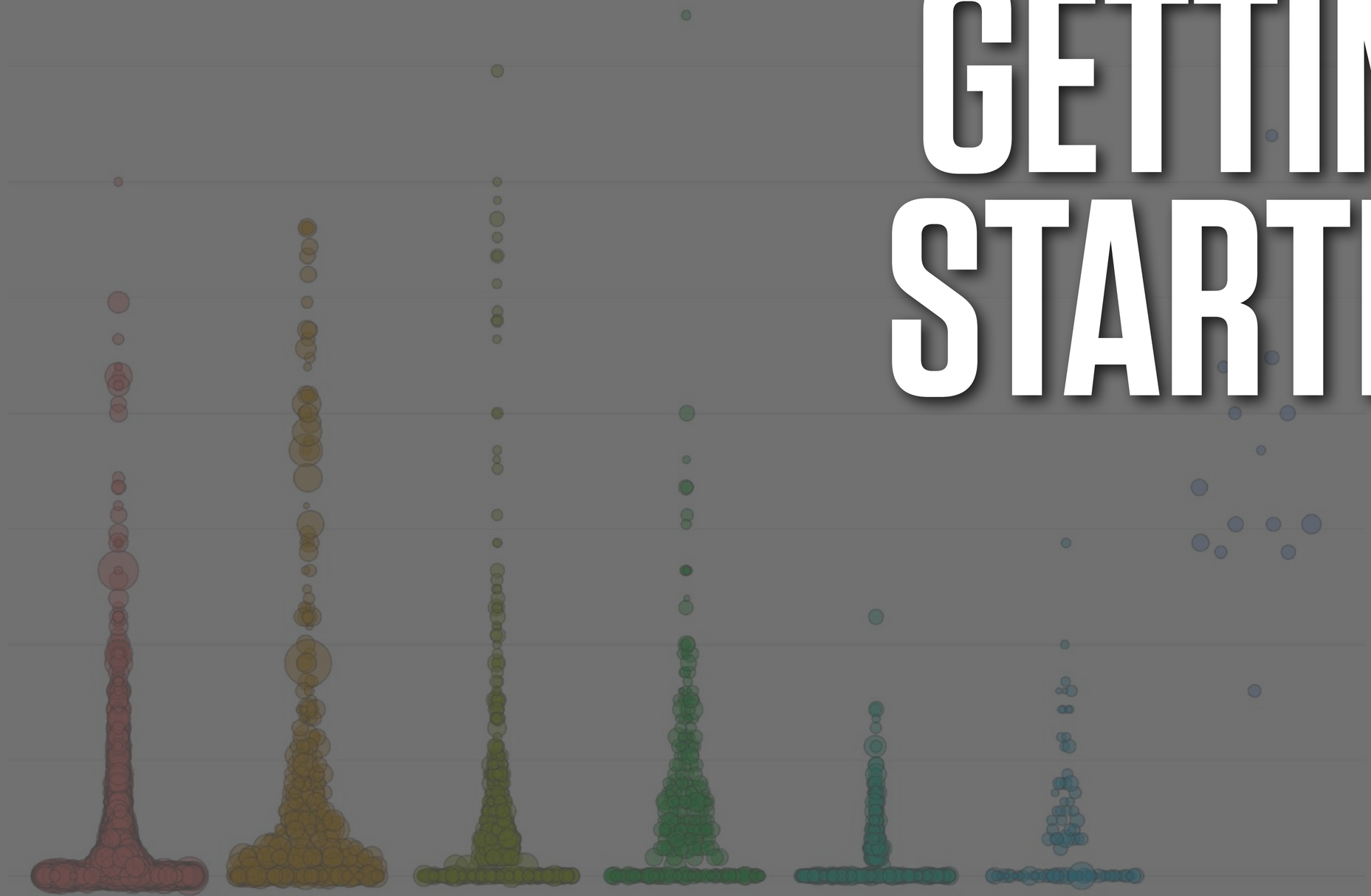
# GETTING STARTED



# 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

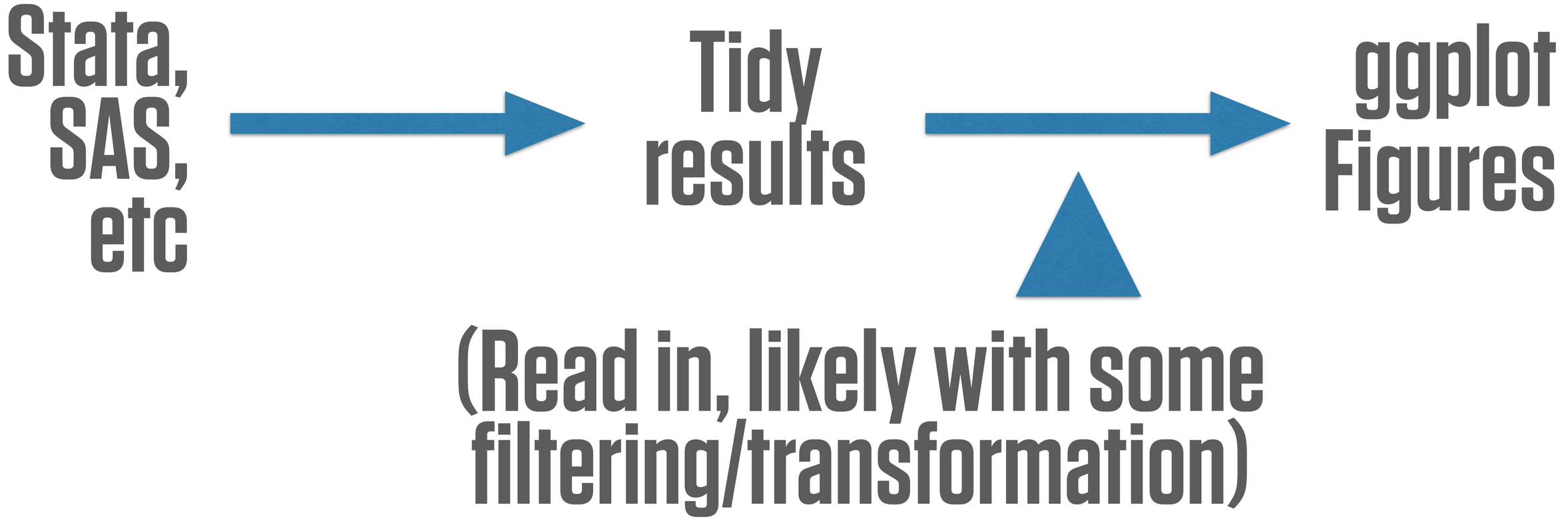
Stata

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

~/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.

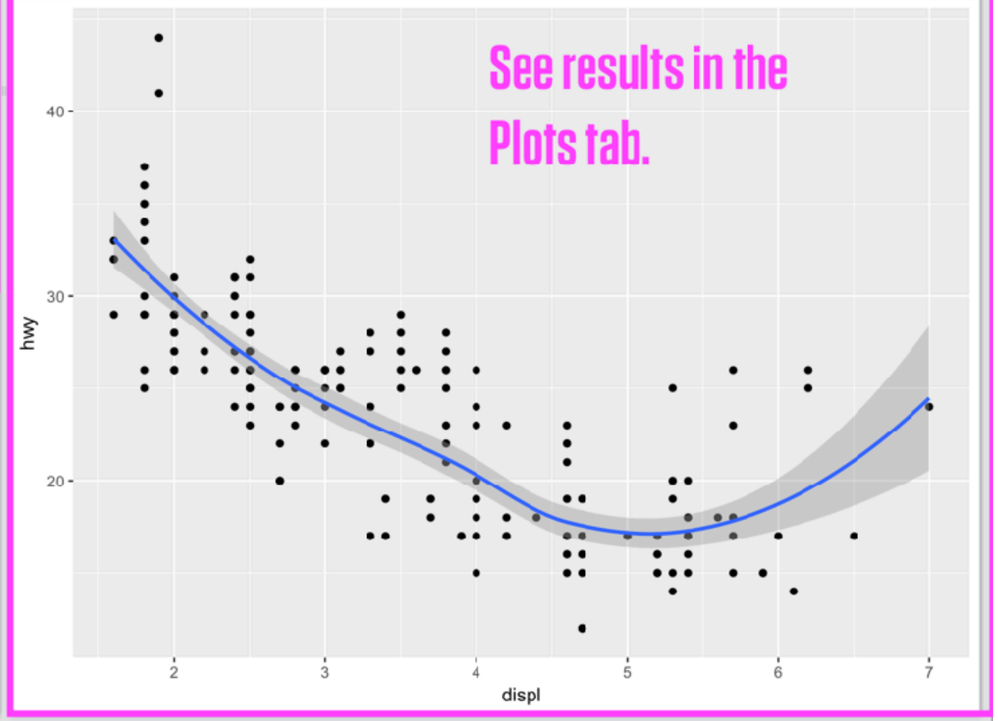
Environment | History | Build

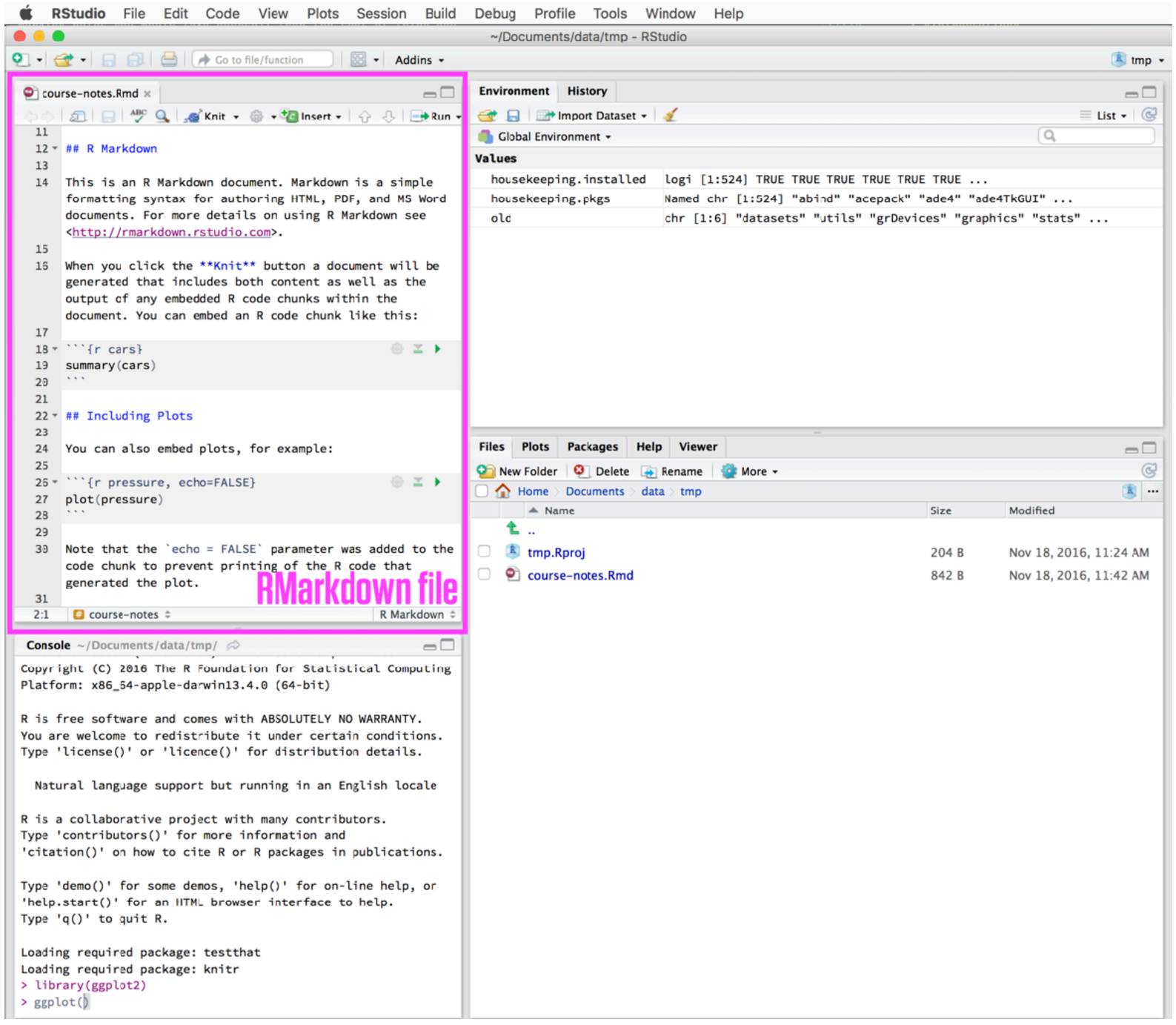
Build Book | More

Files | Plots | Packages | Help | Viewer

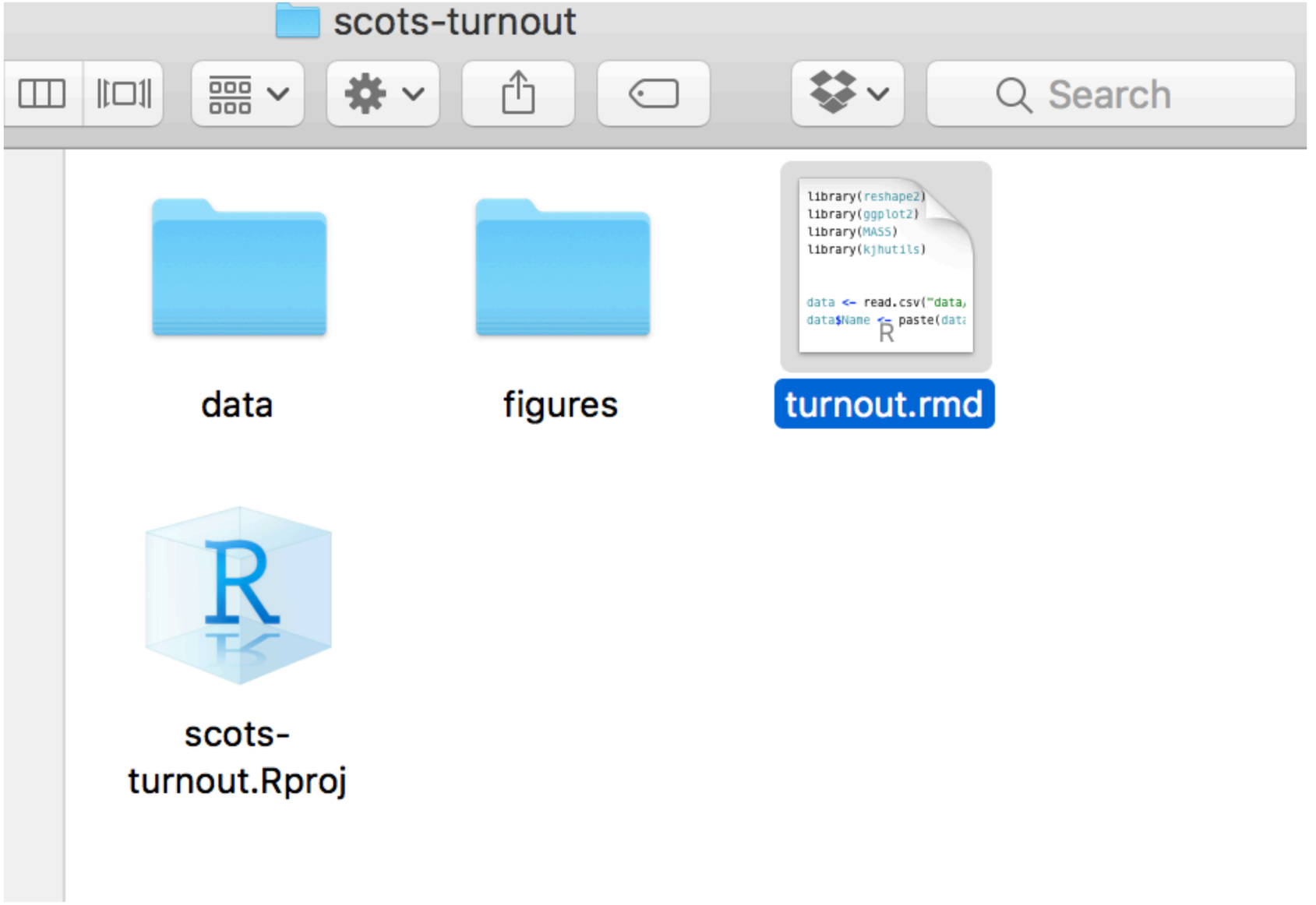
Zoom | Export | Publish

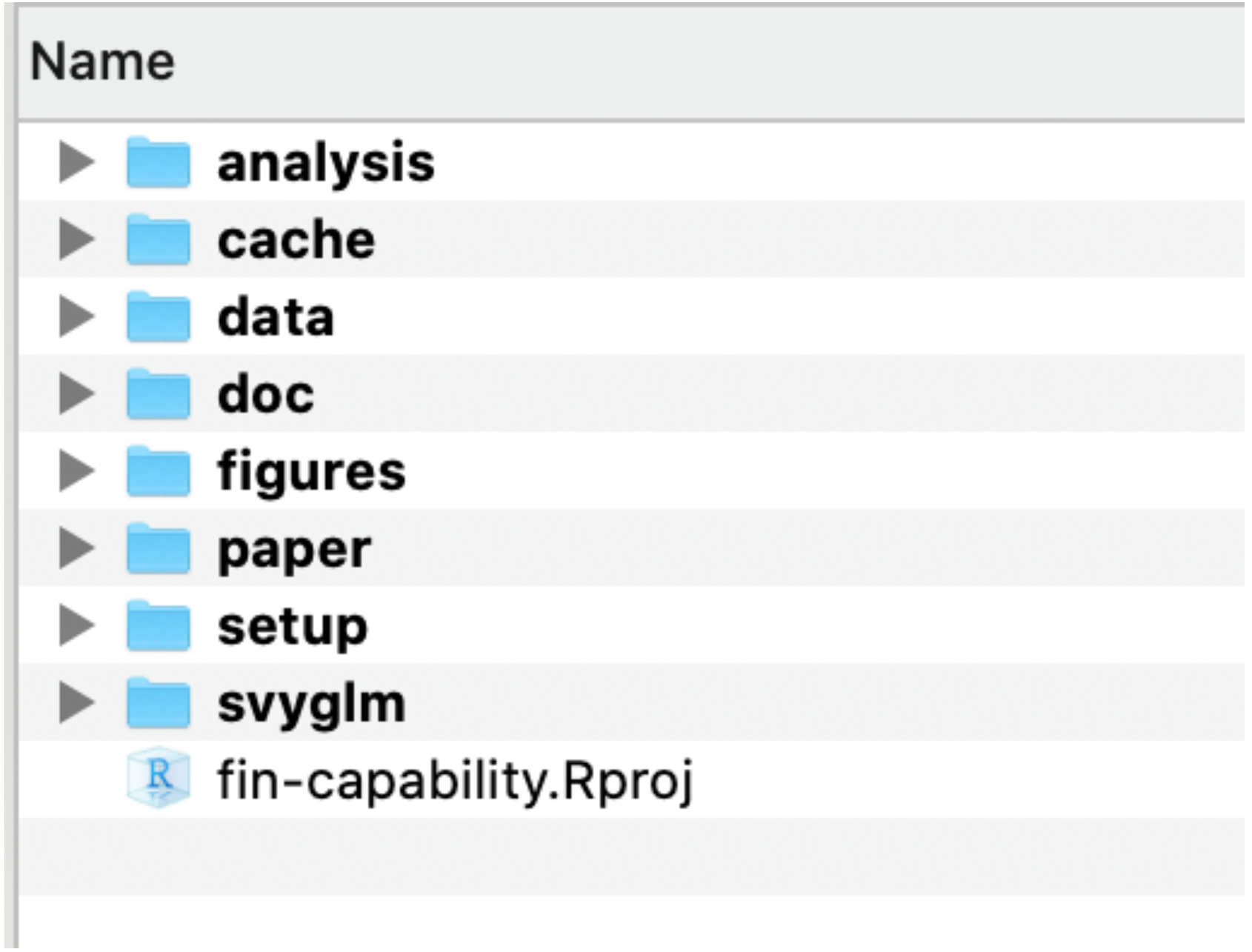
See results in the Plots tab.

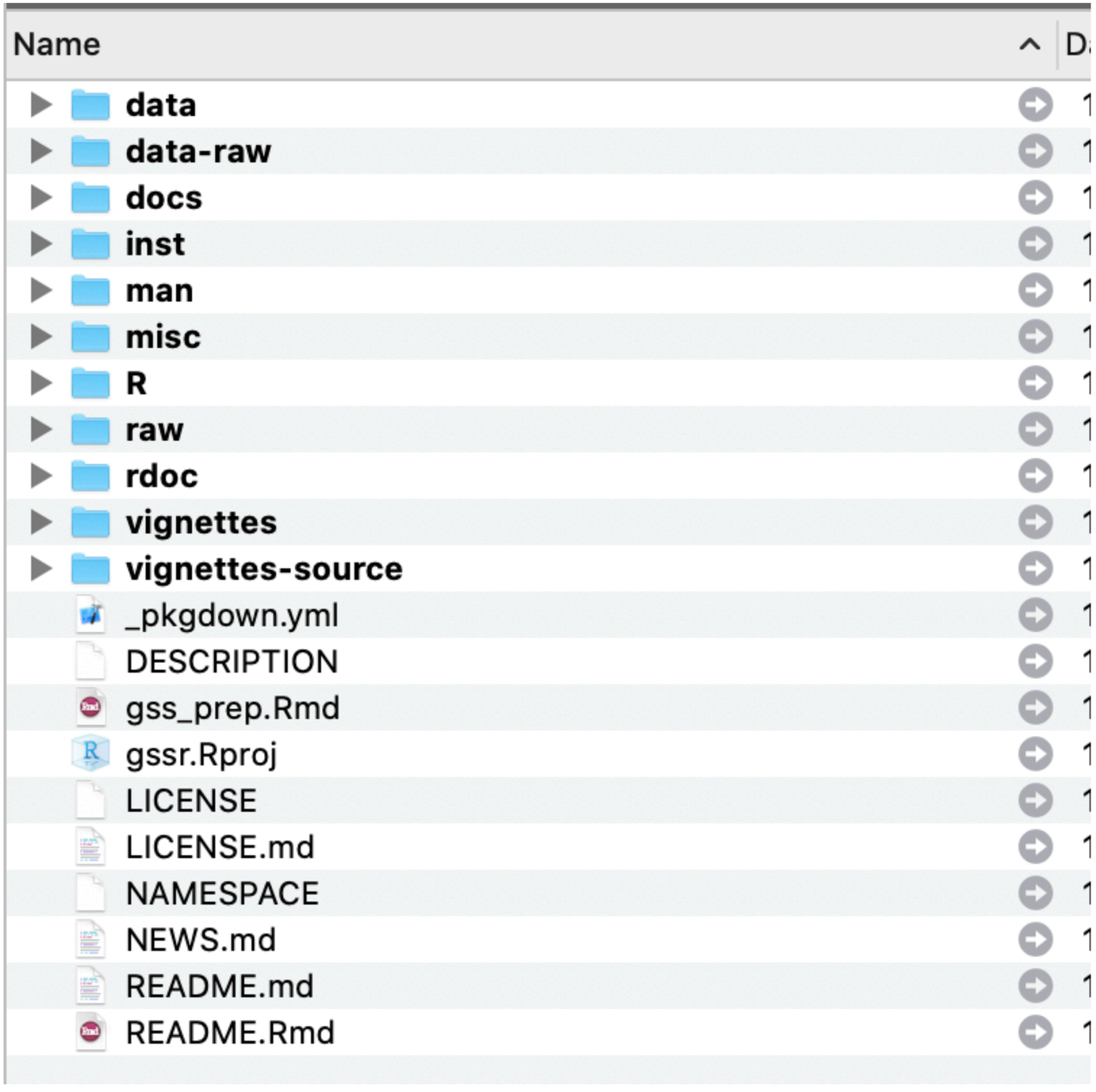



RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help
~/Documents/data/tmp - RStudio
Go to file/function
Addins
tmp
course-notes.Rmd
Knit
Insert
Run
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and 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

# ORGANIZE YOUR PROJECTS


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

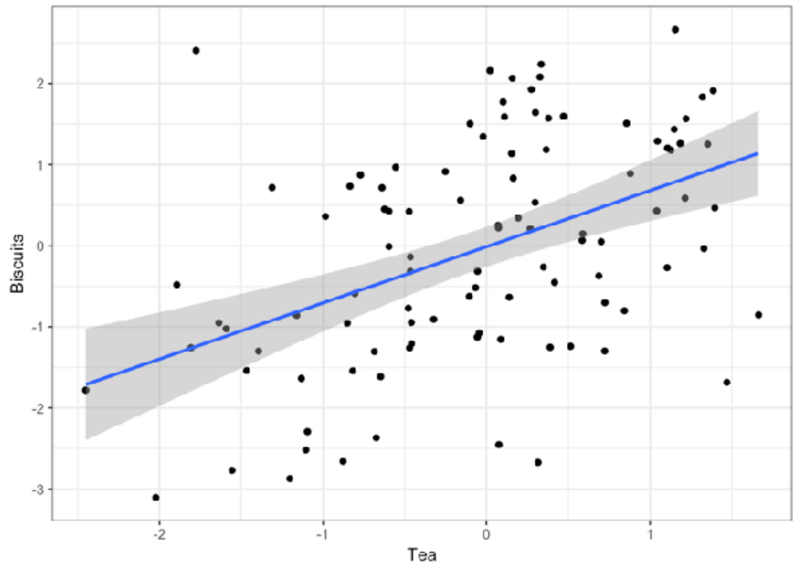

Name
data
data-raw
docs
inst
man
misc
R
raw
rdoc
vignettes
vignettes-source
_pkgdown.yml
DESCRIPTION
gss_prep.Rmd
gssr.Rproj
LICENSE
LICENSE.md
NAMESPACE
NEWS.md
README.md
README.Rmd

# 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.

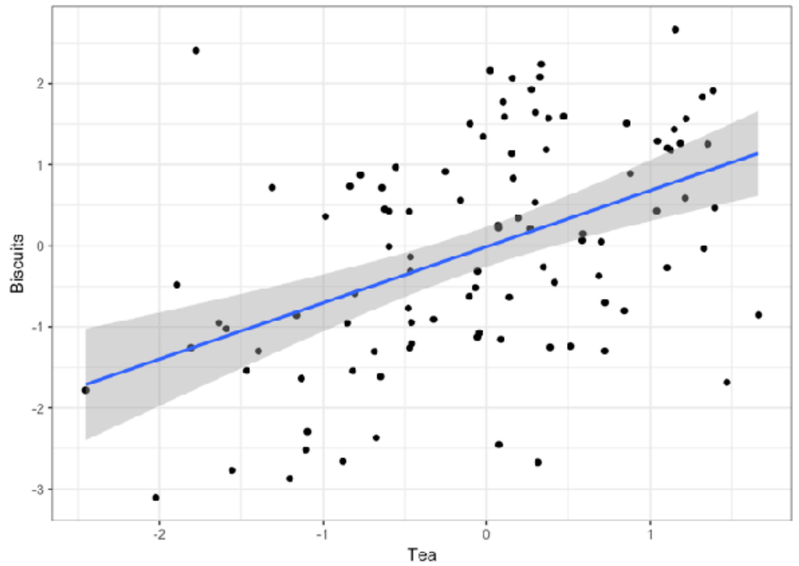
```
library(ggplot2)
tea <- rnorm(100)
biscuits <- tea + rnorm(100, 0, 1.3)
data <- data.frame(tea, biscuits)
p <- ggplot(data, aes(x = tea, y = biscuits)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(x = "Tea", y = "Biscuits") + theme_bw()
print(p)
```

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.

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

# 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.



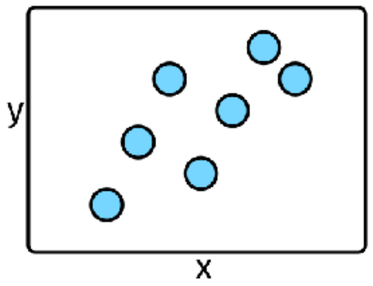

In a **Literate Programming** approach to documents, chunks of code are processed and replaced with their output

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.

notes.Rmd

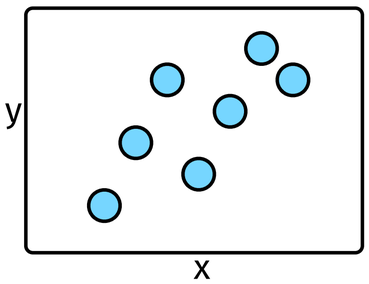
```
# Report
We can see this *relationship*
in a scatterplot.

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

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

knit in R

notes.pdf

**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

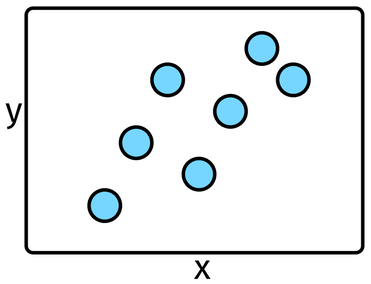
We can see this *relationship*
in a scatterplot.

```{r my-code}

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

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

knit in R

notes.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

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

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.

# 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.
```

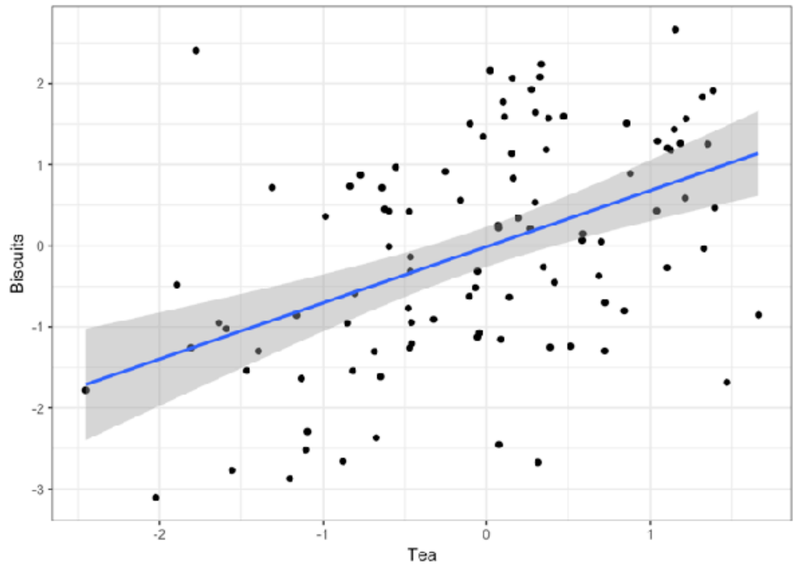
In RStudio, code chunks can be "played" one at a time

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

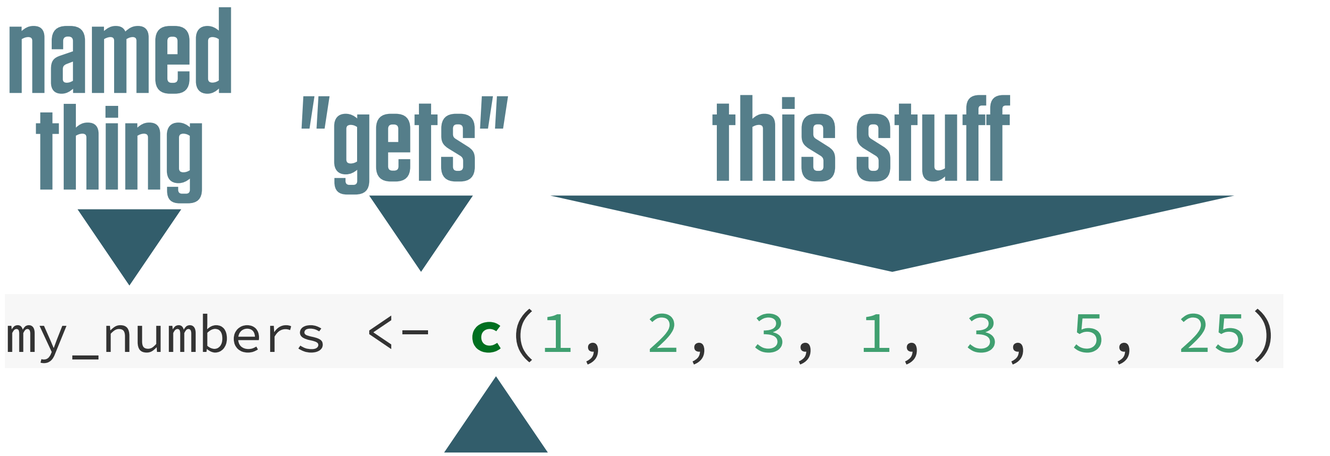
```
my_numbers <- c(1, 2, 3, 1, 3, 5, 25)
```

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 inputs, perform actions, produce outputs

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

```
mean()
```

"Input is this object. Calculate the mean of it."

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

Named argument.
These names are internal to functions.

# Functions

## take inputs, 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 the order given.

# 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))
```

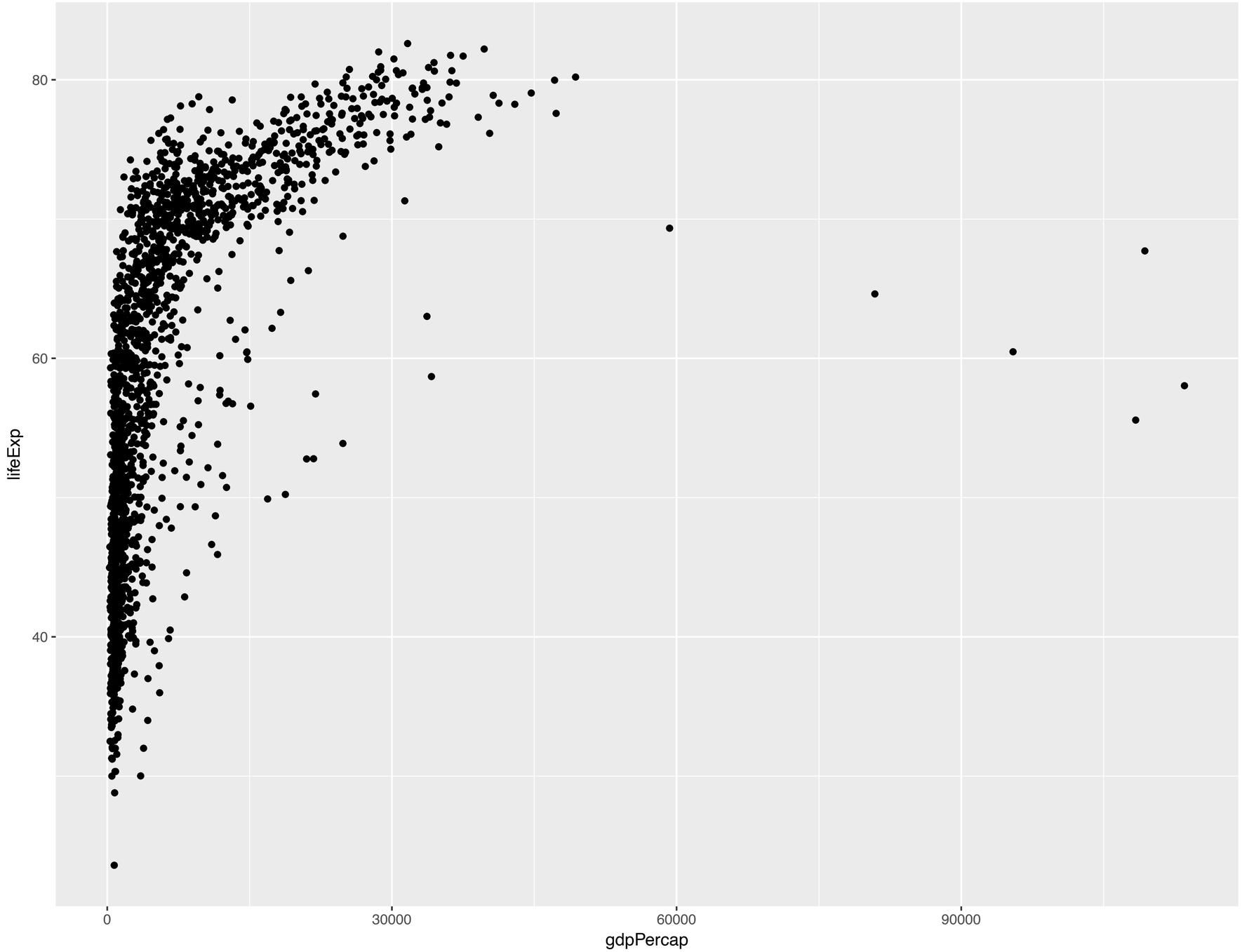
... the output of this function ...

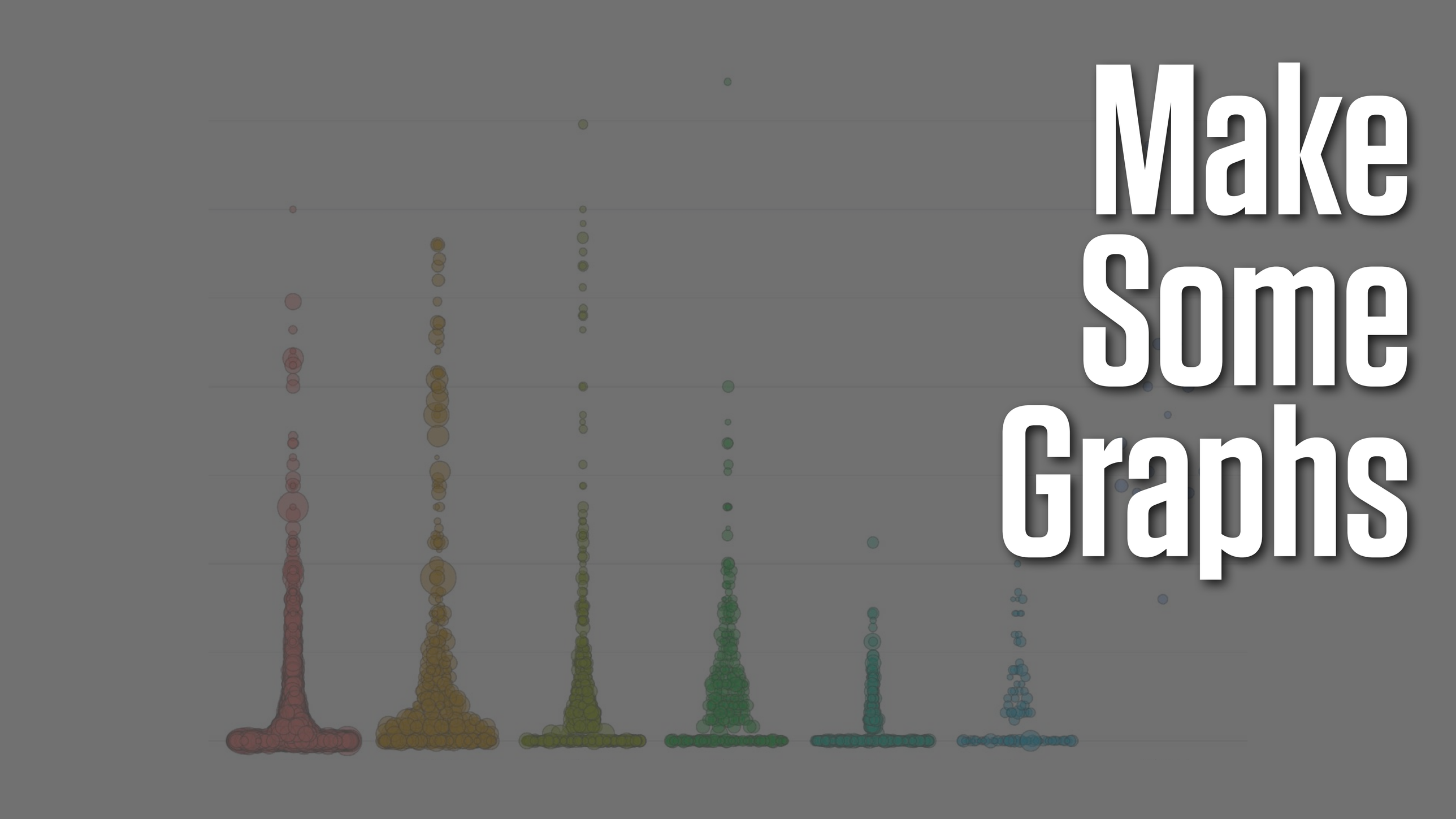
```
p
```

```
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.

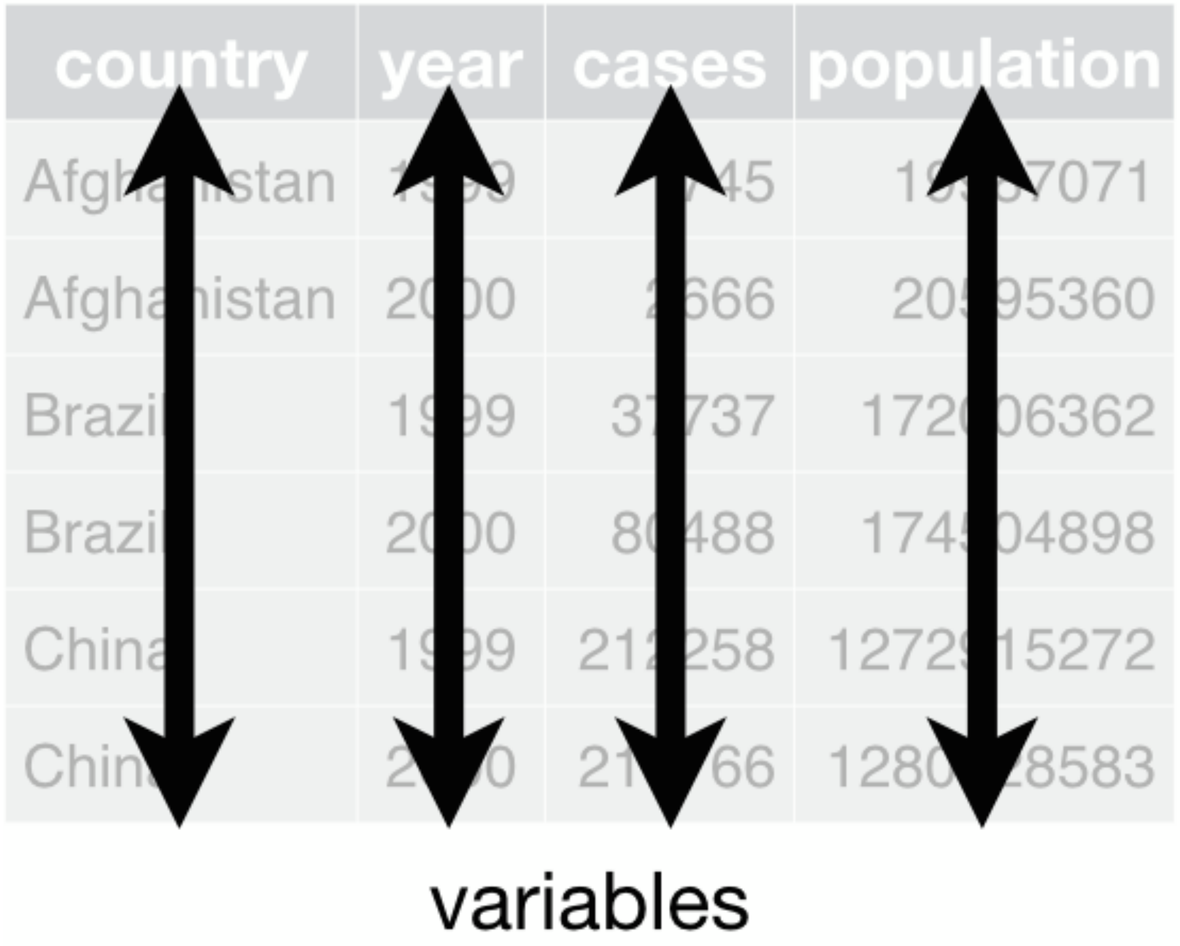
```r
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))

p + geom_point()
```


80
60
40
lifeExp
0
30000
60000
90000
gdpPercap


Make
Some
Graphs

# ggplot wants you to feed it TIDY DATA

| gdp | lifexp | pop | continent |
|---|---|---|---|
| 340 | 65 | 31 | Euro |
| 227 | 51 | 200 | Amer |
| 909 | 81 | 80 | Euro |
| 126 | 40 | 20 | Asia |


country
year
cases
population
variables


observations

| country | year | cases | population |
| --- | --- | --- | --- |
| Afghanistan | 1999 | 745 | 19987071 |
| Afghanistan | 2000 | 2666 | 20595360 |
| Brazil | 1999 | 37737 | 172006362 |
| Brazil | 2000 | 80488 | 174504898 |
| China | 1999 | 212258 | 1272915272 |
| China | 2000 | 213766 | 1280428583 |

values

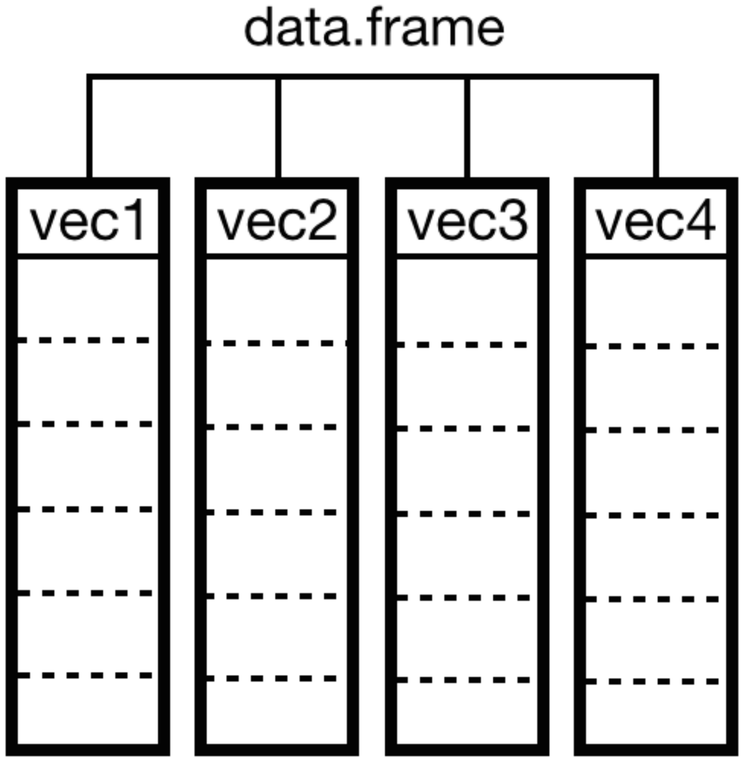



table1

| country | year | cases | pop |
|---|---|---|---|
| Afghan | 1999 | 745 | 19987071 |
| Afghan | 2000 | 2666 | 20595360 |
| Brazil | 1999 | 37737 | 172006362 |
| Brazil | 2000 | 80488 | 174504898 |
| China | 1999 | 212258 | 1272915272 |
| China | 2000 | 213766 | 1280428583 |

State

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | P | Q |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **State** | **CD#** | **2018 Cook PVI Score** | **2018 Winner** | **Party** | **Dem Votes** | **GOP Votes** | **Other Votes** | **Dem %** | **GOP %** | **Other %** | **Dem Margin** | **2016 Clinton Margin** | **Swing vs. 2016 Prez** | **Raw Votes vs. 2016** | **Final?** |
| **New House Breakdown: 235D, 199R, 1 Not Certified** | | | | **D** | **60,619,428** | **50,896,244** | **1,978,795** | **53.4%** | **44.8%** | **1.7%** | **8.6%** | **2.1%** | **6.5%** | **83.3%** | |
| Compiled by: David Wasserman & Ally Flinn, Cook Political Report. @Redistrict/@CookPolitical. *Italics* denotes freshman, **Bold** denotes party change. | | | | | | | | | | | | | | | |
| Alabama | 1 | R+15 | Bradley Byrne | R | 89,226 | 153,228 | 163 | 36.8% | 63.2% | 0.1% | -26.4% | -29.2% | 2.8% | 79.3% | x |
| Alabama | 2 | R+16 | Martha Roby | R | 86,931 | 138,879 | 420 | 38.4% | 61.4% | 0.2% | -23.0% | -31.7% | 8.7% | 78.7% | x |
| Alabama | 3 | R+16 | Mike Rogers | R | 83,996 | 147,770 | 149 | 36.2% | 63.7% | 0.1% | -27.5% | -33.0% | 5.5% | 79.6% | x |
| Alabama | 4 | R+30 | Robert Aderholt | R | 46,492 | 184,255 | 222 | 20.1% | 79.8% | 0.1% | -59.6% | -62.5% | 2.9% | 78.9% | x |
| Alabama | 5 | R+18 | Mo Brooks | R | 101,388 | 159,063 | 222 | 38.9% | 61.0% | 0.1% | -22.1% | -32.9% | 10.8% | 82.8% | x |
| Alabama | 6 | R+26 | Gary Palmer | R | 85,644 | 192,542 | 142 | 30.8% | 69.2% | 0.1% | -38.4% | -43.8% | 5.4% | 82.8% | x |
| Alabama | 7 | D+20 | Terri Sewell | D | 185,010 | 0 | 4,153 | 97.8% | 0.0% | 2.2% | 97.8% | 41.2% | N/A | 64.2% | x |
| Alaska | AL | R+9 | Don Young | R | 131,199 | 149,779 | 1,188 | 46.5% | 53.1% | 0.4% | -6.6% | -14.7% | 8.1% | 88.6% | x |
| Arizona | 1 | R+2 | Tom O'Halleran | D | 143,240 | 122,784 | 65 | 53.8% | 46.1% | 0.0% | 7.7% | -1.1% | 8.8% | 92.0% | x |
| Arizona | 2 | R+1 | ***Ann Kirkpatrick*** | ***D*** | 161,000 | 133,102 | 50 | 54.7% | 45.2% | 0.0% | 9.5% | 4.8% | 4.7% | 91.5% | x |
| Arizona | 3 | D+13 | Raul Grijalva | D | 114,650 | 64,868 | 0 | 63.9% | 36.1% | 0.0% | 27.7% | 29.5% | -1.8% | 84.8% | x |
| Arizona | 4 | R+21 | Paul Gosar | R | 84,521 | 188,842 | 3,672 | 30.5% | 68.2% | 1.3% | -37.7% | -39.4% | 1.7% | 91.1% | x |
| Arizona | 5 | R+15 | Andy Biggs | R | 127,027 | 186,037 | 0 | 40.6% | 59.4% | 0.0% | -18.8% | -20.5% | 1.7% | 91.7% | x |
| Arizona | 6 | R+9 | David Schweikert | R | 140,559 | 173,140 | 0 | 44.8% | 55.2% | 0.0% | -10.4% | -9.8% | -0.6% | 91.2% | x |
| Arizona | 7 | D+23 | Ruben Gallego | D | 113,044 | 301 | 18,706 | 85.6% | 0.2% | 14.2% | 85.4% | 48.3% | N/A | 79.0% | x |
| Arizona | 8 | R+13 | Debbie Lesko | R | 135,569 | 168,835 | 13 | 44.5% | 55.5% | 0.0% | -10.9% | -20.8% | 9.9% | 91.5% | x |
| Arizona | 9 | D+4 | *Greg Stanton* | *D* | 159,583 | 101,662 | 0 | 61.1% | 38.9% | 0.0% | 22.2% | 15.9% | 6.3% | 90.0% | x |
| Arkansas | 1 | R+17 | Rick Crawford | R | 57,907 | 138,757 | 4,581 | 28.8% | 68.9% | 2.3% | -40.2% | -34.8% | -5.4% | 77.2% | x |
| Arkansas | 2 | R+7 | French Hill | R | 116,135 | 132,125 | 5,193 | 45.8% | 52.1% | 2.0% | -6.3% | -10.7% | 4.4% | 82.6% | x |
| Arkansas | 3 | R+19 | Steve Womack | R | 74,952 | 148,717 | 6,039 | 32.6% | 64.7% | 2.6% | -32.1% | -31.4% | -0.7% | 78.6% | x |
| Arkansas | 4 | R+17 | Bruce Westerman | R | 63,984 | 136,740 | 4,168 | 31.2% | 66.7% | 2.0% | -35.5% | -32.8% | -2.7% | 75.7% | x |
| California | 1 | R+11 | Doug LaMalfa | R | 131,506 | 160,006 | 0 | 45.1% | 54.9% | 0.0% | -9.8% | -19.4% | 9.6% | 91.6% | |
| California | 2 | D+22 | Jared Huffman | D | 243,051 | 72,541 | 0 | 77.0% | 23.0% | 0.0% | 54.0% | 45.2% | 8.8% | 90.5% | |
| California | 3 | D+5 | John Garamendi | D | 132,983 | 96,106 | 0 | 58.0% | 42.0% | 0.0% | 16.1% | 12.5% | 3.6% | 86.8% | |
| California | 4 | R+10 | Tom McClintock | R | 156,253 | 184,401 | 0 | 45.9% | 54.1% | 0.0% | -8.3% | -14.5% | 6.2% | 94.6% | |
| California | 5 | D+21 | Mike Thompson | D | 203,012 | 0 | 53,836 | 79.0% | 0.0% | 21.0% | 79.0% | 44.6% | N/A | 83.8% | |
| California | 6 | D+21 | Doris Matsui | D | 201,939 | 0 | 0 | 100.0% | 0.0% | 0.0% | 100.0% | 44.0% | N/A | 81.4% | |
| California | 7 | D+3 | Ami Bera | D | 155,016 | 126,601 | 0 | 55.0% | 45.0% | 0.0% | 10.1% | 11.2% | -1.1% | 91.0% | |
| California | 8 | R+9 | Paul Cook | R | 0 | 170,785 | 0 | 0.0% | 100.0% | 0.0% | -100.0% | -15.1% | N/A | 73.3% | |
| California | 9 | D+8 | Jerry McNerney | D | 113,240 | 87,263 | 0 | 56.5% | 43.5% | 0.0% | 13.0% | 18.2% | -5.2% | 82.4% | |

| country | year | cases | population |
|---|---|---|---|
| Afghanistan | 1999 | 745 | 19987071 |
| Afghanistan | 2000 | 2666 | 20595360 |
| Brazil | 1999 | 37737 | 172006362 |
| Brazil | 2000 | 80488 | 174504898 |
| China | 1999 | 212258 | 1272915272 |
| China | 2000 | 213766 | 1280428583 |

table1

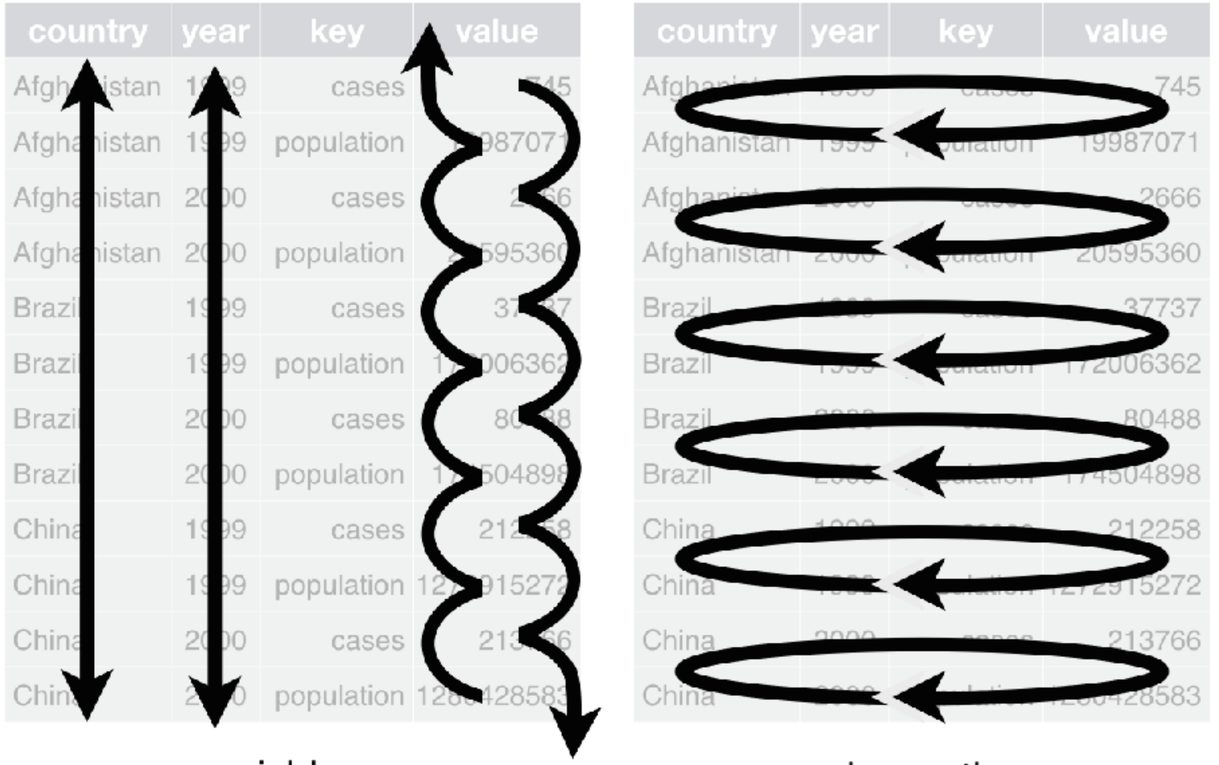
variables

observations

```
      country year  cases population
1 Afghanistan 1999    745   19987071
2 Afghanistan 2000   2666   20595360
3      Brazil 1999  37737  172006362
4      Brazil 2000  80488  174504898
5       China 1999 212258 1272915272
6       China 2000 213766 1280428583
```

| country | year | key | value |
|---|---|---|---|
| Afghanistan | 1999 | cases | 745 |
| Afghanistan | 1999 | population | 19987071 |
| Afghanistan | 2000 | cases | 2666 |
| Afghanistan | 2000 | population | 20595360 |
| Brazil | 1999 | cases | 37737 |
| Brazil | 1999 | population | 172006362 |
| Brazil | 2000 | cases | 80488 |
| Brazil | 2000 | population | 174504898 |
| China | 1999 | cases | 212258 |
| China | 1999 | population | 1272915272 |
| China | 2000 | cases | 213766 |
| China | 2000 | population | 1280428583 |

table2




variables

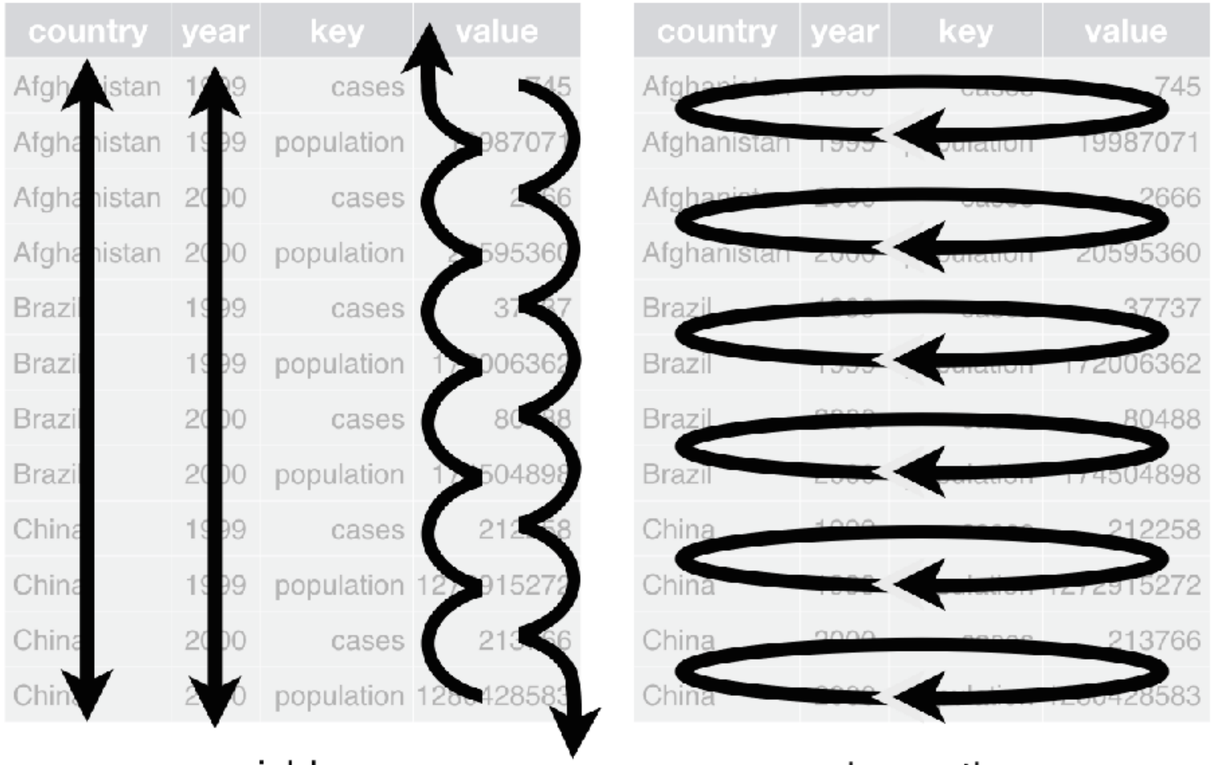



observations

| | country | year | key | value |
|---|---|---|---|---|
| 1 | Afghanistan | 1999 | cases | 745 |
| 2 | Afghanistan | 1999 | population | 19987071 |
| 3 | Afghanistan | 2000 | cases | 2666 |
| 4 | Afghanistan | 2000 | population | 20595360 |
| 5 | Brazil | 1999 | cases | 37737 |
| 6 | Brazil | 1999 | population | 172006362 |
| 7 | Brazil | 2000 | cases | 80488 |
| 8 | Brazil | 2000 | population | 174504898 |
| 9 | China | 1999 | cases | 212258 |
| 10 | China | 1999 | population | 1272915272 |
| 11 | China | 2000 | cases | 213766 |
| 12 | China | 2000 | population | 1280428583 |

| country | year | key | value |
|---|---|---|---|
| Afghanistan | 1999 | cases | 745 |
| Afghanistan | 1999 | population | 19987071 |
| Afghanistan | 2000 | cases | 2666 |
| Afghanistan | 2000 | population | 20595360 |
| Brazil | 1999 | cases | 37737 |
| Brazil | 1999 | population | 172006362 |
| Brazil | 2000 | cases | 80488 |
| Brazil | 2000 | population | 174504898 |
| China | 1999 | cases | 212258 |
| China | 1999 | population | 1272915272 |
| China | 2000 | cases | 213766 |
| China | 2000 | population | 1280428583 |

table2




variables



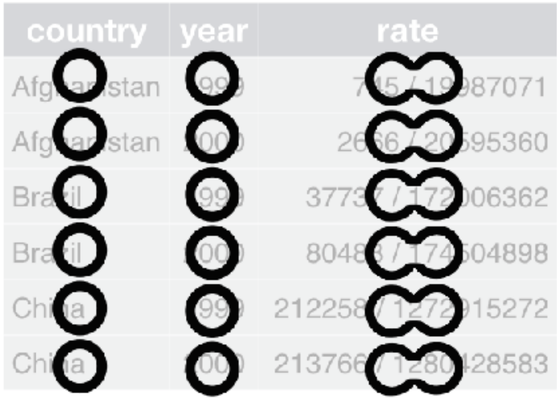

observations

| | country | year | key | value |
|---|---|---|---|---|
| 1 | Afghanistan | 1999 | cases | 745 |
| 2 | Afghanistan | 1999 | population | 19987071 |
| 3 | Afghanistan | 2000 | cases | 2666 |
| 4 | Afghanistan | 2000 | population | 20595360 |
| 5 | Brazil | 1999 | cases | 37737 |
| 6 | Brazil | 1999 | population | 172006362 |
| 7 | Brazil | 2000 | cases | 80488 |
| 8 | Brazil | 2000 | population | 174504898 |
| 9 | China | 1999 | cases | 212258 |
| 10 | China | 1999 | population | 1272915272 |
| 11 | China | 2000 | cases | 213766 |
| 12 | China | 2000 | population | 1280428583 |

| country | year | rate |
|---|---|---|
| Afghanistan | 1999 | 745 / 19987071 |
| Afghanistan | 2000 | 2666 / 20595360 |
| Brazil | 1999 | 37737 / 172006362 |
| Brazil | 2000 | 80488 / 174504898 |
| China | 1999 | 212258 / 1272915272 |
| China | 2000 | 213766 / 1280428583 |

table3


variables




values

```
      country year               rate
1 Afghanistan 1999     745/19987071
2 Afghanistan 2000    2666/20595360
3      Brazil 1999   37737/172006362
4      Brazil 2000   80488/174504898
5       China 1999 212258/1272915272
6       China 2000 213766/1280428583
```

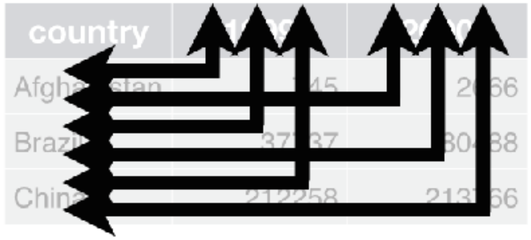
| country | 1999 | 2000 |
|---|---|---|
| Afghanistan | 745 | 2666 |
| Brazil | 37737 | 80488 |
| China | 212258 | 213766 |

table4

| country | 1999 | 2000 |
|---|---|---|
| Afghanistan | 19987071 | 20595360 |
| Brazil | 172006362 | 174504898 |
| China | 1272915272 | 1280428583 |

table5

variables



observations

```
     country    1999    2000
1 Afghanistan     745    2666
2      Brazil   37737   80488
3       China  212258  213766
```

```
     country       1999       2000
1 Afghanistan   19987071   20595360
2      Brazil  172006362  174504898
3       China 1272915272 1280428583
```

# GETTING YOUR DATA INTO R

```
my_data <- read_csv(file = “data/organdonation.csv")
```

Field delimiter is ,

```
read_csv2(file = "data/my_csv_file.csv")
```

Field delimiter is ;

```
read_dta(file = "data/my_stata_file.dta")

read_spss(file = "data/my_spss_file.sav")

read_sas(data_file = "<NAME>",
         catalog_file = "<NAME>")

read_table(file = "<NAME>")
```

Structured but not delimited

# Local File Path

```
organs <- read_csv(file = "data/organdonation.csv")
```

# Remote URL

```
url <- "https://cdn.rawgit.com/kjhealy/viz-
organdata/master/organdonation.csv"

organs <- read_csv(file = url)
```

```
England and Wales, Total Population, Death rates (period 1x1),  Last modified: 02 Apr
2018;  Methods Protocol: v6 (2017)

  Year          Age            Female           Male           Total
  1841            0          0.136067       0.169189        0.152777
  1841            1          0.059577       0.063208        0.061386
  1841            2          0.036406       0.036976        0.036689
  1841            3          0.024913       0.026055        0.025480
  1841            4          0.018457       0.019089        0.018772
  1841            5          0.013967       0.014279        0.014123
  1841            6          0.010870       0.011210        0.011040
  1841            7          0.008591       0.008985        0.008788
  1841            8          0.006860       0.007246        0.007053
  1841            9          0.005772       0.006050        0.005911
  1841           10          0.005303       0.005382        0.005343
  1841           11          0.005114       0.005002        0.005057
  1841           12          0.005145       0.004856        0.004999
  1841           13          0.005455       0.004955        0.005202
  1841          105          0.576967       1.727848        0.700373
  1841          106          0.677711       6.000000        0.795287
  1841          107          0.900000              .        0.900000
  1841          108          1.388430              .        1.388430
  1841          109                 .              .               .
  1841         110+                 .              .               .
  1842            0          0.148491       0.184007        0.166481
  1842            1          0.063038       0.066596        0.064818
  1842            2          0.035203       0.035854        0.035527
```

```
engmort <- read_table(file = "data/mortality.txt",
                      skip = 2, na = ".")
```

# HOW ggplot WORKS

# ggplot's FLOW OF ACTION

## 1. Tidy Data

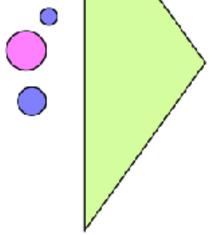
| gdp | lifexp | pop | continent |
|---|---|---|---|
| 340 | 65 | 31 | Euro |
| 227 | 51 | 200 | Amer |
| 909 | 81 | 80 | Euro |
| 126 | 40 | 20 | Asia |

```
ggplot(data = gapminder,
```

## 2. Mapping

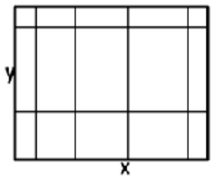

```
mapping =
aes(x = gdp,
y = lifespan,
color = continent,
size = pop))
```

## 3. Geom



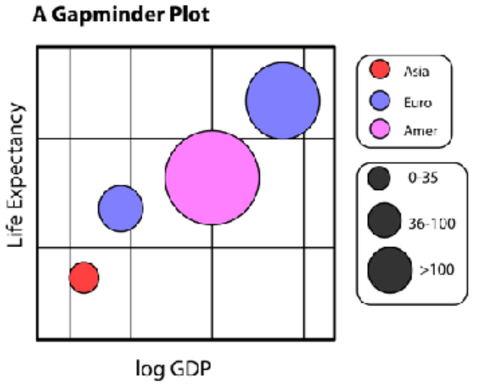
```
geom_point()
```

## 4. Co-Ordinates, Scales



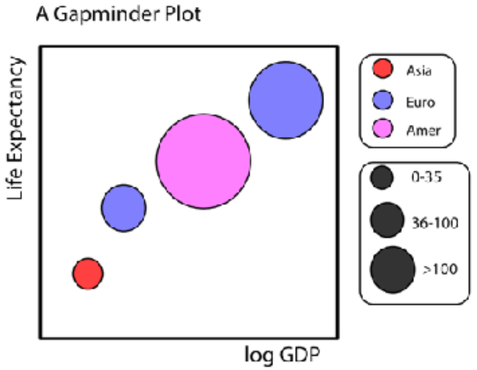

```
coord_cartesian()
scale_x_log10()
```

## 5. Labels & Guides




```
labs()
guides()
```

## 6. Themes



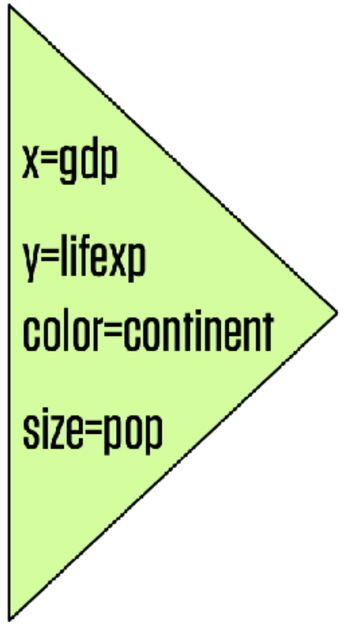

```
theme_minimal()
```

## 1. Tidy Data

| gdp | lifexp | pop | continent |
|---|---|---|---|
| 340 | 65 | 31 | Euro |
| 227 | 51 | 200 | Amer |
| 909 | 81 | 80 | Euro |
| 126 | 40 | 20 | Asia |

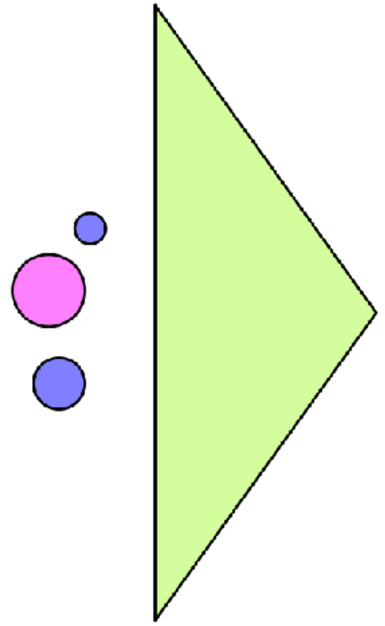
```
ggplot(data = gapminder,
```

## 2. Mapping



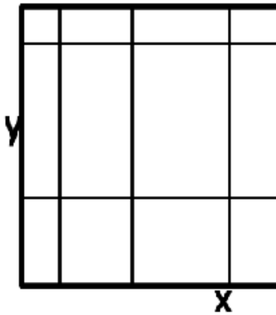

```
mapping =
aes(x = gdp,
y = lifespan,
color = continent,
size = pop))
```

## 3. Geom



```
geom_point()
```

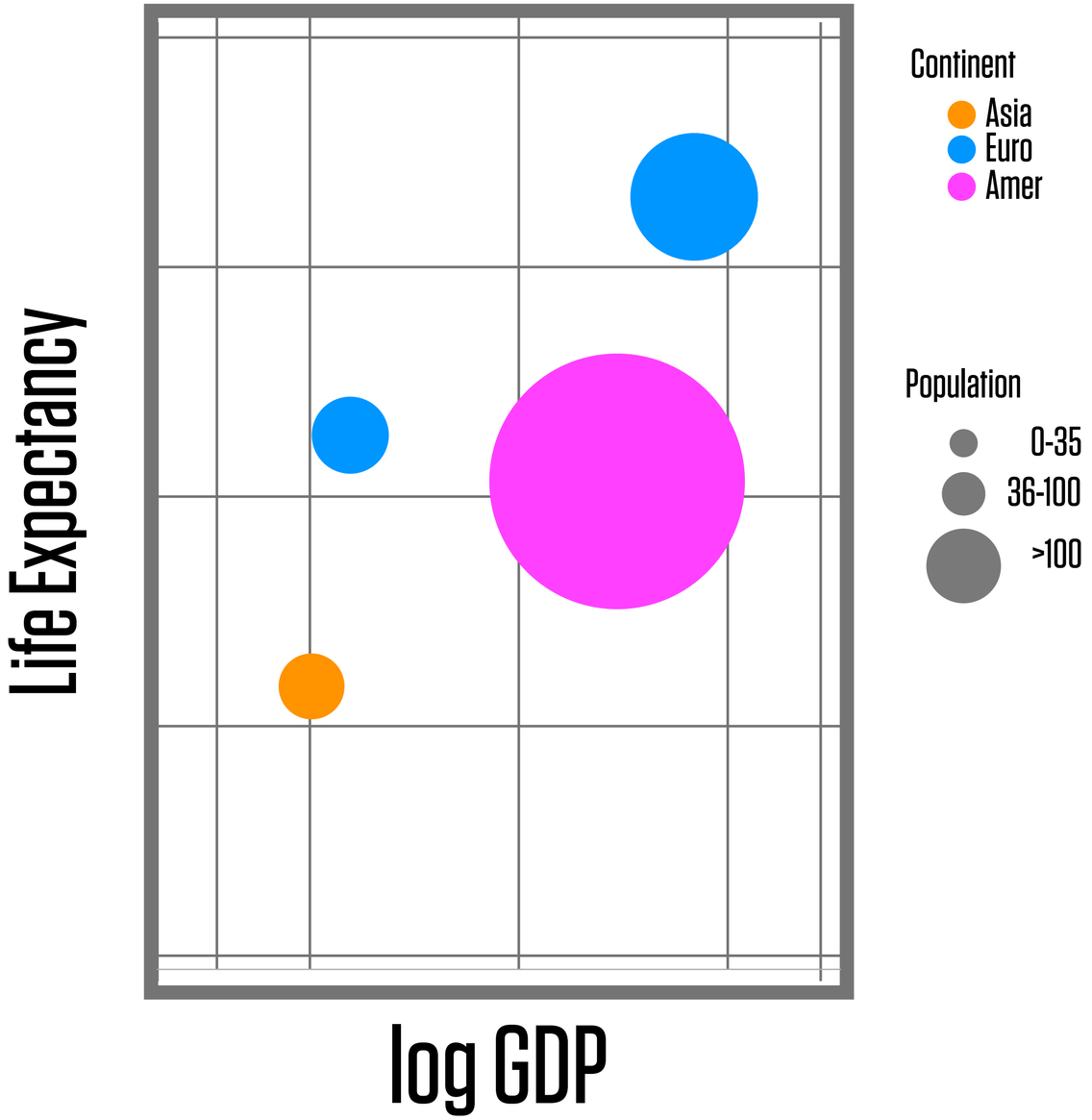
## 4. Co-Ordin Scales




```
coord_cartes
 scale_x_log
```

| gdp | lifexp | pop | continent |
|---|---|---|---|
| 340 | 65 | 31 | Euro |
| 227 | 51 | 200 | Amer |
| 909 | 81 | 80 | Euro |
| 126 | 40 | 20 | Asia |

# A Gapminder Plot



# 1. Tidy Data

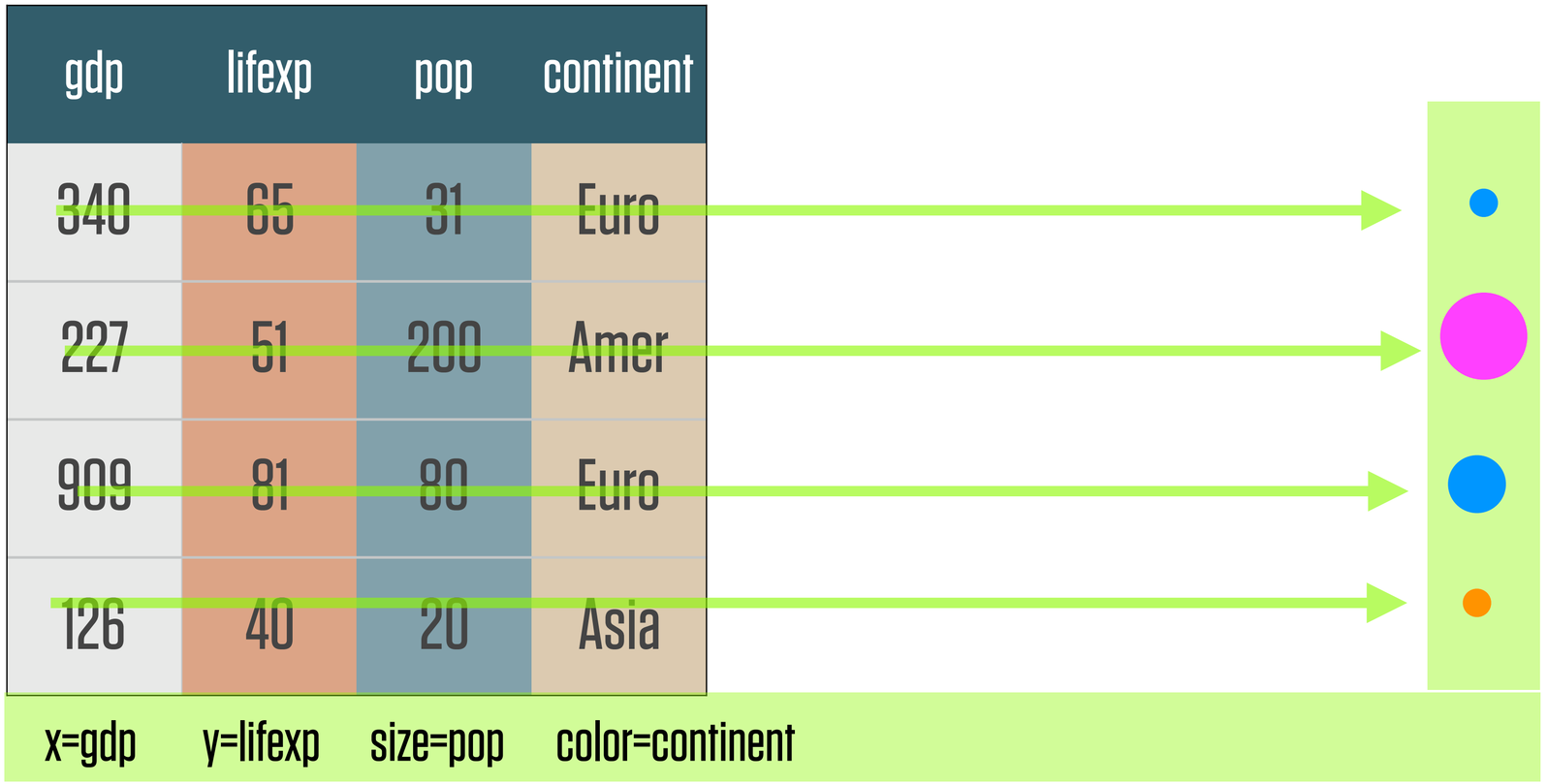
ggplot(data = gapminder)

# 2. Mapping

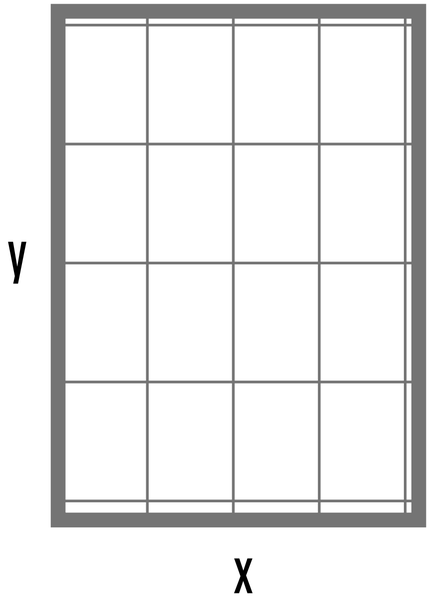
ggplot(mapping = aes(x = ...))

# 3. Geom

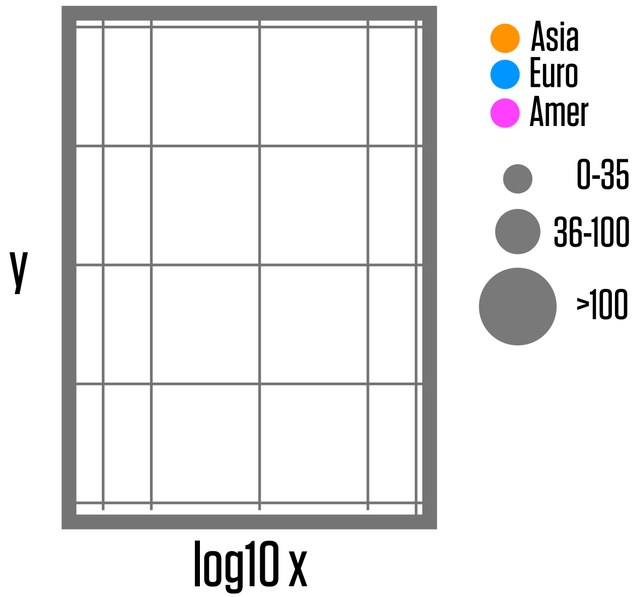
geom_point()



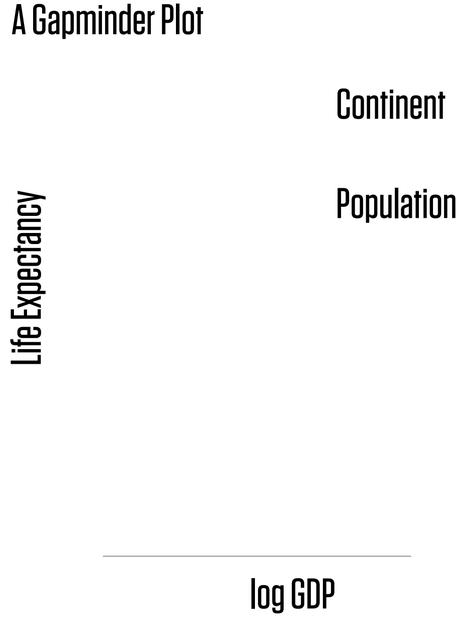
# 4. Coordinate System



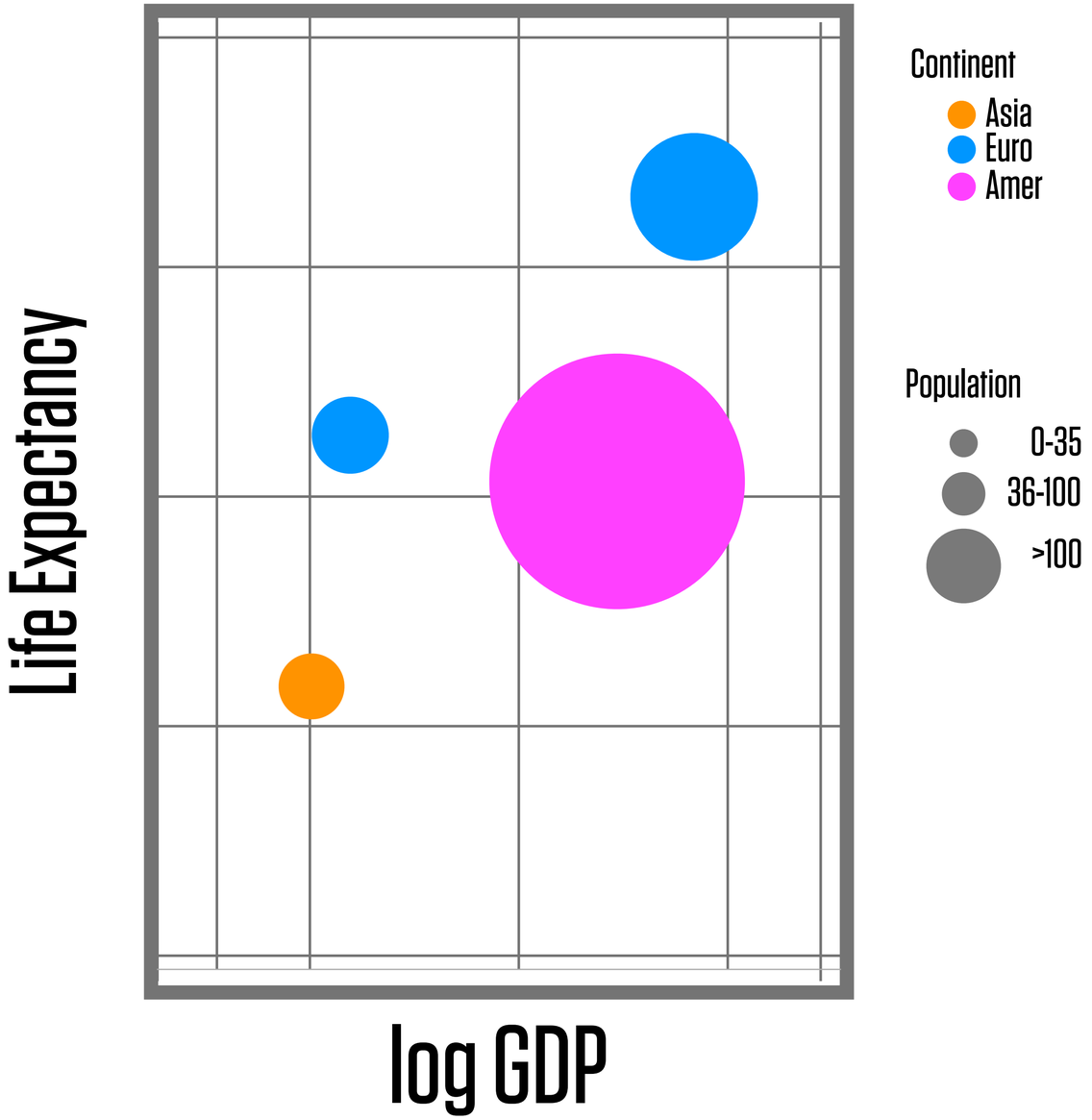

# 5. Scales




# 6. Labels & Guides



# A Gapminder Plot



# PIECE BY PIECE

```
head(gapminder)
```

```
## # A tibble: 6 × 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
```

```
dim(gapminder)
```

```
## [1] 1704    6
```

```
p <- ggplot(data = gapminder)
```

# Create a ggplot object
# Data is gapminder table

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

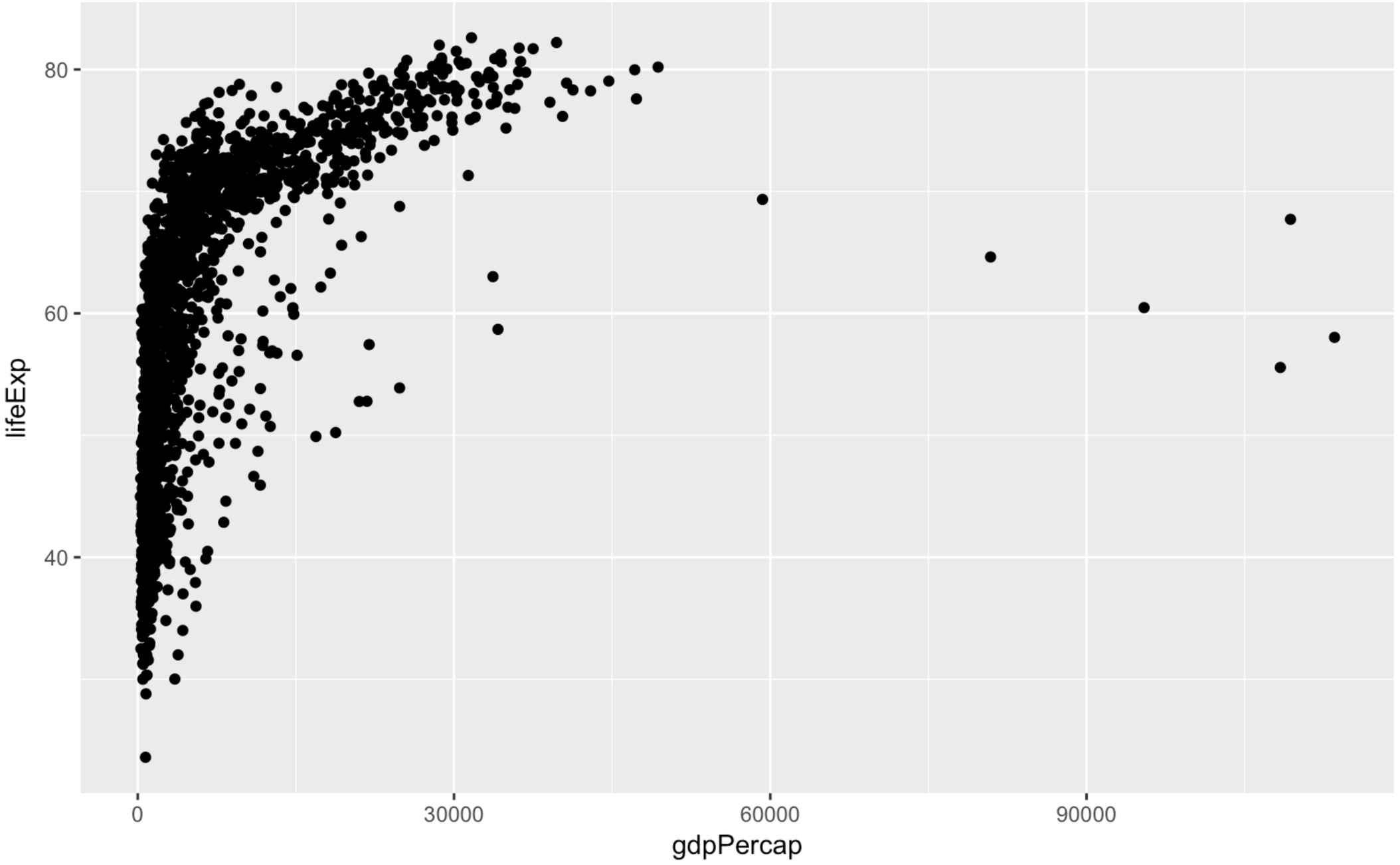
# mapping: tell ggplot the variables you want represented by features of the plot

- The `mapping = aes(...)` instruction **links variables to things you will see** on the plot.
- The `x` and `y` values are the most obvious ones.
- Other aesthetic mappings can include, e.g., `color`, `shape`, and `size`.

Mappings do **not** directly specify the **particular**, e.g., colors, shapes, or line styles that will appear on the plot. Rather they establish which variables in the data will be represented by which visible features on the plot.

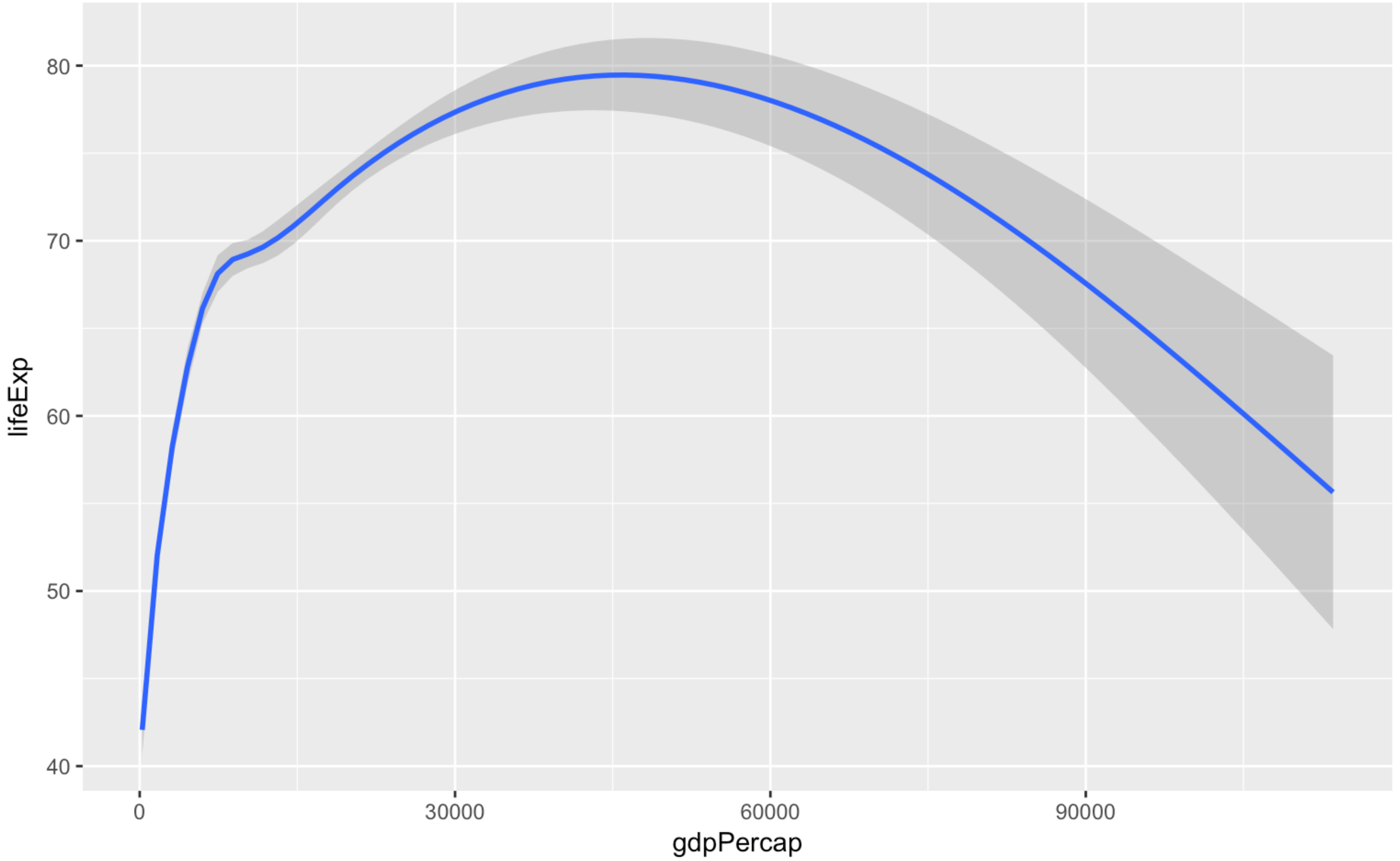
```
p + geom_point()
```

# Add a geom layer to the plot


80
60
40
lifeExp
0
30000
60000
90000
gdpPercap

```
p + geom_smooth()
```

# Try a different geom

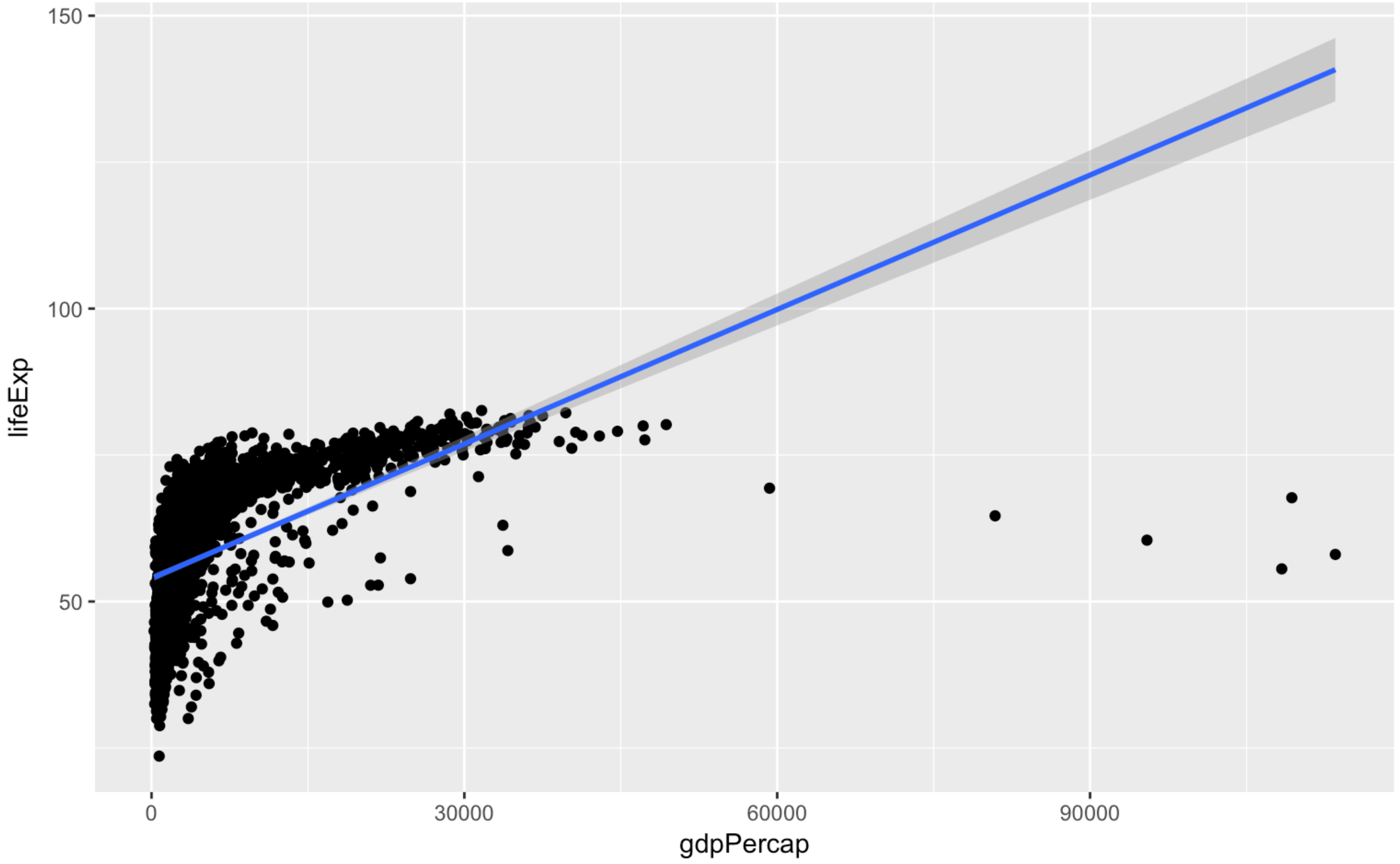

80
70
60
50
40
lifeExp
0
30000
60000
90000
gdpPercap

```
p + geom_point() +
    geom_smooth() +
    scale_x_log10(labels = scales::dollar)
```

# This process is literally additive

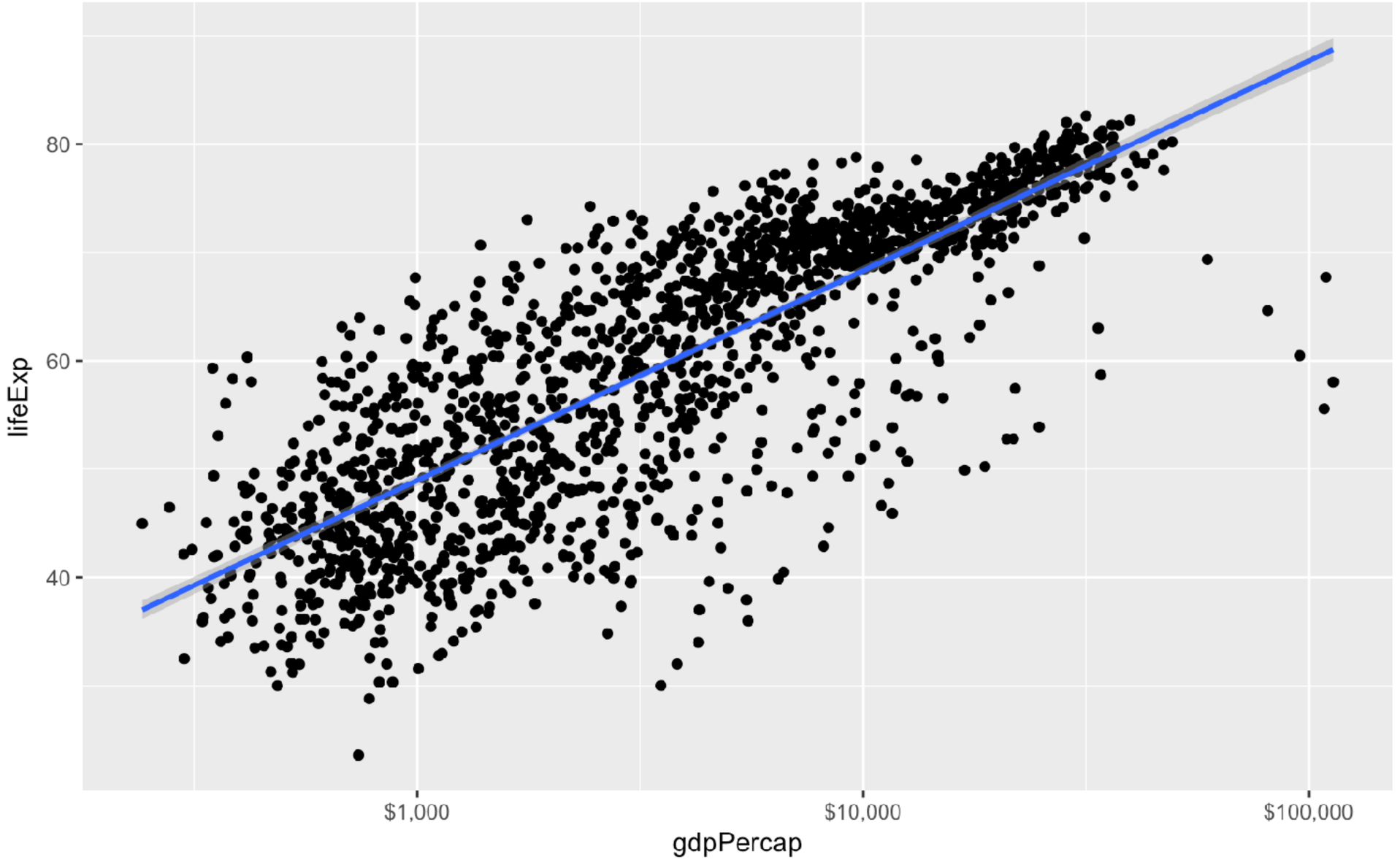
```
p + geom_point() +
    geom_smooth(method = "lm")
```

# Every geom is a function. Functions take arguments.

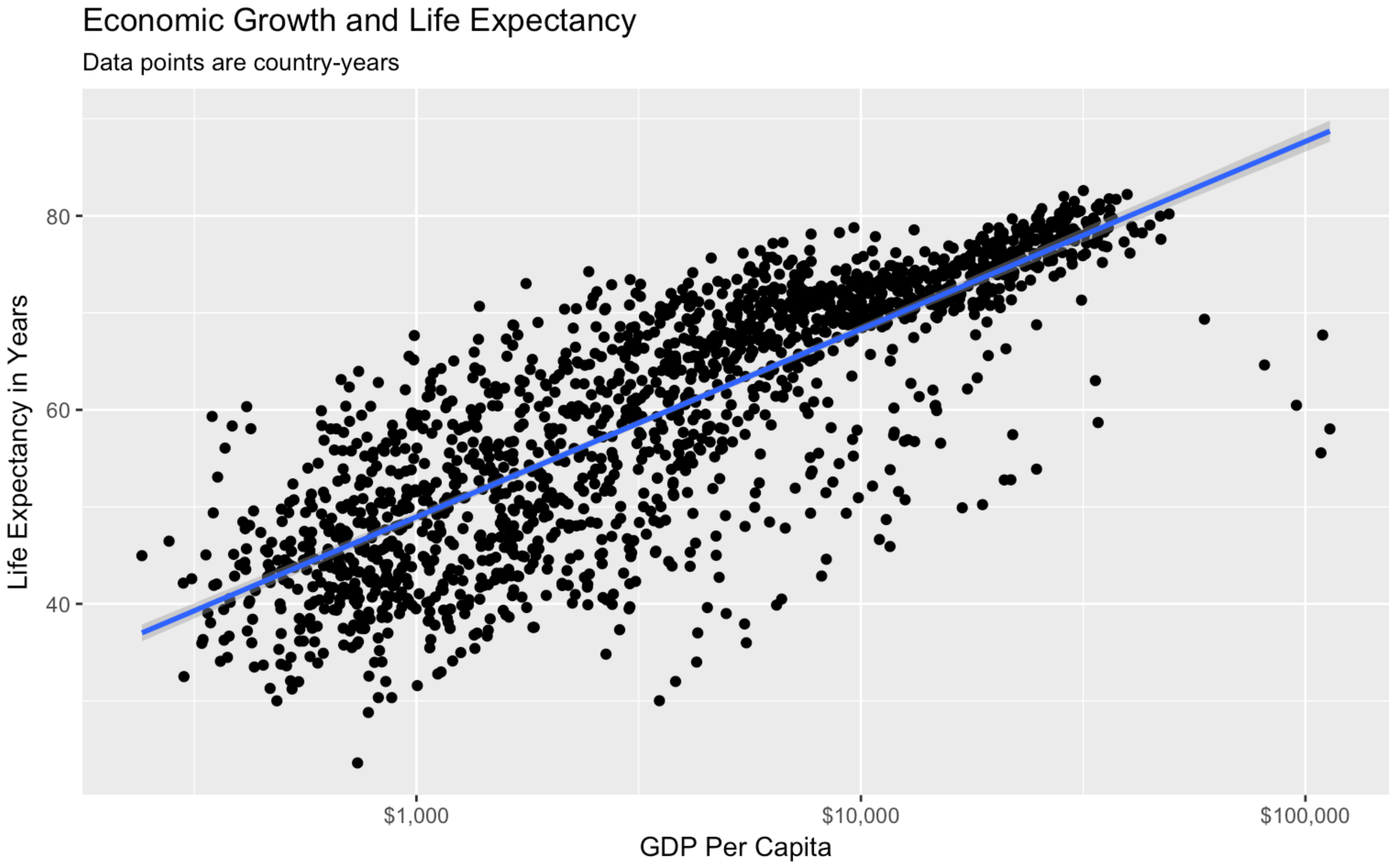

150
100
50
lifeExp
0
30000
60000
90000
gdpPercap

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))
p + geom_point() +
    geom_smooth(method = "lm") +
    scale_x_log10(label = scales::dollar)
```

# Keep Layering

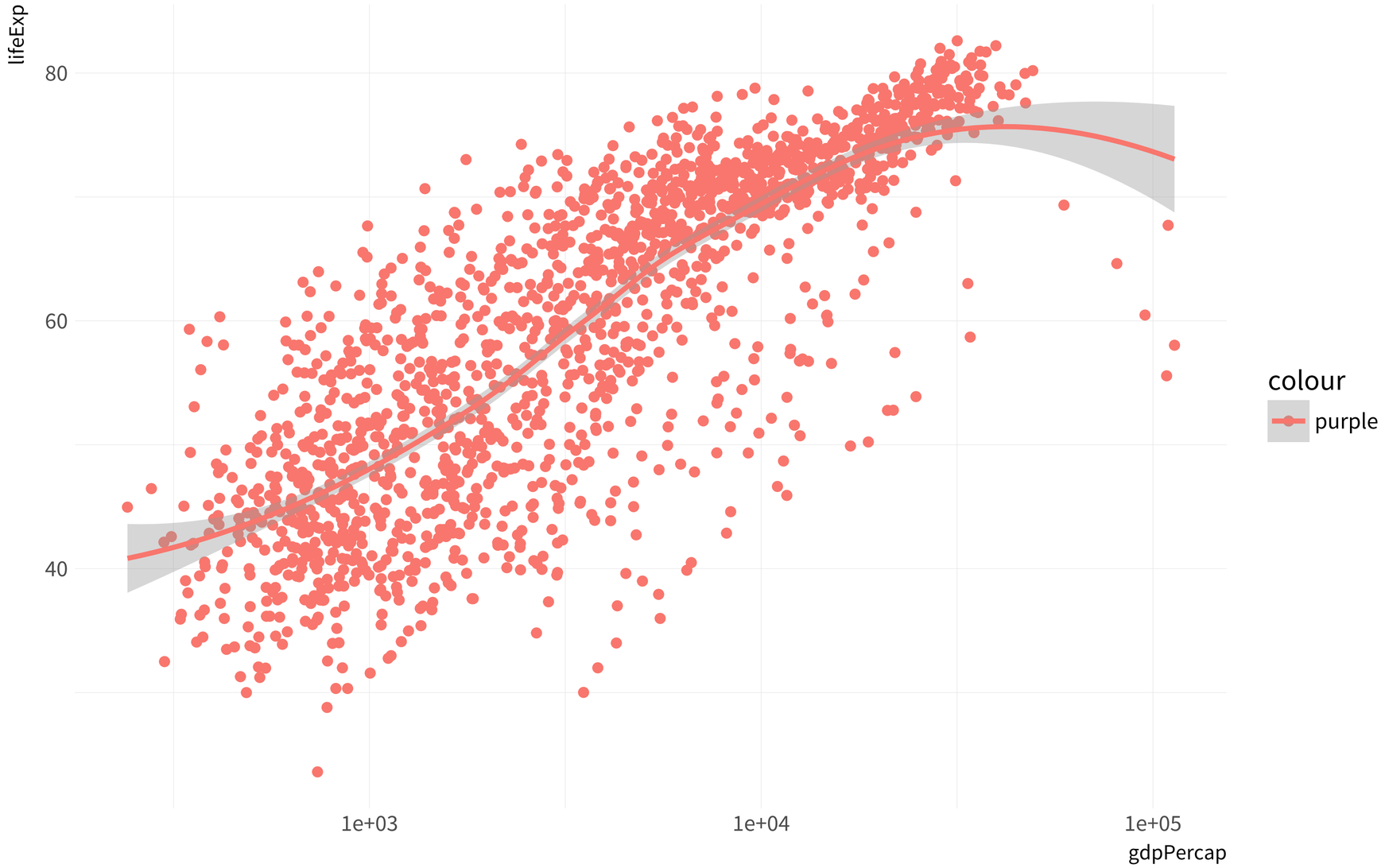

80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap

```r
p + geom_point() +
    geom_smooth(method = "gam") +
    scale_x_log10(labels = scales::dollar) +
    labs(x = "GDP Per Capita",
         y = "Life Expectancy in Years",
         title = "Economic Growth and Life Expectancy",
         subtitle = "Data points are country-years",
         caption = "Data source: Gapminder")
```


Economic Growth and Life Expectancy
Data points are country-years
Life Expectancy in Years
80
60
40
$1,000
$10,000
$100,000
GDP Per Capita

# MAPPING VS SETTING AESTHETICS

```r
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp,
                          color = "purple"))
p + geom_point() +
    geom_smooth(method = "loess") +
    scale_x_log10()
```



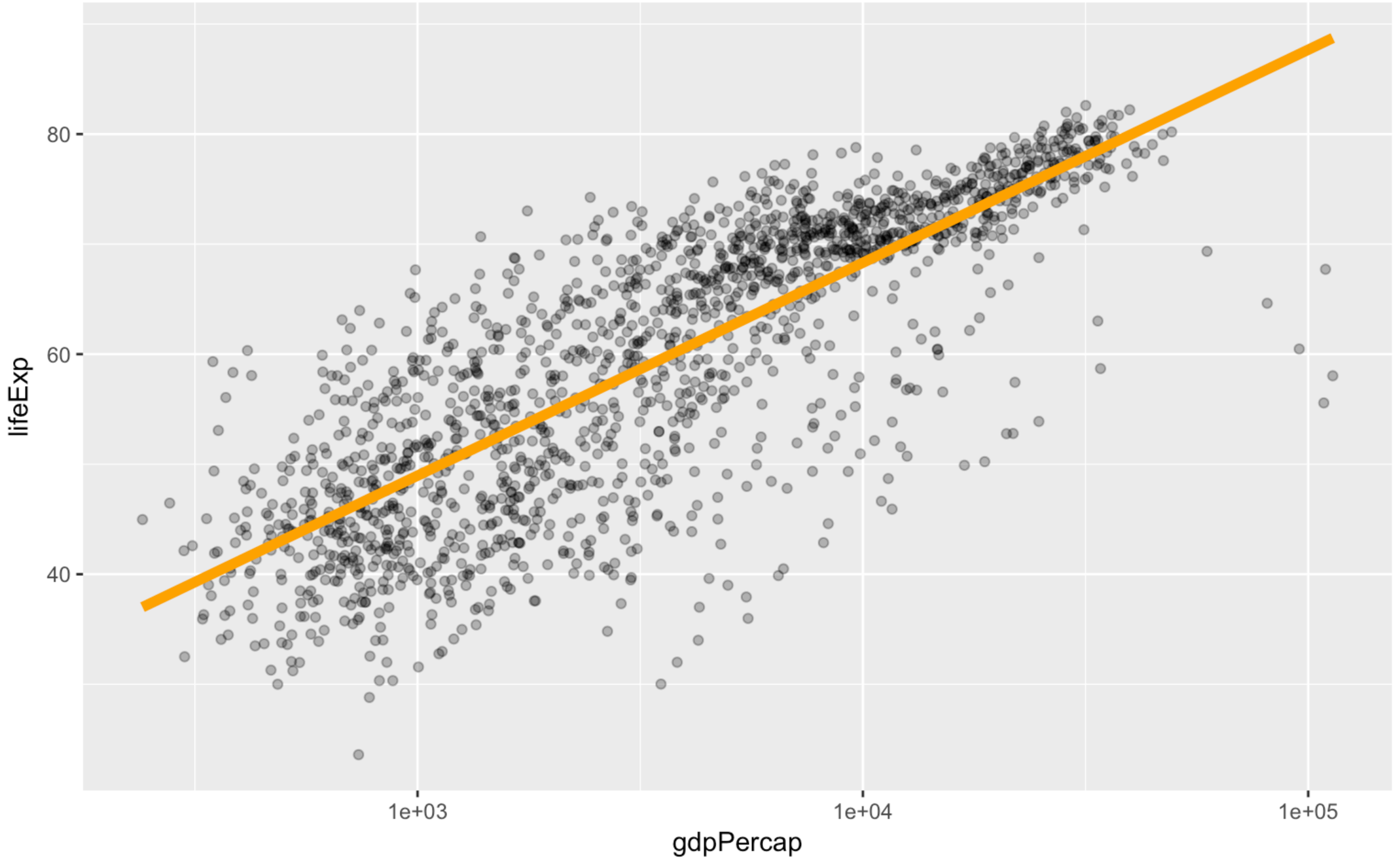

# What has gone wrong here?

```r
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp)
p + geom_point(color = "purple") +
    geom_smooth(method = "loess")) +
    scale_x_log10()
```


80
60
40
lifeExp
1e+03
1e+04
1e+05
gdpPercap

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))
p + geom_point(alpha = 0.3) +
    geom_smooth(color = "orange",
        se = FALSE, size = 2, method = "lm") +
    scale_x_log10()
```

# Here, some aesthetics are mapped, and some are set

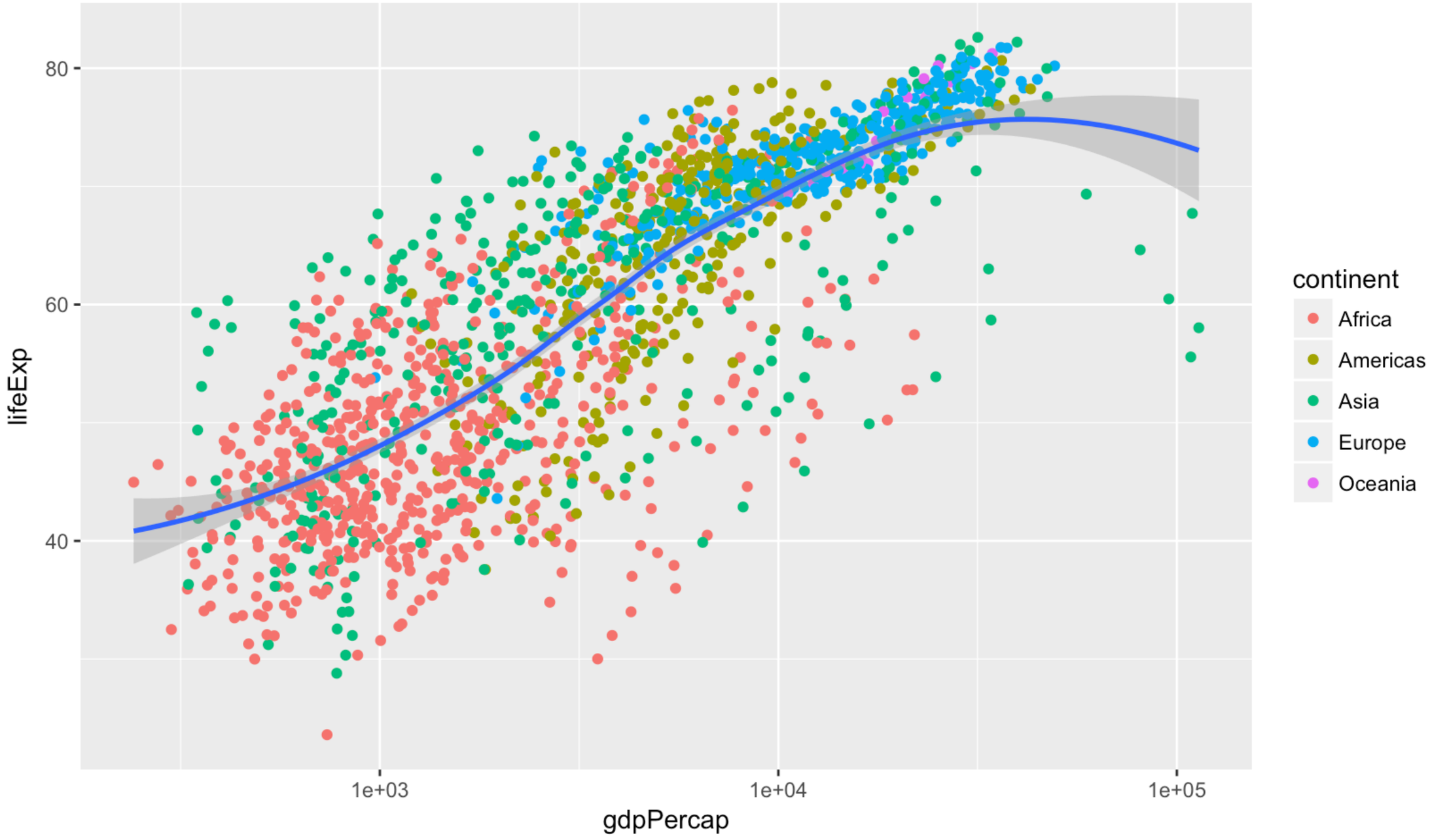

lifeExp
80
60
40
1e+03
1e+04
1e+05
gdpPercap

```r
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp,
                          color = continent,
                          fill = continent))
p + geom_point() +
    geom_smooth(method = "loess") +
    scale_x_log10()
```


80
60
40
lifeExp
1e+03
1e+04
1e+05
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania

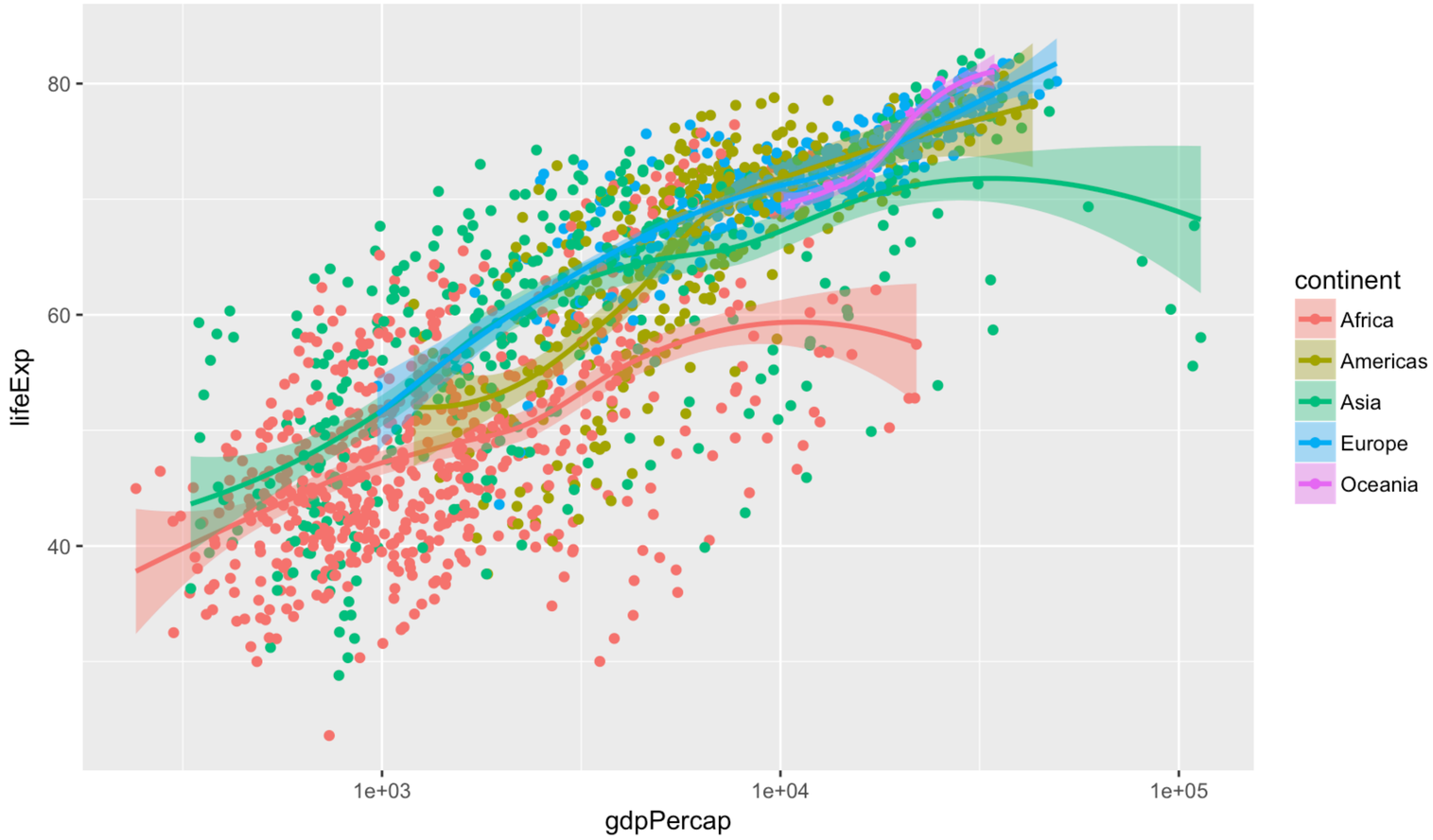
# MAP or SET AESTHETICS per geom

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))
p + geom_point(mapping = aes(color = continent)) +
    geom_smooth(method = "loess") +
    scale_x_log10()
```

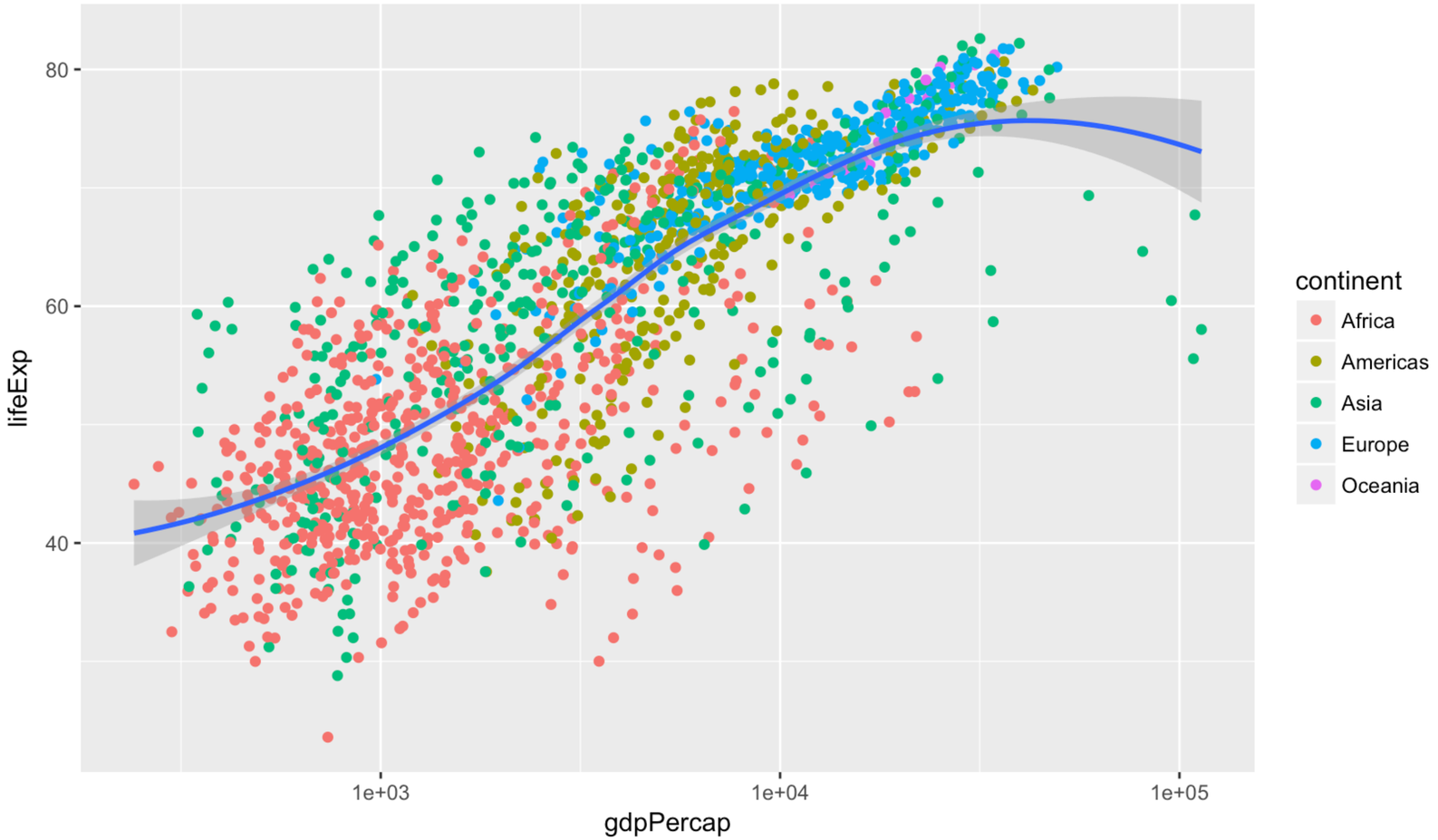

80
60
40
lifeExp
1e+03
1e+04
1e+05
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania

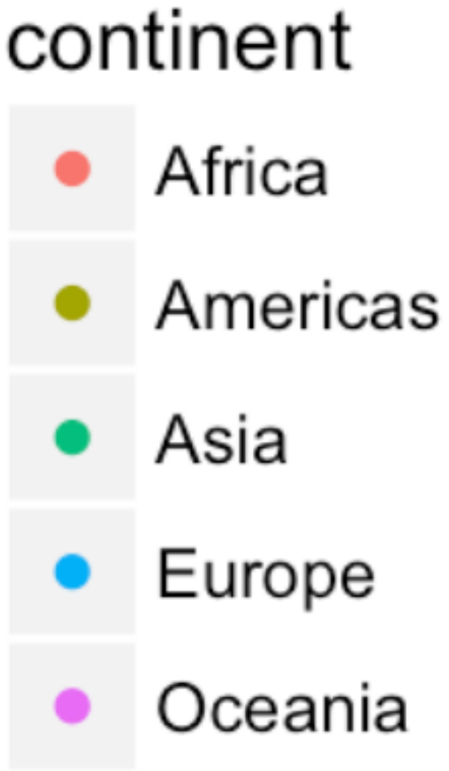
# PAY CLOSE ATTENTION TO HOW SCALES ARE DRAWN, AND WHY

```r
p <- ggplot(data = gapminder,
            mapping =
              aes(x = gdpPercap, y = lifeExp,
                  color = continent, fill = continent))
p + geom_point() +
    geom_smooth(method = "loess") +
    scale_x_log10()
```




```
p <- ggplot(data = gapminder,
            mapping =
              aes(x = gdpPercap, y = lifeExp))
p + geom_point(mapping = aes(color = continent)) +
    geom_smooth(method = "loess") +
    scale_x_log10()
```



# REMEMBER:
# EVERY MAPPED VARIABLE HAS A SCALE

# Saving Your Work

# With ggsave

```
ggsave()

ggsave("figures/my_figure.png")

ggsave("my_figure.pdf")

ggsave("my_figure.pdf",
       plot = p5,
       scale = 1.2)

ggsave("figures/my-figure.pdf",
       plot = p5,
       width = 8,
       height = 5)
```

# With pdf() or other graphics devices

```
pdf(file = "plot.pdf", height = 5in,
    width = 5in)
print(p4)

dev.off()
```

Open device ...

... and close when done

# Within an Rmd chunk

```
```{r my_plot, fig.cap="My Plot", fig.width=9, fig.height=8}

p + geom_point(mapping =
          aes(color = continent)) +
    geom_smooth(method = "loess") +
    scale_x_log10()

```
```

# Set defaults in your first code chunk

```
knitr::opts_chunk$set(fig.width=8, fig.height=5)
```

# Getting Help

The name of the function, and the library it is in.

mean {base} R Documentation

## Arithmetic Mean

### Description

What it does.

Generic function for the (trimmed) arithmetic mean.

### Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

The function's name, and in the parentheses the named arguments it expects, in the order it expects them. If an argument has a default value, it is shown. Arguments without default values (e.g. x) must be provided by you.

### Arguments

More details on each named argument. This will tell you what class of thing each argument has to be—an object, a number, a data frame, a logical value, etc.

| | |
|---|---|
| x | An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only. |
| trim | the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Values of trim outside that range are taken as the nearest endpoint. |
| na.rm | a logical value indicating whether NA values should be stripped before the computation proceeds. |
| ... | further arguments passed to or from other methods. |

The ellipsis allows other arguments to be passed to and from the function.

What the function returns—i.e., the result of whatever operation or calculation it performs. This can be a single number, as here, or a multi-part object such as a list, a data frame, a plot, or a model.

### Value

If `trim` is zero (the default), the arithmetic mean of the values in x is computed, as a numeric or complex vector of length one. If x is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.
If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

### References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

### See Also

`weighted.mean`, `mean.POSIXct`, `colMeans` for row and column means.

Other related functions

### Examples

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
```

Self-contained examples that you can run at the console. These may use built-in datasets or other R functions.

[Package *base* version 3.4.3 Index]

Visit the package's Index page to look for Demos and Vignettes detailing how it works.