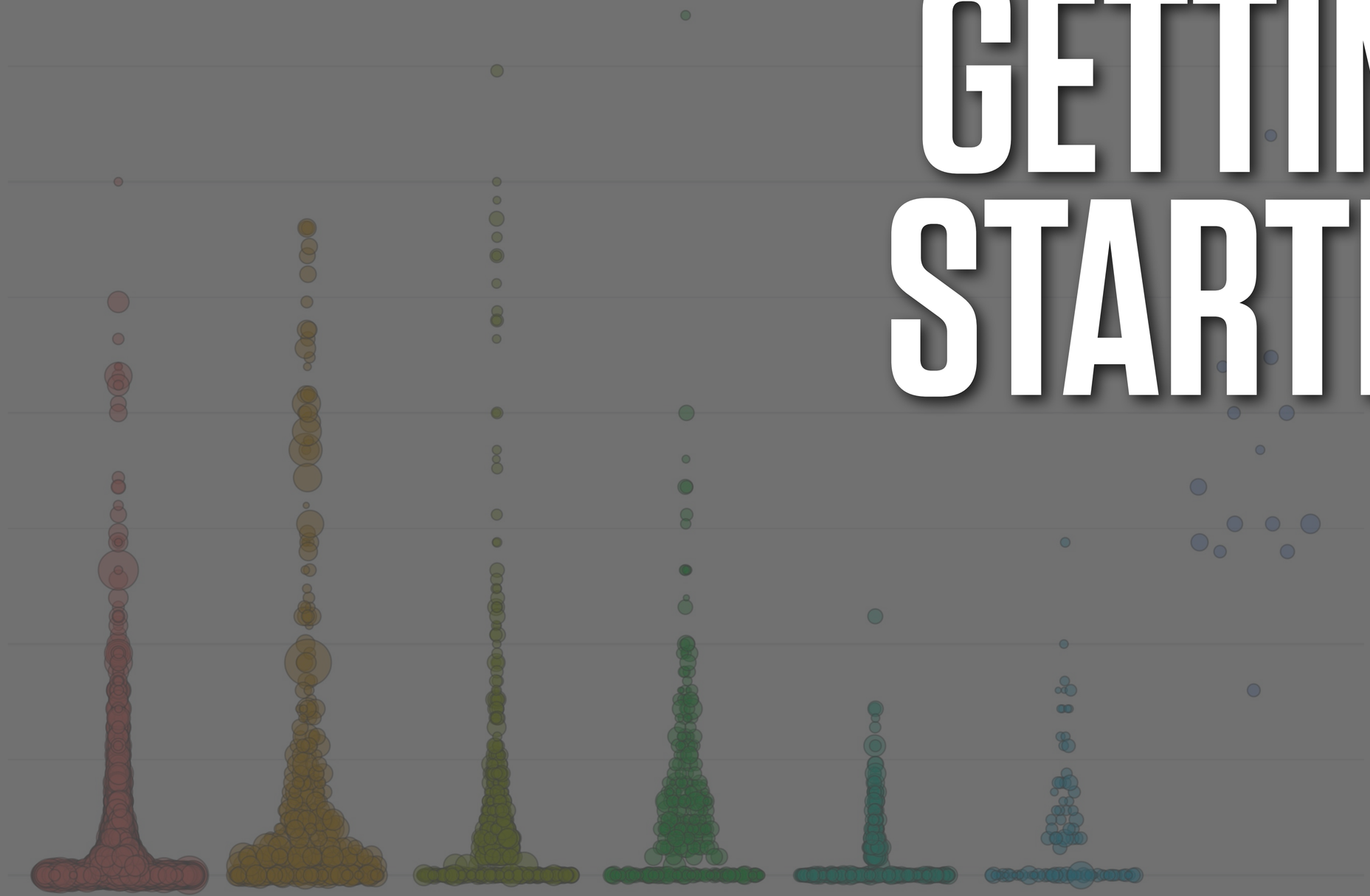
Data Visualization

rstudio::conf

GETTING STARTED



WE WANT TO DRAW GOOD DATA GRAPHICS REPRODUCIBLY

Abstraction in Software

Easy things are awkward

Hard things are straightforward

Really hard things are doable

Easy things are trivial

Hard things are really awkward

Really hard things are impossible

D3

Grid

ggplot

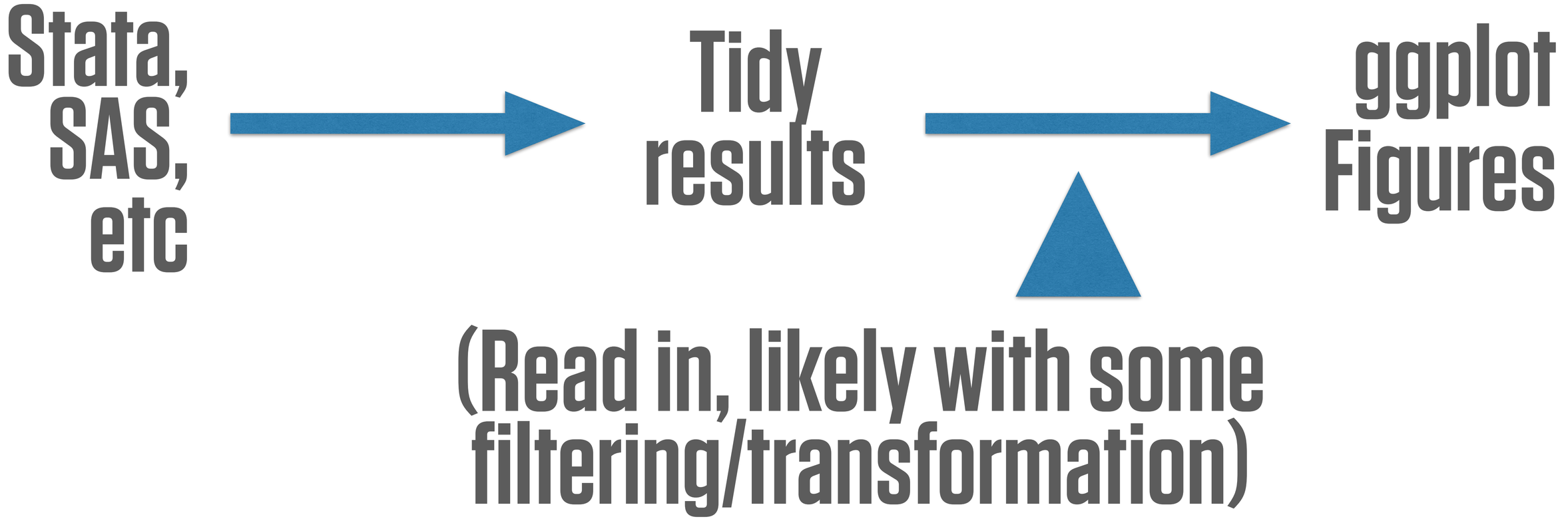
Stata

Excel

Two ways to use R and ggplot

1. Do Everything in R

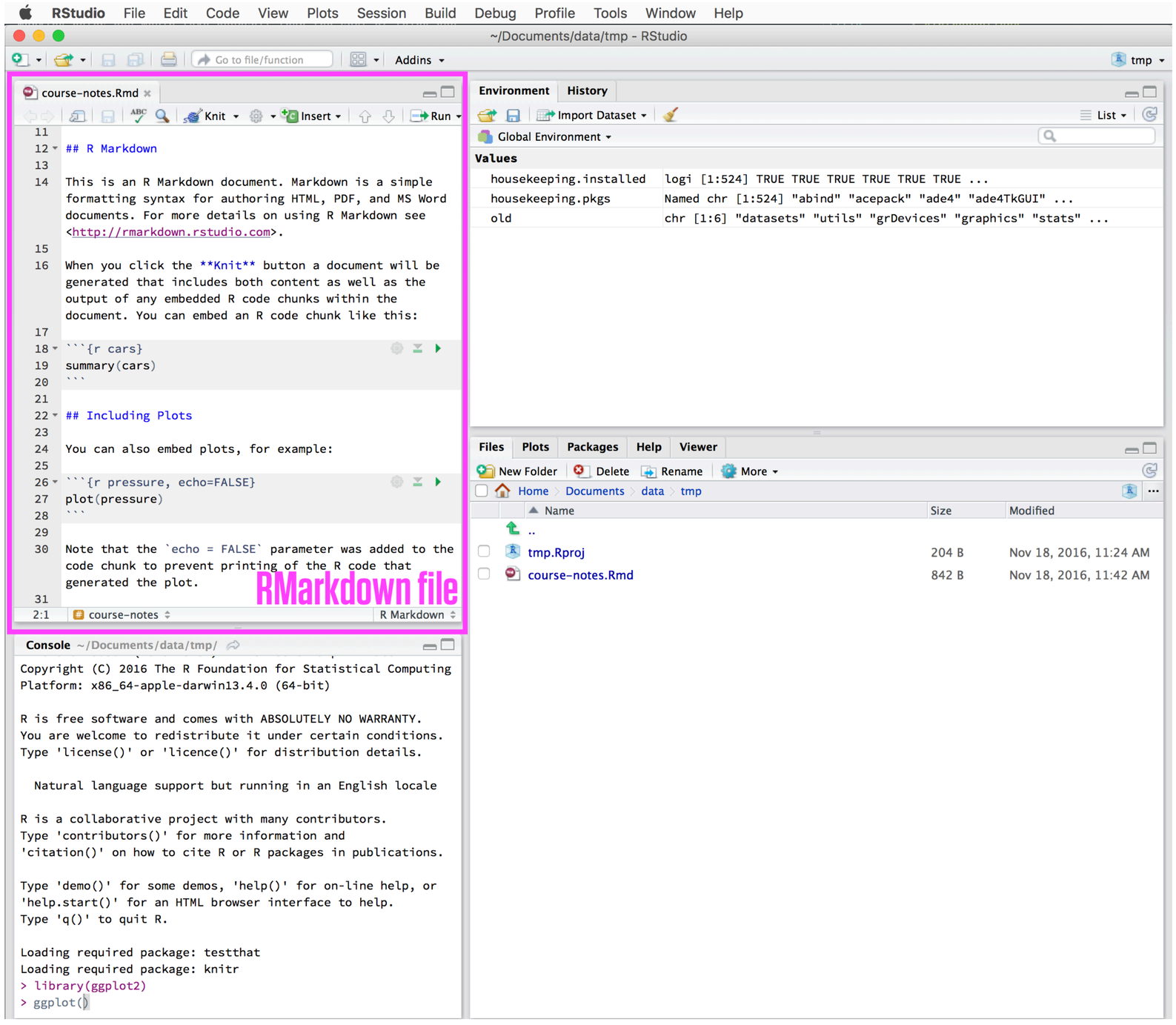
2. Just use ggplot



THE RIGHT FRAME OF MIND

TYPE OUT YOUR CODE BY HAND

RSTUDIO


RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help
~/Documents/data/tmp - RStudio
Go to file/function
Addins
tmp
course-notes.Rmd
Knit
Insert
Run
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
15
16 When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
17
18 ```{r cars}
19 summary(cars)
20 ```
21
22 ## Including Plots
23
24 You can also embed plots, for example:
25
26 ```{r pressure, echo=FALSE}
27 plot(pressure)
28 ```
29
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
31
2:1
course-notes
R Markdown
RMarkdown file
Console ~/Documents/data/tmp/
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (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.
Loading required package: testthat
Loading required package: knitr
> library(ggplot2)
> ggplot()
Environment History
Import Dataset
List
Global Environment
Values
housekeeping.installed logi [1:524] TRUE TRUE TRUE TRUE TRUE TRUE ...
housekeeping.pkgs Named chr [1:524] "abind" "acepack" "ade4" "ade4TkGUI" ...
old chr [1:6] "datasets" "utils" "grDevices" "graphics" "stats" ...
Files Plots Packages Help Viewer
New Folder
Delete
Rename
More
Home Documents data tmp
Name
Size
Modified
..
tmp.Rproj 204 B Nov 18, 2016, 11:24 AM
course-notes.Rmd 842 B Nov 18, 2016, 11:42 AM

~/Documents/courses/stathorizons/dataviz/notes - RStudio

Go to file/function | Addins | notes

Console ~/Documents/courses/stathorizons/dataviz/notes/

```
R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (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.

Loading required package: testthat
Loading required package: knitr
> library(ggplot2)
> ggplot(mpg, aes(displ, hwy)) +
+     geom_point() +
+     geom_smooth()
`geom_smooth()` using method = 'loess'
>
```

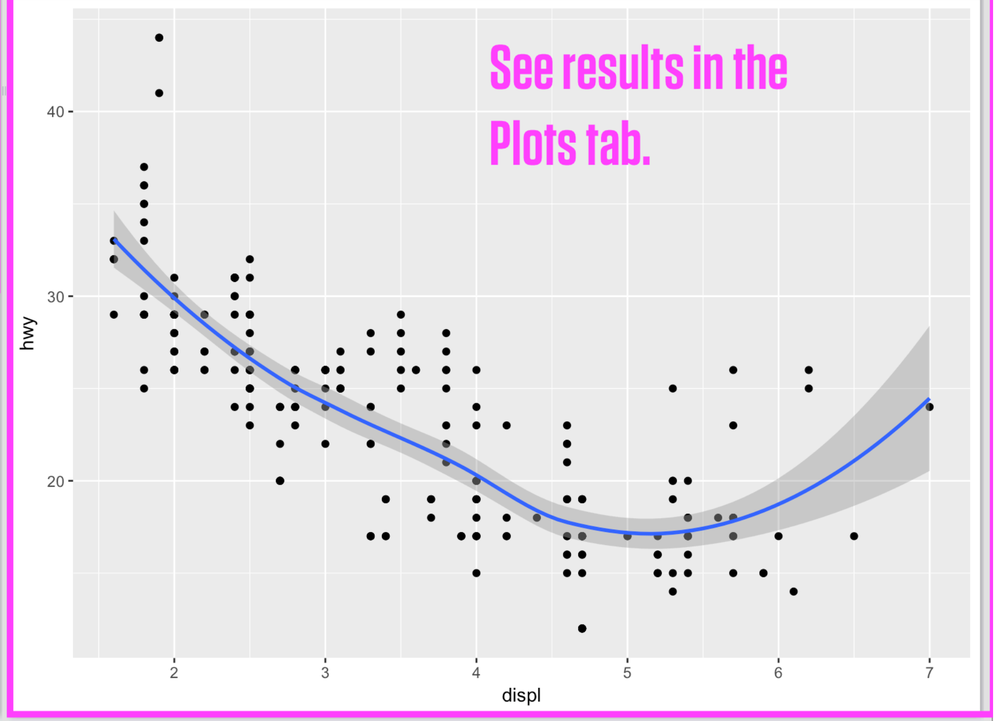
Type commands at the console.

Environment | **History** | **Build**

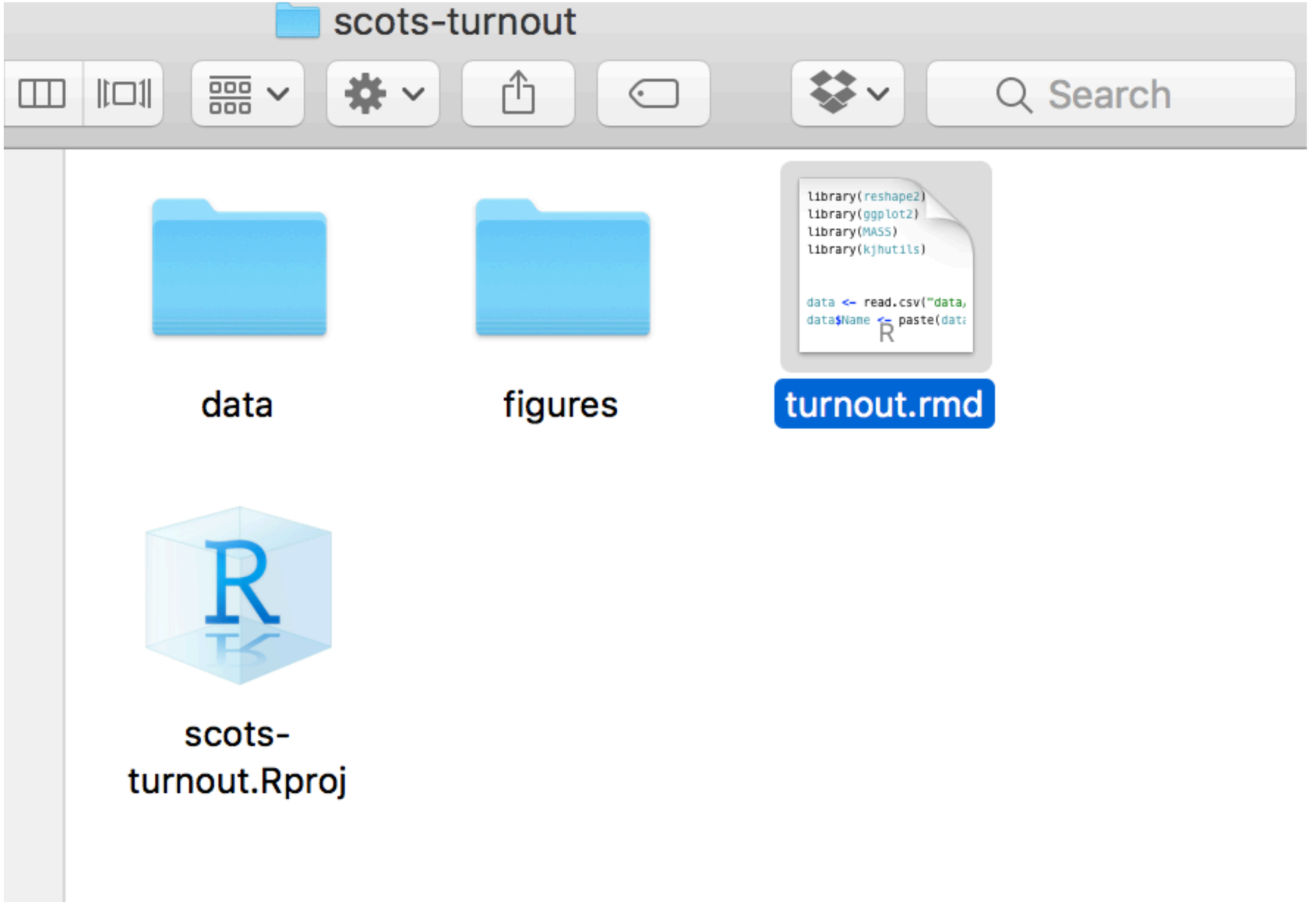
Build Book | More

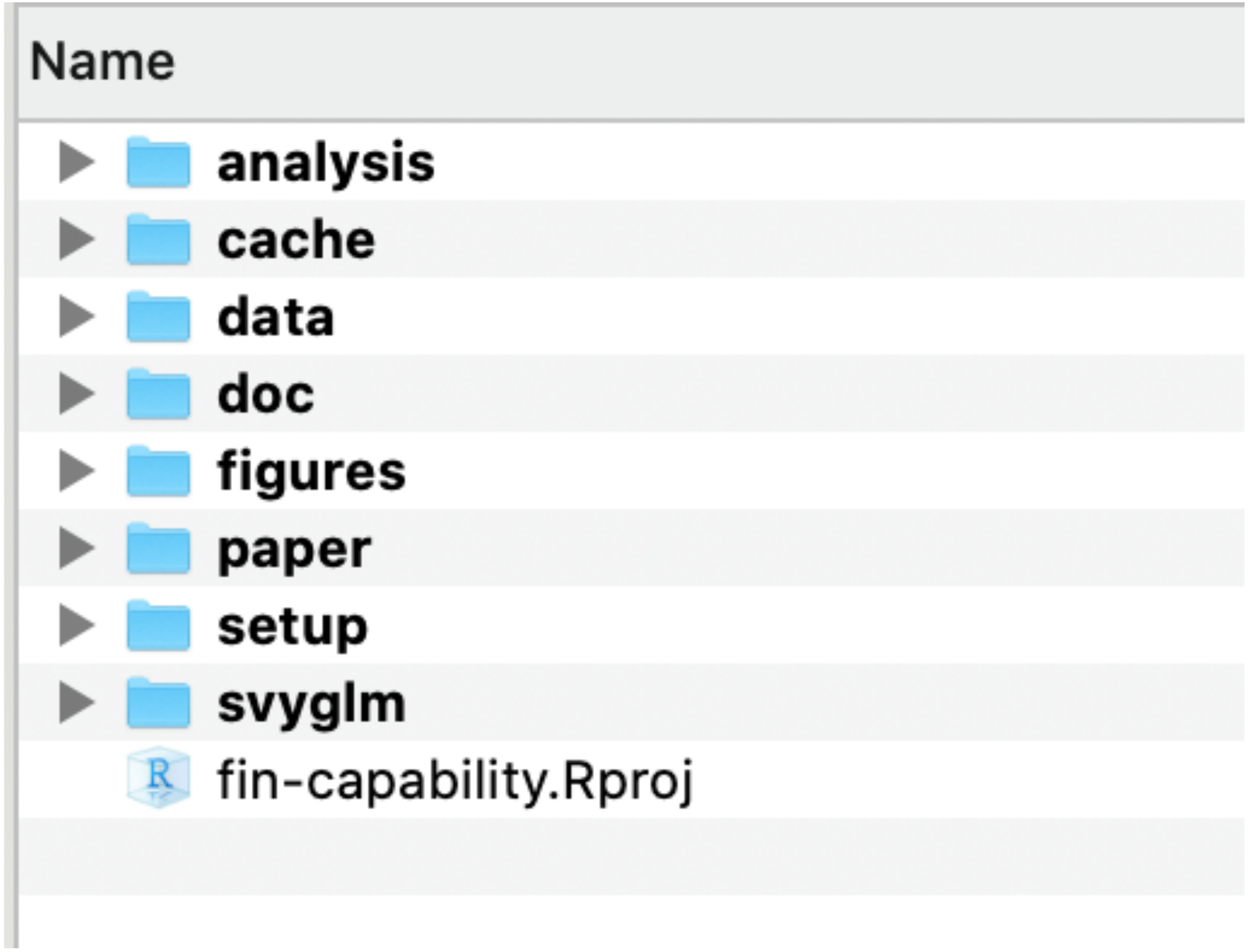
Files | **Plots** | **Packages** | **Help** | **Viewer**

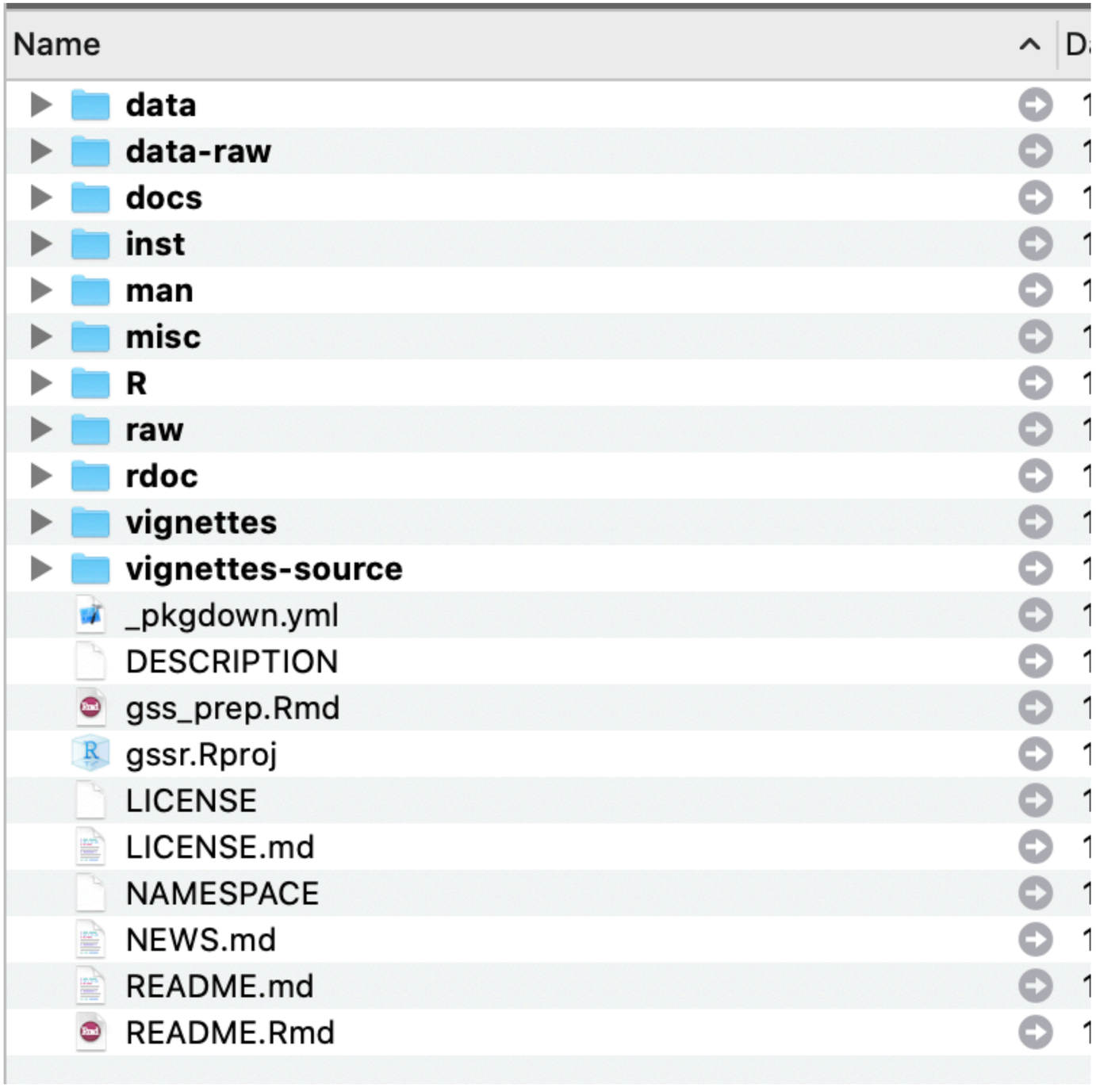
Zoom | Export | Publish



See results in the Plots tab.


scots-turnout
Search
library(reshape2)
library(ggplot2)
library(MASS)
library(kjhutils)
data <- read.csv("data,
data$Name <- paste(dat
data
figures
turnout.rmd
scots-
turnout.Rproj


Name
analysis
cache
data
doc
figures
paper
setup
svyglm
fin-capability.Rproj

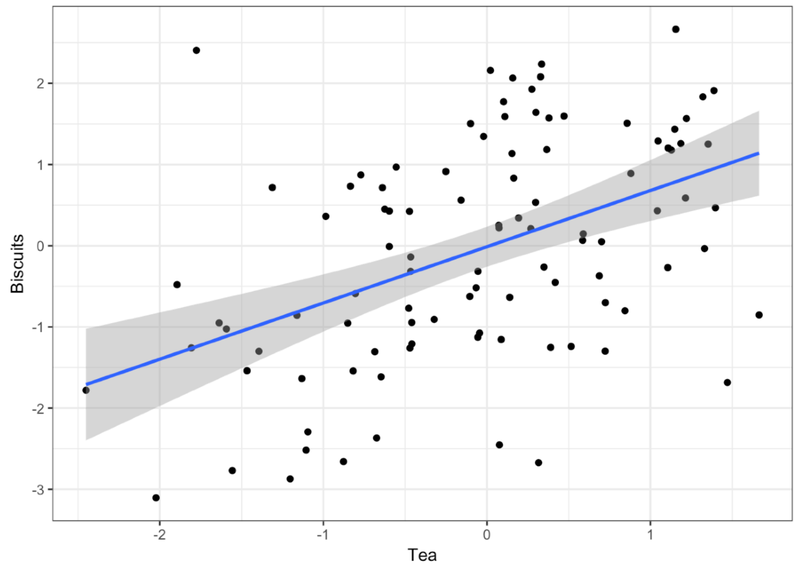

Name
data
data-raw
docs
inst
man
misc
R
raw
rdoc
vignettes
vignettes-source
_pkgdown.yml
DESCRIPTION
gss_prep.Rmd
gssr.Rproj
LICENSE
LICENSE.md
NAMESPACE
NEWS.md
README.md
README.Rmd

Use RMarkdown

TO REPRODUCE YOUR OWN WORK

1. Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do **eiusmod tempor** incididunt ut labore et dolore magna aliqua. Ut enimad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



This is what we want to end up with: nicely formatted text, plots, and tables.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do *eiusmod tempor* incididunt ut labore et dolore magna aliqua. Ut enimad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

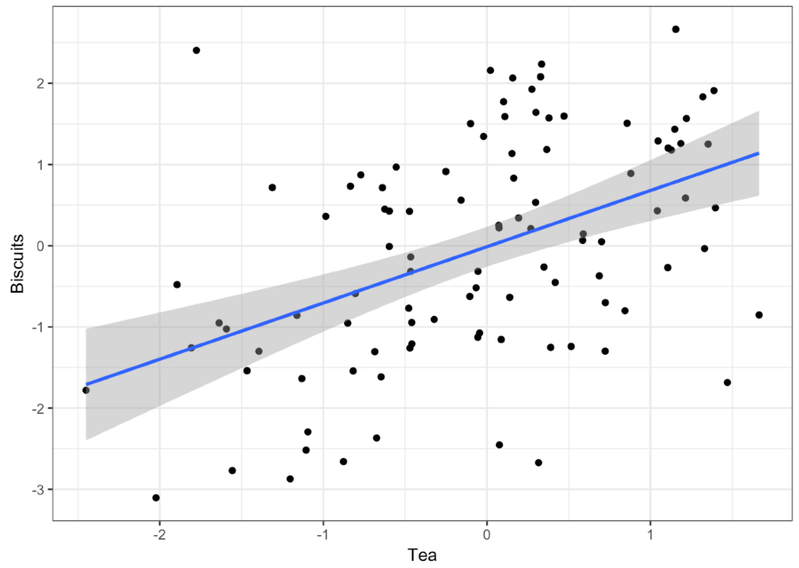
```
library(ggplot2)
tea <- rnorm(100)
biscuits <- tea + rnorm(100, 0, 1.3)
data <- data.frame(tea, biscuits)
p <- ggplot(data, aes(x = tea, y = biscuits)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(x = "Tea", y = "Biscuits") + theme_bw()
print(p)
```

In a **Literate Programming** approach to documents, chunks of code are processed and replaced with their output

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

1. Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do **eiusmod tempor** incididunt ut labore et dolore magna aliqua. Ut enimad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



In a **Literate Programming** approach to documents, chunks of code are processed and replaced with their output

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Report

notes.Rmd

We can see this *relationship* in a scatterplot.

```{r my-code}
p <- ggplot(data, mapping)
p + geom_point()
```

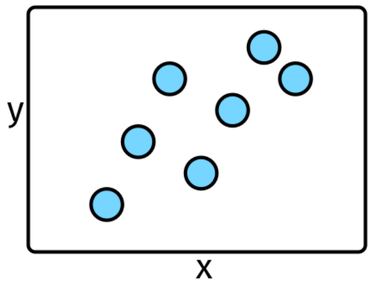
As you can see, this plot looks pretty nice.

knit in R

Report

notes.pdf

We can see this *relationship* in a scatterplot.



As you can see, this plot looks pretty nice.

An Rmd document lets you keep your code and notes together in plain text

And produce good-looking output in a range of formats

notes.Rmd

```
# Report

We can see this *relationship*
in a scatterplot.

```{r my-code}

  p <- ggplot(data, mapping)
  p + geom_point()
```

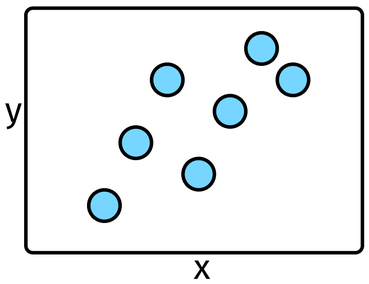
As you can see, this plot
looks pretty nice.
```

knit in R

notes.html

Report

We can see this *relationship*
in a scatterplot.



As you can see, this plot
looks pretty nice.

An Rmd document lets you keep your code and notes together in plain text

And produce good-looking output in a range of formats

notes.Rmd

```
# Report

We can see this *relationship*
in a scatterplot.

```{r my-code}

  p <- ggplot(data, mapping)
  p + geom_point()
```

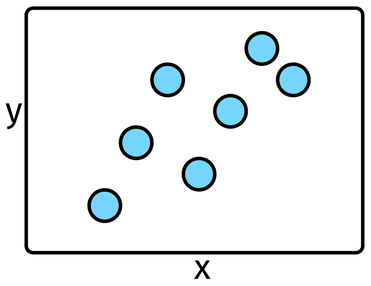
As you can see, this plot
looks pretty nice.
```

knit in R

notes.docx

Report

We can see this *relationship*
in a scatterplot.



As you can see, this plot
looks pretty nice.

An Rmd document lets you keep your code and notes together in plain text

And produce good-looking output in a range of formats

Markdown	Output
`# Header`	**Header**
`## Subhead`	**Subhead**
`Plain text`	Plain text
`*italics*`	*italics*
`**bold**`	**bold**
`` `verbatim` ``	`verbatim`
`1. List`	1. List
`2. List`	2. List
`- Bullet 1`	° Bullet 1
`- Bullet 2`	° Bullet 2
`Footnote.[^1]`	Footnote.[1]
`[^1]: The footnote.`	[1]The footnote.

A **Markdown Processor** turns the marked-up plain text into actually formatted output in HTML, PDF, DOCX or other file types.

Markdown puts formatting instructions in plain-text documents

```
---
title: "My Notes"
author: "Kieran healy"
date: "12/7/2016"
output: html_document
---
```

Header section provides metadata and sets options

```
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
```

Code chunk

```
## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for
authoring HTML, PDF, and MS Word documents. For more details on using R Markdown
see <http://rmarkdown.rstudio.com>.
```

Text with Markdown formatting

```
When you click the **Knit** button a document will be generated that includes
both content as well as the output of any embedded R code chunks within the
document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```

Including Plots

You can also embed plots, for example:

```{r pressure, echo=FALSE}
plot(pressure)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent
printing of the R code that generated the plot.
```

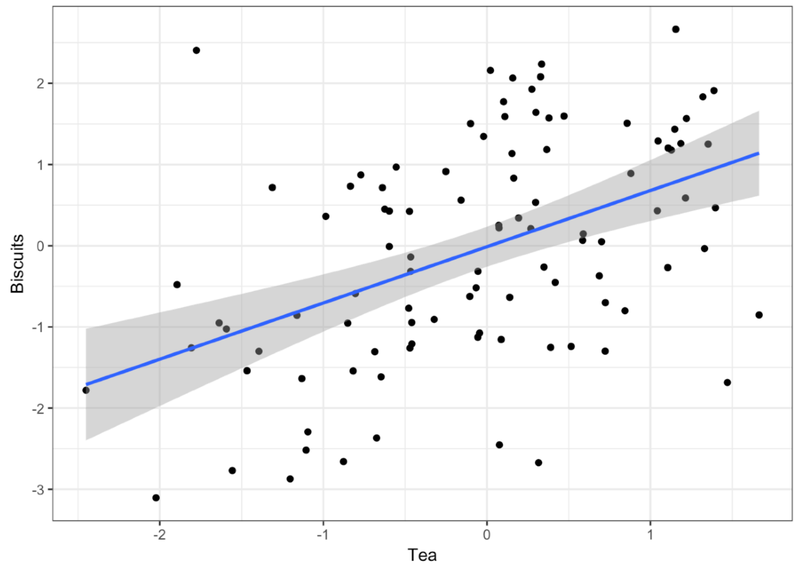
In RStudio, code chunks can be "played" one at a time

Code chunks can have their own names and options

Chunks are replaced by their output when the document is made

1. Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do **eiusmod tempor** incididunt ut labore et dolore magna aliqua. Ut enimad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



RStudio will do all the work for you when it comes to processing your document—i.e., getting it from plain-text Rmd to HTML, Word, or PDF.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

GETTING ORIENTED

The Tidyverse

```
library(tidyverse)
```

```
Loading tidyverse: ggplot2
Loading tidyverse: tibble
Loading tidyverse: tidyr
Loading tidyverse: readr
Loading tidyverse: purrr
Loading tidyverse: dplyr
```

◀ Draw graphs

◀ Nicer data tables

◀ Tidy your data

◀ Get data into R

◀ Cool functional programming stuff

◀ Action verbs for manipulating data

Course-Specific Library

```
library(socviz)
```

CODE YOU CAN TYPE AND RUN

```
## Inside chunks of code, lines beginning with
## the hash character are comments
my_numbers <- c(1, 1, 4, 1, 1, 4, 1)
```

OUTPUT

```
my_numbers
```

```
## [1] 1 1 4 1 1 4 1
```

What R Looks Like

FOUR THINGS TO KNOW ABOUT R

1: Everything has a Name

```
my_numbers
data
p
```

Some names are forbidden

```
FALSE  TRUE  Inf

for  if  break

function
```

2. Everything is an Object

```
letters
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
[20] "t" "u" "v" "w" "x" "y" "z"
```

You create objects by assigning a thing to a name


```
my_numbers <- c(1, 2, 3, 1, 3, 5, 25)
```

You create objects by assigning a thing to a name

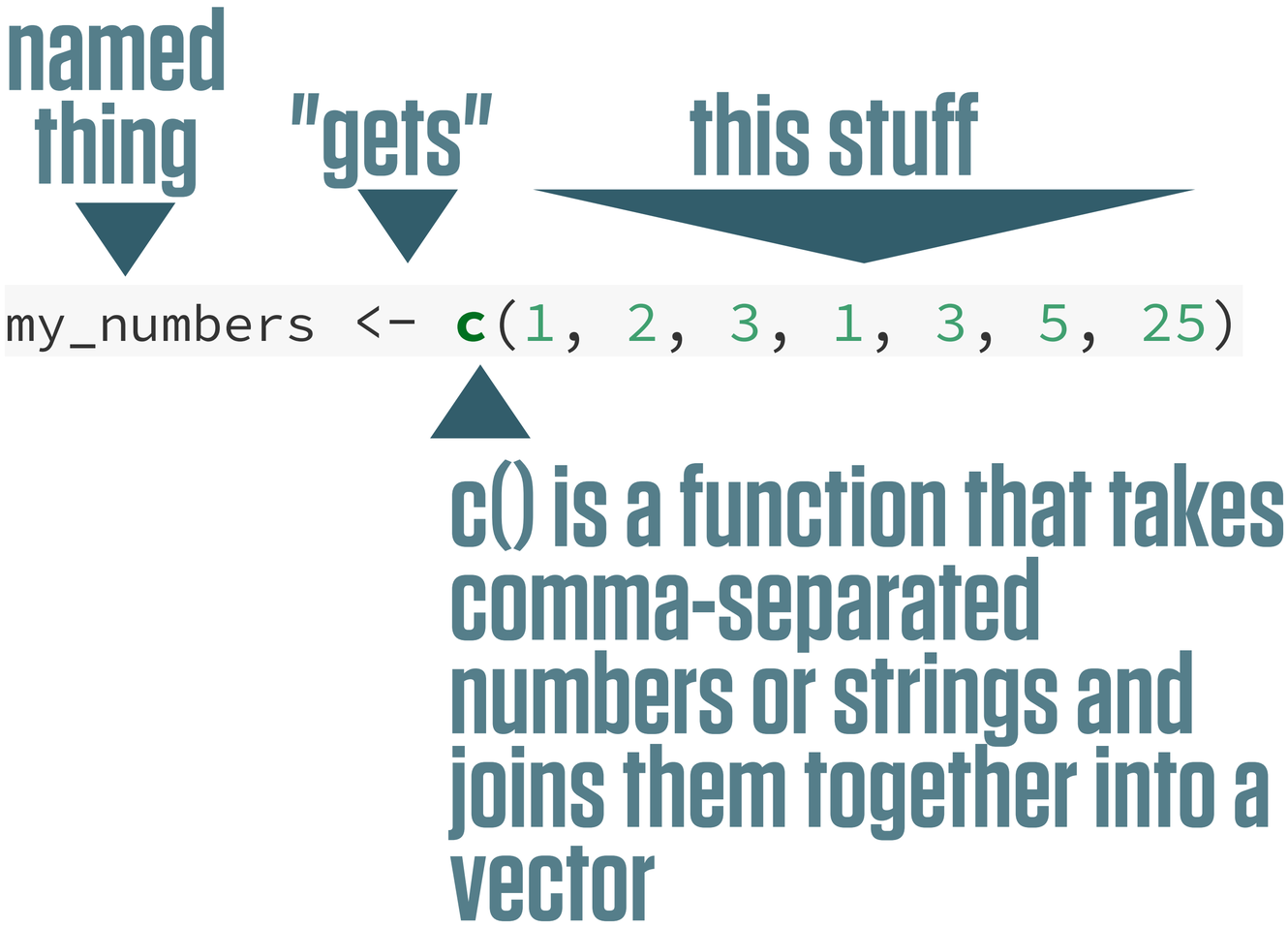
```
my_numbers <- c(1, 2, 3, 1, 3, 5, 25)
```

The assignment operator performs the action of creating objects. Use the keyboard shortcut to type it:

`option -` Mac

`alt -` Windows

3. You do things using functions and operators



Functions

take arguments,
perform actions,
produce outputs

Functions have parentheses
at the end of their name.
This is where the inputs,
or arguments go.

```
mean()
```

“Take this object …”

```
mean(x = my_numbers)
```

Named argument. “Calculate the mean of what, please?”
These names are internal to functions.

Functions

take arguments,
perform actions,
produce outputs

```
mean(my_numbers)
```

If you just write the name of the input,
R assigns it to the function's arguments
in order. Look at the function's help page
to see the order it expects its arguments.

You can assign a function's output to a named object

```
my_summary <- summary(my_numbers)
```

```
my_sd <- sd(my_numbers)
```

```
my_summary
```

```
my_sd
```

Objects you create exist until you overwrite or delete them

```
rm(my_numbers)
```

```
my_numbers
```

```
my_numbers <- c(1, 2, 3, 1, 3, 5, 25)
```

Objects are of different classes

```
class(my_numbers)
```

Vectors

`numeric`

`character`

`factor`

Arrays

`matrix`

`data.frame`

`tibble`

Models

`lm`

`glm`

Things to try on Objects

```
class(my_numbers)
table(my_numbers)
```

Notice that these are functions

```
x <- c(my_numbers, 5)
y <- c(my_numbers, "hello")
```

How do x and y differ?

```
mean(c(my_numbers, my_numbers))
```

Functions can be nested, and will be evaluated from the inside out.

Some operators

`<-` or `=` Assignment ("gets")

`+, -, *, /, ^` Arithmetic

`<, >, <=, >=, ==, !=` Relational

`&, &&, |, ||, !` Logical

`%*%, %in%, %>%` Special

The pipe operator %>% "and then"

```r
mean(my_numbers)
```

```r
my_numbers %>% mean()
```

```r
round(mean(my_numbers))
```

```r
my_numbers %>% mean() %>% round()
```

This will be very convenient later on

R will be Frustrating

We're going to be adding a lot of objects together.

```
ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy)) +
    geom_point()
```

◀ "+" goes here

```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy))
+ geom_point()
```

▲ not here

LET'S GO

```
library(gapminder)
gapminder
```

```
# A tibble: 1,704 x 6
       country continent  year lifeExp      pop gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
 1 Afghanistan      Asia  1952  28.801  8425333  779.4453
 2 Afghanistan      Asia  1957  30.332  9240934  820.8530
 3 Afghanistan      Asia  1962  31.997 10267083  853.1007
 4 Afghanistan      Asia  1967  34.020 11537966  836.1971
 5 Afghanistan      Asia  1972  36.088 13079460  739.9811
 6 Afghanistan      Asia  1977  38.438 14880372  786.1134
 7 Afghanistan      Asia  1982  39.854 12881816  978.0114
 8 Afghanistan      Asia  1987  40.822 13867957  852.3959
 9 Afghanistan      Asia  1992  41.674 16317921  649.3414
10 Afghanistan      Asia  1997  41.763 22227415  635.3414
# ... with 1,694 more rows
```

Named thing gets ...

... using these arguments

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))
```

... the output of this function ...

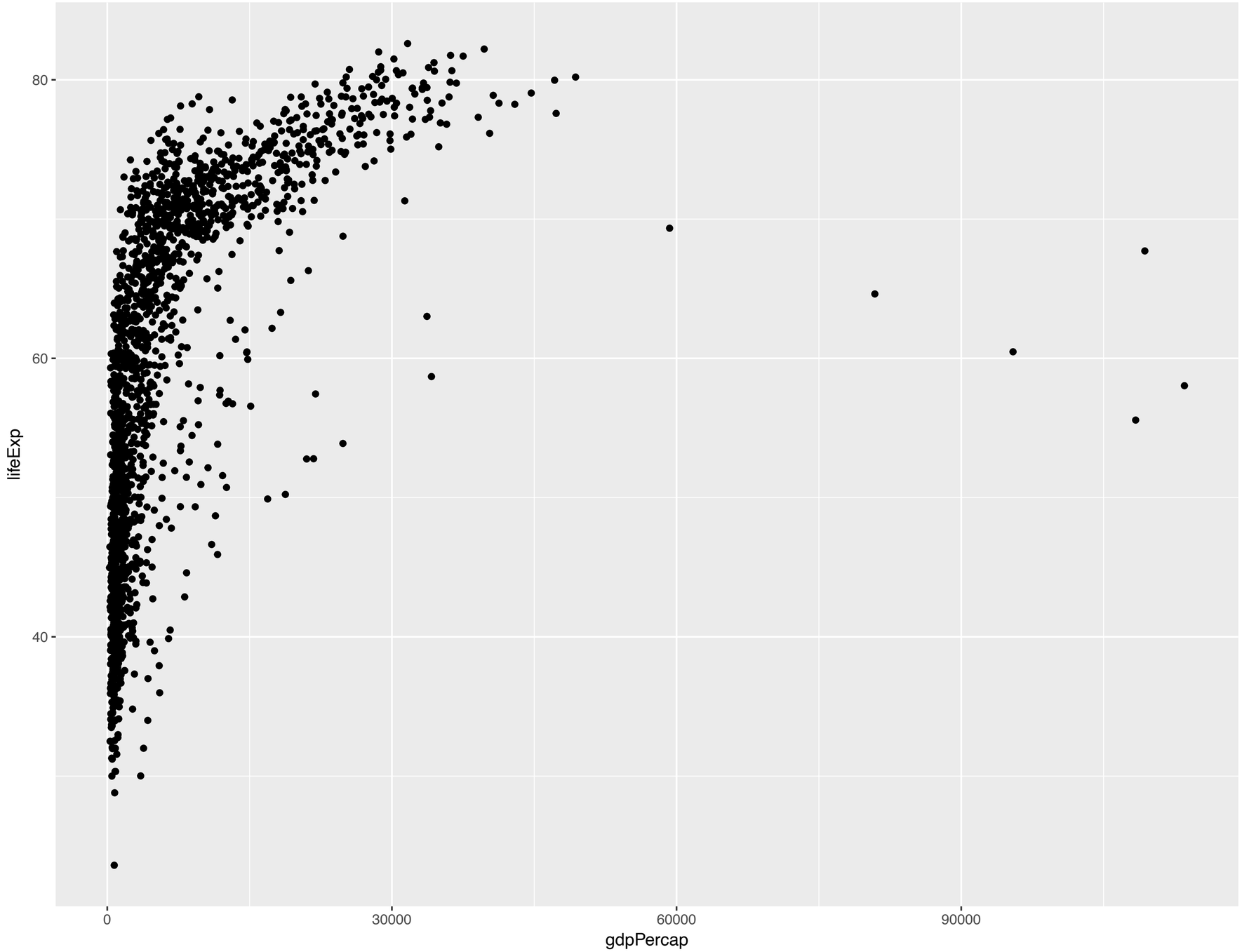
```
p
```

```
p + geom_point()
```

Objects created by ggplot() are unusual in that you can "add" things to them, and they will work as though you wrote all the code at once.

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))

p + geom_point()
```


80
60
40
lifeExp
0
30000
60000
90000
gdpPercap