Day 3

(Very rough) time plan

Friday Nov 17

13:15-14:00

- Introduction to R and RStudio
- Set up and get going
- Do Exercise 1

14:15 - 16:00

- Go through Exercise 1
- R packages and the Tidyverse
- Rectangular and tidy data
- Working with files
- Exercise 2
- Go through Exercise 2

Thursday Nov 23

09:15 - 10:30

- Manipulating data with dplyr
- Exercise 3

10:30 - 12:00

- Go through Exercise 3
- Basic plotting
- Exercise 4
- Go through exercise 4 together

14:15 - 17:00

- Programming basics
  - For loops + Ex 5 (14:15 – 15:00)
  - Ex 5 + If statements + Ex 6 (15:15 - 16:00)
  - Go through exercise 6 + wrap-up (16:00 – 17:00)

Friday Nov 24

09:15 – 12:00

- R scripts
  - Running R on the command line
  - Command line arguments
- Plotting with ggplot2 (not curriculum – brief demo + exercise)

Running R from the command line

R scripts

The first thing we’ll do is to log on to Fox cloud and enter your home directory.

From there type:

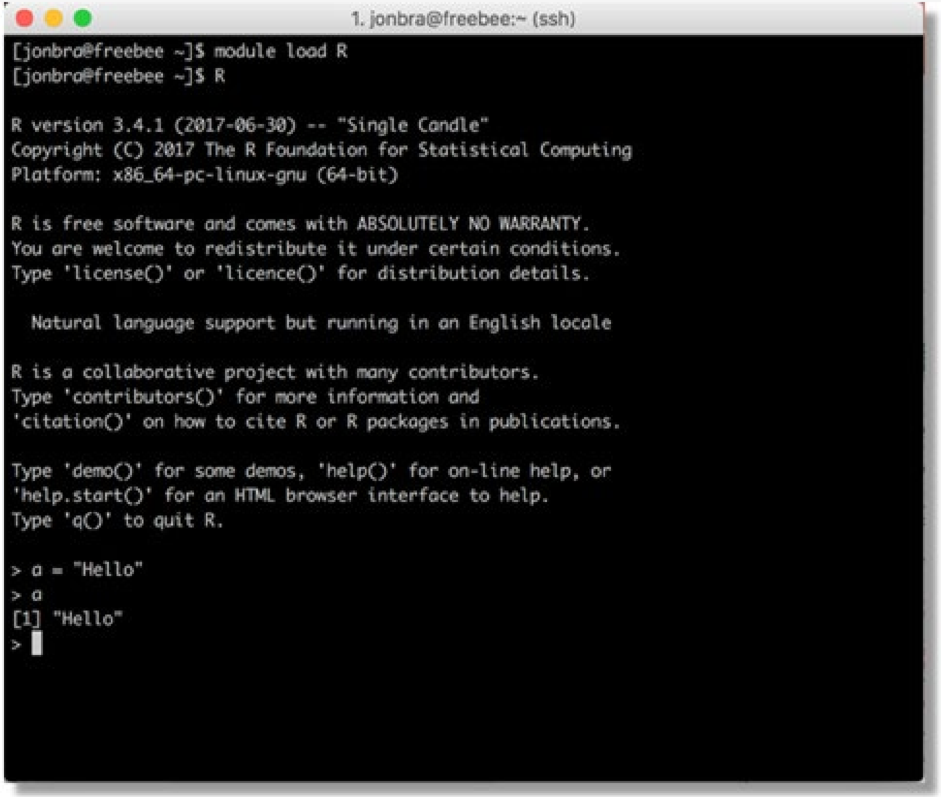
module load R/4.1.0-foss-2021a

Then start R by typing “*R*” and “Enter”.

You should see something similar to the image.

Activate tidyverse by typing

library(tidyverse)



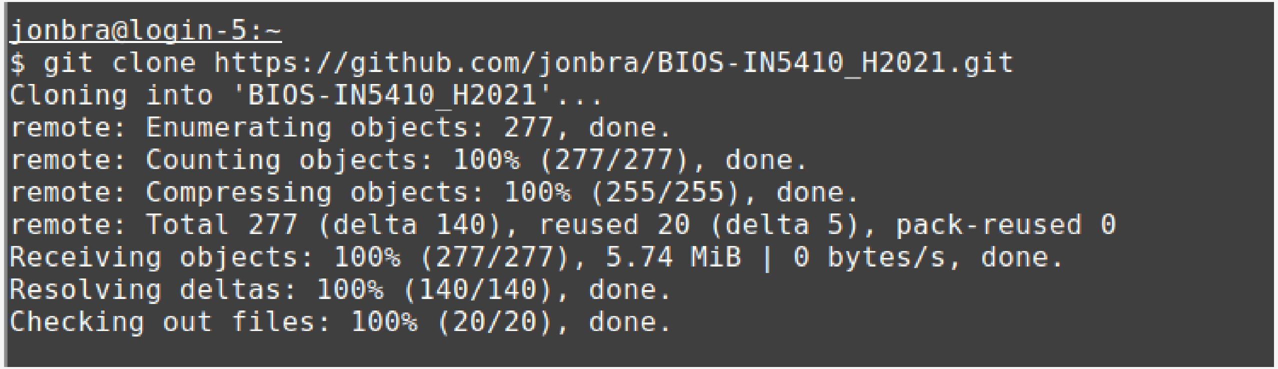
```
[jonbra@freebee ~]$ module load R
[jonbra@freebee ~]$ R

R version 3.4.1 (2017-06-30) -- "Single Candle"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> a = "Hello"
> a
[1] "Hello"
> 
```

R scripts

Then clone the BIOS-IN5410 GitHub repo (either my repo or your own copy if you've created one) to your home directory by first typing *cd* and Enter, and then:

git clone https://github.com/jonbra/BIOS-IN5410_H2021.git

(NB: use the https link).



```
jonbra@login-5:~
$ git clone https://github.com/jonbra/BIOS-IN5410_H2021.git
Cloning into 'BIOS-IN5410_H2021'...
remote: Enumerating objects: 277, done.
remote: Counting objects: 100% (277/277), done.
remote: Compressing objects: 100% (255/255), done.
remote: Total 277 (delta 140), reused 20 (delta 5), pack-reused 0
Receiving objects: 100% (277/277), 5.74 MiB | 0 bytes/s, done.
Resolving deltas: 100% (140/140), done.
Checking out files: 100% (20/20), done.
```

Exercise 7

Log on to Saga and do Exercise 7.

You can try it yourself, but I will go through each part separately and explain what is going on.

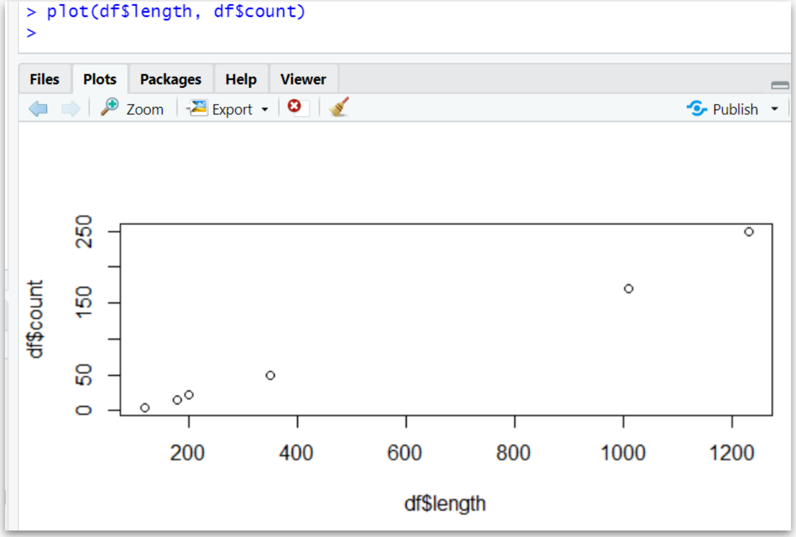
Plotting with ggplot2

Plotting with ggplot2

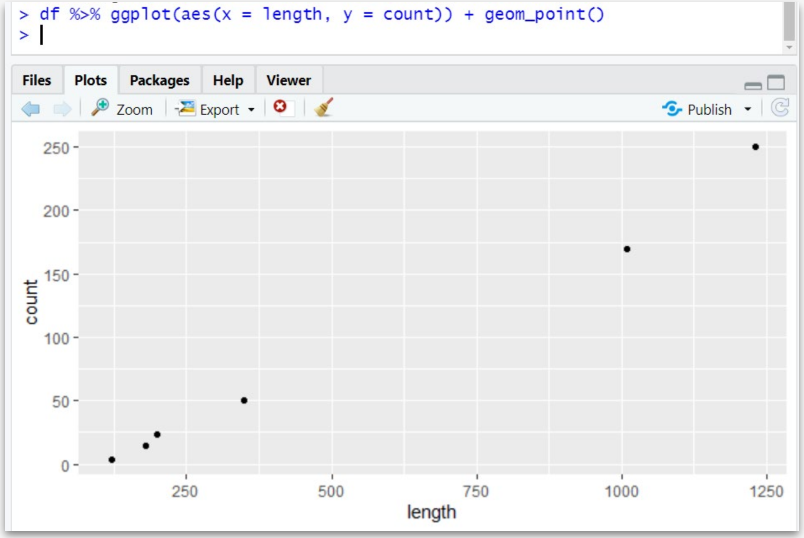
On Friday you made some simple plots with base R plotting functions.

You can make great plots with base R, don’t worry. But there’s a very popular package for plotting in R, ggplot2, that is very useful to know about.

ggplot2 is automatically activated when you load Tidyverse, and it’s particularly suited to operate on tidy data.



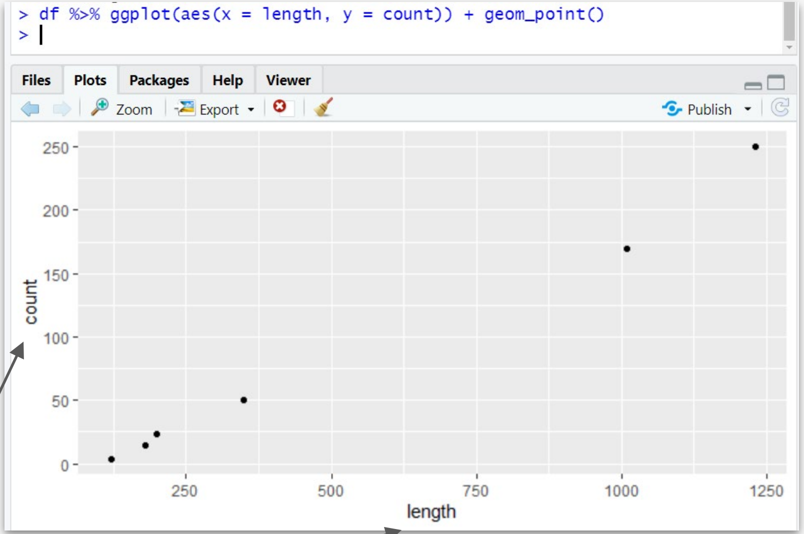
```
> plot(df$length, df$count)
>
```



```
> df %>% ggplot(aes(x = length, y = count)) + geom_point()
>
```

Plotting with ggplot2

```
> df
# A tibble: 6 x 3
  Gene   count  length
  <chr>  <dbl>   <dbl>
1 A          4     120
2 B         50     350
3 C         23     200
4 D        250    1230
5 E         15     180
6 F        170    1010


> df %>%
    ggplot(aes(x = length, y = count)) +
    geom_point()
```



```
> df %>% ggplot(aes(x = length, y = count)) + geom_point()
> |
```

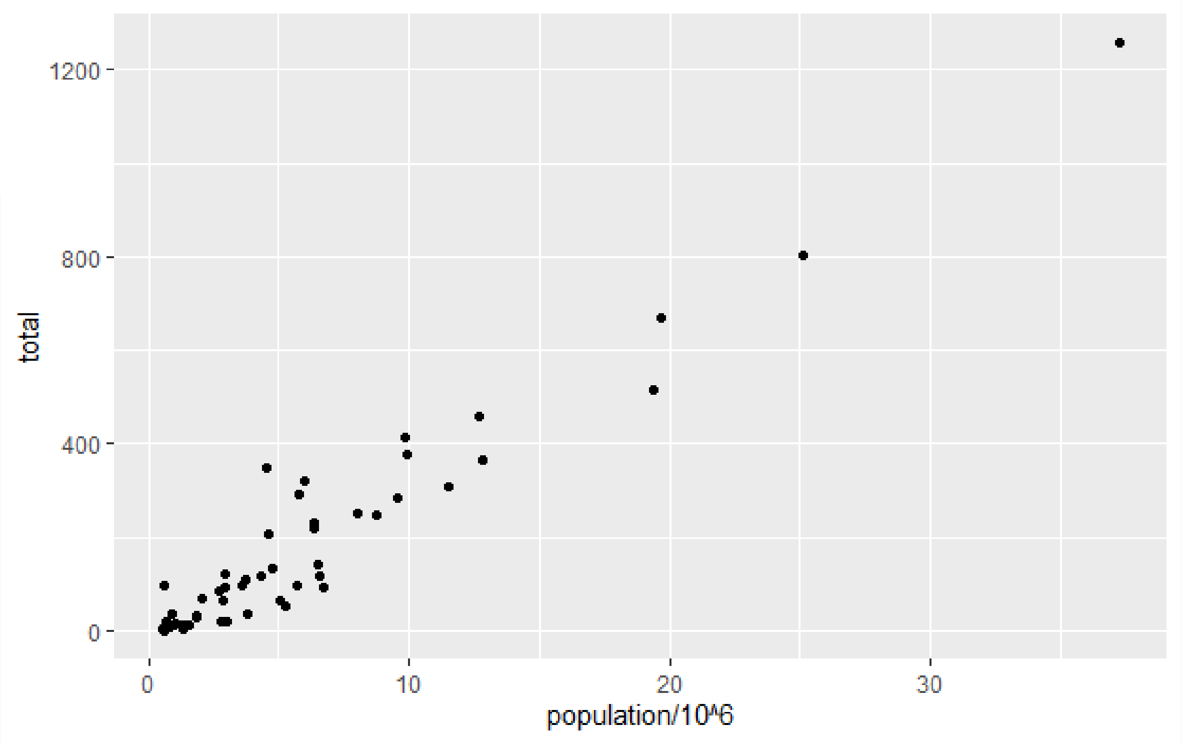
Plots are initiated with the function **ggplot()**. Then the different subfunctions are tied together in layers using the “+” symbol (like a “pipe”).

Aesthetics (aes) is a mapping of the variables in the data the different properties of the plot (the geom), like x and y axes, color, etc.

Plotting with ggplot2

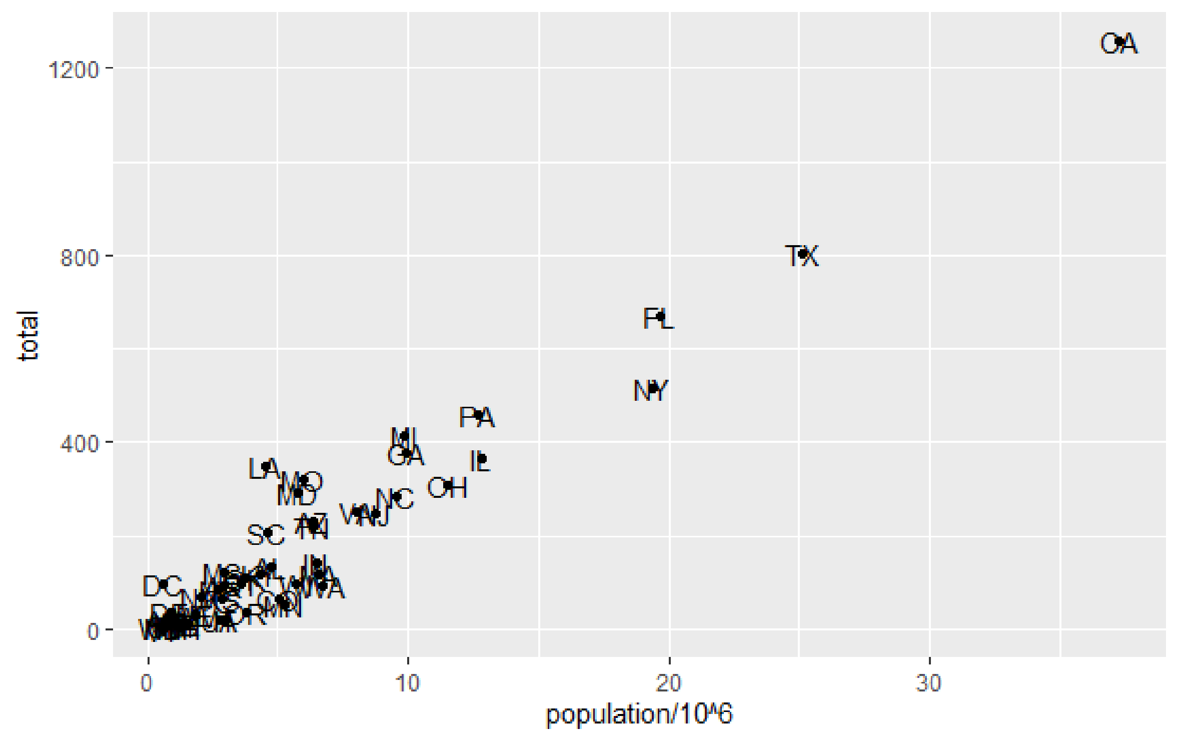
A quick demonstration of how ggplot2 plots are built up by adding layers

```
murders %>% ggplot() +
  geom_point(aes(x =
population/10^6, y = total))
```



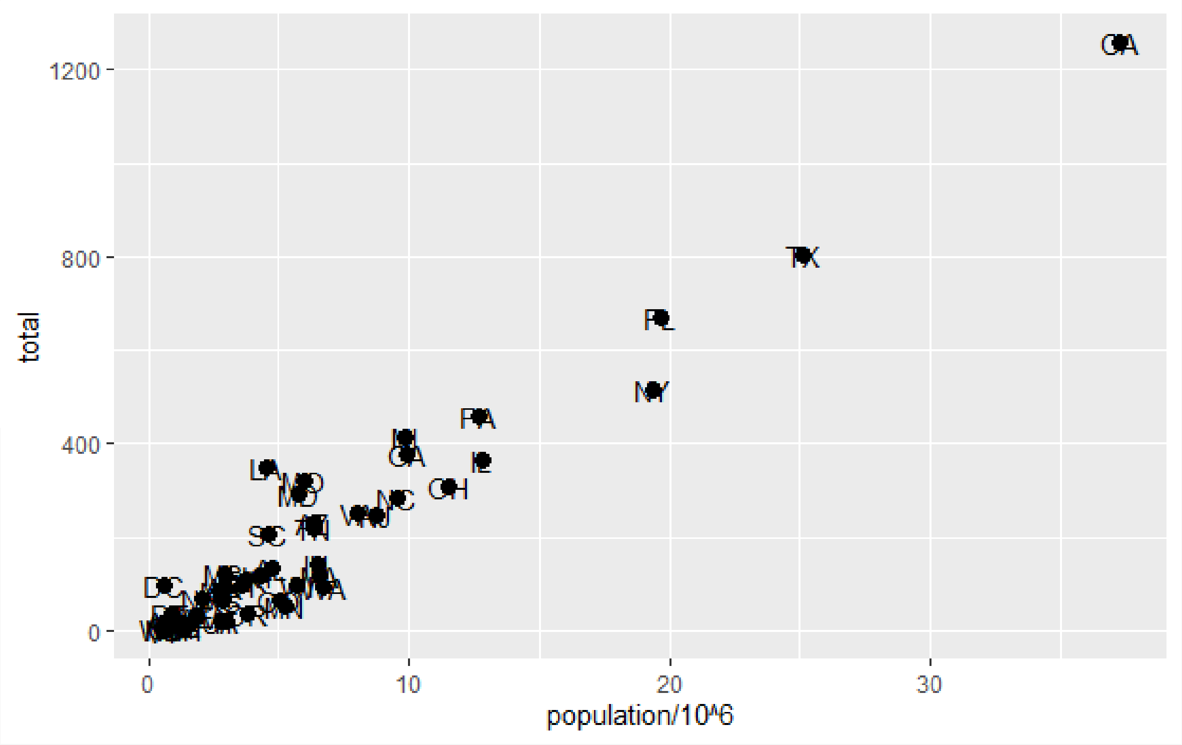
Plotting with ggplot2

```
murders %>% ggplot() +
  geom_point(aes(x =
population/10^6, y = total)) +
  geom_text(aes(population/10^6,
total, label = abb))
```



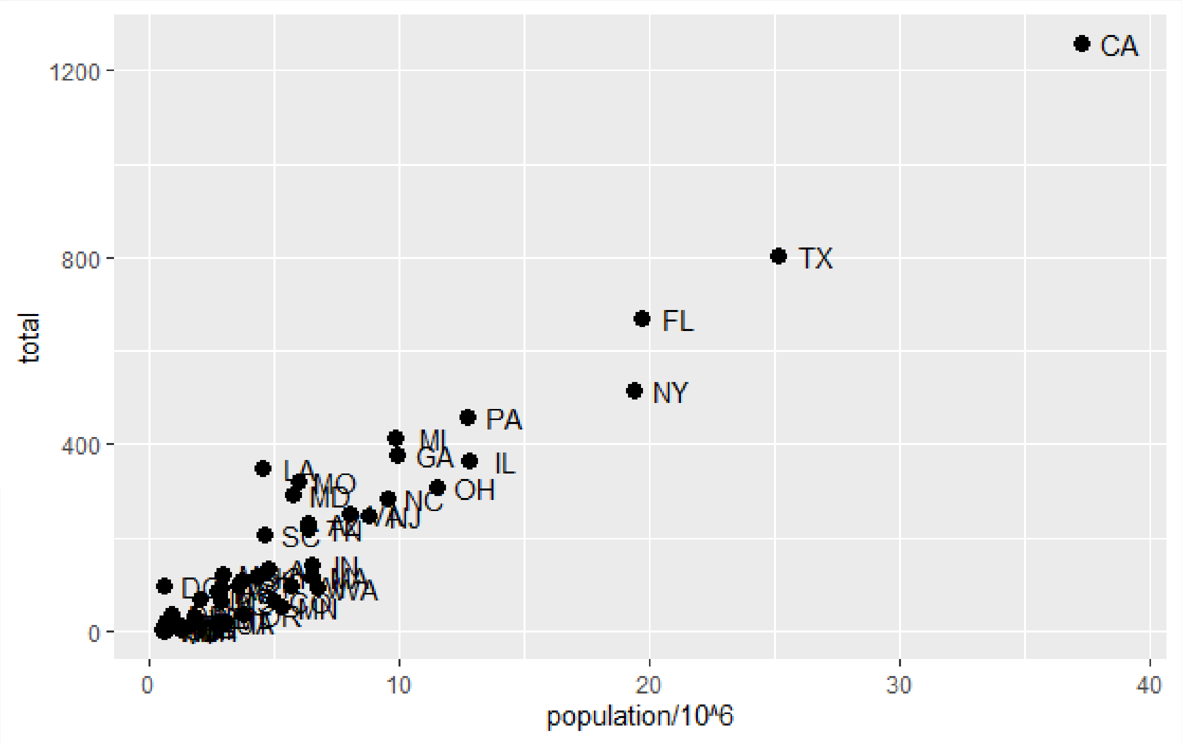
Plotting with ggplot2

```
murders %>% ggplot() +
  geom_point(aes(x =
population/10^6, y = total), size
= 3) +
  geom_text(aes(population/10^6,
total, label = abb))
```



Plotting with ggplot2

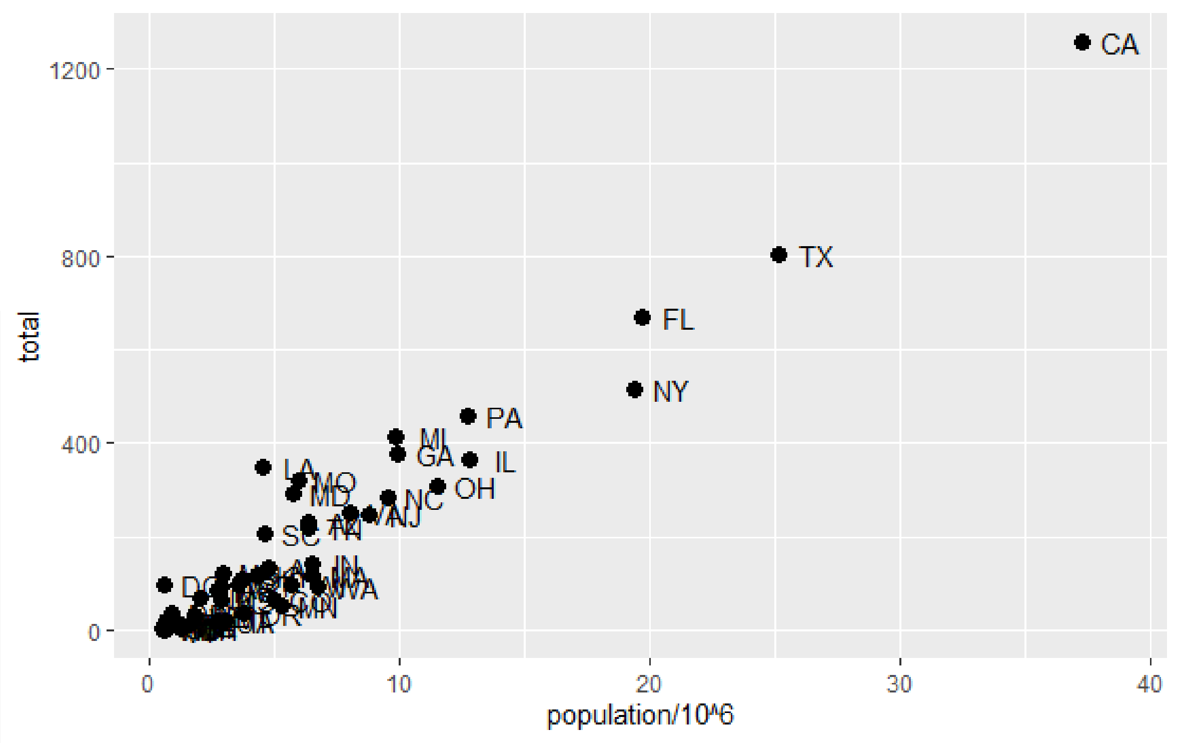
```
murders %>% ggplot() +
  geom_point(aes(x =
population/10^6, y = total), size
= 3) +
     geom_text(aes(population/10^
6, total, label = abb), nudge_x =
1.5)
```



Plotting with ggplot2

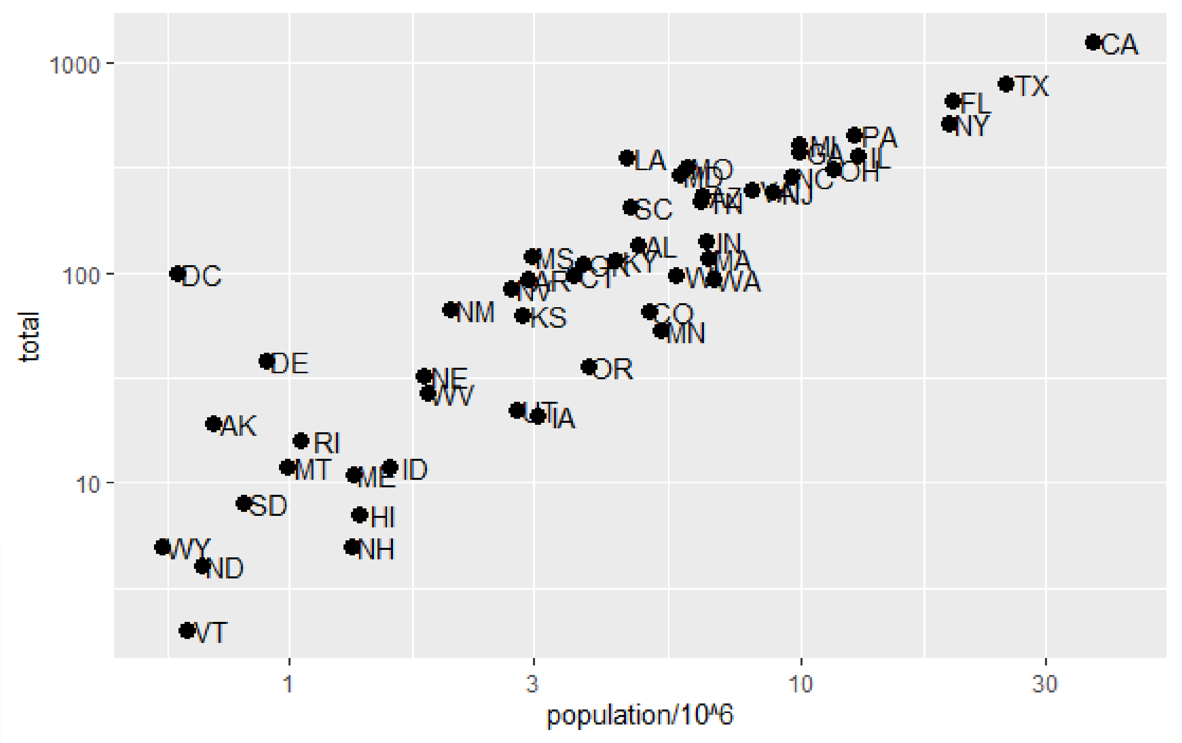
Global aesthetics. Apply to all layers

```
murders %>%
  ggplot(aes(population/10^6,
total, label = abb)) +
  geom_point(size = 3) +
  geom_text(nudge_x = 1.5)
```



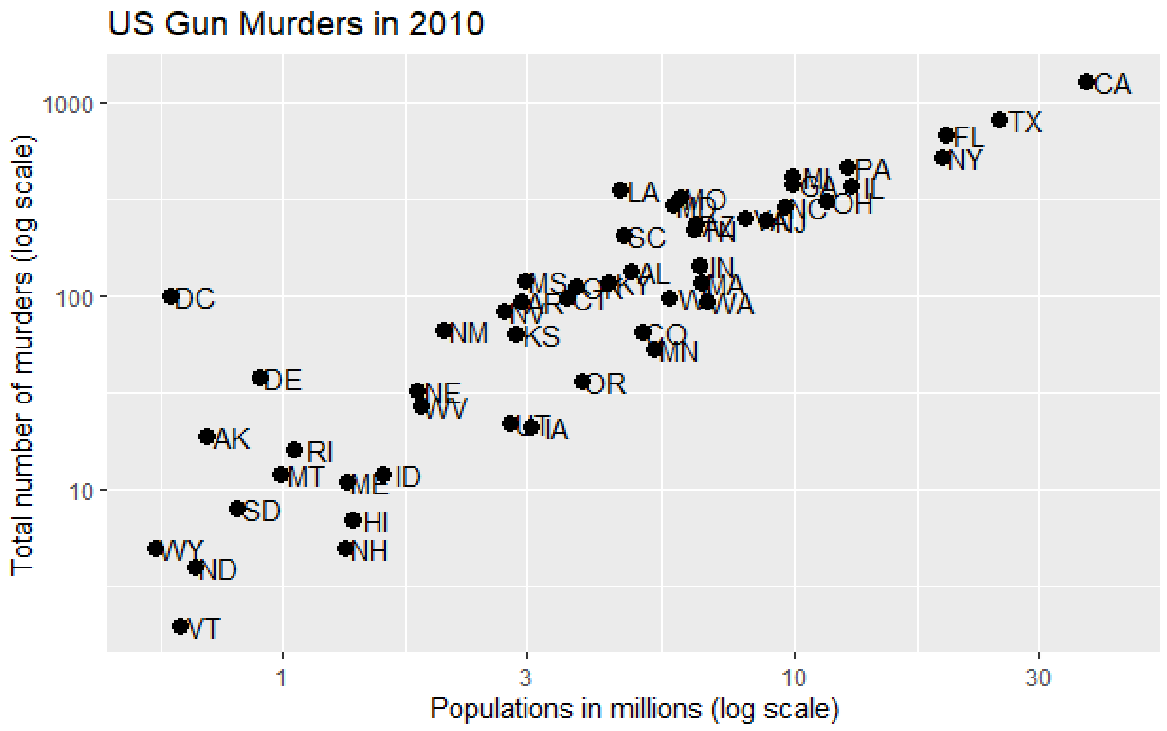
Plotting with ggplot2

```
murders %>%
  ggplot(aes(population/10^6,
total, label = abb)) +
  geom_point(size = 3) +
  geom_text(nudge_x = 0.05) +
  scale_x_continuous(trans =
"log10") +
  scale_y_continuous(trans =
"log10")
```



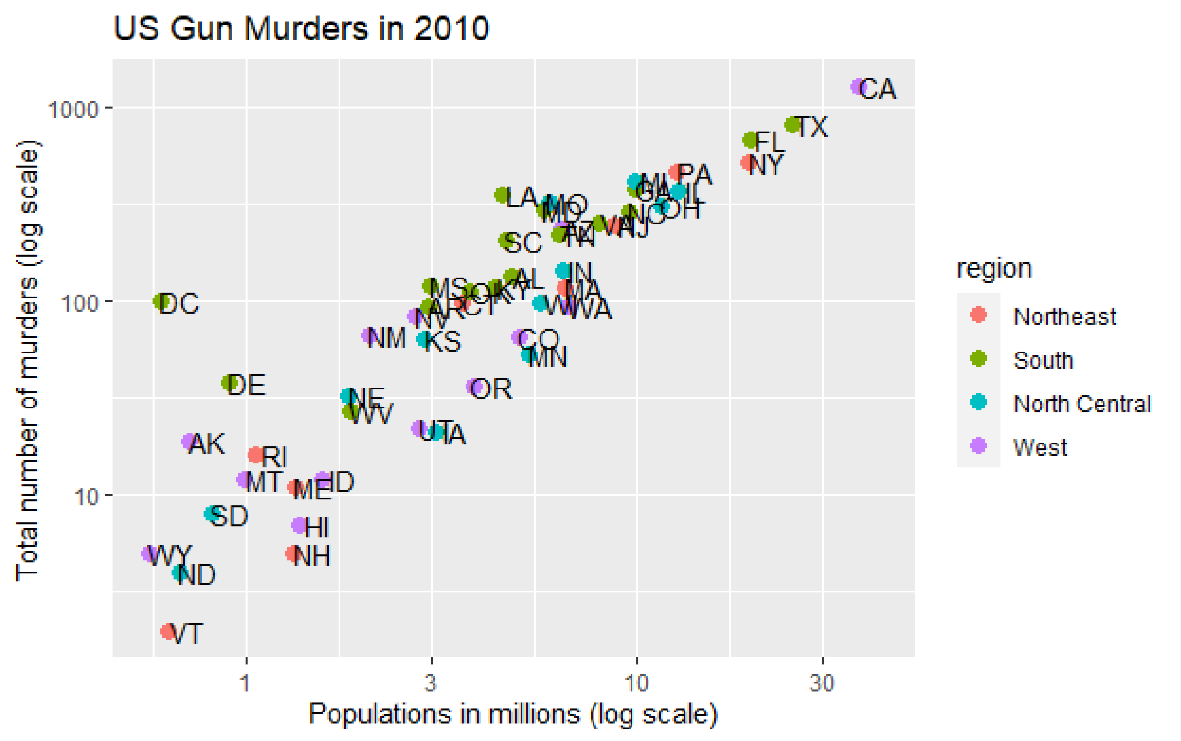
Plotting with ggplot2

```
murders %>%
  ggplot(aes(population/10^6,
total, label = abb)) +
  geom_point(size = 3) +
  geom_text(nudge_x = 0.05) +
  scale_x_continuous(trans =
"log10") +
  scale_y_continuous(trans =
"log10") +
  xlab("Populations in millions
(log scale)") +
  ylab("Total number of murders
(log scale)") +
  ggtitle("US Gun Murders in
2010")
```



Plotting with ggplot2

```
murders %>%
  ggplot(aes(population/10^6, total,
label = abb)) +
  geom_point(aes(col = region), size
= 3) +
  geom_text(nudge_x = 0.05) +
  scale_x_continuous(trans =
"log10") +
  scale_y_continuous(trans =
"log10") +
  xlab("Populations in millions (log
scale)") +
  ylab("Total number of murders (log
scale)") +
  ggtitle("US Gun Murders in 2010")
```



Do Exercise 8