Imperial College
Business School

Imperial means
Intelligent Business

Rrrrr – Software for Pirates

by Ralf Martin (r.martin@imperial.ac.uk)

Why R?

Pros

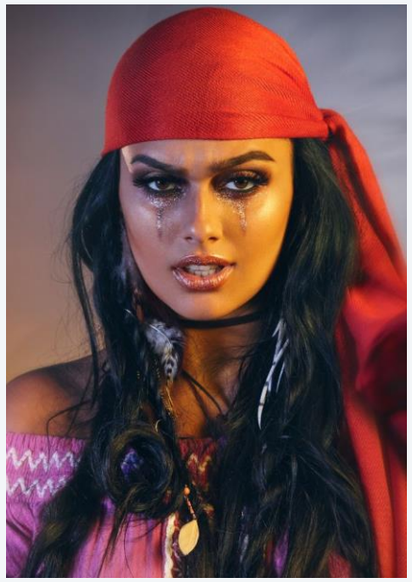
- The Pirates' choice of software
- It's free (like pirates)
- Open source, many contributors
- Many contributed modules and extensions
- Many different ways to do the same thing
- Easy integration with other software
- A new industry standard used across many fields
- Flexible
- You can program stuff

Cons:

- Open source, many contributors
- Many different ways to do the same thing

R vs RStudio

- Rstudio is a nice control software to run your R engine

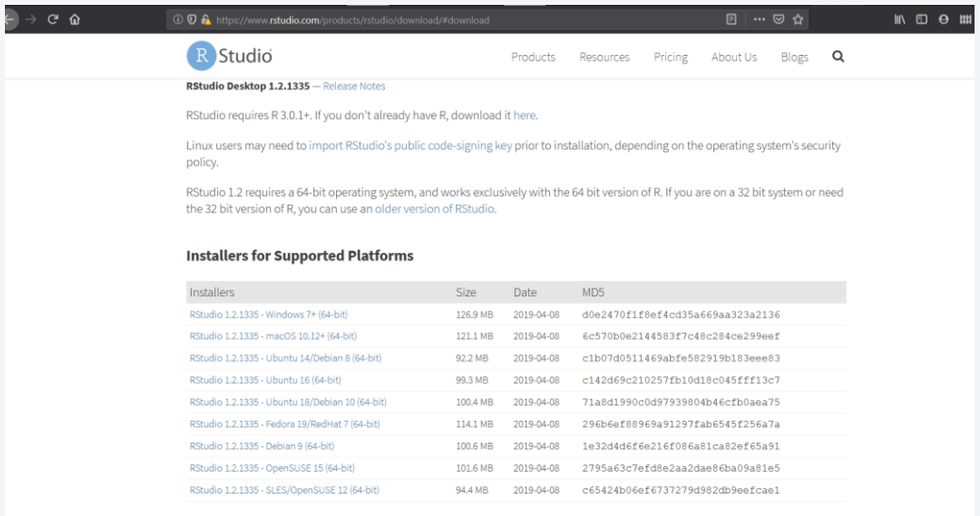


A Pirate (You)

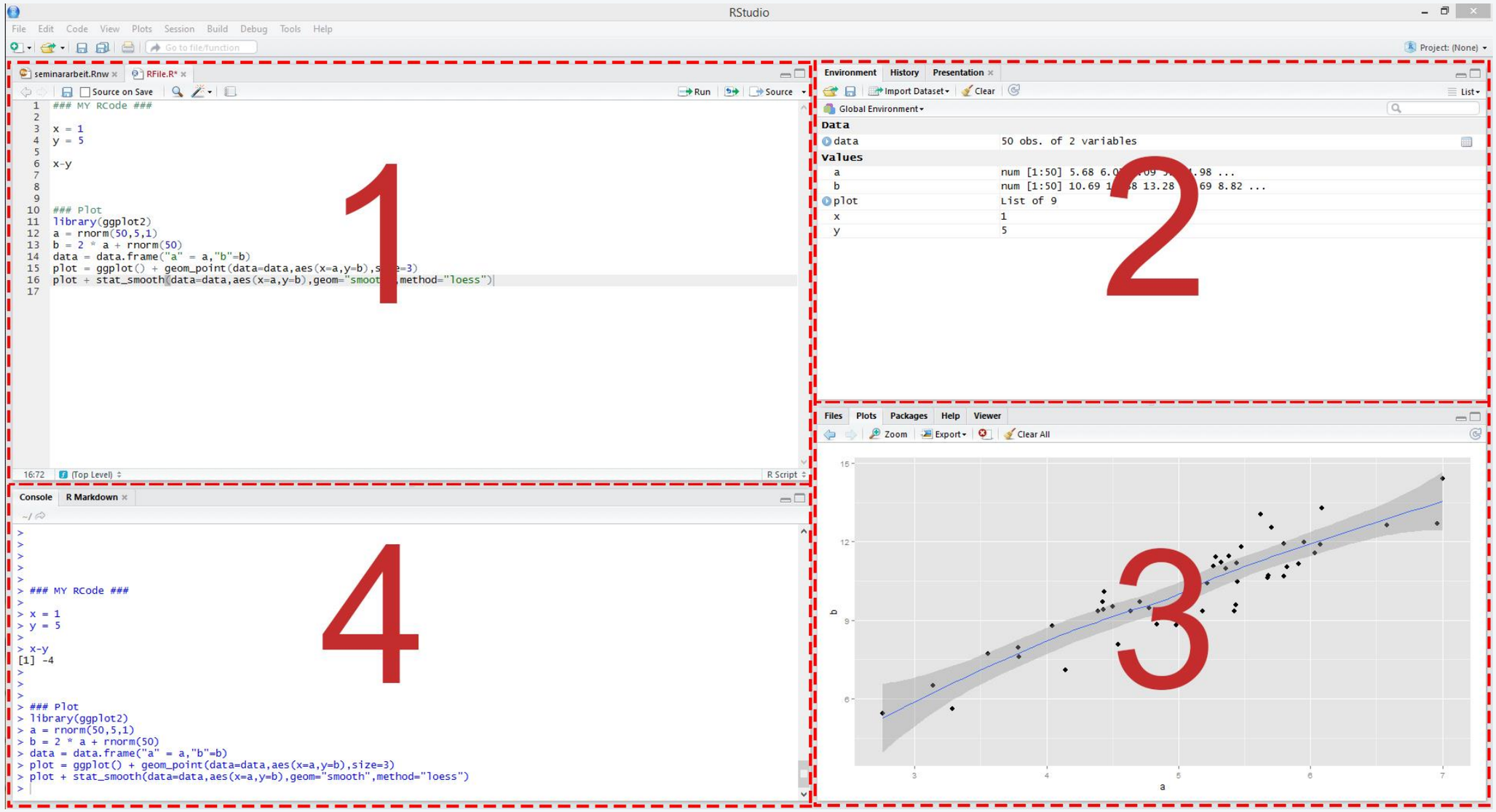
Getting started

- Download R (and install)
  https://www.r-project.org/

- Download Rstudio
  https://www.rstudio.com/



The R studio setup



1= code file

2= variable browser

3= plot browser

4= command console

How to talk to R?

- Write commands in the console to be executed immediately
- Write commands in a script file to executed later or repeatedly

You can use R as a pocket calculator

```
7/6
```

```
## [1] 1.166667
```

Create lists

```
1:10
runif(10)
seq(0,20,2)
sample(1:6,5)
```

The secret of learning to code:

- pLaY
- steal some toys
- and play some more

by that I mean code

Good places to steal code (just google):

- stackexchange
- Stackoverflow
- github

(you are pirates after all)

- Try to understand code of others
- Make small changes
- See what happens
- Adapt code by others for your purposes
- Read about the commands we are using as well as related commands

You can Assign Variables

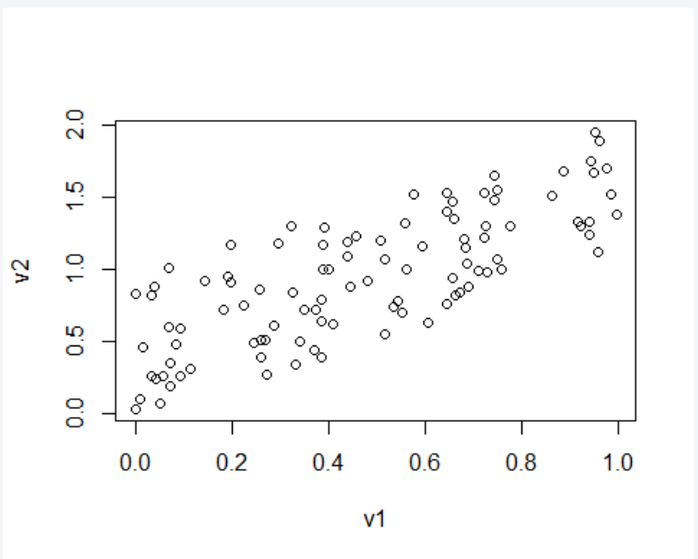
```
v1=runif(100)
```

Create new variables based on already existing one

```
v2=runif(100)+v1
```

Do stuff with variables; e.g. plotting them

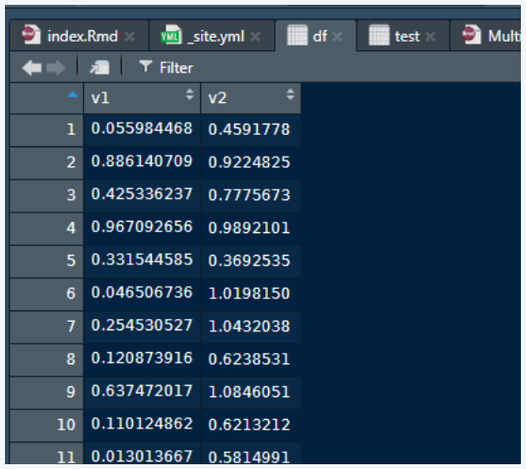
```
plot(v1,v2)
```



Dataframe

- To organise data we can put vectors of data into a dataframe; i.e. table

```
df=data.frame(v1,v2)
```



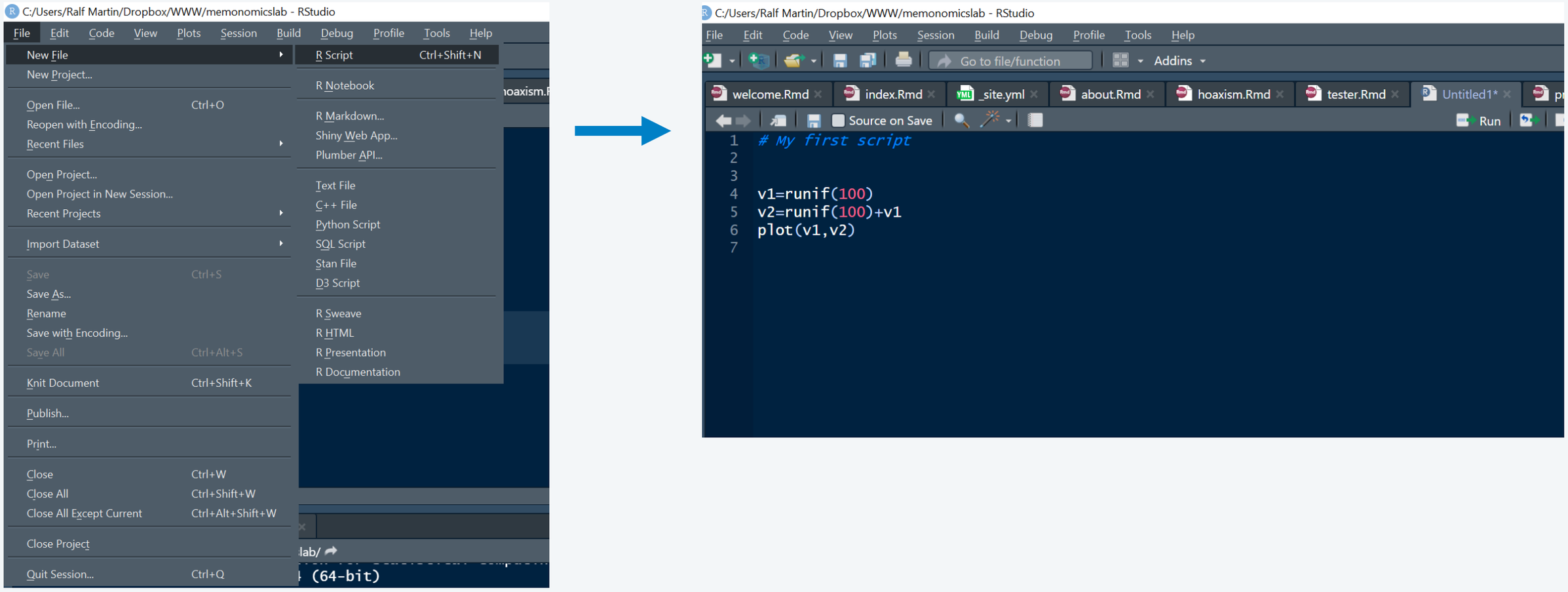
- You can look at it like in an excel table:

- Most of the time a dataset you get from somewhere will be arranged in a dataframe; e.g. the data on foreigners and crime you can load via

```
df=read.csv("https://www.dropbox.com/s/g1w75gkw7g91zef/foreigners.csv?dl=1")
```

To organise research we can combine commands in script file

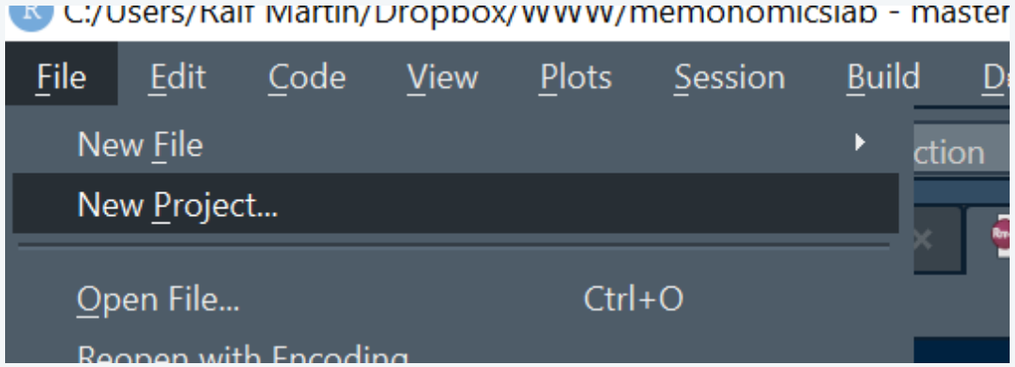
- Documenting what was done (to yourself and others)
- Identifying and correcting errors
- Efficiently executing repeated tasks
- Replication & reproduction of research

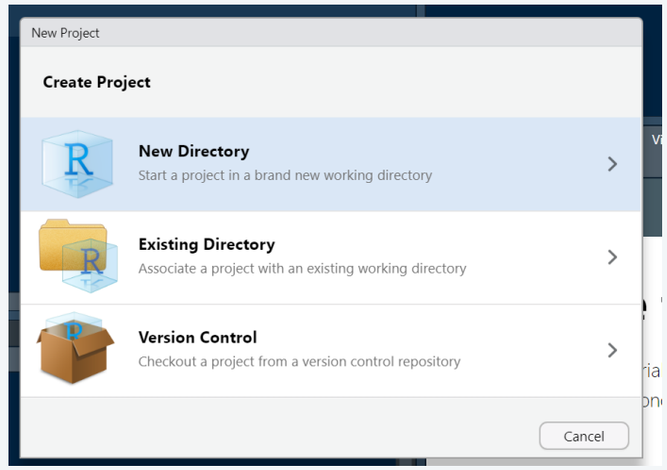


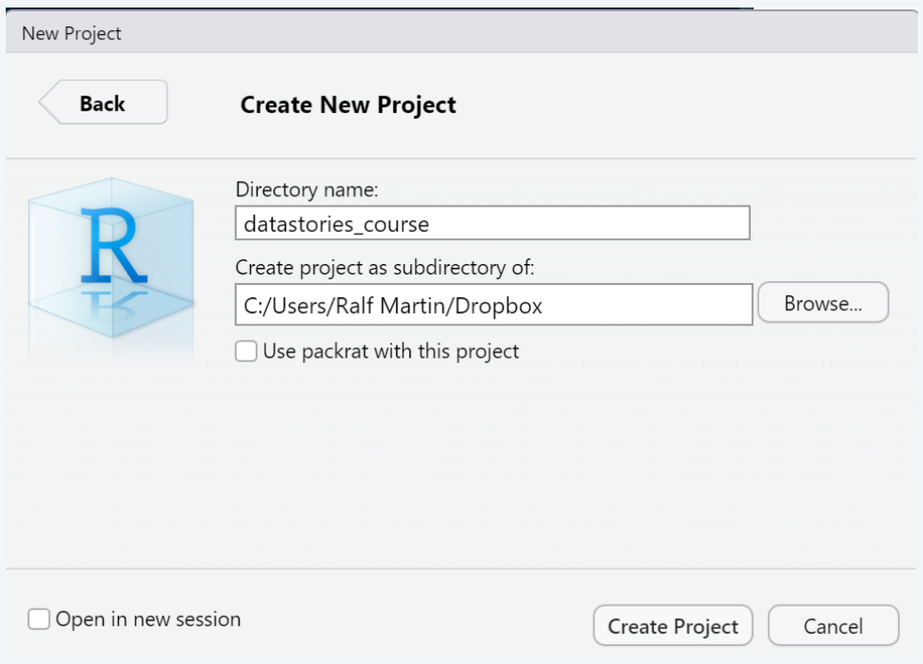
Projects and Folders

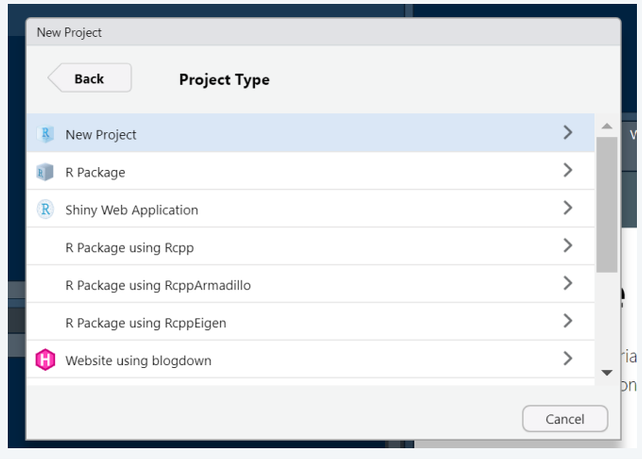
- An additional tool to organise a research project are project files and dedicated folders
- You can do both via the “New Project” menu:

You might want to create one for the course and one for dedicated one for the group project which you can share with your team/group (via dropbox or github)



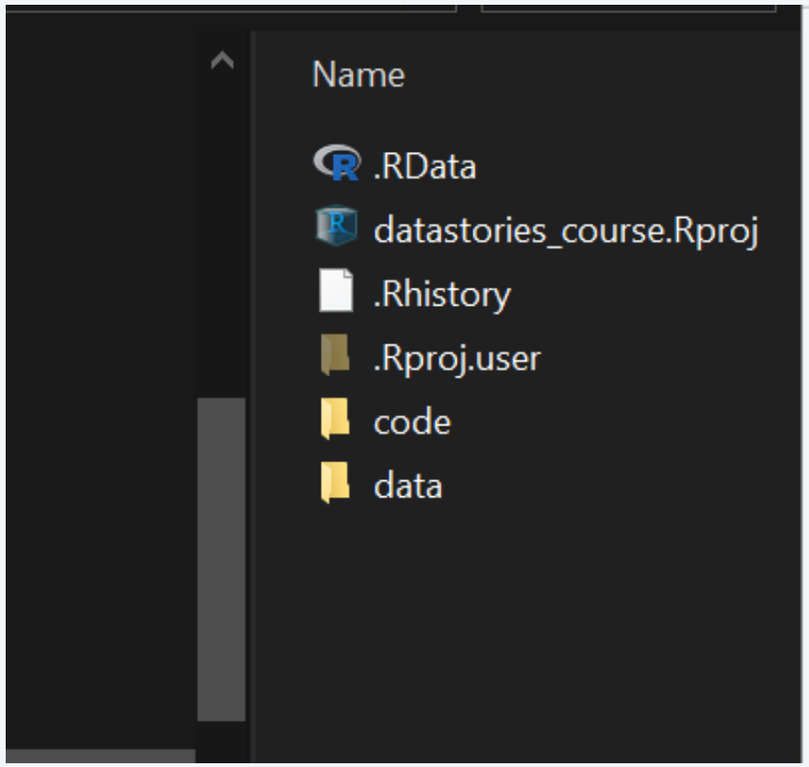






Folder structure

- It's a good idea to separate code and data



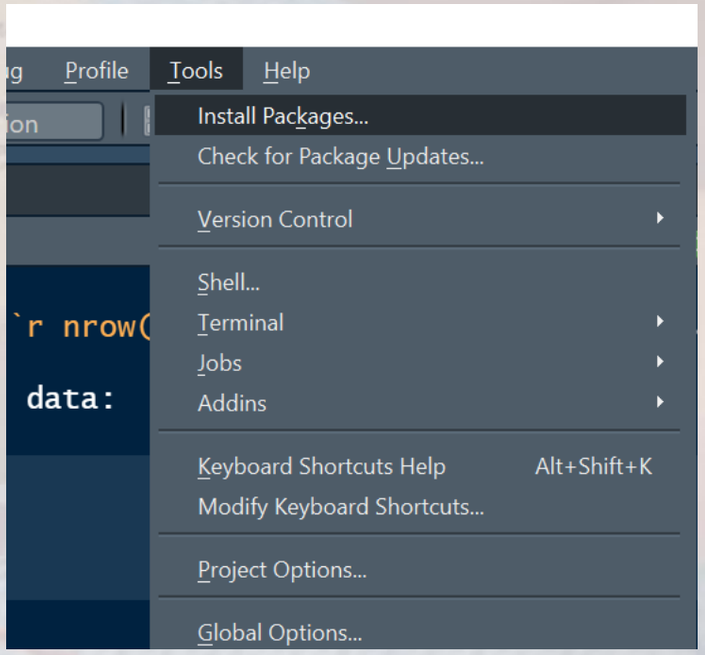
- You have to mindful about the active directory
- Also it's good to use relative paths.
- Play a little with the following code to work out how:

```
 getwd()
## [1] "C:/Users/Ralf Martin/Dropbox/datastories/datastorieshub"
 df=read.csv("../data/foreigners.csv")
 setwd("../data")
 getwd()
## [1] "C:/Users/Ralf Martin/Dropbox/datastories/data"
 df=read.csv("foreigners.csv")
```

Packages

- The power of R is in extensions that are created by many different contributors (will you become one?)
- Before you can use a package you need to install it and load it.
- Installing you only need to do once per computer
- Loading is necessary each time you want to use for a given R session.
- Note that sometimes different packages use the same name for a command that does not necessarily behave in the same way.

- To install packages you can use

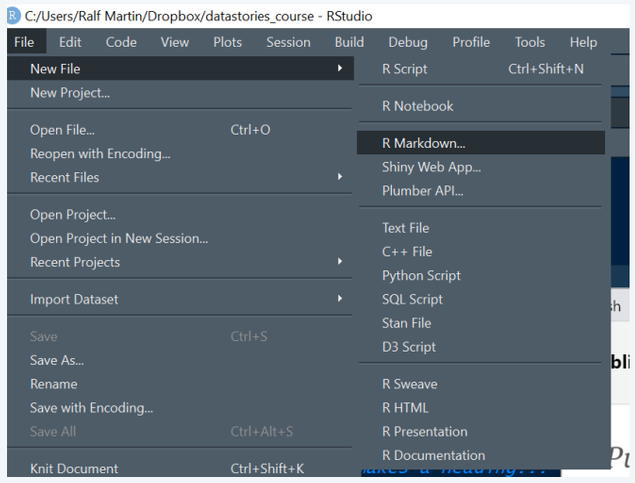


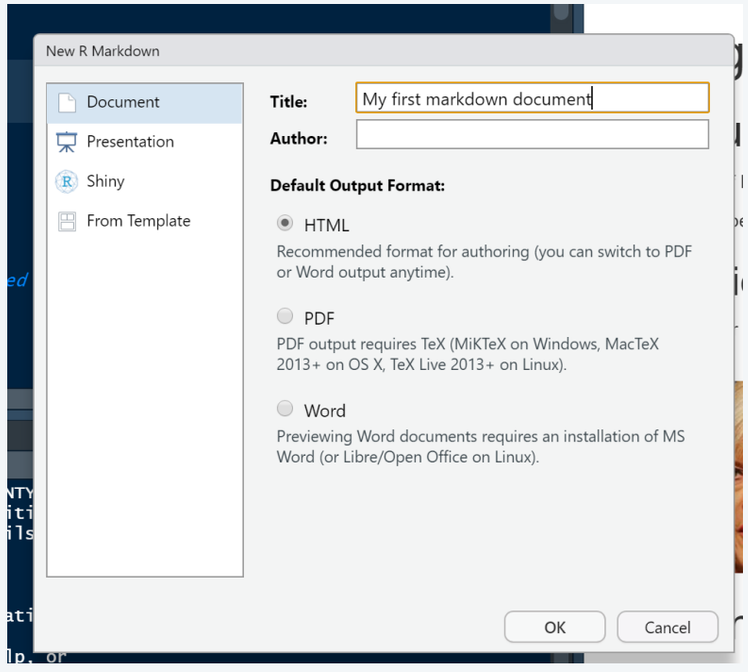
- To load packages after install you can use the library() command.
- Some packages we definitely need include: ggplot2, dplyr, haven

- To check which packages you have loaded use (.packages())

R Markdown

- There is another type of script file called an R Markdown file with .Rmd file extension
- This is like a normal script file but more powerful, because we can blend R code with R results and other content.
- This can be used to create e.g. dashboards, pdfs, word documents or webpages.
- Let's create our first R webpage to workout how.



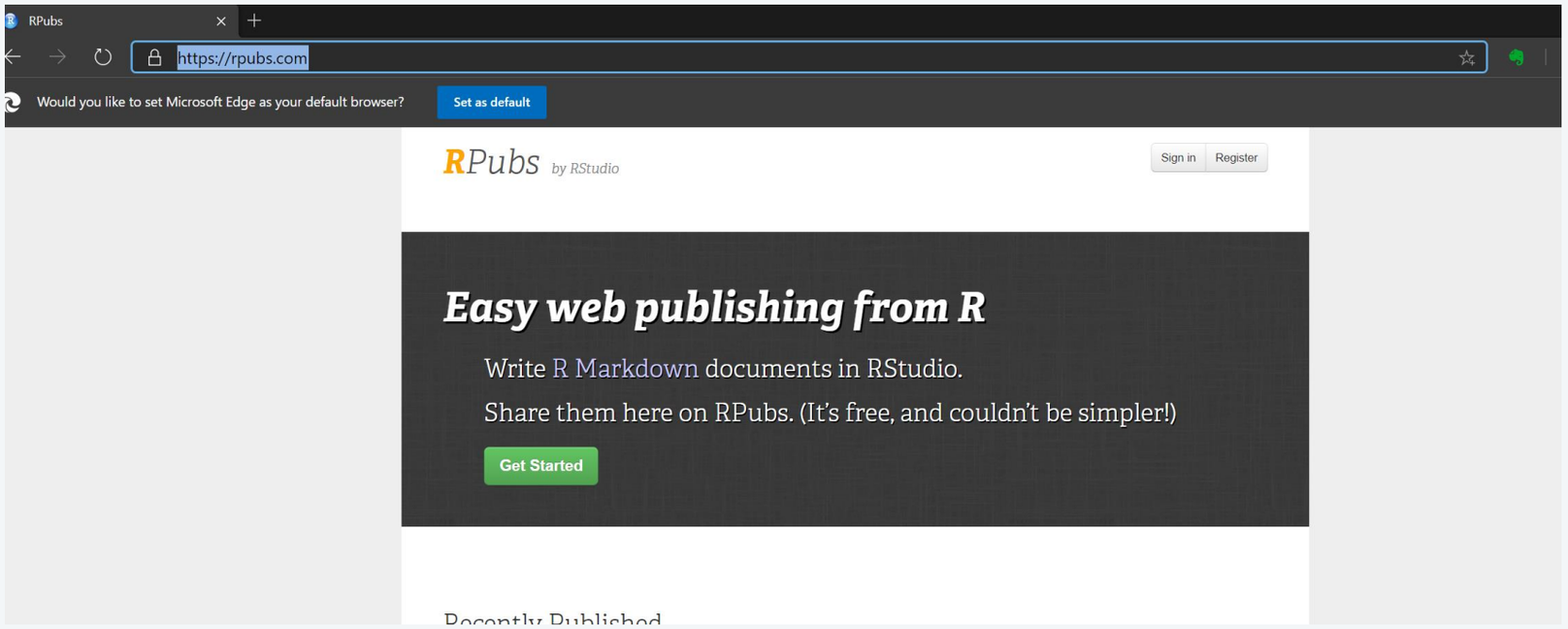


- Save this in your code folder and/or download an Rmd document (FarageGarage.Rmd) I have already created here

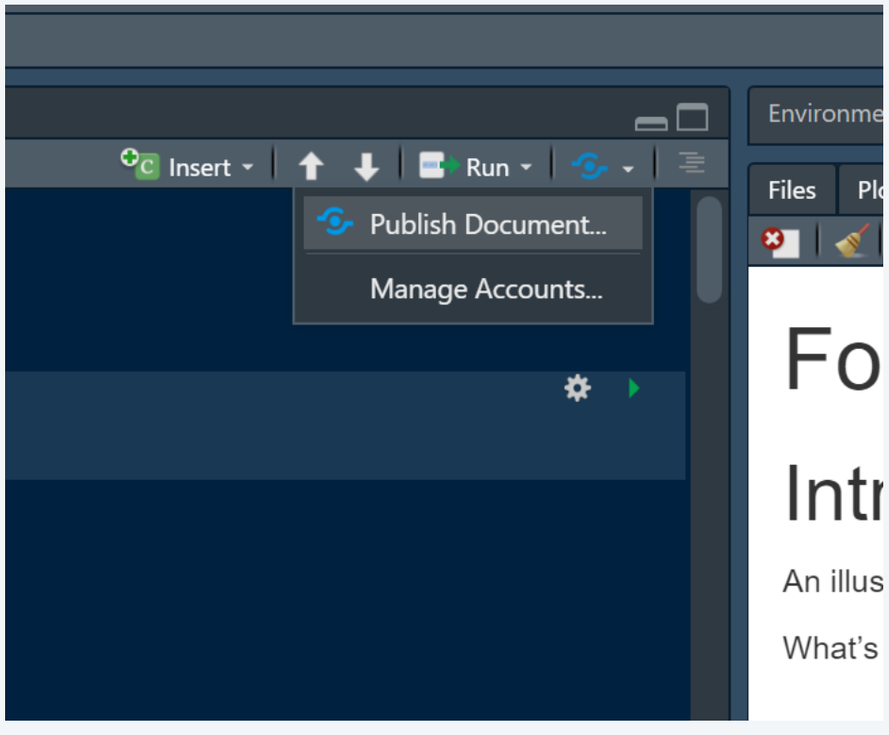
- Let's start **playing** with this code

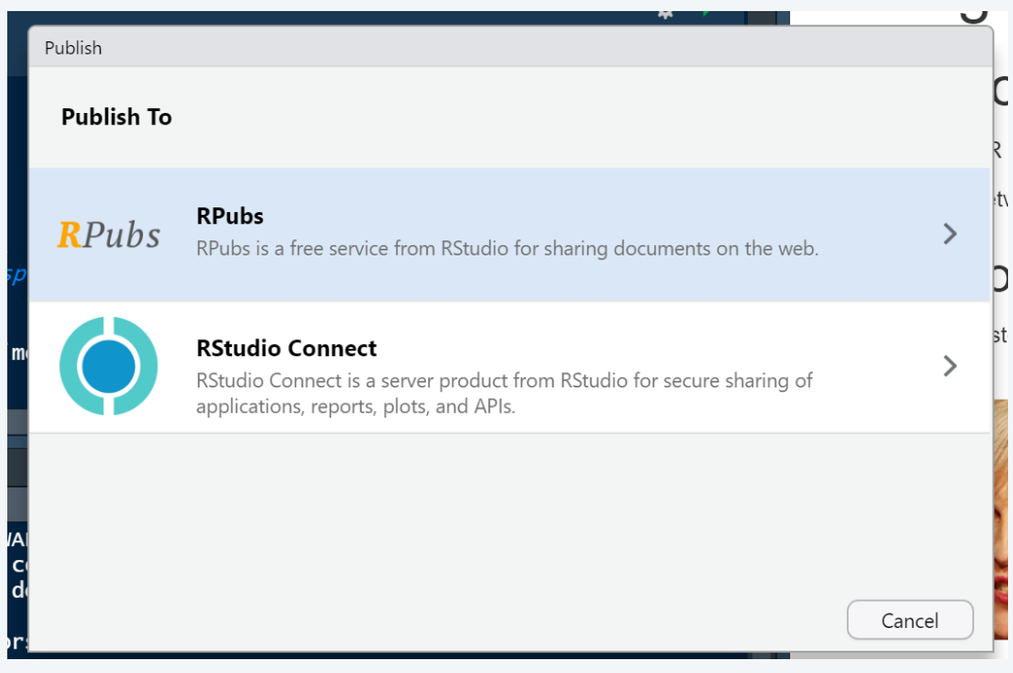
To publish online

- Sign up for account on Rpubs:



- Once you have an account you can publish an html document via the publish button





Fitting a line = Rrrrunning Rrrregressions

- We said that putting in a trend line in a scatter plot is a way of estimating an
- econometric model that describes the relationship between the dependent (or outcome) variable on the Y axis and an explanatory variable on the X axis.
- If you want a computer to do this for you (rather take out a ruler and a pen) you need a precise algorithm.
- The most commonly used algorithm for that is called Ordinary Least Squares estimator (OLS).

```
r1=lm(crimesPc~b_migr11,ff)
summary(r1)
```

```
##
## Call:
## lm(formula = crimesPc ~ b_migr11, data = ff)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.13314 -0.33959 -0.06763  0.22302  2.92572
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.091273   0.045146   24.17  < 2e-16 ***
## b_migr11    0.025164   0.002922    8.61 3.33e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5482 on 321 degrees of freedom
## Multiple R-squared:  0.1876, Adjusted R-squared:  0.1851
## F-statistic: 74.14 on 1 and 321 DF,  p-value: 3.325e-16
```

β_0

β_1

Ordinary Least Squares Regression (OLS)

Note this identifies a linear model of the form

$$Y_i = \beta_0 + \beta_1 \times X_i + \epsilon_i$$

where Y_i is *Crimes per capita* and X_i is the share of foreigners in %. This also shows you how you can integrate formulas in a markdown document and how easily format text to make it italic. More on this under this link.
Now this of course is the true model. What we get out of the OLS procedure is an estimate of the above model; i.e.

$$Y_i = \hat{\beta}_0 + \hat{\beta}_1 \times X_i + \hat{\epsilon}_i$$

which given the above R results becomes

$$Y_i = 1.091 + 0.025 \times X_i + \hat{\epsilon}_i$$

where we round the results up to 3 digit precision.

Interpreting estimation results → *Always depends on the units of X & Y*

Here: A one percentage point increase in the share of foreigners leads to 0.025 more crimes per capita in a given year

Note: This is not necessarily a statement of fact as it depends on the precision of the estimate and the possibility of bias. Rather: it is the implication of our estimate if we took it at face value.

How does the OLS algorithm work?

- R finds the estimates of β_0 and β_1 by minimising the sum of squared residual (hence least squares)
- A cool way of writing this down is as follows: $\min_{\hat{\beta}_0,\hat{\beta}_1} \sum_i \hat{\epsilon}_i$

It's a good exercise to try to do this if you are used to calculus and algebra but I don't expect this from you in any assessment

- With simple calculus you can show that this leads to the following formulas

$$\hat{\beta}_1 = \frac{Cov(X_i,Y_i)}{Var(X_i)}$$

$$\hat{\beta}_0 = Mean(Y) - \hat{\beta}_1 Mean(X)$$

An important implication of the OLS algorithm

An important implication of the OLS estimator is that it works out the β parameters in a way that sets correlation between $\hat{\epsilon}$ and X equal to 0. To check that this is the case we extract the ϵ from the `r1` variable and add it to our `ff` dataframe.

```
ff['residuals']=r1$residuals
#Note this is an alternative way of assigning a new variable to a dataframe.
#Alternatively use:
ff=ff %>% mutate(residuals=r1$residuals)
```

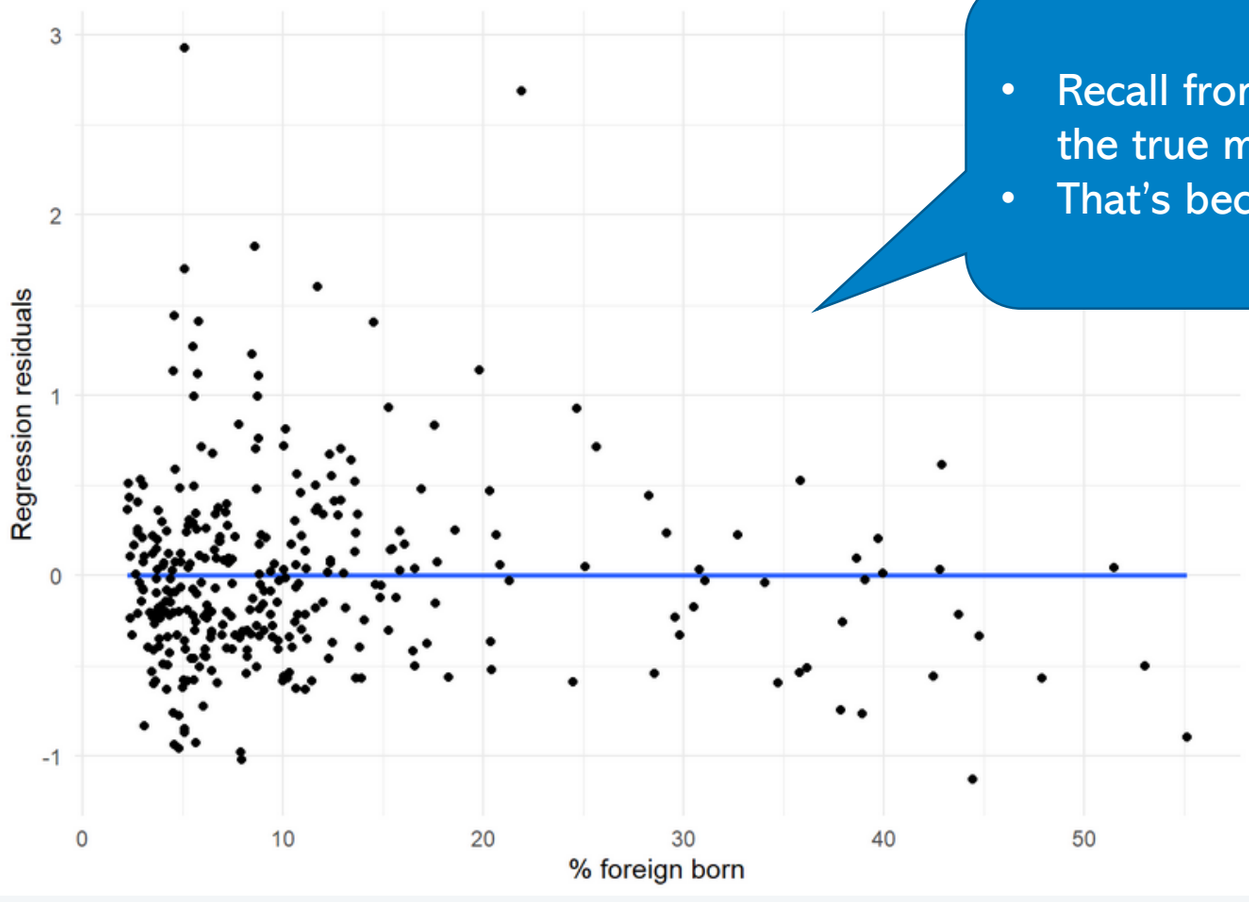
Note that the correlation is indeed 0:

```
  cor(ff %>% select(residuals,b_migr11),use="complete.obs")
```

```
##             residuals    b_migr11
## residuals 1.00000e+00 3.31048e-17
## b_migr11  3.31048e-17 1.00000e+00
```

That is virtually 0

Which we can also see in a scatterplot:



- Recall from the last lecture: We get biased estimates if in the true model X and ϵ are correlated.
- That's because in the estimated model they are **not**

Merging/Joining data

ID	Variable 1
A	4
B	21
C	3

ID	Variable 2
B	6
C	5
D	4

Full join

ID	Variable1	Variable 2
A	4	NA
B	21	6
C	3	5
D	NA	4

Inner join

ID	Variable1	Variable 2
B	21	6
C	3	5

Left join

ID	Variable1	Variable 2
A	4	NA
B	21	6
C	3	5

Right join

ID	Variable1	Variable 2
B	21	6
C	3	5
D	NA	4

MeRrrrrging

```
library(dplyr)

ff_more=read.csv("https://www.dropbox.com/s/gwq2wmmxr8s3v7t/foreigners_more.csv?dl=1")

names(ff_more)
```

```
##  [1] "X"          "urate2010" "urate2012" "urate2004" "urate2011" "area"
##  [7] "meanage"    "medianage" "region"    "pct_leave" "mus_sh"    "citshare"
## [13] "urbshare"
```

```
inner=ff %>% inner_join(ff_more,by="area")
full=ff %>% full_join(ff_more,by="area")
```

Dataframe `ff` has 323 observations. Dataframe `ff_more` has 348. The inner join has 323, the full join 348.

Full join

b_migr11	pop11	area	crimesPc	residuals	X.y	urate2010
2.240523	69814.03	Blaenau Gwent	1.5112720	0.36361862	334	12.6
2.731879	91074.98	Torfaen	1.3946854	0.23466762	335	8.6
4.516303	91319.11	Monmouthshire	0.9971519	-0.20776880	336	5.8
8.797297	145739.90	Newport	2.0739816	0.76133472	337	9.8
4.321093	132976.01	Powys	0.7676344	-0.43237413	338	5.7
4.636715	58801.98	Merthyr Tydfil	1.7954323	0.58748151	339	11.1
NA	*NA*	Craven	*NA*	*NA*	182	3.7
NA	*NA*	Hambleton	*NA*	*NA*	183	6.2
NA	*NA*	Harrogate	*NA*	*NA*	184	5.4
NA	*NA*	Richmondshire	*NA*	*NA*	185	3.6

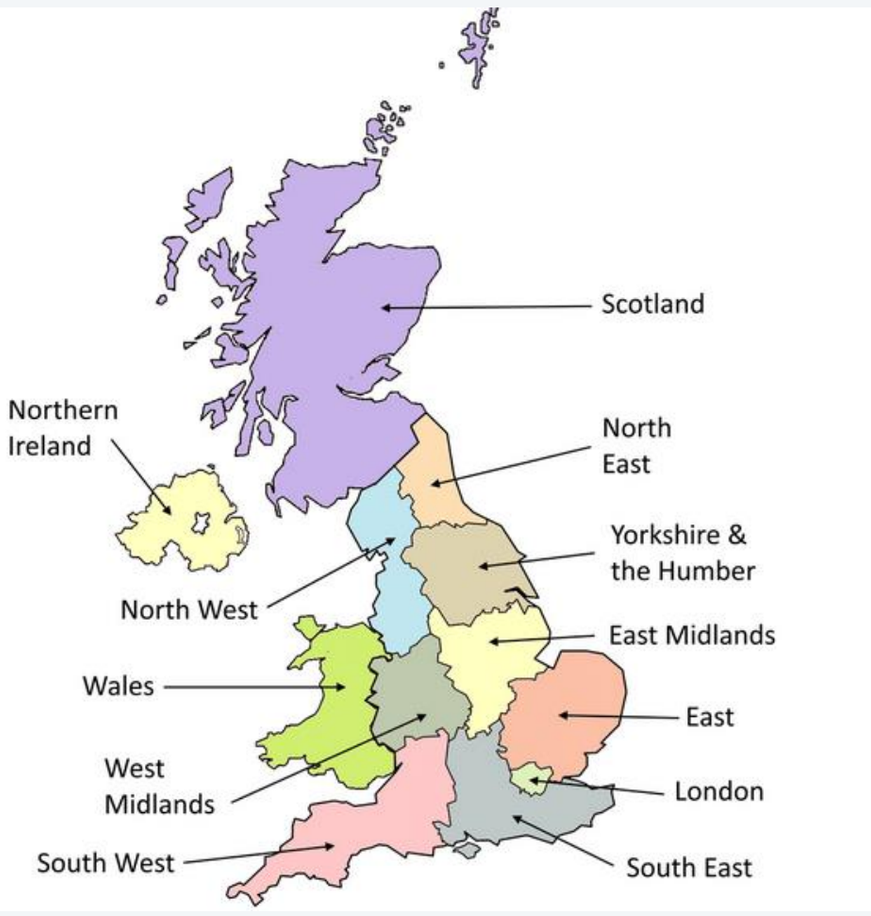
Functions

- Are you starting to like R commands?
- Turns out you can easily create your own
- For instance: Suppose you want to re-create the earlier scatter plot for the different regions of the UK/England

```
inner %>% group_by(region) %>% summarise(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 10 x 2
##    region                  `n()`
##    <fct>                   <int>
##  1 East                       43
##  2 East Midlands              40
##  3 London                     32
##  4 North East                 10
##  5 North West                 39
##  6 South East                 64
##  7 South West                 32
##  8 Wales                      22
##  9 West Midlands              27
## 10 Yorkshire and The Humber   14
```



Defining a function

```
plotter=function(r){

  ffx=inner %>% filter(region==r)

  plot=ggplot(ffx, aes(x = b_migr11, y = crimesPc)) +
    geom_smooth(method = "lm", se = FALSE) +
    geom_point()+theme_minimal()+
    xlab("% foreign born")+ylab("Crimes per capita")+
    ggtitle(r)


  reg=(lm(crimesPc~b_migr11,ffx))
  return(list(plot,reg))
}
```

Global variable

Local variable

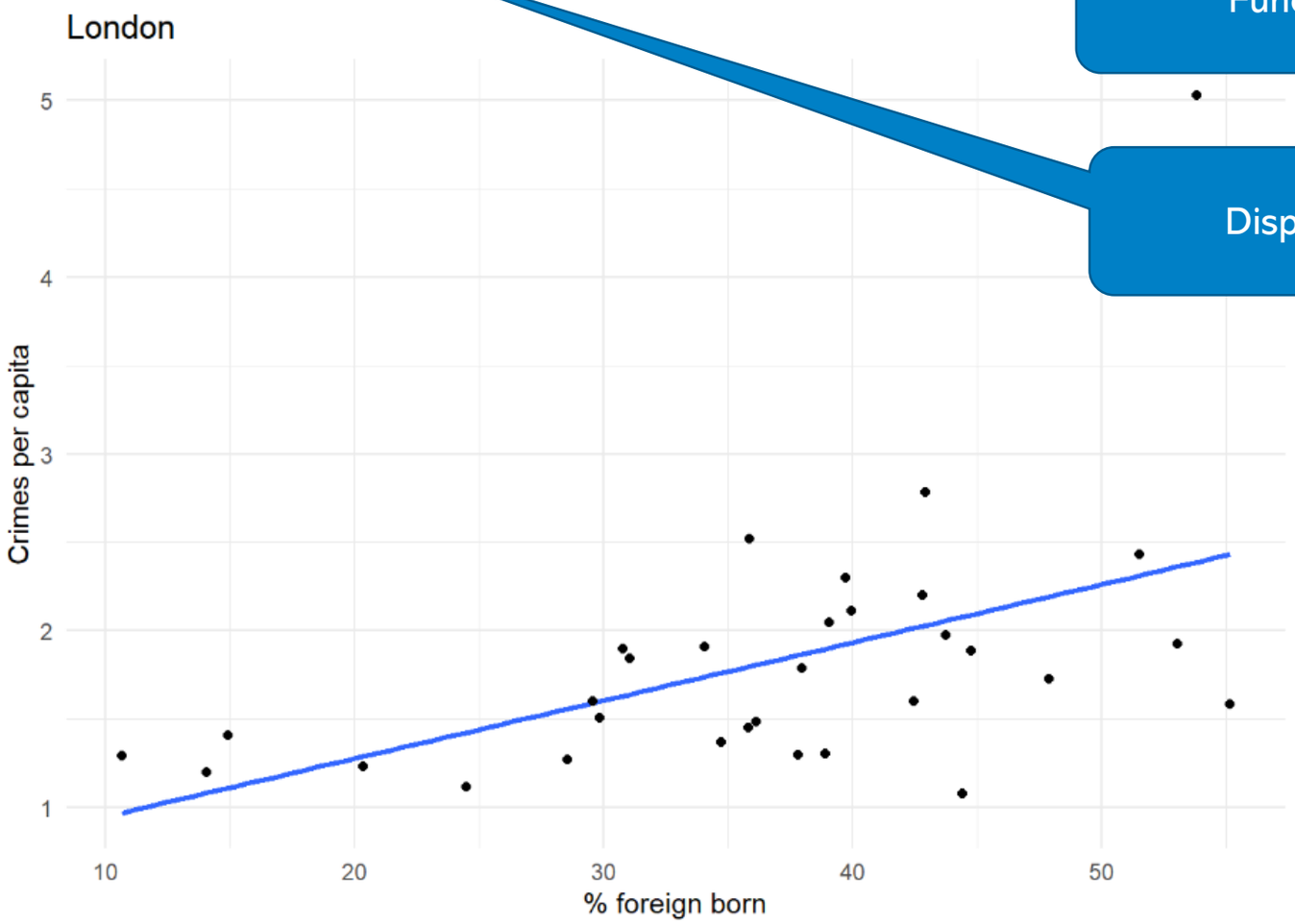
What the function returns
Here it is a list but it can be anything really

Calling functions

```
p=plotter("London")
p[[1]];summary(p[[2]])
```

Function call

Display results




```
##
## Call:
## lm(formula = crimesPc ~ b_migr11, data = ffx)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00220 -0.35611 -0.06671  0.18426  2.64380
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.62044    0.39045   1.589  0.12253
## b_migr11     0.03282    0.01025   3.202  0.00322 **
## ---
```

Loops

```
regions=inner$region %>% unique()

for(rrr in regions){

  p=plotter(rrr)
  print(p[1])
  print(summary(p[[2]]))

}
```

Takeaways

- R is a powerful piece of software that allows you to do statistical and econometric computation and visualisation and many other things
- Set up a dedicated directory and project file
- Get used to working with script files (preferably R Markdown files)
- Make sure to understand the LM command and OLS
- Make sure to understand merging of data
- Play with code:
  - If you see code that does something you like doing (e.g. from me) make sure you understand what different commands do
  - If you don’t understand a command google it (or use the help function) to understand it