Data Visualization in R Programming: A Practical Introduction

From Pro. Kieran Healy

Data Visualization Notes

This is a outcome in PDF file from RMarkdown project template to accompany training with *Data Visualization*. You can use it to take notes, write your code, and produce a good-looking, reproducible document that records the work you have done. At the very top of the file is a section of *metadata*, or information about what the file is and what it does. The metadata is delimited by three dashes at the start and another three at the end. You should change the title, author, and date to the values that suit you. Keep the `output` line as it is for now, however. Each line in the metadata has a structure. First the *key* ("title", "author", etc), then a colon, and then the *value* associated with the key.

This Document is an RMarkdown File

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.

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. A *code chunk* is a specially delimited section of the file. You can add one by moving the cursor to a blank line choosing Code > Insert Chunk from the RStudio menu. When you do, an empty chunk will appear in your file.

Code chunks are delimited by three backticks (found to the left of the 1 key on US and UK keyboards) at the start and end. The opening backticks also have a pair of braces and the letter `r`, to indicate what language the chunk is written in. You write your code inside the code chunks. Write your notes and other material around them, as here.

Things to know about R

Any new piece of software takes a bit of getting used to. This is especially true when using an IDE to work in a language like R.

Everything has a name

In R, everything you deal with has a name. You refer to things by their names as you examine, use, or modify them. Named entities include variables (like `x`, or `y`), data that you have loaded (like `my_data`), and functions that you use. (More about functions momentarily.) You will spend a lot of time talking about, creating, referring to, and modifying things with names.

Some names are forbidden. These include reserved words like `FALSE` and `TRUE`, core programming words like `Inf`, `for`, `else`, `break`, `function`, and words for special entities like `NA` and `NaN`. (These last two are codes designating missing data and "Not a Number", respectively.) You probably won't use these names by accident, but it's good do know that they are not allowed.

Some names you should not use, even if they are technically permitted. These are mostly words that are already in use for objects or functions that form part of the core of R. These include the names of basic functions like `q()` or `c()`, common statistical functions like `mean()`, `range()` or `var()`, and built-in mathematical constants like `pi`.

Names in R are case sensitive, not much difficult from others programming languages. The object `my_data` is not the same as the object `My_Data`. When choosing names for things, be concise, consistent, and informative. Follow the style of the tidyverse and name things in lower case, separating words with the underscore character, `_`, as needed. Do not use spaces when naming things, including variables in your data.

Everything is an object

Some objects are built in to R, some are added via libraries, and some are created by the user. But almost everything is some kind of object. The code you write will create, manipulate, and use named objects as a matter of course. We can start immediately. Let's create a vector of numbers. The command `c()` is a function. It's short for "combine" or "concatenate". It will take a sequence of comma-separated things inside the parentheses and join them together into a vector where each element is still individually accessible.

```
c(1, 2, 3, 1, 3, 2020, 2021)
```

```
## [1]    1    2    3    1    3 2020 2021
```

```
# create object's name
my_numbers <- c(1, 2, 3, 1, 3, 2020, 2021)

futureforum_team <- c("Dara", "Soriya", "Sokhouy", "Kimly", "Heang", "Samnang")
y <- 1:9

# call for object that we want to see
futureforum_team
```

```
## [1] "Dara"    "Soriya"  "Sokhouy" "Kimly"   "Heang"   "Samnang"
```

```
# you do things using functions
x = my_numbers

# call for show function x
x
```

```
## [1]    1    2    3    1    3 2020 2021
```

```
# calculate mean for function x
mean(x)
```

```
## [1] 578.7143
```

```
# sum function x
sum(x)
```

```
## [1] 4051
```

```
# log in `x`
log(x)
```

```
## [1] 0.0000000 0.6931472 1.0986123 0.0000000 1.0986123 7.6108528 7.6113477
```

```
# exponential function `x`
exp(x)
```

```
## [1]  2.718282  7.389056 20.085537  2.718282 20.085537       Inf       Inf
```

```
x^6
```

```
## [1] 1.000000e+00 6.400000e+01 7.290000e+02 1.000000e+00 7.290000e+02
## [6] 6.793729e+19 6.813933e+19
```

```r
# sin and computes arc-sine or sine inverse of `x`
sin(x)
```

```
## [1]  0.84147098  0.90929743  0.14112001  0.84147098  0.14112001  0.04406199
## [7] -0.81684695
```

```r
asin(x)
```

```
## Warning in asin(x): NaNs produced
```

```
## [1] 1.570796      NaN      NaN 1.570796      NaN      NaN      NaN
```

```r
# summary x
x_summary <- summary(x)
x_summary
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     1.5     3.0   578.7  1011.5  2021.0
```

```r
# functions come in libraries
table(my_numbers)
```

```
## my_numbers
##    1    2    3 2020 2021
##    2    1    2    1    1
```

```r
# calculate standard deviation of `my_numbers`
sd(my_numbers)
```

```
## [1] 984.9275
```

```r
# `my_mumbers` multiply with 2021
my_numbers * 2021
```

```
## [1]    2021    4042    6063    2021    6063 4082420 4084441
```

```r
# `my_mumbers` plus 1
my_numbers + 1
```

```
## [1]    2    3    4    2    4 2021 2022
```

```r
# `my_mumbers` divide with 12
my_numbers / 12
```

```
## [1]   0.08333333   0.16666667   0.25000000   0.08333333   0.25000000
## [6] 168.33333333 168.41666667
```

```r
# `my_mumbers` plus `my_mumbers`
my_numbers + my_numbers
```

```
## [1]    2    4    6    2    6 4040 4042
```

```r
# if you're not sure what an object is, ask for its class
class(my_numbers)
```

```
## [1] "numeric"
```

```r
class(futureforum_team)
```

```
## [1] "character"
```

```r
class(x_summary)
```

```
## [1] "summaryDefault" "table"
```

```
class(summary)
```

```
## [1] "function"
```

```
my_new_vector <- c(my_numbers, "Apple")
my_new_vector
```

```
## [1] "1"     "2"     "3"     "1"     "3"     "2020"  "2021"  "Apple"
```

```
class(my_new_vector)
```

```
## [1] "character"
```

Load Libraries

To begin we must load some libraries we will be using. If we do not load them, R will not be able to find the functions contained in these libraries. But for the first time with *R Programming*, you should install library packages that common use.

Here, the braces at the start of the code chunk have some additional options set in them. There is the language, `r`, as before. This is required. Then there is the word `setup`, which is a label for your code chunk. Labels are useful to briefly say what the chunk does. Label names must be unique (no two chunks in the same document can have the same label) and cannot contain spaces. Then, after the comma, an option is set: `include=FALSE`. This tells R to run this code but not to include the output in the final document.

If you have not installed these required libraries yet, make sure you have an internet connection and install them now.

```
# to install these packages, in the line above.

# basic method that common use for new learner
install.packages("tidyverse")
install.packages("broom")

# advanced method
my_packages <- c("tidyverse", "broom", "coefplot", "cowplot", "drat", "fs",
                 "gapminder", "GGally", "ggrepel", "ggridges", "gridExtra",
                 "here", "interplot", "margins", "maps", "mapproj",
                 "mapdata", "MASS", "quantreg", "rlang", "scales",
                 "survey", "srvyr", "viridis", "viridisLite", "devtools")

# R Studio should then download and install these packages for you.
install.packages(my_packages, repos = "http://cran.rstudio.com")

# load libraries for use
# basic method
library(gapminder)
library(here)
library(tidyverse)

# advanced method
Packages <- my_packages

Packages <- c("tidyverse", "broom", "coefplot", "cowplot", "drat", "fs",
              "gapminder", "maps", "mapproj", "survey", "srvyr", "viridis",
              "viridisLite", "devtools")
```

```r
lapply(Packages, library, character.only = TRUE)

# to update our package
update.packages()

# to know which pack need an update
old.packages()

# to know which packages are being loaded
search()

# request for help
?hist
help(package = "tidyverse")
example("hist")

# use ?? to search by keyword
??regression
```

So let we go with some datasets on R

```r
# call the dataset from R
mtcars
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
```

```
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```r
mymtcars <- mtcars

class(mymtcars)
```

```
## [1] "data.frame"
```

```r
# list the variables in `mymtcars`
names(mymtcars)
```

```
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear"
## [11] "carb"
```

```r
# or list objects in the working environment
ls(mymtcars)
```

```
##  [1] "am"   "carb" "cyl"  "disp" "drat" "gear" "hp"   "mpg"  "qsec" "vs"
## [11] "wt"
```

```r
# list the structure of `mymtcars`
str(mymtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

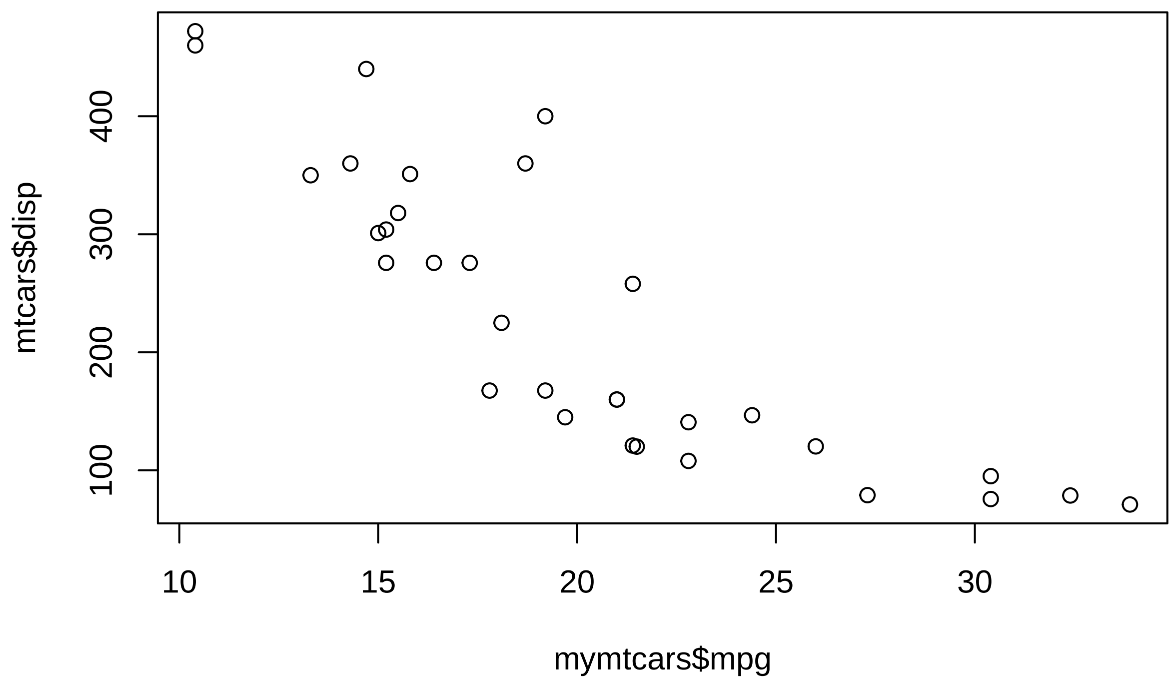
```r
# print first 10 rows of `mymtcars`
head(mymtcars, n=7)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8  360 245 3.21 3.570 15.84  0  0    3    4
```

```r
# print last 5 rows of `mymtcars`
tail(mymtcars, n=4)
```

```
##                 mpg cyl disp  hp drat   wt qsec vs am gear carb
## Ford Pantera L 15.8   8  351 264 4.22 3.17 14.5  0  1    5    4
## Ferrari Dino   19.7   6  145 175 3.62 2.77 15.5  0  1    5    6
## Maserati Bora  15.0   8  301 335 3.54 3.57 14.6  0  1    5    8
## Volvo 142E     21.4   4  121 109 4.11 2.78 18.6  1  1    4    2
```

```r
# we can also create plots:
plot(mymtcars$mpg, mtcars$disp)
```



Now, we try with another example and use the dataset from link

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

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

Or locally:

```r
organs <- read.csv(file = "Data/organdonation.csv")

# tibble package to show only the first 10 of the dataset
library(tibble)
```

```
## Warning: package 'tibble' was built under R version 3.6.3
```

```r
as_tibble(organs)
```

```
## # A tibble: 238 x 21
##    country  year donors   pop pop.dens   gdp gdp.lag health health.lag pubhealth
##    <fct>   <int>  <dbl> <int>    <dbl> <int>   <int>  <dbl>      <dbl>     <dbl>
##  1 Austra~    NA   NA   17065    0.220 16774   16591   1300       1224       4.8
##  2 Austra~  1991   12.1 17284    0.223 17171   16774   1379       1300       5.4
##  3 Austra~  1992   12.4 17495    0.226 17914   17171   1455       1379       5.4
##  4 Austra~  1993   12.5 17667    0.228 18883   17914   1540       1455       5.4
##  5 Austra~  1994   10.2 17855    0.231 19849   18883   1626       1540       5.4
```

```
##  6 Austra~  1995  10.2  18072    0.233 21079   19849   1737        1626       5.5
##  7 Austra~  1996  10.6  18311    0.237 21923   21079   1846        1737       5.6
##  8 Austra~  1997  10.3  18518    0.239 22961   21923   1948        1846       5.7
##  9 Austra~  1998  10.5  18711    0.242 24148   22961   2077        1948       5.9
## 10 Austra~  1999   8.67 18926    0.244 25445   24148   2231        2077       6.1
## # ... with 228 more rows, and 11 more variables: roads <dbl>, cerebvas <int>,
## #   assault <int>, external <int>, txp.pop <dbl>, world <fct>, opt <fct>,
## #   consent.law <fct>, consent.practice <fct>, consistent <fct>, ccode <fct>
```

Make your first figure

```
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

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

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

```
p
```

```
p + geom_point()
```

Build your plots layer by layer

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

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

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

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

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

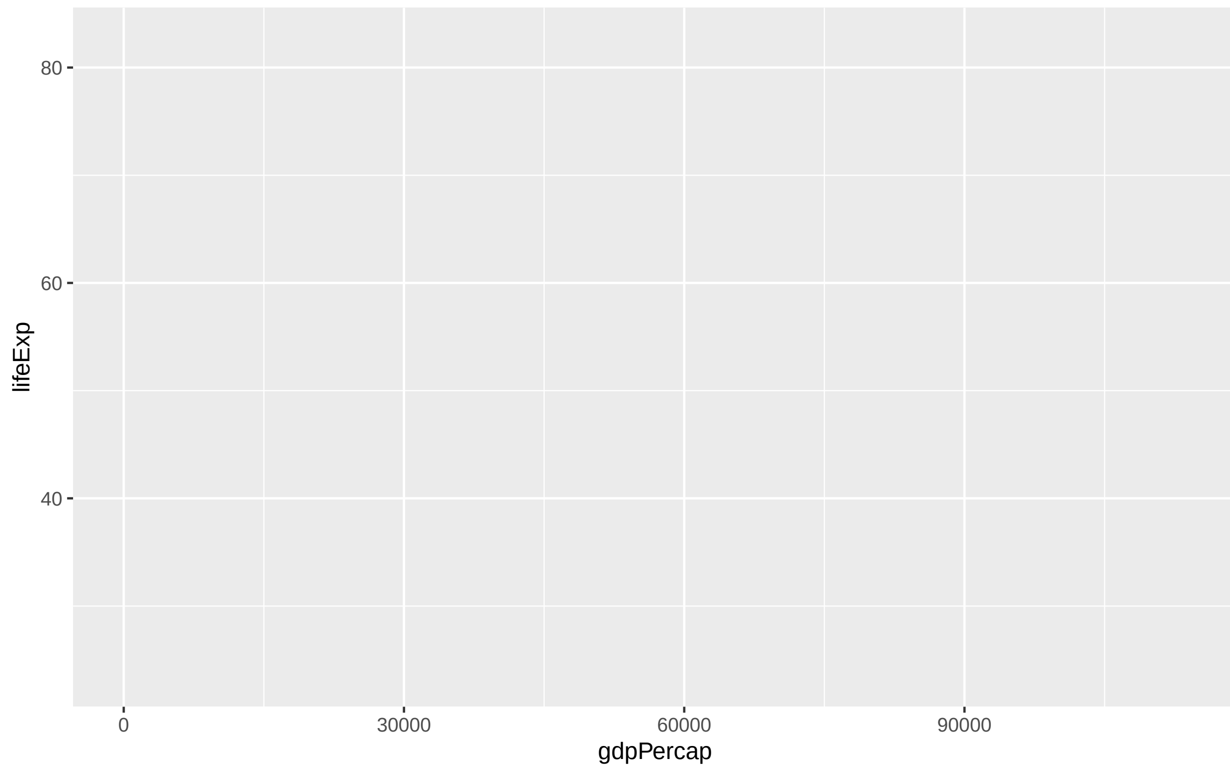


Figure 1: This empty plot has no geoms.

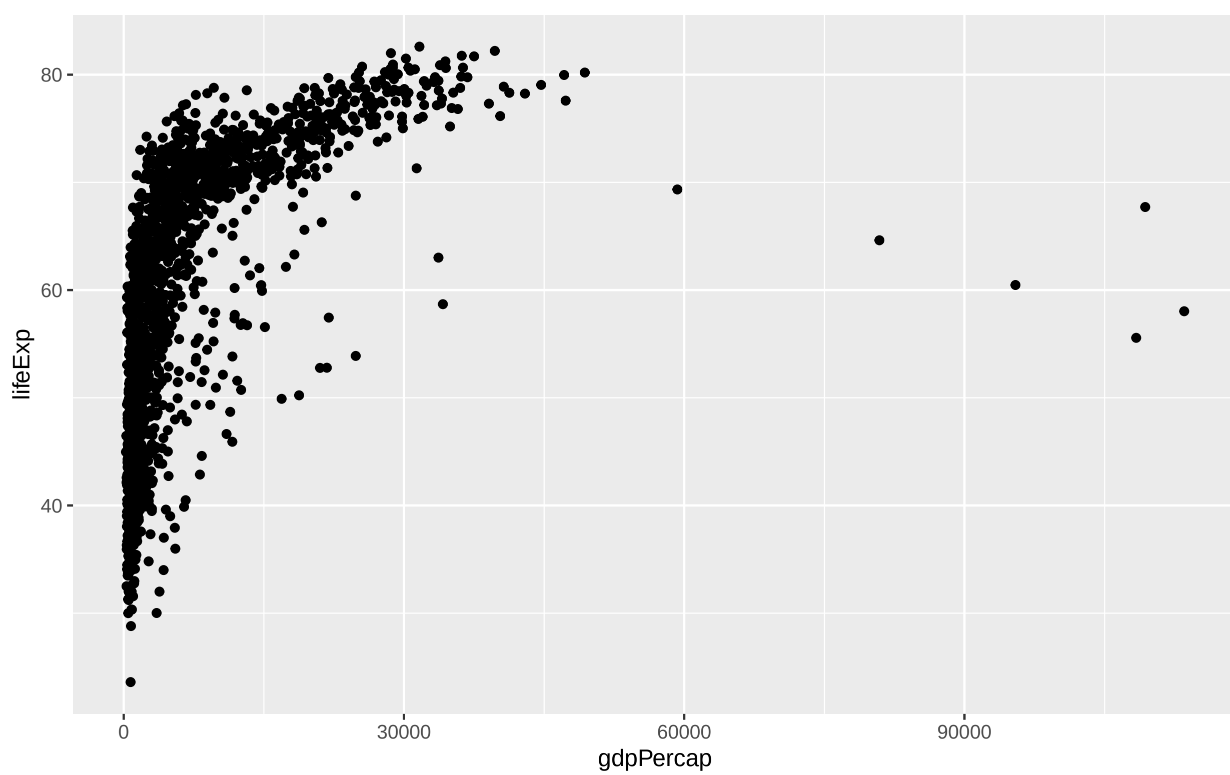


Figure 2: A scatterplot of Life Expectancy vs GDP

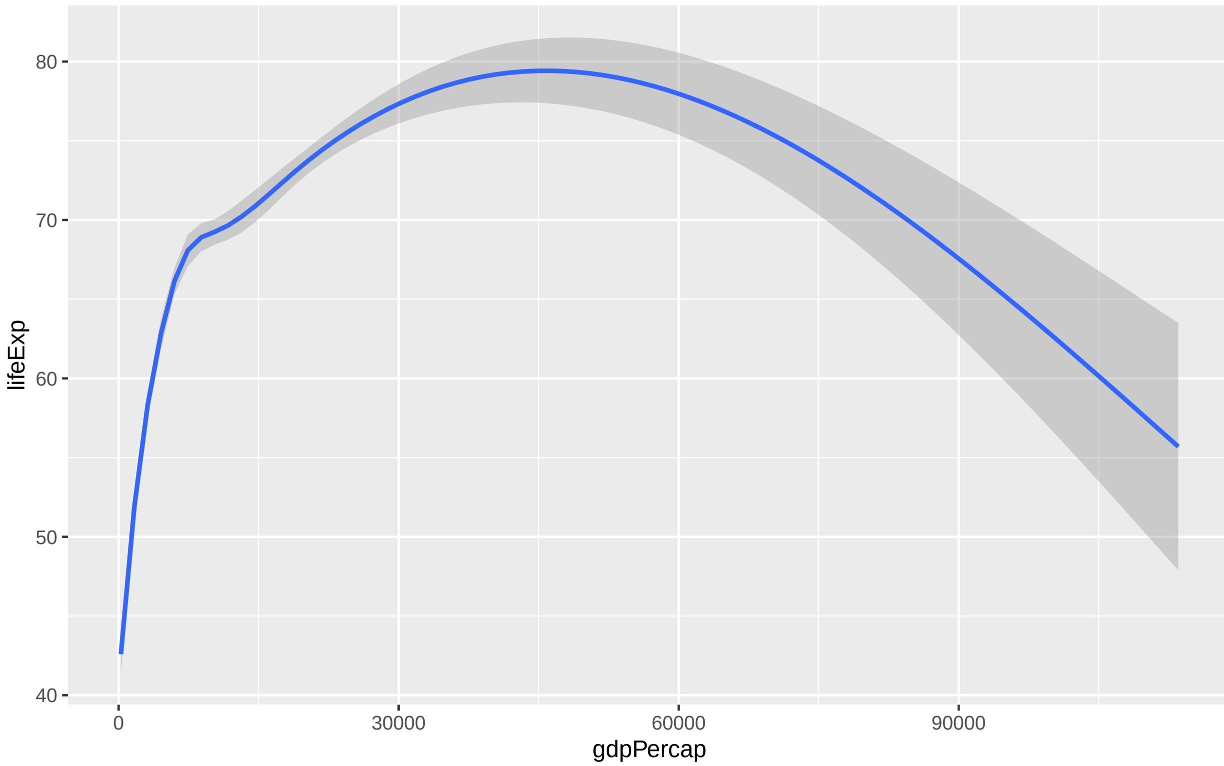


Figure 3: Life Expectancy vs GDP, using a smoother.

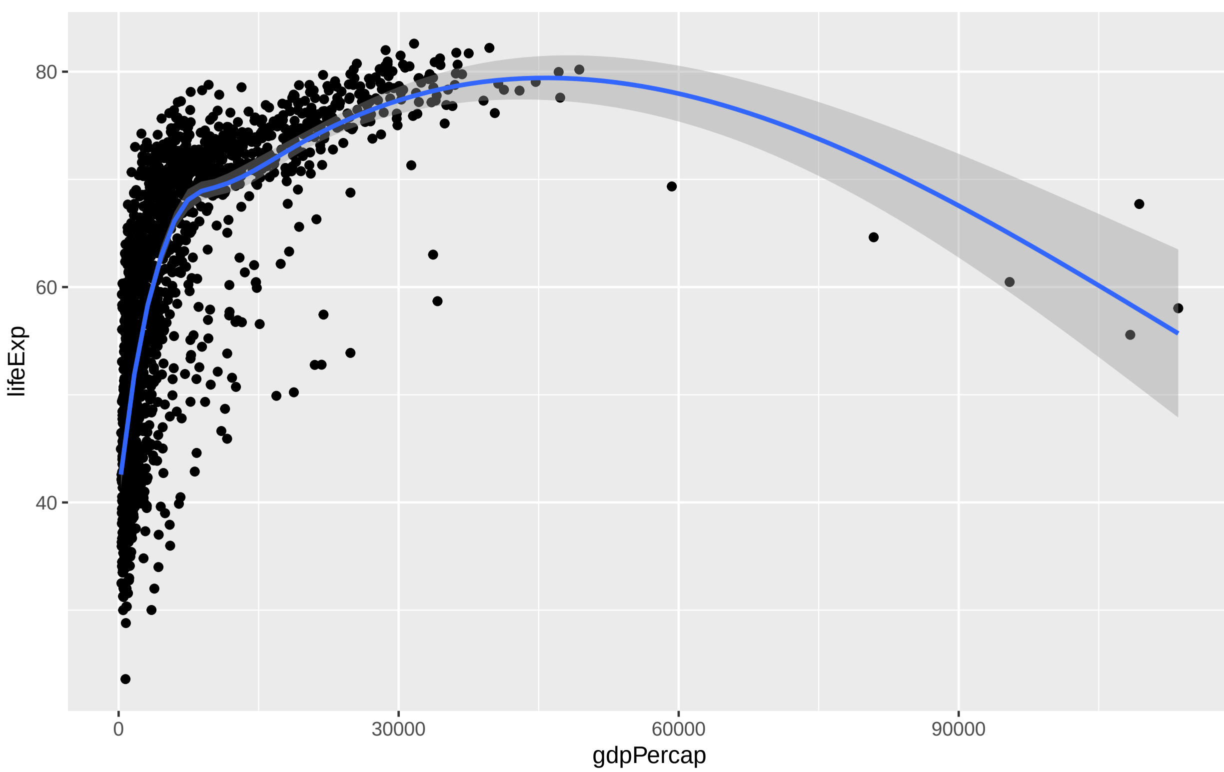


Figure 4: Life Expectancy vs GDP, showing both points and a GAM smoother.

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

```
## `geom_smooth()` using formula 'y ~ x'
```

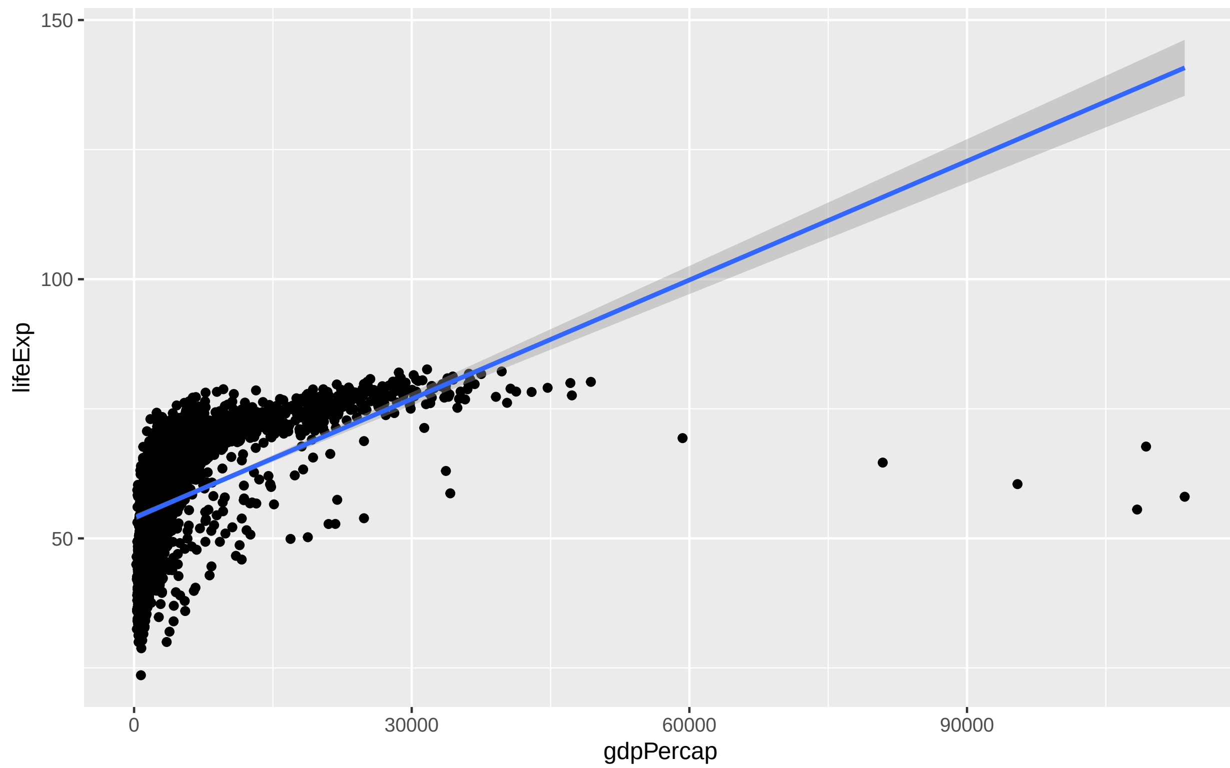


Figure 5: Life Expectancy vs GDP, points and an ill-advised linear fit.

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

```
## `geom_smooth()` using formula 'y ~ s(x, bs = "cs")'
```

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

```
## `geom_smooth()` using formula 'y ~ s(x, bs = "cs")'
```

Mapping aesthetics vs setting them

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

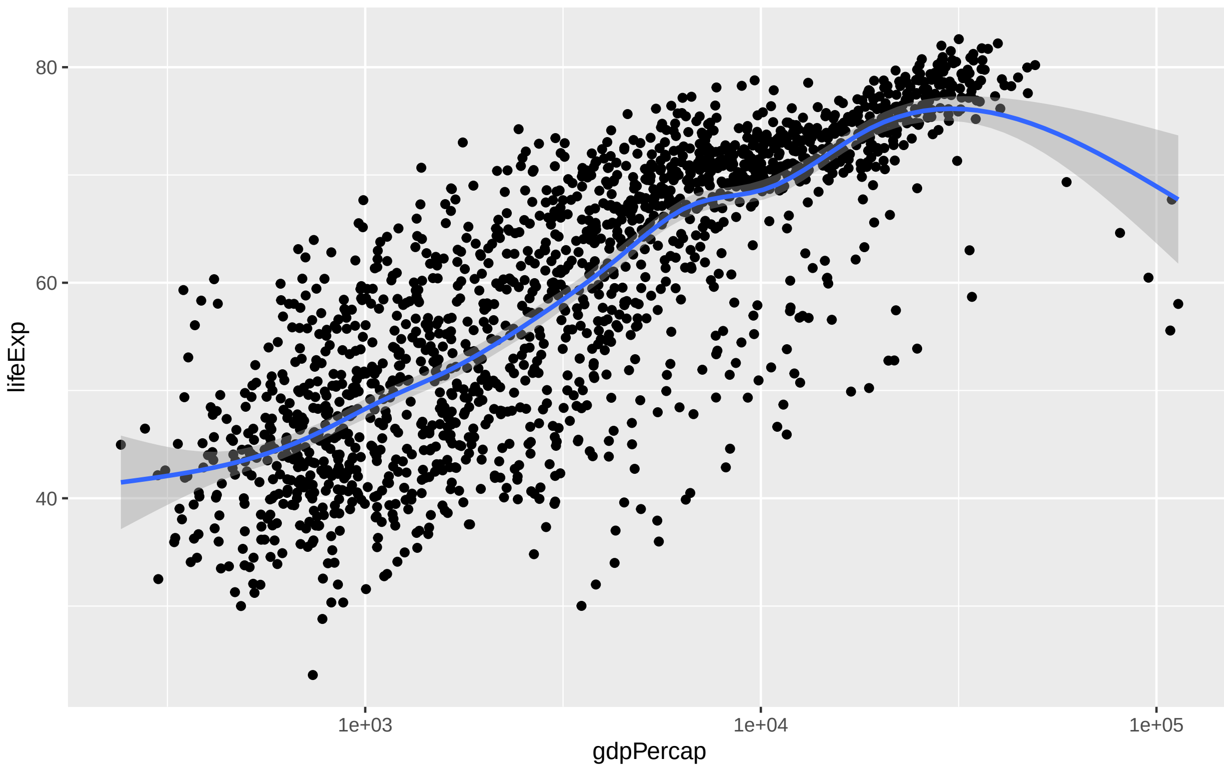


Figure 6: Life Expectancy vs GDP scatterplot, with a GAM smoother and a log scale on the x-axis.

```
p + geom_point() +
    geom_smooth(method = "loess") +
    scale_x_log10()
```

```
## `geom_smooth()` using formula 'y ~ x'
```

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

```
## `geom_smooth()` using formula 'y ~ x'
```

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))
p + geom_point(alpha = 0.3) +
    geom_smooth(color = "orange", se = FALSE, size = 8, method = "lm") +
    scale_x_log10()
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
p <- ggplot(data = gapminder, mapping = aes(x = gdpPercap, y=lifeExp))
p + geom_point(alpha = 0.3) + geom_smooth(method = "gam") +
    scale_x_log10(labels = scales::dollar) +
```

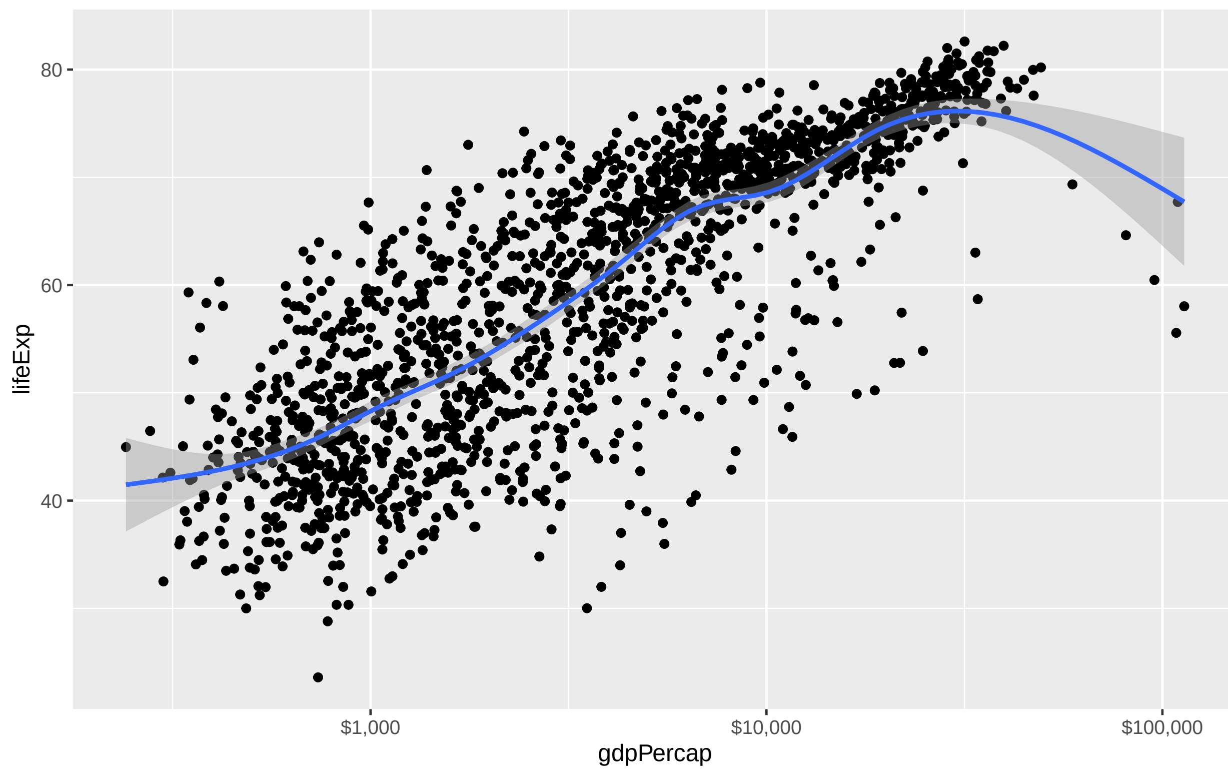


Figure 7: Life Expectancy vs GDP scatterplot, with a GAM smoother and a log scale on the x-axis, with better labels on the tick marks.

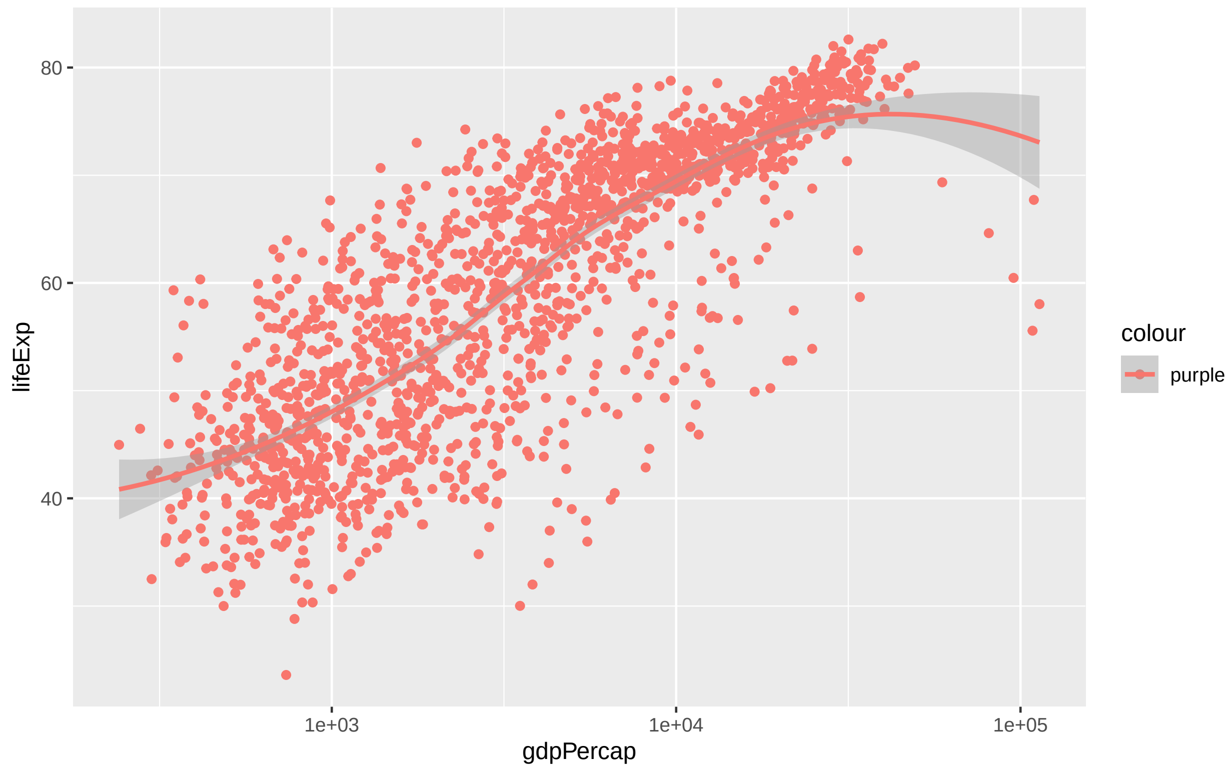


Figure 8: What has gone wrong here?

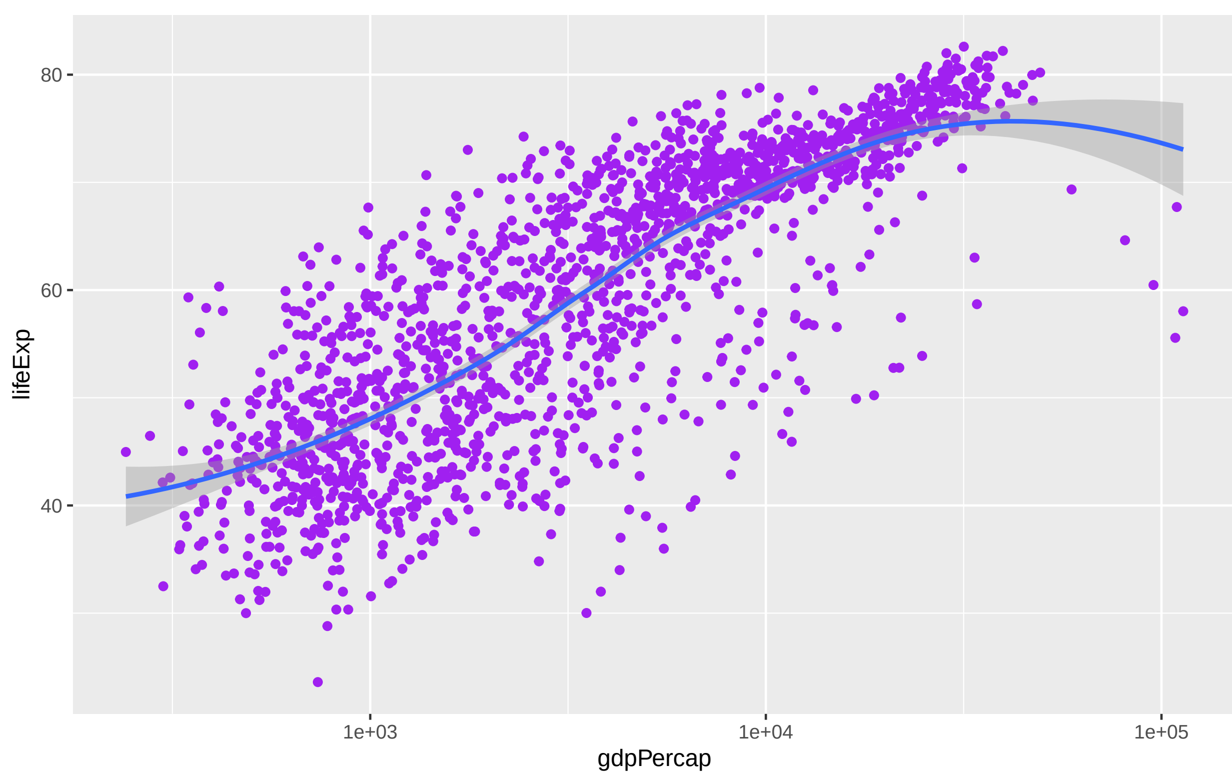


Figure 9: Setting the color attribute of the points directly.

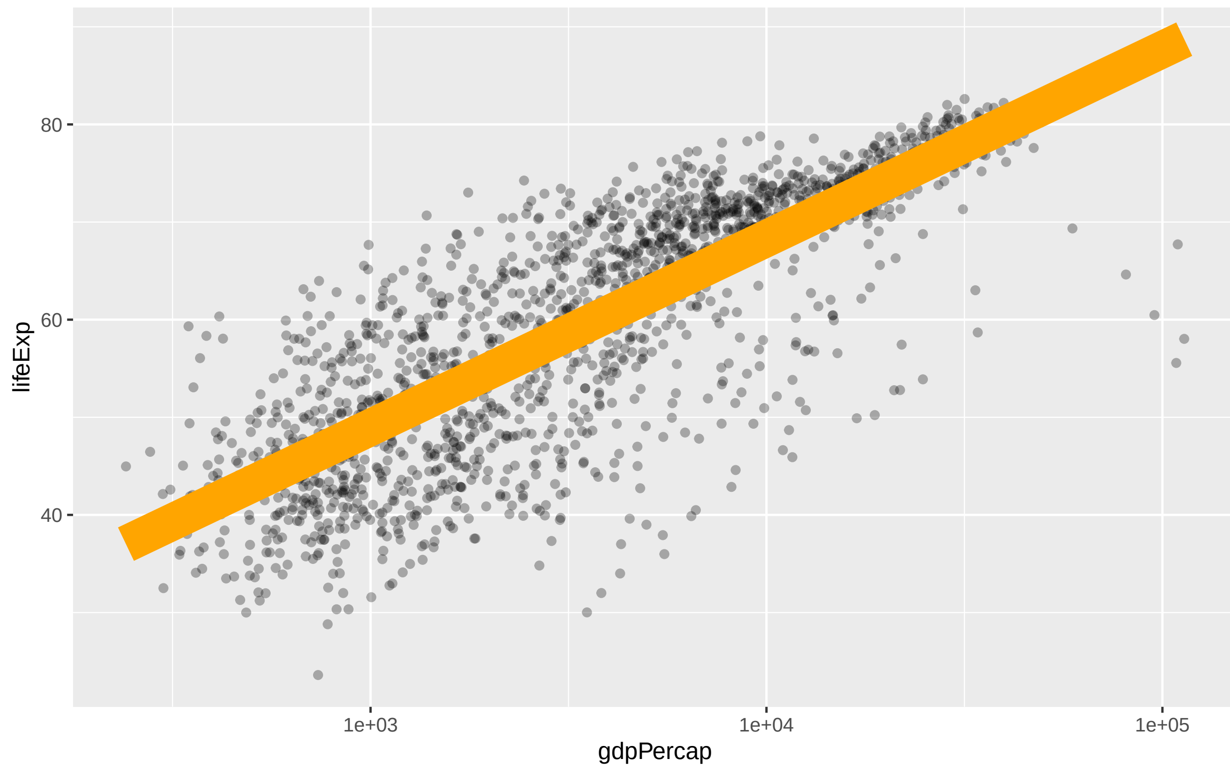


Figure 10: Setting some other arguments.

```
    labs(x = "GDP Per Capita", y = "Life Expectancy in Years",
         title = "Economic Growth and Life Expectancy",
         subtitle = "Data points are country-years",
         caption = "Source: Gapminder.")
```

```
## `geom_smooth()` using formula 'y ~ s(x, bs = "cs")'
```

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

```
## `geom_smooth()` using formula 'y ~ x'
```

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

```
## `geom_smooth()` using formula 'y ~ x'
```

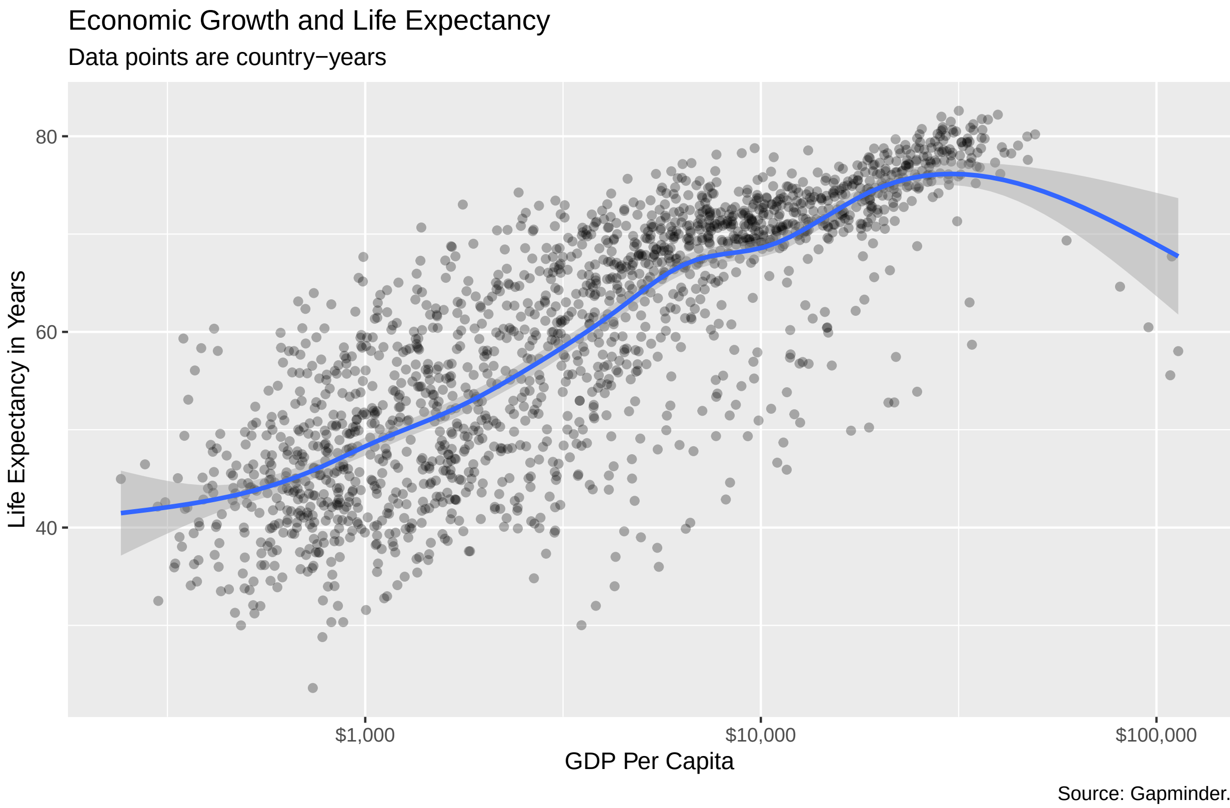


Figure 11: A more polished plot of Life Expectancy vs GDP.

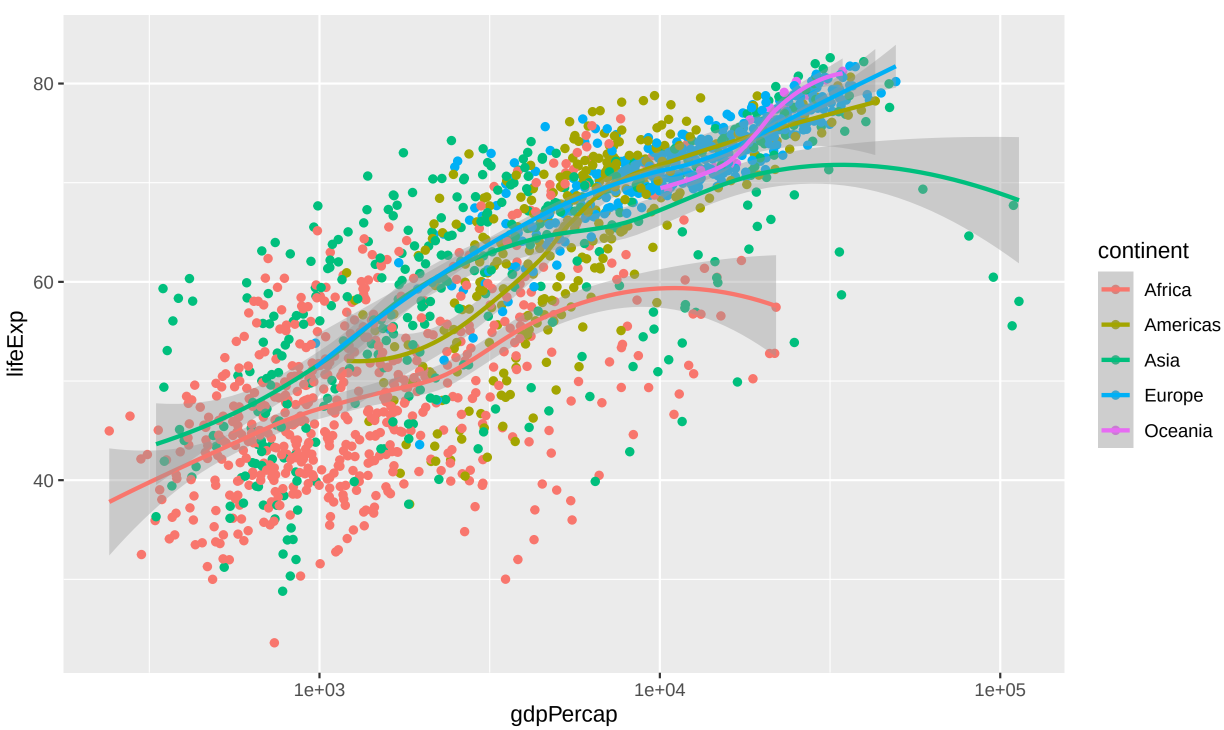


Figure 12: Mapping the continent variable to the color aesthetic.

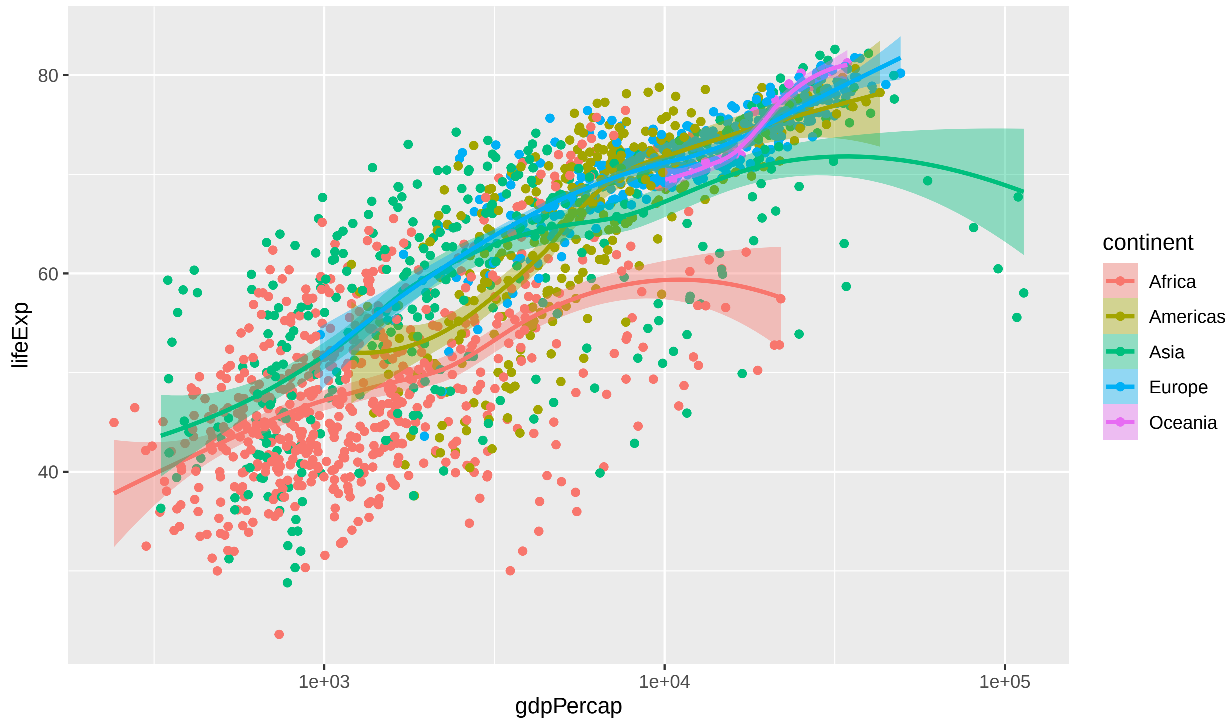


Figure 13: Mapping the continent variable to the color aesthetic, and correcting the error bars using the fill aesthetic.

Aesthetics can be mapped 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()
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdpPercap,
                          y = lifeExp))
p + geom_point(mapping = aes(color = log(pop))) +
    scale_x_log10()
```

Save your work

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

```
ggsave(filename = "Figure/figure1.png")
```

```
here()
```

```
## [1] "C:/Users/Nith Kosal/Documents/Kosal Documents/Teaching/DataVisualization"
```

```
p_out <- p + geom_point(mapping = aes(color = log(pop))) +
    scale_x_log10()
```

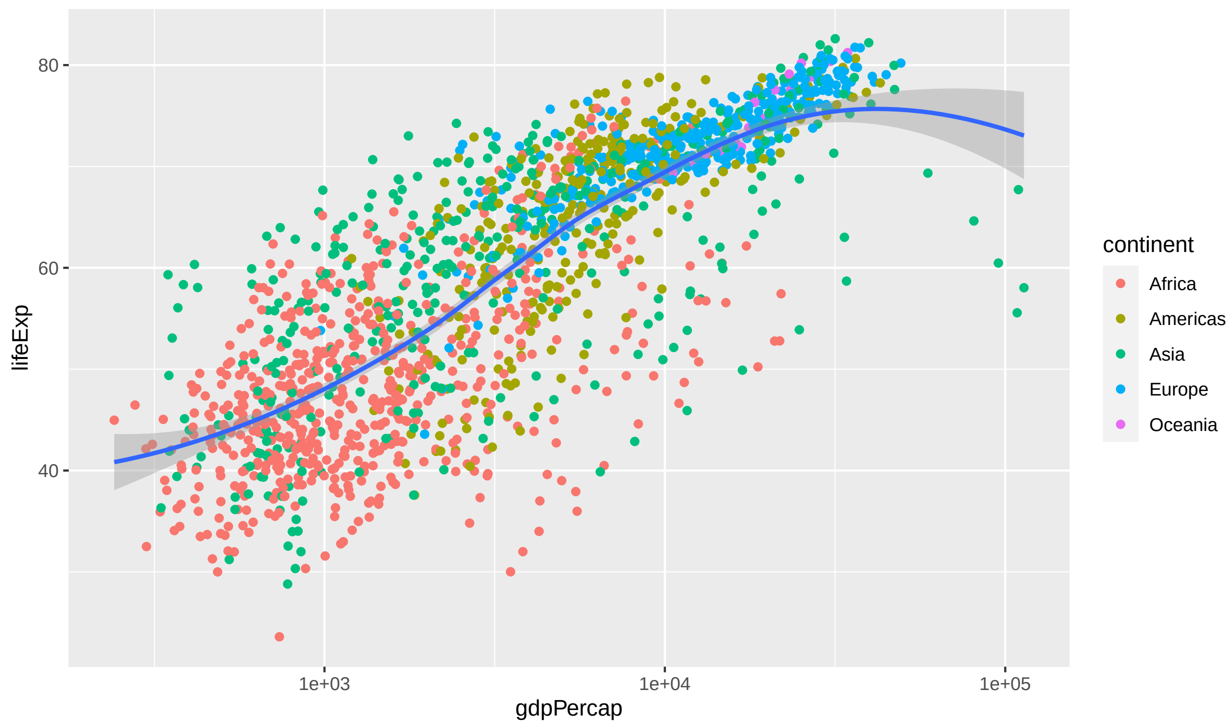


Figure 14: Mapping aesthetics on a per-geom basis. Here color is mapped to continent for the points but not the smoother.

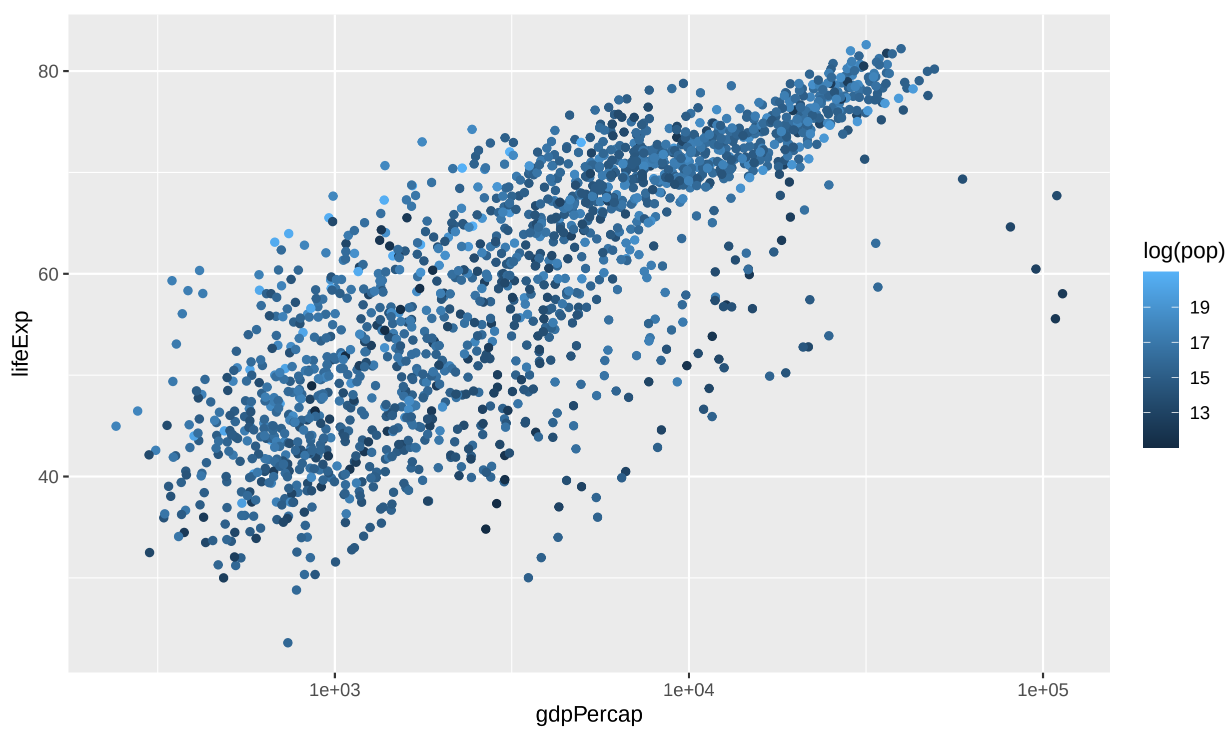
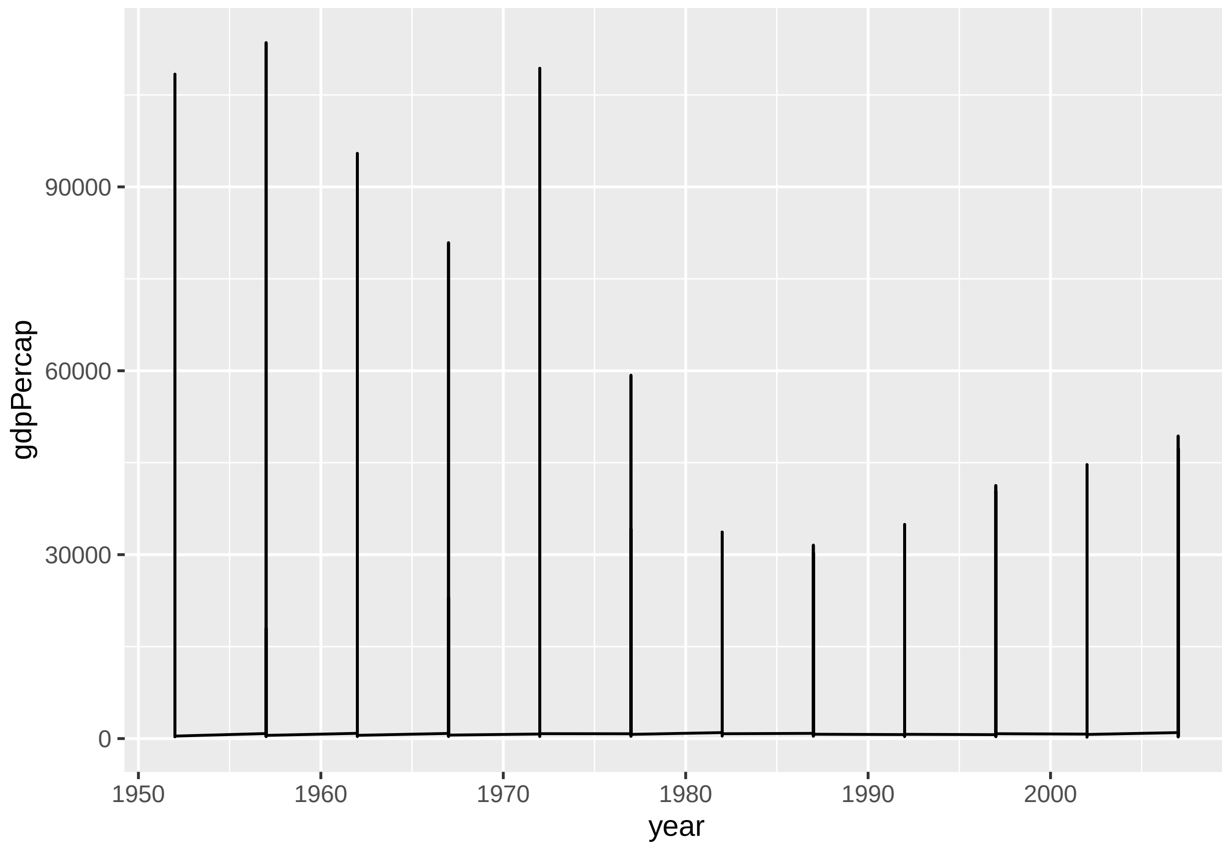


Figure 15: Mapping a continuous variable to color.

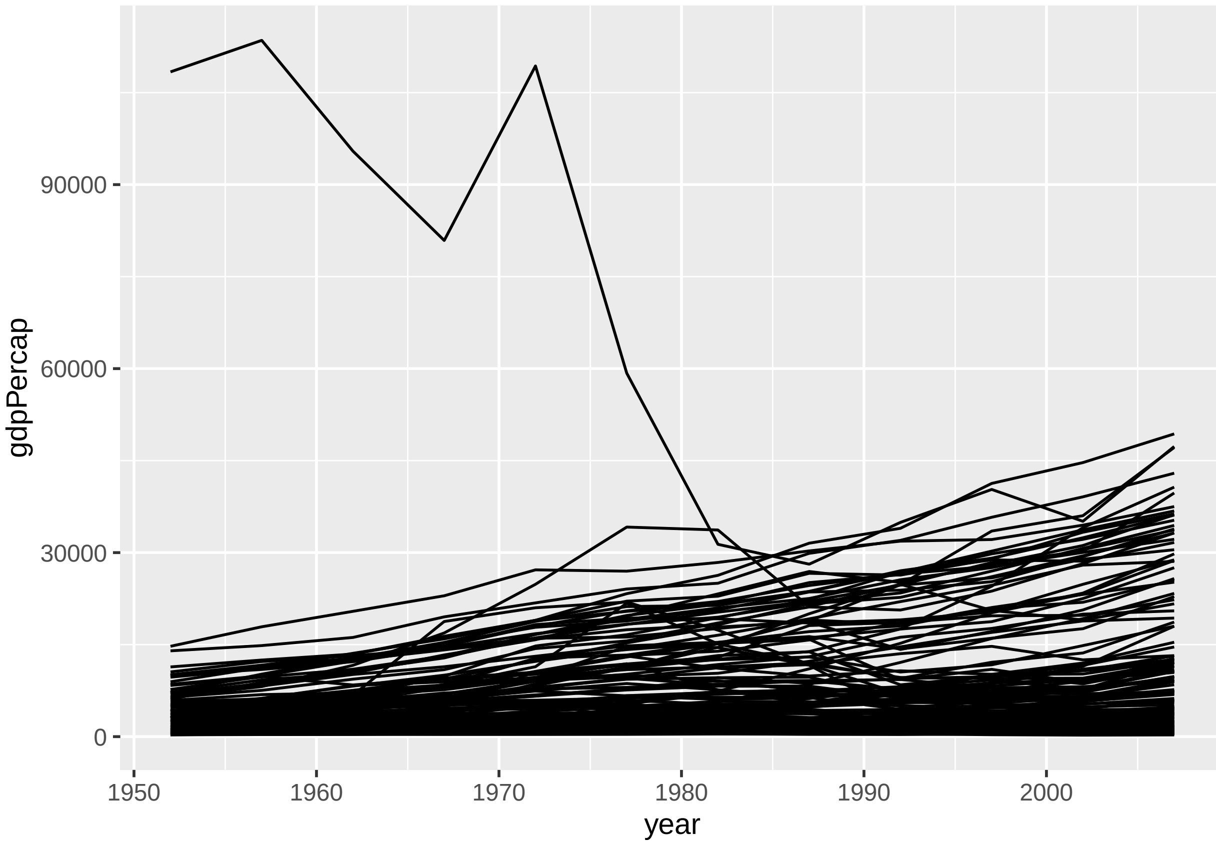
```
ggsave(here("Figure", "lifexp_vs_gdp_gradient.pdf"), plot = p_out)

ggsave(here("Figure", "lifexp_vs_gdp_gradient.png"), plot = p_out)
```

Grouped data and the group aesthetic

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

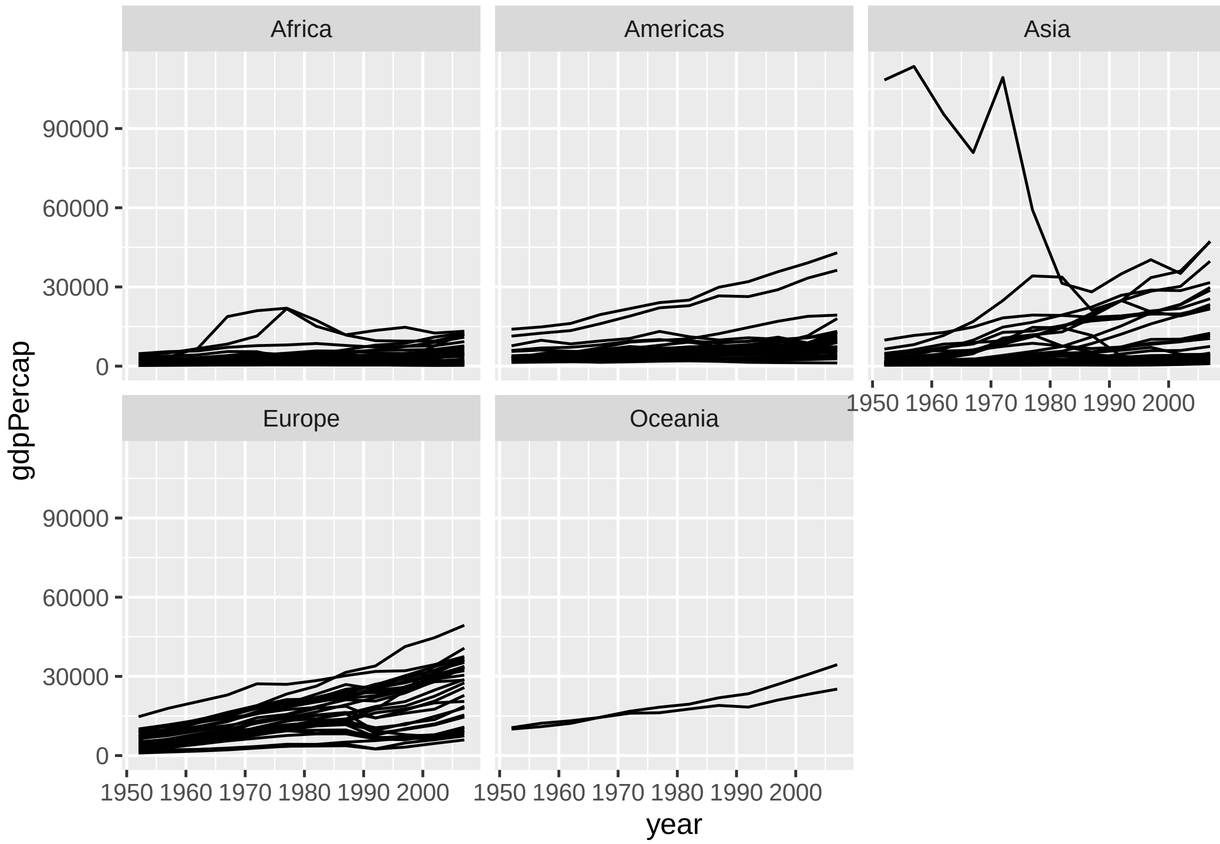



```
p <- ggplot(data = gapminder,
            mapping = aes(x = year,
                          y = gdpPercap))
p + geom_line(mapping =
                aes(group = country))
```



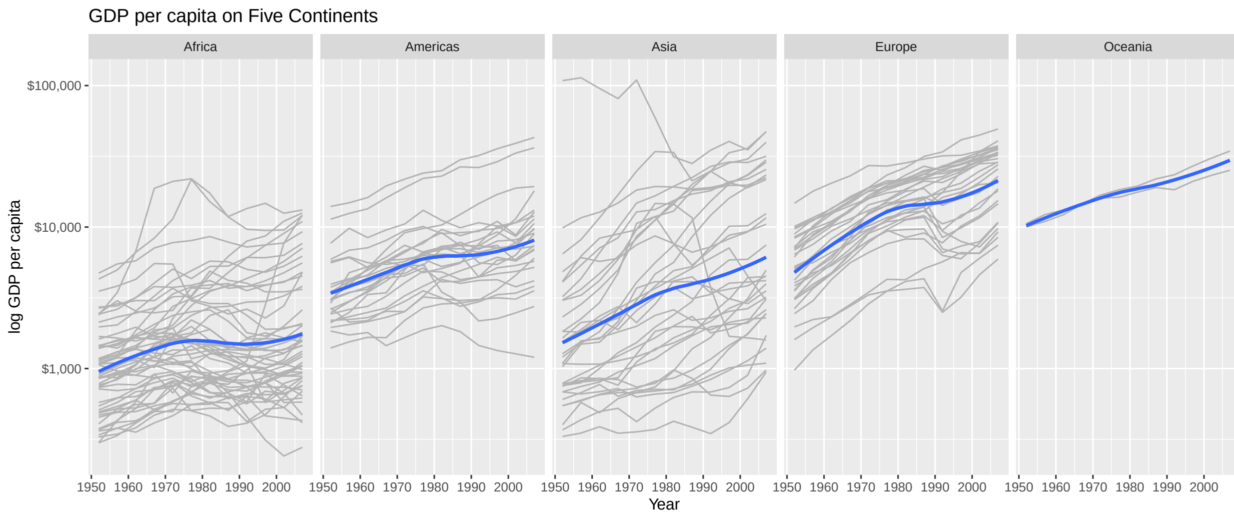
Faceting

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

p + geom_line(mapping =
              aes(group = country)) +
facet_wrap(~ continent)
```



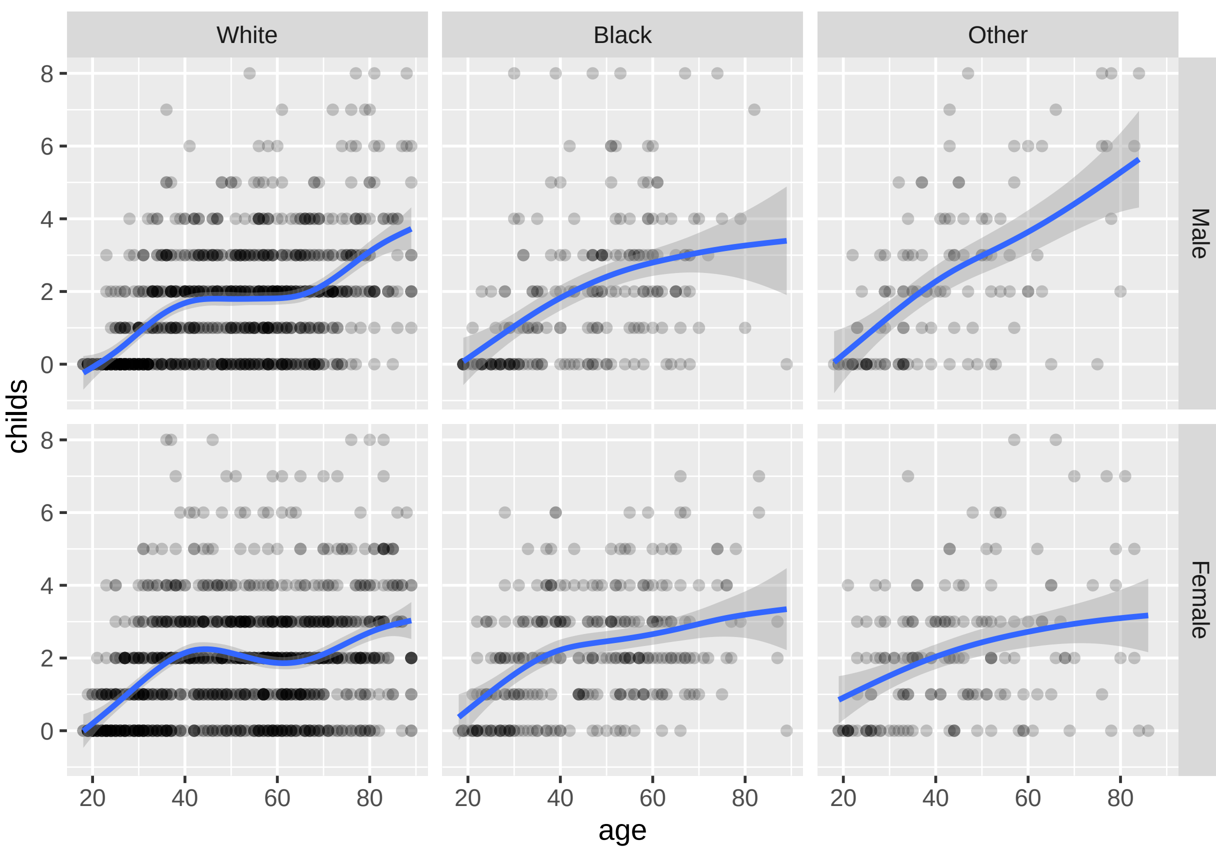

```
p + geom_line(color="gray70",
              mapping=aes(group = country)) +
    geom_smooth(size = 1.1,
                method = "loess",
                se = FALSE) +
    scale_y_log10(labels=scales::dollar) +
    facet_wrap(~ continent, ncol = 5) +
    labs(x = "Year",
         y = "log GDP per capita",
         title = "GDP per capita on Five Continents")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



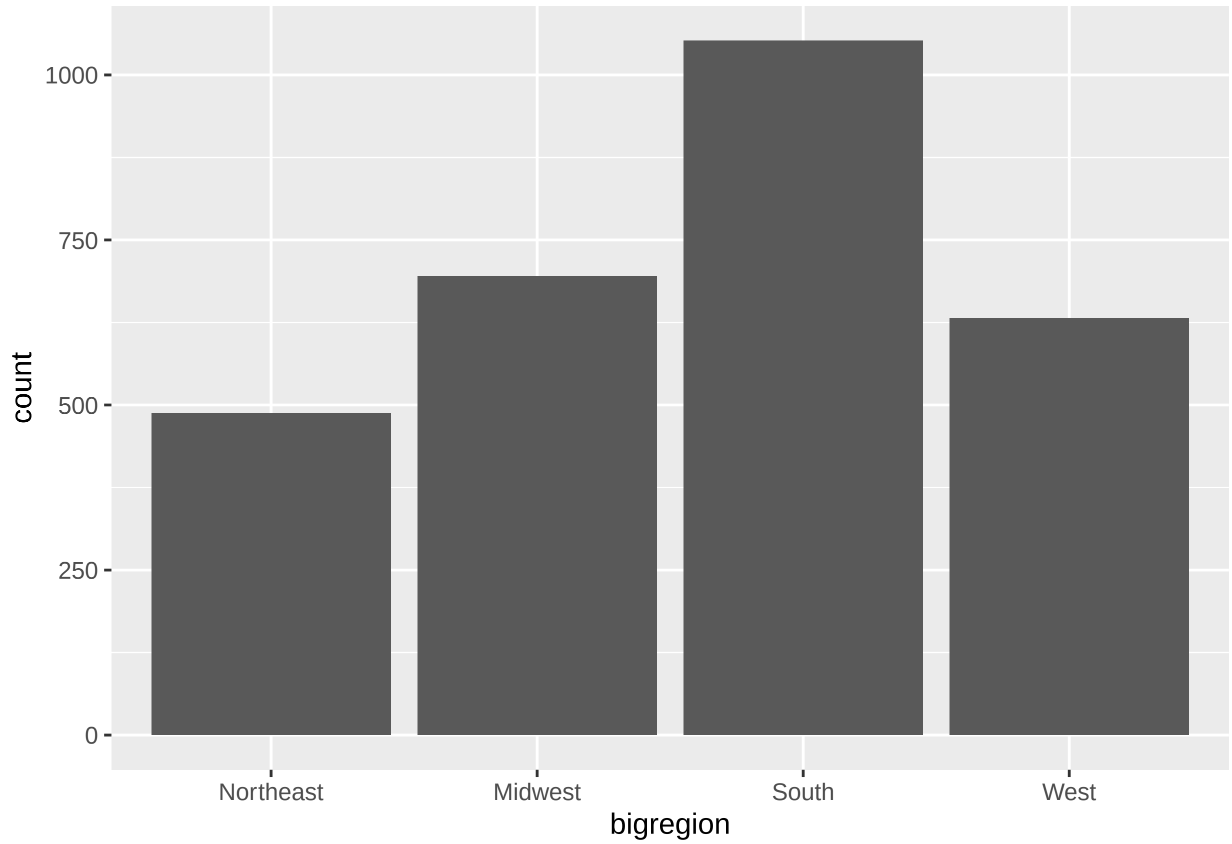

```
p <- ggplot(data = gss_sm,
            mapping = aes(x = age, y = childs))
p + geom_point(alpha = 0.2) + geom_smooth() +
    facet_grid(sex ~ race)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## Warning: Removed 18 rows containing non-finite values (stat_smooth).
## Warning: Removed 18 rows containing missing values (geom_point).
```

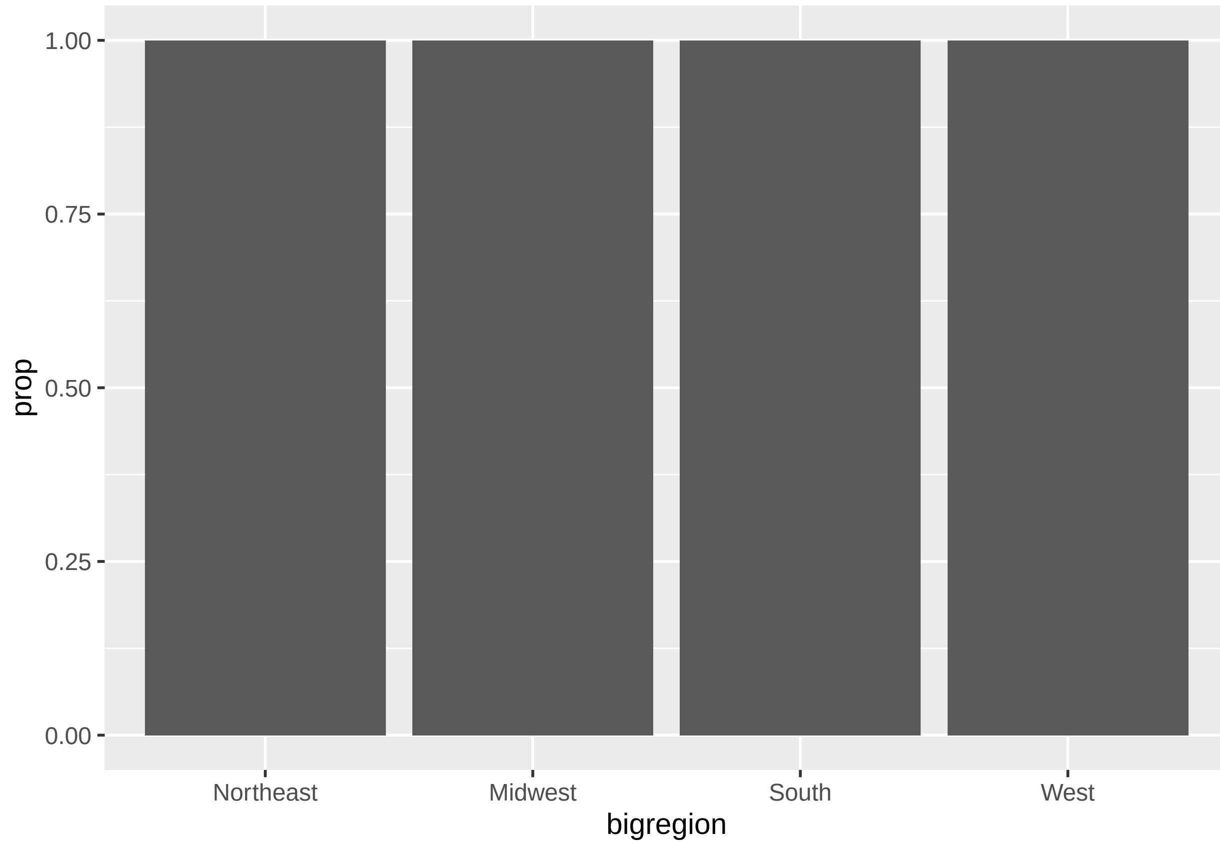


Geoms can transform data

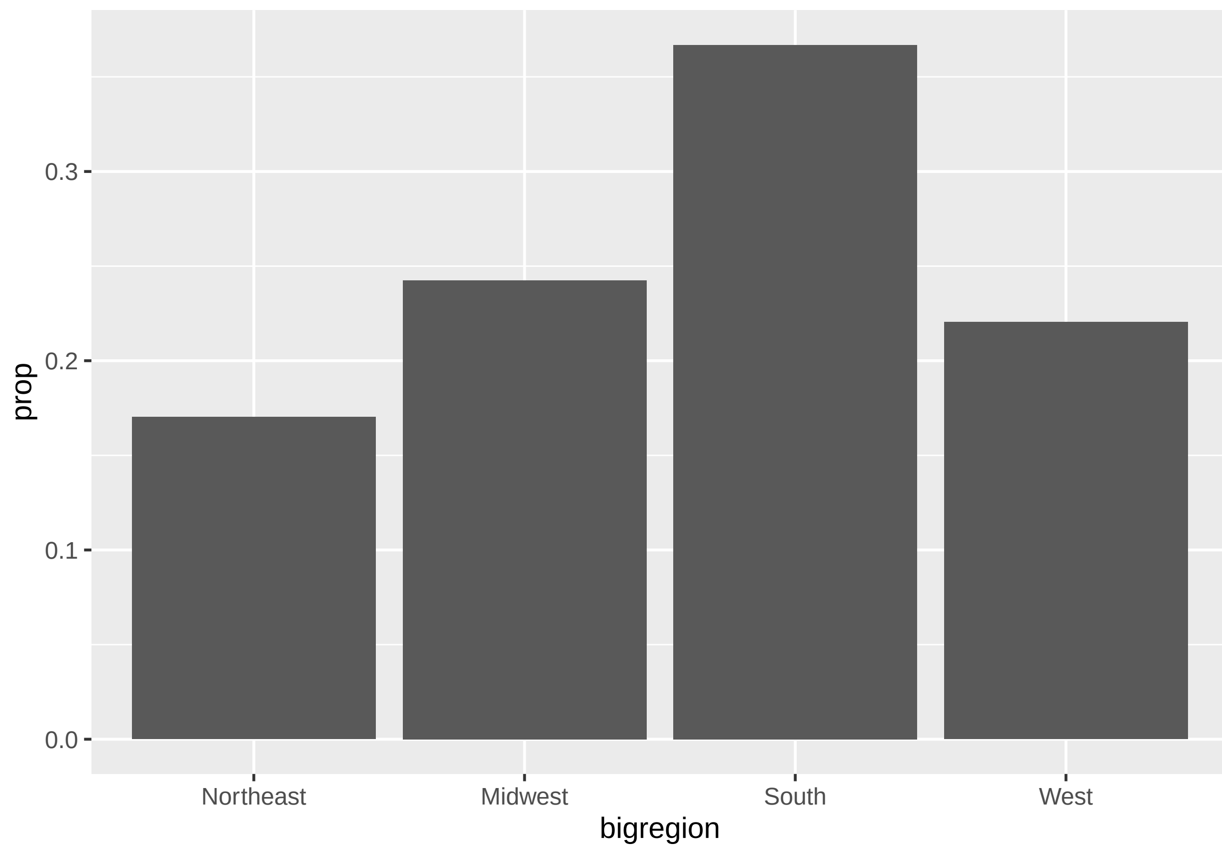
```
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion))
p + geom_bar()
```




```
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion))
p + geom_bar(mapping = aes(y = ..prop..))
```




```
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion))
p + geom_bar(mapping = aes(y = ..prop.., group = 1))
```

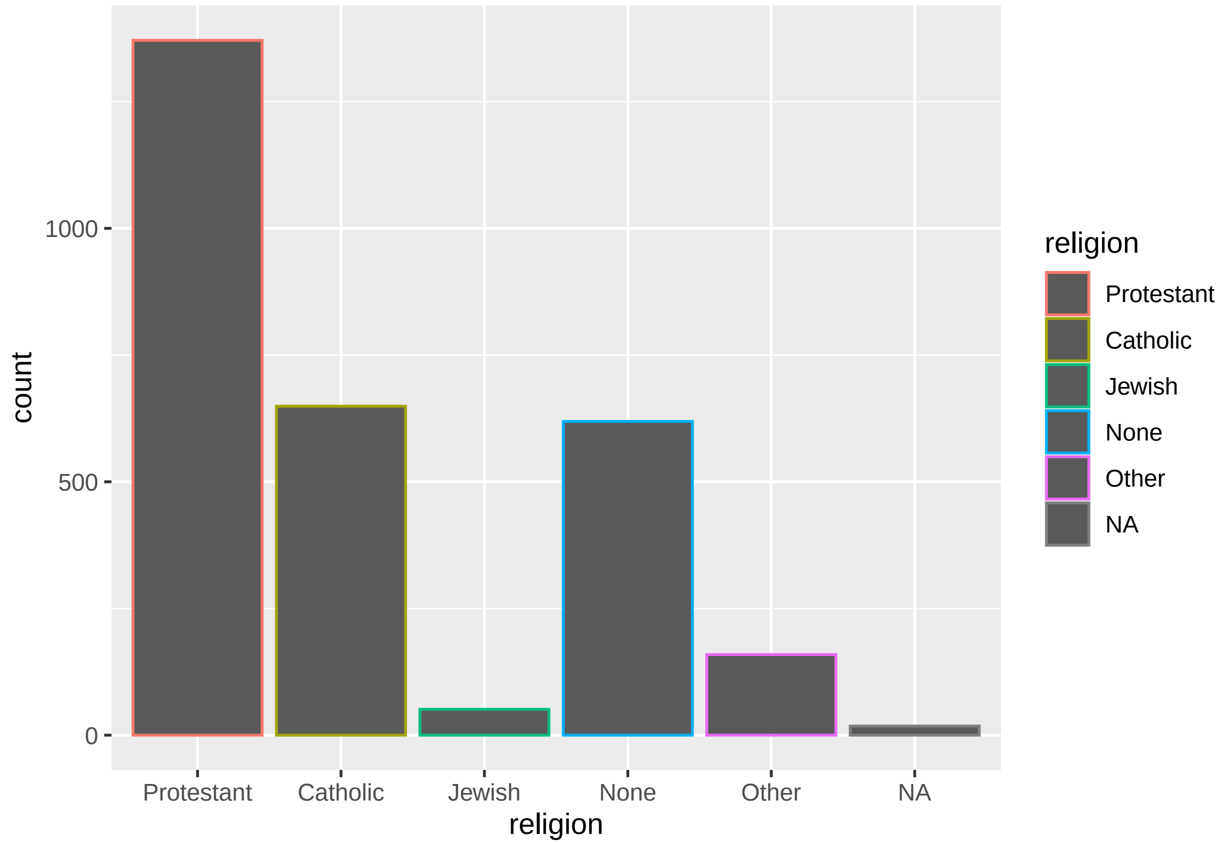


Frequency Plots the Slightly Awkward Way

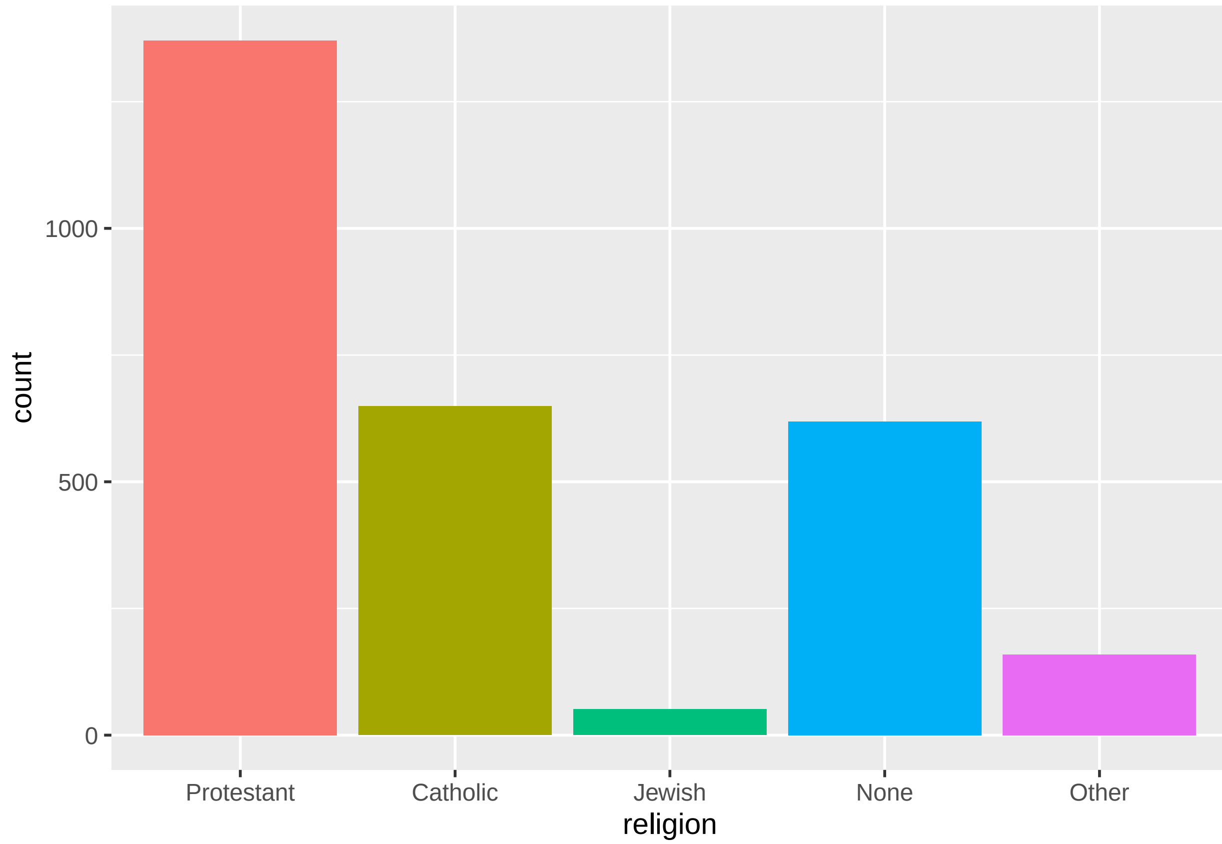
```
table(gss_sm$religion)
```

```
##
## Protestant    Catholic      Jewish        None       Other
##       1371         649          51         619         159
```

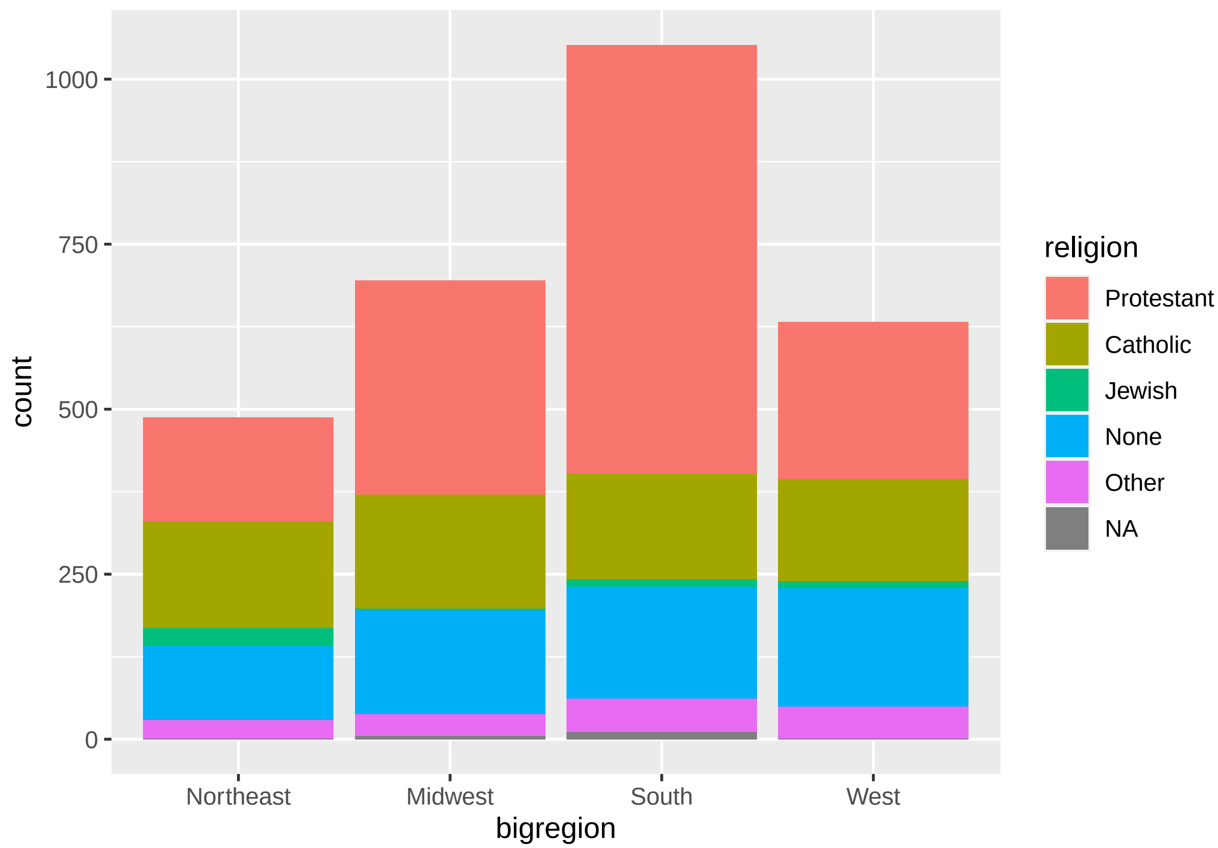
```
p <- ggplot(data = gss_sm,
            mapping = aes(x = religion, color = religion))
p + geom_bar()
```



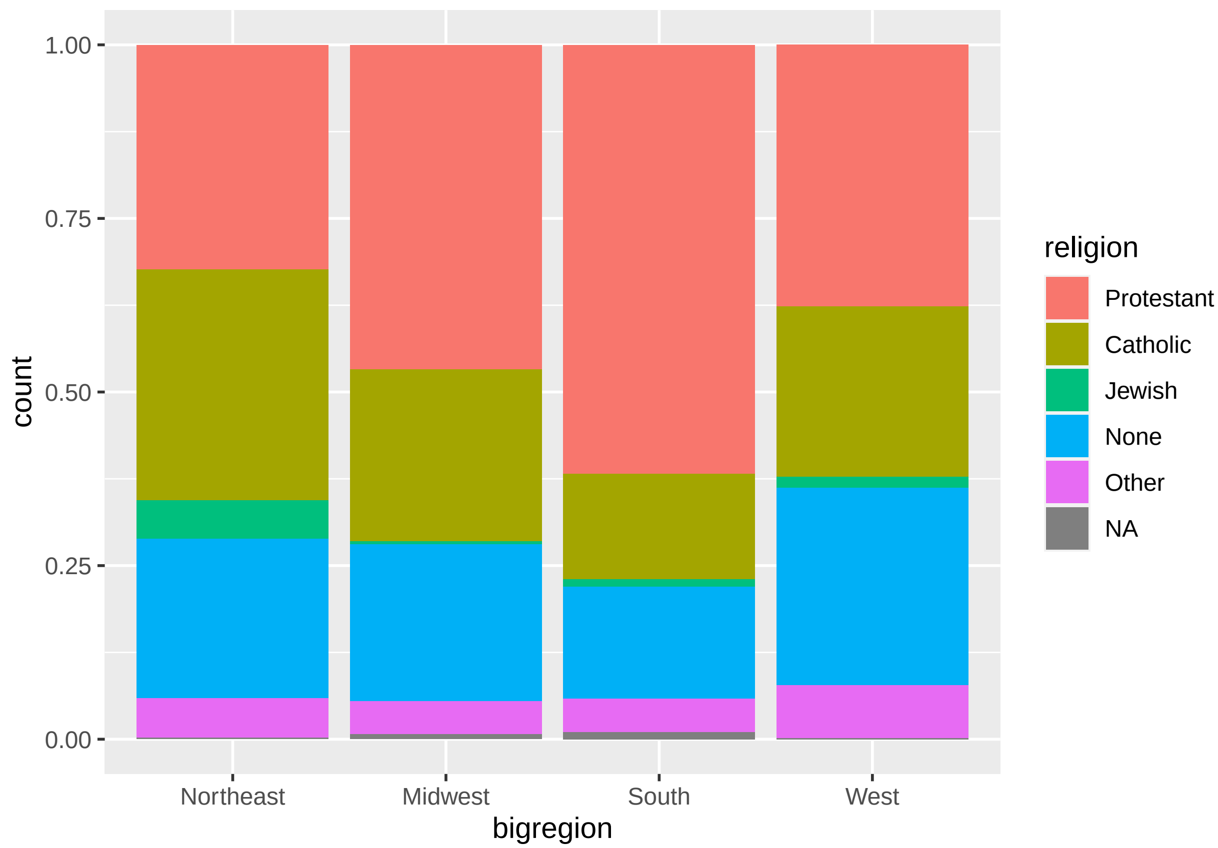

```r
dat.clean<- na.omit(subset(gss_sm, select = religion))
p <- ggplot(data = dat.clean,
            mapping = aes(x = religion, fill = religion))
p + geom_bar() + guides(fill = FALSE)
```



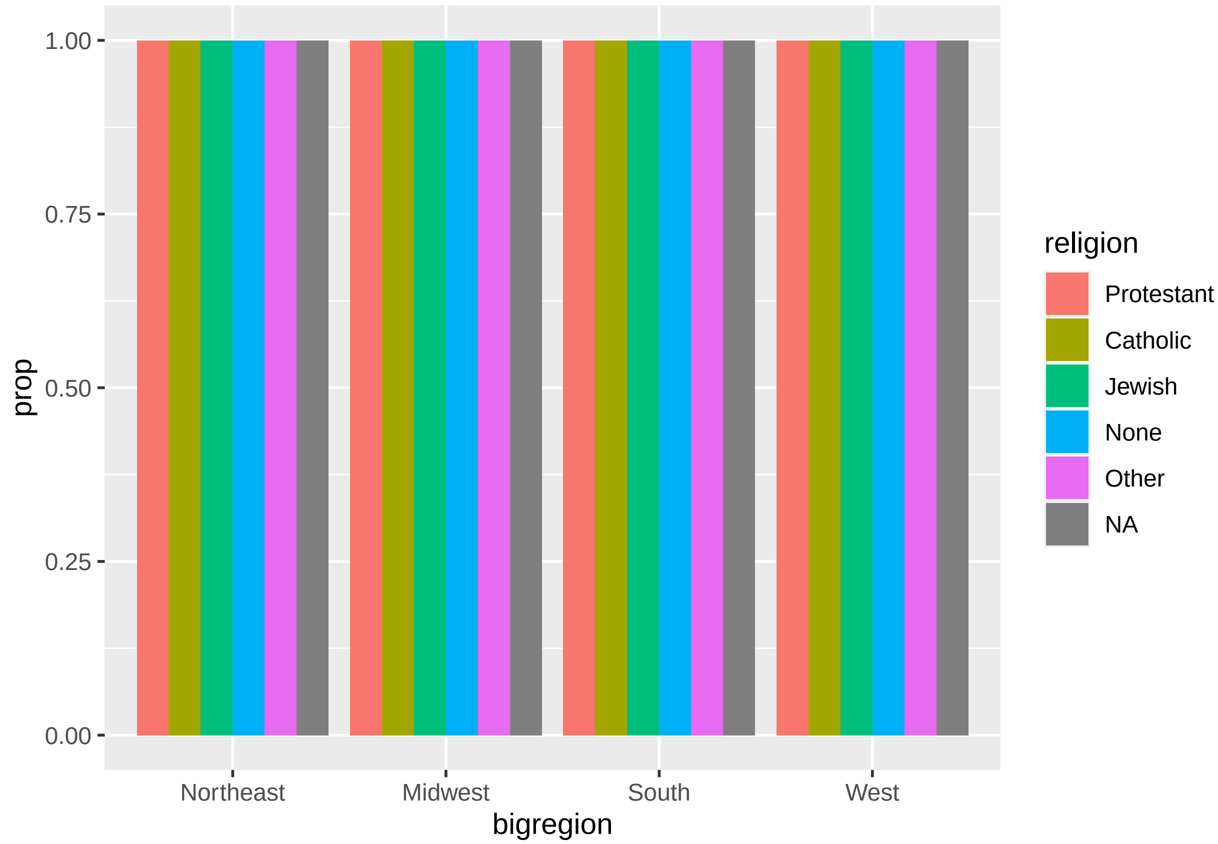

```
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion,
                          fill = religion))
p + geom_bar()
```



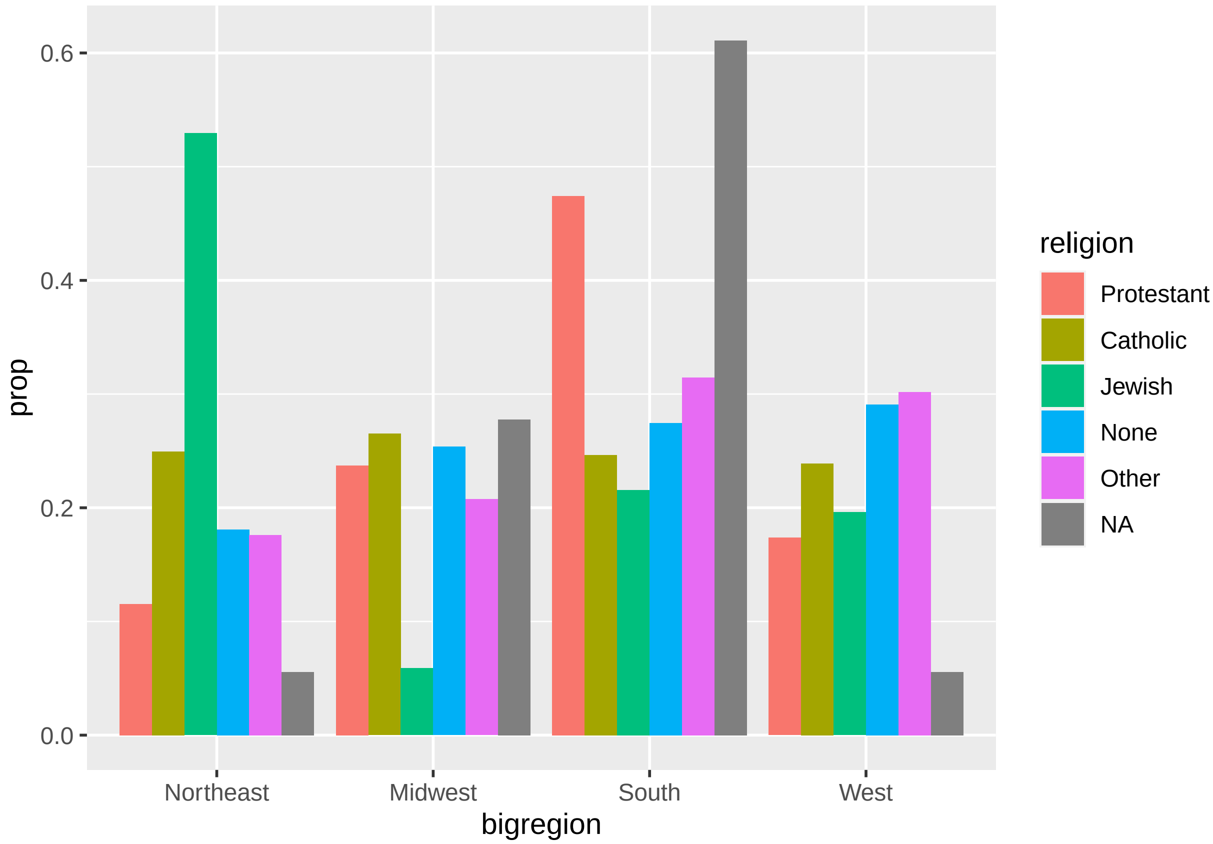

```r
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion,
                          fill = religion))
p + geom_bar(position = "fill")
```




```
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion,
                          fill = religion))
p + geom_bar(position = "dodge",
             mapping = aes(y = ..prop..))
```



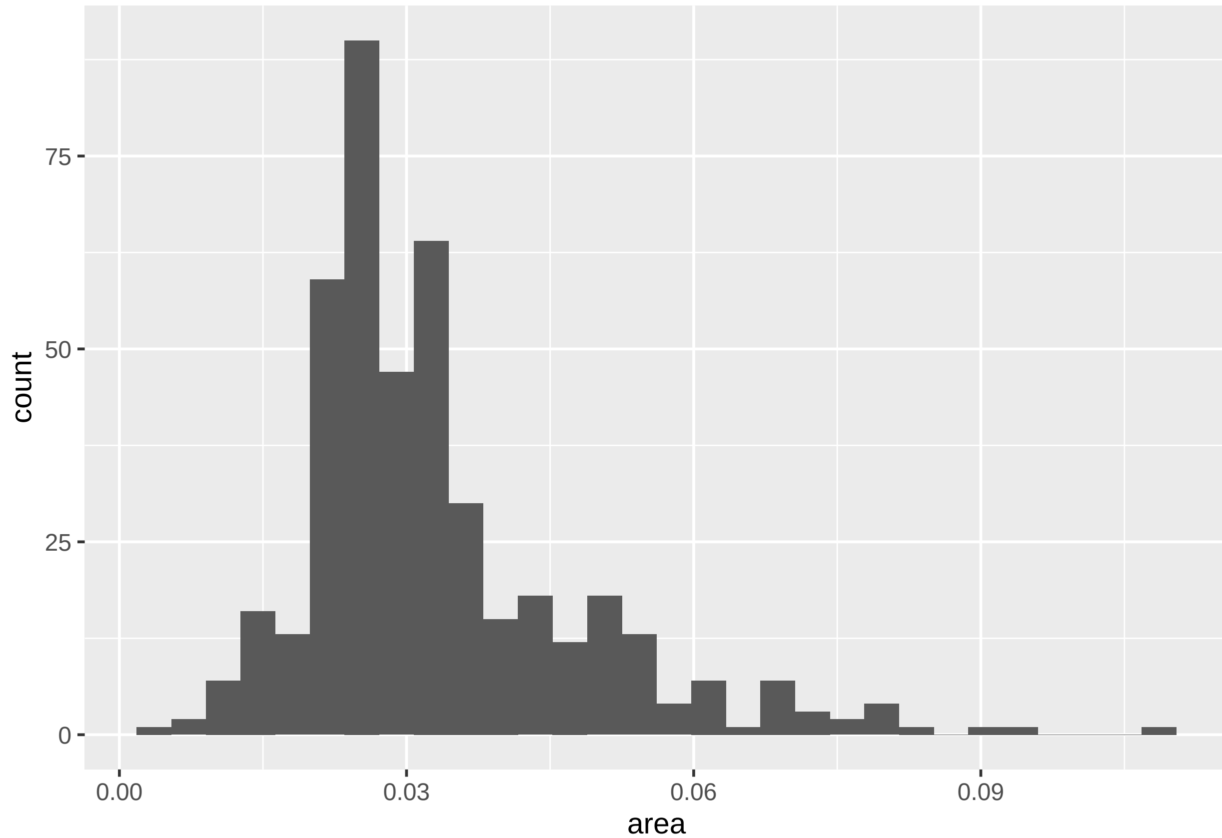

```
p <- ggplot(data = gss_sm,
            mapping = aes(x = bigregion,
                          fill = religion))
p + geom_bar(position = "dodge",
             mapping = aes(y = ..prop..,
                           group = religion))
```



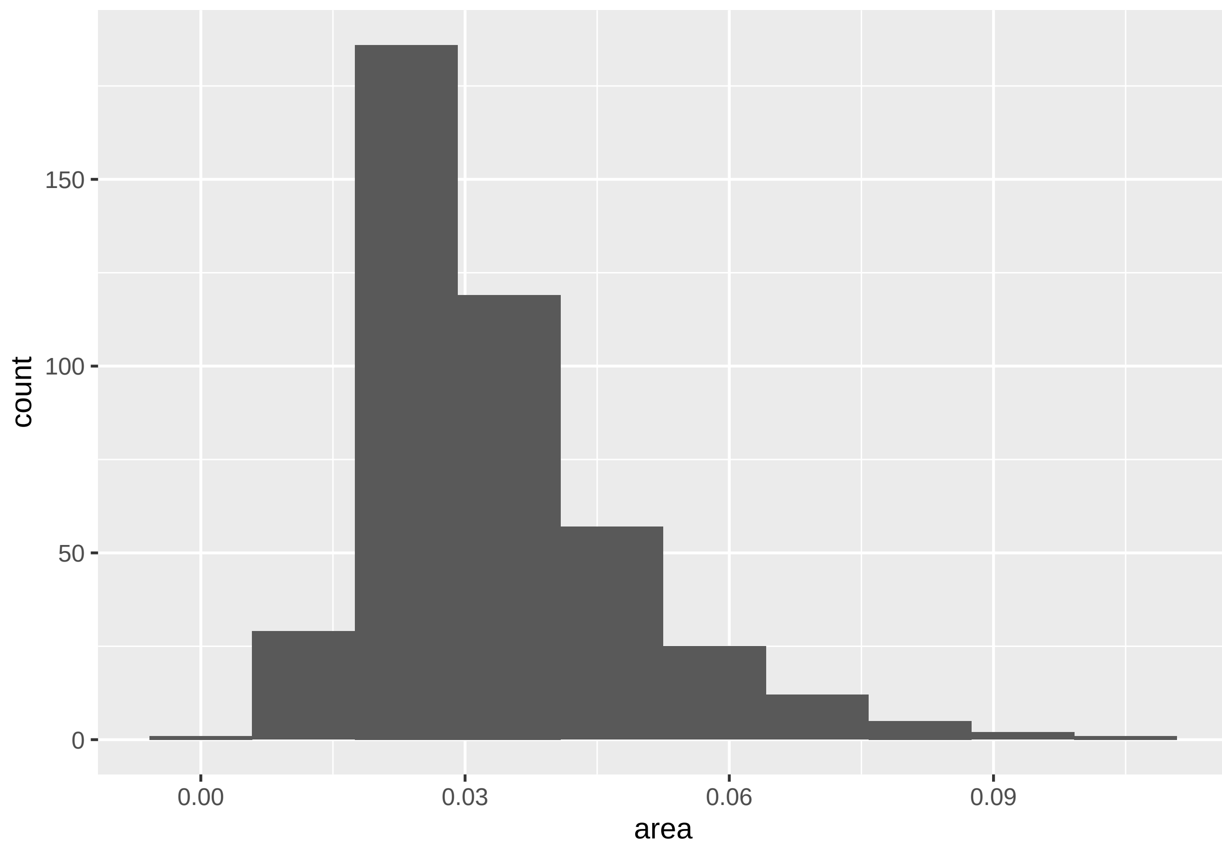
Histograms and Density Plots

```
p <- ggplot(data = midwest,
            mapping = aes(x = area))
p + geom_histogram()
```

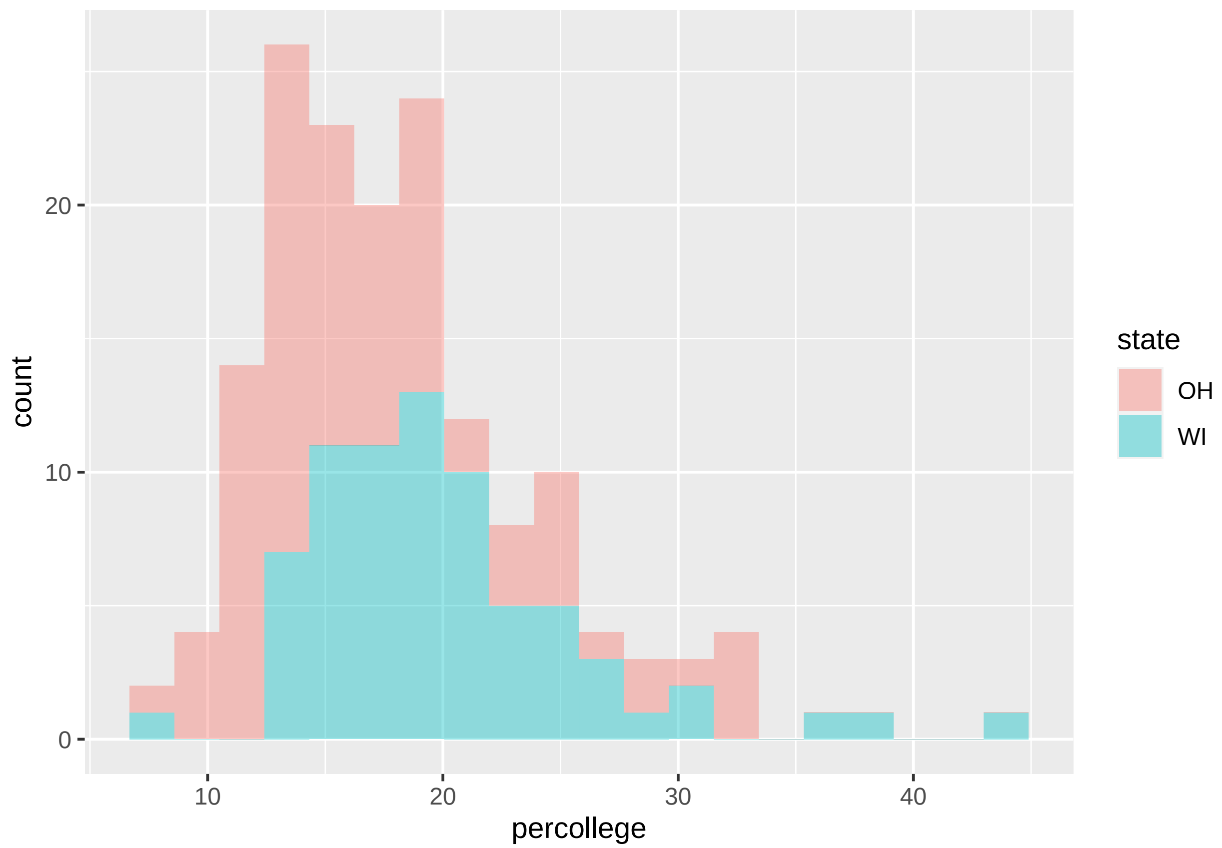
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



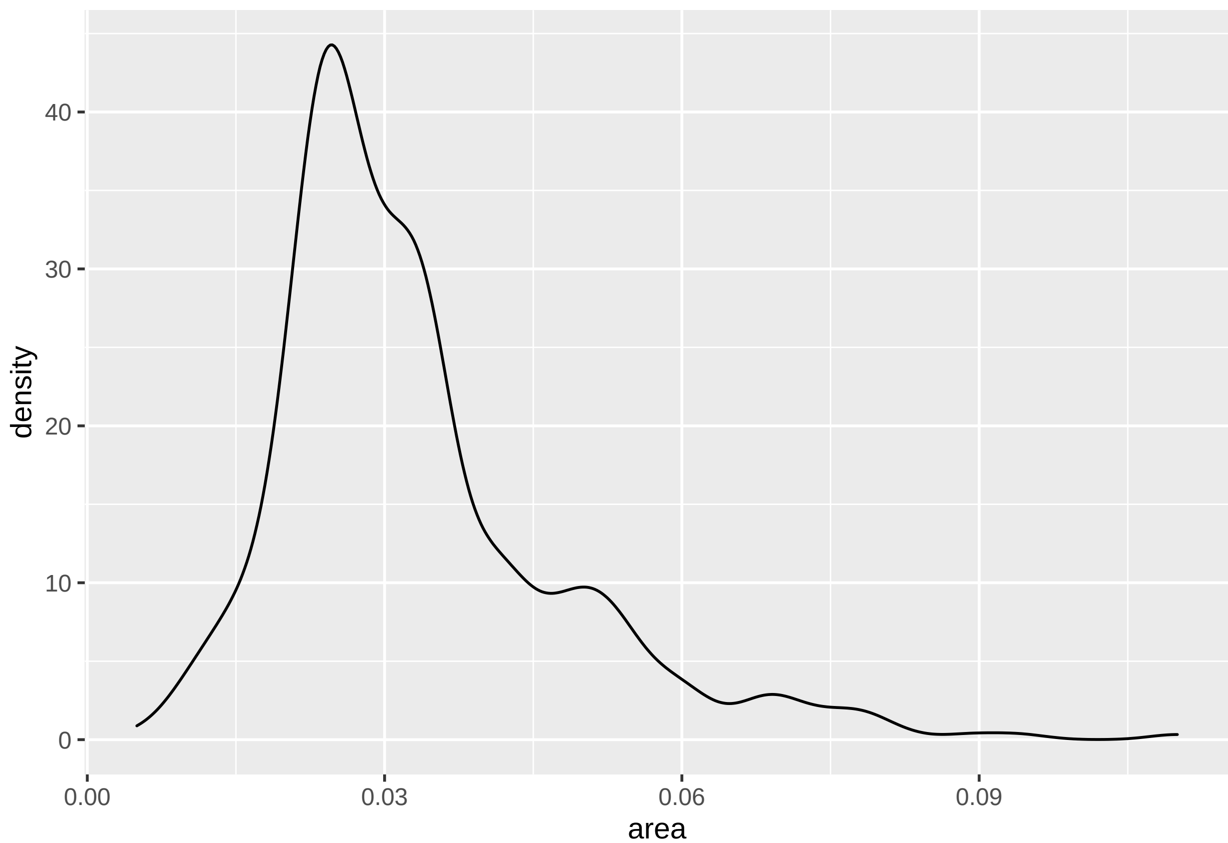

```
p <- ggplot(data = midwest,
            mapping = aes(x = area))
p + geom_histogram(bins = 10)
```



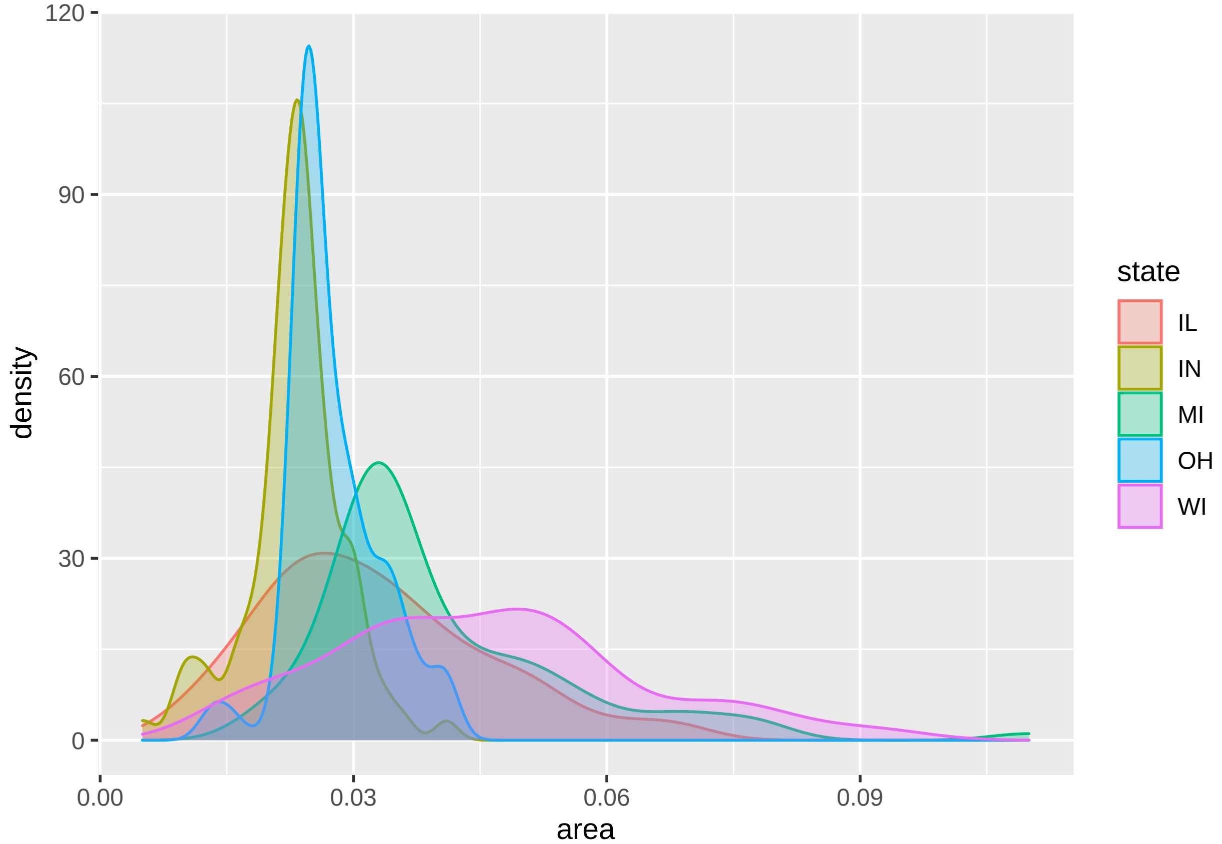

```
oh_wi <- c("OH", "WI")

p <- ggplot(data = subset(midwest, subset = state %in% oh_wi),
            mapping = aes(x = percollege, fill = state))
p + geom_histogram(alpha = 0.4, bins = 20)
```




```
p <- ggplot(data = midwest,
            mapping = aes(x = area))
p + geom_density()
```




```r
p <- ggplot(data = midwest,
            mapping = aes(x = area, fill = state, color = state))
p + geom_density(alpha = 0.3)
```

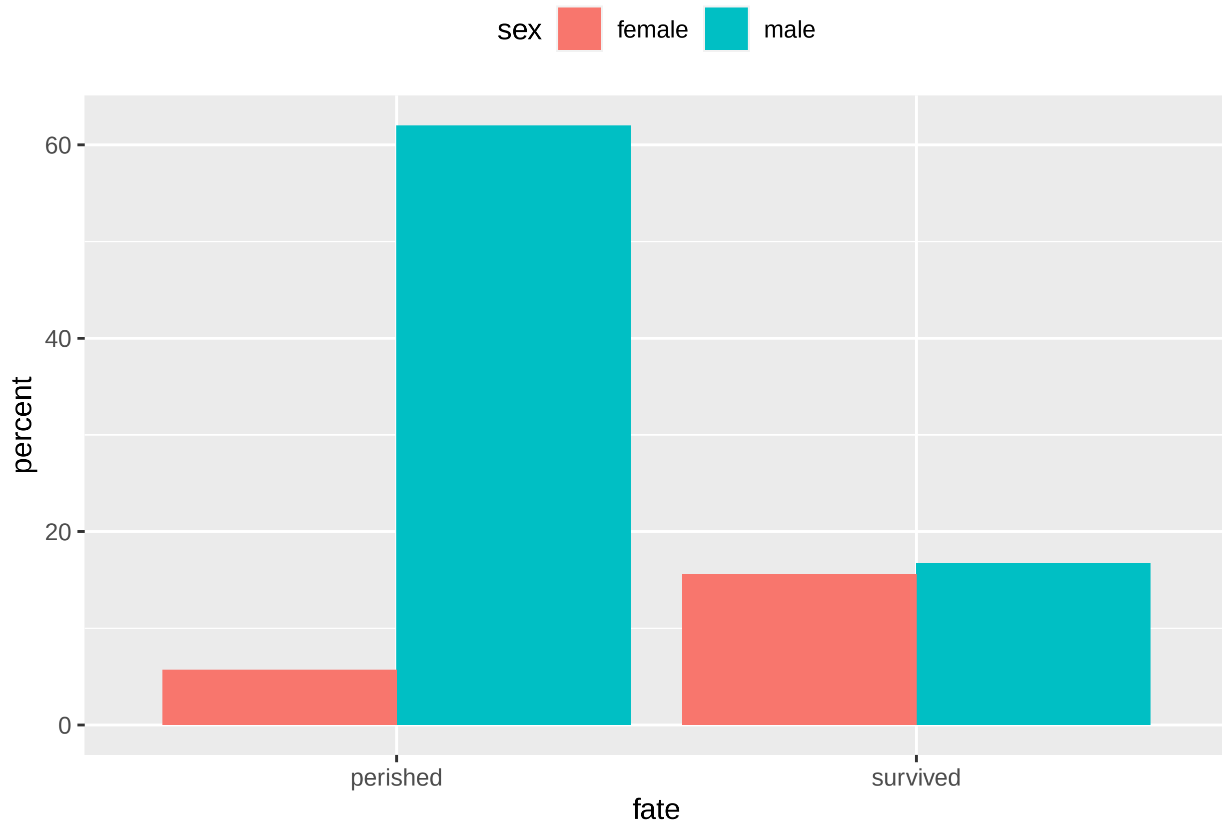


Avoid transformations when necessary

```
titanic
```

```
##       fate    sex    n percent
## 1 perished   male 1364    62.0
## 2 perished female  126     5.7
## 3 survived   male  367    16.7
## 4 survived female  344    15.6
```

```
p <- ggplot(data = titanic,
            mapping = aes(x = fate, y = percent, fill = sex))
p + geom_bar(position = "dodge", stat = "identity") + theme(legend.position = "top")
```

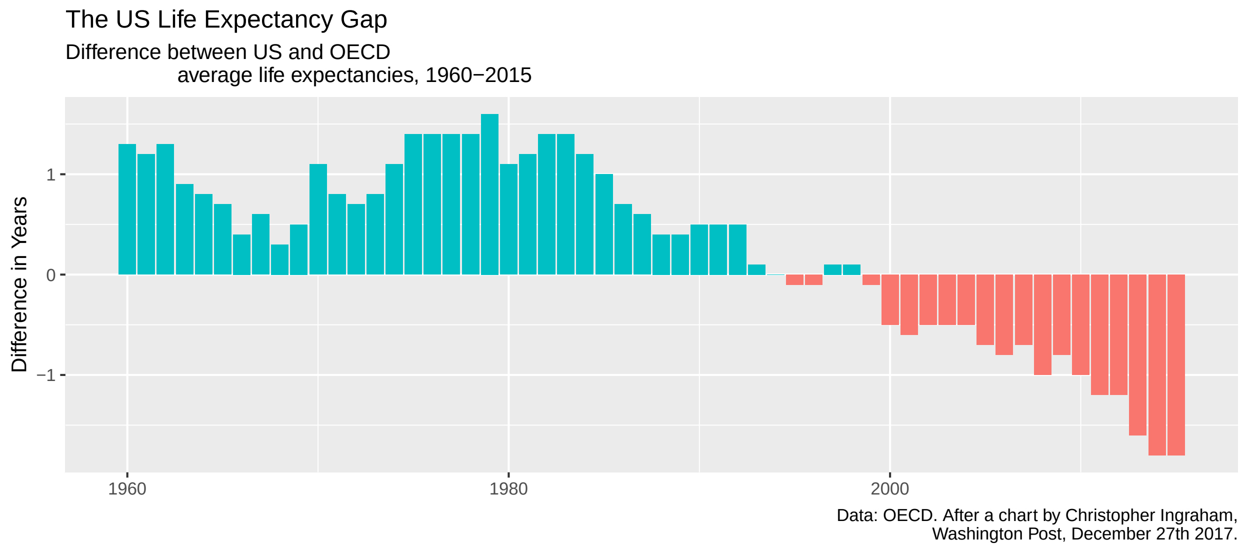



```
oecd_sum
```

```
## # A tibble: 57 x 5
## # Groups:   year [57]
##     year other   usa  diff hi_lo
##    <int> <dbl> <dbl> <dbl> <chr>
##  1  1960  68.6  69.9 1.30  Below
##  2  1961  69.2  70.4 1.20  Below
##  3  1962  68.9  70.2 1.30  Below
##  4  1963  69.1  70   0.900 Below
##  5  1964  69.5  70.3 0.800 Below
##  6  1965  69.6  70.3 0.700 Below
##  7  1966  69.9  70.3 0.400 Below
##  8  1967  70.1  70.7 0.600 Below
##  9  1968  70.1  70.4 0.300 Below
## 10  1969  70.1  70.6 0.5   Below
## # ... with 47 more rows
```

```
p <- ggplot(data = oecd_sum,
            mapping = aes(x = year, y = diff, fill = hi_lo))
p + geom_col() + guides(fill = FALSE) +
  labs(x = NULL, y = "Difference in Years",
       title = "The US Life Expectancy Gap",
       subtitle = "Difference between US and OECD
                   average life expectancies, 1960-2015",
       caption = "Data: OECD. After a chart by Christopher Ingraham,
                  Washington Post, December 27th 2017.")
```

```
## Warning: Removed 1 rows containing missing values (position_stack).
```



Dplyr pipelines

```
rel_by_region <- gss_sm %>%
    group_by(bigregion, religion) %>%
    summarize(N = n()) %>%
    mutate(freq = N / sum(N),
           pct = round((freq*100), 0))
```

```
## `summarise()` has grouped output by 'bigregion'. You can override using the `.groups` argument.
```

```
rel_by_region
```

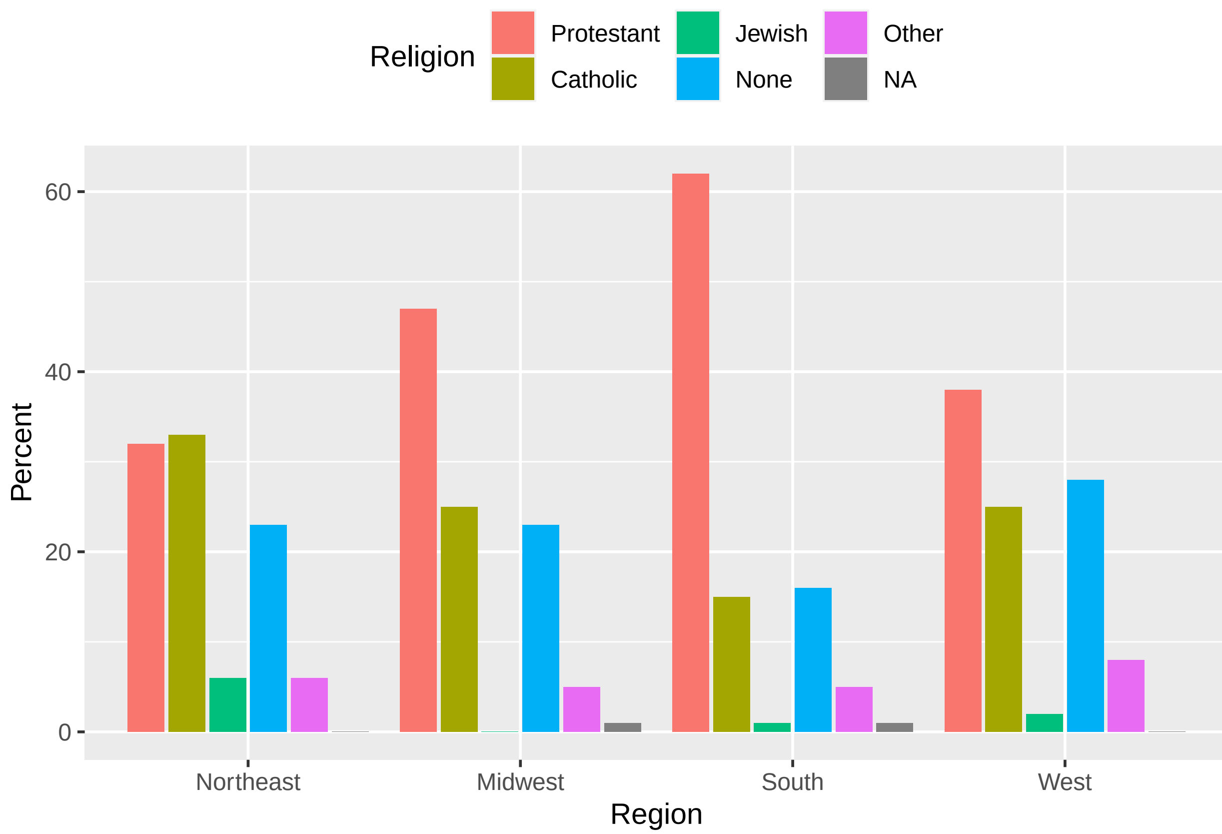
```
## # A tibble: 24 x 5
## # Groups:   bigregion [4]
##    bigregion religion        N    freq   pct
##    <fct>     <fct>       <int>   <dbl> <dbl>
##  1 Northeast Protestant    158 0.324      32
##  2 Northeast Catholic      162 0.332      33
##  3 Northeast Jewish         27 0.0553      6
##  4 Northeast None          112 0.230      23
##  5 Northeast Other          28 0.0574      6
##  6 Northeast <NA>            1 0.00205     0
##  7 Midwest   Protestant    325 0.468      47
##  8 Midwest   Catholic      172 0.247      25
##  9 Midwest   Jewish          3 0.00432     0
## 10 Midwest   None          157 0.226      23
## # ... with 14 more rows
```

```
rel_by_region %>% group_by(bigregion) %>%
    summarize(total = sum(pct))
```

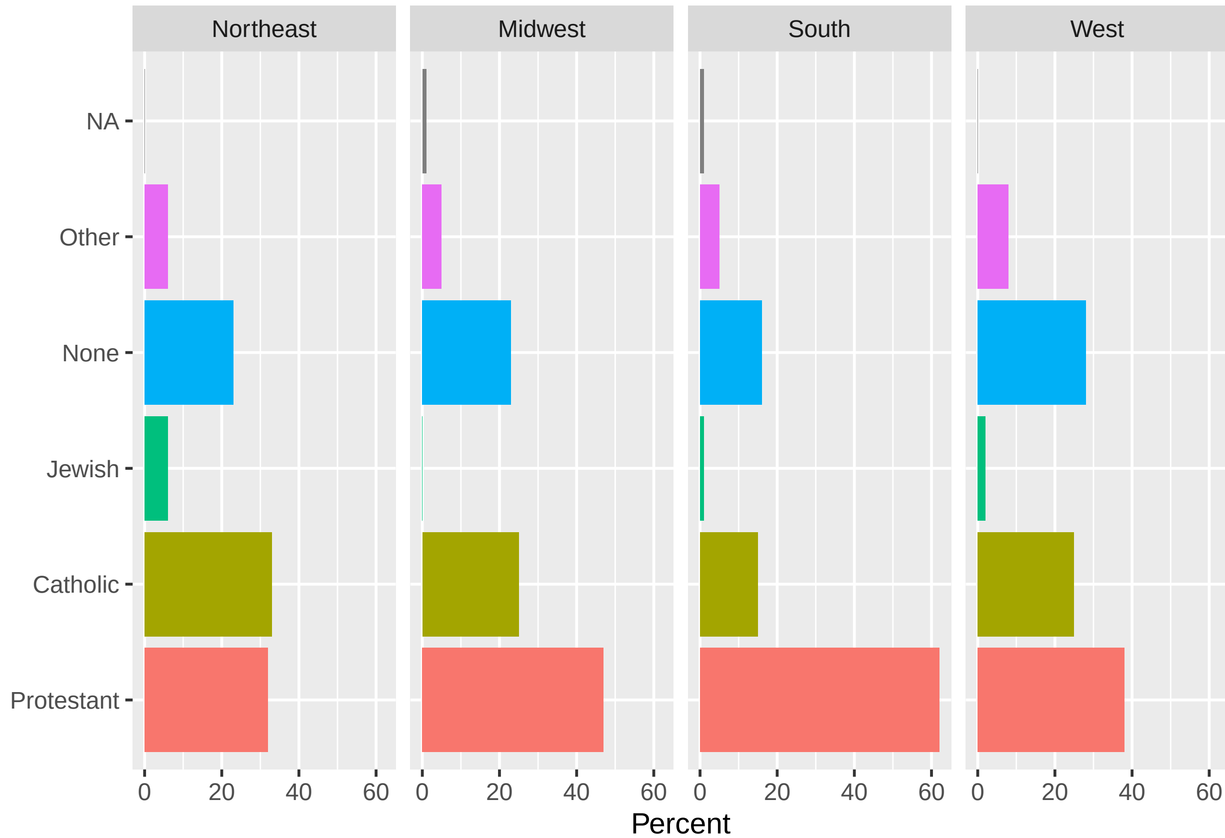
```
## # A tibble: 4 x 2
##   bigregion total
##   <fct>     <dbl>
## 1 Northeast   100
```

```
## 2 Midwest     101
## 3 South       100
## 4 West        101
```

```
## dodge2 presently requires the development version of ggplot
## devtools::install_github("tidyverse/ggplot2")

p <- ggplot(rel_by_region, aes(x = bigregion, y = pct, fill = religion))
p + geom_col(position = "dodge2") +
    labs(x = "Region",y = "Percent", fill = "Religion") +
    theme(legend.position = "top")
```




```
p <- ggplot(rel_by_region, aes(x = religion, y = pct, fill = religion))
p + geom_col(position = "dodge") +
    labs(x = NULL, y = "Percent", fill = "Religion") +
    guides(fill = FALSE) +
    coord_flip() +
    facet_grid(~ bigregion)
```



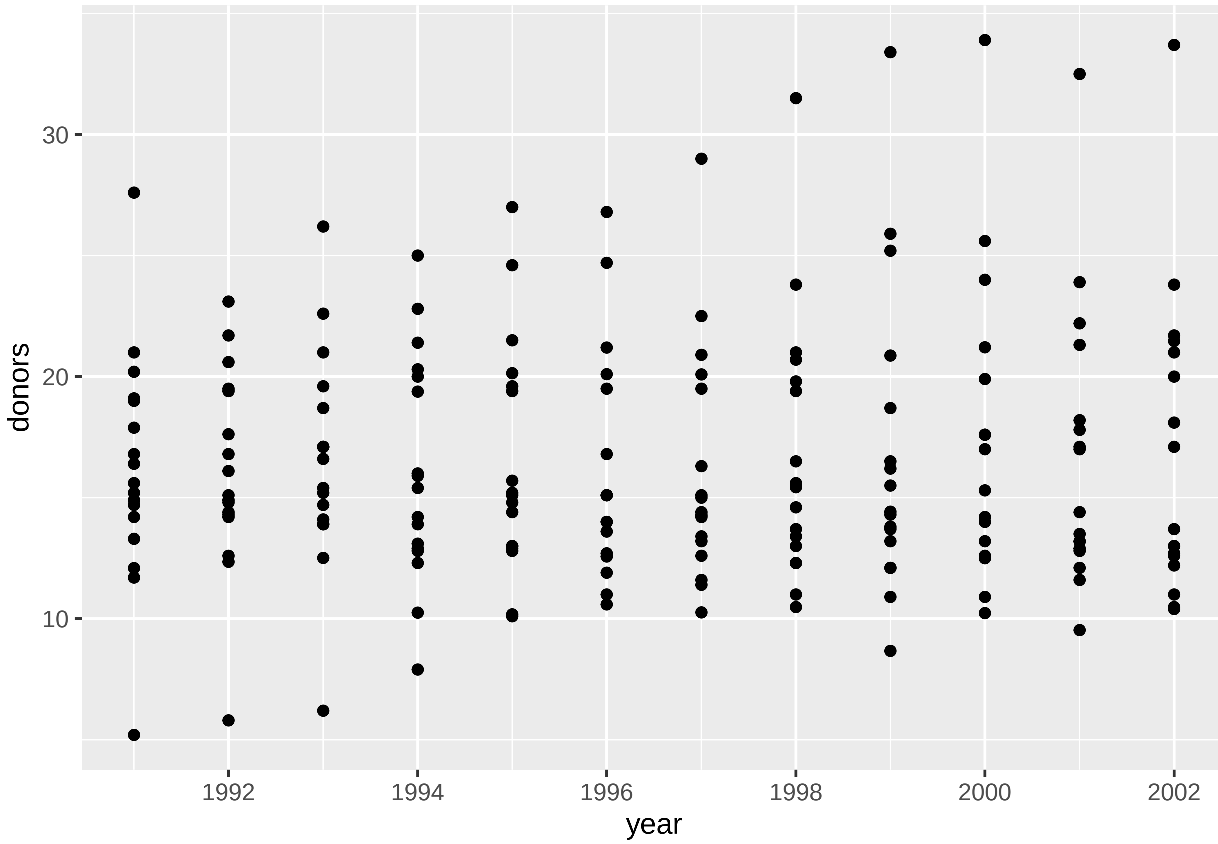
Continuous variables by category

```
organdata %>% select(1:6) %>% sample_n(size = 10)
```

```
## # A tibble: 10 x 6
##    country        year       donors    pop pop_dens   gdp
##    <chr>          <date>      <dbl>  <int>    <dbl> <int>
##  1 Germany        1997-01-01   13.2  82035   23.0   22589
##  2 Switzerland    1993-01-01   16.6   6938   16.8   25316
##  3 Australia      1993-01-01   12.5  17667    0.228 18883
##  4 Norway         1997-01-01   15.1   4405    1.36  27784
##  5 United States  1997-01-01   20.1 272647    2.83  30283
##  6 Switzerland    1997-01-01   14.3   7089   17.2   27675
##  7 Australia      NA           NA    17065    0.220 16774
##  8 United Kingdom 1995-01-01   14.4  58005   23.9   19998
##  9 United States  1991-01-01   17.9 252981    2.63  23443
## 10 United Kingdom 1997-01-01   13.4  58283   24.0   22442
```

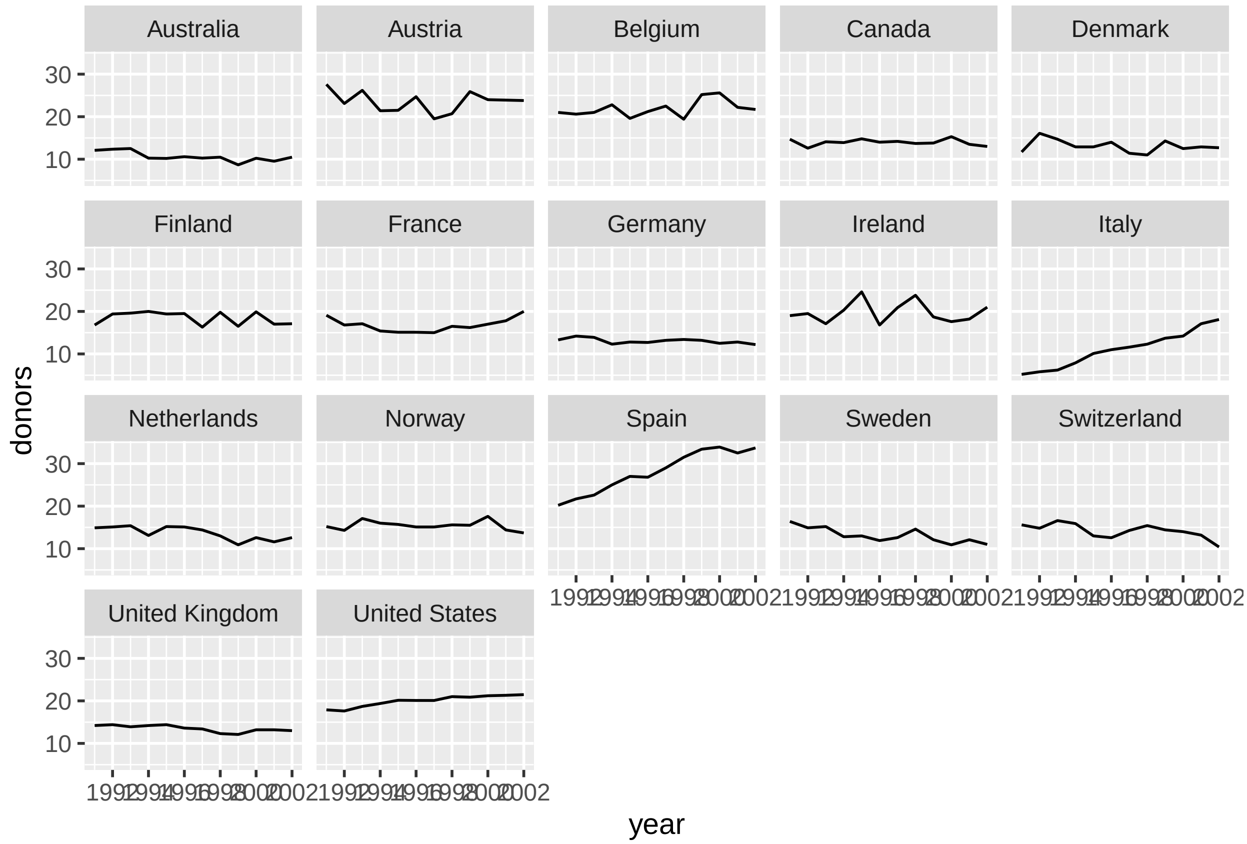
```
p <- ggplot(data = organdata,
            mapping = aes(x = year, y = donors))
p + geom_point()
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```



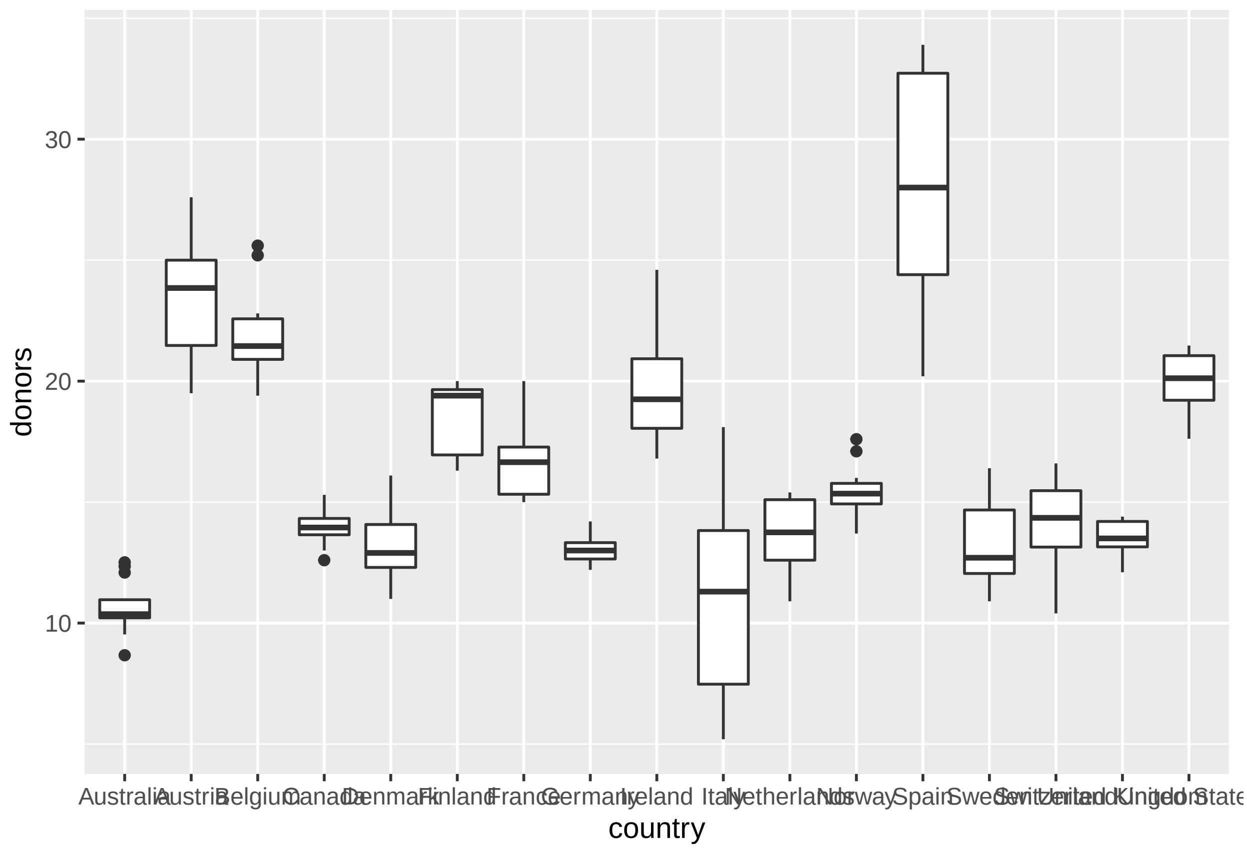

```
p <- ggplot(data = organdata,
            mapping = aes(x = year, y = donors))
p + geom_line(aes(group = country)) +
  facet_wrap(~ country)
```

```
## Warning: Removed 34 row(s) containing missing values (geom_path).
```



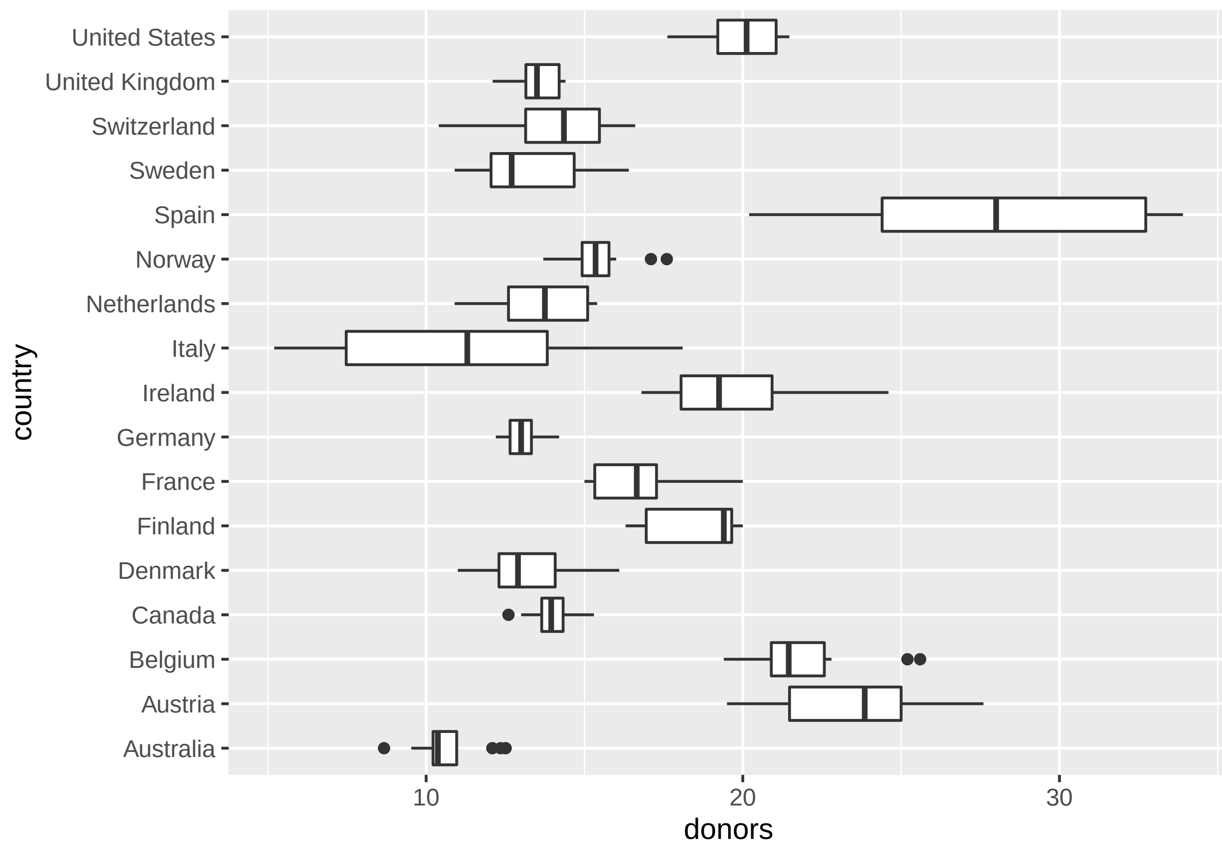

```r
p <- ggplot(data = organdata,
            mapping = aes(x = country, y = donors))
p + geom_boxplot()
```

```
## Warning: Removed 34 rows containing non-finite values (stat_boxplot).
```



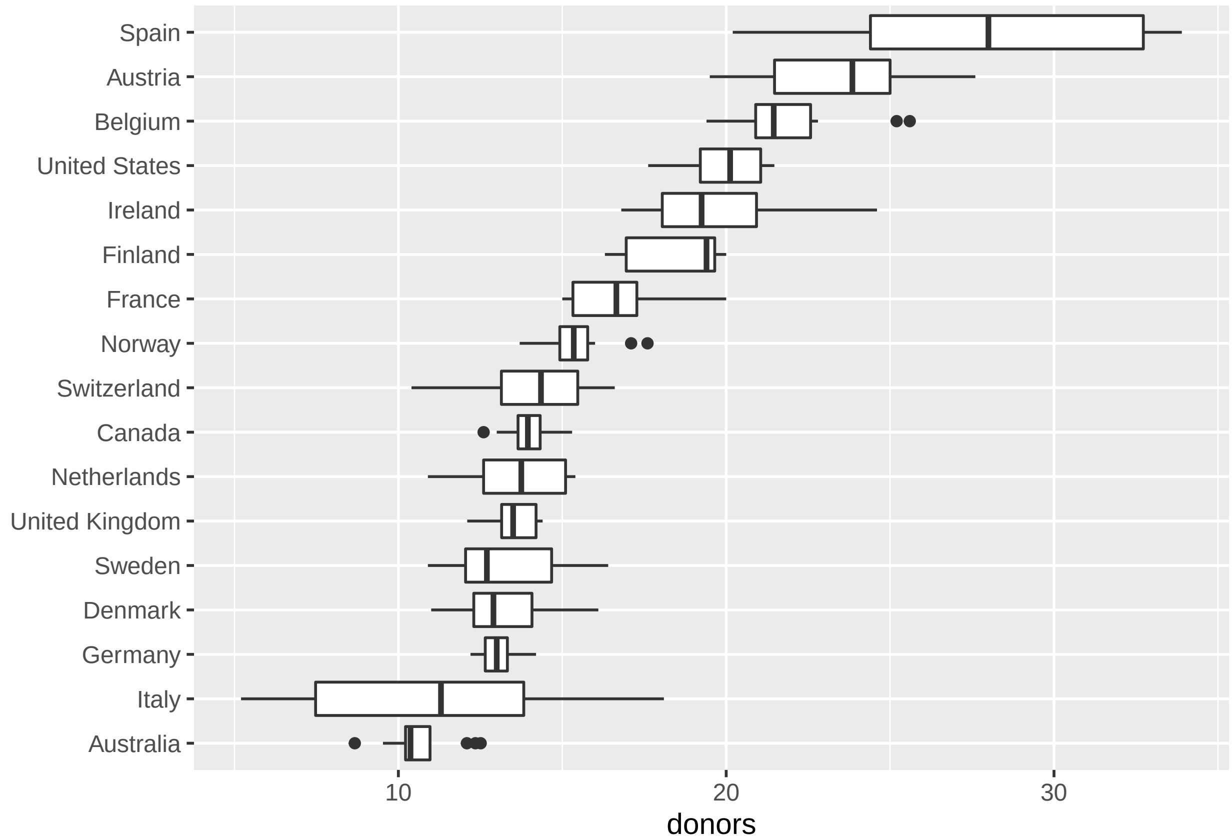

```r
p <- ggplot(data = organdata,
            mapping = aes(x = country, y = donors))
p + geom_boxplot() + coord_flip()
```

```
## Warning: Removed 34 rows containing non-finite values (stat_boxplot).
```



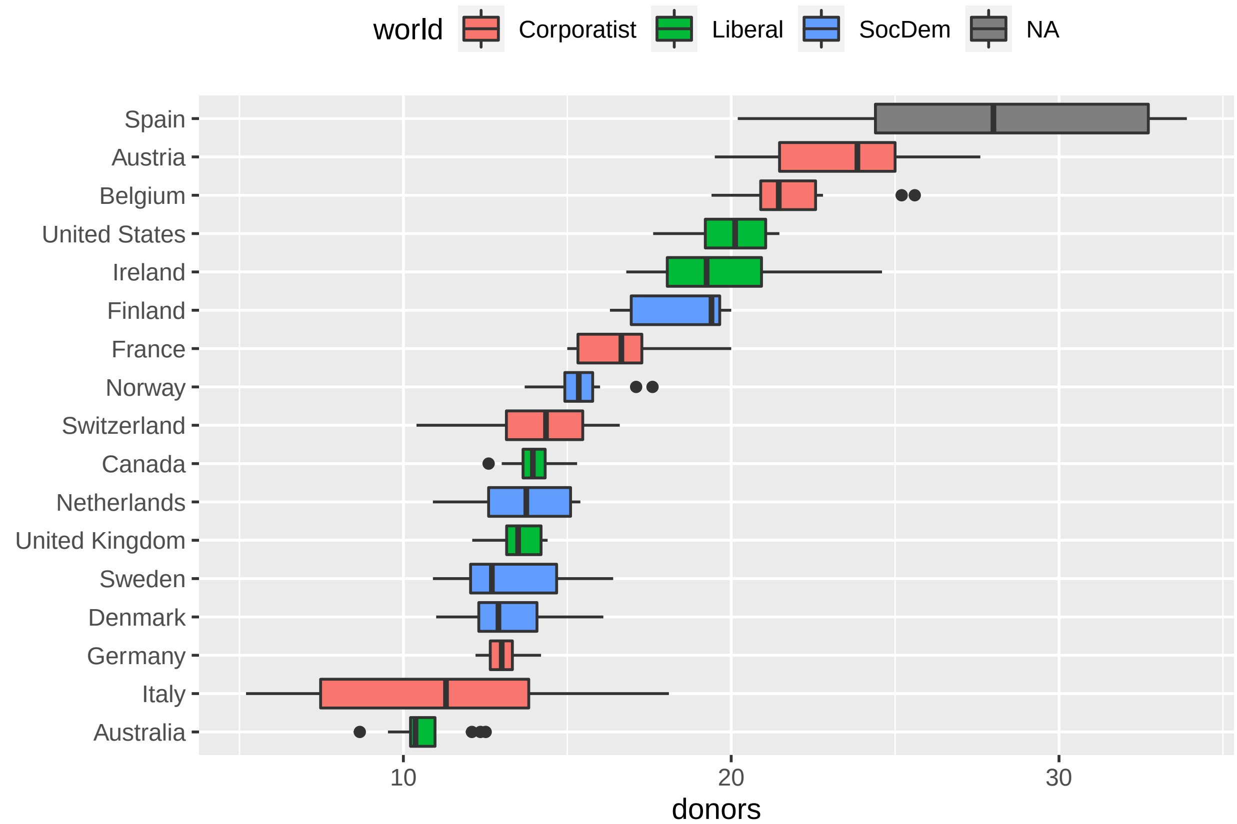

```
p <- ggplot(data = organdata,
            mapping = aes(x = reorder(country, donors, na.rm=TRUE),
                          y = donors))
p + geom_boxplot() +
    labs(x=NULL) +
    coord_flip()
```

```
## Warning: Removed 34 rows containing non-finite values (stat_boxplot).
```



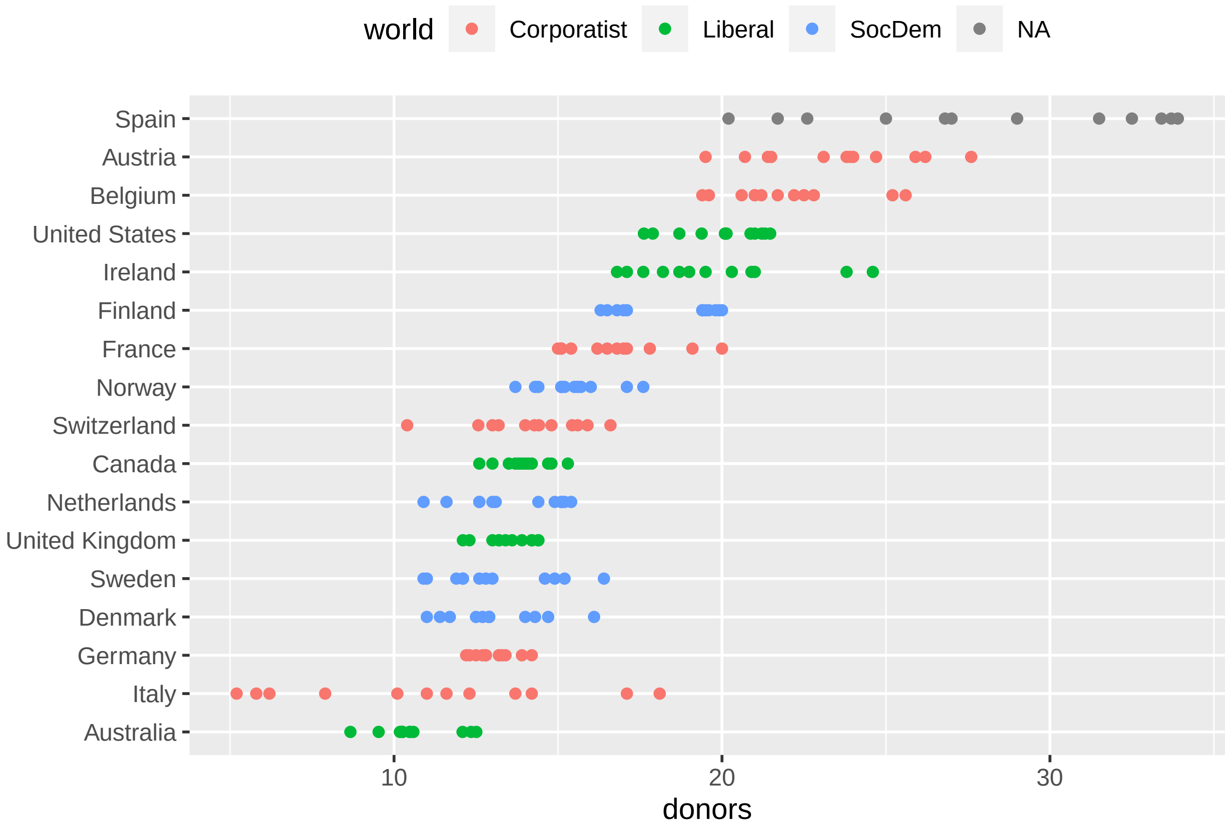

```
p <- ggplot(data = organdata,
            mapping = aes(x = reorder(country, donors, na.rm=TRUE),
                          y = donors, fill = world))
p + geom_boxplot() + labs(x=NULL) +
    coord_flip() + theme(legend.position = "top")
```

```
## Warning: Removed 34 rows containing non-finite values (stat_boxplot).
```



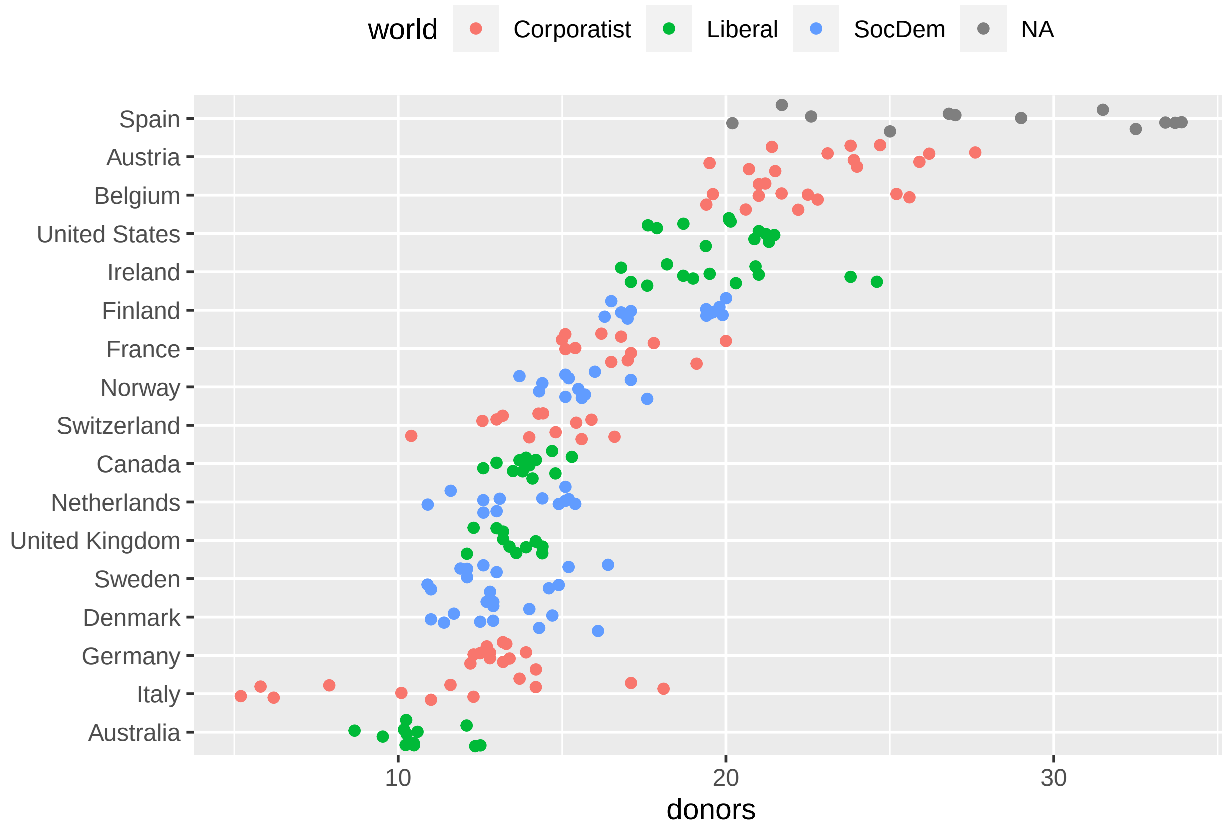

```r
p <- ggplot(data = organdata,
            mapping = aes(x = reorder(country, donors, na.rm=TRUE),
                          y = donors, color = world))
p + geom_point() + labs(x=NULL) +
    coord_flip() + theme(legend.position = "top")
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```



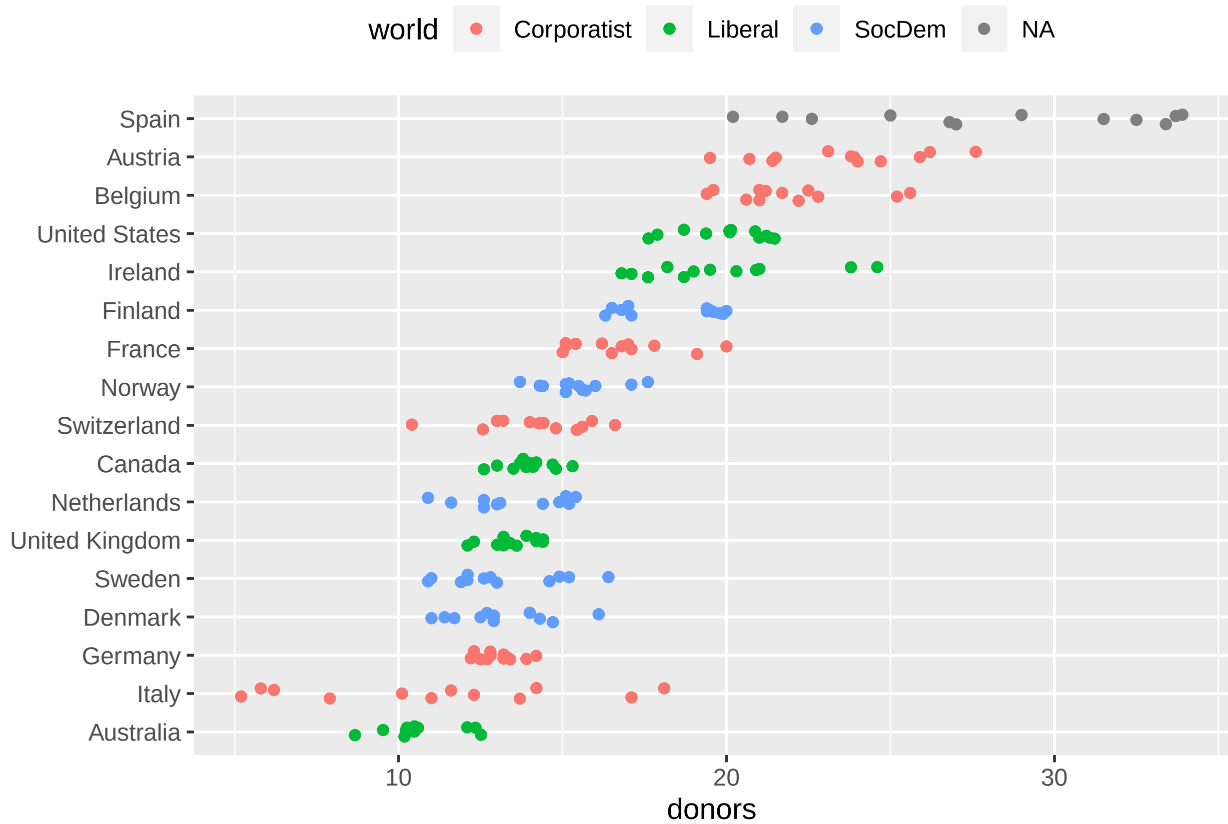

```
p <- ggplot(data = organdata,
            mapping = aes(x = reorder(country, donors, na.rm=TRUE),
                          y = donors, color = world))
p + geom_jitter() + labs(x=NULL) +
    coord_flip() + theme(legend.position = "top")
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```




```r
p <- ggplot(data = organdata,
            mapping = aes(x = reorder(country, donors, na.rm=TRUE),
                          y = donors, color = world))
p + geom_jitter(position = position_jitter(width=0.15)) +
    labs(x=NULL) + coord_flip() + theme(legend.position = "top")
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```




```r
by_country <- organdata %>% group_by(consent_law, country) %>%
    summarize(donors_mean= mean(donors, na.rm = TRUE),
              donors_sd = sd(donors, na.rm = TRUE),
              gdp_mean = mean(gdp, na.rm = TRUE),
              health_mean = mean(health, na.rm = TRUE),
              roads_mean = mean(roads, na.rm = TRUE),
              cerebvas_mean = mean(cerebvas, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'consent_law'. You can override using the `.groups` argument.
```

```r
by_country
```

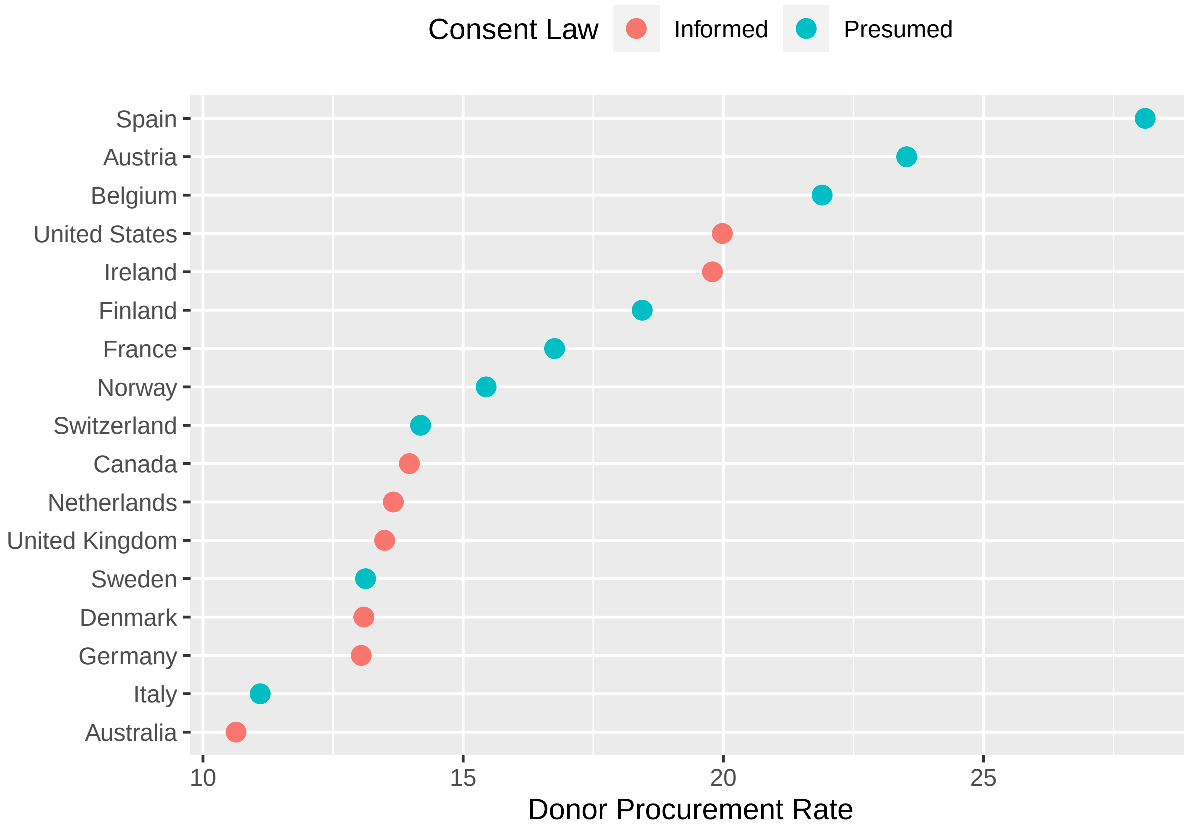
```
## # A tibble: 17 x 8
## # Groups:   consent_law [2]
##    consent_law country     donors_mean donors_sd gdp_mean health_mean roads_mean
##    <chr>       <chr>             <dbl>     <dbl>    <dbl>       <dbl>      <dbl>
##  1 Informed    Australia          10.6     1.14    22179.       1958.      105. 
##  2 Informed    Canada             14.0     0.751   23711.       2272.      109. 
##  3 Informed    Denmark            13.1     1.47    23722.       2054.      102. 
##  4 Informed    Germany            13.0     0.611   22163.       2349.      113. 
##  5 Informed    Ireland            19.8     2.48    20824.       1480.      118. 
##  6 Informed    Netherlands        13.7     1.55    23013.       1993.       76.1
##  7 Informed    United Kin~        13.5     0.775   21359.       1561.       67.9
##  8 Informed    United Sta~        20.0     1.33    29212.       3988.      155. 
##  9 Presumed    Austria            23.5     2.42    23876.       1875.      150. 
## 10 Presumed    Belgium            21.9     1.94    22500.       1958.      155. 
## 11 Presumed    Finland            18.4     1.53    21019.       1615.       93.6
```

```
## 12 Presumed    France            16.8      1.60    22603.        2160.       156.
## 13 Presumed    Italy             11.1      4.28    21554.        1757        122.
## 14 Presumed    Norway            15.4      1.11    26448.        2217.        70.0
## 15 Presumed    Spain             28.1      4.96    16933         1289.       161.
## 16 Presumed    Sweden            13.1      1.75    22415.        1951.        72.3
## 17 Presumed    Switzerland       14.2      1.71    27233         2776.        96.4
## # ... with 1 more variable: cerebvas_mean <dbl>
```

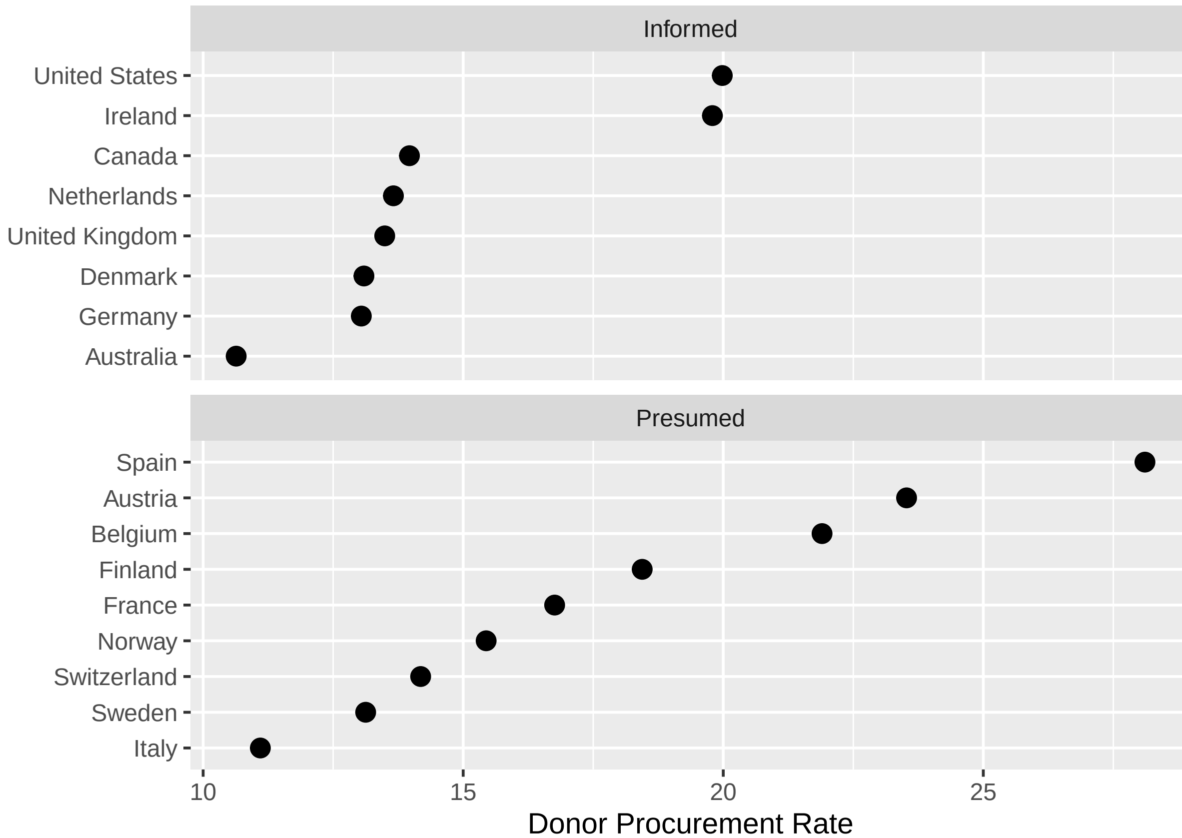
```
by_country <- organdata %>%
  group_by(consent_law, country) %>%
    summarize_if(is.numeric,
                 list(~ mean(., na.rm = TRUE),
                      ~ sd(., na.rm = TRUE))) %>%
    ungroup()

by_country
```

```
## # A tibble: 17 x 28
##    consent_law country  donors_mean pop_mean pop_dens_mean gdp_mean gdp_lag_mean
##    <chr>       <chr>          <dbl>    <dbl>         <dbl>    <dbl>        <dbl>
##  1 Informed    Austral~        10.6   18318.         0.237   22179.       21779.
##  2 Informed    Canada          14.0   29608.         0.297   23711.       23353.
##  3 Informed    Denmark         13.1    5257.        12.2     23722.       23275
##  4 Informed    Germany         13.0   80255.        22.5     22163.       21938.
##  5 Informed    Ireland         19.8    3674.         5.23    20824.       20154.
##  6 Informed    Netherl~        13.7   15548.        37.4     23013.       22554.
##  7 Informed    United ~        13.5   58187.        24.0     21359.       20962.
##  8 Informed    United ~        20.0  269330.         2.80    29212.       28699.
##  9 Presumed    Austria         23.5    7927.         9.45    23876.       23415.
## 10 Presumed    Belgium         21.9   10153.        30.7     22500.       22096.
## 11 Presumed    Finland         18.4    5112.         1.51    21019.       20763
## 12 Presumed    France          16.8   58056.        10.5     22603.       22211.
## 13 Presumed    Italy           11.1   57360.        19.0     21554.       21195.
## 14 Presumed    Norway          15.4    4386.         1.35    26448.       25769.
## 15 Presumed    Spain           28.1   39666.         7.84    16933        16584.
## 16 Presumed    Sweden          13.1    8789.         1.95    22415.       22094
## 17 Presumed    Switzer~        14.2    7037.        17.0     27233        26931.
## # ... with 21 more variables: health_mean <dbl>, health_lag_mean <dbl>,
## #   pubhealth_mean <dbl>, roads_mean <dbl>, cerebvas_mean <dbl>,
## #   assault_mean <dbl>, external_mean <dbl>, txp_pop_mean <dbl>,
## #   donors_sd <dbl>, pop_sd <dbl>, pop_dens_sd <dbl>, gdp_sd <dbl>,
## #   gdp_lag_sd <dbl>, health_sd <dbl>, health_lag_sd <dbl>, pubhealth_sd <dbl>,
## #   roads_sd <dbl>, cerebvas_sd <dbl>, assault_sd <dbl>, external_sd <dbl>,
## #   txp_pop_sd <dbl>
```

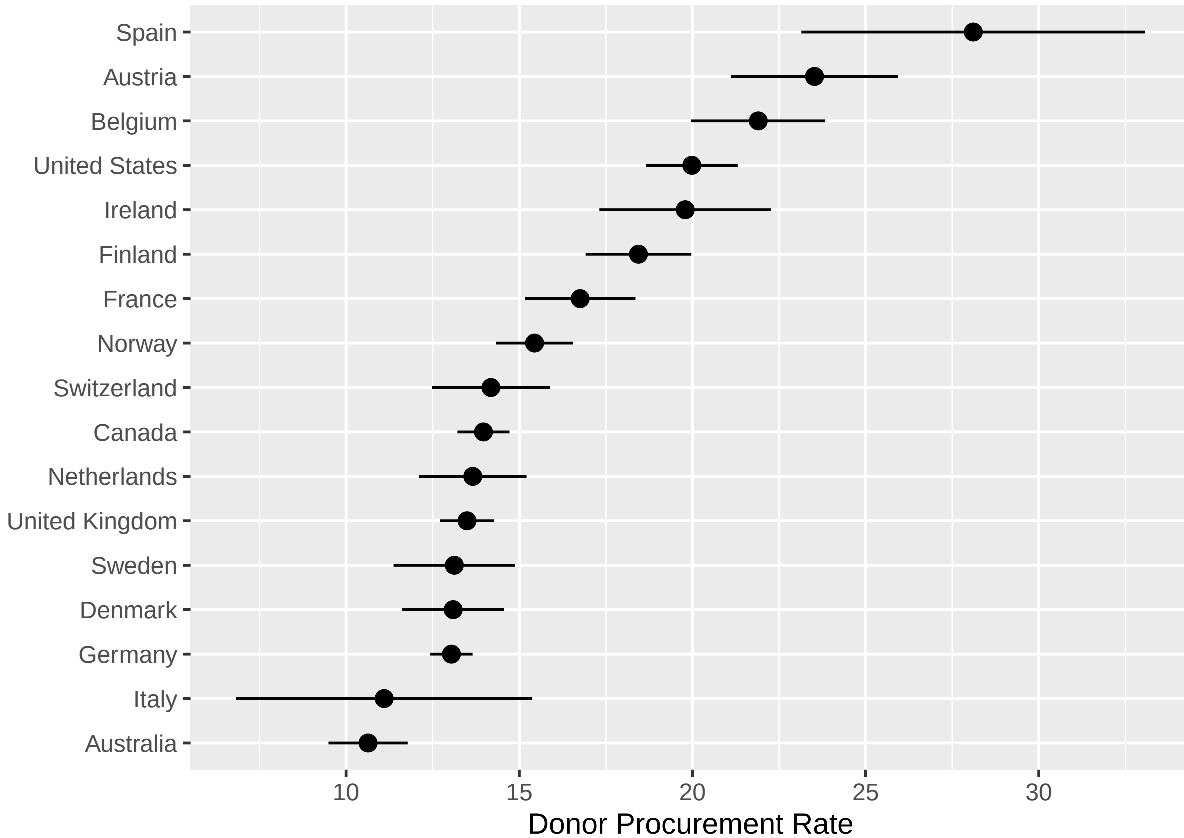
```
p <- ggplot(data = by_country,
            mapping = aes(x = donors_mean,
                          y = reorder(country, donors_mean),
                          color = consent_law))
p + geom_point(size=3) +
    labs(x = "Donor Procurement Rate",
         y = "", color = "Consent Law") +
    theme(legend.position="top")
```




```
p <- ggplot(data = by_country,
            mapping = aes(x = donors_mean,
                          y = reorder(country, donors_mean)))

p + geom_point(size=3) +
    facet_wrap(~ consent_law, scales = "free_y", ncol = 1) +
    labs(x= "Donor Procurement Rate",
         y= "")
```

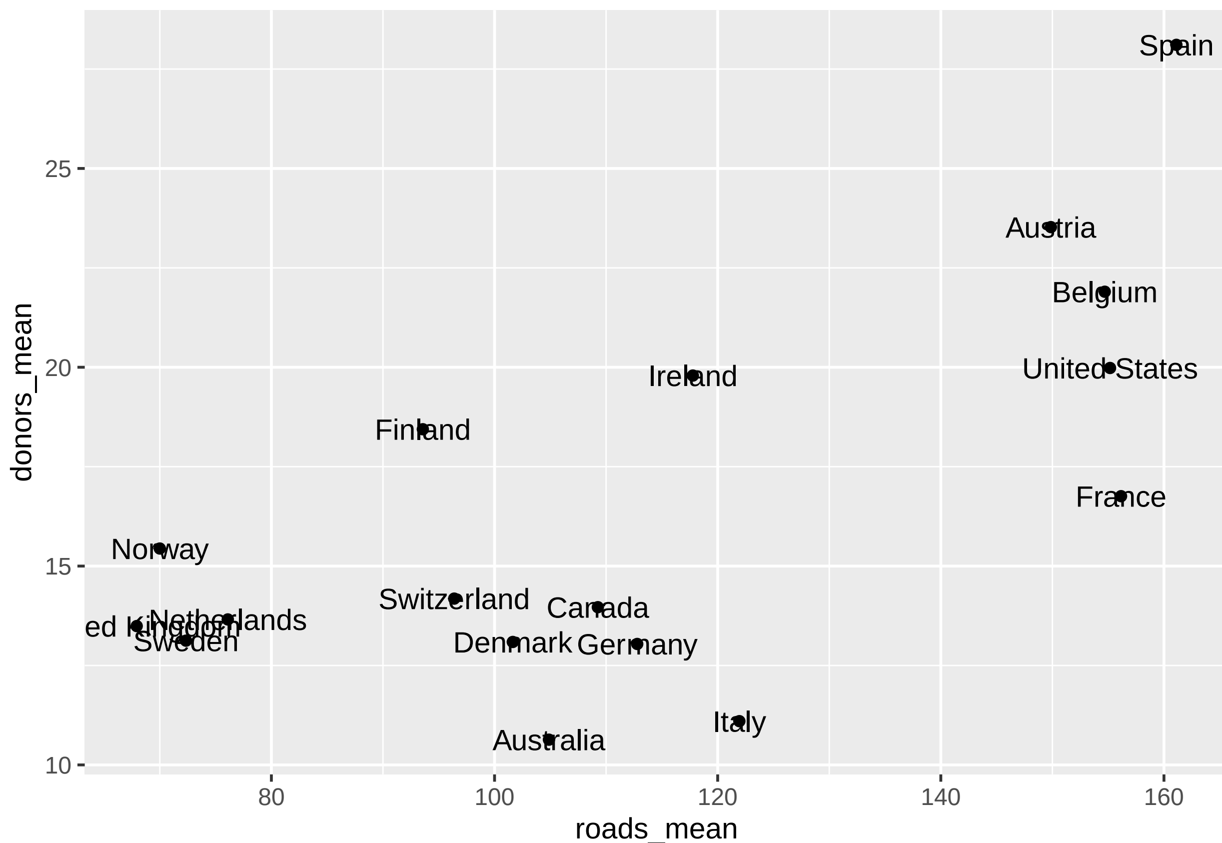



```
p <- ggplot(data = by_country, mapping = aes(x = reorder(country,
              donors_mean), y = donors_mean))

p + geom_pointrange(mapping = aes(ymin = donors_mean - donors_sd,
       ymax = donors_mean + donors_sd)) +
     labs(x= "", y= "Donor Procurement Rate") + coord_flip()
```

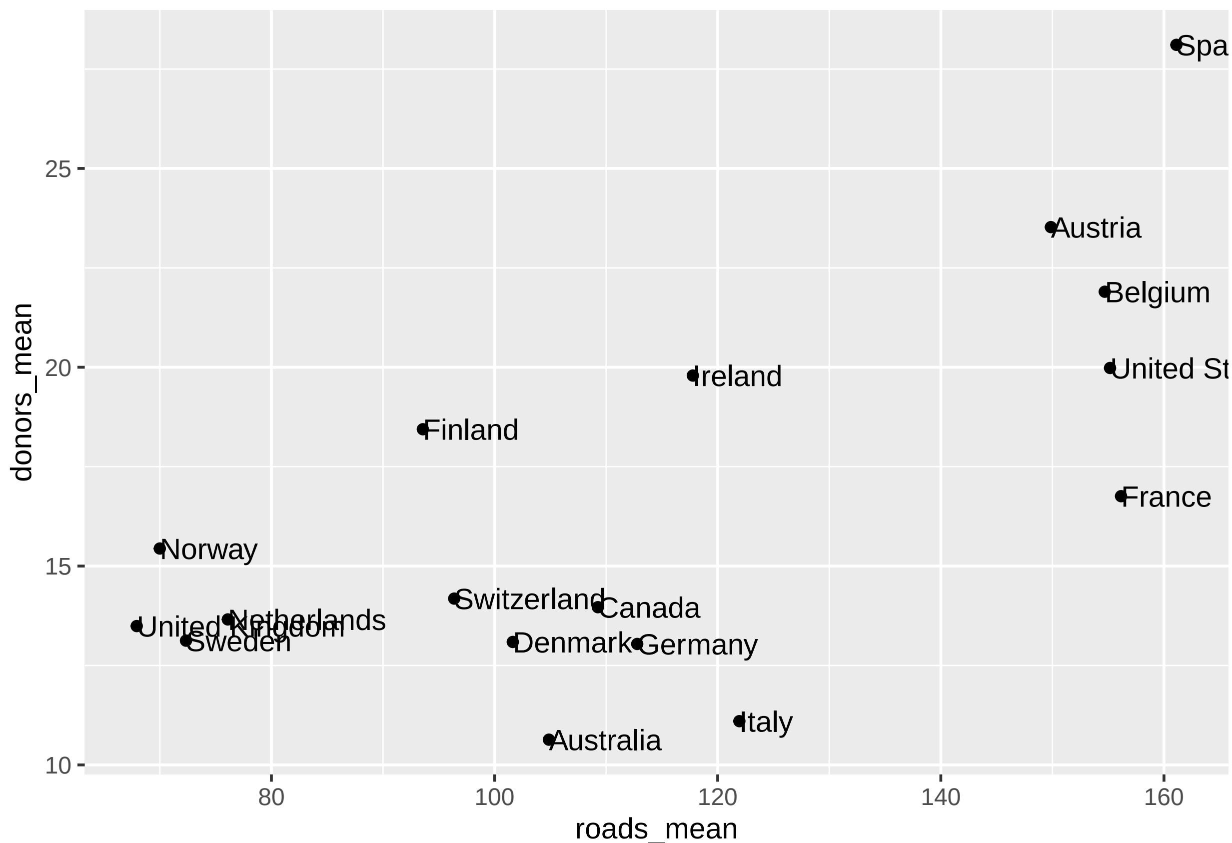


Plot text directly

```
p <- ggplot(data = by_country,
            mapping = aes(x = roads_mean, y = donors_mean))
p + geom_point() + geom_text(mapping = aes(label = country))
```




```r
p <- ggplot(data = by_country,
            mapping = aes(x = roads_mean, y = donors_mean))

p + geom_point() + geom_text(mapping = aes(label = country), hjust = 0)
```




```
library(ggrepel)
```

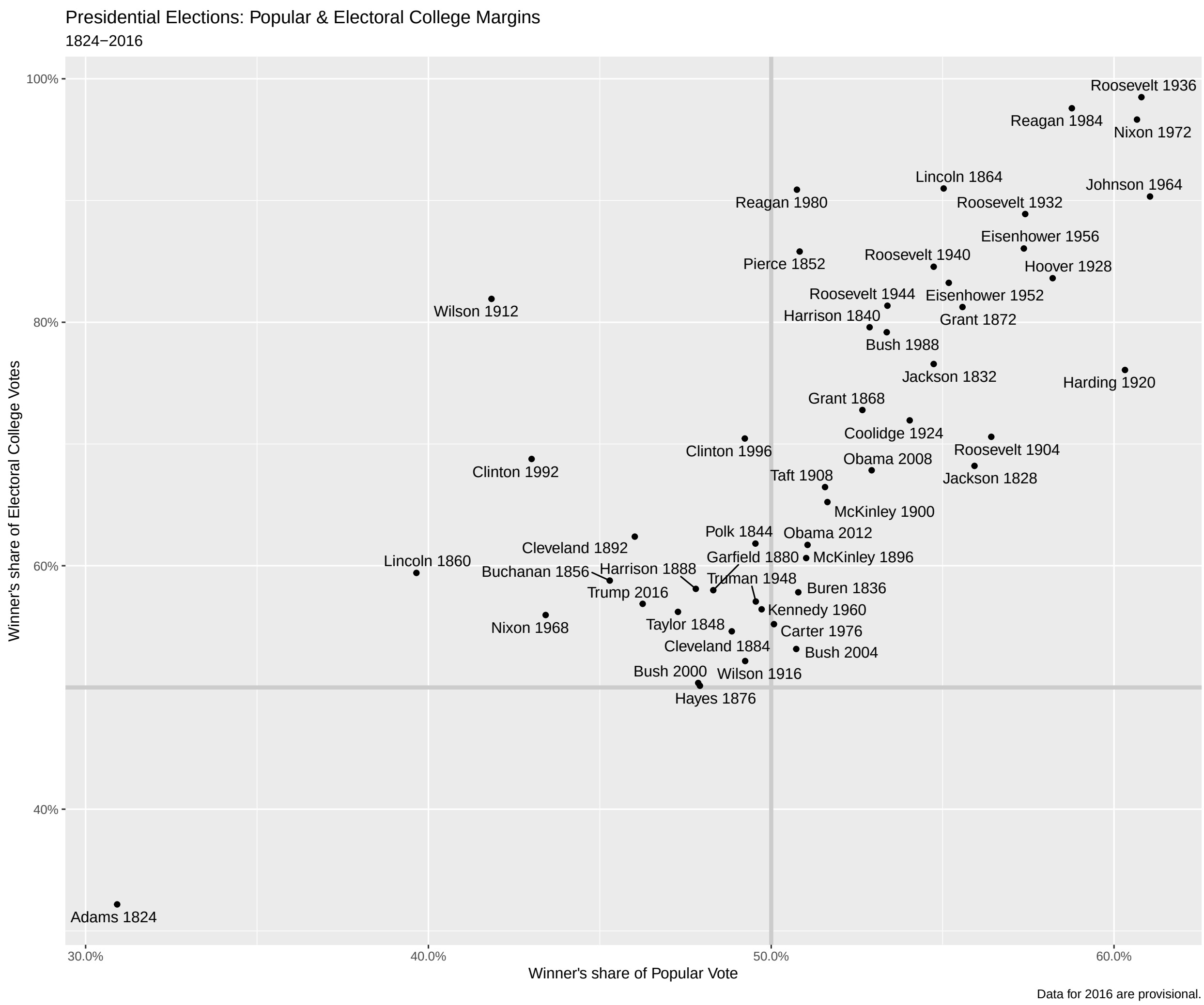
```
## Warning: package 'ggrepel' was built under R version 3.6.3
```

```
elections_historic %>% select(2:7)
```

```
## # A tibble: 49 x 6
##     year winner                 win_party ec_pct popular_pct popular_margin
##    <int> <chr>                  <chr>      <dbl>       <dbl>          <dbl>
##  1  1824 John Quincy Adams      D.-R.      0.322       0.309        -0.104
##  2  1828 Andrew Jackson         Dem.       0.682       0.559         0.122
##  3  1832 Andrew Jackson         Dem.       0.766       0.547         0.178
##  4  1836 Martin Van Buren       Dem.       0.578       0.508         0.142
##  5  1840 William Henry Harrison Whig       0.796       0.529         0.0605
##  6  1844 James Polk             Dem.       0.618       0.495         0.0145
##  7  1848 Zachary Taylor         Whig       0.562       0.473         0.0479
##  8  1852 Franklin Pierce        Dem.       0.858       0.508         0.0695
##  9  1856 James Buchanan         Dem.       0.588       0.453         0.122
## 10  1860 Abraham Lincoln        Rep.       0.594       0.396         0.101
## # ... with 39 more rows
```

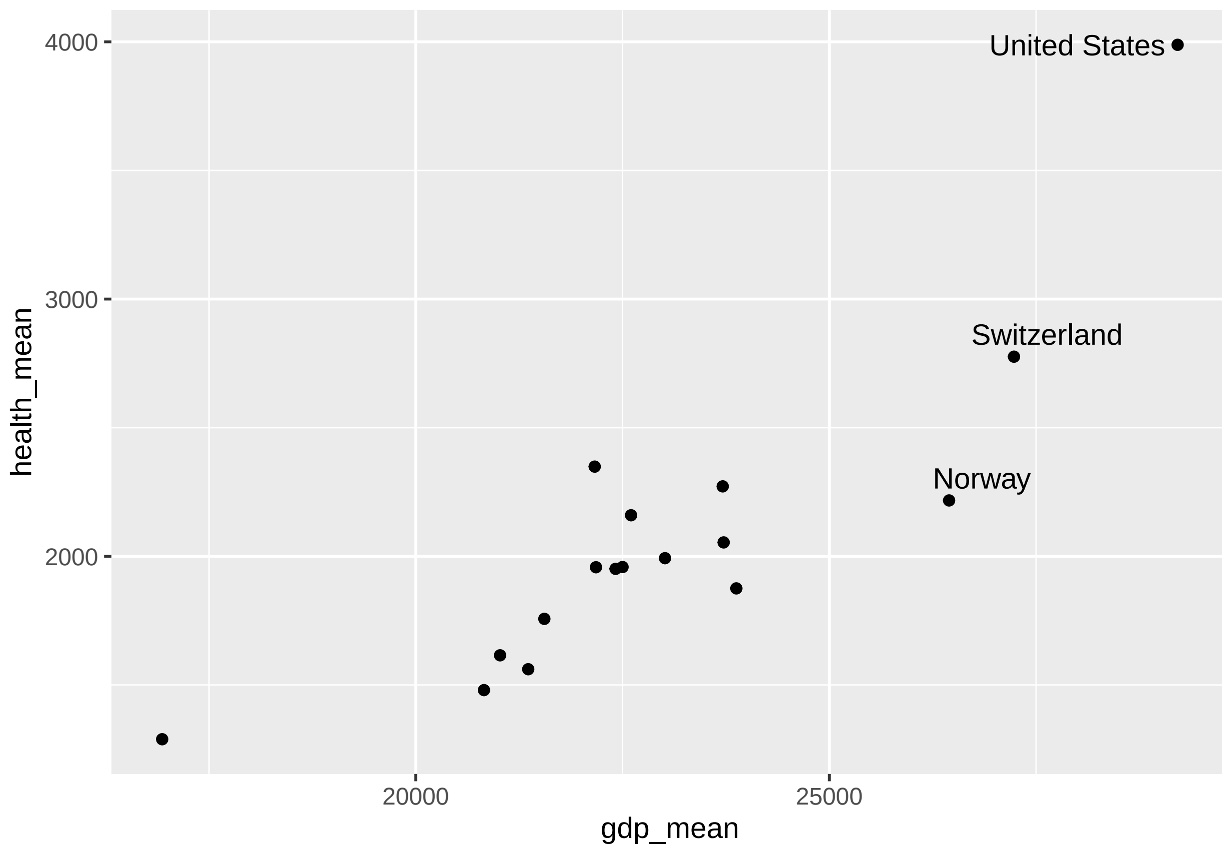
```
p_title <- "Presidential Elections: Popular & Electoral College Margins"
p_subtitle <- "1824-2016"
p_caption <- "Data for 2016 are provisional."
x_label <- "Winner's share of Popular Vote"
y_label <- "Winner's share of Electoral College Votes"

p <- ggplot(elections_historic, aes(x = popular_pct, y = ec_pct,
```

```
                             label = winner_label))

p + geom_hline(yintercept = 0.5, size = 1.4, color = "gray80") +
    geom_vline(xintercept = 0.5, size = 1.4, color = "gray80") +
    geom_point() +
    geom_text_repel() +
    scale_x_continuous(labels = scales::percent) +
    scale_y_continuous(labels = scales::percent) +
    labs(x = x_label, y = y_label, title = p_title, subtitle = p_subtitle,
         caption = p_caption)
```

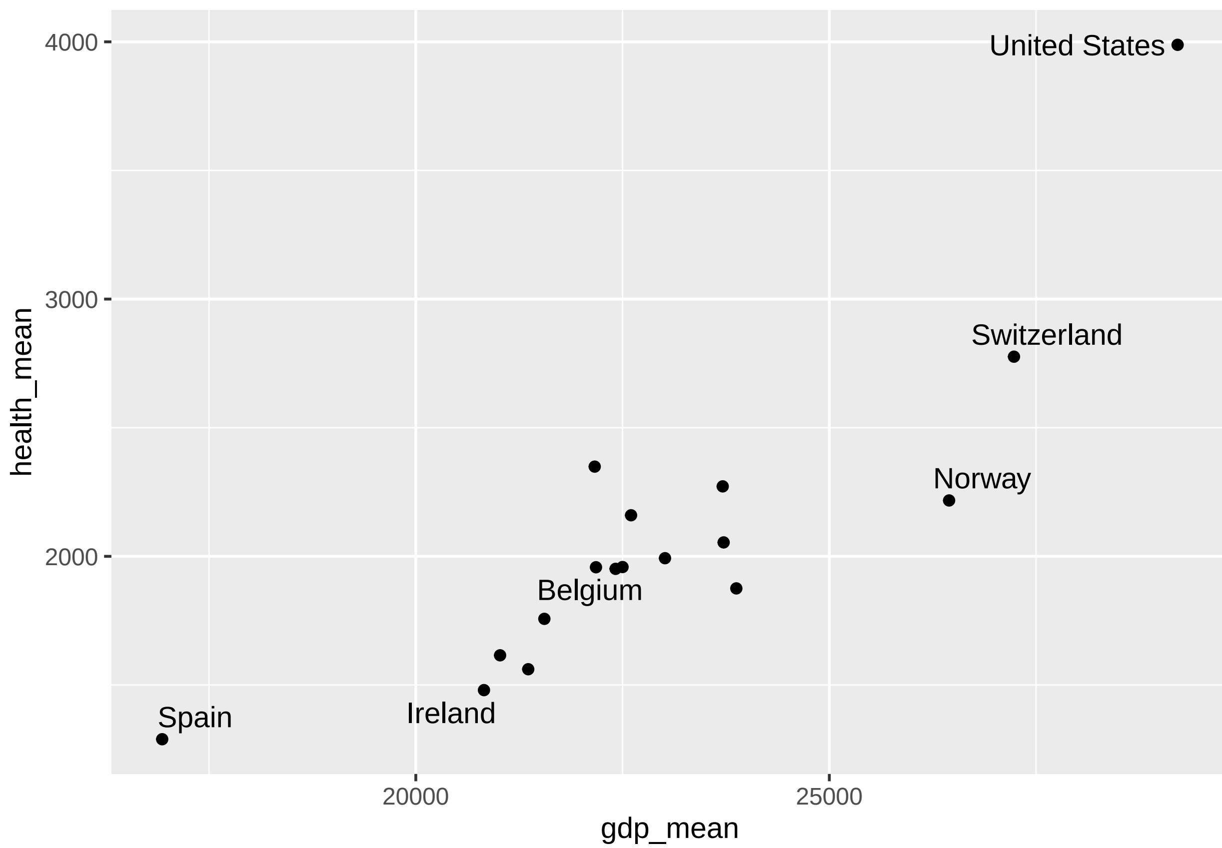


Selective labels

```
p <- ggplot(data = by_country,
            mapping = aes(x = gdp_mean, y = health_mean))

p + geom_point() +
    geom_text_repel(data = subset(by_country, gdp_mean > 25000),
                    mapping = aes(label = country))
```

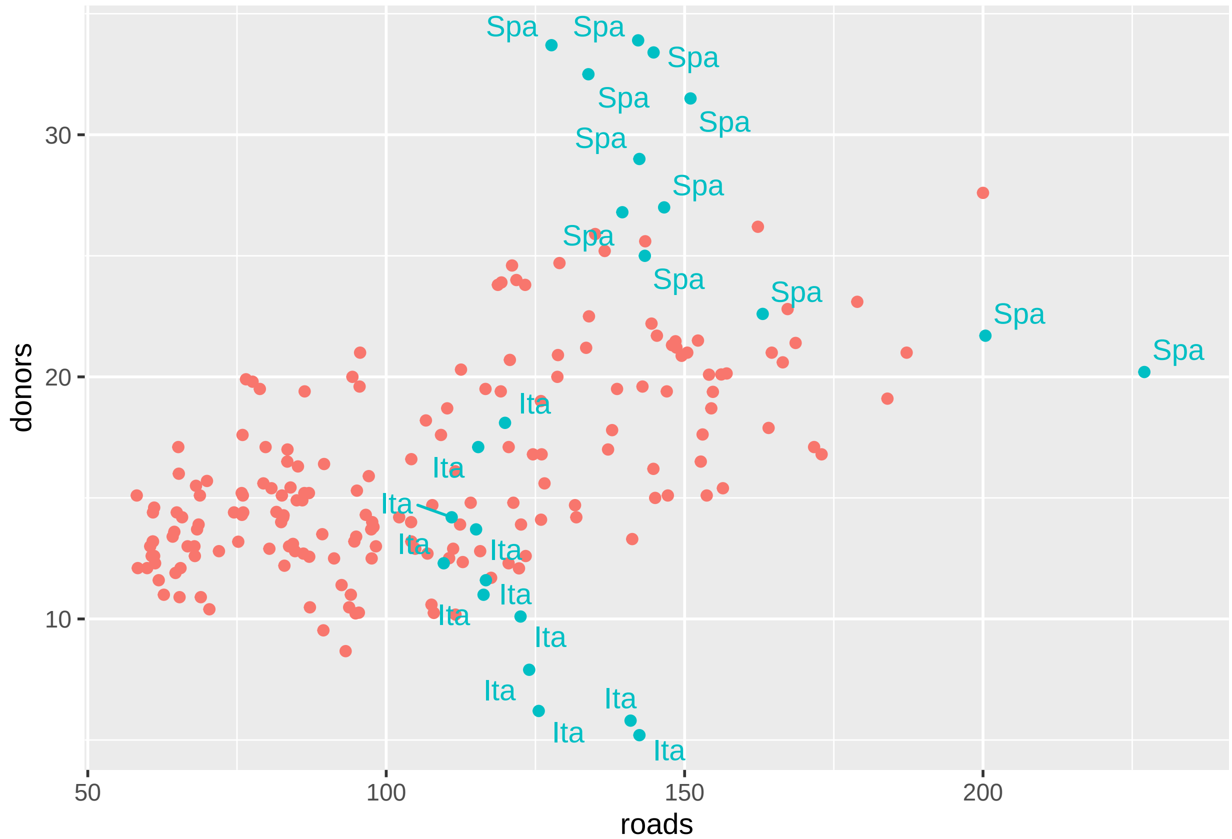



```
p <- ggplot(data = by_country,
            mapping = aes(x = gdp_mean, y = health_mean))

p + geom_point() +
    geom_text_repel(data = subset(by_country,
                                  gdp_mean > 25000 | health_mean < 1500 |
                                  country %in% "Belgium"),
                    mapping = aes(label = country))
```




```
organdata$ind <- organdata$ccode %in% c("Ita", "Spa") &
                    organdata$year > 1998

p <- ggplot(data = organdata,
            mapping = aes(x = roads,
                          y = donors, color = ind))
p + geom_point() +
    geom_text_repel(data = subset(organdata, ind),
                    mapping = aes(label = ccode)) +
    guides(label = FALSE, color = FALSE)
```

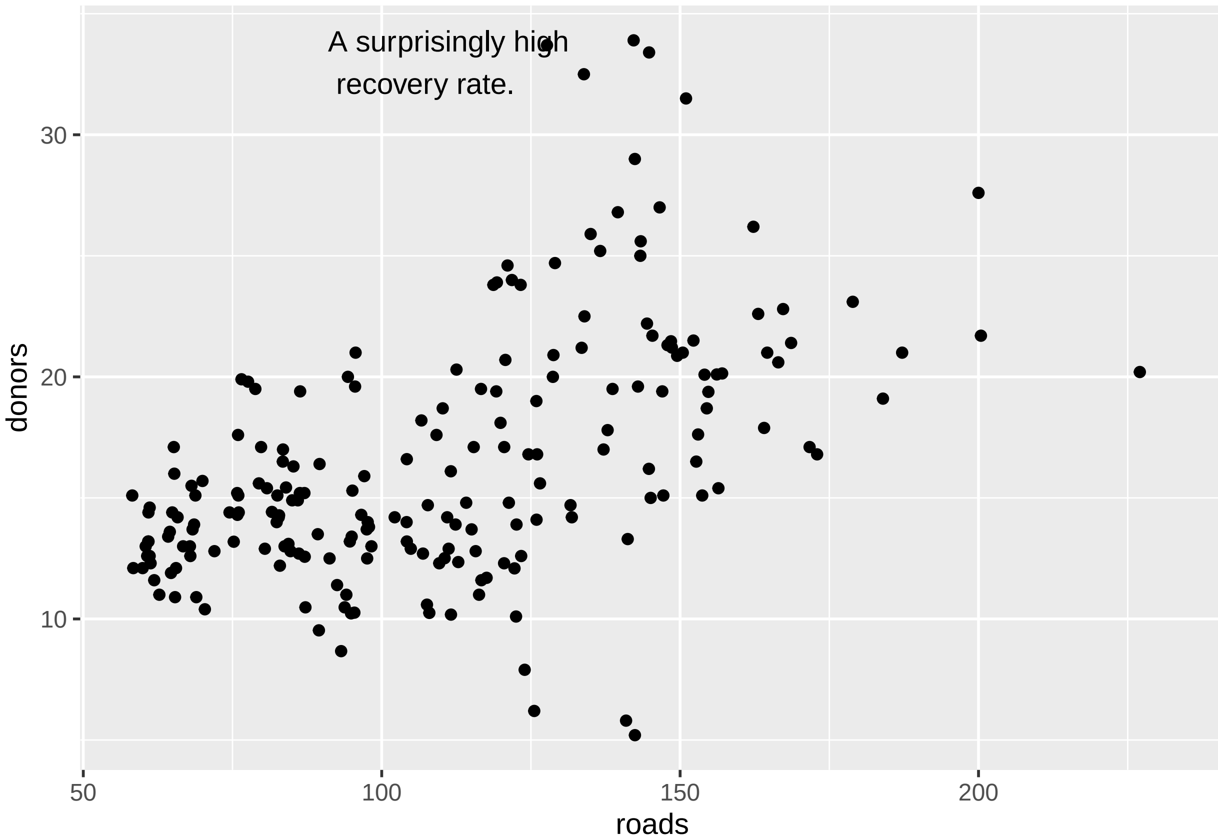
```
## Warning: Removed 34 rows containing missing values (geom_point).
```



Arbitrary annotation

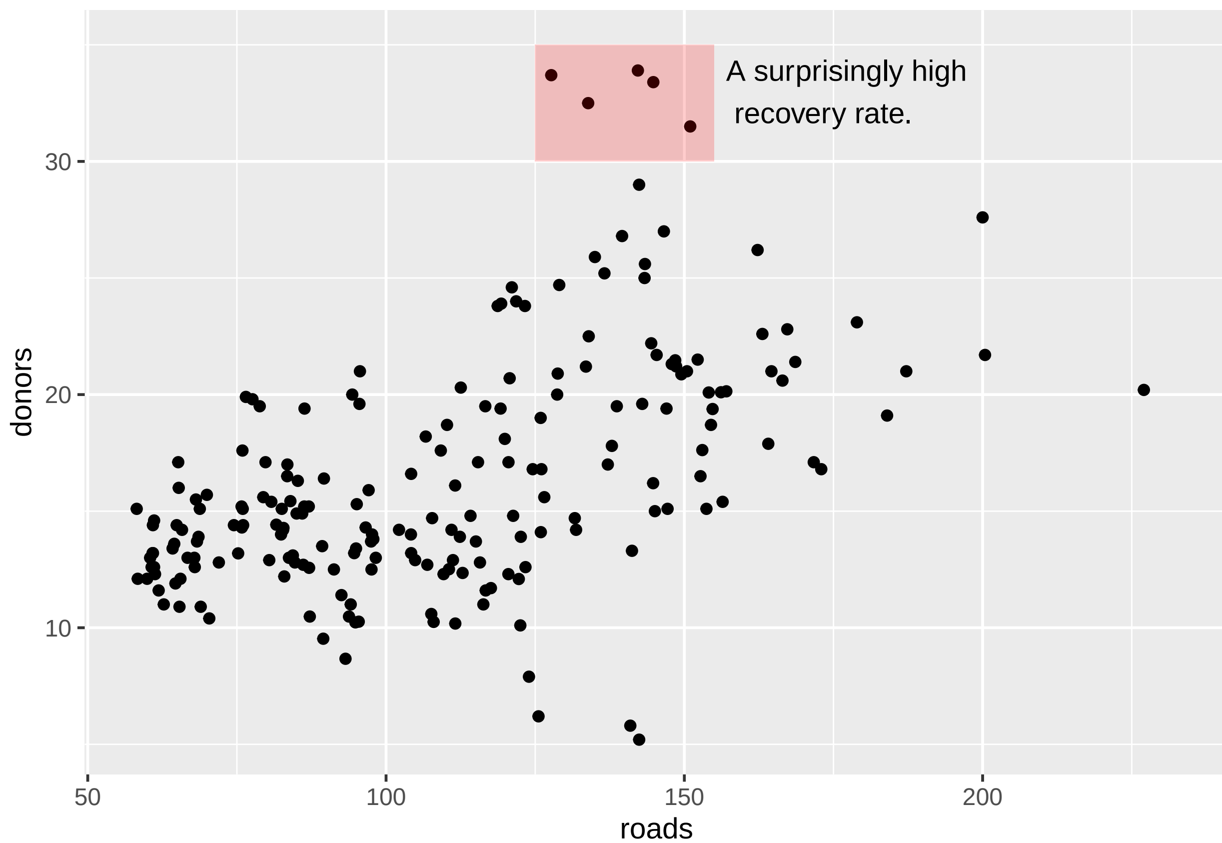
```r
p <- ggplot(data = organdata, mapping = aes(x = roads, y = donors))
p + geom_point() + annotate(geom = "text", x = 91, y = 33,
                            label = "A surprisingly high \n recovery rate.",
                            hjust = 0)
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```




```r
p <- ggplot(data = organdata,
            mapping = aes(x = roads, y = donors))
p + geom_point() +
    annotate(geom = "rect", xmin = 125, xmax = 155,
             ymin = 30, ymax = 35, fill = "red", alpha = 0.2) +
    annotate(geom = "text", x = 157, y = 33,
             label = "A surprisingly high \n recovery rate.", hjust = 0)
```

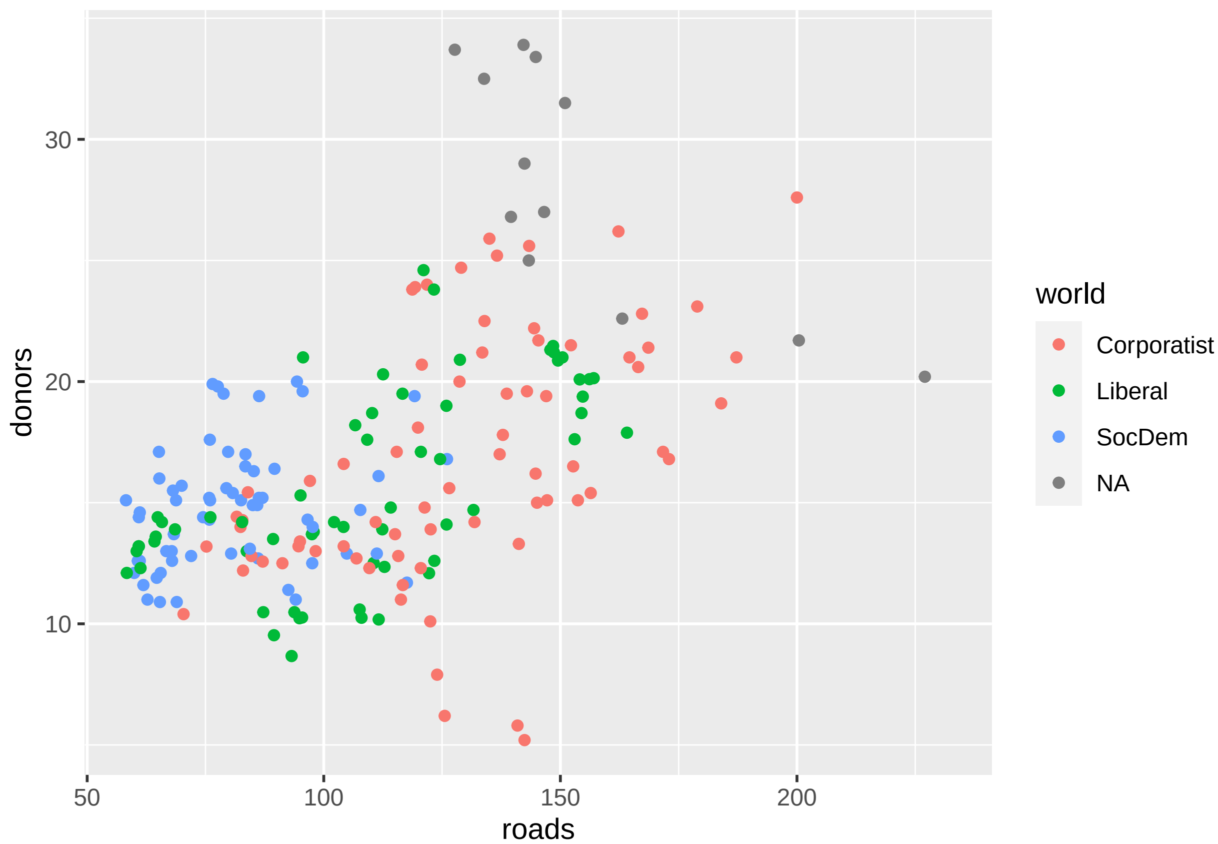
```
## Warning: Removed 34 rows containing missing values (geom_point).
```



Scales and Guides

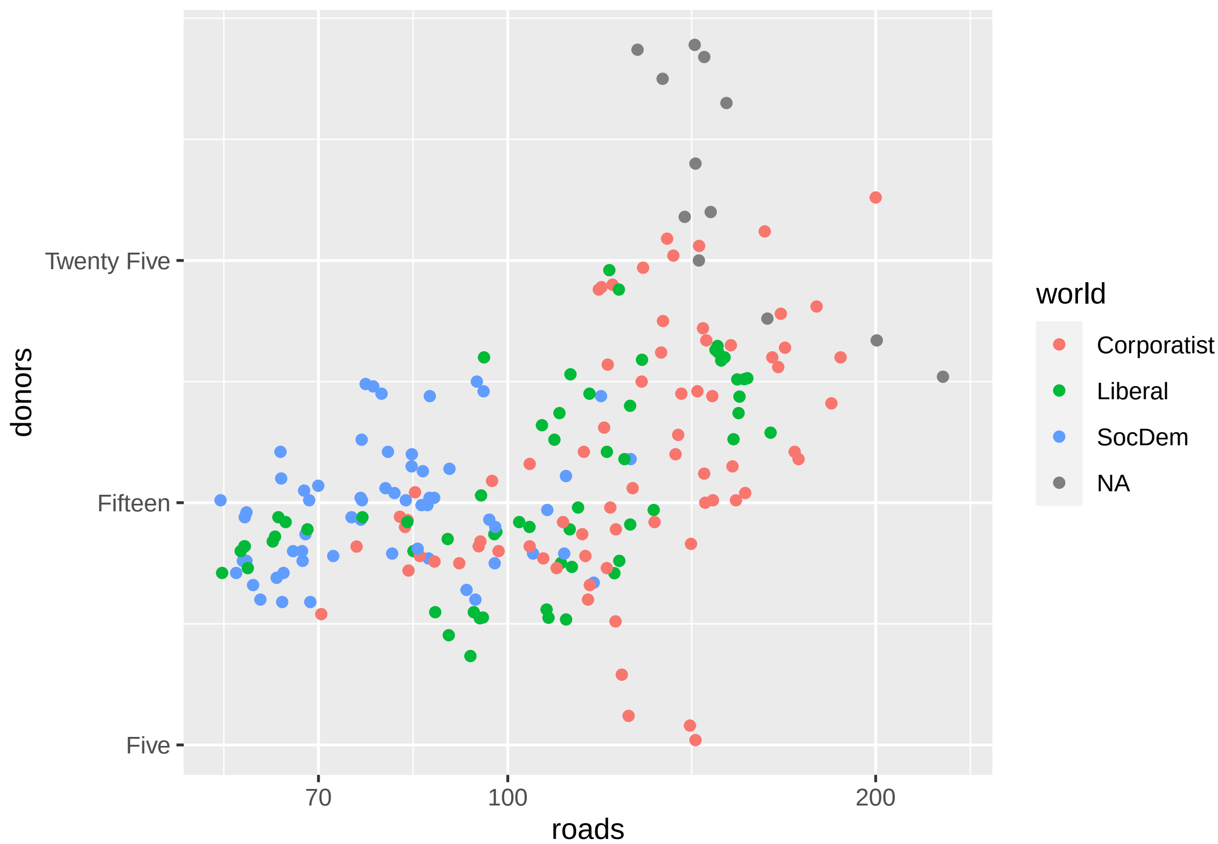
```
p <- ggplot(data = organdata,
            mapping = aes(x = roads,
                          y = donors,
                          color = world))
p + geom_point()
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```




```
p <- ggplot(data = organdata,
            mapping = aes(x = roads,
                          y = donors,
                          color = world))
p + geom_point() +
    scale_x_log10() +
    scale_y_continuous(breaks = c(5, 15, 25),
                       labels = c("Five", "Fifteen", "Twenty Five"))
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```

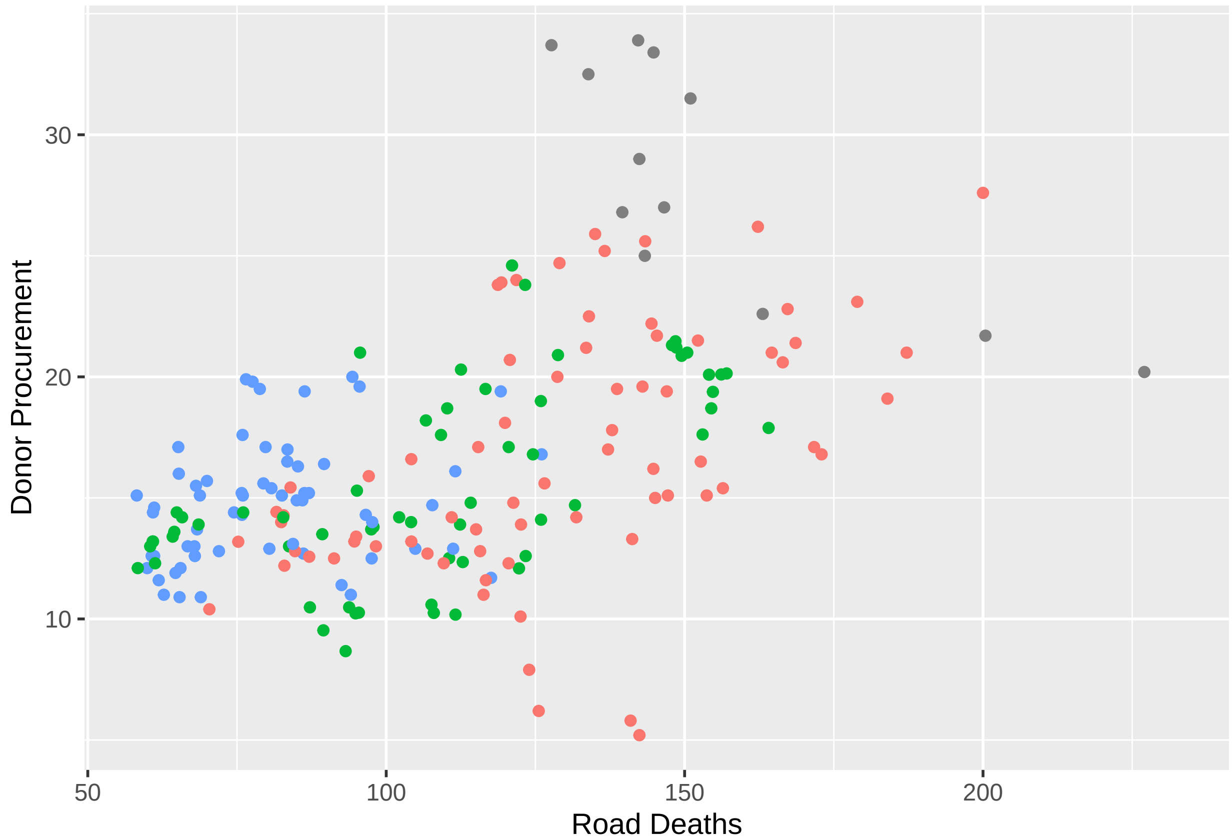



```
p <- ggplot(data = organdata,
            mapping = aes(x = roads,
                          y = donors,
                          color = world))
p + geom_point() +
    scale_color_discrete(labels =
                             c("Corporatist", "Liberal",
                               "Social Democratic", "Unclassified")) +
    labs(x = "Road Deaths",
         y = "Donor Procurement",
        color = "Welfare State")
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```

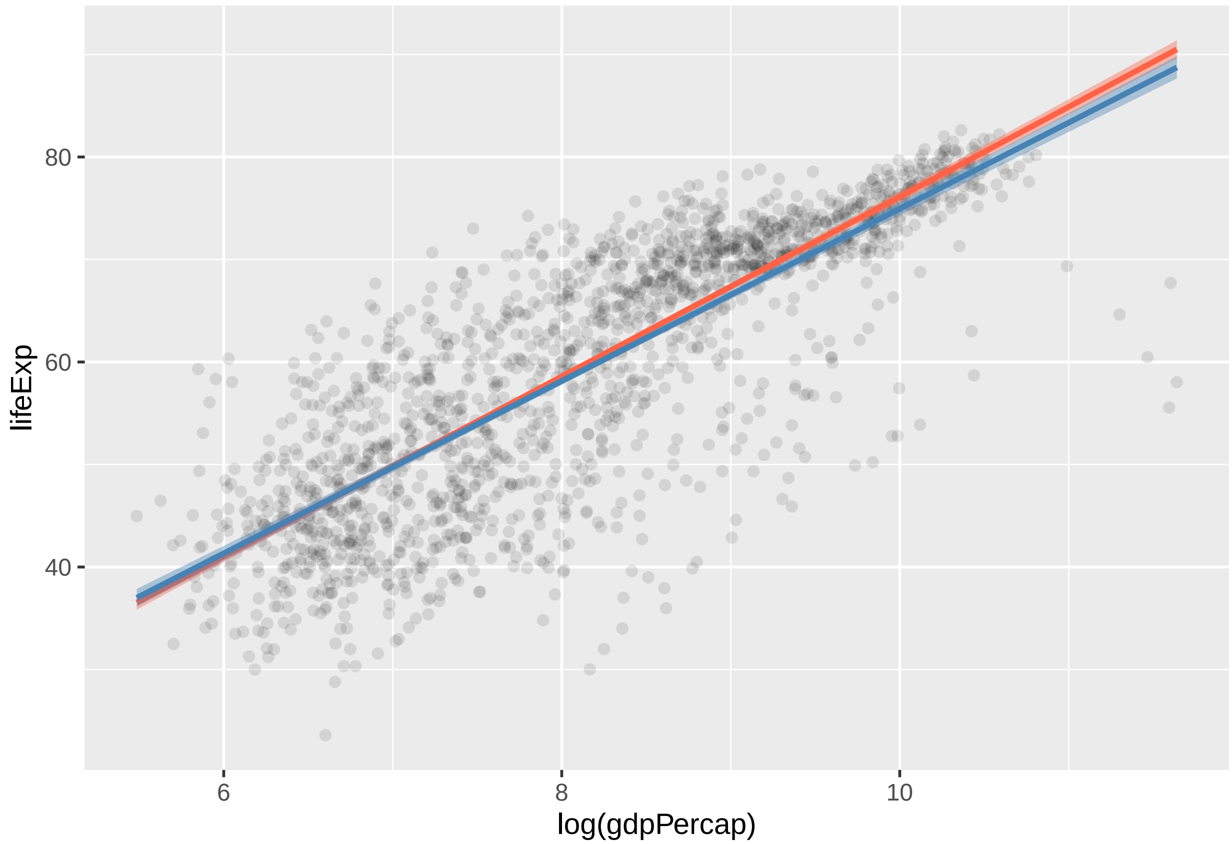

```
p <- ggplot(data = organdata,
            mapping = aes(x = roads,
                          y = donors,
                          color = world))
p + geom_point() +
    labs(x = "Road Deaths",
         y = "Donor Procurement") +
    guides(color = FALSE)
```

```
## Warning: Removed 34 rows containing missing values (geom_point).
```

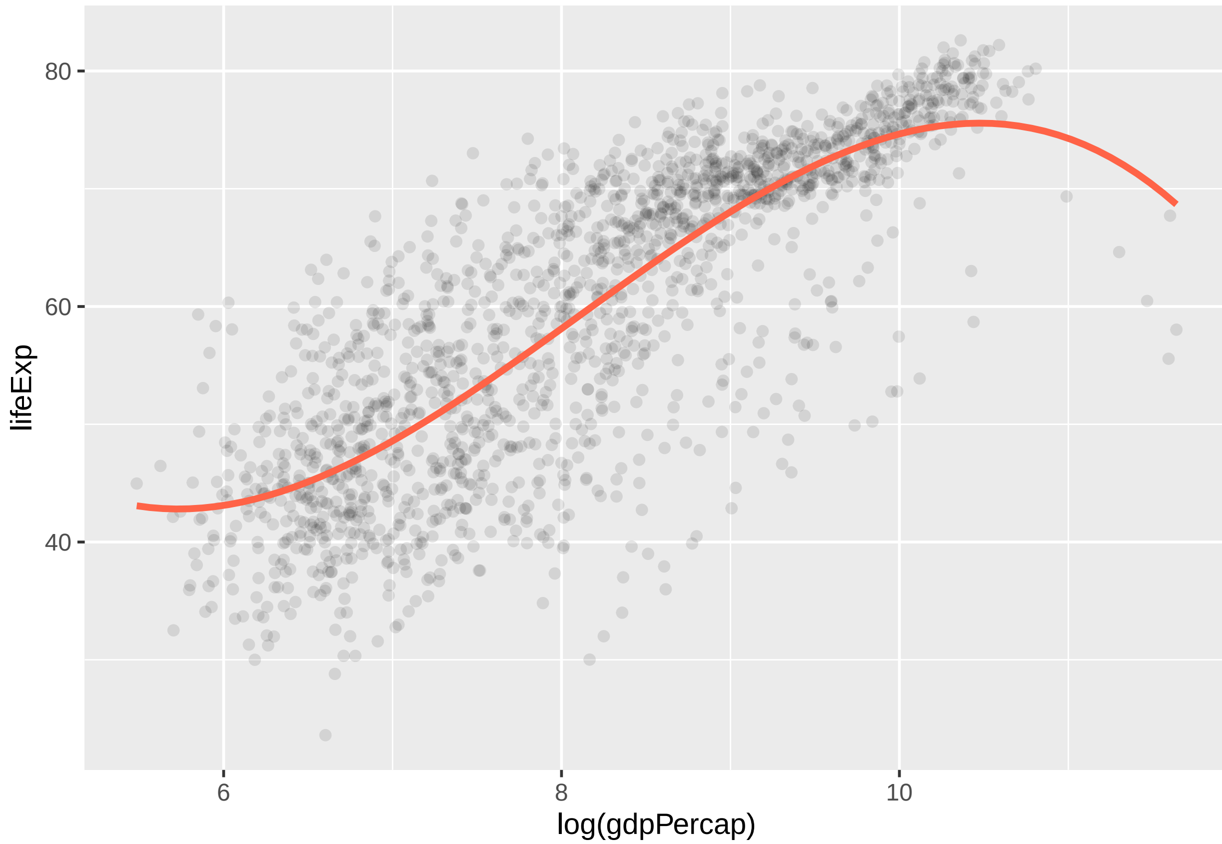



```r
p <- ggplot(data = gapminder,
            mapping = aes(x = log(gdpPercap), y = lifeExp))

p + geom_point(alpha=0.1) +
    geom_smooth(color = "tomato", fill="tomato", method = MASS::rlm) +
    geom_smooth(color = "steelblue", fill="steelblue", method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

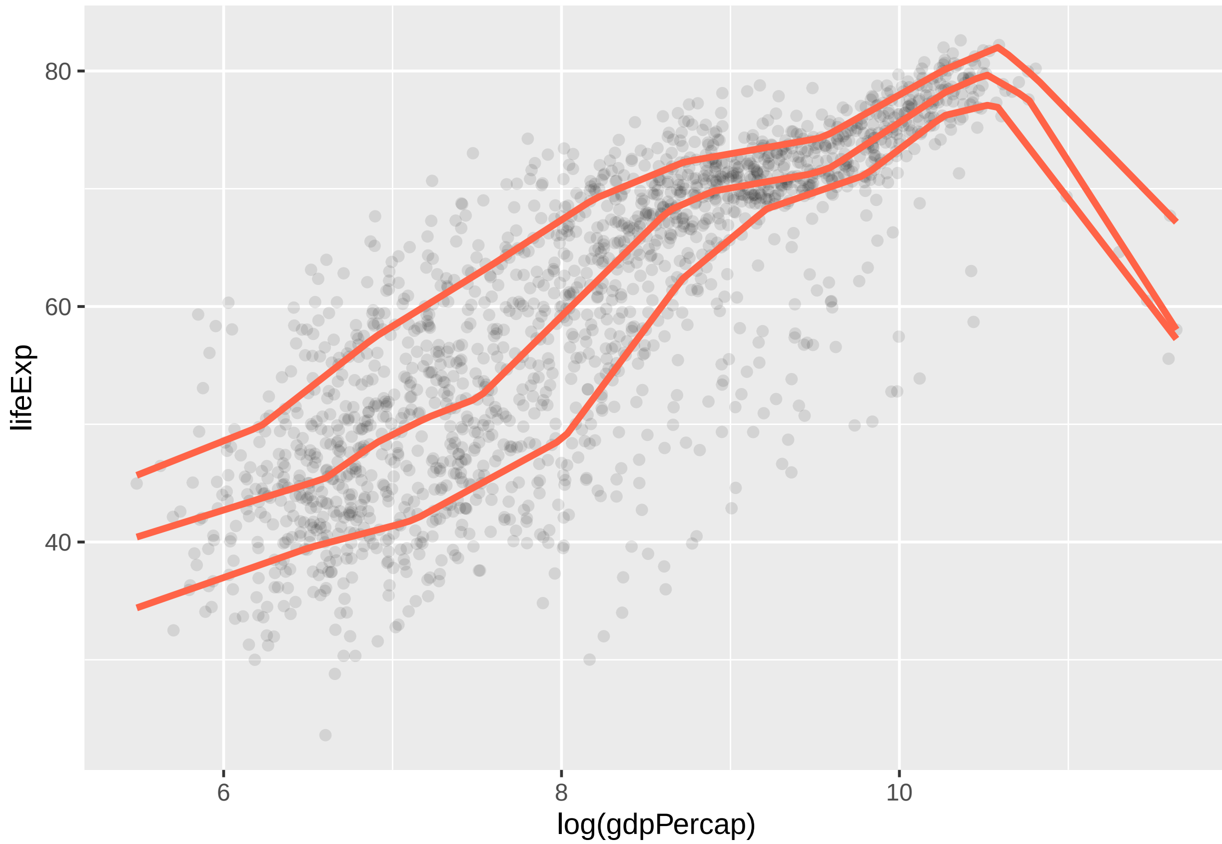



```
p + geom_point(alpha=0.1) +
    geom_smooth(color = "tomato", method = "lm", size = 1.2,
                formula = y ~ splines::bs(x, 3), se = FALSE)
```



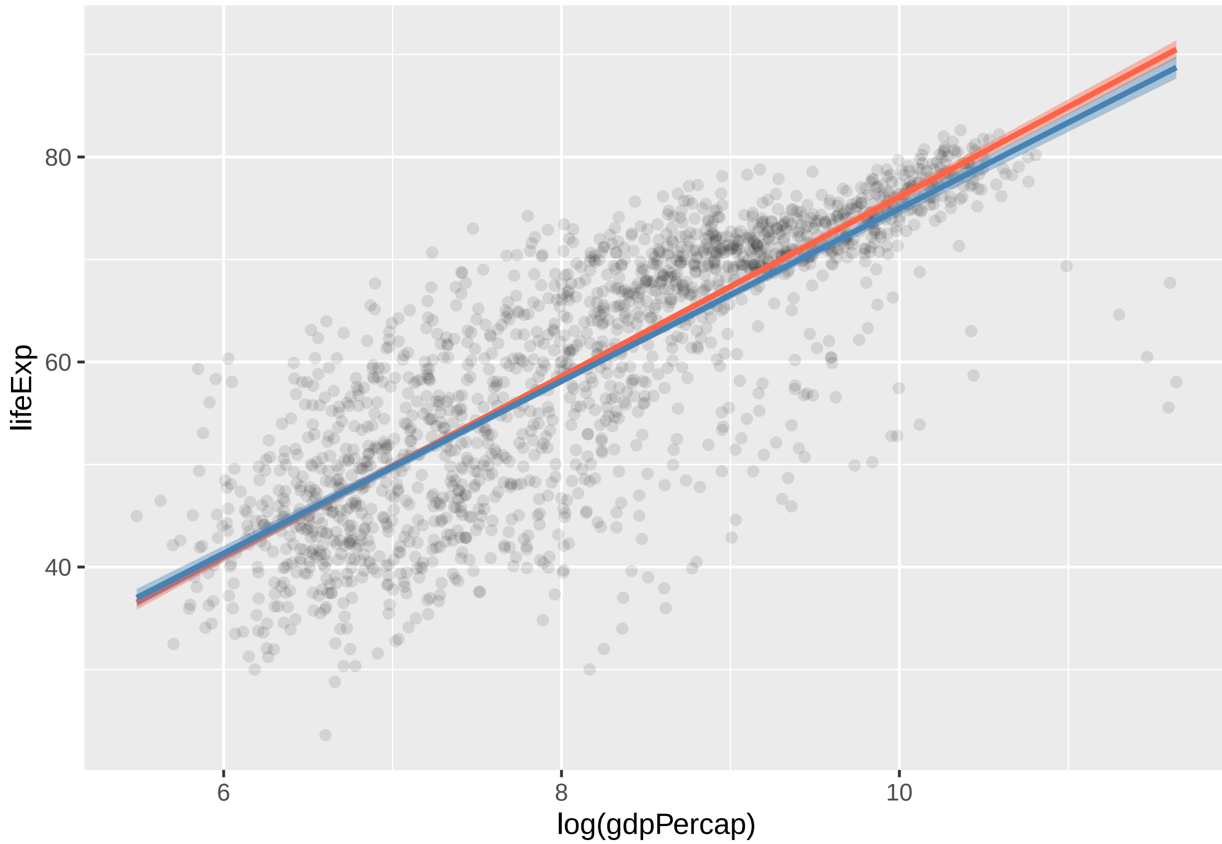

```
p + geom_point(alpha=0.1) +
    geom_quantile(color = "tomato", size = 1.2, method = "rqss",
                  lambda = 1, quantiles = c(0.20, 0.5, 0.85))
```

```
## Smoothing formula not specified. Using: y ~ qss(x, lambda = 1)
```

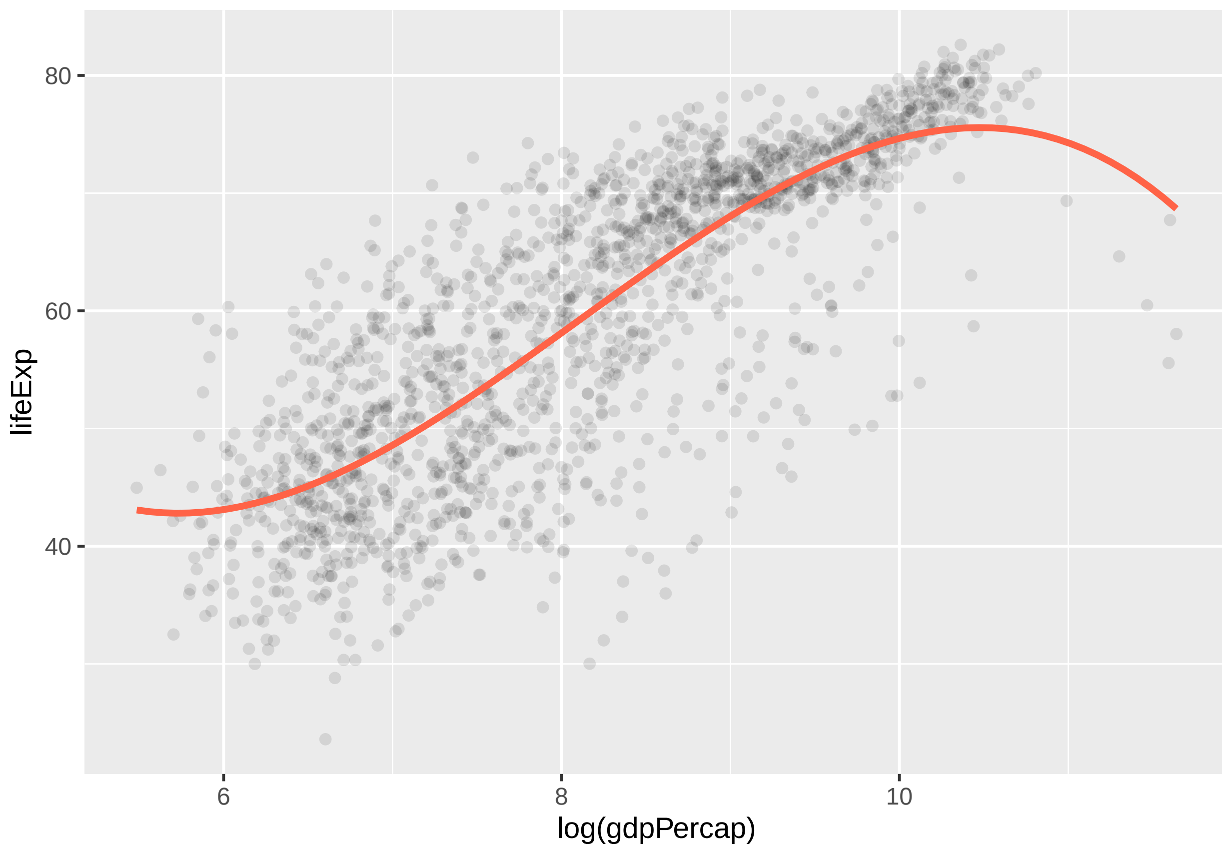



```
p <- ggplot(data = gapminder,
            mapping = aes(x = log(gdpPercap), y = lifeExp))

p + geom_point(alpha=0.1) +
    geom_smooth(color = "tomato", fill="tomato", method = MASS::rlm) +
    geom_smooth(color = "steelblue", fill="steelblue", method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

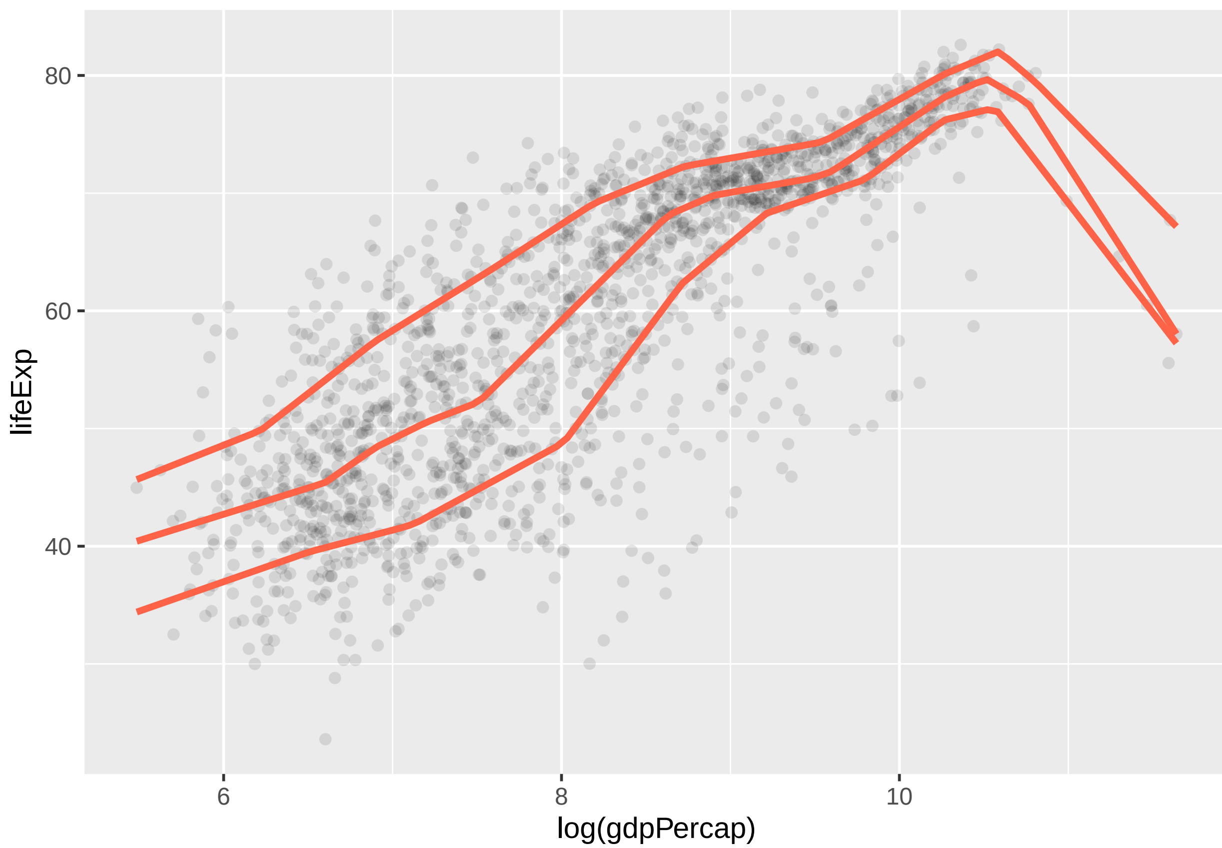



```
p + geom_point(alpha=0.1) +
    geom_smooth(color = "tomato", method = "lm", size = 1.2,
                formula = y ~ splines::bs(x, 3), se = FALSE)
```




```
p + geom_point(alpha=0.1) +
    geom_quantile(color = "tomato", size = 1.2, method = "rqss",
                  lambda = 1, quantiles = c(0.20, 0.5, 0.85))
```

```
## Smoothing formula not specified. Using: y ~ qss(x, lambda = 1)
```



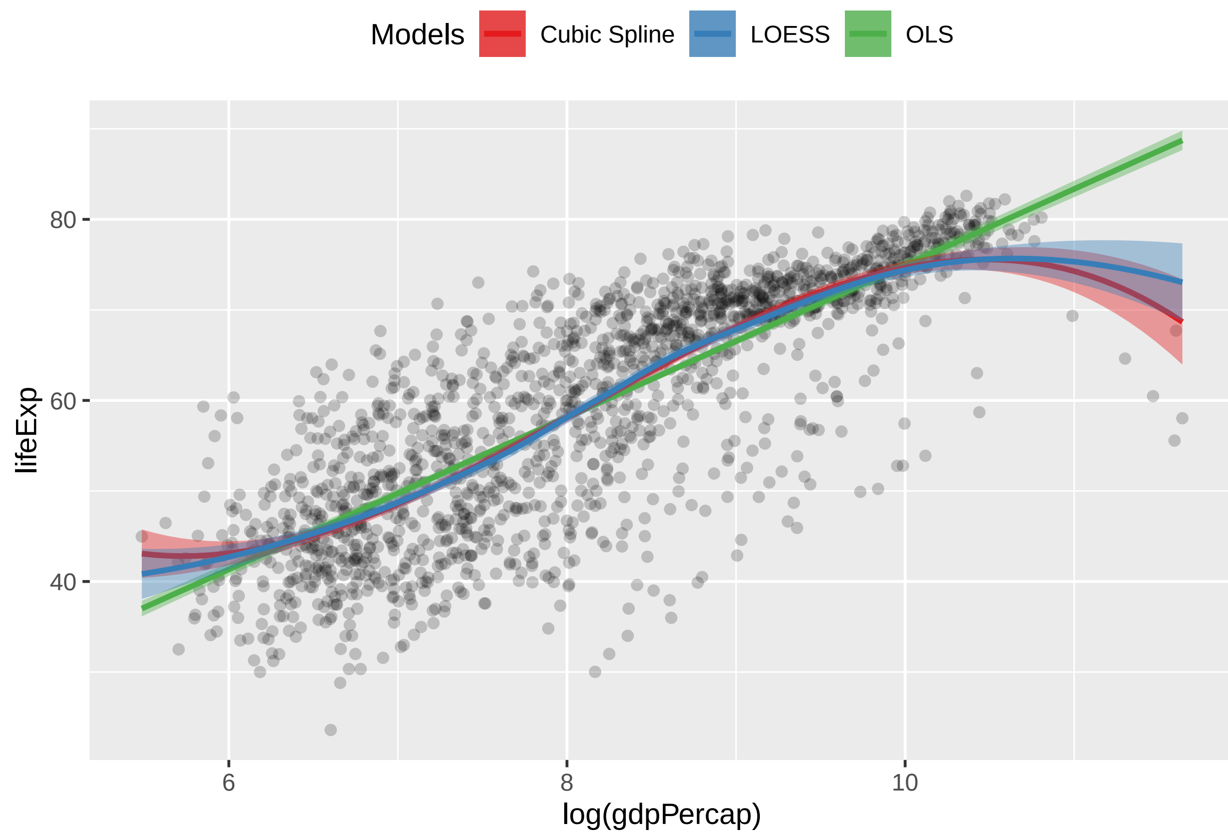
Show several fits at once, with a legend

```
model_colors <- RColorBrewer::brewer.pal(3, "Set1")
model_colors
```

```
## [1] "#E41A1C" "#377EB8" "#4DAF4A"
```

```
p0 <- ggplot(data = gapminder,
             mapping = aes(x = log(gdpPercap), y = lifeExp))

p1 <- p0 + geom_point(alpha = 0.2) +
    geom_smooth(method = "lm", aes(color = "OLS", fill = "OLS")) +
    geom_smooth(method = "lm", formula = y ~ splines::bs(x, df = 3),
                aes(color = "Cubic Spline", fill = "Cubic Spline")) +
    geom_smooth(method = "loess",
                aes(color = "LOESS", fill = "LOESS"))


p1 + scale_color_manual(name = "Models", values = model_colors) +
    scale_fill_manual(name = "Models", values = model_colors) +
    theme(legend.position = "top")
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



Look inside model objects

```
## tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
## $ country : Factor w/ 142 levels "Afghanistan",..: 1 1 ...
## $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 ...
## $ year : int [1:1704] 1952 1957 ...
## $ lifeExp : num [1:1704] 28.8 ...
## $ pop : int [1:1704] 8425333 9240934 ...
## $ gdpPercap: num [1:1704] 779 ...
```

```
out <- lm(formula = lifeExp ~ gdpPercap + pop + continent,
          data = gapminder)
```

```
summary(out)
```

```
##
## Call:
## lm(formula = lifeExp ~ gdpPercap + pop + continent, data = gapminder)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.161  -4.486   0.297   5.110  25.175
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        4.781e+01  3.395e-01 140.819  < 2e-16 ***
## gdpPercap          4.495e-04  2.346e-05  19.158  < 2e-16 ***
```

```
## pop               6.570e-09  1.975e-09   3.326 0.000901 ***
## continentAmericas 1.348e+01  6.000e-01  22.458  < 2e-16 ***
## continentAsia     8.193e+00  5.712e-01  14.342  < 2e-16 ***
## continentEurope   1.747e+01  6.246e-01  27.973  < 2e-16 ***
## continentOceania  1.808e+01  1.782e+00  10.146  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.365 on 1697 degrees of freedom
## Multiple R-squared:  0.5821, Adjusted R-squared:  0.5806
## F-statistic: 393.9 on 6 and 1697 DF,  p-value: < 2.2e-16
```

Generate predictions to graph

```
min_gdp <- min(gapminder$gdpPercap)
max_gdp <- max(gapminder$gdpPercap)
med_pop <- median(gapminder$pop)

pred_df <- expand.grid(gdpPercap = (seq(from = min_gdp,
                                        to = max_gdp,
                                        length.out = 100)),
                       pop = med_pop,
                       continent = c("Africa", "Americas",
                                     "Asia", "Europe", "Oceania"))

dim(pred_df)
```

```
## [1] 500   3
```

```
head(pred_df)
```

```
##   gdpPercap     pop continent
## 1  241.1659 7023596    Africa
## 2 1385.4282 7023596    Africa
## 3 2529.6905 7023596    Africa
## 4 3673.9528 7023596    Africa
## 5 4818.2150 7023596    Africa
## 6 5962.4773 7023596    Africa
```

```
pred_out <- predict(object = out,
                    newdata = pred_df,
                    interval = "predict")
head(pred_out)
```

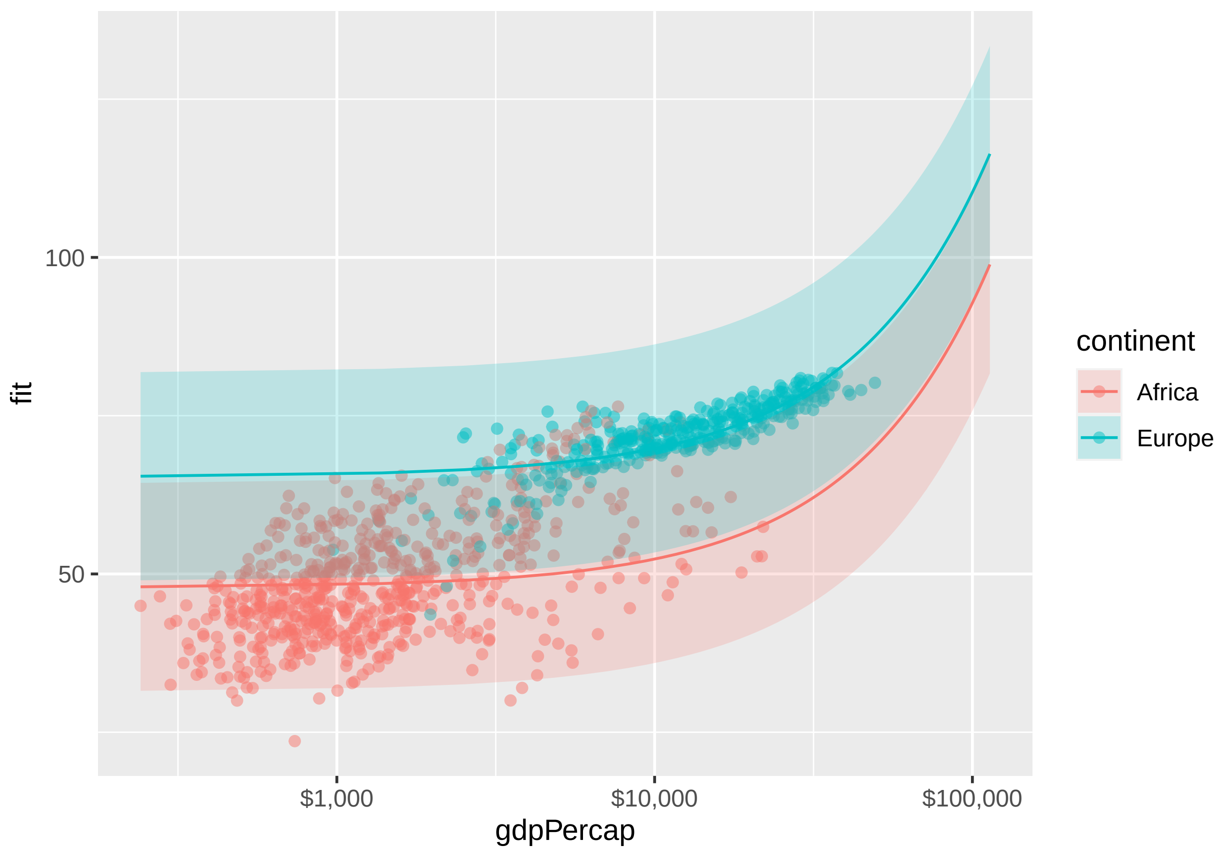
```
##        fit      lwr      upr
## 1 47.96863 31.54775 64.38951
## 2 48.48298 32.06231 64.90365
## 3 48.99733 32.57670 65.41797
## 4 49.51169 33.09092 65.93245
## 5 50.02604 33.60497 66.44711
## 6 50.54039 34.11885 66.96193
```

```
pred_df <- cbind(pred_df, pred_out)
head(pred_df)
```

```
##   gdpPercap     pop continent      fit      lwr      upr
```

```
## 1  241.1659 7023596    Africa 47.96863 31.54775 64.38951
## 2 1385.4282 7023596    Africa 48.48298 32.06231 64.90365
## 3 2529.6905 7023596    Africa 48.99733 32.57670 65.41797
## 4 3673.9528 7023596    Africa 49.51169 33.09092 65.93245
## 5 4818.2150 7023596    Africa 50.02604 33.60497 66.44711
## 6 5962.4773 7023596    Africa 50.54039 34.11885 66.96193
```

```
p <- ggplot(data = subset(pred_df, continent %in% c("Europe", "Africa")),
            aes(x = gdpPercap,
                y = fit, ymin = lwr, ymax = upr,
                color = continent,
                fill = continent,
                group = continent))

p + geom_point(data = subset(gapminder,
                             continent %in% c("Europe", "Africa")),
               aes(x = gdpPercap, y = lifeExp,
                   color = continent),
               alpha = 0.5,
               inherit.aes = FALSE) +
    geom_line() +
    geom_ribbon(alpha = 0.2, color = FALSE) +
    scale_x_log10(labels = scales::dollar)
```



Tidy model objects with broom

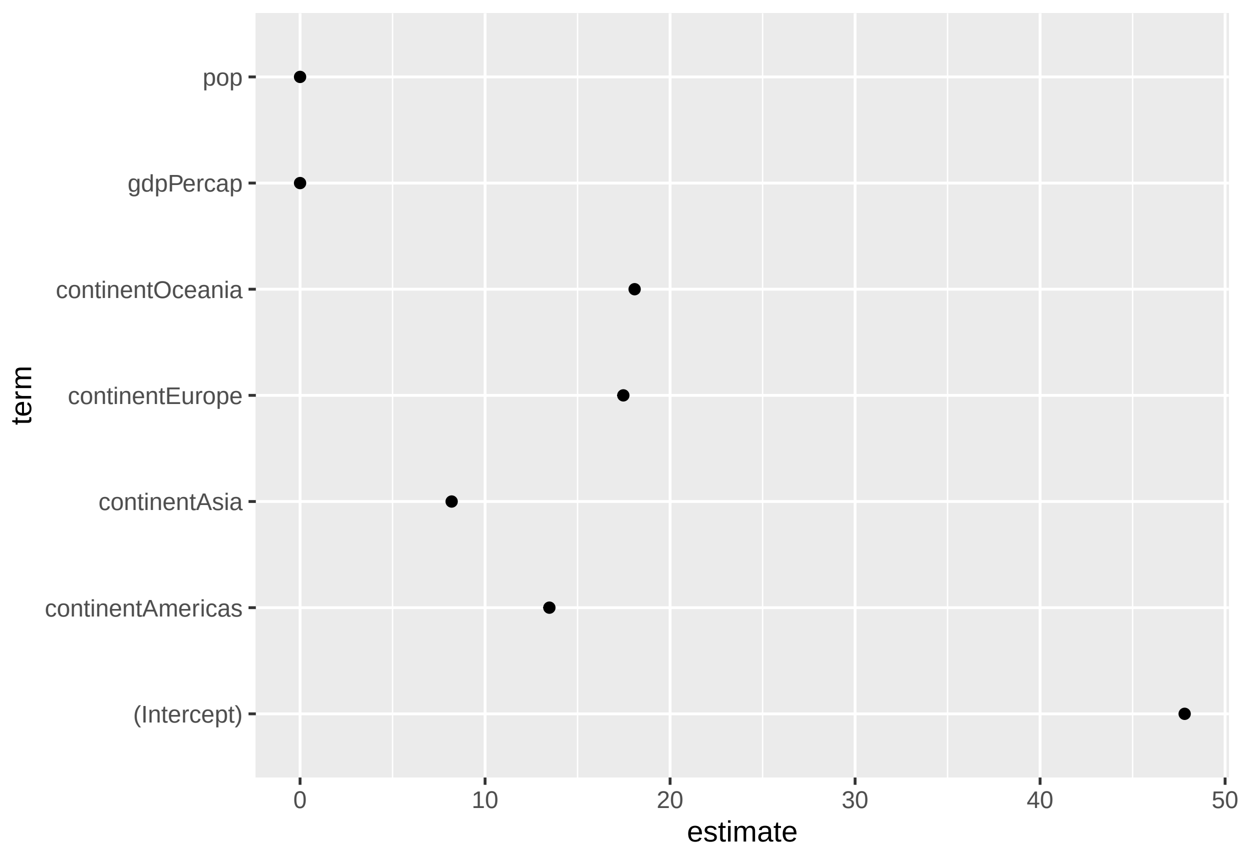
```
library(broom)
```

```
## Warning: package 'broom' was built under R version 3.6.3
```

```
out_comp <- tidy(out)
out_comp %>% round_df()
```

```
## # A tibble: 7 x 5
##   term              estimate std.error statistic p.value
##   <chr>                <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)          47.8       0.34     141.        0
## 2 gdpPercap             0         0         19.2       0
## 3 pop                   0         0          3.33      0
## 4 continentAmericas    13.5       0.6       22.5       0
## 5 continentAsia         8.19      0.57      14.3       0
## 6 continentEurope      17.5       0.62      28.0       0
## 7 continentOceania     18.1       1.78      10.2       0
```

```
p <- ggplot(out_comp, mapping = aes(x = term,
                                    y = estimate))

p + geom_point() + coord_flip()
```

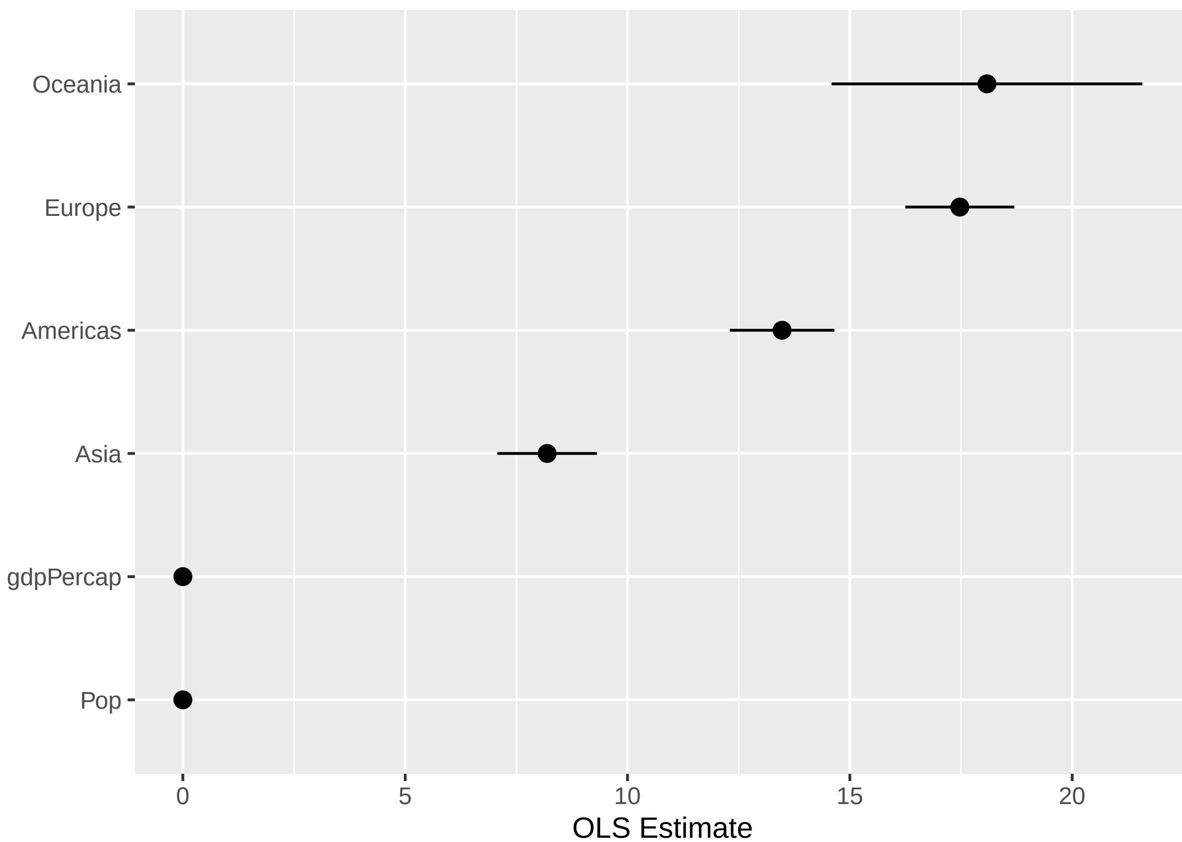



```
out_conf <- tidy(out, conf.int = TRUE)
out_conf %>% round_df()
```

```
## # A tibble: 7 x 7
##   term                estimate std.error statistic p.value conf.low conf.high
##   <chr>                  <dbl>     <dbl>     <dbl>   <dbl>    <dbl>     <dbl>
## 1 (Intercept)            47.8       0.34    141.         0    47.2      48.5
## 2 gdpPercap               0         0        19.2        0     0         0
## 3 pop                     0         0         3.33       0     0         0
## 4 continentAmericas      13.5       0.6      22.5        0    12.3      14.6
## 5 continentAsia           8.19      0.57     14.3        0     7.07      9.31
## 6 continentEurope        17.5       0.62     28.0        0    16.2      18.7
## 7 continentOceania       18.1       1.78     10.2        0    14.6      21.6
```

```
## out_conf <- subset(out_conf, term %nin% "(Intercept)")
## out_conf$nicelabs <- prefix_strip(out_conf$term, "continent")

out_conf <- out_conf %>%
    filter(term %nin% "(Intercept)") %>%
    mutate(nicelabs = prefix_strip(term, "continent")) %>%
    select(nicelabs, everything())
```

```
p <- ggplot(out_conf, mapping = aes(x = reorder(nicelabs, estimate),
                                    y = estimate, ymin = conf.low, ymax = conf.high))
p + geom_pointrange() + coord_flip() + labs(x="", y="OLS Estimate")
```



Get observation-level statistics with augment()

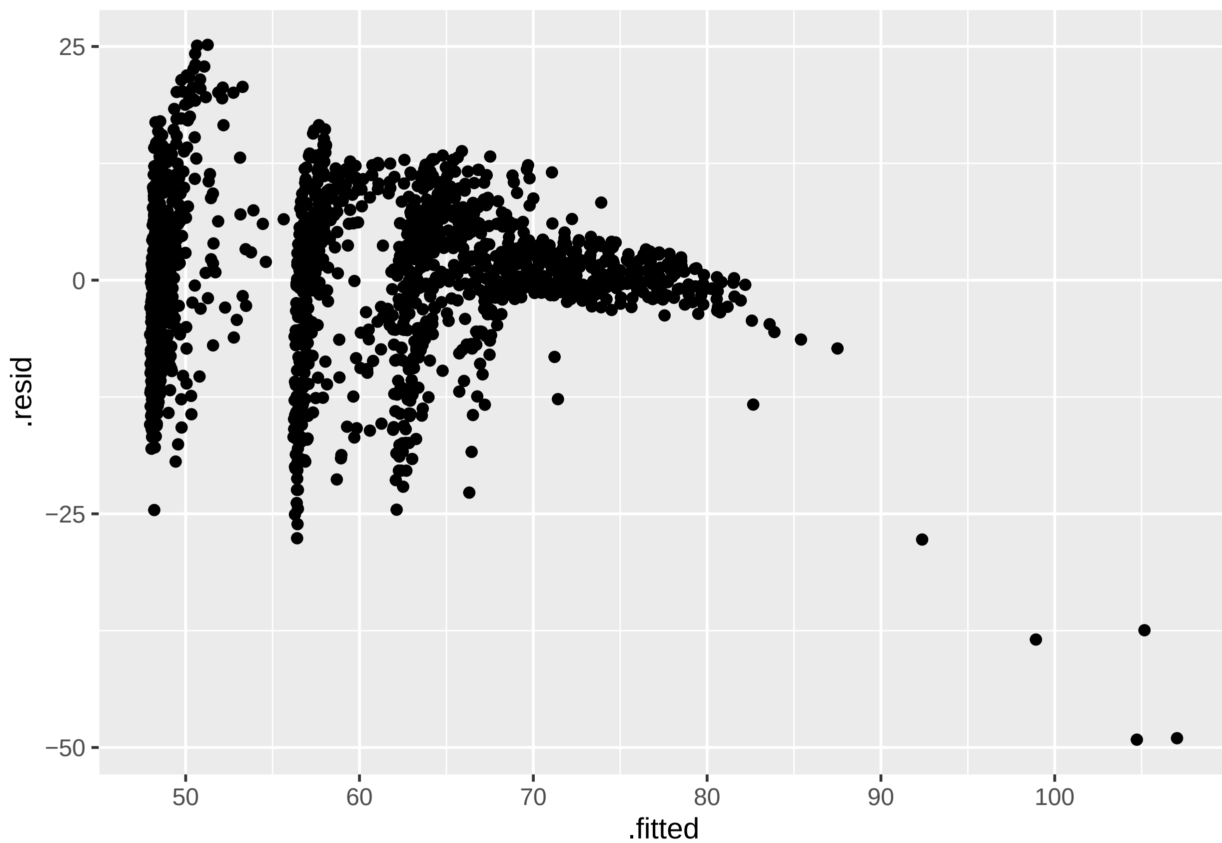
```r
out_aug <- augment(out)
head(out_aug) %>% round_df()
```

```
## # A tibble: 6 x 10
##   lifeExp gdpPercap      pop continent .fitted .resid  .hat .sigma .cooksd
##     <dbl>     <dbl>    <dbl> <fct>       <dbl>  <dbl> <dbl>  <dbl>   <dbl>
## 1    28.8      779.  8425333 Asia         56.4  -27.6     0   8.34    0.01
## 2    30.3      821.  9240934 Asia         56.4  -26.1     0   8.34    0
## 3    32        853. 10267083 Asia         56.5  -24.5     0   8.35    0
## 4    34.0      836. 11537966 Asia         56.5  -22.4     0   8.35    0
## 5    36.1      740. 13079460 Asia         56.4  -20.3     0   8.35    0
## 6    38.4      786. 14880372 Asia         56.5  -18.0     0   8.36    0
## # ... with 1 more variable: .std.resid <dbl>
```

```r
out_aug <- augment(out, data = gapminder)
head(out_aug) %>% round_df()
```

```
## # A tibble: 6 x 12
##   country   continent  year lifeExp    pop gdpPercap .fitted .resid  .hat .sigma
##   <fct>     <fct>     <dbl>   <dbl>  <dbl>     <dbl>   <dbl>  <dbl> <dbl>  <dbl>
## 1 Afghanis~ Asia       1952    28.8 8.43e6      779.    56.4  -27.6     0   8.34
## 2 Afghanis~ Asia       1957    30.3 9.24e6      821.    56.4  -26.1     0   8.34
## 3 Afghanis~ Asia       1962    32   1.03e7      853.    56.5  -24.5     0   8.35
## 4 Afghanis~ Asia       1967    34.0 1.15e7      836.    56.5  -22.4     0   8.35
## 5 Afghanis~ Asia       1972    36.1 1.31e7      740.    56.4  -20.3     0   8.35
## 6 Afghanis~ Asia       1977    38.4 1.49e7      786.    56.5  -18.0     0   8.36
## # ... with 2 more variables: .cooksd <dbl>, .std.resid <dbl>
```

```r
p <- ggplot(data = out_aug,
            mapping = aes(x = .fitted, y = .resid))
p + geom_point()
```



Get model-level statistics with glance()

```
glance(out) %>% round_df()
```

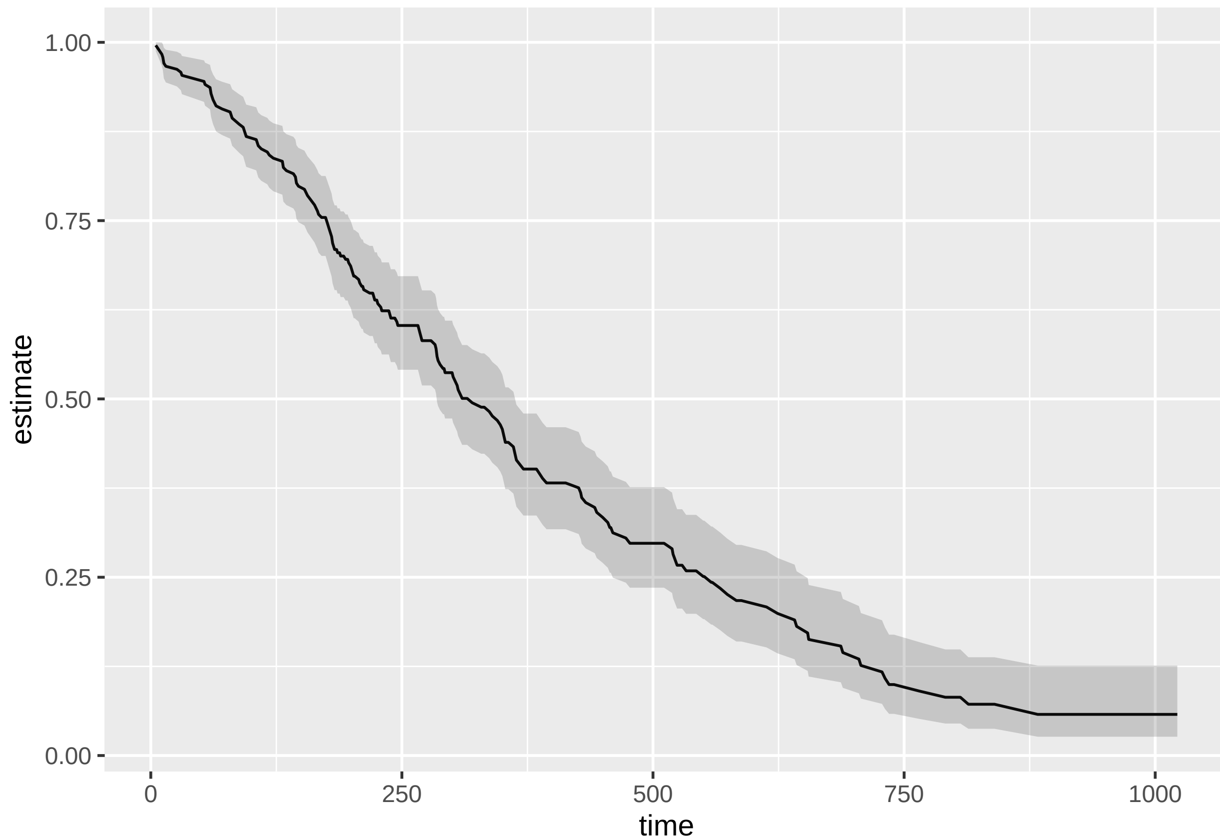
```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl>  <dbl>  <dbl>
## 1      0.58          0.58  8.37      394.       0     6 -6034. 12084. 12127.
## # ... with 3 more variables: deviance <dbl>, df.residual <dbl>, nobs <dbl>
```

```
library(survival)
```

```
## Warning: package 'survival' was built under R version 3.6.3
```

```
out_cph <- coxph(Surv(time, status) ~ age + sex, data = lung)
out_surv <- survfit(out_cph)
```

```
out_tidy <- tidy(out_surv)

p <- ggplot(data = out_tidy, mapping = aes(time, estimate))
p + geom_line() +
    geom_ribbon(mapping = aes(ymin = conf.low, ymax = conf.high), alpha = .2)
```



Grouped analysis and list columns

```
eu77 <- gapminder %>% filter(continent == "Europe", year == 1977)
```

```
fit <- lm(lifeExp ~ log(gdpPercap), data = eu77)
summary(fit)
```

```
##
## Call:
## lm(formula = lifeExp ~ log(gdpPercap), data = eu77)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4956 -1.0306  0.0935  1.1755  3.7125
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      29.489      7.161   4.118 0.000306 ***
## log(gdpPercap)    4.488      0.756   5.936 2.17e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.114 on 28 degrees of freedom
## Multiple R-squared:  0.5572, Adjusted R-squared:  0.5414
## F-statistic: 35.24 on 1 and 28 DF,  p-value: 2.173e-06
```

```r
out_le <- gapminder %>%
    group_by(continent, year) %>%
    nest()

out_le
```

```
## # A tibble: 60 x 3
## # Groups:   continent, year [60]
##    continent  year data
##    <fct>     <int> <list>
##  1 Asia       1952 <tibble [33 x 4]>
##  2 Asia       1957 <tibble [33 x 4]>
##  3 Asia       1962 <tibble [33 x 4]>
##  4 Asia       1967 <tibble [33 x 4]>
##  5 Asia       1972 <tibble [33 x 4]>
##  6 Asia       1977 <tibble [33 x 4]>
##  7 Asia       1982 <tibble [33 x 4]>
##  8 Asia       1987 <tibble [33 x 4]>
##  9 Asia       1992 <tibble [33 x 4]>
## 10 Asia       1997 <tibble [33 x 4]>
## # ... with 50 more rows
```

```r
out_le %>% filter(continent == "Europe" & year == 1977) %>%
    unnest(cols = c(data))
```

```
## # A tibble: 30 x 6
## # Groups:   continent, year [1]
##    continent  year country                lifeExp      pop gdpPercap
##    <fct>     <int> <fct>                    <dbl>    <int>     <dbl>
##  1 Europe     1977 Albania                   68.9  2509048     3533.
##  2 Europe     1977 Austria                   72.2  7568430    19749.
##  3 Europe     1977 Belgium                   72.8  9821800    19118.
##  4 Europe     1977 Bosnia and Herzegovina    69.9  4086000     3528.
##  5 Europe     1977 Bulgaria                  70.8  8797022     7612.
##  6 Europe     1977 Croatia                   70.6  4318673    11305.
##  7 Europe     1977 Czech Republic            70.7 10161915    14800.
##  8 Europe     1977 Denmark                   74.7  5088419    20423.
##  9 Europe     1977 Finland                   72.5  4738902    15605.
## 10 Europe     1977 France                    73.8 53165019    18293.
## # ... with 20 more rows
```

```r
fit_ols <- function(df) {
    lm(lifeExp ~ log(gdpPercap), data = df)
}

out_le <- gapminder %>%
    group_by(continent, year) %>%
    nest() %>%
    mutate(model = map(data, fit_ols))

out_le
```

```
## # A tibble: 60 x 4
## # Groups:   continent, year [60]
##    continent  year data              model
```

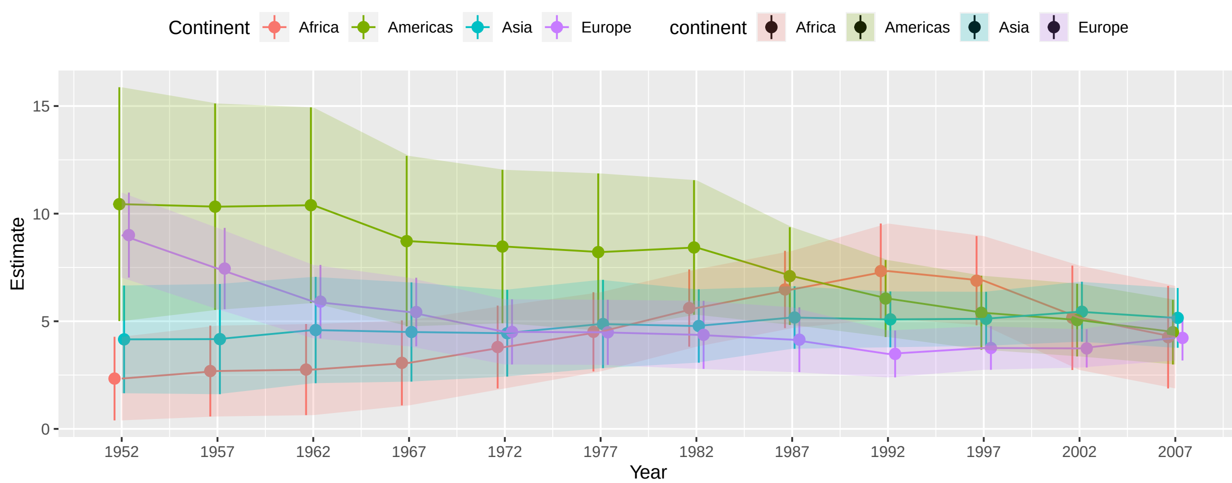
```
##    <fct>     <int> <list>            <list>
##  1 Asia       1952 <tibble [33 x 4]> <lm>
##  2 Asia       1957 <tibble [33 x 4]> <lm>
##  3 Asia       1962 <tibble [33 x 4]> <lm>
##  4 Asia       1967 <tibble [33 x 4]> <lm>
##  5 Asia       1972 <tibble [33 x 4]> <lm>
##  6 Asia       1977 <tibble [33 x 4]> <lm>
##  7 Asia       1982 <tibble [33 x 4]> <lm>
##  8 Asia       1987 <tibble [33 x 4]> <lm>
##  9 Asia       1992 <tibble [33 x 4]> <lm>
## 10 Asia       1997 <tibble [33 x 4]> <lm>
## # ... with 50 more rows
```

```
fit_ols <- function(df) {
    lm(lifeExp ~ log(gdpPercap), data = df)
}

out_tidy <- gapminder %>%
    group_by(continent, year) %>%
    nest() %>%
    mutate(model = map(data, fit_ols),
           tidied = map(model, tidy)) %>%
    unnest(cols = c(tidied)) %>%
    filter(term %nin% "(Intercept)" &
           continent %nin% "Oceania")

out_tidy %>%
    ungroup() %>%
    sample_n(5)
```

```
## # A tibble: 5 x 9
##   continent  year data      model  term    estimate std.error statistic  p.value
##   <fct>     <int> <list>    <list> <chr>      <dbl>     <dbl>     <dbl>    <dbl>
## 1 Europe     2002 <tibble ~ <lm>   log(gd~     3.74     0.445      8.40  3.91e-9
## 2 Africa     1997 <tibble ~ <lm>   log(gd~     6.89     1.04       6.64  2.16e-8
## 3 Asia       1987 <tibble ~ <lm>   log(gd~     5.17     0.727      7.12  5.31e-8
## 4 Americas   2002 <tibble ~ <lm>   log(gd~     5.05     0.844      5.99  4.18e-6
## 5 Europe     1997 <tibble ~ <lm>   log(gd~     3.76     0.507      7.42  4.45e-8
```

```
p <- ggplot(data = out_tidy,
            mapping = aes(x = year, y = estimate,
                          ymin = estimate - 2*std.error,
                          ymax = estimate + 2*std.error,
                          color = continent, group = continent,
                          fill = continent))


p + geom_pointrange(position = position_dodge(width = 1)) +
    geom_line() +
    geom_ribbon(mapping = aes(x = year,
                          ymin = estimate - 2*std.error,
                          ymax = estimate + 2*std.error,
                          group = continent,
                          fill = continent),
                 alpha = 0.2,
```

```
                inherit.aes = FALSE) +
    scale_x_continuous(breaks = unique(gapminder$year)) +
    theme(legend.position = "top") +
    labs(x = "Year", y = "Estimate", color = "Continent")
```



Grouped Analysis: PCA Example

On the full dataset ...

```
mw_pca <- midwest %>%
    group_by(state) %>%
    select_if(is.numeric) %>%
    select(-PID)

mw_pca
```

```
## # A tibble: 437 x 25
## # Groups:   state [5]
##    state  area poptotal popdensity popwhite popblack popamerindian popasian
##    <chr> <dbl>    <int>      <dbl>    <int>    <int>         <int>    <int>
##  1 IL    0.052    66090      1271.    63917     1702            98      249
##  2 IL    0.014    10626       759      7054     3496            19       48
##  3 IL    0.022    14991       681.    14477      429            35       16
##  4 IL    0.017    30806      1812.    29344      127            46      150
##  5 IL    0.018     5836       324.     5264      547            14        5
##  6 IL    0.05     35688       714.    35157       50            65      195
##  7 IL    0.017     5322       313.     5298        1             8       15
##  8 IL    0.027    16805       622.    16519      111            30       61
##  9 IL    0.024    13437       560.    13384       16             8       23
## 10 IL    0.058   173025      2983.   146506    16559           331     8033
## # ... with 427 more rows, and 17 more variables: popother <int>,
## #   percwhite <dbl>, percblack <dbl>, percamerindan <dbl>, percasian <dbl>,
## #   percother <dbl>, popadults <int>, perchsd <dbl>, percollege <dbl>,
## #   percprof <dbl>, poppovertyknown <int>, percpovertyknown <dbl>,
## #   percbelowpoverty <dbl>, percchildbelowpovert <dbl>, percadultpoverty <dbl>,
## #   percelderlypoverty <dbl>, inmetro <int>
```

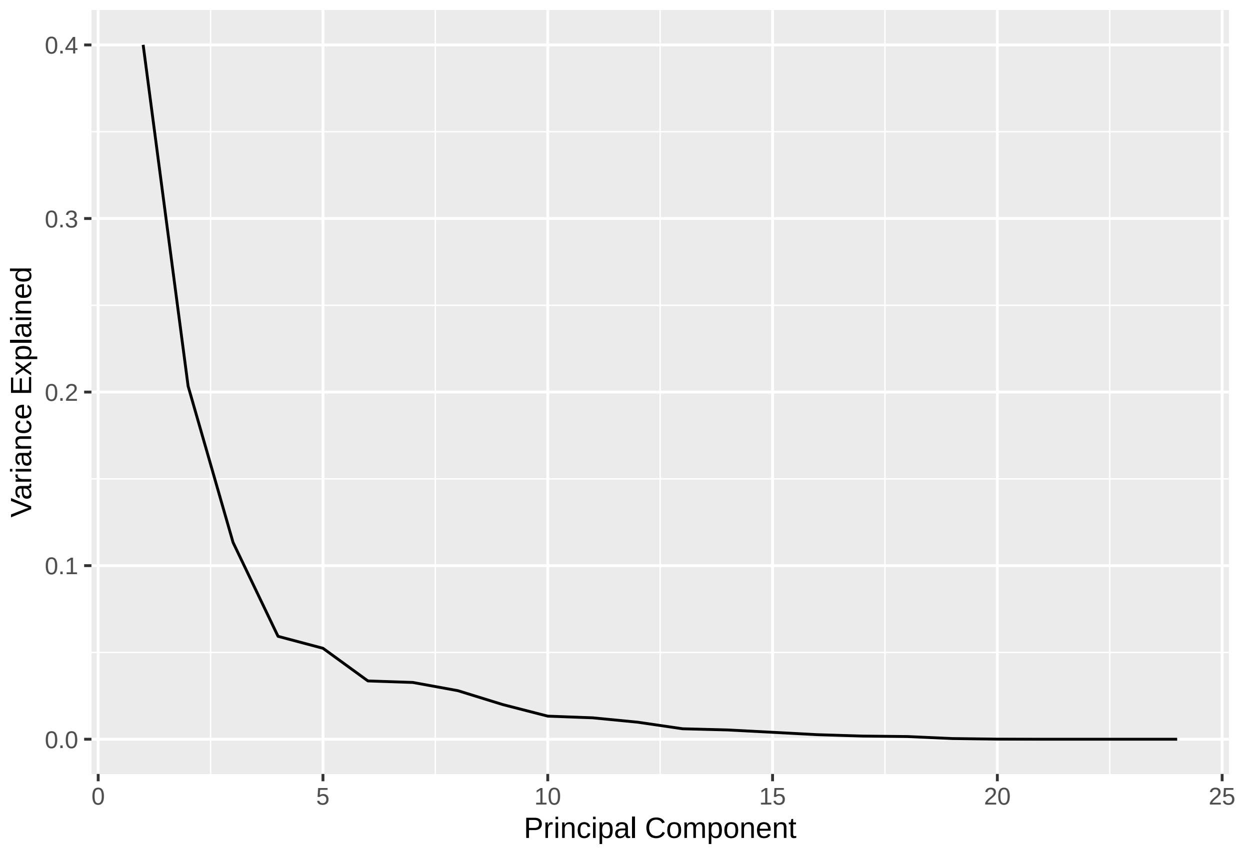
```r
do_pca <- function(df){
  prcomp(df,
         center = TRUE, scale = TRUE)
}

out_pca <- mw_pca %>%
    ungroup() %>%
    select(-state) %>%
    do_pca()

summary(out_pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6    PC7
## Standard deviation     3.0986 2.2096 1.6495 1.19289 1.12159 0.89776 0.8859
## Proportion of Variance 0.4001 0.2034 0.1134 0.05929 0.05241 0.03358 0.0327
## Cumulative Proportion  0.4001 0.6035 0.7168 0.77614 0.82856 0.86214 0.8948
##                            PC8     PC9   PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.81948 0.69212 0.5650 0.54394 0.48541 0.38000 0.35833
## Proportion of Variance 0.02798 0.01996 0.0133 0.01233 0.00982 0.00602 0.00535
## Cumulative Proportion  0.92283 0.94278 0.9561 0.96841 0.97823 0.98425 0.98960
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.30948 0.25009 0.20879 0.19244 0.09654 0.03473 0.01328
## Proportion of Variance 0.00399 0.00261 0.00182 0.00154 0.00039 0.00005 0.00001
## Cumulative Proportion  0.99359 0.99619 0.99801 0.99955 0.99994 0.99999 1.00000
##                            PC22      PC23      PC24
## Standard deviation     0.003862 2.886e-09 5.193e-16
## Proportion of Variance 0.000000 0.000e+00 0.000e+00
## Cumulative Proportion  1.000000 1.000e+00 1.000e+00
```

```r
tidy_pca <- tidy(out_pca, matrix = "pcs")

tidy_pca
```

```
## # A tibble: 24 x 4
##       PC std.dev percent cumulative
##    <dbl>   <dbl>   <dbl>      <dbl>
##  1     1   3.10   0.400       0.400
##  2     2   2.21   0.203       0.603
##  3     3   1.65   0.113       0.717
##  4     4   1.19   0.0593      0.776
##  5     5   1.12   0.0524      0.829
##  6     6   0.898  0.0336      0.862
##  7     7   0.886  0.0327      0.895
##  8     8   0.819  0.0280      0.923
##  9     9   0.692  0.0200      0.943
## 10    10   0.565  0.0133      0.956
## # ... with 14 more rows
```

```r
tidy_pca %>%
    ggplot(aes(x = PC, y = percent)) +
    geom_line() +
    labs(x = "Principal Component", y = "Variance Explained")
```



... or nested by state

```
mw_pca <- mw_pca %>%
    group_by(state) %>%
    nest()

mw_pca
```

```
## # A tibble: 5 x 2
## # Groups:   state [5]
##   state data
##   <chr> <list>
## 1 IL    <tibble [102 x 24]>
## 2 IN    <tibble [92 x 24]>
## 3 MI    <tibble [83 x 24]>
## 4 OH    <tibble [88 x 24]>
## 5 WI    <tibble [72 x 24]>
```

```
state_pca <- mw_pca %>%
    mutate(pca = map(data, do_pca))

state_pca
```

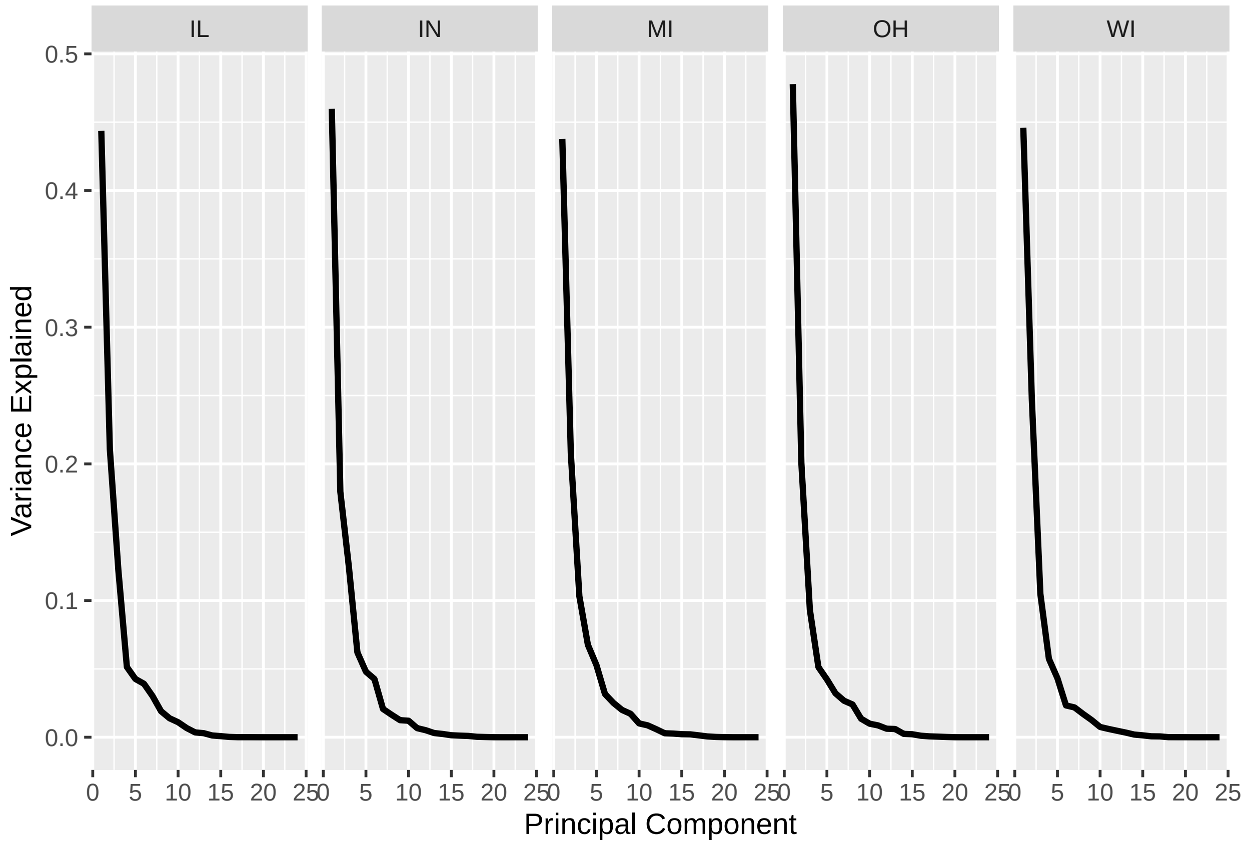
```
## # A tibble: 5 x 3
## # Groups:   state [5]
##   state data                pca
```

```
##   <chr> <list>               <list>
## 1 IL    <tibble [102 x 24]> <prcomp>
## 2 IN    <tibble [92 x 24]>  <prcomp>
## 3 MI    <tibble [83 x 24]>  <prcomp>
## 4 OH    <tibble [88 x 24]>  <prcomp>
## 5 WI    <tibble [72 x 24]>  <prcomp>
```

```
do_tidy <- function(pr){
    broom::tidy(pr, matrix = "pcs")
}
state_pca  <- mw_pca %>%
    mutate(pca = map(data, do_pca),
           pcs = map(pca, do_tidy))

state_pca
```

```
## # A tibble: 5 x 4
## # Groups:   state [5]
##   state data                pca      pcs
##   <chr> <list>              <list>   <list>
## 1 IL    <tibble [102 x 24]> <prcomp> <tibble [24 x 4]>
## 2 IN    <tibble [92 x 24]>  <prcomp> <tibble [24 x 4]>
## 3 MI    <tibble [83 x 24]>  <prcomp> <tibble [24 x 4]>
## 4 OH    <tibble [88 x 24]>  <prcomp> <tibble [24 x 4]>
## 5 WI    <tibble [72 x 24]>  <prcomp> <tibble [24 x 4]>
```

```
state_pca %>%
    unnest(cols = c(pcs)) %>%
    ggplot(aes(x = PC, y = percent)) +
    geom_line(size = 1.1) +
    facet_wrap(~ state, nrow = 1) +
    labs(x = "Principal Component",
         y = "Variance Explained")
```

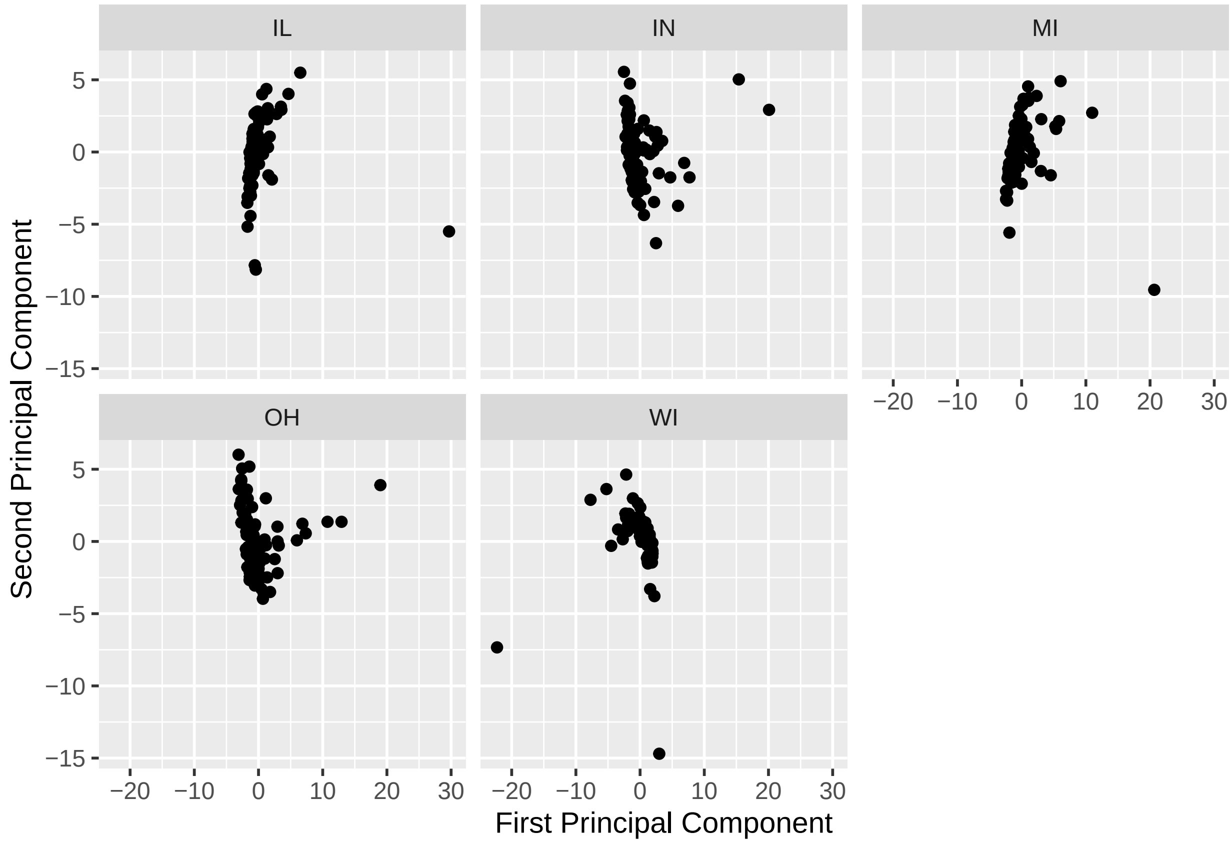



```
do_aug <- function(pr){
    broom::augment(pr)
}


state_pca  <- mw_pca %>%
    mutate(pca = map(data, do_pca),
           pcs = map(pca, do_tidy),
           fitted = map(pca, do_aug))

state_pca
```

```
## # A tibble: 5 x 5
## # Groups:   state [5]
##   state data                pca      pcs                fitted
##   <chr> <list>              <list>   <list>             <list>
## 1 IL    <tibble [102 x 24]> <prcomp> <tibble [24 x 4]> <tibble [102 x 25]>
## 2 IN    <tibble [92 x 24]>  <prcomp> <tibble [24 x 4]> <tibble [92 x 25]>
## 3 MI    <tibble [83 x 24]>  <prcomp> <tibble [24 x 4]> <tibble [83 x 25]>
## 4 OH    <tibble [88 x 24]>  <prcomp> <tibble [24 x 4]> <tibble [88 x 25]>
## 5 WI    <tibble [72 x 24]>  <prcomp> <tibble [24 x 4]> <tibble [72 x 25]>
```

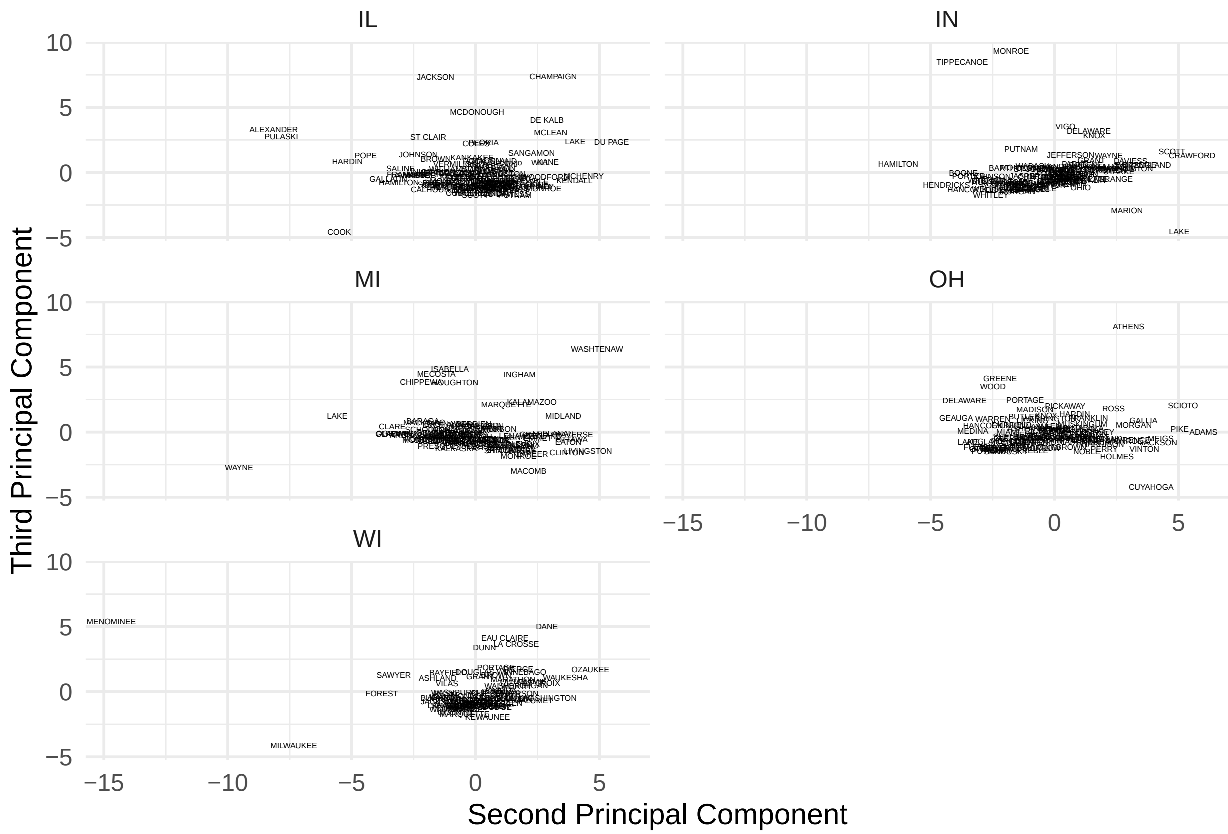
```
state_pca %>%
    unnest(cols = c(fitted)) %>%
    ggplot(aes(x = .fittedPC1,
               y = .fittedPC2)) +
    geom_point() +
```

```
    facet_wrap(~ state) +
    labs(x = "First Principal Component",
         y = "Second Principal Component")
```



Grouped PCA in a single sequence

```
midwest %>%
    group_by(state) %>%
    select_if(is.numeric) %>%
    select(-PID) %>%
    nest() %>%
    mutate(pca = map(data, do_pca),
           pcs = map(pca, do_tidy),
           fitted = map(pca, do_aug)) %>%
    unnest(cols = c(fitted)) %>%
    add_column(county = midwest$county) %>%
    ggplot(mapping = aes(x = .fittedPC2,
               y = .fittedPC3,
               label = county)) +
    geom_text(size = 1.1) +
    labs(x = "Second Principal Component",
         y = "Third Principal Component") +
    theme_minimal() + facet_wrap(~ state, ncol = 2)
```



Plot marginal effects

Note that calculating marginal effects can take some time!

```
library(margins)
```

```
## Warning: package 'margins' was built under R version 3.6.3
```

```
gss_sm$polviews_m <- relevel(gss_sm$polviews, ref = "Moderate")

out_bo <- glm(obama ~ polviews_m + sex*race,
              family = "binomial", data = gss_sm)
summary(out_bo)
```

```
##
## Call:
## glm(formula = obama ~ polviews_m + sex * race, family = "binomial",
##     data = gss_sm)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9045  -0.5541   0.1772   0.5418   2.2437
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       0.296493   0.134091   2.211  0.02703 *
## polviews_mExtremely Liberal       2.372950   0.525045   4.520 6.20e-06 ***
```

```
## polviews_mLiberal                  2.600031   0.356666   7.290 3.10e-13 ***
## polviews_mSlightly Liberal         1.293172   0.248435   5.205 1.94e-07 ***
## polviews_mSlightly Conservative   -1.355277   0.181291  -7.476 7.68e-14 ***
## polviews_mConservative            -2.347463   0.200384 -11.715  < 2e-16 ***
## polviews_mExtremely Conservative -2.727384   0.387210  -7.044 1.87e-12 ***
## sexFemale                          0.254866   0.145370   1.753  0.07956 .
## raceBlack                          3.849526   0.501319   7.679 1.61e-14 ***
## raceOther                         -0.002143   0.435763  -0.005  0.99608
## sexFemale:raceBlack               -0.197506   0.660066  -0.299  0.76477
## sexFemale:raceOther                1.574829   0.587657   2.680  0.00737 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2247.9  on 1697  degrees of freedom
## Residual deviance: 1345.9  on 1686  degrees of freedom
##   (1169 observations deleted due to missingness)
## AIC: 1369.9
##
## Number of Fisher Scoring iterations: 6
```

```
bo_m <- margins(out_bo)
summary(bo_m)
```

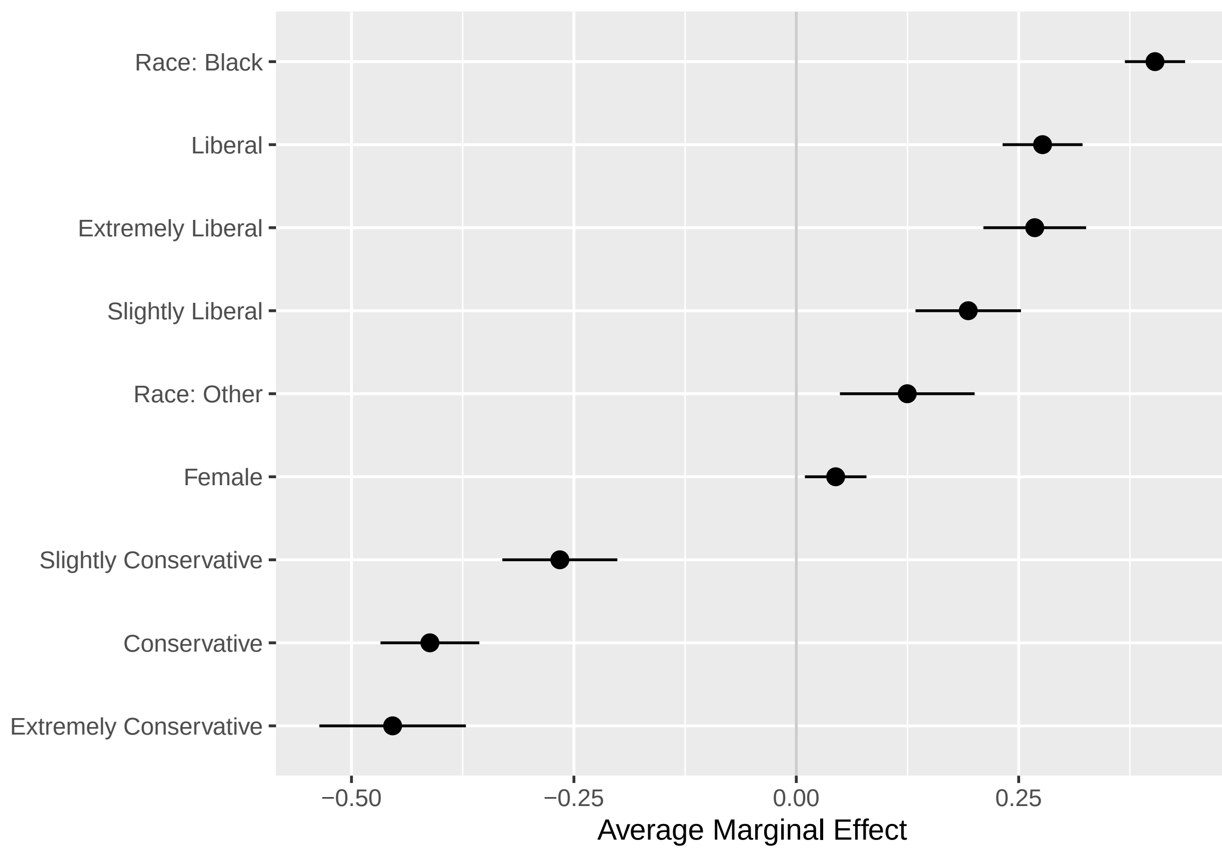
```
##                            factor     AME     SE        z      p   lower
##            polviews_mConservative -0.4119 0.0283 -14.5394 0.0000 -0.4674
##  polviews_mExtremely Conservative -0.4538 0.0420 -10.7971 0.0000 -0.5361
##       polviews_mExtremely Liberal  0.2681 0.0295   9.0996 0.0000  0.2103
##                 polviews_mLiberal  0.2768 0.0229  12.0736 0.0000  0.2319
##   polviews_mSlightly Conservative -0.2658 0.0330  -8.0596 0.0000 -0.3304
##        polviews_mSlightly Liberal  0.1933 0.0303   6.3896 0.0000  0.1340
##                         raceBlack  0.4032 0.0173  23.3568 0.0000  0.3694
##                         raceOther  0.1247 0.0386   3.2297 0.0012  0.0490
##                         sexFemale  0.0443 0.0177   2.5073 0.0122  0.0097
##    upper
##  -0.3564
##  -0.3714
##   0.3258
##   0.3218
##  -0.2011
##   0.2526
##   0.4371
##   0.2005
##   0.0789
```

```
bo_gg <- as_tibble(summary(bo_m))
prefixes <- c("polviews_m", "sex")
bo_gg$factor <- prefix_strip(bo_gg$factor, prefixes)
bo_gg$factor <- prefix_replace(bo_gg$factor, "race", "Race: ")

bo_gg %>% select(factor, AME, lower, upper)
```

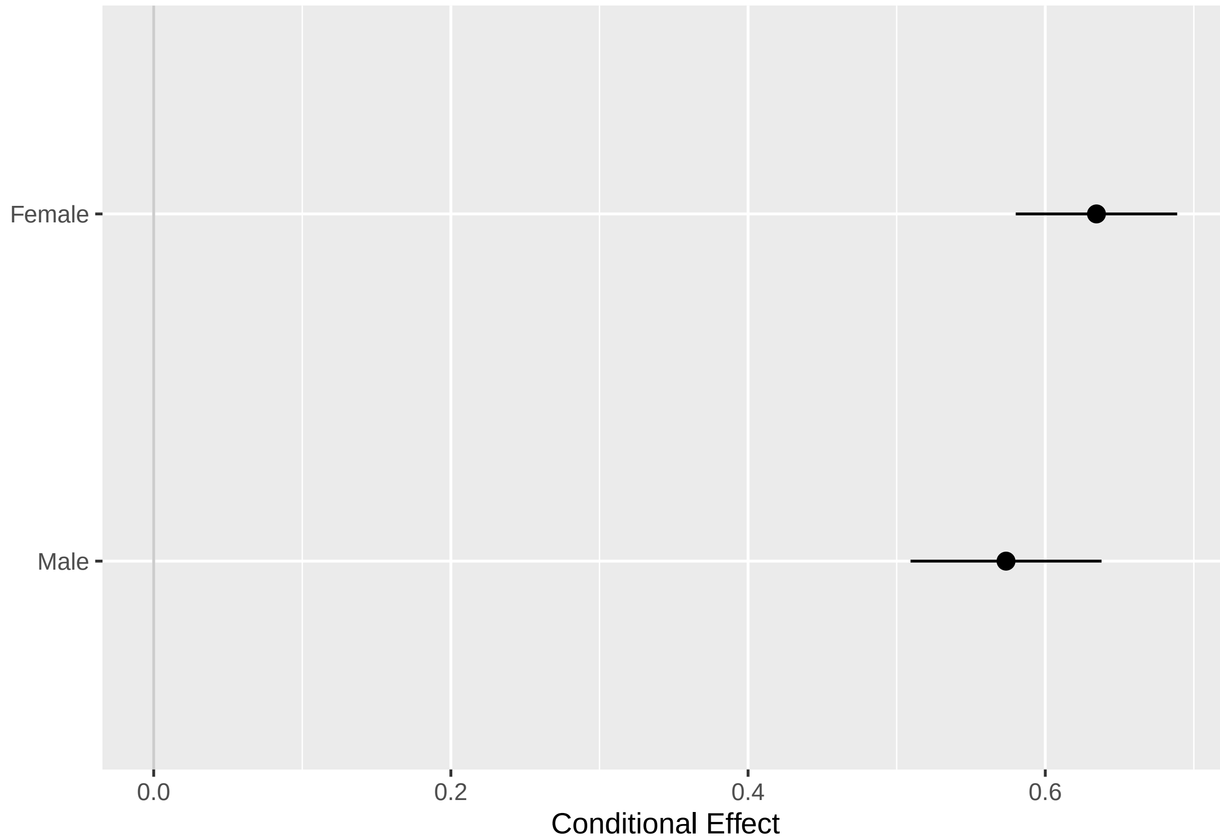
```
## # A tibble: 9 x 4
##   factor                      AME    lower   upper
```

```
##   <chr>                      <dbl>    <dbl>   <dbl>
## 1 Conservative             -0.412  -0.467   -0.356
## 2 Extremely Conservative -0.454  -0.536   -0.371
## 3 Extremely Liberal        0.268   0.210    0.326
## 4 Liberal                  0.277   0.232    0.322
## 5 Slightly Conservative  -0.266  -0.330   -0.201
## 6 Slightly Liberal         0.193   0.134    0.253
## 7 Race: Black              0.403   0.369    0.437
## 8 Race: Other              0.125   0.0490   0.200
## 9 Female                   0.0443  0.00967  0.0789
```

```
p <- ggplot(data = bo_gg, aes(x = reorder(factor, AME),
                              y = AME, ymin = lower, ymax = upper))

p + geom_hline(yintercept = 0, color = "gray80") +
    geom_pointrange() + coord_flip() +
    labs(x = NULL, y = "Average Marginal Effect")
```




```
pv_cp <- cplot(out_bo, x = "sex", draw = FALSE)

p <- ggplot(data = pv_cp, aes(x = reorder(xvals, yvals),
                              y = yvals, ymin = lower, ymax = upper))

p + geom_hline(yintercept = 0, color = "gray80") +
    geom_pointrange() + coord_flip() +
    labs(x = NULL, y = "Conditional Effect")
```



Plots from complex surveys

```
library(survey)
```

```
## Warning: package 'survey' was built under R version 3.6.3

## Loading required package: grid

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

##
## Attaching package: 'survey'

## The following object is masked from 'package:graphics':
##
##     dotchart
```

```
library(srvyr)
```

```
## Warning: package 'srvyr' was built under R version 3.6.3

##
## Attaching package: 'srvyr'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
options(survey.lonely.psu = "adjust")
options(na.action="na.pass")

gss_wt <- subset(gss_lon, year > 1974) %>%
    mutate(stratvar = interaction(year, vstrat)) %>%
    as_survey_design(ids = vpsu,
                     strata = stratvar,
                     weights = wtssall,
                     nest = TRUE)
```

```
out_grp <- gss_wt %>%
    filter(year %in% seq(1976, 2016, by = 4)) %>%
    group_by(year, race, degree) %>%
    summarize(prop = survey_mean(na.rm = TRUE))

out_grp
```

```
## # A tibble: 162 x 5
## # Groups:   year, race [30]
##     year race  degree            prop prop_se
##    <dbl> <fct> <fct>            <dbl>   <dbl>
##  1  1976 White Lt High School 0.327   0.0160
##  2  1976 White High School    0.517   0.0161
##  3  1976 White Junior College 0.0128  0.00298
##  4  1976 White Bachelor       0.101   0.00955
##  5  1976 White Graduate       0.0392  0.00642
##  6  1976 White <NA>           0.00285 0.00151
##  7  1976 Black Lt High School 0.558   0.0603
##  8  1976 Black High School    0.335   0.0476
##  9  1976 Black Junior College 0.0423  0.0192
## 10  1976 Black Bachelor       0.0577  0.0238
## # ... with 152 more rows
```

```
out_mrg <- gss_wt %>%
    filter(year %in% seq(1976, 2016, by = 4)) %>%
    mutate(racedeg = interaction(race, degree)) %>%
    group_by(year, racedeg) %>%
    summarize(prop = survey_mean(na.rm = TRUE))

out_mrg
```

```
## # A tibble: 155 x 4
## # Groups:   year [10]
##     year racedeg                 prop prop_se
##    <dbl> <fct>                  <dbl>   <dbl>
##  1  1976 White.Lt High School 0.297   0.0146
##  2  1976 Black.Lt High School 0.0470  0.00837
##  3  1976 Other.Lt High School 0.00194 0.00138
##  4  1976 White.High School    0.469   0.0159
##  5  1976 Black.High School    0.0282  0.00593
##  6  1976 Other.High School    0.00324 0.00166
##  7  1976 White.Junior College 0.0117  0.00268
```

```
## 8  1976 Black.Junior College 0.00356 0.00162
## 9  1976 White.Bachelor       0.0916  0.00883
## 10 1976 Black.Bachelor       0.00486 0.00213
## # ... with 145 more rows
```

```
out_mrg <- gss_wt %>%
    filter(year %in% seq(1976, 2016, by = 4)) %>%
    mutate(racedeg = interaction(race, degree)) %>%
    group_by(year, racedeg) %>%
    summarize(prop = survey_mean(na.rm = TRUE)) %>%
    separate(racedeg, sep = "\\.", into = c("race", "degree"))

out_mrg
```

```
## # A tibble: 155 x 5
## # Groups:   year [10]
##     year race  degree           prop prop_se
##    <dbl> <chr> <chr>           <dbl>   <dbl>
## 1  1976 White Lt High School 0.297   0.0146
## 2  1976 Black Lt High School 0.0470  0.00837
## 3  1976 Other Lt High School 0.00194 0.00138
## 4  1976 White High School    0.469   0.0159
## 5  1976 Black High School    0.0282  0.00593
## 6  1976 Other High School    0.00324 0.00166
## 7  1976 White Junior College 0.0117  0.00268
## 8  1976 Black Junior College 0.00356 0.00162
## 9  1976 White Bachelor       0.0916  0.00883
## 10 1976 Black Bachelor       0.00486 0.00213
## # ... with 145 more rows
```

```
p <- ggplot(data = subset(out_grp, race %nin% "Other"),
            mapping = aes(x = degree, y = prop,
                          ymin = prop - 2*prop_se,
                          ymax = prop + 2*prop_se,
                          fill = race,
                          color = race,
                          group = race))

dodge <- position_dodge(width=0.9)

p + geom_col(position = dodge, alpha = 0.2) +
    geom_errorbar(position = dodge, width = 0.2) +
    scale_x_discrete(labels = scales::wrap_format(10)) +
    scale_y_continuous(labels = scales::percent) +
    scale_color_brewer(type = "qual", palette = "Dark2") +
    scale_fill_brewer(type = "qual", palette = "Dark2") +
    labs(title = "Educational Attainment by Race",
         subtitle = "GSS 1976-2016",
         fill = "Race",
         color = "Race",
         x = NULL, y = "Percent") +
    facet_wrap(~ year, ncol = 2) +
    theme(legend.position = "top")
```

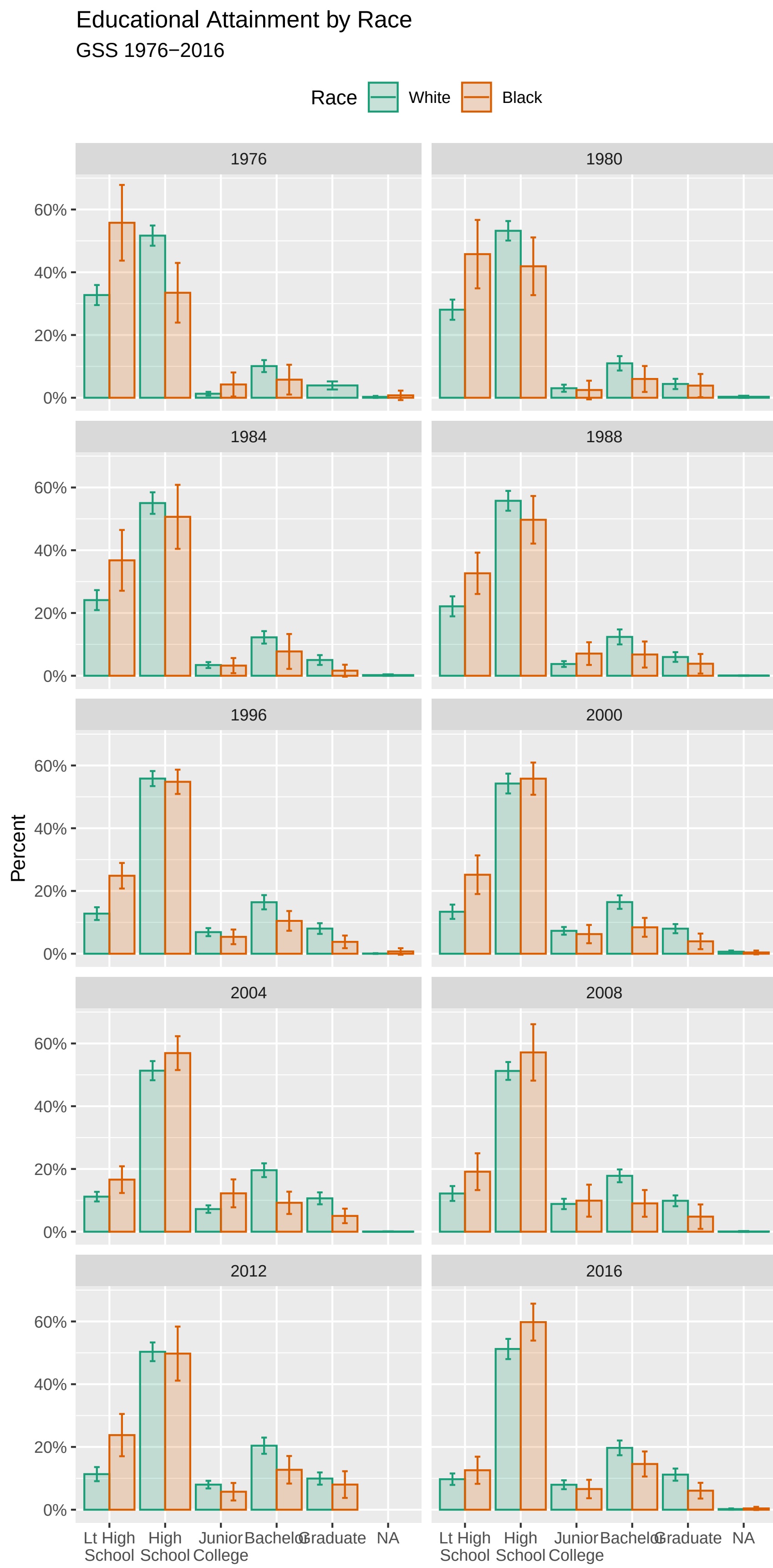


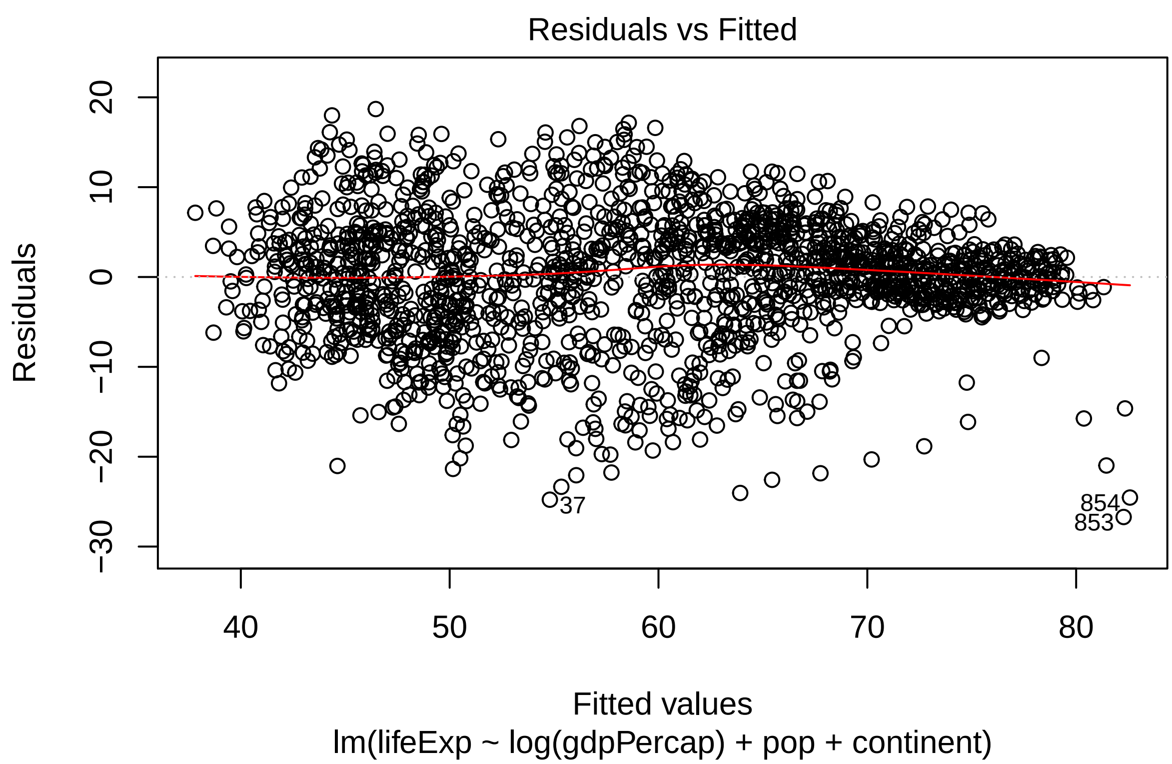
Figure 16: Weighted estimates of educational attainment for Whites and Blacks, GSS selected years 1976-2016. Faceting barplots is often a bad idea, and the more facets there are the worse an idea it is. With a small-multiple plot the viewer wants to compare across panels (in this case, over time), but this is difficult to do when the data inside the panels are categorical comparisons shown as bars (in this case, education level by group).

```r
p <- ggplot(data = subset(out_grp, race %nin% "Other"),
            mapping = aes(x = year, y = prop, ymin = prop - 2*prop_se,
                          ymax = prop + 2*prop_se, fill = race, color = race,
                          group = race))

p + geom_ribbon(alpha = 0.3, aes(color = NULL)) +
    geom_line() +
    facet_wrap(~ degree, ncol = 1) +
    scale_y_continuous(labels = scales::percent) +
    scale_color_brewer(type = "qual", palette = "Dark2") +
    scale_fill_brewer(type = "qual", palette = "Dark2") +
    labs(title = "Educational Attainment\nby Race",
         subtitle = "GSS 1976-2016", fill = "Race",
         color = "Race", x = NULL, y = "Percent") +
    theme(legend.position = "top")
```

Default plots for models

```r
out <- lm(formula = lifeExp ~ log(gdpPercap) + pop + continent, data = gapminder)
```

```r
plot(out, which = c(1,2), ask=FALSE)
```



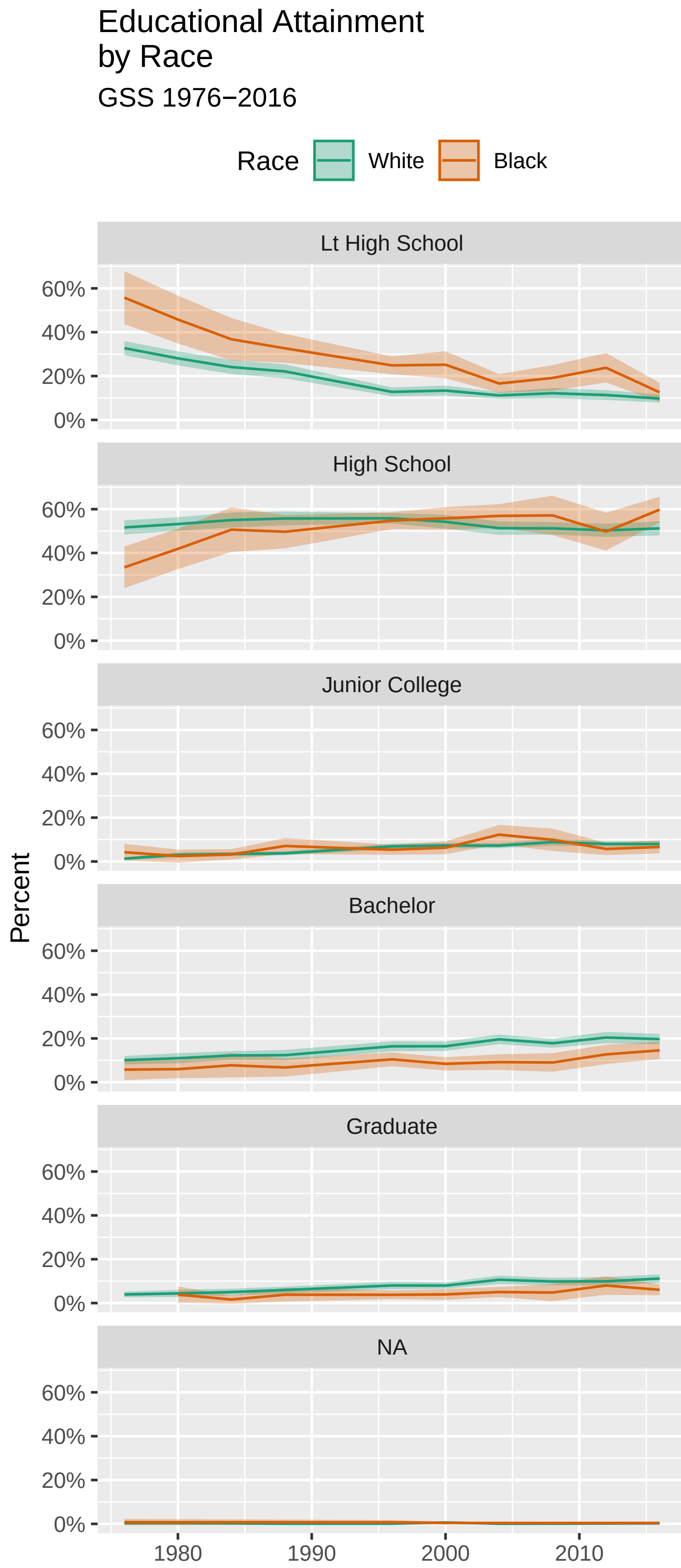
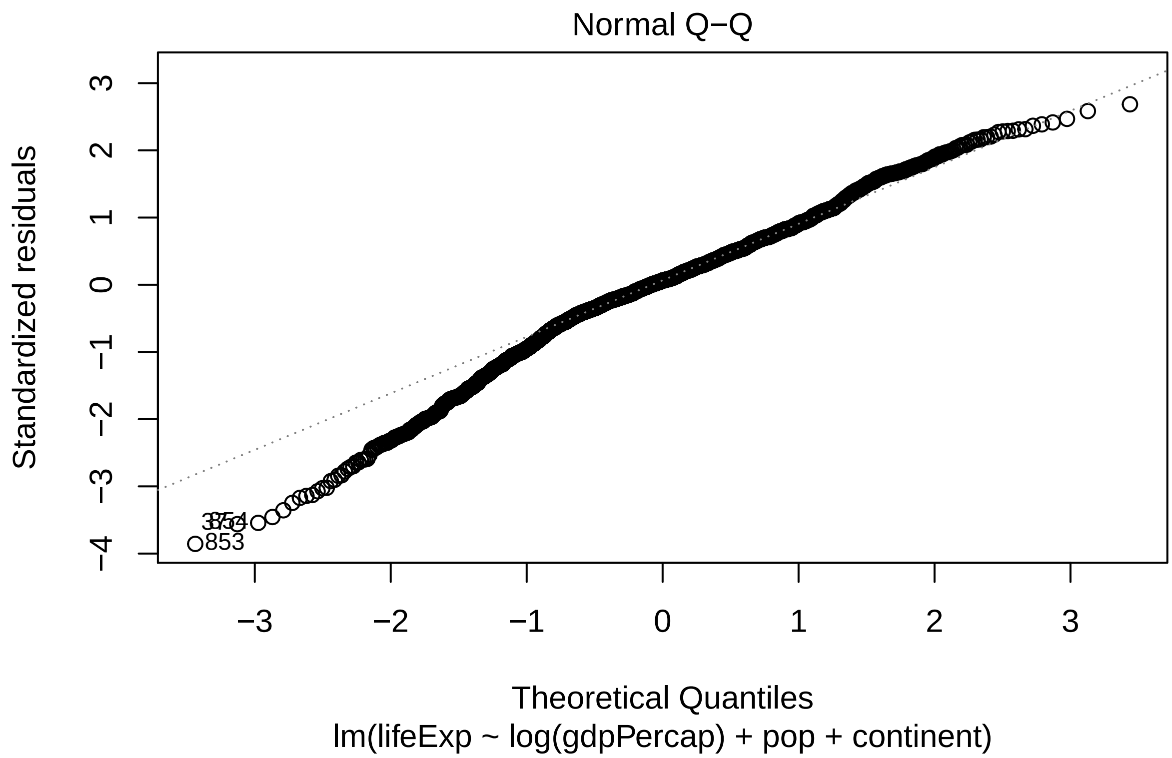


Figure 17: Faceting by education instead.


Normal Q−Q
Standardized residuals
3
2
1
0
−1
−2
−3
−4
375
354
853
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(lifeExp ~ log(gdpPercap) + pop + continent)