Introducing R - one day course

Anthony Staines, Sara McQuinn, and Gillian Paul

December 14 2022

Contents

1 What are we doing today

We are going to show you how to use RStudio for simple data analysis. These tools will take you a very long way in answering questions about data.

1.1 First steps

Install RStudio Desktop from https://posit.co/downloads/ Install R from https://www.stats.bris.ac.uk/R/

Start and run both, to make sure they work. We'll be using RStudio exclusively, but you must have R installed too, for RStudio to work.

In Rstudio click on the file we sent you, which is called *2023 MSc Nursing day.zip*. This is a compressed file containing one folder.

Unzip it, to take out the folder, following these instructions :- https://support.microsoft.com/en-us/windows /zip-and-unzip-files-f6dde0a7-0fec-8294-e1d3-703ed85e7ebc

Put the resulting folder *2023 MSc Nursing day* anywhere suitable, but be sure you **know** where it is.

Click on the file in that folder called *2023 MSc Nursing day.Rproj* which will open the project workspace for the seminar.

Click File -> Open File in the menu bar at the top, and select and open the file called *Intro.Rmd* - this file.

Please be **sure** all this is working before the day of the seminar.

2 What will we cover today

1. Some basic words - package, and the *library()* command
2. Numbers, characters and dates
3. Reading in data from Excel files (examples provided, or bring your own)
4. Summary statistics for data
5. Simple graphs and tables
6. Summary statistics for graphs and tables

3 Basic ideas

R is a system, built from three pieces.

1. There's a core;
2. There's a set of default packages, which are always available;
3. There are packages **you** choose to add.

Packages do things, for example we have a package of statistical tests, called *stats*, a package for graphs called *graphics*, a package for dates called *lubridate*, and many more.

Some packages, the default set, are already installed and are automatically loaded when you start RStudio. However, for many others, you have to install them yourself from the internet, and then load them yourself.

To do this you need to type commands, and run them. You run a command by either typing it into the Console (below), and pressing the [Enter] key when you are done, or by running the code below this line.

There are several ways to do that, for this example we recommend moving the cursor to the line you want to run, (beginning with `install.packages`) and pressing the [Control] (or [Ctrl]) and [Enter] keys together. Please do that now.

```
if (!require('tidyverse')) { # Do we have it already? If not, get it.
  install.packages('tidyverse', quietly = TRUE,
           repos =  'https://www.stats.bris.ac.uk/R/')
  }
# Note the ! in front of require/

# require(tidyverse) would load the tidyverse package, if it was already installed.
#
# if (require(tidyverse)) {...} is TRUE if it's already installed, and would go on to do whatever was b
#
# if (! require(tidyverse)) {...} is FALSE if it's already installed, and would go on to do whatever wa
```

The text in grey is called a chunk, it's a discrete piece of R code that does something. In the original file it's the text between “`{r}" and "`”.

All going well, this command (or this chunk, if that's how you choose to run it) will install multiple packages on your computer, in an area called the library.

The next two commands will load these packages, from the library. You can run these one at a time with [Ctrl][Enter] as before, or you can press the little green right pointing arrowhead at the top right of the code chunk to run the whole chunk.

```
library('tidyverse', quietly = TRUE)
library('readxl', quietly = TRUE)
```

The first of these *library('tidyverse')* loads a set of packages for drawing graphs, and storing data. The second *library('readxl')* loads a package for reading data from Excel files.

4 Numbers, characters and dates

A quick diversion. Think about what goes into an Excel file. Typically there may be one or more sheets, each with a name. Often there is a row of names across the top of the file, where each name tells you what is in the column underneath it. There is often a column, usually the first column, which has either identifiers of some kind, or text descriptions of some kind, giving information on the remainder of the row. (R can read from lots of sources, but we'll start simple).

R, like pretty much every other statistics package on earth, works on two assumptions

Rows are observations (for example one person, or one clinic visit, or one blood test result, or one nursing intervention, is one row);

Columns are data - for a prescription perhaps

1. Patient ID number,
2. Drug name,
3. Dose,
4. Route of administration
5. Frequency of administration
6. Date of the prescription
7. Code to say if this is a repeat prescription or a new prescription, and so on.

If any of this isn't true, it can be fixed, but that takes a good bit more work, and we won't cover this today.

Some columns are just numbers, some are just characters, either words or letters, some are dates, and some are codes. R can read all of these.

What is important, is always to check. For example, Excel has some odd ideas about what a date is. If all your dates are out by a few years, that's because there are (at least) three versions of Excel dates out there. It also allows you to put letters in columns of numbers. If you read in data from Excel, and get a column of letters when you expected numbers, have a look and see if there's a capital O hiding in the 0's or a capital I hiding in the 1's.

So, check - the *str()* command (short for stucture) is your friend here.

5 Reading in data from Excel files (examples provided, or bring your own)

People often have data in Excel spreadsheets. R makes it easy to read these. I've given you one to get you started, but feel free to use your own. This one is the number of practicing nurses per 1,000 population for the last 12 years, for a set of wealthy countries, from the OECD data site https://stats.oecd.org/.

It's not a bad idea to start by listing the sheets in the Excel file provided with the *excel_sheets()* command. Notice that the result of the command pops up in the Global Environment pane on the top right.

```
Sheets <- excel_sheets('data/Practicing_Nurses_per_1000.xlsx')
```

The little <- symbol is called an assignment operator - it sets the value of Sheets to whatever *excel_sheets()* produces. Another way to say exactly the same thing is that the <- operator gives a name to the output of *excel_sheets()*, so you can refer to it again.

(If you've done programming before, you may be more familiar with the use of the = sign for this, but there are good reasons for a separate operator for assignment.)

Typing the word 'Sheets', prints the value in the console.

```
Sheets
```

```
## [1] "Nurses"
```

It would now be good to bring in the data. Open the spreadsheet *Practicing_Nurses_per_1000.xlsx* in Excel, or LibreOffice, and see what it looks like.

Now we want to read it in, and we can use the *read_excel()* command to do this. We assign the output of the *read_excel()* command to the name *OECD_Nurses*, but we could have called it almost anything, Akallabeth, syzygy or cf2ce19-c1f5-40c3-9198-26fe2a122cc4, as takes your fancy.

```
OECD_Nurses <- read_excel('data/Practicing_Nurses_per_1000.xlsx',
                          sheet = 'Nurses',
                          na = "")
```

As there's only one sheet, you don't really need the *sheet = 'Nurses'* bit, but you might have more than one sheet in your excel file.

What about the *na = " "* bit? R has a clear idea of a missing value, which is shown as NA. In these data the number of practicing nurses is not known for quite a few years in some countries, and so is blank - go look at the spreadsheet again. In fact blank is the default missing value option for *read_excel()* Excel, so we don't need that either, but you might have a file with another value, so at least you know where to look.

To see what we've got the easiest way is to click on the name *OECD_Nurses* in the Global Environment pane. If you look down at the console, you will see that this puts the command *View(OECD_Nurses)* into the console and runs it. If you prefer you can type the command.

R keeps data internally in things called 'dataframes'. Each looks like a spreadsheet table, and you can see them using the *View()* command, of just by clicking on them in the *Global Environment* pane.

You may also hear of *tibbles*, which are a slightly updated version of a *dataframe*, but otherwise much the same.

6 Summary statistics for data

There are a few things you might want to know about your data. The *summary()* command will give you quite a lot of useful information.

```
summary(OECD_Nurses)
```

```
##    Country           Year_2010        Year_2011        Year_2012
##  Length:38          Min.   : 0.670   Min.   : 0.730   Min.   : 0.800
##  Class :character   1st Qu.: 4.935   1st Qu.: 4.920   1st Qu.: 4.830
##  Mode  :character   Median : 7.330   Median : 7.785   Median : 7.940
##                     Mean   : 7.380   Mean   : 7.521   Mean   : 7.547
##                     3rd Qu.: 9.918   3rd Qu.:10.125   3rd Qu.:10.200
##                     Max.   :16.130   Max.   :16.400   Max.   :16.530
##                     NA's   :6        NA's   :6        NA's   :5
##    Year_2013        Year_2014        Year_2015        Year_2016
##  Min.   : 0.880   Min.   : 0.970   Min.   : 0.880   Min.   : 1.080
##  1st Qu.: 4.865   1st Qu.: 5.005   1st Qu.: 4.780   1st Qu.: 4.902
```

```
##  Median : 7.120   Median : 7.930   Median : 7.660   Median : 7.785
##  Mean   : 7.415   Mean   : 7.783   Mean   : 7.578   Mean   : 7.800
##  3rd Qu.:10.242   3rd Qu.:10.650   3rd Qu.:10.625   3rd Qu.:10.890
##  Max.   :16.670   Max.   :16.890   Max.   :17.310   Max.   :17.480
##  NA's   :4        NA's   :3        NA's   :3        NA's   :2
##    Year_2017        Year_2018        Year_2019       Year_2020
##  Min.   : 1.140   Min.   : 1.210   Min.   : 1.10   Min.   : 1.030
##  1st Qu.: 4.255   1st Qu.: 4.520   1st Qu.: 5.01   1st Qu.: 5.380
##  Median : 7.330   Median : 7.915   Median : 8.20   Median : 8.415
##  Mean   : 7.627   Mean   : 7.951   Mean   : 8.17   Mean   : 8.461
##  3rd Qu.:10.925   3rd Qu.:11.023   3rd Qu.:10.37   3rd Qu.:10.960
##  Max.   :17.660   Max.   :17.710   Max.   :17.96   Max.   :18.370
##  NA's   :2        NA's   :4        NA's   :9       NA's   :12
##    Year_2021
##  Min.   : 1.550
##  1st Qu.: 6.425
##  Median : 8.680
##  Mean   : 9.309
##  3rd Qu.:11.855
##  Max.   :18.370
##  NA's   :31
```

The output to the console includes (for every numeric variable) the smallest and largest values Min and Max; the 1st, 2nd, and third quartiles, 1st Qu., Median, and 3rd Qu.; and the Mean. It doesn't include the standard deviation, nor the inter-quartile range, which can both be useful. Missing values are ignored in these calculations, but it does tell you the number of missing values - NA's.

For a character variable, *Country* in this case, all it gives is that it is a character variable.

A more useful summary comes from a different package, *describe()* from the *Hmisc* package. This package is installed by default in R.

```
library(Hmisc, quietly = TRUE)
  describe(OECD_Nurses)
```

```
## OECD_Nurses
##
##  13  Variables      38  Observations
## --------------------------------------------------------------------------------
## Country
##        n  missing distinct
##       38        0       38
##
## lowest : Argentina      Australia      Austria        Belgium        Brazil
## highest: South Africa   Spain          Sweden         Switzerland    United Kingdom
## --------------------------------------------------------------------------------
## Year_2010
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       32        6       32        1     7.38    4.645   0.9525   1.5920
##      .25      .50      .75      .90      .95
##   4.9350   7.3300   9.9175  11.8780  14.5850
##
## lowest :  0.67  0.87  1.02  1.50  2.42, highest: 11.05 11.97 14.54 14.64 16.13
## --------------------------------------------------------------------------------
## Year_2011
##        n  missing distinct     Info     Mean      Gmd      .05      .10
```

```
##       32        6       32        1    7.521    4.723    1.034    1.719
##      .25      .50      .75      .90      .95
##    4.920    7.785   10.125   12.089   14.977
##
## lowest :  0.73  0.99  1.07  1.63  2.52, highest: 11.27 12.18 14.82 15.17 16.40
## ---------------------------------------------------------------------------
## Year_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       33        5       31        1    7.547     4.93    1.110    1.272
##      .25      .50      .75      .90      .95
##    4.830    7.940   10.200   12.344   15.324
##
## lowest :  0.80  1.11  1.14  1.80  2.56, highest: 11.92 12.45 15.16 15.57 16.53
## ---------------------------------------------------------------------------
## Year_2013
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       34        4       34        1    7.415    4.918    1.209    1.874
##      .25      .50      .75      .90      .95
##    4.865    7.120   10.242   12.406   15.611
##
## lowest :  0.88  1.19  1.22  1.82  2.00, highest: 11.93 12.61 15.45 15.91 16.67
## ---------------------------------------------------------------------------
## Year_2014
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       35        3       34        1    7.783    4.897    1.328    2.108
##      .25      .50      .75      .90      .95
##    5.005    7.930   10.650   12.492   15.579
##
## lowest :  0.97  1.23  1.37  2.08  2.15, highest: 11.97 12.84 15.33 16.16 16.89
## ---------------------------------------------------------------------------
## Year_2015
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       35        3       35        1    7.578    5.086    1.198    1.698
##      .25      .50      .75      .90      .95
##    4.780    7.660   10.625   12.528   15.789
##
## lowest :  0.88  1.03  1.27  1.45  2.07, highest: 11.91 12.94 15.45 16.58 17.31
## ---------------------------------------------------------------------------
## Year_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       36        2       36        1      7.8     4.94    1.453    2.330
##      .25      .50      .75      .90      .95
##    4.902    7.785   10.890   12.350   14.920
##
## lowest :  1.08  1.31  1.50  2.18  2.48, highest: 11.72 12.98 14.22 17.02 17.48
## ---------------------------------------------------------------------------
## Year_2017
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       36        2       36        1    7.627    5.098    1.317    1.870
##      .25      .50      .75      .90      .95
##    4.255    7.330   10.925   12.495   15.183
##
## lowest :  1.14  1.31  1.32  1.53  2.21, highest: 11.72 13.27 14.50 17.23 17.66
## ---------------------------------------------------------------------------
```

```
## Year_2018
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       34        4       33        1    7.951     5.17    1.448    1.922
##      .25      .50      .75      .90      .95
##    4.520    7.915   11.023   13.075   15.692
##
## lowest :  1.21  1.22  1.57  1.70  2.44, highest: 11.92 13.57 14.67 17.59 17.71
## ---------------------------------------------------------------------------
## Year_2019
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       29        9       29        1     8.17    5.077    1.630    2.714
##      .25      .50      .75      .90      .95
##    5.010    8.200   10.370   12.848   16.872
##
## lowest :  1.10  1.27  2.17  2.85  3.10, highest: 11.79 12.22 15.36 17.88 17.96
## ---------------------------------------------------------------------------
## Year_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       26       12       26        1    8.461    5.399    1.635    2.595
##      .25      .50      .75      .90      .95
##    5.380    8.415   10.960   13.945   17.415
##
## lowest :  1.03  1.42  2.28  2.91  3.27, highest: 12.10 12.26 15.63 18.01 18.37
## ---------------------------------------------------------------------------
## Year_2021
##        n  missing distinct     Info     Mean      Gmd
##        7       31        7        1    9.309    6.463
##
## lowest :  1.55  6.26  6.59  8.68 10.91, highest:  6.59  8.68 10.91 12.80 18.37
##
## Value       1.55  6.26  6.59  8.68 10.91 12.80 18.37
## Frequency      1     1     1     1     1     1     1
## Proportion 0.143 0.143 0.143 0.143 0.143 0.143 0.143
## ---------------------------------------------------------------------------
```

In either case, please look carefully to see if this is right. There is little point in engaging in an elaborate analysis of the wrong data.

7 Simple graphs and tables

Suppose you want to know how the mean has changed over time, which is quite a reasonable question. One catch is this - you have to tell the *mean()* function explicitly to ignore missing values, by using the *na.rm* option.

```
mean(OECD_Nurses$Year_2010) # Missing value
```

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

```
mean(OECD_Nurses$Year_2010, na.rm = TRUE) # 7.38
```

```
## [1] 7.379688
```

```
mean(OECD_Nurses$Year_2011, na.rm = TRUE) # 7.52
```

```
## [1] 7.520937
```

```r
mean(OECD_Nurses$Year_2012, na.rm = TRUE) # 7.54
```

```
## [1] 7.54697
```

```r
mean(OECD_Nurses$Year_2013, na.rm = TRUE) # 7.42
```

```
## [1] 7.415
```

```r
mean(OECD_Nurses$Year_2014, na.rm = TRUE) # 7.78
```

```
## [1] 7.783143
```

```r
mean(OECD_Nurses$Year_2015, na.rm = TRUE) # 7.58
```

```
## [1] 7.578286
```

```r
mean(OECD_Nurses$Year_2016, na.rm = TRUE) # 7.8
```

```
## [1] 7.8
```

```r
mean(OECD_Nurses$Year_2017, na.rm = TRUE) # 7.63
```

```
## [1] 7.626667
```

```r
mean(OECD_Nurses$Year_2018, na.rm = TRUE) # 7.95
```

```
## [1] 7.950588
```

```r
mean(OECD_Nurses$Year_2019, na.rm = TRUE) # 8.16
```

```
## [1] 8.169655
```

```r
mean(OECD_Nurses$Year_2020, na.rm = TRUE) # 8.46
```

```
## [1] 8.461154
```

```r
mean(OECD_Nurses$Year_2021, na.rm = TRUE) # 9.30
```

```
## [1] 9.308571
```

This does look as if they are going up, but notice that the number of countries with data is falling in the later years.

However thinking further, you have a set of values for each year. There are only 12 years here, so you can do it by hand, but suppose you had 250 hospital wards, 1,200 primary care centres, or the like. The 'Year' is actually data, not the name of a variable. So we treat it as such.

We use the command *pivot_longer()* which "lengthens" data, increasing the number of rows and decreasing the number of columns. The reverse, if you need it, is *pivot_wider()*.

```r
Nurses <-
  pivot_longer(OECD_Nurses,
               cols = -Country, # Leave these
               names_to = 'Year', # New column
               names_prefix = 'Year_', # Lose this
               values_to = 'Nurses_per_k') # New variable name
```

Check!

```r
View(Nurses) # have a look at it
  str(Nurses)
```

```
## tibble [456 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Country      : chr [1:456] "Australia" "Australia" "Australia" "Australia" ...
```

```
##  $ Year        : chr [1:456] "2010" "2011" "2012" "2013" ...
##  $ Nurses_per_k: num [1:456] NA 10.2 10.2 11.1 11.3 ...
```

Oops - Year is not a number, which may bite us later on, so we fix it with the *mutate()* command. Guess what *as.numeric()* does.

```
Nurses <- Nurses %>%
  mutate(Year = as.numeric(Year))
  # the mutate command makes a new variable
  #   (or renames an existing one)

str(Nurses) # Year is now a number
```

```
## tibble [456 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Country     : chr [1:456] "Australia" "Australia" "Australia" "Australia" ...
##  $ Year        : num [1:456] 2010 2011 2012 2013 2014 ...
##  $ Nurses_per_k: num [1:456] NA 10.2 10.2 11.1 11.3 ...
```

I said earlier that we could reverse the *pivot_longer()* command, and this is how it is done with *pivot_wider()*.

```
Wider <- pivot_wider(Nurses,
            names_from = 'Year', # New column names are the values of Year
            names_prefix = 'Year_', # Put it back
            values_from = 'Nurses_per_k')
# As promised - back where we started!
```

7.1 Tables

We'll stick with the *Nurses* dataframe for the moment, and try to answer our original questions.

```
Country_SUMMARY <- Nurses %>% # The dataframe we're working on
  group_by(Country) %>% # Divided up by country
  summarise(Mean = mean(Nurses_per_k,
                        na.rm = TRUE), # Average - measure of centre
            SD = sd(Nurses_per_k,
                        na.rm = TRUE), # Standard deviation - measure of variability
            Min = min(Nurses_per_k,
                        na.rm = TRUE), # Smallest
            Median = median(Nurses_per_k,
                        na.rm = TRUE), # Middle value
            Max = max(Nurses_per_k,
                        na.rm = TRUE), # Largest
            NA_Count = sum(is.na(Nurses_per_k)) # Number of missing values
            )
```

This looks long, and complicated, but it isn't. First, it's the exact same thing written six times, for six slightly different functions (or commands), and it covers every country.

Second to change it to do the same thing by *Year* involves changing exactly one word. In real life, this is a big win.

```
Year_SUMMARY <- Nurses %>% # The dataframe we're working on
  group_by(Year) %>%  # divided up by Year
  summarise(Mean = mean(Nurses_per_k,
                        na.rm = TRUE),
            SD = sd(Nurses_per_k,
                        na.rm = TRUE),
            Min = min(Nurses_per_k,
```

```
                  na.rm = TRUE),
            Median = median(Nurses_per_k,
                  na.rm = TRUE),
            Max = max(Nurses_per_k,
                  na.rm = TRUE),
            NA_Count = sum(is.na(Nurses_per_k))
            )
```

The framework here is very simple

```
Output <- Dataframe %>%
    group_by(Variable to report on) %>%
    summarise(Thing or things to report)
```

We've already seen the assignment operator *<-* and now we see what is called the pipe operator *%>%* which just passes things on from one command or function to the next.

So we have two tables - Country_SUMMARY and Year_SUMMARY. These are not maybe the smartest names, but they will do for now. To print them, and make them look nice, we can use the command *kable()* from the *knitr* package. There are lots of other choices, but this will do for a start.

```
library(knitr)
  kable(Country_SUMMARY)
```

Country	Mean	SD	Min	Median	Max	NA_Count
Argentina	3.140000	0.9613012	2.58	2.590	4.25	9
Australia	11.385000	0.7254769	10.19	11.480	12.26	2
Austria	7.401818	1.4983846	6.53	6.790	10.48	1
Belgium	10.486667	0.5855766	9.59	10.580	11.22	3
Brazil	1.062500	0.2718664	0.67	1.055	1.55	0
Canada	9.751818	0.2910951	9.29	9.910	10.06	1
China (People's Republic of)	2.343636	0.5927103	1.50	2.300	3.27	1
Czech Republic	8.208182	0.2858957	7.93	8.060	8.66	1
Denmark	9.966000	0.0962866	9.82	9.950	10.13	2
Estonia	6.101818	0.2272364	5.64	6.170	6.38	1
Finland	12.756667	0.5101960	11.97	12.840	13.57	3
Germany	10.864546	0.6974577	9.87	10.720	12.06	1
Greece	3.330000	0.0888194	3.21	3.325	3.47	2
Hungary	6.450833	0.1453809	6.21	6.455	6.62	0
Iceland	15.011818	0.4770496	14.22	15.160	15.63	1
India	1.290000	0.2539193	0.87	1.370	1.57	3
Indonesia	1.670000	0.5851496	0.88	1.700	2.28	7
Ireland	12.800000	NA	12.80	12.800	12.80	11
Israel	4.925454	0.1326924	4.71	4.880	5.14	1
Italy	5.595833	0.4479541	5.08	5.505	6.28	0
Japan	11.135000	0.7482179	10.11	11.150	12.10	6
Korea	6.198182	1.3353787	4.61	5.940	8.37	1
Latvia	4.667273	0.2709646	4.18	4.680	5.01	1
Lithuania	7.640909	0.1274719	7.37	7.660	7.81	1
Luxembourg	11.686250	0.3433007	11.05	11.815	11.97	4
Mexico	2.722727	0.1701230	2.42	2.770	2.91	1
Netherlands	10.782857	0.2992053	10.33	10.770	11.16	5
New Zealand	10.242500	0.2590937	10.02	10.170	10.91	0
Norway	17.253333	0.7151775	16.13	17.395	18.37	0
Peru	1.991429	0.4189045	1.14	2.080	2.44	5

Country	Mean	SD	Min	Median	Max	NA_Count
Poland	5.218571	0.0689030	5.10	5.240	5.28	5
Russia	8.468000	0.4487959	7.29	8.495	8.95	2
Slovenia	9.163636	0.9267284	8.16	8.780	10.47	1
South Africa	1.169091	0.1078383	1.02	1.190	1.31	1
Spain	5.481818	0.3560567	5.14	5.290	6.10	1
Sweden	10.920000	0.0452155	10.85	10.930	10.97	2
Switzerland	16.563636	1.1890692	14.64	16.580	18.37	1
United Kingdom	8.119167	0.2728456	7.83	7.995	8.68	0

This isn't very pretty, but modest effort can make it much better.

```
kable(Country_SUMMARY,
      digits = 2,
      caption = 'Summary of Practicing nurses per 1,000 population, by Country - Source OECD https://sta
```

Table 2: Summary of Practicing nurses per 1,000 population, by Country - Source OECD https://stats.oecd.org

Country	Mean	SD	Min	Median	Max	NA_Count
Argentina	3.14	0.96	2.58	2.59	4.25	9
Australia	11.38	0.73	10.19	11.48	12.26	2
Austria	7.40	1.50	6.53	6.79	10.48	1
Belgium	10.49	0.59	9.59	10.58	11.22	3
Brazil	1.06	0.27	0.67	1.06	1.55	0
Canada	9.75	0.29	9.29	9.91	10.06	1
China (People's Republic of)	2.34	0.59	1.50	2.30	3.27	1
Czech Republic	8.21	0.29	7.93	8.06	8.66	1
Denmark	9.97	0.10	9.82	9.95	10.13	2
Estonia	6.10	0.23	5.64	6.17	6.38	1
Finland	12.76	0.51	11.97	12.84	13.57	3
Germany	10.86	0.70	9.87	10.72	12.06	1
Greece	3.33	0.09	3.21	3.33	3.47	2
Hungary	6.45	0.15	6.21	6.46	6.62	0
Iceland	15.01	0.48	14.22	15.16	15.63	1
India	1.29	0.25	0.87	1.37	1.57	3
Indonesia	1.67	0.59	0.88	1.70	2.28	7
Ireland	12.80	NA	12.80	12.80	12.80	11
Israel	4.93	0.13	4.71	4.88	5.14	1
Italy	5.60	0.45	5.08	5.51	6.28	0
Japan	11.13	0.75	10.11	11.15	12.10	6
Korea	6.20	1.34	4.61	5.94	8.37	1
Latvia	4.67	0.27	4.18	4.68	5.01	1
Lithuania	7.64	0.13	7.37	7.66	7.81	1
Luxembourg	11.69	0.34	11.05	11.82	11.97	4
Mexico	2.72	0.17	2.42	2.77	2.91	1
Netherlands	10.78	0.30	10.33	10.77	11.16	5
New Zealand	10.24	0.26	10.02	10.17	10.91	0
Norway	17.25	0.72	16.13	17.40	18.37	0
Peru	1.99	0.42	1.14	2.08	2.44	5
Poland	5.22	0.07	5.10	5.24	5.28	5
Russia	8.47	0.45	7.29	8.50	8.95	2
Slovenia	9.16	0.93	8.16	8.78	10.47	1

Country	Mean	SD	Min	Median	Max	NA_Count
South Africa	1.17	0.11	1.02	1.19	1.31	1
Spain	5.48	0.36	5.14	5.29	6.10	1
Sweden	10.92	0.05	10.85	10.93	10.97	2
Switzerland	16.56	1.19	14.64	16.58	18.37	1
United Kingdom	8.12	0.27	7.83	8.00	8.68	0

Now we have something useful, something you could put in a report, for example. Every number is rounded to a reasonable number of digits, and there is a useful caption.

```
kable(Year_SUMMARY,
      digits = 2,
      caption = 'Summary of Practicing nurses per 1,000 population, by Year - Source OECD https://stats
```

Table 3: Summary of Practicing nurses per 1,000 population, by Year - Source OECD https://stats.oecd.org

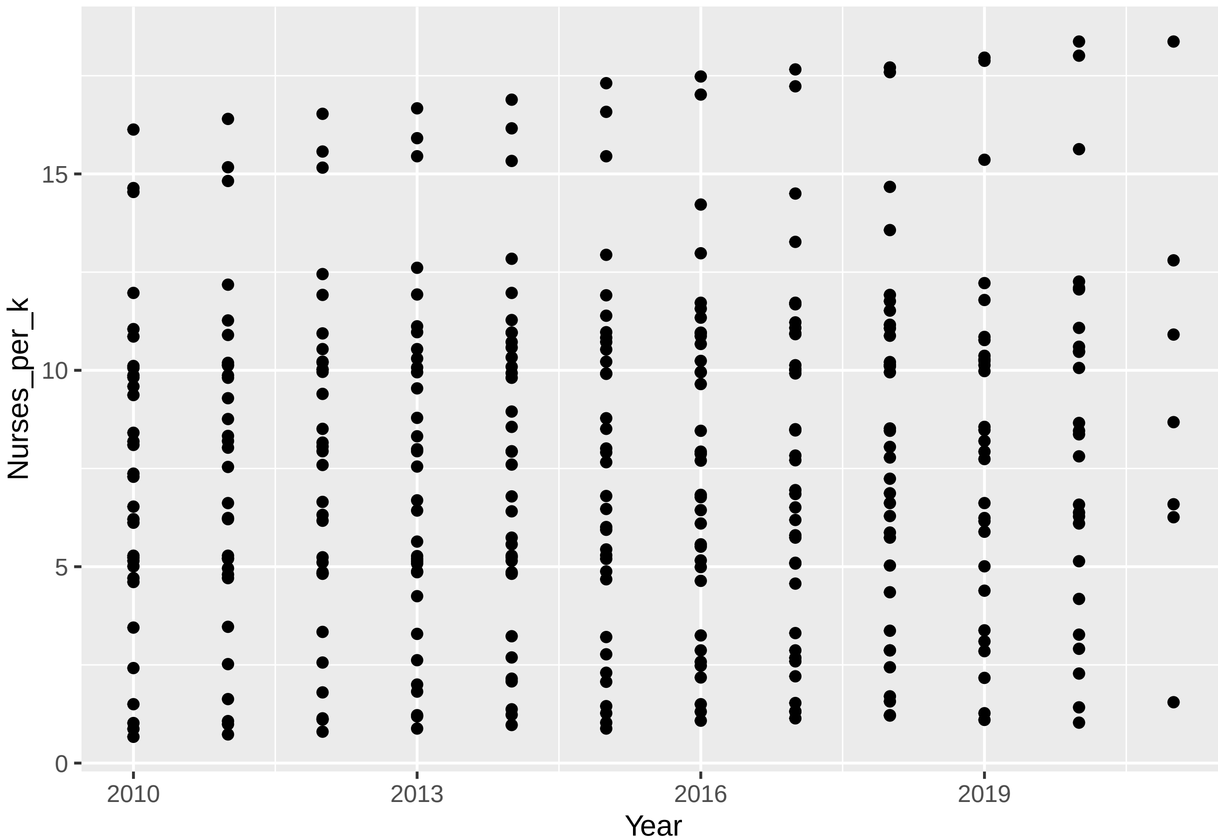
Year	Mean	SD	Min	Median	Max	NA_Count
2010	7.38	4.04	0.67	7.33	16.13	6
2011	7.52	4.11	0.73	7.78	16.40	6
2012	7.55	4.28	0.80	7.94	16.53	5
2013	7.42	4.28	0.88	7.12	16.67	4
2014	7.78	4.25	0.97	7.93	16.89	3
2015	7.58	4.42	0.88	7.66	17.31	3
2016	7.80	4.30	1.08	7.78	17.48	2
2017	7.63	4.44	1.14	7.33	17.66	2
2018	7.95	4.49	1.21	7.92	17.71	4
2019	8.17	4.46	1.10	8.20	17.96	9
2020	8.46	4.70	1.03	8.41	18.37	12
2021	9.31	5.39	1.55	8.68	18.37	31

Note how little we had to change to get a second good table.

7.2 Pictures

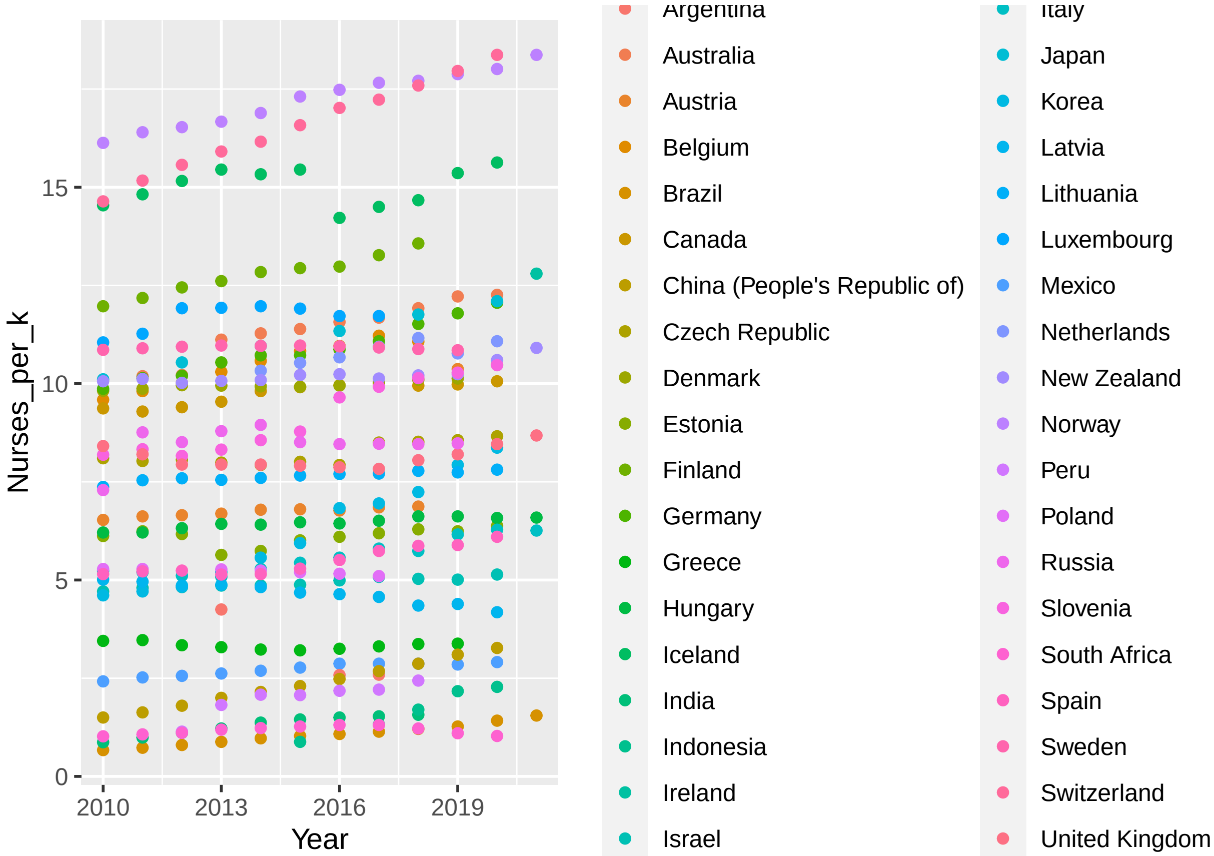
You might also want to make a picture, or two. It can be easier to get your message across in a graphic than a table. The way graphs are done in R looks not unlike the technique we've been showing for doing calculations. You start with something very simple, and go on to make it more complex, and better looking. I'm going to show a series of graphs, but in reality you would start with one, and then edit it to make it better.

```
library(ggplot2)
ggplot(data = Nurses, # Data to graph
       aes(x = Year,
           y = Nurses_per_k)) + # What we are graphing to what
       geom_point() # How we are graphing it (points)
```



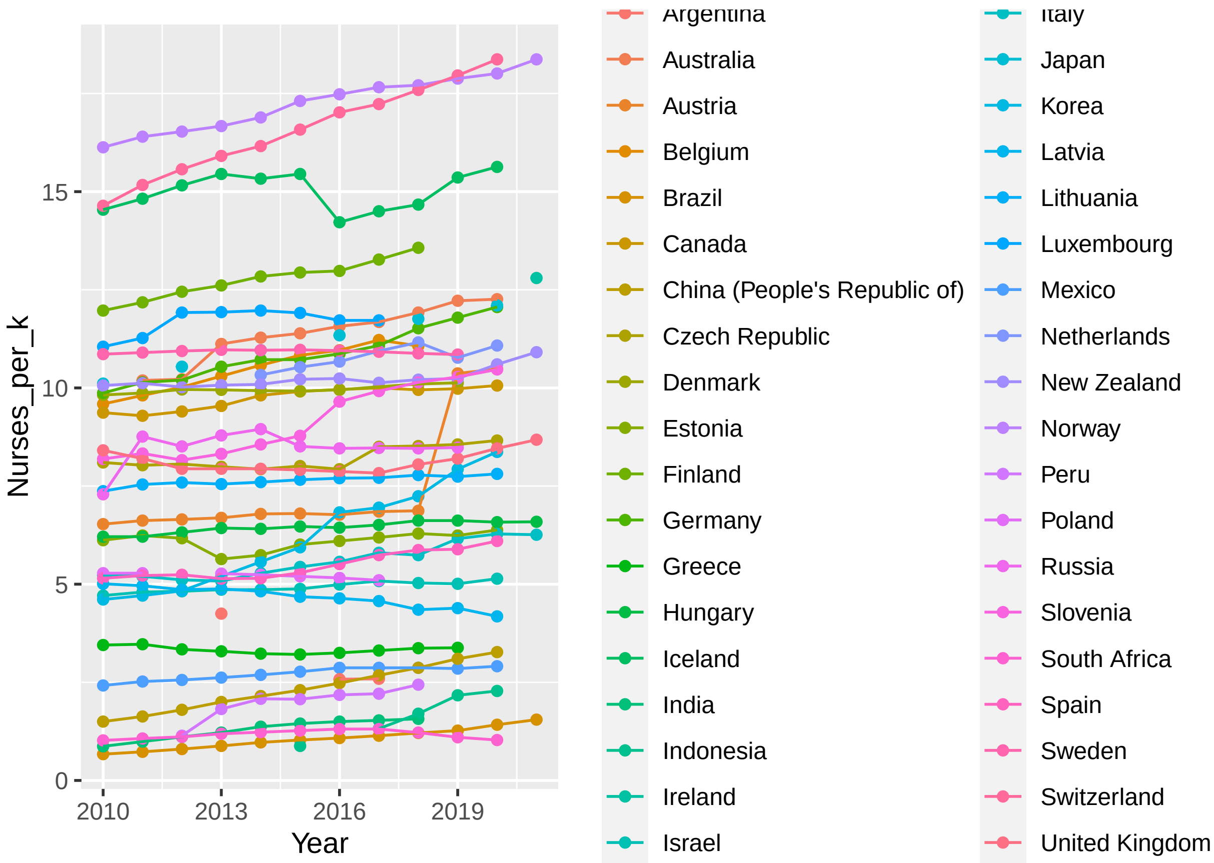
This is not pretty, but not wholly useless either. It tells us quite a lot about what is going on.

```
ggplot(data = Nurses, # Data to graph
       aes(x = Year,
           y = Nurses_per_k,
           colour = Country)) + # What we are graphing to what
       geom_point() # How we are graphing it (points)
```



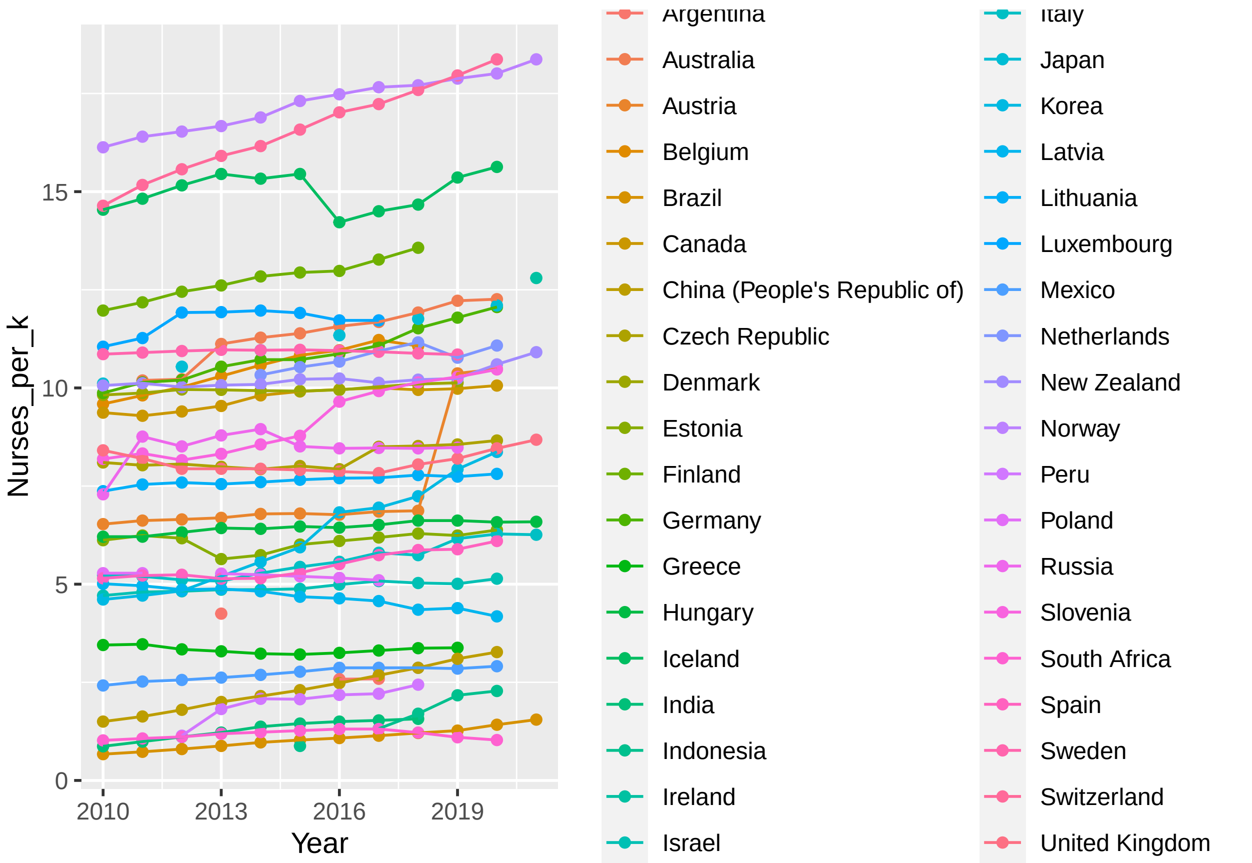
This is more useful - every country now has a colour, and we can see some patterns.

```
ggplot(data = Nurses, # Data to graph
       aes(x = Year,
           y = Nurses_per_k,
           colour = Country)) + # What we are graphing to what
       geom_point() +
       geom_line()    # How we are graphing it (points and lines)
```



We need to tell the software we want one line per country, and not, as we have, one line per observation.

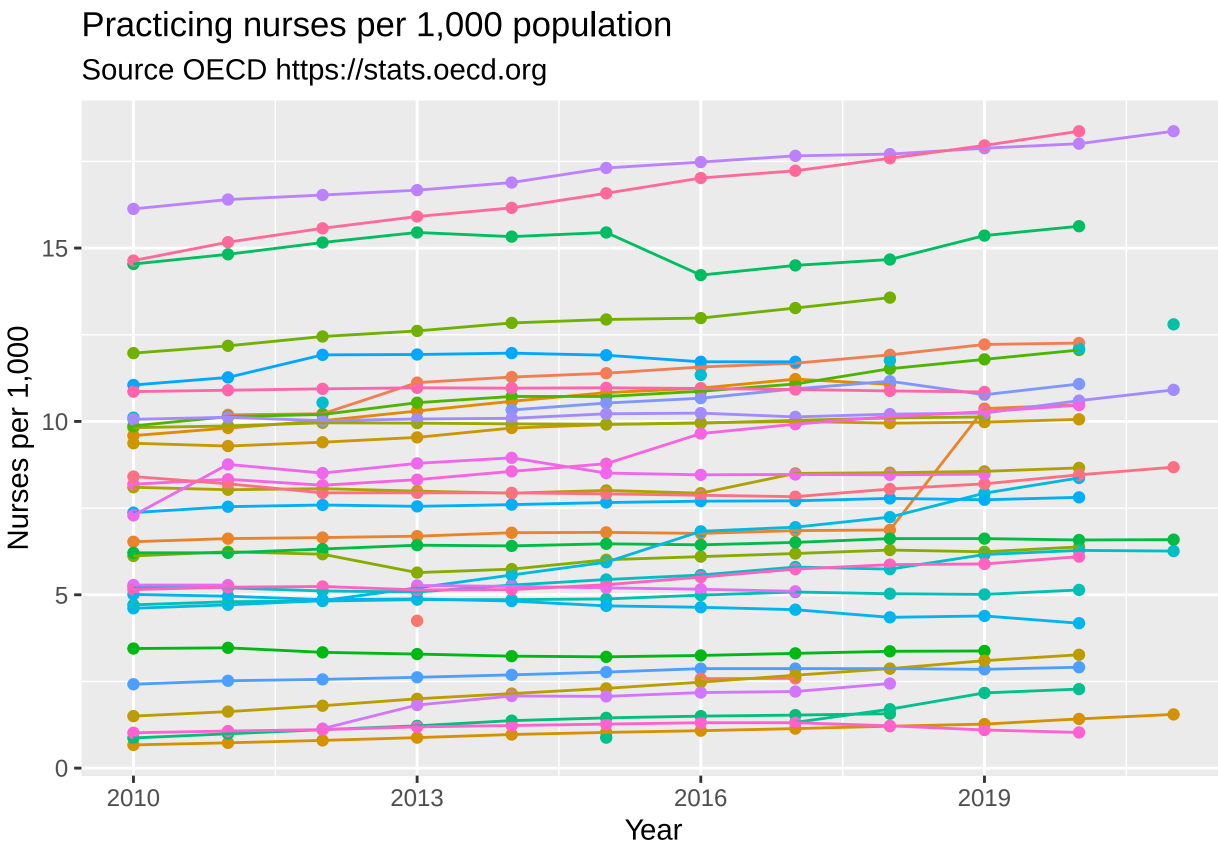
```
ggplot(data = Nurses, # Data to graph
       aes(x = Year,
           y = Nurses_per_k,
           colour = Country,
           group = Country)) + # What we are graphing to what
       geom_point() + # How we are graphing it
       geom_line() #  points and lines, one line per country
```



There are **way** too many countries for this to be a sensible approach. Two-thirds of our space is filled with a list of colours and countries (called a guide or a key). We can do better, but this isn't especially simple.

So, first remove the guide, and add a title, and a better name for the y-axis.

```
ggplot(data = Nurses, # Data to graph
       aes(x = Year,
           y = Nurses_per_k,
           colour = Country,
           group = Country)) + # What we are graphing to what
       geom_point() + # How we are graphing it
       geom_line() + #  points and lines, one line per country
       guides(color = "none") + # Completely removes the guide
       ggtitle('Practicing nurses per 1,000 population',
               subtitle = 'Source OECD https://stats.oecd.org') +
  ylab('Nurses per 1,000')
```



This is quite nice, but does not tell you which country is which.

The basic idea for labelling the lines is to identify the end point of each line (the one with the highest value of year), and put a text with the country name there. In many situations you could replace the longer names with shorter abbreviations.

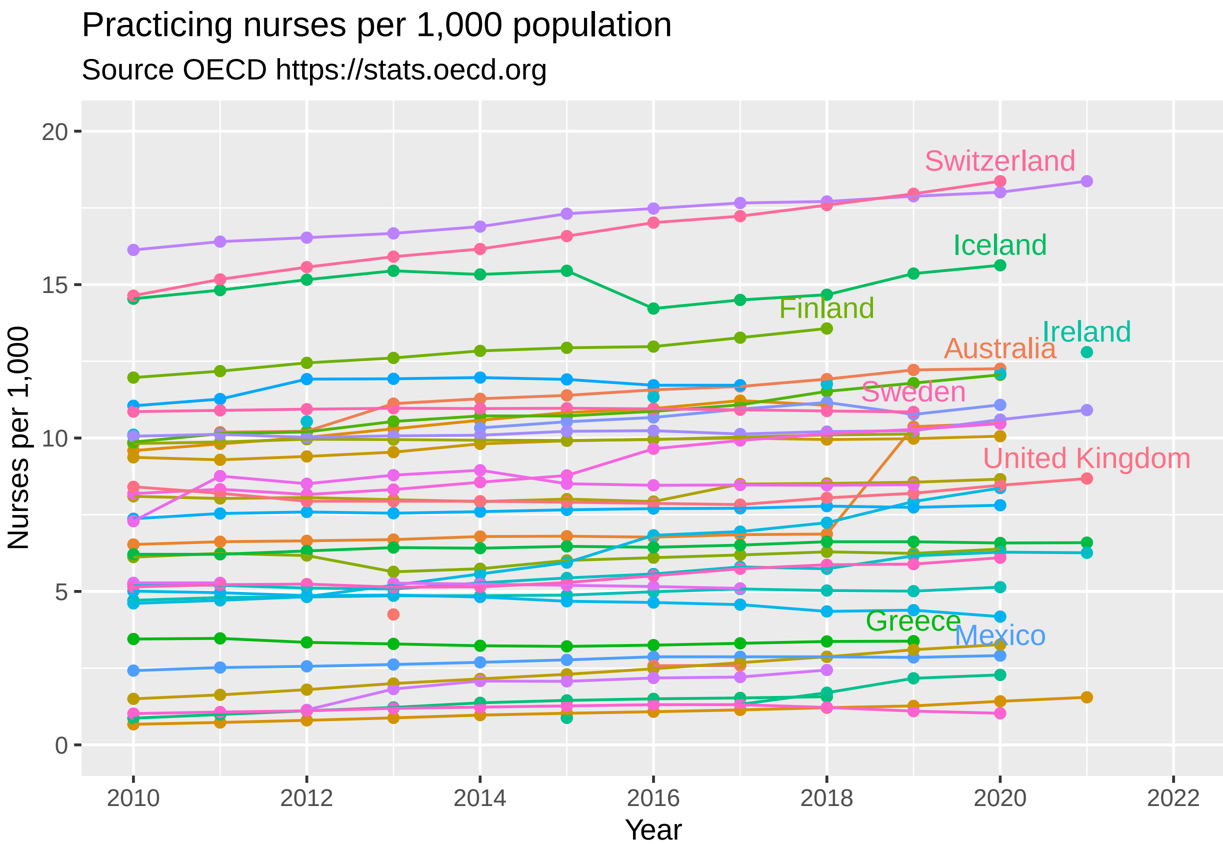
There's no easy way to do this, but it's far from impossible. I attach one way of doing it for your interest. This shows how the plotting and data handling facilities of R can be combined to great effect. I don't expect you to follow it, but I'm happy to go through it with anyone interested after the session.

```
library(scales, quietly = TRUE) # Allows us to select pretty breaks

g <- # We're saving the graph and calling it g
  ggplot(data = Nurses, # Data to graph
       aes(x = Year,
           y = Nurses_per_k,
           colour = Country,
           group = Country)) + # What we are graphing to what
       geom_point() + # How we are graphing it
       geom_line()  + #  points and lines, one line per country
       guides(color = "none") + # Completely removes the guide
       ggtitle('Practicing nurses per 1,000 population',
               subtitle = 'Source OECD https://stats.oecd.org') +
       ylab('Nurses per 1,000') + # Y- axis label
       scale_x_continuous(breaks = breaks_extended(8)) + # Nice labels for years
       coord_cartesian(xlim = c(2010,2022),
                       ylim = c(0,20)) + # Make space for the text labels
# Add the text labels
         geom_text(  # Add text labels
```

```
      data = Nurses %>% # Make a new data set from Nurses
            group_by(Country) %>%
            filter(!is.na(Nurses_per_k)) %>% #Remove missing values
            filter(Country %in% # Select countries to label
                      c('Australia', 'Ireland',
                        'Sweden', 'United Kingdom',
                        'Greece', 'Iceland', 'Finland',
                        'Mexico', 'Switzerland')) %>%
            arrange(desc(Year)) %>% # Put in order of year
            slice(1), # Take off the highest year for which there is data
        aes(x = Year, label=Country), # Draw the labels
          hjust = 0.50, vjust = -0.5) # Adjust their position

g # Now we display the graph
```




```
# Now we save the graph. you can also save from RStudio, which can be easier.

ggsave('images/Saved_Graph.png', g,
       units = 'cm',
       width = 5, height = 4,
       dpi = 1200)
```

This is a graph you could send off to a journal.

7.3 Does it change over time?

Looking at the picture, it's hard to say - but not much anyway. We can use a tool called **linear regression** to see what is going on. To do this we use the *lm()* function.

```
m1 <-  lm(Nurses_per_k ~ Year, data = Nurses)
  summary(m1)
```

```
##
## Call:
## lm(formula = Nurses_per_k ~ Year, data = Nurses)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.2098 -2.8413  0.0158  2.8158 10.1302
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -186.52888  143.92151  -1.296    0.196
## Year           0.09642    0.07142   1.350    0.178
##
## Residual standard error: 4.304 on 367 degrees of freedom
##   (87 observations deleted due to missingness)
## Multiple R-squared:  0.004941,   Adjusted R-squared:  0.00223
## F-statistic: 1.822 on 1 and 367 DF,  p-value: 0.1779
```

```
  confint(m1)
```

```
##                    2.5 %     97.5 %
## (Intercept) -469.5431702 96.4854161
## Year          -0.0440319  0.2368722
```

Looking at year alone, there isn't much sign of a change over year. The estimate is 0.1, with a confidence interval from -0.04 to 0.25, nurses per 100,000 population per year. A confidence interval is, roughly, the range within which we are pretty sure the real value lies. A sensible way to interpret this is that the change could be zero, but might be a small positive change.

```
m2 <-  lm(Nurses_per_k ~ Year + Country, data = Nurses)
  summary(m2)
```

```
##
## Call:
## lm(formula = Nurses_per_k ~ Year + Country, data = Nurses)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.37134 -0.24575 -0.00235  0.22592  2.52635
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        -2.195e+02  1.555e+01 -14.117  < 2e-16 ***
## Year                                1.105e-01  7.713e-03  14.321  < 2e-16 ***
## CountryAustralia                    8.227e+00  2.979e-01  27.619  < 2e-16 ***
## CountryAustria                      4.299e+00  2.947e-01  14.585  < 2e-16 ***
## CountryBelgium                      7.494e+00  3.018e-01  24.829  < 2e-16 ***
## CountryBrazil                      -2.096e+00  2.921e-01  -7.176 4.77e-12 ***
## CountryCanada                       6.649e+00  2.947e-01  22.558  < 2e-16 ***
```

```
## CountryChina (People's Republic of) -7.595e-01  2.947e-01  -2.577 0.010397 *
## CountryCzech Republic               5.105e+00  2.947e-01  17.321  < 2e-16 ***
## CountryDenmark                      6.918e+00  2.979e-01  23.221  < 2e-16 ***
## CountryEstonia                      2.999e+00  2.947e-01  10.174  < 2e-16 ***
## CountryFinland                      9.764e+00  3.018e-01  32.349  < 2e-16 ***
## CountryGermany                      7.761e+00  2.947e-01  26.334  < 2e-16 ***
## CountryGreece                       2.820e-01  2.979e-01   0.947 0.344483
## CountryHungary                      3.292e+00  2.921e-01  11.272  < 2e-16 ***
## CountryIceland                      1.191e+01  2.947e-01  40.405  < 2e-16 ***
## CountryIndia                       -1.703e+00  3.018e-01  -5.641 3.63e-08 ***
## CountryIndonesia                   -1.742e+00  3.310e-01  -5.264 2.54e-07 ***
## CountryIreland                      9.034e+00  5.243e-01  17.231  < 2e-16 ***
## CountryIsrael                       1.822e+00  2.947e-01   6.183 1.86e-09 ***
## CountryItaly                        2.437e+00  2.921e-01   8.345 1.97e-15 ***
## CountryJapan                        8.032e+00  3.200e-01  25.102  < 2e-16 ***
## CountryKorea                        3.095e+00  2.947e-01  10.501  < 2e-16 ***
## CountryLatvia                       1.564e+00  2.947e-01   5.307 2.05e-07 ***
## CountryLithuania                    4.538e+00  2.947e-01  15.396  < 2e-16 ***
## CountryLuxembourg                   8.749e+00  3.067e-01  28.530  < 2e-16 ***
## CountryMexico                      -3.805e-01  2.947e-01  -1.291 0.197656
## CountryNetherlands                  7.459e+00  3.125e-01  23.868  < 2e-16 ***
## CountryNew Zealand                  7.084e+00  2.921e-01  24.254  < 2e-16 ***
## CountryNorway                       1.409e+01  2.921e-01  48.258  < 2e-16 ***
## CountryPeru                        -1.112e+00  3.123e-01  -3.560 0.000425 ***
## CountryPoland                       2.257e+00  3.125e-01   7.224 3.52e-12 ***
## CountryRussia                       5.420e+00  2.979e-01  18.192  < 2e-16 ***
## CountrySlovenia                     6.060e+00  2.947e-01  20.563  < 2e-16 ***
## CountrySouth Africa                -1.934e+00  2.947e-01  -6.562 2.05e-10 ***
## CountrySpain                        2.379e+00  2.947e-01   8.071 1.32e-14 ***
## CountrySweden                       7.872e+00  2.979e-01  26.423  < 2e-16 ***
## CountrySwitzerland                  1.346e+01  2.947e-01  45.671  < 2e-16 ***
## CountryUnited Kingdom               4.961e+00  2.921e-01  16.984  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4525 on 330 degrees of freedom
##   (87 observations deleted due to missingness)
## Multiple R-squared:  0.9901, Adjusted R-squared:  0.989
## F-statistic: 869.6 on 38 and 330 DF,  p-value: < 2.2e-16
```

```
confint(m2)
```

```
##                                           2.5 %       97.5 %
## (Intercept)                         -250.05441167 -188.8879040
## Year                                   0.09528559    0.1256319
## CountryAustralia                       7.64064387    8.8125366
## CountryAustria                         3.71885362    4.8784219
## CountryBelgium                         6.90019480    8.0876951
## CountryBrazil                         -2.67047767   -1.5213419
## CountryCanada                          6.06885362    7.2284219
## CountryChina (People's Republic of)   -1.33932820   -0.1797599
## CountryCzech Republic                  4.52521726    5.6847855
## CountryDenmark                         6.33197164    7.5041262
## CountryEstonia                         2.41885362    3.5784219
## CountryFinland                         9.17019480   10.3576951
```

```
## CountryGermany                          7.18158089    8.3411492
## CountryGreece                          -0.30402836    0.8681262
## CountryHungary                          2.71785567    3.8669914
## CountryIceland                         11.32885362   12.4884219
## CountryIndia                           -2.29647186   -1.1089715
## CountryIndonesia                       -2.39358448   -1.0913452
## CountryIreland                          8.00266880   10.0654656
## CountryIsrael                           1.24248998    2.4020583
## CountryItaly                            1.86285567    3.0119914
## CountryJapan                            7.40239778    8.6612414
## CountryKorea                            2.51521726    3.6747855
## CountryLatvia                           0.98430817    2.1438764
## CountryLithuania                        3.95794453    5.1175128
## CountryLuxembourg                       8.14550992    9.3520054
## CountryMexico                          -0.96023729    0.1993310
## CountryNetherlands                      6.84400596    8.0735126
## CountryNew Zealand                      6.50952233    7.6586581
## CountryNorway                          13.52035567   14.6694914
## CountryPeru                            -1.72600562   -0.4974981
## CountryPoland                           1.64268536    2.8721334
## CountryRussia                           4.83397164    6.0061262
## CountrySlovenia                         5.48067180    6.6402401
## CountrySouth Africa                    -2.51387365   -1.3543054
## CountrySpain                            1.79885362    2.9584219
## CountrySweden                           7.28597164    8.4581262
## CountrySwitzerland                     12.88067180   14.0402401
## CountryUnited Kingdom                   4.38618900    5.5353248
```

Looking at the graph, the effect of country is much greater than any effect of year. We can fit a country level effect, by adding *+ Country* to the model. This gives a slightly different, and more precise estimate of the effect of year - 0.11, with a confidence interval from 0.09 to 0.13, nurses per 100,000 population. This suggests that there is a real, but modest, effect of time.

You can also install the *lme4* package and use the *lmer()* function to fit what's called a multi-level model to this type of data where you have loads of groups. Again, we're not going to cover this further today, but note that the estimate of year is about the same.

```r
require(lme4)

m3 <-  lmer(Nurses_per_k ~ Year + (1|Country), data = Nurses)
  summary(m3)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Nurses_per_k ~ Year + (1 | Country)
##    Data: Nurses
##
## REML criterion at convergence: 722.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.0111 -0.5391 -0.0038  0.4934  5.5824
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Country  (Intercept) 18.8585  4.3426
##  Residual              0.2047  0.4525
```

```
## Number of obs: 369, groups:  Country, 38
##
## Fixed effects:
##                Estimate Std. Error t value
## (Intercept) -2.149e+02  1.556e+01  -13.81
## Year         1.105e-01  7.712e-03   14.33
##
## Correlation of Fixed Effects:
##      (Intr)
## Year -0.999
```

```
confint(m3)
```

```
##                     2.5 %       97.5 %
## .sig01         3.47491443    5.4653625
## .sigma         0.41944886    0.4885262
## (Intercept) -245.45648870 -184.3875763
## Year           0.09535282    0.1256272
```

Overall, it would be unwise to interpret this too strongly. It would be best to report the effect of year, and to note that this effect is much smaller than the between-countries effect.

You might be tempted to fit a model with an interaction term here.

```
m4 <-  lm(Nurses_per_k ~ Year + Country + Year:Country, data = Nurses)
#Identical to
#m4 <-  lm(Nurses_per_k ~ Year * Country, data = Nurses)
  summary(m4)
```

```
##
## Call:
## lm(formula = Nurses_per_k ~ Year + Country + Year:Country, data = Nurses)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50573 -0.06782  0.00237  0.07600  1.66964
##
## Coefficients: (1 not defined because of singularities)
##                                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                               9.054e+02  1.810e+02   5.001 9.82e-07
## Year                                     -4.477e-01  8.983e-02  -4.984 1.07e-06
## CountryAustralia                         -1.358e+03  1.903e+02  -7.137 7.47e-12
## CountryAustria                           -1.552e+03  1.880e+02  -8.255 5.21e-15
## CountryBelgium                           -1.316e+03  1.937e+02  -6.798 5.91e-11
## CountryBrazil                            -1.055e+03  1.864e+02  -5.659 3.62e-08
## CountryCanada                            -1.059e+03  1.880e+02  -5.633 4.14e-08
## CountryChina (People's Republic of)      -1.263e+03  1.880e+02  -6.715 9.71e-11
## CountryCzech Republic                    -1.031e+03  1.880e+02  -5.485 8.94e-08
## CountryDenmark                           -9.533e+02  1.903e+02  -5.009 9.44e-07
## CountryEstonia                           -9.564e+02  1.880e+02  -5.087 6.51e-07
## CountryFinland                           -1.264e+03  1.937e+02  -6.526 2.95e-10
## CountryGermany                           -1.311e+03  1.880e+02  -6.973 2.06e-11
## CountryGreece                            -8.823e+02  1.903e+02  -4.636 5.34e-06
## CountryHungary                           -9.753e+02  1.864e+02  -5.231 3.21e-07
## CountryIceland                           -9.477e+02  1.880e+02  -5.040 8.14e-07
## CountryIndia                             -1.086e+03  1.937e+02  -5.610 4.68e-08
```

```
## CountryIndonesia                         -1.509e+03  2.281e+02  -6.617 1.74e-10
## CountryIreland                            1.220e+01  5.936e-01  20.547  < 2e-16
## CountryIsrael                            -9.772e+02  1.880e+02  -5.197 3.79e-07
## CountryItaly                             -1.135e+03  1.864e+02  -6.088 3.56e-09
## CountryJapan                             -1.297e+03  1.919e+02  -6.758 7.51e-11
## CountryKorea                             -1.699e+03  1.880e+02  -9.036  < 2e-16
## CountryLatvia                            -7.403e+02  1.880e+02  -3.937 0.000103
## CountryLithuania                         -9.708e+02  1.880e+02  -5.163 4.48e-07
## CountryLuxembourg                        -1.060e+03  1.988e+02  -5.333 1.93e-07
## CountryMexico                            -1.001e+03  1.880e+02  -5.325 2.01e-07
## CountryNetherlands                       -1.127e+03  2.072e+02  -5.437 1.14e-07
## CountryNew Zealand                       -1.008e+03  1.864e+02  -5.408 1.32e-07
## CountryNorway                            -1.285e+03  1.864e+02  -6.890 3.40e-11
## CountryPeru                              -1.247e+03  2.072e+02  -6.021 5.15e-09
## CountryPoland                            -8.505e+02  1.999e+02  -4.254 2.82e-05
## CountryRussia                            -9.821e+02  1.903e+02  -5.161 4.53e-07
## CountrySlovenia                          -1.435e+03  1.880e+02  -7.633 3.22e-13
## CountrySouth Africa                      -9.192e+02  1.880e+02  -4.889 1.67e-06
## CountrySpain                             -1.099e+03  1.880e+02  -5.846 1.34e-08
## CountrySweden                            -8.898e+02  1.903e+02  -4.676 4.47e-06
## CountrySwitzerland                       -1.610e+03  1.880e+02  -8.562 6.28e-16
## CountryUnited Kingdom                    -9.488e+02  1.864e+02  -5.089 6.44e-07
## Year:CountryAustralia                     6.781e-01  9.443e-02   7.180 5.71e-12
## Year:CountryAustria                       7.723e-01  9.330e-02   8.278 4.45e-15
## Year:CountryBelgium                       6.570e-01  9.610e-02   6.837 4.68e-11
## Year:CountryBrazil                        5.225e-01  9.251e-02   5.648 3.84e-08
## Year:CountryCanada                        5.289e-01  9.330e-02   5.668 3.44e-08
## Year:CountryChina (People's Republic of)  6.261e-01  9.330e-02   6.711 9.94e-11
## Year:CountryCzech Republic                5.142e-01  9.330e-02   5.512 7.78e-08
## Year:CountryDenmark                       4.764e-01  9.443e-02   5.045 7.95e-07
## Year:CountryEstonia                       4.761e-01  9.330e-02   5.102 6.03e-07
## Year:CountryFinland                       6.320e-01  9.610e-02   6.577 2.19e-10
## Year:CountryGermany                       6.544e-01  9.330e-02   7.014 1.60e-11
## Year:CountryGreece                        4.379e-01  9.443e-02   4.637 5.33e-06
## Year:CountryHungary                       4.856e-01  9.251e-02   5.249 2.94e-07
## Year:CountryIceland                       4.762e-01  9.330e-02   5.103 6.00e-07
## Year:CountryIndia                         5.382e-01  9.610e-02   5.600 4.91e-08
## Year:CountryIndonesia                     7.477e-01  1.131e-01   6.610 1.80e-10
## Year:CountryIreland                              NA         NA      NA       NA
## Year:CountryIsrael                        4.858e-01  9.330e-02   5.207 3.62e-07
## Year:CountryItaly                         5.644e-01  9.251e-02   6.101 3.31e-09
## Year:CountryJapan                         6.475e-01  9.523e-02   6.800 5.84e-11
## Year:CountryKorea                         8.447e-01  9.330e-02   9.053  < 2e-16
## Year:CountryLatvia                        3.681e-01  9.330e-02   3.945 9.99e-05
## Year:CountryLithuania                     4.840e-01  9.330e-02   5.187 3.99e-07
## Year:CountryLuxembourg                    5.304e-01  9.866e-02   5.376 1.55e-07
## Year:CountryMexico                        4.966e-01  9.330e-02   5.323 2.03e-07
## Year:CountryNetherlands                   5.627e-01  1.028e-01   5.474 9.45e-08
## Year:CountryNew Zealand                   5.038e-01  9.251e-02   5.446 1.09e-07
## Year:CountryNorway                        6.444e-01  9.251e-02   6.966 2.15e-11
## Year:CountryPeru                          6.184e-01  1.028e-01   6.016 5.31e-09
## Year:CountryPoland                        4.230e-01  9.921e-02   4.264 2.71e-05
## Year:CountryRussia                        4.900e-01  9.443e-02   5.189 3.95e-07
## Year:CountrySlovenia                      7.152e-01  9.330e-02   7.666 2.61e-13
```

```
## Year:CountrySouth Africa                 4.551e-01  9.330e-02   4.878 1.76e-06
## Year:CountrySpain                        5.466e-01  9.330e-02   5.858 1.25e-08
## Year:CountrySweden                       4.454e-01  9.443e-02   4.717 3.71e-06
## Year:CountrySwitzerland                  8.056e-01  9.330e-02   8.634 3.81e-16
## Year:CountryUnited Kingdom               4.733e-01  9.251e-02   5.116 5.65e-07
##
## (Intercept)                             ***
## Year                                    ***
## CountryAustralia                        ***
## CountryAustria                          ***
## CountryBelgium                          ***
## CountryBrazil                           ***
## CountryCanada                           ***
## CountryChina (People's Republic of)     ***
## CountryCzech Republic                   ***
## CountryDenmark                          ***
## CountryEstonia                          ***
## CountryFinland                          ***
## CountryGermany                          ***
## CountryGreece                           ***
## CountryHungary                          ***
## CountryIceland                          ***
## CountryIndia                            ***
## CountryIndonesia                        ***
## CountryIreland                          ***
## CountryIsrael                           ***
## CountryItaly                            ***
## CountryJapan                            ***
## CountryKorea                            ***
## CountryLatvia                           ***
## CountryLithuania                        ***
## CountryLuxembourg                       ***
## CountryMexico                           ***
## CountryNetherlands                      ***
## CountryNew Zealand                      ***
## CountryNorway                           ***
## CountryPeru                             ***
## CountryPoland                           ***
## CountryRussia                           ***
## CountrySlovenia                         ***
## CountrySouth Africa                     ***
## CountrySpain                            ***
## CountrySweden                           ***
## CountrySwitzerland                      ***
## CountryUnited Kingdom                   ***
## Year:CountryAustralia                   ***
## Year:CountryAustria                     ***
## Year:CountryBelgium                     ***
## Year:CountryBrazil                      ***
## Year:CountryCanada                      ***
## Year:CountryChina (People's Republic of) ***
## Year:CountryCzech Republic              ***
## Year:CountryDenmark                     ***
## Year:CountryEstonia                     ***
```

```
## Year:CountryFinland                         ***
## Year:CountryGermany                         ***
## Year:CountryGreece                          ***
## Year:CountryHungary                         ***
## Year:CountryIceland                         ***
## Year:CountryIndia                           ***
## Year:CountryIndonesia                       ***
## Year:CountryIreland
## Year:CountryIsrael                          ***
## Year:CountryItaly                           ***
## Year:CountryJapan                           ***
## Year:CountryKorea                           ***
## Year:CountryLatvia                          ***
## Year:CountryLithuania                       ***
## Year:CountryLuxembourg                      ***
## Year:CountryMexico                          ***
## Year:CountryNetherlands                     ***
## Year:CountryNew Zealand                     ***
## Year:CountryNorway                          ***
## Year:CountryPeru                            ***
## Year:CountryPoland                          ***
## Year:CountryRussia                          ***
## Year:CountrySlovenia                        ***
## Year:CountrySouth Africa                    ***
## Year:CountrySpain                           ***
## Year:CountrySweden                          ***
## Year:CountrySwitzerland                     ***
## Year:CountryUnited Kingdom                  ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2645 on 294 degrees of freedom
##   (87 observations deleted due to missingness)
## Multiple R-squared:  0.997,  Adjusted R-squared:  0.9962
## F-statistic:  1316 on 74 and 294 DF,  p-value: < 2.2e-16
```

```
confint(m4)
```

```
##                                             2.5 %        97.5 %
## (Intercept)                           549.0939208  1261.6845408
## Year                                   -0.6244845    -0.2709001
## CountryAustralia                    -1732.8476635  -983.7566162
## CountryAustria                      -1922.1903433 -1182.0690273
## CountryBelgium                      -1697.6282286  -935.3715663
## CountryBrazil                       -1422.0014823  -688.1313149
## CountryCanada                       -1429.2794342  -689.1581182
## CountryChina (People's Republic of) -1632.6921615  -892.5708455
## CountryCzech Republic               -1401.3307979  -661.2094818
## CountryDenmark                      -1327.8221807  -578.7664627
## CountryEstonia                      -1326.5007979  -586.3794818
## CountryFinland                      -1645.0082286  -882.7515663
## CountryGermany                      -1681.1407979  -941.0194818
## CountryGreece                       -1256.8083625  -507.7526445
## CountryHungary                      -1342.1946875  -608.3245200
## CountryIceland                      -1317.7739797  -577.6526637
```

```
## CountryIndia                                -1467.4945619  -705.2378996
## CountryIndonesia                            -1957.9092990 -1060.2091625
## CountryIreland                                 11.0286669    13.3651792
## CountryIsrael                               -1347.2776161  -607.1563000
## CountryItaly                                -1502.0345826  -768.1644151
## CountryJapan                                -1674.6677769  -919.2649704
## CountryKorea                                -2069.2067070 -1329.0853909
## CountryLatvia                               -1110.3153433  -370.1940273
## CountryLithuania                            -1340.8985252  -600.7772091
## CountryLuxembourg                           -1451.5684240  -669.0238471
## CountryMexico                               -1371.2789797  -631.1576637
## CountryNetherlands                          -1534.3656069  -718.7571404
## CountryNew Zealand                          -1375.1894194  -641.3192519
## CountryNorway                               -1651.5466105  -917.6764431
## CountryPeru                                 -1655.0956591  -839.6785168
## CountryPoland                               -1243.9373895  -457.0315068
## CountryRussia                               -1356.6685443  -607.6128263
## CountrySlovenia                             -1805.3903433 -1065.2690273
## CountrySouth Africa                         -1289.3017070  -549.1803909
## CountrySpain                                -1469.2698888  -729.1485728
## CountrySweden                               -1264.3576353  -515.3019172
## CountrySwitzerland                          -1980.0730706 -1239.9517546
## CountryUnited Kingdom                       -1315.7202003  -581.8500328
## Year:CountryAustralia                           0.4922096     0.8639023
## Year:CountryAustria                             0.5887040     0.9559534
## Year:CountryBelgium                             0.4678956     0.8461556
## Year:CountryBrazil                              0.3404118     0.7045532
## Year:CountryCanada                              0.3452494     0.7124988
## Year:CountryChina (People's Republic of)        0.4425222     0.8097716
## Year:CountryCzech Republic                      0.3306131     0.6978625
## Year:CountryDenmark                             0.2905732     0.6622660
## Year:CountryEstonia                             0.2924312     0.6596806
## Year:CountryFinland                             0.4428956     0.8211556
## Year:CountryGermany                             0.4707949     0.8380443
## Year:CountryGreece                              0.2520277     0.6237205
## Year:CountryHungary                             0.3034887     0.6676302
## Year:CountryIceland                             0.2925222     0.6597716
## Year:CountryIndia                               0.3490623     0.7273223
## Year:CountryIndonesia                           0.5250755     0.9703091
## Year:CountryIreland                                    NA            NA
## Year:CountryIsrael                              0.3021585     0.6694079
## Year:CountryItaly                               0.3823698     0.7465113
## Year:CountryJapan                               0.4601323     0.8349666
## Year:CountryKorea                               0.6610676     1.0283170
## Year:CountryLatvia                              0.1844312     0.5516806
## Year:CountryLithuania                           0.3003403     0.6675897
## Year:CountryLuxembourg                          0.3362526     0.7246082
## Year:CountryMexico                              0.3129767     0.6802261
## Year:CountryNetherlands                         0.3603812     0.7650034
## Year:CountryNew Zealand                         0.3217405     0.6858819
## Year:CountryNorway                              0.4623349     0.8264763
## Year:CountryPeru                                0.4160955     0.8207177
## Year:CountryPoland                              0.2277605     0.6182763
## Year:CountryRussia                              0.3041489     0.6758417
```

```
## Year:CountrySlovenia                              0.5316131     0.8988625
## Year:CountrySouth Africa                          0.2715222     0.6387716
## Year:CountrySpain                                 0.3629767     0.7302261
## Year:CountrySweden                                0.2595429     0.6312357
## Year:CountrySwitzerland                           0.6219767     0.9892261
## Year:CountryUnited Kingdom                        0.2911810     0.6553225
```

If you do, the results are evidently meaningless. This is obvious here, but might not be so obvious in a more complex model, or if you had started out by fitting a more complex model. This is because the regression is using all the variation in the data, because there are no other data. You have a dataset with two variables and an outcome. There isn't enough variation for it to make sense to look at an interaction.

8 Another example - BigCities health data

BigCities - details are at https://bigcitieshealthdata.org/

This is a complicated data set to explain. It covers 35 of the larger US cities. There is one row per city per year. Each row contains one value - in this file the years of potential life lost per 100,000 people per year, adjusted for differences in the age distribution between the cities. This is a good summary measure of health, and higher is worse, lower is better.

It is broken down by year, city poverty, city size, US region (and State), race, sex and sometimes race and sex combined. Each of these is a variable taking a small number of values, what is sometimes called a categorical variable.

For some there is a natural ordering - time for Year, less and more segregated, less and more poor; for others there is no such ordering - e.g. Sex 'Both', 'Female, 'Male', and the 5 categories of race 'All', 'Asian/PI', 'Black', 'Hispanic' and 'White'. Note that both of these have categories 'Both' and 'All' which combine other categories.

For our next piece of work, we're going to use a different example. These are data from the US Big Cities project, a repository of comparative data from different cities. We're using a subset of this focussed on premature deaths, specifically on Years of Potential Life Lost (YPLL) before age 75.

```r
BigCities <- read_csv('data/BigCitiesHealth_Deaths_Premature_Death.csv')

  str(BigCities) # Check! Always check!
```

```
## spc_tbl_ [5,775 x 31] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ metric_item_label              : chr [1:5775] "Premature Death" "Premature Death" "Premature Deat
##  $ metric_cat_label               : chr [1:5775] "Life Expectancy and Deaths" "Life Expectancy and De
##  $ metric_subcat_label            : chr [1:5775] "Deaths" "Deaths" "Deaths" "Deaths" ...
##  $ metric_item_label_subtitle     : chr [1:5775] "Years of potential life lost before age 75 (per 100
##  $ metric_cat_item_yaxis_label    : chr [1:5775] "Years per 100,000 population aged <75" "Years per 1
##  $ metric_source_desc_label_fn    : chr [1:5775] "National Vital Statistics System (NVSS), Centers fo
##  $ metric_source_desc_label_url_fn: chr [1:5775] "https://www.cdc.gov/nchs/nvss/index.htm" "https://w
##  $ geo_label_city                 : chr [1:5775] "Indianapolis" "Los Angeles" "Washington" "Houston"
##  $ geo_label_state                : chr [1:5775] "IN" "CA" "DC" "TX" ...
##  $ geo_label_citystate            : chr [1:5775] "Indianapolis, IN" "Los Angeles, CA" "Washington, DC
##  $ geo_fips_code                  : chr [1:5775] "1836003" "0644000" "1150000" "4835000" ...
##  $ value                          : num [1:5775] 11358 3853 17094 10254 2444 ...
##  $ date_label                     : num [1:5775] 2012 2010 2018 2013 2019 ...
##  $ geo_label_proxy_or_real        : chr [1:5775] "real" "real" "real" "real" ...
##  $ geo_label_proxy_footnote       : chr [1:5775] NA NA NA NA ...
##  $ geo_fips_desc                  : chr [1:5775] "place" "place" "place" "place" ...
##  $ date_label_proxy_or_real       : chr [1:5775] "real" "real" "real" "real" ...
##  $ date_label_proxy_footnote      : logi [1:5775] NA NA NA NA NA NA ...
```

```
##  $ value_ci_flag_yesno           : chr [1:5775] "yes" "yes" "yes" "yes" ...
##  $ value_95_ci_low               : num [1:5775] 10891 3470 16362 9837 1706 ...
##  $ value_95_ci_high              : num [1:5775] 11826 4237 17826 10670 3182 ...
##  $ value_90_ci_low               : logi [1:5775] NA NA NA NA NA NA ...
##  $ value_90_ci_high              : logi [1:5775] NA NA NA NA NA NA ...
##  $ geo_strata_region             : chr [1:5775] "Midwest" "West" "South" "South" ...
##  $ geo_strata_poverty            : chr [1:5775] "Less poor cities (<20% poor)" "Less poor cities (<
##  $ geo_strata_Population         : chr [1:5775] "Smaller (<1.3 million)" "Largest (>1.3 million)" "
##  $ geo_strata_PopDensity         : chr [1:5775] "Lower pop. density (<10k per sq mi)" "Lower pop. d
##  $ geo_strata_Segregation        : chr [1:5775] "Less Segregated (<50%)" "Highly Segregated (50%+)"
##  $ strata_race_label             : chr [1:5775] "Black" "Asian/PI" "Black" "Black" ...
##  $ strata_sex_label              : chr [1:5775] "Both" "Male" "Male" "Female" ...
##  $ strata_race_sex_label         : chr [1:5775] NA "Asian/PI Male" "Black Male" "Black Female" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   metric_item_label = col_character(),
##   ..   metric_cat_label = col_character(),
##   ..   metric_subcat_label = col_character(),
##   ..   metric_item_label_subtitle = col_character(),
##   ..   metric_cat_item_yaxis_label = col_character(),
##   ..   metric_source_desc_label_fn = col_character(),
##   ..   metric_source_desc_label_url_fn = col_character(),
##   ..   geo_label_city = col_character(),
##   ..   geo_label_state = col_character(),
##   ..   geo_label_citystate = col_character(),
##   ..   geo_fips_code = col_character(),
##   ..   value = col_double(),
##   ..   date_label = col_double(),
##   ..   geo_label_proxy_or_real = col_character(),
##   ..   geo_label_proxy_footnote = col_character(),
##   ..   geo_fips_desc = col_character(),
##   ..   date_label_proxy_or_real = col_character(),
##   ..   date_label_proxy_footnote = col_logical(),
##   ..   value_ci_flag_yesno = col_character(),
##   ..   value_95_ci_low = col_double(),
##   ..   value_95_ci_high = col_double(),
##   ..   value_90_ci_low = col_logical(),
##   ..   value_90_ci_high = col_logical(),
##   ..   geo_strata_region = col_character(),
##   ..   geo_strata_poverty = col_character(),
##   ..   geo_strata_Population = col_character(),
##   ..   geo_strata_PopDensity = col_character(),
##   ..   geo_strata_Segregation = col_character(),
##   ..   strata_race_label = col_character(),
##   ..   strata_sex_label = col_character(),
##   ..   strata_race_sex_label = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

8.1 Looking at a whole dataset

There are several ways to look at a whole dataset, as we discussed before. *summary()* is always available - loaded by default, but not necessarily very helpful.

```r
summary(BigCities) # Not very much use
```

```
##  metric_item_label  metric_cat_label   metric_subcat_label
##  Length:5775        Length:5775        Length:5775
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##  metric_item_label_subtitle metric_cat_item_yaxis_label
##  Length:5775                Length:5775
##  Class :character           Class :character
##  Mode  :character           Mode  :character
##
##
##
##  metric_source_desc_label_fn metric_source_desc_label_url_fn geo_label_city
##  Length:5775                 Length:5775                     Length:5775
##  Class :character            Class :character                Class :character
##  Mode  :character            Mode  :character                Mode  :character
##
##
##
##  geo_label_state    geo_label_citystate geo_fips_code          value
##  Length:5775        Length:5775         Length:5775        Min.   :  874.2
##  Class :character   Class :character    Class :character   1st Qu.: 4148.2
##  Mode  :character   Mode  :character    Mode  :character   Median : 5943.2
##                                                            Mean   : 6888.7
##                                                            3rd Qu.: 8812.9
##                                                            Max.   :23625.6
##    date_label   geo_label_proxy_or_real geo_label_proxy_footnote
##  Min.   :2010   Length:5775             Length:5775
##  1st Qu.:2012   Class :character        Class :character
##  Median :2015   Mode  :character        Mode  :character
##  Mean   :2015
##  3rd Qu.:2018
##  Max.   :2020
##  geo_fips_desc      date_label_proxy_or_real date_label_proxy_footnote
##  Length:5775        Length:5775              Mode:logical
##  Class :character   Class :character         NA's:5775
##  Mode  :character   Mode  :character
##
##
##
##  value_ci_flag_yesno value_95_ci_low value_95_ci_high value_90_ci_low
##  Length:5775         Min.   : -148   Min.   : 1551    Mode:logical
##  Class :character    1st Qu.: 3501   1st Qu.: 4647    NA's:5775
##  Mode  :character    Median : 5420   Median : 6522
##                      Mean   : 6234   Mean   : 7544
##                      3rd Qu.: 8149   3rd Qu.: 9444
##                      Max.   :22651   Max.   :29194
##  value_90_ci_high geo_strata_region  geo_strata_poverty geo_strata_Population
##  Mode:logical     Length:5775        Length:5775        Length:5775
##  NA's:5775        Class :character   Class :character   Class :character
```

```
##                   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  geo_strata_PopDensity geo_strata_Segregation strata_race_label
##  Length:5775           Length:5775            Length:5775
##  Class :character      Class :character       Class :character
##  Mode  :character      Mode  :character       Mode  :character
##
##
##
##  strata_sex_label   strata_race_sex_label
##  Length:5775        Length:5775
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

glimpse(), which is part of the *dplyr* package within the overall *tidyverse* package is more useful, and gives you a feel for the data.

```
glimpse(BigCities) # One way to look at it
```

```
## Rows: 5,775
## Columns: 31
## $ metric_item_label                <chr> "Premature Death", "Premature Death", ~
## $ metric_cat_label                 <chr> "Life Expectancy and Deaths", "Life Ex~
## $ metric_subcat_label              <chr> "Deaths", "Deaths", "Deaths", "Deaths"~
## $ metric_item_label_subtitle       <chr> "Years of potential life lost before a~
## $ metric_cat_item_yaxis_label      <chr> "Years per 100,000 population aged <75~
## $ metric_source_desc_label_fn      <chr> "National Vital Statistics System (NVS~
## $ metric_source_desc_label_url_fn <chr> "https://www.cdc.gov/nchs/nvss/index.h~
## $ geo_label_city                   <chr> "Indianapolis", "Los Angeles", "Washin~
## $ geo_label_state                  <chr> "IN", "CA", "DC", "TX", "NC", "MD", "M~
## $ geo_label_citystate              <chr> "Indianapolis, IN", "Los Angeles, CA",~
## $ geo_fips_code                    <chr> "1836003", "0644000", "1150000", "4835~
## $ value                            <dbl> 11358.189, 3853.326, 17094.173, 10253.~
## $ date_label                       <dbl> 2012, 2010, 2018, 2013, 2019, 2011, 20~
## $ geo_label_proxy_or_real          <chr> "real", "real", "real", "real", "real"~
## $ geo_label_proxy_footnote         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ geo_fips_desc                    <chr> "place", "place", "place", "place", "p~
## $ date_label_proxy_or_real         <chr> "real", "real", "real", "real", "real"~
## $ date_label_proxy_footnote        <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ value_ci_flag_yesno              <chr> "yes", "yes", "yes", "yes", "yes", "ye~
## $ value_95_ci_low                  <dbl> 10890.513, 3469.724, 16361.965, 9837.2~
## $ value_95_ci_high                 <dbl> 11825.866, 4236.928, 17826.380, 10670.~
## $ value_90_ci_low                  <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ value_90_ci_high                 <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ geo_strata_region                <chr> "Midwest", "West", "South", "South", "~
## $ geo_strata_poverty               <chr> "Less poor cities (<20% poor)", "Less ~
## $ geo_strata_Population            <chr> "Smaller (<1.3 million)", "Largest (>1~
## $ geo_strata_PopDensity            <chr> "Lower pop. density (<10k per sq mi)",~
## $ geo_strata_Segregation           <chr> "Less Segregated (<50%)", "Highly Segr~
## $ strata_race_label                <chr> "Black", "Asian/PI", "Black", "Black",~
```

```
## $ strata_sex_label                <chr> "Both", "Male", "Male", "Female", "Fem~
## $ strata_race_sex_label           <chr> NA, "Asian/PI Male", "Black Male", "Bl~
```

For this type of dataset the *describe()* function in the *Hmisc* package is probably the most useful, and the output will repay close study.

```
describe(BigCities) # Most useful
```

```
## BigCities
##
##  31  Variables      5775  Observations
## --------------------------------------------------------------------------------
## metric_item_label
##              n         missing        distinct           value
##           5775               0               1 Premature Death
##
## Value      Premature Death
## Frequency             5775
## Proportion               1
## --------------------------------------------------------------------------------
## metric_cat_label
##                         n                     missing
##                      5775                           0
##                  distinct                       value
##                         1 Life Expectancy and Deaths
##
## Value      Life Expectancy and Deaths
## Frequency                        5775
## Proportion                          1
## --------------------------------------------------------------------------------
## metric_subcat_label
##        n  missing distinct    value
##     5775        0        1   Deaths
##
## Value      Deaths
## Frequency    5775
## Proportion      1
## --------------------------------------------------------------------------------
## metric_item_label_subtitle
##                                                                                n
##                                                                             5775
##                                                                          missing
##                                                                                0
##                                                                         distinct
##                                                                                1
##                                                                            value
## Years of potential life lost before age 75 (per 100,000 population, age-adjusted)
##
## Value      Years of potential life lost before age 75 (per 100,000 population, age-adjusted)
## Frequency                                                5775
## Proportion                                               1
## --------------------------------------------------------------------------------
## metric_cat_item_yaxis_label
##                                     n                              missing
##                                  5775                                    0
```

```
##                              distinct                                 value
##                                     1 Years per 100,000 population aged <75
##
## Value      Years per 100,000 population aged <75
## Frequency                                   5775
## Proportion                                     1
## ---------------------------------------------------------------------------------
## metric_source_desc_label_fn
##                                                                                n
##                                                                             5775
##                                                                          missing
##                                                                                0
##                                                                         distinct
##                                                                                1
##                                                                            value
## National Vital Statistics System (NVSS), Centers for Disease Control and Prevention
##
## Value      National Vital Statistics System (NVSS), Centers for Disease Control and Prevention
## Frequency                                          5775
## Proportion                                         1
## ---------------------------------------------------------------------------------
## metric_source_desc_label_url_fn
##                                    n                               missing
##                                 5775                                     0
##                             distinct                                 value
##                                    1 https://www.cdc.gov/nchs/nvss/index.htm
##
## Value      https://www.cdc.gov/nchs/nvss/index.htm
## Frequency                                     5775
## Proportion                                       1
## ---------------------------------------------------------------------------------
## geo_label_city
##        n  missing distinct
##     5775        0       35
##
## lowest : Austin        Baltimore     Boston        Charlotte     Chicago
## highest: San Francisco San Jose      Seattle       Tucson        Washington
## ---------------------------------------------------------------------------------
## geo_label_state
##        n  missing distinct
##     5775        0       23
##
## lowest : AZ CA CO DC IL, highest: PA TN TX WA WI
## ---------------------------------------------------------------------------------
## geo_label_citystate
##        n  missing distinct
##     5775        0       35
##
## lowest : Austin, TX        Baltimore, MD     Boston, MA        Charlotte, NC     Chicago, IL
## highest: San Francisco, CA San Jose, CA      Seattle, WA       Tucson, AZ        Washington, DC
## ---------------------------------------------------------------------------------
## geo_fips_code
##        n  missing distinct
##     5775        0       35
```

```
##
## lowest : 0455000 0477000 0643000 0644000 0653000
## highest: 4827000 4835000 4865000 5363000 5553000
## ---------------------------------------------------------------------------
## value
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     5775        0     5775        1     6889     3991     2577     2970
##      .25      .50      .75      .90      .95
##     4148     5943     8813    12292    14370
##
## lowest :   874.2494  1014.6176  1191.5933  1243.0977  1250.5757
## highest: 22750.6721 22759.5573 23103.2065 23307.4240 23625.5989
## ---------------------------------------------------------------------------
## date_label
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     5775        0       11    0.992     2015    3.637     2010     2011
##      .25      .50      .75      .90      .95
##     2012     2015     2018     2019     2020
##
## lowest : 2010 2011 2012 2013 2014, highest: 2016 2017 2018 2019 2020
##
## Value       2010  2011  2012  2013  2014  2015  2016  2017  2018  2019  2020
## Frequency    525   525   525   525   525   525   525   525   525   525   525
## Proportion 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091
## ---------------------------------------------------------------------------
## geo_label_proxy_or_real
##        n  missing distinct
##     5775        0        2
##
## Value      proxy  real
## Frequency    330  5445
## Proportion 0.057 0.943
## ---------------------------------------------------------------------------
## geo_label_proxy_footnote
##                           n                     missing
##                         330                        5445
##                    distinct                       value
##                           1 County level data were used
##
## Value      County level data were used
## Frequency                          330
## Proportion                           1
## ---------------------------------------------------------------------------
## geo_fips_desc
##        n  missing distinct    value
##     5775        0        1    place
##
## Value      place
## Frequency   5775
## Proportion     1
## ---------------------------------------------------------------------------
## date_label_proxy_or_real
##        n  missing distinct    value
##     5775        0        1     real
```

```
##
## Value      real
## Frequency  5775
## Proportion    1
## ---------------------------------------------------------------------------
## value_ci_flag_yesno
##        n  missing distinct    value
##     5775        0        1      yes
##
## Value       yes
## Frequency  5775
## Proportion    1
## ---------------------------------------------------------------------------
## value_95_ci_low
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     5775        0     5775        1     6234     3940     1832     2370
##      .25      .50      .75      .90      .95
##     3501     5420     8149    11443    13498
##
## lowest :  -148.0370   197.1375   201.6486   246.1310   282.5401
## highest: 21640.0513 22086.2535 22174.3086 22513.4104 22650.6009
## ---------------------------------------------------------------------------
## value_95_ci_high
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     5775        0     5775        1     7544     4168     3065     3475
##      .25      .50      .75      .90      .95
##     4647     6522     9444    13022    15341
##
## lowest :  1551.361  1660.125  1744.886  1751.769  1806.661
## highest: 24101.438 24452.522 24579.977 24600.597 29193.761
## ---------------------------------------------------------------------------
## geo_strata_region
##        n  missing distinct
##     5775        0        4
##
## Value        Midwest Northeast     South      West
## Frequency       1320       495      1980      1980
## Proportion     0.229     0.086     0.343     0.343
## ---------------------------------------------------------------------------
## geo_strata_poverty
##        n  missing distinct
##     5775        0        2
##
## Value      Less poor cities (<20% poor)   Poorest cities (20%+ poor)
## Frequency                          4620                         1155
## Proportion                          0.8                          0.2
## ---------------------------------------------------------------------------
## geo_strata_Population
##        n  missing distinct
##     5775        0        2
##
## Value      Largest (>1.3 million) Smaller (<1.3 million)
## Frequency                    1485                   4290
## Proportion                  0.257                  0.743
```

```
## --------------------------------------------------------------------------------
## geo_strata_PopDensity
##        n  missing distinct
##     5775        0        2
##
## Value      Highest pop. density (>10k per sq mi)
## Frequency                                    990
## Proportion                                 0.171
##
## Value        Lower pop. density (<10k per sq mi)
## Frequency                                   4785
## Proportion                                 0.829
## --------------------------------------------------------------------------------
## geo_strata_Segregation
##        n  missing distinct
##     5775        0        2
##
## Value      Highly Segregated (50%+)   Less Segregated (<50%)
## Frequency                      2145                     3630
## Proportion                    0.371                    0.629
## --------------------------------------------------------------------------------
## strata_race_label
##        n  missing distinct
##     5775        0        5
##
## lowest : All      Asian/PI Black    Hispanic White
## highest: All      Asian/PI Black    Hispanic White
##
## Value           All Asian/PI    Black Hispanic    White
## Frequency      1155     1155     1155     1155     1155
## Proportion      0.2      0.2      0.2      0.2      0.2
## --------------------------------------------------------------------------------
## strata_sex_label
##        n  missing distinct
##     5775        0        3
##
## Value        Both Female   Male
## Frequency    1925   1925   1925
## Proportion  0.333  0.333  0.333
## --------------------------------------------------------------------------------
## strata_race_sex_label
##        n  missing distinct
##     3080     2695        8
##
## lowest : Asian/PI Female Asian/PI Male   Black Female    Black Male      Hispanic Female
## highest: Black Male      Hispanic Female Hispanic Male   White Female    White Male
##
## Value      Asian/PI Female   Asian/PI Male    Black Female      Black Male
## Frequency              385             385             385             385
## Proportion           0.125           0.125           0.125           0.125
##
## Value      Hispanic Female   Hispanic Male    White Female      White Male
## Frequency              385             385             385             385
## Proportion           0.125           0.125           0.125           0.125
```

```
## ------------------------------------------------------------------------------
##
## Variables with all observations missing:
##
## [1] date_label_proxy_footnote value_90_ci_low
## [3] value_90_ci_high
```

8.2 Our questions

Our question is how does YPLL differ by racial group, by the city level variables, over time.

Lets begin with the overall distribution, and remove the overlap groups - Both (genders) and All (races). We also take the opportunity to drop variables we're not going to use, to make looking at the dataset using the *View()* command easier.

```
BigC <- BigCities %>%
  filter(strata_sex_label != 'Both') %>% #Remove undesired overlap groups
  filter(strata_race_label != 'All') %>%
  select(-c('metric_item_label':'metric_source_desc_label_url_fn'),
         -c('geo_label_proxy_or_real':'value_ci_flag_yesno'),
         -c('geo_fips_code', 'value_90_ci_low', 'value_90_ci_high'),
         ) # Remove unloved variables
```

= is used quite often as a comparison operator - it's `TRUE` if the two things either side of it are the same in some sense. *!=* is also a comparison operator - it's `TRUE` if the two things either side of it are not equal. It's used in the two *filter()* statements to exclude rows that we don't want.

```
BigC %>% group_by(strata_sex_label) %>%
  summarise(N=n())
```

```
## # A tibble: 2 x 2
##   strata_sex_label     N
##   <chr>            <int>
## 1 Female            1540
## 2 Male              1540
```

```
BigC %>% group_by(strata_race_label) %>%
  summarise(N=n())
```

```
## # A tibble: 4 x 2
##   strata_race_label     N
##   <chr>             <int>
## 1 Asian/PI            770
## 2 Black               770
## 3 Hispanic            770
## 4 White               770
```

```
BigC %>% group_by(strata_race_sex_label) %>%
  summarise(N=n())
```

```
## # A tibble: 8 x 2
##   strata_race_sex_label     N
##   <chr>                 <int>
## 1 Asian/PI Female         385
## 2 Asian/PI Male           385
## 3 Black Female            385
## 4 Black Male              385
## 5 Hispanic Female         385
```

```
## 6 Hispanic Male           385
## 7 White Female            385
## 8 White Male              385
```

```
describe(BigC)
```

```
## BigC
##
##  15  Variables      3080  Observations
## --------------------------------------------------------------------------------
## geo_label_city
##        n  missing distinct
##     3080        0       35
##
## lowest : Austin        Baltimore     Boston        Charlotte     Chicago
## highest: San Francisco San Jose      Seattle       Tucson        Washington
## --------------------------------------------------------------------------------
## geo_label_state
##        n  missing distinct
##     3080        0       23
##
## lowest : AZ CA CO DC IL, highest: PA TN TX WA WI
## --------------------------------------------------------------------------------
## geo_label_citystate
##        n  missing distinct
##     3080        0       35
##
## lowest : Austin, TX        Baltimore, MD     Boston, MA        Charlotte, NC     Chicago, IL
## highest: San Francisco, CA San Jose, CA      Seattle, WA       Tucson, AZ        Washington, DC
## --------------------------------------------------------------------------------
## value
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     3080        0     3080        1     6824     4356     2317     2741
##      .25      .50      .75      .90      .95
##     3636     5798     8866    12968    15726
##
## lowest :   874.2494  1014.6176  1191.5933  1243.0977  1250.5757
## highest: 22750.6721 22759.5573 23103.2065 23307.4240 23625.5989
## --------------------------------------------------------------------------------
## date_label
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     3080        0       11    0.992     2015    3.638     2010     2011
##      .25      .50      .75      .90      .95
##     2012     2015     2018     2019     2020
##
## lowest : 2010 2011 2012 2013 2014, highest: 2016 2017 2018 2019 2020
##
## Value       2010  2011  2012  2013  2014  2015  2016  2017  2018  2019  2020
## Frequency    280   280   280   280   280   280   280   280   280   280   280
## Proportion 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091 0.091
## --------------------------------------------------------------------------------
## value_95_ci_low
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     3080        0     3080        1     5987     4243     1575     1972
##      .25      .50      .75      .90      .95
```

```
##     2960     5136     8022    11629    14747
##
## lowest :  -148.0370   197.1375   201.6486   246.1310   282.5401
## highest: 21640.0513 22086.2535 22174.3086 22513.4104 22650.6009
## ---------------------------------------------------------------------------
## value_95_ci_high
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     3080        0     3080        1     7661     4624     2897     3244
##      .25      .50      .75      .90      .95
##     4301     6491     9665    14253    17039
##
## lowest :  1551.361  1660.125  1744.886  1751.769  1806.661
## highest: 24101.438 24452.522 24579.977 24600.597 29193.761
## ---------------------------------------------------------------------------
## geo_strata_region
##        n  missing distinct
##     3080        0        4
##
## Value        Midwest Northeast     South      West
## Frequency        704       264      1056      1056
## Proportion     0.229     0.086     0.343     0.343
## ---------------------------------------------------------------------------
## geo_strata_poverty
##        n  missing distinct
##     3080        0        2
##
## Value      Less poor cities (<20% poor)   Poorest cities (20%+ poor)
## Frequency                          2464                          616
## Proportion                          0.8                          0.2
## ---------------------------------------------------------------------------
## geo_strata_Population
##        n  missing distinct
##     3080        0        2
##
## Value      Largest (>1.3 million) Smaller (<1.3 million)
## Frequency                     792                   2288
## Proportion                  0.257                  0.743
## ---------------------------------------------------------------------------
## geo_strata_PopDensity
##        n  missing distinct
##     3080        0        2
##
## Value      Highest pop. density (>10k per sq mi)
## Frequency                                    528
## Proportion                                 0.171
##
## Value        Lower pop. density (<10k per sq mi)
## Frequency                                   2552
## Proportion                                 0.829
## ---------------------------------------------------------------------------
## geo_strata_Segregation
##        n  missing distinct
##     3080        0        2
##
```

```
## Value      Highly Segregated (50%+)   Less Segregated (<50%)
## Frequency                      1144                     1936
## Proportion                    0.371                    0.629
## ---------------------------------------------------------------------------------
## strata_race_label
##        n  missing distinct
##     3080        0        4
##
## Value      Asian/PI    Black Hispanic    White
## Frequency       770      770      770      770
## Proportion     0.25     0.25     0.25     0.25
## ---------------------------------------------------------------------------------
## strata_sex_label
##        n  missing distinct
##     3080        0        2
##
## Value      Female   Male
## Frequency    1540   1540
## Proportion    0.5    0.5
## ---------------------------------------------------------------------------------
## strata_race_sex_label
##        n  missing distinct
##     3080        0        8
##
## lowest : Asian/PI Female Asian/PI Male   Black Female    Black Male      Hispanic Female
## highest: Black Male      Hispanic Female Hispanic Male   White Female    White Male
##
## Value      Asian/PI Female   Asian/PI Male    Black Female      Black Male
## Frequency              385             385             385             385
## Proportion           0.125           0.125           0.125           0.125
##
## Value      Hispanic Female   Hispanic Male    White Female      White Male
## Frequency              385             385             385             385
## Proportion           0.125           0.125           0.125           0.125
## ---------------------------------------------------------------------------------
```

So much for the data, now for some results.

8.3 Tables of means and standard deviations

We start with a simple table of means. There are tools for doing long table of multiple variables, specifically for journals, but it would take us too far afield to use these today - try the *rtables* or *arsenal* packages. There are lots more.

```
BigC %>% summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 1 x 1
##    Mean
##   <dbl>
## 1 6824.
```

```
BigC %>%  group_by(geo_strata_region) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 4 x 2
##   geo_strata_region  Mean
```

```
##   <chr>              <dbl>
## 1 Midwest            7766.
## 2 Northeast          5876.
## 3 South              6781.
## 4 West               6476.
```

```
BigC %>%  group_by(geo_strata_poverty) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 2
##   geo_strata_poverty            Mean
##   <chr>                        <dbl>
## 1 Less poor cities (<20% poor) 6374.
## 2 Poorest cities (20%+ poor)   8625.
```

```
BigC %>%  group_by(geo_strata_Population) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 2
##   geo_strata_Population   Mean
##   <chr>                  <dbl>
## 1 Largest (>1.3 million) 6554.
## 2 Smaller (<1.3 million) 6917.
```

```
BigC %>%  group_by(geo_strata_PopDensity) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 2
##   geo_strata_PopDensity                  Mean
##   <chr>                                 <dbl>
## 1 Highest pop. density (>10k per sq mi) 6021.
## 2 Lower pop. density (<10k per sq mi)   6990.
```

```
BigC %>%  group_by(strata_race_label) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 4 x 2
##   strata_race_label   Mean
##   <chr>              <dbl>
## 1 Asian/PI           4165.
## 2 Black             11583.
## 3 Hispanic           4992.
## 4 White              6555.
```

```
BigC %>%  group_by(strata_sex_label) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 2
##   strata_sex_label  Mean
##   <chr>            <dbl>
## 1 Female           4903.
## 2 Male             8745.
```

```
BigC %>%  group_by(strata_race_sex_label) %>%
  summarise(Mean = mean(value, na.rm=TRUE))
```

```
## # A tibble: 8 x 2
##   strata_race_sex_label   Mean
##   <chr>                  <dbl>
```

```
## 1 Asian/PI Female         2990.
## 2 Asian/PI Male           5341.
## 3 Black Female            8397.
## 4 Black Male             14770.
## 5 Hispanic Female         3344.
## 6 Hispanic Male           6640.
## 7 White Female            4879.
## 8 White Male              8231.
```

To get an idea of variability we can do a few things, but I'm going to do standard deviations.

```
BigC %>% summarise(Mean = mean(value, na.rm=TRUE),
                             SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 1 x 2
##    Mean    SD
##   <dbl> <dbl>
## 1 6824. 4085.
```

```
BigC %>%  group_by(geo_strata_region) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                             SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 4 x 3
##   geo_strata_region  Mean    SD
##   <chr>             <dbl> <dbl>
## 1 Midwest           7766. 4643.
## 2 Northeast         5876. 3462.
## 3 South             6781. 4067.
## 4 West              6476. 3724.
```

```
BigC %>%  group_by(geo_strata_poverty) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                             SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 3
##   geo_strata_poverty            Mean    SD
##   <chr>                        <dbl> <dbl>
## 1 Less poor cities (<20% poor) 6374. 3793.
## 2 Poorest cities (20%+ poor)   8625. 4676.
```

```
BigC %>%  group_by(geo_strata_Population) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                             SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 3
##   geo_strata_Population   Mean    SD
##   <chr>                  <dbl> <dbl>
## 1 Largest (>1.3 million) 6554. 3859.
## 2 Smaller (<1.3 million) 6917. 4157.
```

```
BigC %>%  group_by(geo_strata_PopDensity) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                             SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 3
##   geo_strata_PopDensity                  Mean    SD
##   <chr>                                 <dbl> <dbl>
```

```
## 1 Highest pop. density (>10k per sq mi) 6021. 4322.
## 2 Lower pop. density (<10k per sq mi)   6990. 4015.
```

```
BigC %>%  group_by(strata_race_label) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                              SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 4 x 3
##   strata_race_label   Mean    SD
##   <chr>              <dbl> <dbl>
## 1 Asian/PI           4165. 2060.
## 2 Black             11583. 4095.
## 3 Hispanic           4992. 2166.
## 4 White              6555. 2814.
```

```
BigC %>%  group_by(strata_sex_label) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                              SD = sd(value, na.rm=TRUE))
```

```
## # A tibble: 2 x 3
##   strata_sex_label  Mean    SD
##   <chr>            <dbl> <dbl>
## 1 Female           4903. 2541.
## 2 Male             8745. 4420.
```

```
BigC %>%  group_by(strata_race_sex_label) %>%
  summarise(Mean = mean(value, na.rm=TRUE),
                              SD = sd(value, na.rm=TRUE))
```

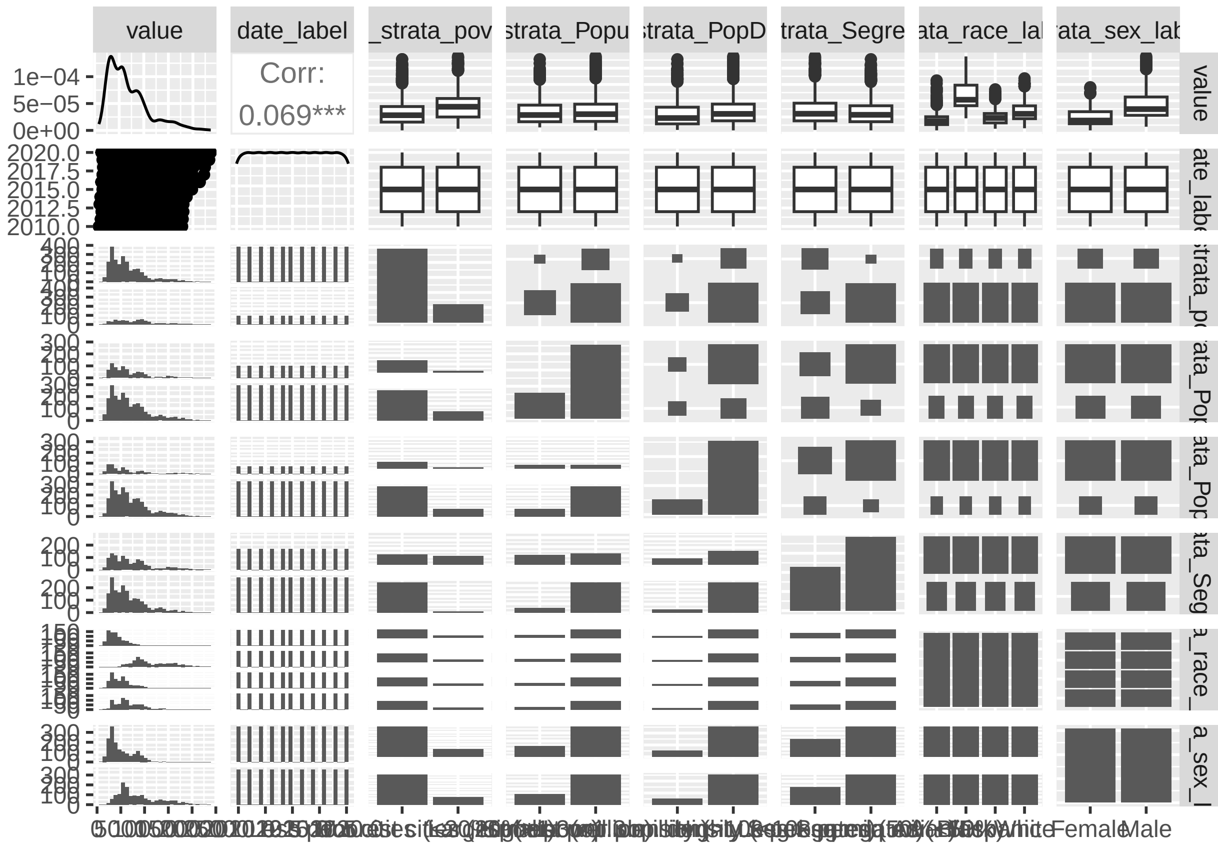
```
## # A tibble: 8 x 3
##   strata_race_sex_label   Mean    SD
##   <chr>                  <dbl> <dbl>
## 1 Asian/PI Female        2990. 1128.
## 2 Asian/PI Male          5341. 2112.
## 3 Black Female           8397. 1568.
## 4 Black Male            14770. 3281.
## 5 Hispanic Female        3344.  820.
## 6 Hispanic Male          6640. 1810.
## 7 White Female           4879. 1771.
## 8 White Male             8231. 2663.
```

This is one extra line, and a fair bit of cutting and pasting, If you don't know how to cut and paste, please learn how to use [Ctrl][C] (copy) (or [Ctrl][X] (cut)) and [Ctrl][V] (paste) respectively.

8.4 Pictures

It would be nice to visualise some of this! Start by installing the *GGally* library. Once that's done, the code in the next chunk will work.

```
library(GGally)
  ggpairs(BigC %>% # Reduce the dataset to more useful variables
            select(value,
                   date_label,
                 geo_strata_poverty:strata_sex_label)
          )
```



These *pair plots* are useful tools for looking across a range of variables, while studying a dataset. Like the output of *describe()* they repay careful study. They are seldom helpful if published.

Here are a series of graphs of YPLL against year. We plot the difference (DeltaValue) from the overall mean, in standard deviation units, and draw a red line at the 0 mark (equal to the overall mean) on each graph.

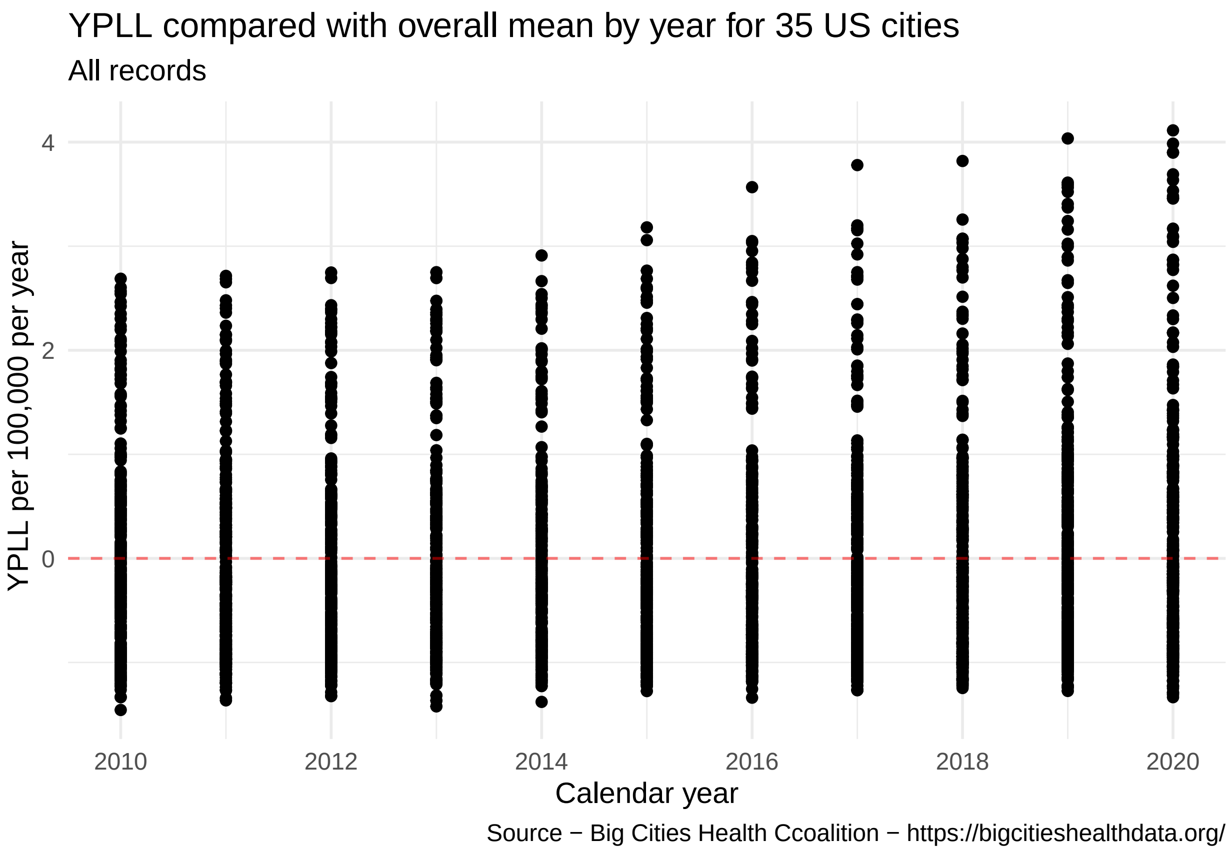
```
BigC <- BigC %>%
  mutate(DeltaValue =
           (value -
              mean(value, na.rm=TRUE))/
           sd(value, na.rm=TRUE))

fivenum(BigC$DeltaValue)
```

```
## [1] -1.4564624 -0.7803802 -0.2510804  0.4998413  4.1130051
```

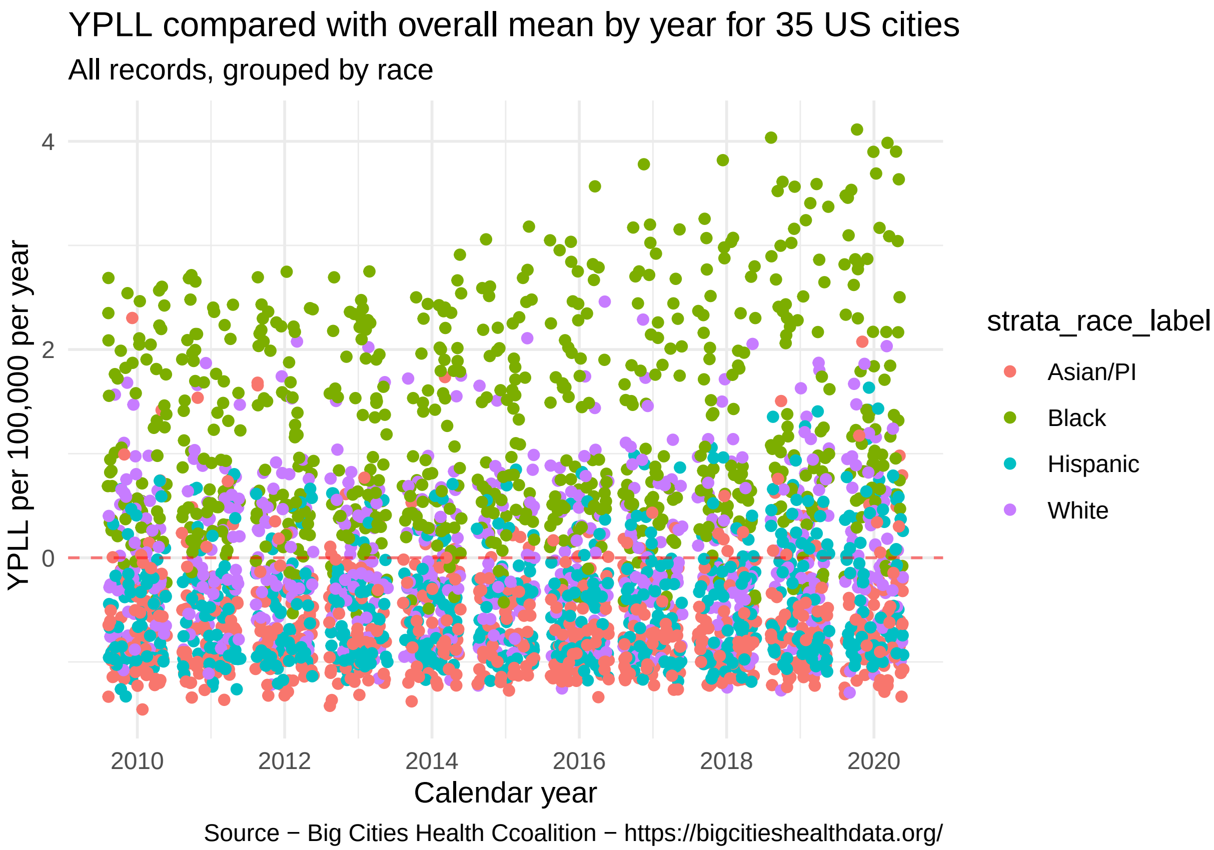
```
ggplot(BigC %>% mutate(DeltaValue),
       aes(x=date_label, y=DeltaValue)) +
  geom_point() +
  scale_x_continuous(breaks = breaks_extended(8)) + # Nice labels for years
  geom_hline(yintercept = 0,
             linetype = 'dashed',
             colour = 'red', alpha=0.5) +
  labs(
    title = 'YPLL compared with overall mean by year for 35 US cities',
    subtitle = 'All records',
    x = 'Calendar year',
    y = ' YPLL per 100,000 per year',
    caption = 'Source - Big Cities Health Ccoalition - https://bigcitieshealthdata.org/'
```

```
) +
theme_minimal() # Sets the overall style of the graph - there are lots and lots of these.
```



This graph looks odd because lots of points are overplotted and all are the same colour. This next version is more helpful.

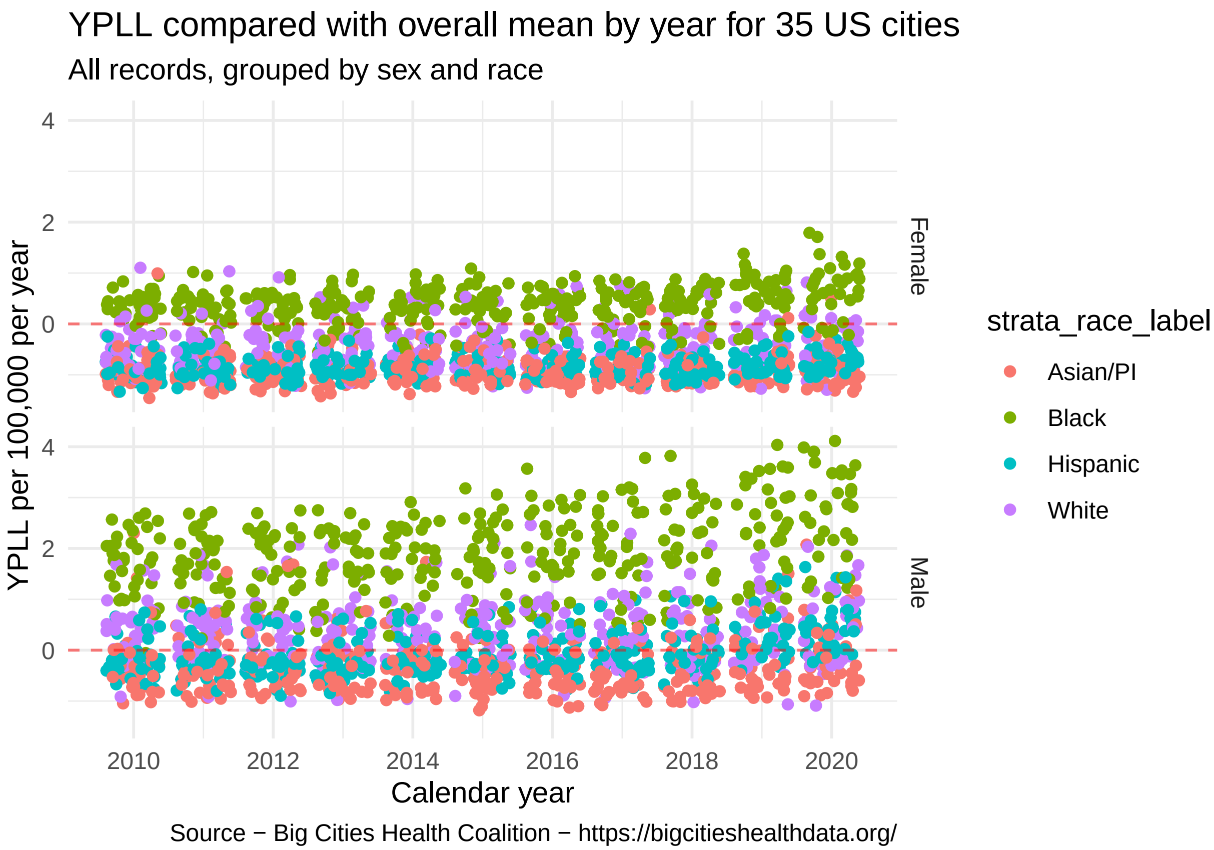
```
ggplot(BigC %>% mutate(DeltaValue),
       aes(x=date_label, y=DeltaValue,
           colour=strata_race_label)) +
  geom_point(position = 'jitter') +
  scale_x_continuous(breaks = breaks_extended(8)) + # Nice labels for years
  geom_hline(yintercept = 0,
             linetype = 'dashed',
             colour = 'red', alpha = 0.5) +
  labs(
    title = 'YPLL compared with overall mean by year for 35 US cities',
    subtitle = 'All records, grouped by race',
    x = 'Calendar year',
    y = ' YPLL per 100,000 per year',
    caption = 'Source - Big Cities Health Ccoalition - https://bigcitieshealthdata.org/'
  ) +
  theme_minimal() # Sets the overall style of the graph - there are lots and lots of these.
```



We've jittered the points - *geom_point(position = 'jitter')* or *geom_jitter()* (which are identical). and we've coloured them, with *aes(colour = strata_race_label))*, by racial group.

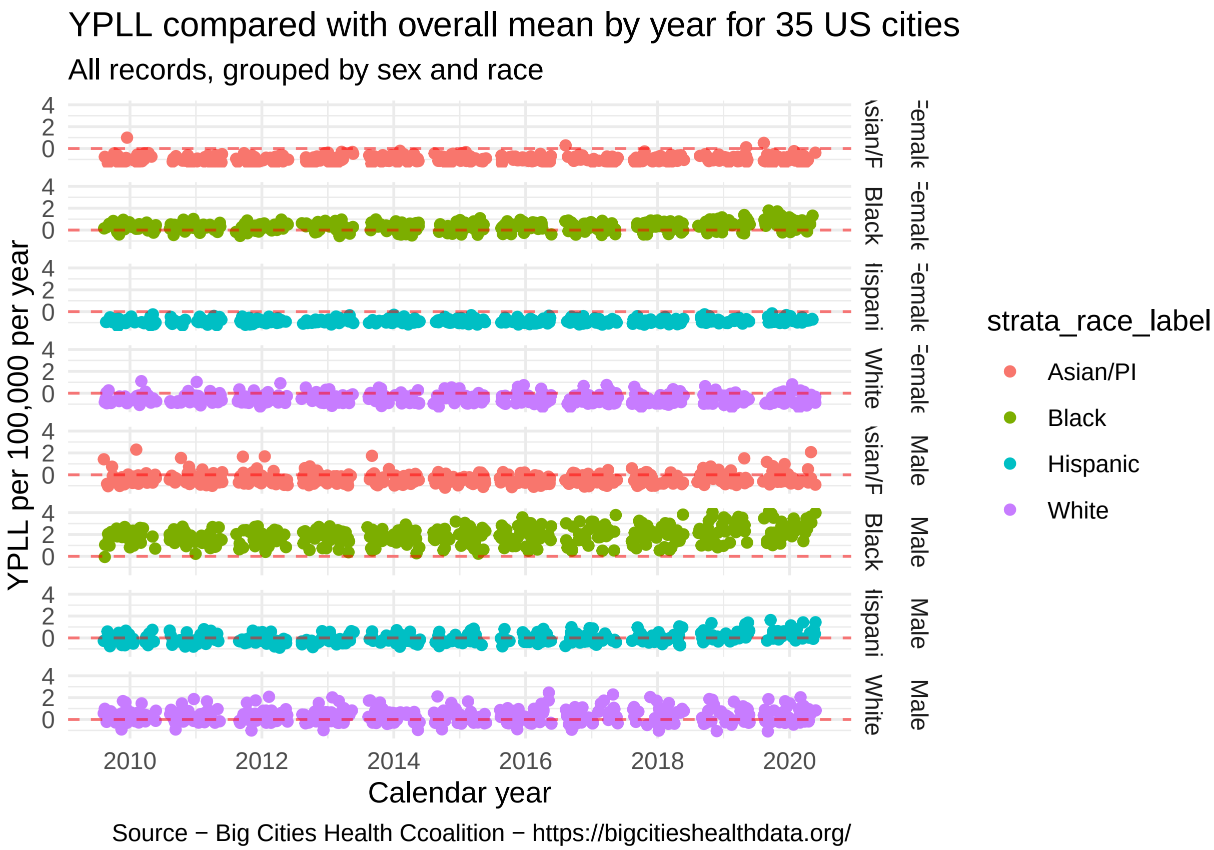
We can also do multiple graphs on the same sheet, using the *facet_wrap()* and *facet_grid()* functions, to make separate graphs for males and females.

```
ggplot(BigC %>% mutate(DeltaValue),
       aes(x=date_label, y=DeltaValue,
           colour=strata_race_label)) +
  geom_point(position = 'jitter') +
  scale_x_continuous(breaks = breaks_extended(8)) + # Nice labels for years
  geom_hline(yintercept = 0,
             linetype = 'dashed',
             colour = 'red', alpha = 0.5) +
  labs( # Lets us set most of teh text in the graph
    title = 'YPLL compared with overall mean by year for 35 US cities',
    subtitle = 'All records, grouped by sex and race',
    x = 'Calendar year',
    y = ' YPLL per 100,000 per year',
    caption = 'Source - Big Cities Health Coalition - https://bigcitieshealthdata.org/'
  ) +
  theme_minimal() + # Sets the overall style of the graph - there are lots and lots of themes.
  facet_grid(strata_sex_label ~ ., scales='fixed')
```



This already tells us a lot. Look carefully at the graph and try to interpret it. The next two graphs might work well at a conference.

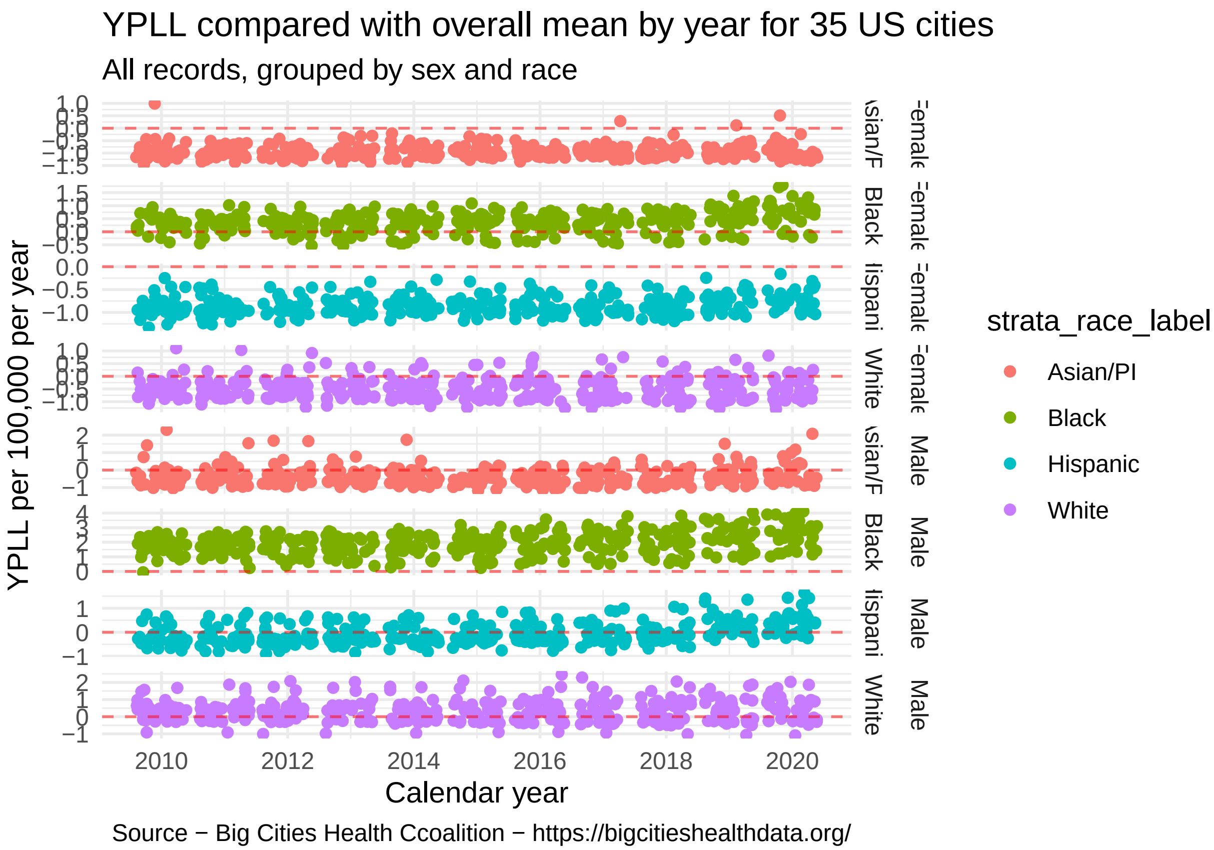
```
ggplot(BigC %>% mutate(DeltaValue),
       aes(x=date_label, y=DeltaValue,
           colour=strata_race_label)) +
  geom_point(position = 'jitter') +
  scale_x_continuous(breaks = breaks_extended(8)) + # Nice labels for years
  geom_hline(yintercept = 0,
             linetype = 'dashed',
             colour = 'red', alpha = 0.5) +
  labs( # Lets us set most of the text in the graph
    title = 'YPLL compared with overall mean by year for 35 US cities',
    subtitle = 'All records, grouped by sex and race',
    x = 'Calendar year',
    y = ' YPLL per 100,000 per year',
    caption = 'Source - Big Cities Health Ccoalition - https://bigcitieshealthdata.org/'
  ) +
  theme_minimal() + # Sets the overall style of the graph - there are lots and lots of these themes to 
  facet_grid(vars(strata_sex_label, strata_race_label),
             scales='fixed') # Same scale every graph
```



There are over 3,000 points on this graph, which is pretty good.

With this type of graph, and a statistically sophisticated audience, it can also be helpful to give each strip it's own scale.

```
ggplot(BigC %>% mutate(DeltaValue),
       aes(x=date_label, y=DeltaValue,
           colour=strata_race_label)) +
  geom_point(position = 'jitter') +
  scale_x_continuous(breaks = breaks_extended(8)) + # Nice labels for years
  geom_hline(yintercept = 0,
             linetype = 'dashed',
             colour = 'red', alpha = 0.5) +
  labs( # Lets us set most of the text in the graph
    title = 'YPLL compared with overall mean by year for 35 US cities',
    subtitle = 'All records, grouped by sex and race',
    x = 'Calendar year',
    y = ' YPLL per 100,000 per year',
    caption = 'Source - Big Cities Health Ccoalition - https://bigcitieshealthdata.org/'
  ) +
  theme_minimal() + # Sets the overall style of the graph - there are lots and lots of these themes to
  facet_grid(vars(strata_sex_label, strata_race_label),
             scales='free')
```



Please do this carefully. It's risky.

8.5 Statistics

A fairly obvious question is whether the apparent difference in means we saw earlier are real. The tool to answer this is linear regression. t-tests (which you may have heard of) are thinly disguised linear regression.

To do this we need to install the *broom* package - please do so. when this is done, the next chunk will work.

```
require(broom)

BigC %>%
    do(tidy( # Run the regressions
      lm(DeltaValue ~ strata_race_label, .),
      conf.int = TRUE))
```

```
## # A tibble: 4 x 7
##   term                        estimate std.error stati~1   p.value conf.~2 conf.~3
##   <chr>                          <dbl>     <dbl>   <dbl>     <dbl>   <dbl>   <dbl>
## 1 (Intercept)                   -0.651    0.0256  -25.4  8.86e-130  -0.701  -0.601
## 2 strata_race_labelBlack         1.82     0.0362   50.2  0            1.74    1.89
## 3 strata_race_labelHispanic      0.202    0.0362    5.60 2.39e-  8    0.131   0.273
## 4 strata_race_labelWhite         0.585    0.0362   16.2  1.60e- 56    0.514   0.656
## # ... with abbreviated variable names 1: statistic, 2: conf.low, 3: conf.high
```

```
BigC %>%
    do(glance( # One line summary of results
      lm(DeltaValue ~ strata_race_label, .),
      conf.int = TRUE))
```

```
## # A tibble: 1 x 12
##   r.squ~1 adj.r~2 sigma stati~3 p.value    df logLik   AIC   BIC devia~4 df.re~5
##     <dbl>   <dbl> <dbl>   <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>   <int>
## 1   0.497   0.496 0.710   1012.       0     3 -3312. 6635. 6665.   1549.    3076
## # ... with 1 more variable: nobs <int>, and abbreviated variable names
## #   1: r.squared, 2: adj.r.squared, 3: statistic, 4: deviance, 5: df.residual
```

```
#or#
 Regression1 <- lm(DeltaValue ~ strata_race_label, BigC)
  summary(Regression1)
```

```
##
## Call:
## lm(formula = DeltaValue ~ strata_race_label, data = BigC)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7116 -0.4922 -0.1389  0.3688  2.9539
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -0.65081    0.02558 -25.445  < 2e-16 ***
## strata_race_labelBlack      1.81589    0.03617  50.202  < 2e-16 ***
## strata_race_labelHispanic   0.20239    0.03617   5.595 2.39e-08 ***
## strata_race_labelWhite      0.58497    0.03617  16.172  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7097 on 3076 degrees of freedom
## Multiple R-squared:  0.4968, Adjusted R-squared:  0.4963
## F-statistic:  1012 on 3 and 3076 DF,  p-value: < 2.2e-16
```

```
   confint(Regression1)
```

```
##                                 2.5 %     97.5 %
## (Intercept)                -0.7009644 -0.6006634
## strata_race_labelBlack      1.7449654  1.8868124
## strata_race_labelHispanic   0.1314709  0.2733179
## strata_race_labelWhite      0.5140487  0.6558957
```

I recommend the first version!

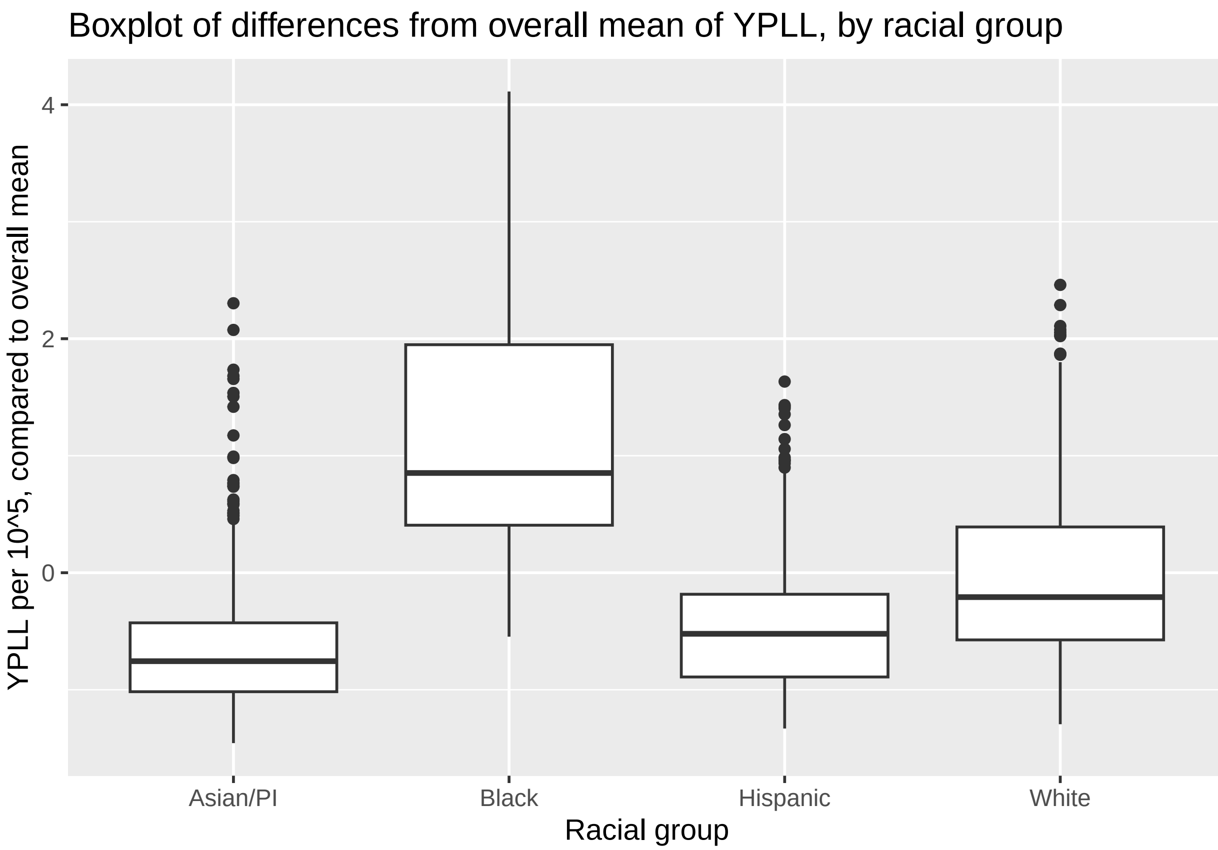
What does this say?

```
BigC %>% # Start with our dataframe (Table)
    do( # Self explanatory - run the code here across the data
     tidy( # Make the output easy to use
      lm(value ~ strata_race_label, .), # Linear regression
      conf.int = TRUE) # with Confidence Intervals please
      )
```

So what is linear regression doing? Look at this graph (called a boxplot).

```
ggplot(BigC,
       aes(x = strata_race_label, y = DeltaValue)) +
         geom_boxplot() +
  labs(
    title = 'Boxplot of differences from overall mean of YPLL, by racial group',
```

```
    x = "Racial group",
    y = "YPLL per 10^5, compared to overall mean"
  )
```



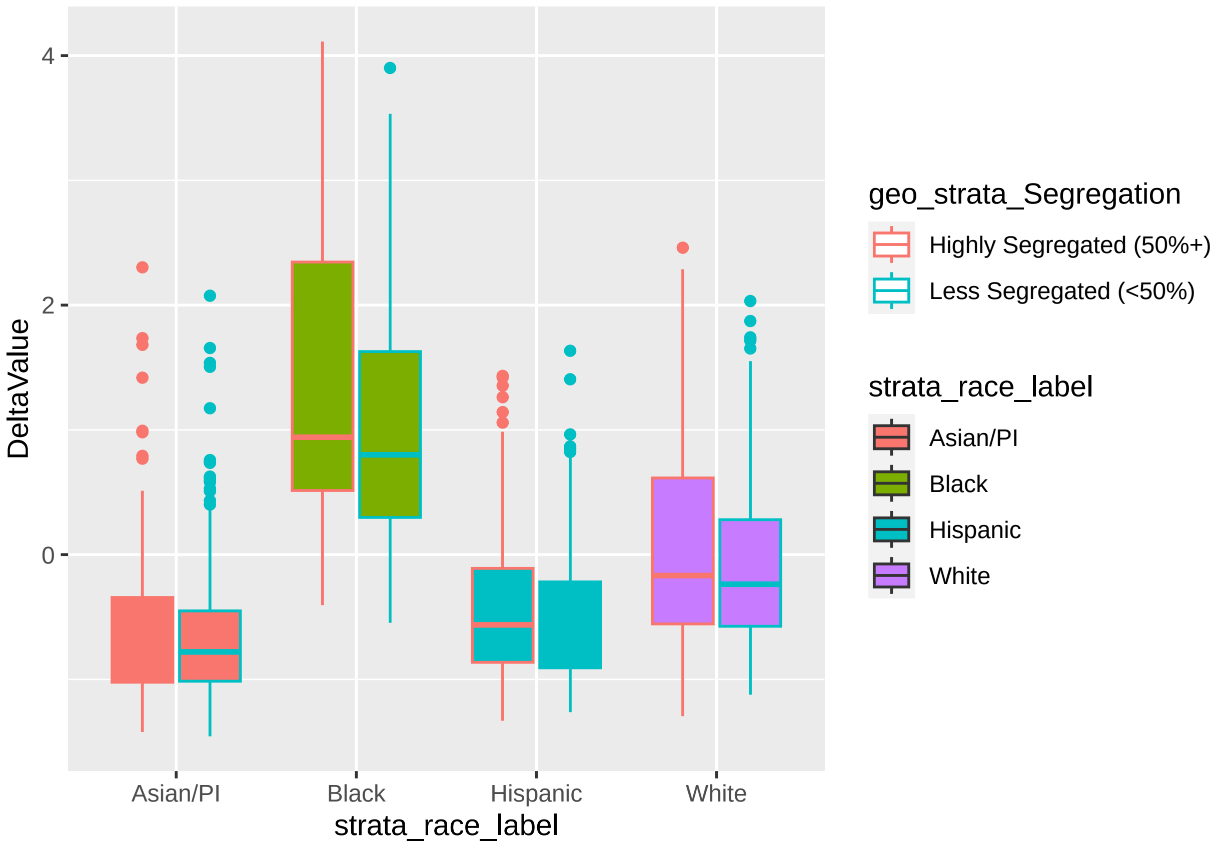
Linear regression asks the question 'Is there evidence that the apparent differences between these four groups, Blacks, Hispanics and Whites, compared with Asian/PI are real, or are they likely to be due to chance?'

The answer is they are probably real. More complex tools (notably a thing called 'Tukey's Honest Significant Difference') exist to answer very specific questions like - are Hispanics and Asians/PIs different? We're not going to cover these today either.

Let's look at a slightly different question. Are more segregated cities likely to have worse health, within each racial group than less segregated cities?

This is what it looks like :-

```
#require(ggmosaic)
ggplot(BigC,
       aes(x =  strata_race_label,
         y = DeltaValue,
         fill = strata_race_label,
         colour = geo_strata_Segregation)) +
  geom_boxplot()
```



In terms of regression we are fitting a regression that looks at the effects of race and segregation separately (m1 and m2 below), and then together (m3 below).

```
m1 <- (lm(DeltaValue ~ strata_race_label, data=BigC))
m2 <- (lm(DeltaValue ~ geo_strata_Segregation, data=BigC))
m3 <- (lm(DeltaValue ~ strata_race_label*geo_strata_Segregation, data=BigC))

tidy(m1)
```

```
## # A tibble: 4 x 5
##   term                       estimate std.error statistic   p.value
##   <chr>                         <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)                  -0.651    0.0256    -25.4  8.86e-130
## 2 strata_race_labelBlack        1.82     0.0362     50.2  0
## 3 strata_race_labelHispanic     0.202    0.0362      5.60 2.39e-  8
## 4 strata_race_labelWhite        0.585    0.0362     16.2  1.60e- 56
```

```
tidy(m2)
```

```
## # A tibble: 2 x 5
##   term                                        estimate std.er~1 stati~2 p.value
##   <chr>                                          <dbl>    <dbl>   <dbl>   <dbl>
## 1 (Intercept)                                    0.118   0.0294    4.02 5.92e-5
## 2 geo_strata_SegregationLess Segregated (<50%)   -0.188   0.0371   -5.07 4.16e-7
## # ... with abbreviated variable names 1: std.error, 2: statistic
```

```
tidy(m3)
```

```
## # A tibble: 8 x 5
```

```
##   term                                          estim~1 std.e~2 stati~3   p.value
##   <chr>                                           <dbl>   <dbl>   <dbl>     <dbl>
## 1 (Intercept)                                    -0.613   0.0415 -14.8   8.54e- 48
## 2 strata_race_labelBlack                          2.04    0.0586  34.9   1.48e-224
## 3 strata_race_labelHispanic                       0.223   0.0586   3.81  1.44e-  4
## 4 strata_race_labelWhite                          0.658   0.0586  11.2   1.15e- 28
## 5 geo_strata_SegregationLess Segregated (<50%) -0.0608  0.0523  -1.16  2.45e-  1
## 6 strata_race_labelBlack:geo_strata_Segregati~ -0.361   0.0739  -4.89  1.07e-  6
## 7 strata_race_labelHispanic:geo_strata_Segreg~ -0.0330  0.0739  -0.446 6.56e-  1
## 8 strata_race_labelWhite:geo_strata_Segregati~ -0.116   0.0739  -1.57  1.17e-  1
## # ... with abbreviated variable names 1: estimate, 2: std.error, 3: statistic
```

Note how you can assign all sorts of stuff, including models to a variable in R. This means you can do clever stuff with them later on, without calculating them every time.

To make nice tables install the *gtsummary* package now please.

```
require(gtsummary)

t1 <- tbl_regression(m1)
t1
```

Characteristic	Beta	95% CI	p-value
strata_race_label			
Asian/PI	—	—	
Black	1.8	1.7, 1.9	<0.001
Hispanic	0.20	0.13, 0.27	<0.001
White	0.58	0.51, 0.66	<0.001

```
t2 <- tbl_regression(m2)
t2
```

Characteristic	Beta	95% CI	p-value
geo_strata_Segregation			
Highly Segregated (50%+)	—	—	
Less Segregated (<50%)	-0.19	-0.26, -0.12	<0.001

```
t3 <- tbl_regression(m3)
t3
```

Characteristic	Beta	95% CI	p-value
strata_race_label			
Asian/PI	—	—	
Black	2.0	1.9, 2.2	<0.001
Hispanic	0.22	0.11, 0.34	<0.001
White	0.66	0.54, 0.77	<0.001
geo_strata_Segregation			
Highly Segregated (50%+)	—	—	
Less Segregated (<50%)	-0.06	-0.16, 0.04	0.2
strata_race_label * geo_strata_Segregation			
Black * Less Segregated (<50%)	-0.36	-0.51, -0.22	<0.001
Hispanic * Less Segregated (<50%)	-0.03	-0.18, 0.11	0.7

Characteristic	Beta	95% CI	p-value
White * Less Segregated (<50%)	-0.12	-0.26, 0.03	0.12

```r
Model.Table <-
  tbl_merge(
    tbls = list(t1,t2,t3),
    tab_spanner = c('Model 1', 'Model 2', 'Model 3')
  )
Model.Table
```

Characteristic	Beta	95% CI	p-value	Beta	95% CI	p-value	Beta	95% CI	p-value
strata_race_label									
Asian/PI	—	—					—	—	
Black	1.8	1.7, 1.9	<0.001				2.0	1.9, 2.2	<0.001
Hispanic	0.20	0.13, 0.27	<0.001				0.22	0.11, 0.34	<0.001
White	0.58	0.51, 0.66	<0.001				0.66	0.54, 0.77	<0.001
geo_strata_Segregation									
Highly Segregated (50%+)				—	—		—	—	
Less Segregated (<50%)				-0.19	-0.26, -0.12	<0.001	-0.06	-0.16, 0.04	0.2
strata_race_label * geo_strata_Segregation									
Black * Less Segregated (<50%)							-0.36	-0.51, -0.22	<0.001
Hispanic * Less Segregated (<50%)							-0.03	-0.18, 0.11	0.7
White * Less Segregated (<50%)							-0.12	-0.26, 0.03	0.12

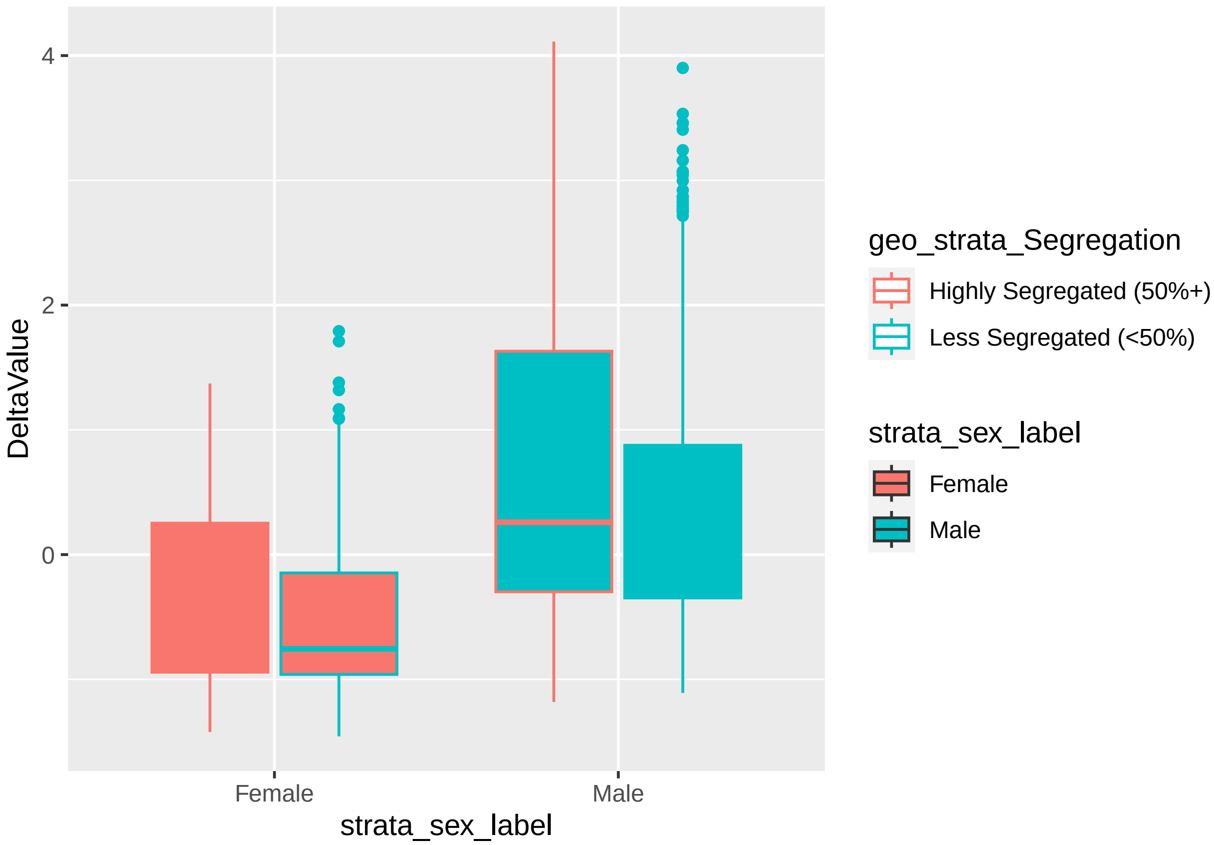
In our case, the key question is the __strata_race_label*geo_strata_Segregation__. This is a shorter version of *strata_race_label + geo_strata_Segregation + strata_race_label : geo_strata_Segregation*.

Unpicking this, we have a model that looks at the effect of race, and segregation, and the additional effect of high and low segregation for each race. The formal terms for these are the *main effect* of race and segregation, and their *interaction*. Note the very large effect of black race, and the impact (lower YPLL is good, remember) of being Black in a less segregated city.

A general warning :- it seldom makes sense to fit interaction terms without main effects!

Another topic of interest is gender. This is what it looks like :-

```r
#require(ggmosaic)
ggplot(BigC,
       aes(x =  strata_sex_label,
         y = DeltaValue,
         fill = strata_sex_label,
         colour = geo_strata_Segregation)) +
  geom_boxplot()
```



A more thorough analysis of these data might look a bit like this.

```
m4 <- lm(DeltaValue ~
                 strata_sex_label * strata_race_label +
           geo_strata_poverty + geo_strata_region +
           geo_strata_PopDensity + geo_strata_Population +
           geo_strata_Segregation,
         data = BigC )
tbl_regression(m4)
```

Characteristic	**Beta**	**95% CI**	**p-value**
strata_sex_label			
Female	—	—	
Male	0.58	0.51, 0.64	<0.001
strata_race_label			
Asian/PI	—	—	
Black	1.3	1.3, 1.4	<0.001
Hispanic	0.09	0.03, 0.15	0.005
White	0.46	0.40, 0.52	<0.001
geo_strata_poverty			
Less poor cities (<20% poor)	—	—	
Poorest cities (20%+ poor)	0.57	0.52, 0.62	<0.001
geo_strata_region			
Midwest	—	—	
Northeast	-0.40	-0.48, -0.32	<0.001
South	-0.14	-0.18, -0.10	<0.001
West	-0.19	-0.23, -0.14	<0.001

Characteristic	Beta	95% CI	p-value
geo_strata_PopDensity			
Highest pop. density (>10k per sq mi)	—	—	
Lower pop. density (<10k per sq mi)	0.09	0.03, 0.15	0.004
geo_strata_Population			
Largest (>1.3 million)	—	—	
Smaller (<1.3 million)	-0.09	-0.14, -0.05	<0.001
geo_strata_Segregation			
Highly Segregated (50%+)	—	—	
Less Segregated (<50%)	0.06	0.02, 0.11	0.007
strata_sex_label * strata_race_label			
Male * Black	0.98	0.90, 1.1	<0.001
Male * Hispanic	0.23	0.15, 0.32	<0.001
Male * White	0.24	0.16, 0.33	<0.001

Can you interpret this? Which variables matter most?

9 Next steps

R is a language for data analysis and graphics. Like any other language, and indeed like any other skill, you learn it by using it. You get better as you go along. Start by using R in your next project.

We have shown you the first steps on the way using two key ideas - tidy data, and a powerful tool for making graphs based on a formal grammar of graphics. To move further try these sites :-

- tidyverse for beginners - https://rladiessydney.org/post/little-miss-tidyverse/
- knitr - making documents with R - https://rmarkdown.rstudio.com/ or https://sachsmc.github.io/knit-git-markr-guide/knitr/knit.html
- R homepage - https://www.r-project.org/
- RStudio homepage - https://posit.co/
- tidyverse homepage - https://www.tidyverse.org/
- ggplot2 homepage - https://ggplot2.tidyverse.org/
- R for data science - https://r4ds.hadley.nz/
- Stack overflow - https://stackoverflow.com/ (A very useful question and answer site for R (among many other topics))
- R packages [**library()**] live on CRAN - https://cran.r-project.org/
- Sets of linked packages are given as *Task Views* - https://cran.r-project.org/web/views/

Data sources are :-

- OECD data store - https://data.oecd.org/
- Big Cities Health Inventory Data Platform. Big Cities Health Coalition. Bigcitieshealthdata.org accessed [14/11/2022].

Two small notes * In this file, for educational reasons, we've loaded the packages (with the *library()* command) as we needed them. In real work it's much better to load all the libraries in the first chunk, and to add any extra ones you need at the bottom of the list in the first chunk. Ask me to show you some of my own code.

- The very start of the file is not R, nor it it text. It's code to control document production, and it's written in a format called YAML, if you ever need to learn more about it. You can take what I've written and adjust it to suit your needs.