the white noise and silence conditions were not significantly different from each other, nor were those between vocals and instrumental music. Only the four comparisons between ‘music’ and ‘non-music’ conditions were significant.

Chapter 7

Interaction terms in linear models

Watch this video for an introduction to interaction terms in linear models:

The section below essentially contains the code needed for the examples in the video, with some further annotations. Before that, just one note on terminology:

- Generally, if two variables interact, then the effect that one has on the predictor (the regression slope) depends on the value of the other. Mathematically, that relationship is symetrical - if A interacts with B, then B interacts with A.
- For interpretation, we often designate one of the variables as the *moderator*. That just means that we are primarily interested in the effect of the other variable on the outcome and in how the moderator changes that effect. Often, demographic variables such as age or gender serve as moderators.

7.1 Example 1: Memory and chess

The first example draws on research by Gobet & Simon (1996) but uses simulated data.

Show the code to simulate the data

```r
library(tidyverse)

#Generate data (errors committed) - *roughly* based on Gobet & Simon 1996
set.seed(300688) #for reproducible results
ER <- rnorm(50, 4.9,3.5) + rnorm(50, 0, 2)
NR <- rnorm(50, 15.7, 4.0) + rnorm(50, 0, 2)
```

```r
EF <- rnorm(50, 21.4, 5) + rnorm(50, 0, 2)
NF <- rnorm(50, 21.8, 5) + rnorm(50, 0, 2)

obs <- data.frame(player = "expert", type = "real", errors = ER) %>%
  rbind(data.frame(player = "novice", type = "real", errors = NR)) %>%
  rbind(data.frame(player = "expert", type = "fake", errors = EF)) %>%
  rbind(data.frame(player = "novice", type = "fake", errors = NF)) %>%
  mutate(type=factor(type),
         player = factor(player, levels = c("novice", "expert")))

#Adding the centrally mirrored condition
EM <- rnorm(50, 7.8, 3.5) + rnorm(50, 0, 2)
NM <- rnorm(50, 18, 3.5) + rnorm(50, 0, 2)

obs2 <- data.frame(player = "expert", type = "real", errors = ER) %>%
  rbind(data.frame(player = "novice", type = "real", errors = NR)) %>%
  rbind(data.frame(player = "expert", type = "fake", errors = EF)) %>%
  rbind(data.frame(player = "novice", type = "fake", errors = NF)) %>%
  rbind(data.frame(player = "expert", type = "mirrored", errors = EM)) %>%
  rbind(data.frame(player = "novice", type = "mirrored", errors = NM)) %>%
  mutate(type=factor(type),
         player = factor(player, levels = c("novice", "expert")))
```

The first model tests whether player level and position type interact. That is the case, based on the very small p-value of the interaction term. Then we use a plot to understand the nature of that interaction further - because the predictor is categorical, we use the `cat_plot()` function.

```r
pacman::p_load(tidyverse)
mod <- lm(errors ~ player + type + player:type, obs)
summary(mod)
```

```
##
## Call:
## lm(formula = errors ~ player + type + player:type, data = obs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.8157  -3.6637  -0.0812   3.6591  14.2213
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            21.5744     0.7398  29.164  < 2e-16 ***
## playerexpert           -1.2240     1.0462  -1.170    0.243
## typereal               -6.5092     1.0462  -6.222 2.91e-09 ***
## playerexpert:typereal  -8.0039     1.4795  -5.410 1.83e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.231 on 196 degrees of freedom
## Multiple R-squared:  0.5892,Adjusted R-squared:  0.5829
## F-statistic: 93.69 on 3 and 196 DF,  p-value: < 2.2e-16
```

```
pacman::p_load(interactions)
cat_plot(mod, pred="player", modx = "type", geom="line")
```

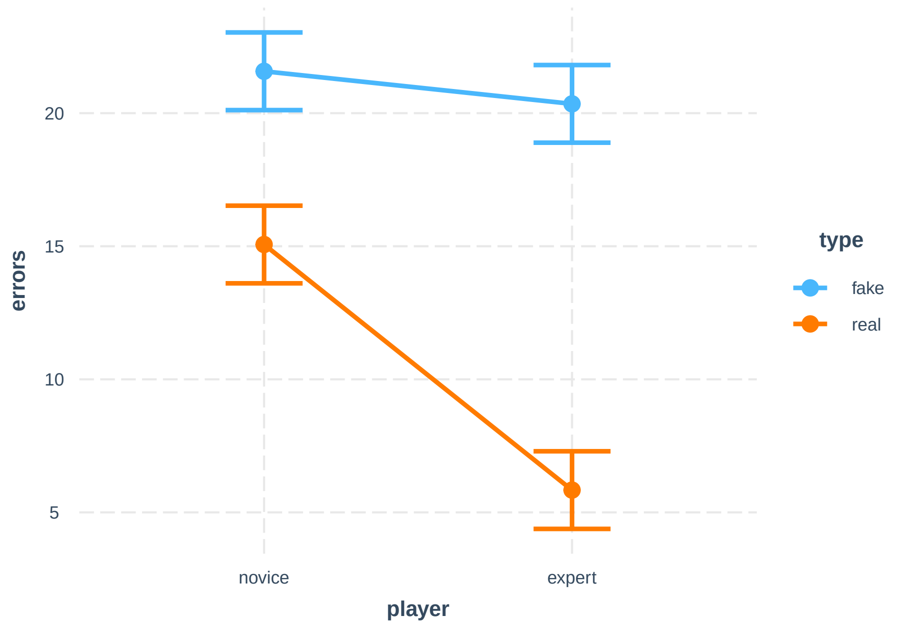


Figure 7.1: Simple interaction plot - note that lines are not parallel

Next we consider a third condition - chess positions that are neither quite real nor entirely fake, but positions that are mirrored. With that, we get multiple dummy interaction terms. To test whether they are collectively significant, we need to use the `Anova()` function from the `car` package.

```
mod <- lm(errors ~ player + type + player:type, obs2)
summary(mod)
```

```
##
## Call:
## lm(formula = errors ~ player + type + player:type, data = obs2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.8157  -3.3006  -0.0503   3.4305  14.2213
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               21.5744     0.6881  31.354  < 2e-16 ***
## playerexpert              -1.2240     0.9731  -1.258    0.209
## typemirrored              -4.6796     0.9731  -4.809 2.43e-06 ***
## typereal                  -6.5092     0.9731  -6.689 1.13e-10 ***
## playerexpert:typemirrored  -8.3088     1.3762  -6.038 4.71e-09 ***
## playerexpert:typereal     -8.0039     1.3762  -5.816 1.57e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.866 on 294 degrees of freedom
## Multiple R-squared:  0.6085,Adjusted R-squared:  0.6018
## F-statistic: 91.38 on 5 and 294 DF,  p-value: < 2.2e-16
```

```r
pacman::p_load(car)
car::Anova(mod, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: errors
##              Sum Sq  Df  F value    Pr(>F)
## (Intercept) 23272.7   1 983.0610 < 2.2e-16 ***
## player         37.5   1   1.5821    0.2095
## type         1126.9   2  23.8013 2.627e-10 ***
## player:type  1109.9   2  23.4420 3.580e-10 ***
## Residuals    6960.1 294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

An interaction plot can then help again to understand what is going on. Here it shows that two of the three conditions are very similar.

```r
cat_plot(mod, pred="player", modx = "type", geom="line")
```

7.2 Example 2: link between obesity and negative emotions in the European Social Survey

In this example, I considered the link between obesity and negative emotions in Germany in the European Social Survey 2014. (Note that this relationship does not appear in the UK, which indicates that it should probably be treated as an interesting observation rather than a likely general relationship for now.)

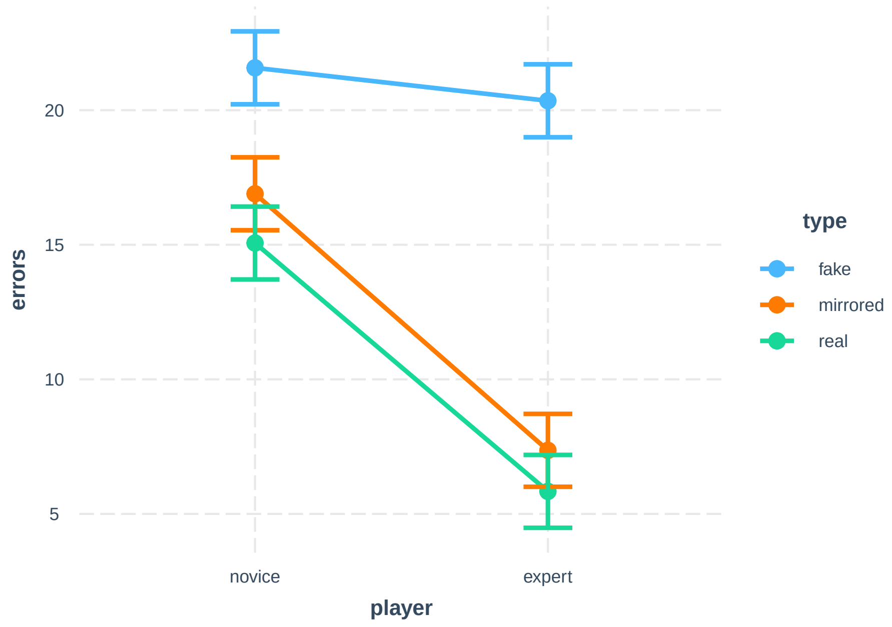


Figure 7.2: Simple interaction plot - now 2 lines are parallel

Click here to see the code that loads and prepares the data

```
ess <- read_rds(url("http://empower-training.de/Gold/round7.RDS"))

pacman::p_load(psych) #Used to create a scale
```

```
## Installing package into '/home/runner/work/_temp/Library'
## (as 'lib' is unspecified)

## also installing the dependencies 'tmvnsim', 'mnormt'

##
## psych installed
```

```
a <- ess %>% select(cldgng, fltsd, enjlf, fltlnl, wrhpp, slprl, flteeff, fltdpr) %>%
  haven::zap_labels() %>%
  #This is sometimes needed when SPSS files have been imported with haven
  #and errors related to data classes appear.
  psych::alpha(check.keys = TRUE)

ess$depr <- a$scores

ess$bmi <- ess$weight/((ess$height/100)^2)
```

```
#Filter for Germans with realistic BMI who are not underweight
#and reported their gender
essDE <- ess %>% filter(bmi < 60, bmi>=19, gndr != "No answer", cntry=="DE")

#Data prep for example 3 - strip unnecessary labels
ess$stflife <- haven::zap_labels(ess$stflife)
```

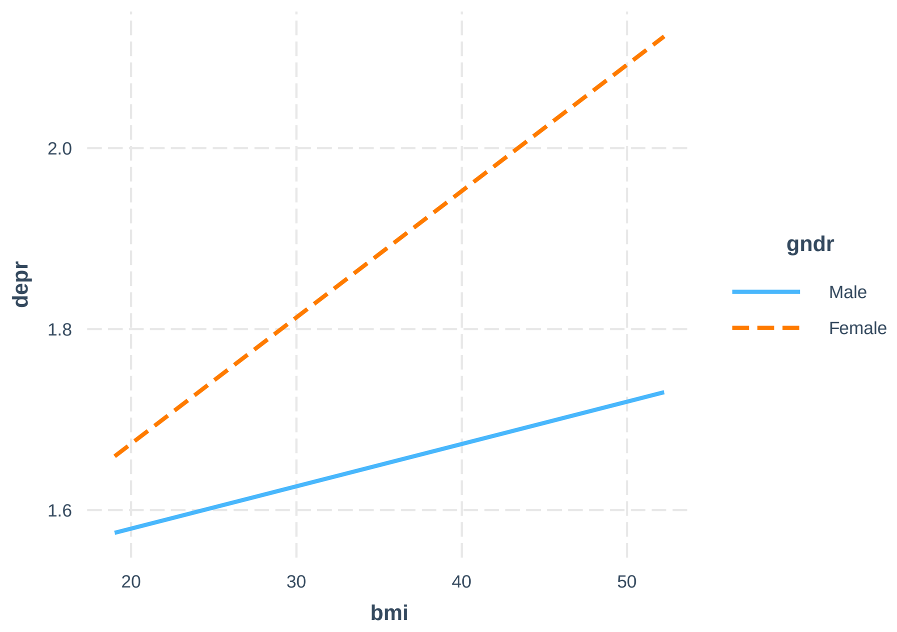
The `lm()` output below shows that there is a significant interaction between gender and BMI, with women having a stronger relationship between BMI and the frequency of experiencing negative emotions. This is again shown in an interaction plot - as the predictor variable is continuous, we now use the `interact_plot()` function.

```
pacman::p_load(interactions)
mod <- lm(depr ~ bmi + gndr + bmi:gndr, essDE)
summary(mod)
```

```
##
## Call:
## lm(formula = depr ~ bmi + gndr + bmi:gndr, data = essDE)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.85559 -0.31936 -0.09363  0.25624  2.00783
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.486028   0.072229  20.574   <2e-16 ***
## bmi              0.004674   0.002667   1.753   0.0797 .
## gndrFemale      -0.091874   0.096134  -0.956   0.3393
## bmi:gndrFemale   0.009284   0.003609   2.573   0.0101 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.438 on 2860 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.037,Adjusted R-squared:  0.03599
## F-statistic: 36.63 on 3 and 2860 DF,  p-value: < 2.2e-16
```

```
interact_plot(mod, pred="bmi", modx = "gndr")
```

\begin{figure}



{

}

\caption{interact_plot shows different slope for men and women} \end{figure}

Note that `ggplot2` automatically includes an interaction when fitting regression lines as soon as a categorical variable is mapped to the `colour` (or `linetype`) aesthetic - so in simple cases, `geom_smooth(method = "lm")` can be used as an alternative to `interact_plot()`

```
ggplot(essDE, aes(x=bmi, y=depr, colour=gndr)) + geom_smooth(method="lm", se=FALSE)
```

```
## Warning: Removed 2 rows containing non-finite values (stat_smooth).
```

7.3 Example 3: link between working hours, income and life satisfaction

This example considers whether working hours (`wkhtot`) affect the link between income (`hinctnta`) and life satisfaction - i.e. does working very long hours make income less valuable? This time, the effect appeared in the UK data in the 2014 European Social Survey - again, the question might be whether that is just an incident of spurious data mining, or whether it reveals a broader relationship.

In any case, let's have a look at the interaction. Note that `*` in the `lm()` formula is an abbreviation for `+` and `:`, so that the command below could also

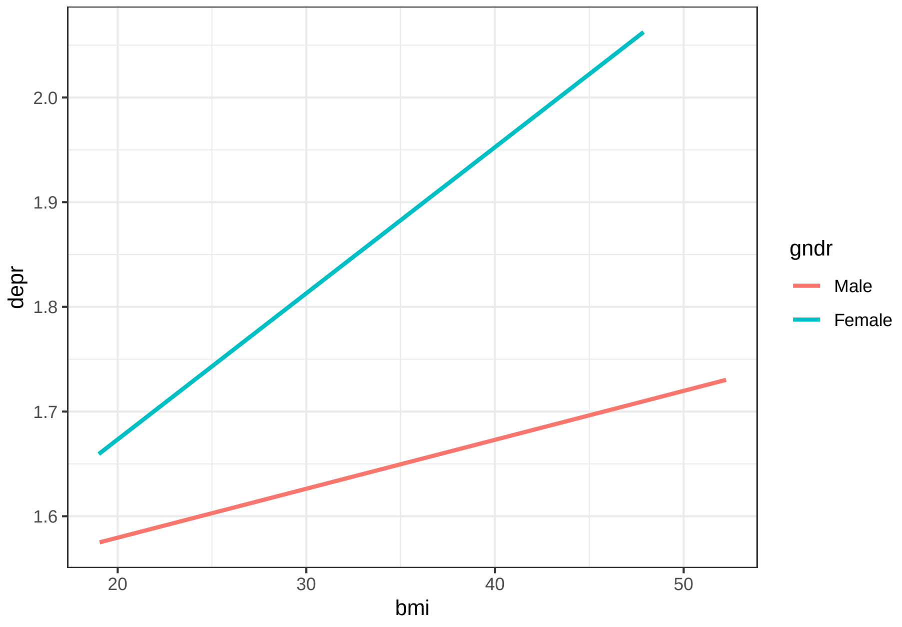


Figure 7.3: ggplot automatically includes interaction

be written as `lm(stflife ~ wkhtot + hinctnta + wkhtot:hinctnta)`

```
essUK <- filter(ess, cntry == "GB")
mod <- lm(stflife ~ wkhtot*hinctnta, essUK)
summary(mod)
```

```
##
## Call:
## lm(formula = stflife ~ wkhtot * hinctnta, data = essUK)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9013 -0.9305  0.3023  1.3331  3.6527
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       6.085338   0.242722  25.071  < 2e-16 ***
## wkhtot            0.008065   0.006235   1.293   0.1960
## hinctnta          0.238197   0.043684   5.453 5.64e-08 ***
## wkhtot:hinctnta -0.002129   0.001055  -2.019   0.0437 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.023 on 1810 degrees of freedom
##   (450 observations deleted due to missingness)
## Multiple R-squared:  0.05154,Adjusted R-squared:  0.04997
## F-statistic: 32.79 on 3 and 1810 DF,  p-value: < 2.2e-16
```

We can see that while income positively predicts life satisfaction, this effects appears to be weaker when both income and working hours are high. However, we need to be careful with interpretation here - technically, the coefficients for income and working hours now reflect the impact of that variable when the other variable is 0, and are thus unlikely to be meaningful. Therefore, it is better to look at the interaction plot and at simple slopes analyses. Both can now be done in two ways, depending on which variable we see as the primary predictor/variable of interest.

```
interact_plot(mod, pred="hinctnta", modx = "wkhtot")
```

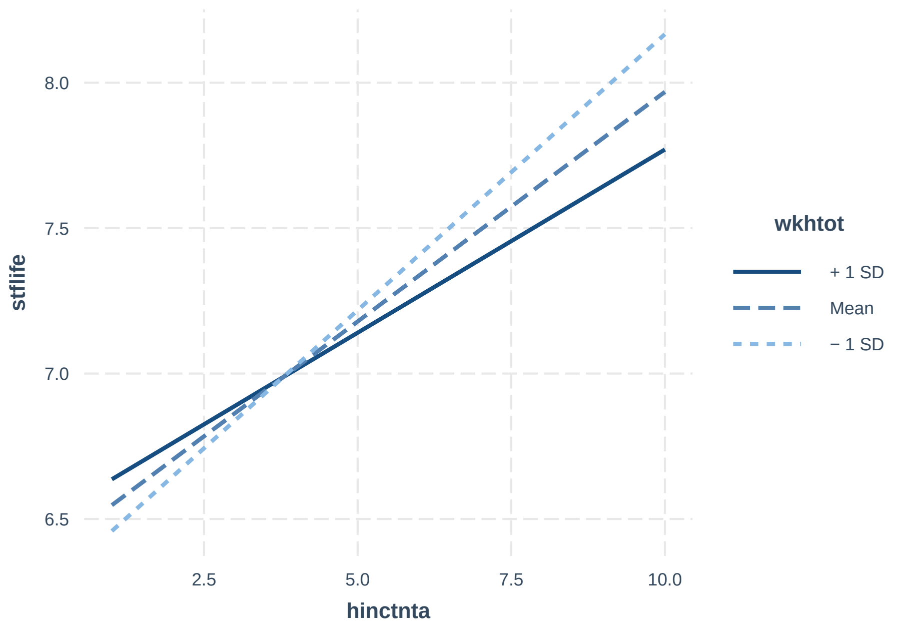


Figure 7.4: Option A: strong positive effect of income, tempered by working hours

```
interact_plot(mod, pred="wkhtot", modx = "hinctnta")
```

Simple slopes analyses (`sim_slopes()`) offer similar information together with significance tests and thus help to decide which of the slopes should be interpreted. They contain the Johnson-Neyman interval, which is the range of values of the moderator for which the predictor has a significant effect on the outcome.

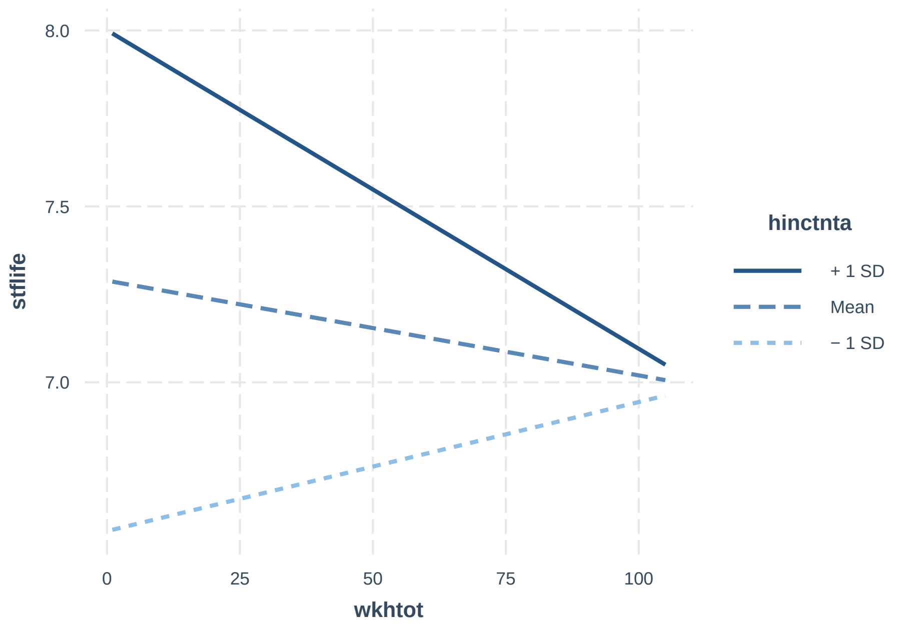


Figure 7.5: Option B: very different effects of working hours, depending on income

```r
pacman::p_load(interactions)
sim_slopes(mod, pred="hinctnta", modx = "wkhtot")
```

```
## JOHNSON-NEYMAN INTERVAL
##
## When wkhtot is OUTSIDE the interval [74.17, 2622.20], the slope of hinctnta
## is p < .05.
##
## Note: The range of observed values of wkhtot is [1.00, 105.00]
##
## SIMPLE SLOPES ANALYSIS
##
## Slope of hinctnta when wkhtot = 22.73 (- 1 SD):
##
##   Est.   S.E.   t val.      p
## ------ ------ -------- ------
##   0.19   0.02     8.19   0.00
##
## Slope of hinctnta when wkhtot = 37.72 (Mean):
##
##   Est.   S.E.   t val.      p
## ------ ------ -------- ------
##   0.16   0.02     9.76   0.00
##
## Slope of hinctnta when wkhtot = 52.71 (+ 1 SD):
##
##   Est.   S.E.   t val.      p
## ------ ------ -------- ------
##   0.13   0.02     5.71   0.00
```

```
sim_slopes(mod, pred="wkhtot", modx = "hinctnta")
```

```
## JOHNSON-NEYMAN INTERVAL
##
## When hinctnta is OUTSIDE the interval [-43.03, 7.73], the slope of wkhtot
## is p < .05.
##
## Note: The range of observed values of hinctnta is [1.00, 10.00]
##
## SIMPLE SLOPES ANALYSIS
##
## Slope of wkhtot when hinctnta = 2.07 (- 1 SD):
##
##   Est.   S.E.   t val.      p
## ------ ------ -------- ------
##   0.00   0.00     0.81   0.42
##
## Slope of wkhtot when hinctnta = 5.05 (Mean):
##
##    Est.   S.E.   t val.      p
## ------- ------ -------- ------
##   -0.00   0.00    -0.84   0.40
##
## Slope of wkhtot when hinctnta = 8.04 (+ 1 SD):
##
##    Est.   S.E.   t val.      p
## ------- ------ -------- ------
##   -0.01   0.00    -2.01   0.04
```

Finally, the `johnson_neyman()` function creates a plot showing the slope of one variable depending on the value of the other, while highlighting the regions of significance. This can help with an understanding of the relationship, but is not (yet?) widely used.

```
johnson_neyman(mod, pred="wkhtot", modx = "hinctnta")
```

```
## JOHNSON-NEYMAN INTERVAL
##
```

```
## When hinctnta is OUTSIDE the interval [-43.03, 7.73], the slope of wkhtot
## is p < .05.
##
## Note: The range of observed values of hinctnta is [1.00, 10.00]
```

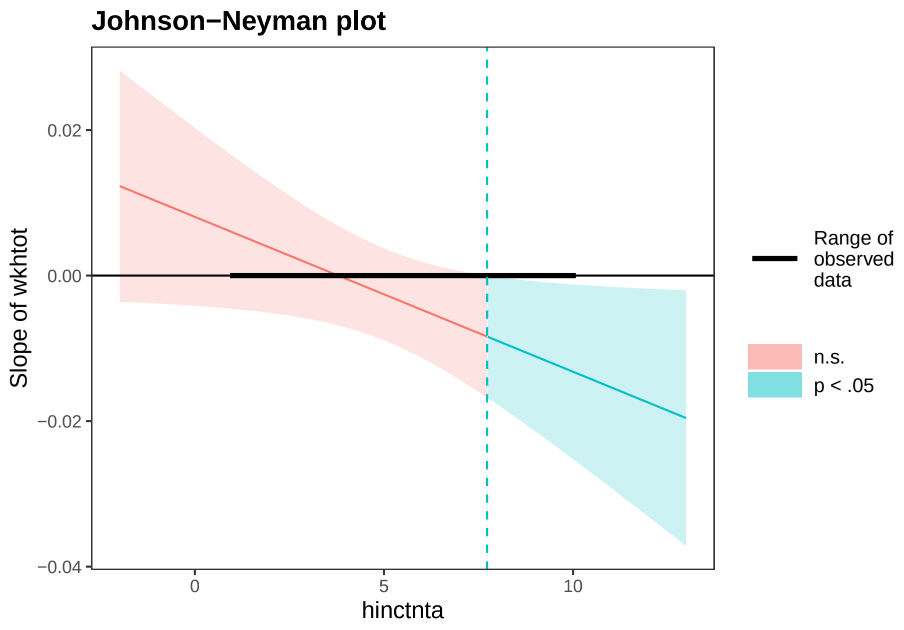


Figure 7.6: Johnson-Neyman plot shows regions of significance

Chapter 8

Chi-squared tests for associations between categorical variables

Watch this video for an introduction to this topic:

When we want to see whether two categorical variables are associated with each other, typical linear models won't work. Instead, we need a new way to decide whether a difference between the data we observe and that which we expect under the null-hypothesis is unlikely to occur due to chance. But before we get to that, we should look at the data.

8.1 Tables of frequencies and proportions

With two categorical variables, the data we observe can best be shown in a frequency table - i.e. a table that shows how many observations fall into each combination of categories of the two variables. This is created with the `table()` function. Let's look an example of the association between political parties and their winning candidates' gender in the 2019 UK general election. (Note that this is based on the official statistics published by the election authorities, who registered candidate gender as a binary variable.)

```r
table(constituencies$ElectionWon, constituencies$WinnerGender)
```

```
##
##       Female Male
##   Con     87  278
##   Lab    104   98
##   LD       7    4
##   SNP     16   32
```

If there is an association between the two variables, then the proportional distribution will differ between the rows (i.e. one party would have a higher *share* of women than another party). To see whether that is the case, proportion tables that show shares are helpful. These can be created from the `table()` output with the `prop.table()` function. However, there are three options for proportions: are we interested in the share of each cell as a part of the total number of candidates? Or as a share of all winning candidates of a gender (i.e what share of female winning candidates are from the Conservative party)? Or rather the shares of women and men as a share of the total winning candidates of each party? In this case the latter seems most relevant and interpretable.

```r
x <- table(constituencies$ElectionWon, constituencies$WinnerGender)
#prop.table(x) - share of each cell as part of the total
#prop.table(x, margin = 2) - share of each cell as part of the column
prop.table(x, margin = 1) %>% #Share of each cell as part of the row
   round(2) #Round to 2 decimal places
```

```
##
##        Female Male
##   Con    0.24 0.76
##   Lab    0.51 0.49
##   LD     0.64 0.36
##   SNP    0.33 0.67
```

8.2 The null hypothesis and the χ^2 statistic

To be able to test whether any differences between the rows or between the columns that we observe in a frequency table are statistically significant, we need to be clear on what the null hypothesis is. If two variables are statistically independent, then their distribution needs to be the same across each row and along each column. This does not mean that all cells need to have the same value, In our case, given that there are many more male than female election winners and many more Conservative MPs than MPs of any other party, we would expect the male Conservatives cell to have the highest number of observations if there was no relationship between party and winning candidates' gender, i.e. if the null hypothesis was true.

The expected number of observations in each cell under the null hypothesis can be calculated as the share of its entire row of the total observations (e.g., the total share of female candidates) * the share of its entire column of the total (e.g., the total share of Labour candidates) * the total number of observations. Based on that logic, the expected number of observations per cell would be the following.

```
##                            constituencies$WinnerGender
## constituencies$ElectionWon Female Male
##                        Con    125  240
##                        Lab     69  133
##                        LD       4    7
##                        SNP     16   32
```

Once we know what to expect under the null-hypothesis, we need a way to see how far our observed data diverges from that. The χ^2 (chi-squared) test statistic provides a way of measuring that distance in a way that is independent of our sample size and therefore comparable across studies. It is calculated as follows, where O is the *observed* number of cases per cell of the frequency table and E is the *expected* number under the null-hypothesis:

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

In R, easiest way to calculate it and test whether the distance of the observed distribution from the null distribution is statistically significant is to use the `chisq.test()` function.

8.3 The chisq.test() function

As a simple fictional example, we might be interested in whether Scottish and British people differ in their preference for tea versus coffee. If we ask 100 people, we might observe the following distribution:

```
table(prefData$nationality, prefData$preference)
```

```
##
##           Coffee Tea
##   English     45  95
##   Scottish    30  30
```

To calculate the distance from distribution expected if there was no association between nationality and tea preference, we use the chisq.test() function. That also gives us an associated p-value.

```
chisq.test(prefData$nationality, prefData$preference, correct = FALSE)
```

```
##
##  Pearson's Chi-squared test
##
```

```
## data:  prefData$nationality and prefData$preference
## X-squared = 5.7143, df = 1, p-value = 0.01683
```

In this case, we would conclude that there is a statistically significant difference, with English people preferring tea more frequently than Scottish people do, $\chi^2(1) = 4.98$, $p = .016$.

8.3.1 Dealing with small samples

When testing the significance of χ^2-values, the calculated values are compared to a continuous distribution. When you have a small sample, however, your possible χ^2-values cannot be continous; rather, a single observation shifting from one cell to another would be associated with a substantial jump in the χ^2 statistic. This can make the normal p-values far too low in such cases, and therefore lead to Type I errors (false positives).

It is often recommended that χ^2 should not be used with samples where the expected frequency in more than 20% of the cells is less than 5. In a 2x2 table (i.e when each of the two variables has only two possible categories), the Yates correction can be used to reduce the χ^2-value and therefore make the test more conservative (R does that by default in `chisq.test()`, but it can be turned off by setting `correct = FALSE`). However, this sometimes goes too far, so my recommendation is to simulate p-values for any samples where you have cells with low expected frequencies (say less than 20 expected cases in the smallest cell). You can do that by setting `simulate.p.value = TRUE` in the function call. Note that like any simulation, this is based on random number generation, so that `set.seed()` should be used with a fixed number of your choice to ensure that the results can be reproduced.

If we were to rerun the example above with a smaller sample but more extreme differences, we can compare the three possible methods of calculating χ^2. To see how small the smallest expected value is, we can look at the `expected` element of the output of the chisq.test() function.

```r
table(prefDataSmall$nationality, prefDataSmall$preference)
```

```
##
##            Coffee Tea
##   English      12  30
##   Scottish     10   8
```

```r
chisq.test(prefDataSmall$nationality, prefDataSmall$preference, correct = FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  prefDataSmall$nationality and prefDataSmall$preference
## X-squared = 3.9508, df = 1, p-value = 0.04685
```

It looks like the difference is significant. However, we should consider how small the smallest expected cell count is, to decide whether we need to adjust our way of testing significance. For that, we can look at the `expected` element of the output of the chisq.test() function.

```r
chisq.test(prefDataSmall$nationality, prefDataSmall$preference, correct = F)$expected
```

```
##                           prefDataSmall$preference
## prefDataSmall$nationality Coffee  Tea
##                  English    15.4 26.6
##                  Scottish    6.6 11.4
```

With fewer than 7 expected cases of Scottish coffee drinkers, the standard χ^2 significance test is not reliable. Therefore, we either need to use the correction or simulation. Given that the simulation is based on random numbers, we should use the `set.seed()` function to initialise the random number generator - that makes sure that the results are reproducible.

```r
set.seed(300688)
chisq.test(prefDataSmall$nationality, prefDataSmall$preference, correct = TRUE)
chisq.test(prefDataSmall$nationality, prefDataSmall$preference, simulate.p.value = TRUE)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  prefDataSmall$nationality and prefDataSmall$preference
## X-squared = 2.8742, df = 1, p-value = 0.09001
##
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  prefDataSmall$nationality and prefDataSmall$preference
## X-squared = 3.9508, df = NA, p-value = 0.07496
```

These other two methods agree that caution is needed. Usually, the simulated p-value will be a bit lower and a bit more accurate than the result of the Yates' continuity correction, so I would report that the data shows a trend towards a greater preference for tea among English respondents that falls short of the conventional standard of statistical significance, with $\chi^2 = 3.95$, $p = .075$ (based on 2000 Monte Carlo simulations).

8.4 Post-hoc tests

χ^2-tests can be used to test for an association between variables with many different levels, such as music preferences or political parties. In that case, a significant result only tells us that at least one of the cell counts is significantly different from the expectation under the null-hypothesis. To get more details, post-hoc tests would be needed. To go back to the example at the start, let's test whether the winning candidates' gender differed significantly between the major UK parties in the 2019 General Election. It certainly looks that way from the proportions table, and the `chisq.test()` agrees

```
prop.table(x, margin = 1) %>% #Share of each cell as part of the row
   round(2) #Round to 2 decimal places
```

```
##
##       Female Male
##   Con   0.24 0.76
##   Lab   0.51 0.49
##   LD    0.64 0.36
##   SNP   0.33 0.67
```

```
chisq.test(constituencies$ElectionWon, constituencies$WinnerGender,
           simulate.p.value = TRUE)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  constituencies$ElectionWon and constituencies$WinnerGender
## X-squared = 48.504, df = NA, p-value = 0.0004998
```

However, this does not allow us to answer questions about any specific parties. For that, we need post-hoc tests. The most common tests compare each cell to its expected value under the null-hypothesis, but other tests are possible. For instance, you might be more interested to compare parties to each other rather than to this expected (average) value. For that, you would need to filter your data and run multiple chisq.tests.

To compare each cell to the expected value, we can use the `chisq.posthoc.test` package. This package contains the `chisq.posthoc.test()` function that takes a table as the input and computes standardised residuals (i.e. distances) and *p*-values for each cell. To avoid an inflated error rate, it can correct for multiple comparisons; by default the Bonferroni correction is used.

```
pacman::p_load(chisq.posthoc.test)
table(constituencies$ElectionWon, constituencies$WinnerGender) %>% chisq.posthoc.test(s
```

```
##   Dimension     Value     Female       Male
## 1       Con Residuals -6.4559388  6.4559388
## 2       Con  p values  0.0000000  0.0000000
## 3       Lab Residuals  6.2985557 -6.2985557
## 4       Lab  p values  0.0000000  0.0000000
## 5        LD Residuals  2.0776181 -2.0776181
## 6        LD  p values  0.3019560  0.3019560
## 7       SNP Residuals -0.1295052  0.1295052
## 8       SNP  p values  1.0000000  1.0000000
```

In this table, the *p*-values can tell us which party/gender combinations differ significantly from the expectation under the null-hypothesis, while the sign of the residuals indicates the direction of the effect. Thus we can see that the Conservatives had significantly fewer male winning candidates while Labour had significantly more.

A note on statistical power and the importance of thinking about corrections: This approach to running post-hoc tests runs one test per cell. In this situation with 2 genders and 4 parties, we have 8 cells, so that the Bonferroni correction multiplies each *p*-value by 8, which makes it relatively hard to detect effects. However, as should be clear from the table, in cases where one variable has only two levels, half of the tests are redundant. If we know that the Conservatives have significantly fewer female winners, it follows necessarily that they have significantly more male winners. *p*-values in each row are identical, and the residuals are symmetric. Therefore, we should only count half of the tests, and multiply p-values by 4 rather than 8. The general point is that it is essential to check how many tests a post-hoc functions counts in adjusting *p*-values by comparing the *p*-values reported when the adjustment parameter is set to "none" to those reported with "bonferroni" correction. Then make sure that you understand where that number comes from and that it makes sense in your situation.

8.5 Effect sizes

As always, finding that the variables are related to a statistically significant degree is only the first step, and you also want to report the strength of the relationship. Clear proportion tables are essential here, and then there are two options to sum up the strength of relationships into a single number: **Cramer's V** as a standardised measure of the strength of an association between two categorical variables, and the **Odds Ratio** of being in one cell rather than another.

Cramer's V describes a full table, and is scaled from 0 (no relationship) to 1 (one variable is entirely predictable by the other). It is often recommended that one should be weary of values greater than .5, as that suggests that the two variables might be redundant. Many packages offer functions to calculate Cramer's V, including the `effectsize` package that allows us to calculate it based on the χ^2-value, the sample size, and the number of levels of each variable.

```r
pacman::p_load(effectsize)
chisq_to_cramers_v(chisq = 48.5, n = 626, nrow=4, ncol=2)
```

```
## Cramer's V |       95% CI
## -------------------------
## 0.28       | [0.19, 0.35]
```

While this is a helpful statistic to compare effect strengths across multiple tests, it does not have an intuitive interpretation. Here, Odds Ratios might help. They can only be calculated for 2x2 tables, for instance for the fictional example regarding tea and coffee preferences.

```
##
##             Coffee Tea
##   English       45  95
##   Scottish      30  30
```

Here an **Odds Ratio** would state how much more likely it is to get a tea afficionado if we ask an English rather than a Scottish person. Odds are the ratio of the frequency of specified outcomes over the frequency of other outcomes. For instance, the odds of it being Sunday on a random day are 1 to 6. In this case, the odds for getting a tea afficionado when picking out an English person would be 95:45, while they would be 30:30 for a Scottish person. Thus the odds ratio would be

$$OR = \frac{\frac{95}{45}}{\frac{30}{30}} = 2.1$$

so that I would be

- 2.1x more likely to get a tea afficionado rather than a coffee afficionado when picking out an English person from this sample than a Scottish person,
- 2.1x more likely to get a coffee afficionado rather than a tea afficionado when picking out a Scottish person rather than an English person from this sample,
- 2.1x more likely to have an English person in front of me when I encounter a tea afficionado from this sample,
- and 2.1x more likely to have a Scottish person in front of me when I encounter a coffee afficionado from this sample.

All of these statements should be easier to interpret than a Cramer's V index of .17 (which happens to be the value of this sample). Of course you should pick which one of the ways of framing the relationship is the best answer to the question at hand, rather than reporting the same information multiple times.

Chapter 9

Reporting the results of statistical tests

Statistical test can often result in a lot of different numbers, but should also result in clear decisions and messages. To facilitate communication, there are sets of guidelines regarding what to report. In this course, we are using the APA (American Psychological Association) guidelines, which are also commonly used in journals in psychology and beyond.

9.1 Essential pieces of information

When reporting inferential statistical tests, always make sure to:

- State whether the results were **significant** (or which results were), and what the exact *p*-values were (typically rounded to three decimal places; if smaller than .001 then reported as $< .001$). Results with *p*-values between .05 and .1 can be reported as a trend, or as marginally significant, but that should (usually) only be done if the estimated effect size is substantial.
- Report **which test** you carried out, and how it was carried out, if you did anything non-standard (e.g., if *p*-values were simulated). For each test, relevant degrees of freedom and test statistics should be reported, see below.
- Include relevant **descriptive statistics**. If a difference is significant, the reader will want to know how large it was.
- Consider including **effect sizes**, ideally with confidence intervals, so that readers can judge how important the effect is. In reality, there will often be some statistically significant differences between groups, but not all are of a size that makes them interesting or practically significant.

9.1.1 What to include for specific tests

- ***t*-tests:** $t(\text{df}) = 1.23$, $p = .012$. df stands for degrees of freedom and is shown in the `t.test()`-output. If equal variances are assumed, it is the sample size - 1; otherwise, typically a bit less. In any case, it describes which *t*-distribution was used to find the *p*-value.
- ***F*-tests** (for ANOVAs and model evaluation): $F(\text{df1, df2}) = 1.23$, $p = .012$. Here you have two degrees of freedom, df1 for the model and df2 for the error term. Both are provided in the output of various functions; a simple heuristic for checking whether you got them the right way round is that df2 should be the larger one in the vast majority of cases.
- **Correlations:** $r = .12$, $p = .012$. Given that correlations are bounded by -1 and 1, the leading 0 is always ommitted. If you use Spearman's rank order correlation, it is reported as *rho* = .12; the Greek letter ρ is not typically used.
- **χ^2-tests:** $\chi^2(\text{df}) = 1.23$, $p = .012$. Here the degrees of freedom are the number of frequency table rows - 1 * the number of columns - 1. When you `simulate.p.values`, R does not report it because it does not use a fixed χ^2-distribution to look up the *p*-values; you should report it nonetheless as it also gives an indication for the complexity of your hypothesis.

Examples can be found all over the internet; this set of notes contains detailed instructions for most statistical tests.

9.1.2 Formatting

- Any **Roman letters** (*p, r, F, t, R^2, M*) that stand for specific statistics are printed in *italics*
- If a statistic cannot have a value higher than 1 (i.e. for *p, r, rho, R^2*) the **leading 0** should be omitted, i.e. $p = .04$ rather than $p = 0.04$
- If you want to use **Greek letters** (e.g., $\alpha, \beta, \chi, \Delta$) in RMarkdown, the easiest way to get them is to create an inline formula (an expression wrapped into dollar signs) and then type the name of the letter after a \, for instance `$\alpha$` You can use the same approach to show sums, fractions etc., though you would generally not need that in this course. The easiest way to get special characters like Greek letters into Word documents is to Google and copy-paste them.
- To insert a **superscript**, most often ^2, into RMarkdown, type ^2. In Word, there is a superscript button close to the font size selection box.
- When it comes to **descriptive statistics**, *N* typically denotes the number of cases/participants, *M* the mean, and *SD* the standard deviation

9.1.3 Rounding

It is important to round numbers consistently, partly because too many decimal places might convey a false impression of accuracy, but also because numbers that do not convey relevant information can quickly clutter up papers and presentations - or does it help if I report that I spend 2.978536776 hours working on this book today?

In general, test statistics and most descriptive statistics should be rounded to two decimal places (i.e. places after the decimal point) and *p*-values to three decimal places. If that leads to descriptive statistics or regression coefficients to be rounded down to 0.00, that is a strong indication that you should consider changing the scale of the variables (e.g., from hours to minutes). *p*-values by definition can never be equal to 0, so here you need to pay attention to report small *p*-values as $< .001$ rather than $= .000$

How to round? (With one change to what you learned in school) Cross out all numbers after the last decimal place you want to keep. If the first number you crossed out is less than 5, then you are done. If it more than 5, then increase the last number you keep by 1. If it is 5 and any later number if more than 0, then also increase the last number you keep by one. If it is exactly 5 with only 0s following, then keep the last number unchanged if it is even, and increase it by 1 if it is odd. Some examples:

- 1.2349 –> 1.23
- 1.2351 –> 1.24
- 1.2350 –> 1.24
- 1.2250 –> 1.22 (!)

Why not just round up 5 like we all learned in school? If we always did that, we would introduce a small but consistent bias into our numbers. Therefore, this would better be randomised. The rule set out here goes towards randomisation without being too complicated, and is therefore what the APA guidelines recommend.

Chapter 10

Survey research - testing scales

A lot of social science research is done with surveys. In this video, I am talking about some examples and challenges, as well as some ways of asking good questions. I am also talking about how to create scales that use multiple items to measure one construct, and how to test their internal consistency.

10.1 Creating scales

Often, surveys use multiple items to measure the same construct. This is particularly common when participants might be either unable or unwilling to answer a direct question (e.g., how racist are you?). In such cases, asking several related questions can help to get an overall measure of the construct, while reducing measurement error, because the errors on different items will cancel each other out (to some extent).

Once you have data on your items, the most common way of creating a scale is just to calculate the mean of the items. If some of them were worded in the opposite direction (known as reverse-coding, and often recommended to increase data quality), those items need to be reversed first, most easily by subtracting their value from 1 + the scale maximum. [1]

[1] The `alpha()` function from the `psych` package can support reverse-coding - or even automate it. To learn more, look at `?psych::alpha`

10.1.1 Testing internal consistency

We might expect that our items all measure the same construct (e.g., prejudice against foreigners). However, we should test whether that is the case. The most common measure to use is **Cronbach's α**. It ranges from 0 to 1, and can be roughly understood as the intercorrelation of all items. Here is how to calculate it in R, using some data from the European Social Survey 2014.

```
pacman::p_load(psych, tidyverse)
ess <- read_rds(url("http://empower-training.de/Gold/round7b.RDS"))
ess %>% select(imtcjob, imbleco, imwbcrm) %>% psych::alpha()
```

```
## Number of categories should be increased  in order to count frequencies.

##
## Reliability analysis
## Call: psych::alpha(x = .)
##
##   raw_alpha std.alpha G6(smc) average_r S/N    ase mean  sd median_r
##       0.71      0.71    0.63      0.45 2.4 0.0025  4.4 1.8     0.45
##
##  lower alpha upper     95% confidence boundaries
## 0.7 0.71 0.71
##
##  Reliability if an item is dropped:
##         raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
## imtcjob      0.62      0.62    0.45      0.45 1.6   0.0038    NA  0.45
## imbleco      0.53      0.53    0.36      0.36 1.1   0.0047    NA  0.36
## imwbcrm      0.69      0.69    0.53      0.53 2.2   0.0031    NA  0.53
##
##  Item statistics
##             n raw.r std.r r.cor r.drop mean  sd
## imtcjob 38737  0.82  0.79  0.63   0.53  4.8 2.3
## imbleco 37872  0.83  0.83  0.71   0.60  4.5 2.2
## imwbcrm 38025  0.75  0.76  0.55   0.46  3.7 2.0
```

The key part of the output is the standardised alpha value (*std.alpha*). This indicates how internally consistent the scale is, and is usually interpreted in line with established cut-offs (DeVellis, 2016):

- $\alpha < .6$: poor,
- 0.6 α .7: questionable,
- 0.7 α .8: acceptable,
- 0.8 α .9: good,
- 0.9 α: excellent.

However, it should be noted that α increases with the length of the scale, so that a a short scale (e.g., 3 items) might well be of sufficient quality with $\alpha =$

.65, while 'excellent' levels of fit might require long scales that are costly or tiresome to administer.

The section on *Reliability if an item is dropped* can be helpful when you find low internal consistency; in those cases it might be necessary to exclude items from the scale that reduce its consistency. Decisions about droppping items should not be made just on these numbers as that would risk overfitting your scale to the particular sample at hand. Instead, always also consider the content of the items.

10.1.2 Calculating scale scores

The `alpha()` function from the `psych` package automatically calculates scores for each participant on the scale. To use them, you need to save the object that the function returns and then access the `scores` element.

```
scale_data <- ess %>% select(imtcjob, imbleco, imwbcrm) %>% psych::alpha()
#To print the details as above, just type scale_data

ess$immigrant_attitudes <- scale_data$scores
```

This calculates the mean of the items, automatically ignoring any missing values. If you prefer that participants who omitted one of the scale items have a missing value for the scale, it will be easiest to calculate the mean manually:

```
ess <- ess %>% mutate(immigrant_attitudes = (imbleco + imtcjob + imwbcrm)/3)
```

10.2 Further resources

- You can check this chapter of the Research Methods Knowledge Base for more details on various measures of internal consistency (and of other forms of reliability)

Supplementary topics

Chapter 11

Accessing online data sources

This section introduces some examples of R packages that allow you to access large secondary datasets. They are often a good way to understand wider trends, and thereby provide a high-level justification for doing research into a specific question. However, they can also be data sources for research projects in their own right.

11.1 World Bank data

The World Bank offers a rich dataset with a particular focus on indicators relevant for the study of poverty, inequality and global development (in fact, much of gapminder is based on World Bank data). You can explore their data online on data.worldbank.com, or access it directly from R using the `wbstats` package.

Here, I will explore the question whether life expectancy and literacy have increased in line with GDP in the BRICS countries (Brasil, Russia, India, China and South Africa, a group that has often been seen as representing emerging economies).

```
pacman::p_load(tidyverse)
pacman::p_load(wbstats)
```

```
#Download current list of indicators
new_wb_cache <- wb_cache()

#Search for indicators - you can best do this on data.worldbank.com
```

```
#and find the IndicatorID in the URL. The wbsearch() function also works
#but often returns too many hits.

#GDP per capita, purchasing power adjusted
#(to remove effect of exchange rates)
wb_search("gdp.*capita.*PPP", cache = new_wb_cache)
```

```
## # A tibble: 4 x 3
##   indicator_id    indicator                 indicator_desc
##   <chr>           <chr>                     <chr>
## 1 6.0.GDPpc_cons~ GDP per capita, PPP (co~ GDP per capita based on purchasing p~
## 2 NY.GDP.PCAP.PP~ GDP per capita, PPP (cu~ This indicator provides per capita v~
## 3 NY.GDP.PCAP.PP~ GDP per capita, PPP (co~ GDP per capita based on purchasing p~
## 4 NY.GDP.PCAP.PP~ GDP per capita, PPP ann~ Annual percentage growth rate of GDP~
```

Once we know the names of the indicators, we can download them.

```
#Note: to get the country names, you can download all countries once
#and then check the names
#wb_dat <- wb(indicator = c("NY.GDP.PCAP.PP.KD", "SI.POV.GINI"), country = "all")
#wb_dat %>% count(iso2c, country) %>% view()

wb_dat <- wb_data(indicator = c("NY.GDP.PCAP.PP.KD", "SP.DYN.LE00.IN", "SE.TER.ENRR"),
             country = c("IN", "BR", "CN", "ZA", "RU"), return_wide = FALSE)
```

Now we have the data in a "long" format - with one combination of countries, indicators and years per row. That is a good layout for plotting, for other analyses you would need to reshape the data into a wide format where each indicator is in its own variable - look for the spread() function if you need that.

A simple way of comparing the data is plotting the indicators side-by-side. One interesting take-away is that Brazil massively improved life expectancy and expanded education, even though GDP growth was rather modest, while South Africa stagnated in comparison.

```
#Our GDP series only starts in 1990 - so it does not make sense
#to consider earlier life expectancy
wb_datF <- wb_dat %>% filter(as.numeric(date)>=1990)

ggplot(wb_datF, aes(x=as.numeric(date), y=value, col=country)) +
  geom_point() + geom_line() +
  facet_wrap(.~indicator, scales = "free", labeller = labeller(indicator = label_wrap_
  #scales = "free" means that each indicator has its own y-axis,
  #the labeller() function is needed for line breaks in the facet titles
  labs(title = "Uneven progress in BRICS countries", subtitle = "World Bank data",
       x = "Year", y="", col = "Country")
```

You can find a similar but slightly more detailed example for how to use the

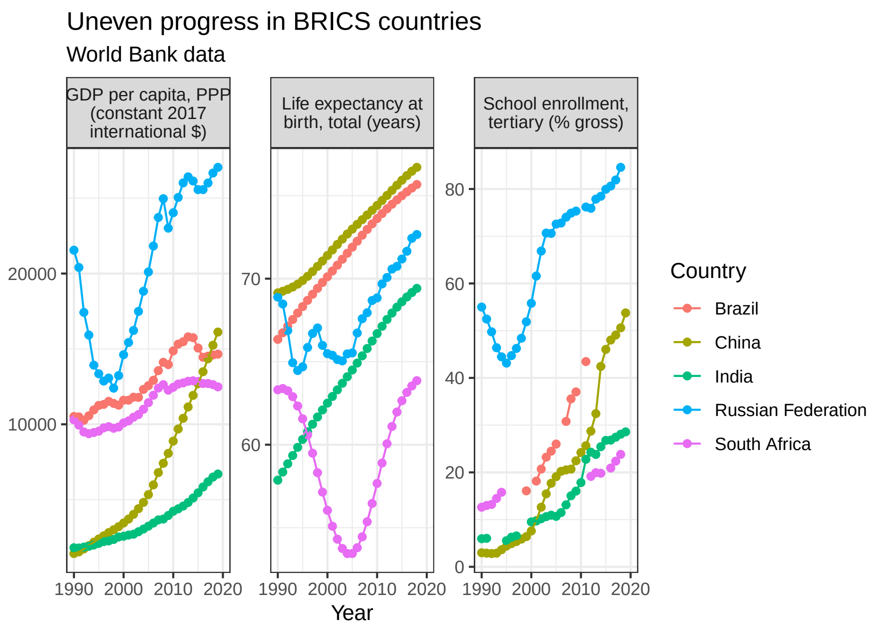


Figure 11.1: Example plot from World Bank data

package here and very clear instructions in the the full README file of the package.

11.2 Wikidata

Wikidata is where most data from Wikipedia and much else lives. So if there are Wikipedia articles on the topic you are interested in, you can likely find underlying data on Wikidata. For example, this might be used to quickly extract data on the gender of heads of government of many countries.

Wikidata is based on data items that are connected by multiple relationships. So there would be an item for *Germany*, an item for *Angela Merkel* and a relationship for *is the head of government of*. Similarly, there is an item for *country* and a relationship for *is an instance of* that connects it to *Germany*. SPARQL queries are used to get the data - this article explains the logic quite well, but unless you want to spend a couple more weeks learning how to code, you can just take examples from Wikidata and adjust them as needed. For adjusting them, the online Wikidata Query Service works well, as it allows you to run the query again and again, until you get the data you need.

I got curious about what share of the world's population lives in countries with a female head of government, and how that varies by region. For that, I used the following code. (gt is a package to make nice tables easily.) **Note that a key step is missing: I did not clean the data.** It contains some duplicates, for instance because countries that span two continents are included twice in the Wikidata output. Data cleaning is a crucial part of analysing online data, but not the focus of this chapter.

```
pacman::p_load(WikidataQueryServiceR, gt)

headsOfGov <- query_wikidata('
SELECT ?country ?head ?gender ?countryLabel ?headLabel ?genderLabel  ?continentLabel ?g
WHERE
{
    ?country wdt:P31 wd:Q6256 .
    ?country wdt:P6 ?head .
    ?head wdt:P21 ?gender .
    ?country wdt:P30 ?continent .
    ?country wdt:P1082 ?population .
    SERVICE wikibase:label { bd:serviceParam wikibase:language "en" }
}
ORDER BY ?countryLabel
')

regional <- headsOfGov %>% group_by(continentLabel, genderLabel) %>% summarise(pop = su

world <- data.frame(continentLabel = "World",
               headsOfGov %>% group_by(genderLabel) %>%
                 summarise(pop=sum(population), n=n()),
               stringsAsFactors = FALSE)


world %>% mutate(ShareOfCountries = n/sum(n)*100, ShareOfPopulation = pop/sum(pop)*100)

regionalAndWorld <- rbind(regional, world)
```

Women rule
Percent

ShareOfCountries	ShareOfPopulation
8.8	4.5

```
ggplot(regionalAndWorld, aes(x=continentLabel, y=pop, fill=genderLabel)) + geom_col(pos
    #Turns chart into bars rather than columns
    coord_flip() +
    #Show percentages rather than fractions on y-axis (now shown as x-axis)
    scale_y_continuous(labels=scales::percent) +
    labs(title="Only a small fraction of the world's population is ruled by women", sub
```

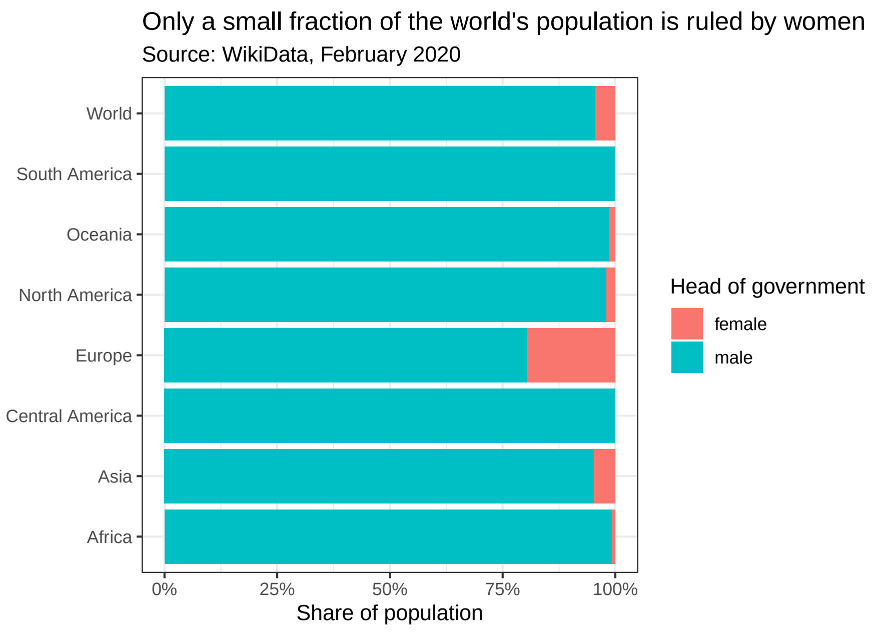


Figure 11.2: Example plot from Wikidata

11.3 Other data sources

Beyond the examples here, there are many other datasets to access. You might want to check out some of the following:

- The extensive list of political datasets compiled by Erik Gahner, with lots of current and historical data on anything from terrorism to government revenues and gender in politics. At the bottom, it also has a list of links to other lists of datasets.
- Eurostat offers a lot of statistics on all countries in Europe. In R, it can be accesses with the `eurostat` package; there is a good cheatsheet to help you get started
- The webpage asdfree.com, with (sparse) instructions of how to access a wide range of online data sources, from very focused surveys such as the US *National Longitudinal Study of Adolescent to Adult Health* to international and very widely used datasets such as the *World Values Survey*.
- You can also use R to scrape data from pretty much any public webpage. This tutorial shows how to get data from IMDB, for instance.
- Finally, the `essurvey` package is the easiest way to get data from the European Social Survey into R. There is a good example for how to use it here

Chapter 12

Dealing with missing data

Missing data can pose serious problems for any data analysis.

12.1 Example

Let's imagine that we want to analyse the effect of a mentoring programme on boys and girls. For that, they take a test (score1), then experience the mentoring, and then take another test (score2). Let's load the required packages.

```r
pacman::p_load(tidyverse) #As always
pacman::p_load(naniar) #To summarise and visualise missing data
pacman::p_load(simputation) #For imputation of missing data
```

I will simulate the data from our experiment as follows.

```r
set.seed(1234)
results <- data.frame(gender = c(rep("M", 250), rep("F", 250)),
                      score1 = round(c(rnorm(250, 45, 10), rnorm(250,60,10))),
                      score2 = round(c(rnorm(250, 55, 10), rnorm(250,70,10))))

results$score2[results$score1<40] <- NA
```

Note that I have assumed that only students who pass the first test (i.e. get at least 40) return to the second test. For all others, that data is missing. What pattern of missingness is that? MCAR MAR MNAR

Explanation

The data is **missing at random (MAR)** - whether score2 is missing depends on the value of score1, so that it is *not* missing completely at random

(MCAR). Whether score2 is missing does not depend on the value of score2 (beyond any association it might have with score1) - therefore it is also *not* missing not at random (MNAR).

Let's see how much missing data that results in.

```
miss_var_summary(results)
```

```
## # A tibble: 3 x 3
##   variable n_miss pct_miss
##   <chr>     <int>    <dbl>
## 1 score2       77     15.4
## 2 gender        0      0
## 3 score1        0      0
```

12.1.1 Case deletion

15% missing data is quite a lot, but still within the range where analysts might choose to use case deletion and move on.

Given that the data is MAR, do you think that case deletion can bias the results? Yes No

Let's try case deletion, and test whether the effects on boys and girls are different.

```
resultsDel <- results %>% na.omit() %>% #Deletes all rows with missing values
                    mutate(change = score2 - score1)

t.test(change ~ gender, resultsDel)
```

```
##
##  Welch Two Sample t-test
##
## data:  change by gender
## t = 3.3375, df = 401.71, p-value = 0.000924
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.665645 6.440134
## sample estimates:
## mean in group F mean in group M
##        9.008197        4.955307
```

Here we would conclude with great confidence ($p < .001$) that the intervention is more effective for girls than for boys. But given that we simulated the data ourselves, we know that the conclusion is false. It just arises because among the boys, the participants with the lowest scores who were most likely to score higher the next time round (*regression to the mean*) did not show up for the second test. This shows that in such a case, imputation is critical.

12.1.2 Imputation

Imputation means filling in the gaps, to try to ensure that the dataset we use for our analysis looks as similar as possible to the complete dataset, without missing data.

For imputation, we need to decide how the missing data should be predicted based on the observed data. Here, it would seem to make most sense to use linear regression to predict score2 from score1. However, before we do that, we need to make sure that we can identify imputed values later on by saving which values were missing to begin with. We use two functions from the `naniar` package for that.

```
resultsImp <- results %>% bind_shadow(only_miss = TRUE) %>% add_label_missings()

head(resultsImp)
```

```
## # A tibble: 6 x 5
##   gender score1 score2 score2_NA any_missing
##   <chr>   <dbl>  <dbl> <fct>     <chr>
## 1 M          33     NA NA        Missing
## 2 M          48     43 !NA       Not Missing
## 3 M          56     62 !NA       Not Missing
## 4 M          22     NA NA        Missing
## 5 M          49     73 !NA       Not Missing
## 6 M          50     53 !NA       Not Missing
```

`bind_shadow()` has added a new column for each column that had missing values - in this case, `score2_NA` was added, with two values showing whether the value was missing (NA) or not missing (!NA). Without the `only_miss = TRUE` argument, it would add one 'shadow' column for each column. `add_label_missings()` added just one new column (`any_missing`) indicating whether there were any missing values in that row.

Now we can impute the missing data. Functions from the `simputation` package help with that.

```
resultsImp <- resultsImp %>% group_by(gender) %>%
  impute_lm(score2 ~ score1, add_residual = "observed")
```

This tells `simputation` to split the data by gender, calculate a regression to predict score2 from score1 for each gender, use that to impute score2 values and then *add a residual* to each score2 value. The last part is crucial - if we don't do that, then all imputed values will be right on the regression line, which will artificially increase the correlation between score1 and score2 and the fit of any regression model we want to calculate later. `add_residual` can either be `"observed"` or `"normal"`. In the first case, one of the observed residuals is randomly sampled, in the second case, the added residuals follow a normal distribution. I would always recommend `"observed"` as that makes one fewer assumption about the distribution of our data.

Let's see whether the imputed values look reasonable and whether they are different from the observed values in a way that they are likely to influence our results.

```
resultsImp <- resultsImp %>% mutate(change = score2 - score1)
ggplot(resultsImp, aes(x=gender, y=change)) + geom_boxplot() + geom_jitter(aes(color =
```

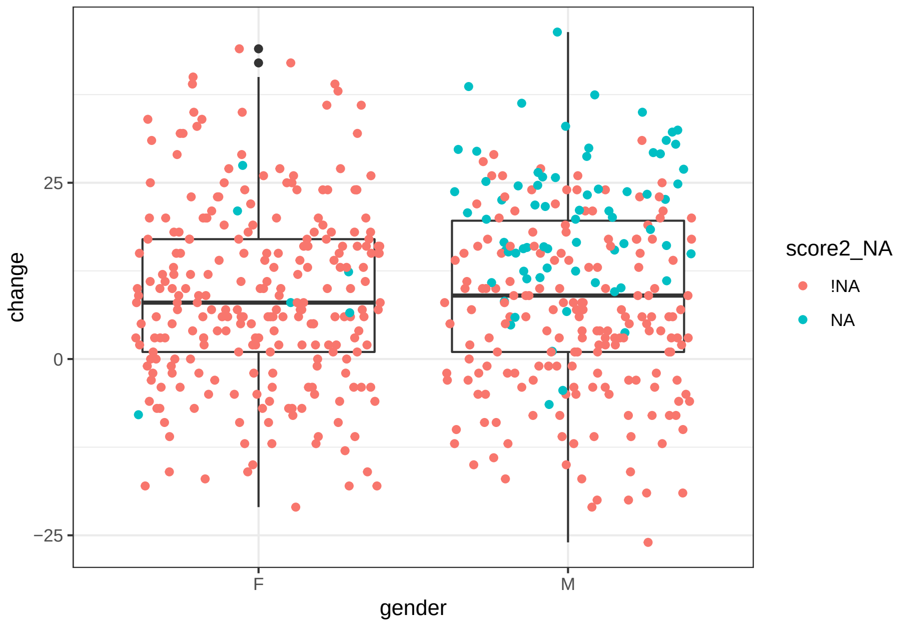


Figure 12.1: Visualisation shows that imputed data matters

Note that the colour aesthetic is only assigned inside `geom_jitter()` (which is the same as geom_point() but adds some random noise to the points to reduce overlap). If it was assigned within `ggplot()`, we would get separate boxplots for observed and imputed values, which is not the aim here.

Looking at the distribution of values that were missing and have now been imputed, we can see that the values are generally plausible, in that they fall within the observed range. However, particularly for boys, they are higher than the observed values and are therefore likely to influence our conclusion.

Let's test again whether there is a difference between boys and girls.

```
t.test(change ~ gender, resultsImp)
```

```
##
##  Welch Two Sample t-test
##
## data:  change by gender
## t = -0.13676, df = 497.8, p-value = 0.8913
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.459884  2.139711
## sample estimates:
## mean in group F mean in group M
##        9.062020        9.222106
```

This time, we get the correct result - there is no difference between boys and girls. This shows that when data is MAR (i.e. when missingness depends on other variables we are interested in) we have to consider imputation.

12.2 Further resources

- The official vignette for `naniar` provides provides good step-by-step instructions for exploring and visualising missing data
- Similarly, the official vignette for `simputation` contains details on how to use different imputation models and how to impute several variables at once

Chapter 13

Probability in statistics and real life

Much of statistics (namely *inferential* statistics) is about probability. Based on the data we have, is it likely that our hypothesis is true? But much of our other work as scientists, and much of everyday life, is also about probabilities. To be able to evaluate scientific claims, as well as news reporting and various situations in life, it helps to have a basic understanding of probability. By the end of this, you should be able to find an answer to questions like the following:

- How big does a group need to be until it is 95% likely that two of them share a birthday?
- Should it be news that there is a village in Poland where not a single boy has been born in more than a decade?
- How likely is it that someone has breast cancer when they receive a positive mammogram result?

13.1 What are probabilities?

There is still a lot of technical disagreement on definitions in this area. However, for our purposes, we just need to distinguish three concepts: procedures, events and probabilities. *Procedures* are anything that generates data - from rolling a die and writing down the number to conducting a medical test or surveying 500 people and calculating the mean on some outcome. An *event* is the result of a procedure, be that a 3 on a die or a positive result on a medical test. Finally, the *probability* expresses the likelihood with which an unobserved event will occur. Probabilities thereby offer guidance as to a reasonable degree of belief.

Probabilities can be based on experience and inference. I can count how often it rains later in the day when I wake up to a grey sky; if I do that for long enough and calculate the share of days when it rained as a total of all days I observed, this will give me a good guidance regarding the probability of rain. Weather forecasting apps do something a bit more sophisticated, but still quite similar.

Alternatively, we can use a more formal and logical way to find probabilities. This involves counting the number of possible outcomes of our procedure, and the number of possible outcomes that would lead to the event we are interested in. Probabilities are then expressed as fractions, dividing the number of ways in which our specified event can occur by the total number of outcomes. For example, the probability of rolling a 3 with a die is $\frac{1}{6}$, as there are 6 possible outcomes, and only one of them results in a three. Rolling an even numbers, on the other hand, has a probability of $\frac{3}{6}$ (or $\frac{1}{2}$) as there are now 3 out of 6 possible outcomes that lead to the event we are interested in.

Over to you - what is the probability of rolling a number smaller than 5? 2/6 or 1/3 3/6 or 1/2 4/6 or 2/3

These fractions can often be expressed more easily by a percentage. The probability of rolling an odd number is $\frac{3}{6}$ or 50%.

What is the probability of pulling a red card from a standard deck of cards, in percent?

Finally, probabilities can range from 0 (0 %) to 1 (100 %) - an event with a probability of 0 is impossible, one with a probability of 1 is inevitable.

13.2 Combining multiple events together

Often we are interested in more than one possible outcome, for instance when there are several paths to a desired outcome. So what is the probability of rolling an even number *or* a 3 with a die? We already know that it is $\frac{3}{6}$ for an even number and $\frac{1}{6}$ for a 3. So the probability for either of them to occur is $\frac{4}{6}$ - we can add them up.

However, what is the probability of rolling a 3 *or* an odd number? The probability for rolling an odd number is $\frac{3}{6}$, and the probability for rolling a 3 is $\frac{1}{6}$. Can we add them up? Remember that probabilities are the ways in which the event we are interested in can occur, divided by the total number of outcomes. Here, those are three ways (1, 3 and 5) of the possible six outcomes of rolling a die. Since rolling a 3 satisfies both sides of our *or* statement, we must not double count it.

As a shorthand, we can write $P(A)$ to mean the probability of event A. Formally, the probability of either of two events occurring is

$$P(AorB) = P(A) + P(B) - P(AandB)$$

Over to you - what is the probability of getting a King or Queen when you draw a card from a standard deck of cards (52 cards)? 4/52 or 7.5 % 8/52 or 15% 16/52 or 30%

I need a hint

Consider the total number of possible results as well as the possible number of ways you can get a King or a Queen

Give me the explanation

$P(King)$ is $\frac{4}{52}$ because there are 4 Kings among the 52 cards. Likewise, $P(Queen)$ is $\frac{4}{52}$ because there are 4 Queens. There is no card that is both a King and a Queen, so that we can just add up the probabilities and thus arrive at $\frac{8}{52}$

One more - what is the probability of getting a red card or a King or Queen when you draw a card from a standard deck of cards (52 cards)? 26/52 30/52 34/52

I need a hint

This time, there is a risk of double-counting. Avoid doing that.

Give me the explanation

$P(KingorQueen)$ is $\frac{8}{52}$, while $P(Red)$ $\frac{26}{52}$. However, there are 2 red Kings and 2 red Queens that are counted in both probabilities - therefore, we can get a red card or a King or a Queen in only 30 rather than 34 ways.

13.3 Looking at independent events

We are always interested in patterns that emerge when we look across multiple events. If we keep on tossing a coin, what is the probability to get two heads in a row? We can again count the number of possible outcomes versus the number of ways in which our event can occur.

As you can see in the figure, there are four possible outcomes, only one of which results in two heads, so that the probability for that is $\frac{1}{4}$ or 25%. What about four coins in a row turning up head? We know that there is only one way in which that can happen, but how many outcomes are there in total? We have seen that 2 coins result in four outcomes. Now tossing a third coin results in two possibilities for each of these 4 outcomes so far, turning them into 8. Tossing a fourth coin results in 16. (If you can't see the pattern, try to draw out the possibilities - it will help.). So the probability for four heads in a row is $\frac{1}{16}$ or about 6%

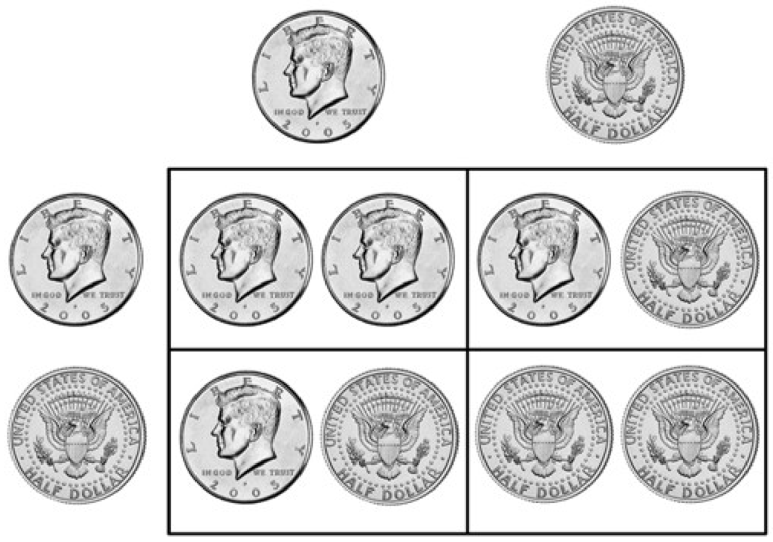


Figure 13.1: Results of two coin tosses

The probability of two events both occurring in sequence is $P(A) * P(B)$. That implies that the probability of the same event occurring n times in a row is $P(A)^n$.

Over to you - if you assume that 50% of babies born are girls, what is the probability of a woman to give birth to three girls in a row? 1/8 or 12.5% 1/3 or 33 % 1/2 or 50%

Give me the explanation

The probability for the first child to be a girl is 50% ($\frac{1}{2}$). For the second it is 50% as well. If we multiply them, we get 25%($\frac{1}{4}$). The third child again has a 50% chance of being a girl. Multiplying 25% by 50% results in 12.5% ($\frac{1}{8}$).

Quite often we are interested in the **likelihood of something occurring at least once,** rather than on every attempt. So we might want to know how likely it is that a family with three children has at least one boy among them. Assuming that gender is binary, that is the same as saying that they do not have three girls. Given that they either have 3 girls or at least one boy, we know that these probabilities add up to 1 (certainty) - such events are known as *complementary* events. Therefore, the probability of having at least one boy is 1 - 12.5% = 87.5%

Remembering that **something happening at least once is the *complement* of it never happening** is very helpful, because it is much easier to calculate the chance of something never happening across a given number (n) of procedures. It is $P(notA)^n$. We have already discussed this in the context of multiple comparisons.

Over to you. If the chance of finding a false positive is 5% on each attempt, what is the chance of finding a false positive in at least one of seven tests? For that, you will need a calculator. Note that x^n should be entered as `x^n` in R or the Google search box. Enter your answer in percent, rounded to the nearest whole number:

I need a hint

The answer is 100% minus the probability of not getting a false positive 7 times in a row.

Give me the explanation

The probability of not getting a false positive is 95%, so if we try that 7 times in a row, we get a probability of $0.95^7 = 0.70$ of getting *no* false positives. Given that getting at least one false positive is the complementary event to this, the chances of that are 100% - 70% = 30%

13.4 Updating our beliefs - Bayes' theorem

What you have looked at so far are the basic rules of probability that are good to remember (and that you might have seen a while ago in school). So far, we have treated probabilities as something abstract: we think about a situation, or collect data, and then assign a probability. In reality, that is not how it works. We rather use the probabilities of events to update our present beliefs. For example, if I leave my front door and see that the street is wet, I use that to update my belief (i.e. the probability I believe in) about whether it has rained earlier in the day, but in a way that depends on more than just the fact that the street is wet.

To explore how such an updating of beliefs is done, we are making quite a far jump to Bayes' theorem. You do not need to remember the formula, but the intuition is important. A key part of it are conditional probabilities - the probability of an event given that another event has occurred. We have worked with one kind of them extensively - p-values. They are the probability of getting the observed data, or data that is more extreme, if the null hypothesis is true. This can be written as $P(O|H_0)$

What Bayes' theorem reveals is how we can link that probability to the probability we are actually interested in, namely the probability that the hypothesis is true, given the data. In statistics, we often slip from one in the other, but if we think about examples, it becomes clear that they are not the same. For example, $P(StreetWet|Rained) \neq P(Rained|StreetWet)$. With regard to the first part, it is very likely that the street is wet if it has rained.

However, whether it is likely that it has rained if the street is wet depends on other factors - and those factors are what Bayes' theorem adds to our thinking. It suggests that

$$P(Rained|StreetWet) = \frac{P(StreetWet|Rained) * P(Rained)}{P(StreetWet)}$$

So, my belief whether is has rained given that the street is wet should be based on the probability that the street is wet *if* it has rained, but also on the probability I could assign to rain before I got that information, and the probability that the street is wet regardless of rain. If I saw beautiful sunshine through the window all day long, my belief for $P(Rained)$ might be very low, so that I would probably not give a high value to $P(Rained|StreetWet)$. Similarly, if the street in front of my house is wet all the time due to a broken pipe (high $P(StreetWet)$), I would not give a high value to $P(Rained|StreetWet)$.

Thinking in this way can help to interpret probabilistic results. A frequently used example (Gigerenzer and Hoffrage, 1995) goes as follows:

> The probability of breast cancer is 1% for women aged forty who participate in routine screening. If a woman has breast cancer, the probability is 80% that she will get a positive mammogram. If a woman does not have breast cancer, the probability is 9.6% that she will also get a positive mammogram. A woman in this age group has a positive mammogram in a routine screening. What is the probability that she actually has breast cancer?

What do you think? Type your *estimate* into this box before you read on, rounded to the nearest percent:

Let's think what would happen to 10,000 women who participate in the screening. We would expect that 100 of them will have breast cancer, and 80% of them will be tested positive, resulting in 80 true positives. Out of the remaining 9,900, 9.6% will also test positive, resulting in 950 false positives. So only 80 of 80+950=1030 positive tests are true positives, so that a woman who tests positive has a 7.6% chance of actually having breast cancer.

Over to you - let's assume a woman who has just tested positive goes through another mammogram and tests positive again. What is the probability now that she actually has breast cancer? (Hint: start with 10,000 women again - only one of the numbers changed compared to the example above.) Again, enter the number rounded to the nearest percent:

Give me the explanation

This time, out of 10,000 women, 7.6% would have breast cancer, so 760. 80% of them would test positive, resulting in 608 true positives. Out of the remaining 9,240 women, 9.6% would test positive even though they do not have breast cancer, resulting in 887 false positives. So this time round, 608 of

1,495 positive tests would be true positives (41%), which shows that the chance of having breast cancer increases to 41% after two positive tests.

The key message for evaluating the results of statistical tests, as well as any other tests, is that it helps to think about *prior probabilities.* If a hypothesis is very implausible, then even data that is only likely if that hypothesis was true should not cause us to entirely forget our informed (!) previous beliefs. Rather we should think about how to shift them.

Bayesian statistics starts from this idea, and has developed into a whole different approach to statistical inference. Unlike frequentist statistics (which we have focused on), it does not consider hypothetical repeated experiments to test whether data is inconsistent with a specific null hypothesis. Instead, it tries to use the data we actually obtain to shift our estimates of specific parameters. A big barrier to its wide-spread adoption is that it only adds a lot of value if it uses specific prior probabilities, which can be hard to obtain and justify - why would a specific hypothesis have a prior probability of being true of 8% rather than 1%? (And you have seen above what a difference that makes for our belief after we have obtained new evidence.) One surprisingly powerful way around that is to survey experts … but this is all material for your own further study or a more advanced course. As a starting point, you can have a look at the first presentation linked to under Further Resources.

13.5 Further resources

- This presentation provides a fairly brief introduction to Bayesian statistics
- Examples like the one regarding the breast cancer screening have led to lots of discussion about how to communicate the reliability of medical tests in a way that patients (and policy makers) understand. This article by Navarette et al. (2015) goes through some further interesting examples and ways of presenting information and highlights why mass screenings are often not as helpful as they appear at first glance.

Appendix A

Packages and references

This guide is created in RStudio, using the `bookdown` package (Xie, 2020). Its documentation, as well as the other relevant packages, are included in the references below.

Appendix B

Two more pipes

If you are clear on the pipe operator `%>%` then two more pipes can help you save some typing and write more efficient code. However, neither are critical for this course. All of them require that you load the `magrittr` package explicitly - it is installed as part of the tidyverse, but not loaded.

```
library(tidyverse)
library(magrittr)
#Example data
constituencies <- read_csv(url("http://empower-training.de/Gold/ConstituencyData2019.csv"),
                           col_types = "_cfddddfffddddfffdfdddd")
```

B.1 Expanding the tidyverse: the exposition pipe (%$%)

Inside dplyr pipes, you do not need to use the `$` to access variables within the dataframe. However, that does not work in some base-R functions such as `cor.test()`. For them, `%$%` can expose the variables in a dataframe temporarily (until the end of that command) so that they can be accessed as if they were separate variables in your environment.

```
#Normal code
cor.test(constituencies$MedianAge, constituencies$ElectionConShare)
```

```
##
##  Pearson's product-moment correlation
##
## data:  constituencies$MedianAge and constituencies$ElectionConShare
## t = 16.454, df = 648, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4862318 0.5949037
## sample estimates:
##      cor
## 0.542836
```

```
#With the exposition pipe
constituencies %$% cor.test(MedianAge, ElectionConShare)
```

```
##
##  Pearson's product-moment correlation
##
## data:  MedianAge and ElectionConShare
## t = 16.454, df = 648, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4862318 0.5949037
## sample estimates:
##      cor
## 0.542836
```

Clearly in this case, the difference is slight but you might find it more readable as well. In other cases, this can spare you from having to save a filtered dataset or otherwise processed dataset into a new variable that you are only using once.

```
constituencies %>% filter(ContType == "County") %$%
  cor.test(MedianAge, ElectionConShare)
```

```
##
##  Pearson's product-moment correlation
##
## data:  MedianAge and ElectionConShare
## t = 7.4961, df = 366, p-value = 5.003e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2727644 0.4502600
## sample estimates:
##       cor
## 0.3648222
```

B.2 A somewhat risky time-saver: the assignment pipe (%<>%)

Quite often, we want to edit an element in place, which leads to some repetition as you need to specify that you want to put it into the pipe and then save it back in place.

```
#Normal code
constituencies <- constituencies %>% filter(ConstituencyName != "Chorley")

#With assignment pipe
constituencies %<>% filter(ConstituencyName != "Chorley")
```

The assignment pipe here takes `constituencies`, puts it into the pipeline, and takes the result at the end to save back into `constituencies`. It can be particularly pleasant when you want to apply a function to a single variable, but be careful not to use it accidentally - if you type `%<>%` instead of `%>%` you will create bugs in your code that are not easy to spot.

```
constituencies$ElectionWon %<>% relevel(ref="Lab")
```

Bibliography

DeVellis, R. F. (2016). *Scale development: Theory and applications*, volume 26. Sage publications.

Gigerenzer, G. and Hoffrage, U. (1995). How to improve bayesian reasoning without instruction: frequency formats. *Psychological review*, 102(4):684.

Xie, Y. (2020). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.20.