Report Class 5: Lab

AUTHOR
Lisanne Stouthart (PID = A69036187)

Quarto

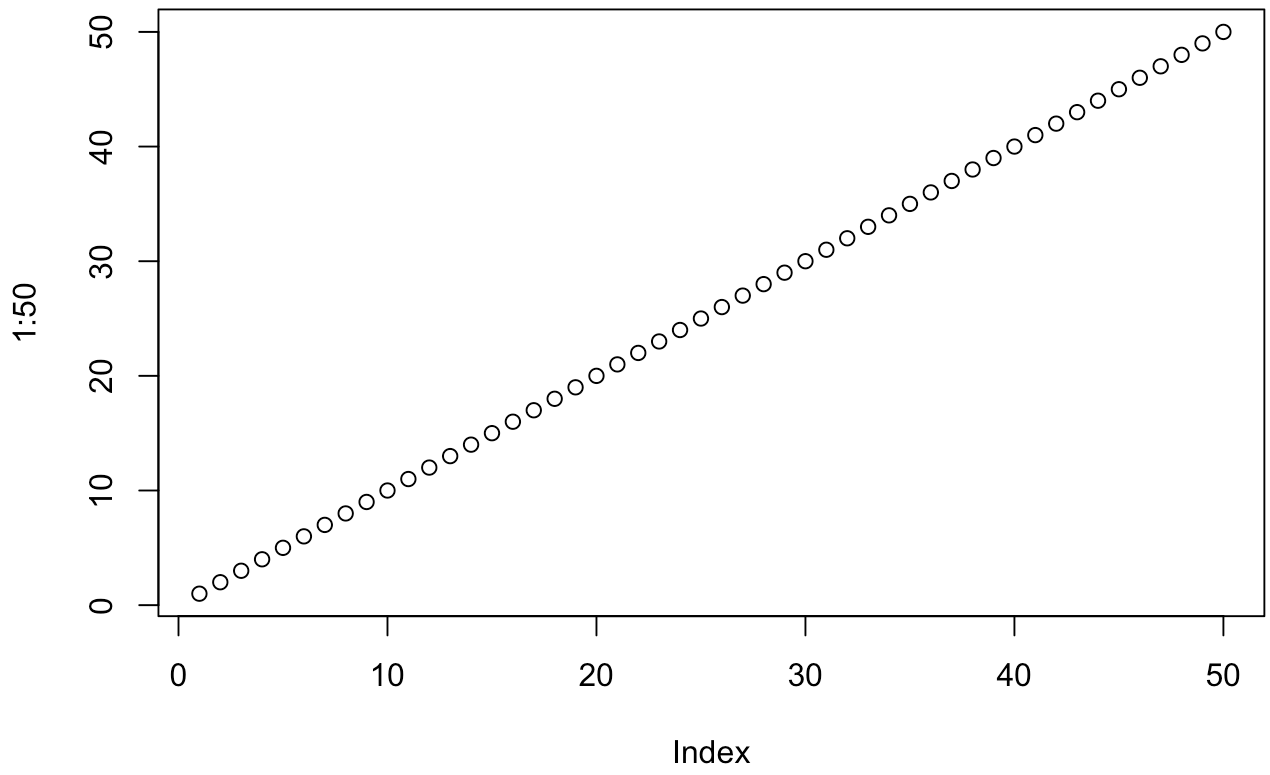
Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see https://quarto.org.

Running Code

When you click the **Render** button a *document* will be generated that includes both content and the output of embedded code. You can embed code like this:

Introduction

```
        plot(1:50)
```



Naming a chunk

```
1 + 1
```

```
[1] 2
```

Starting class

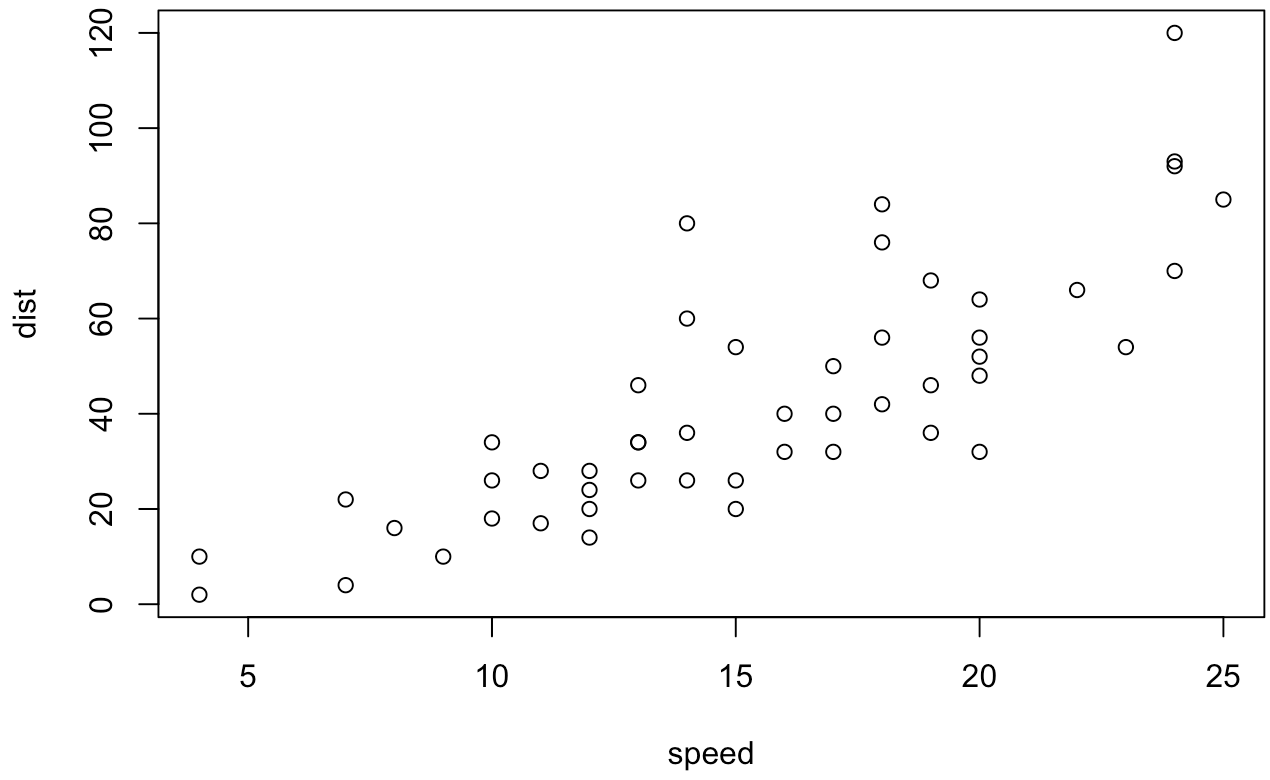
R has lot's of ways to make figures and graphs in particular. One that comes with R out of the box is called **"base" R** - the `plot()` function.

```
cars
```

```
   speed dist
1      4    2
2      4   10
3      7    4
4      7   22
5      8   16
6      9   10
7     10   18
8     10   26
9     10   34
10    11   17
11    11   28
12    12   14
13    12   20
14    12   24
15    12   28
16    13   26
17    13   34
18    13   34
19    13   46
20    14   26
21    14   36
22    14   60
23    14   80
24    15   20
25    15   26
26    15   54
27    16   32
28    16   40
29    17   32
30    17   40
31    17   50
32    18   42
33    18   56
```

```
34    18   76
35    18   84
36    19   36
37    19   46
38    19   68
39    20   32
40    20   48
41    20   52
42    20   56
43    20   64
44    22   66
45    23   54
46    24   70
47    24   92
48    24   93
49    24  120
50    25   85
```

```
plot(cars)
```



A very popular package in this area is called **ggplot2**.

```
library("ggplot2")
```

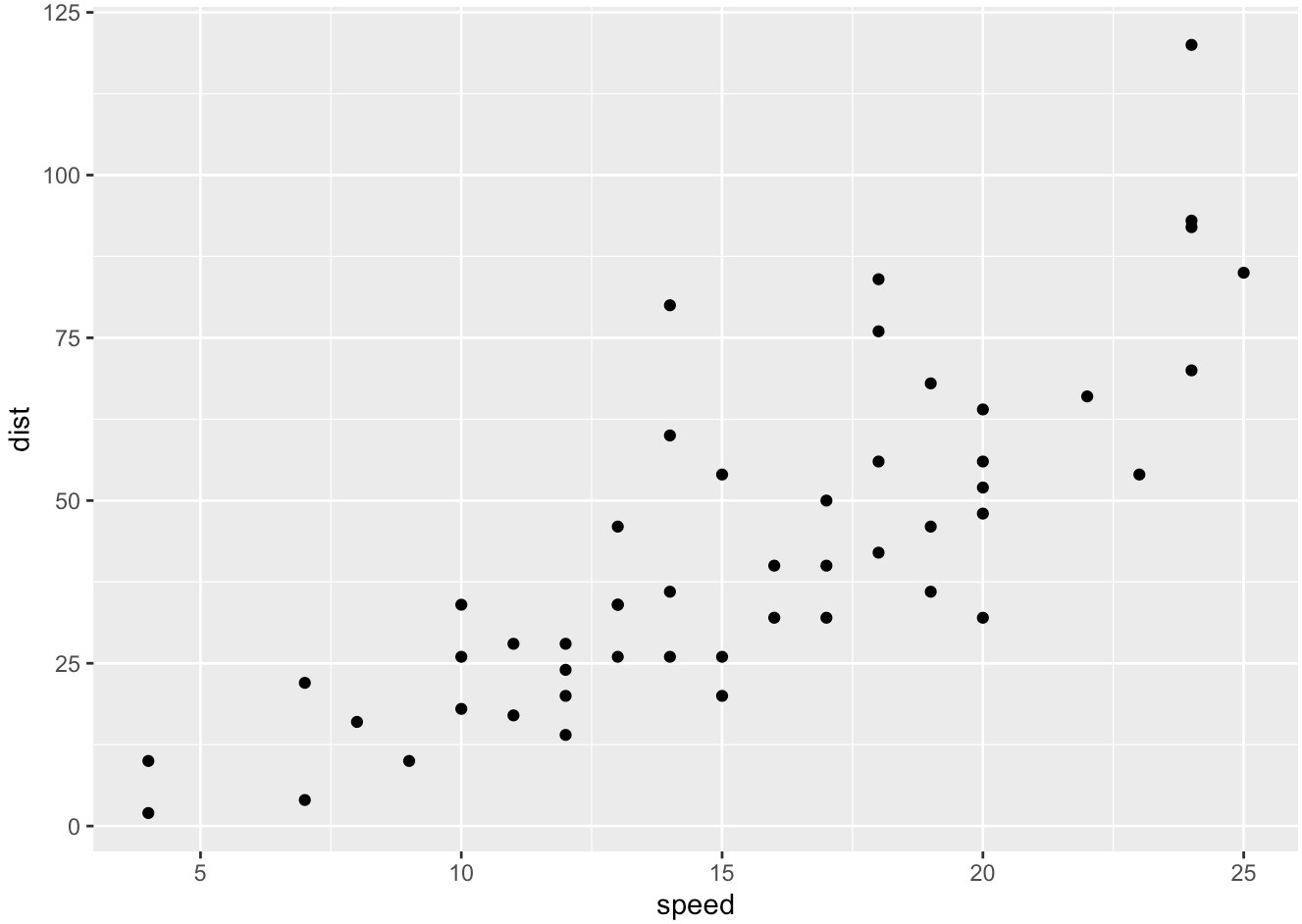
```
packageVersion("ggplot2") #3.5.1
```

```
[1] '3.5.1'
```

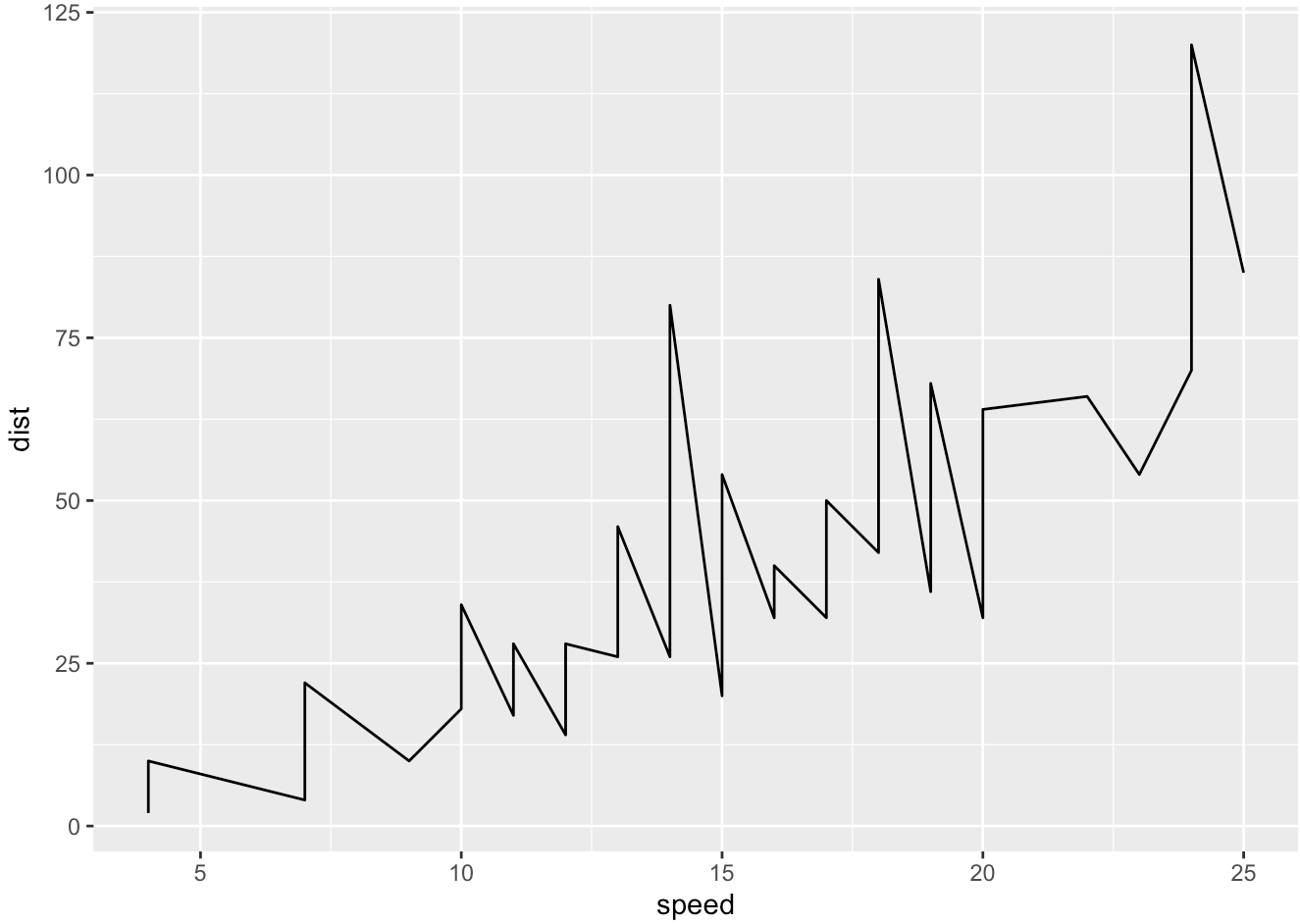
```
ggplot(cars)
```



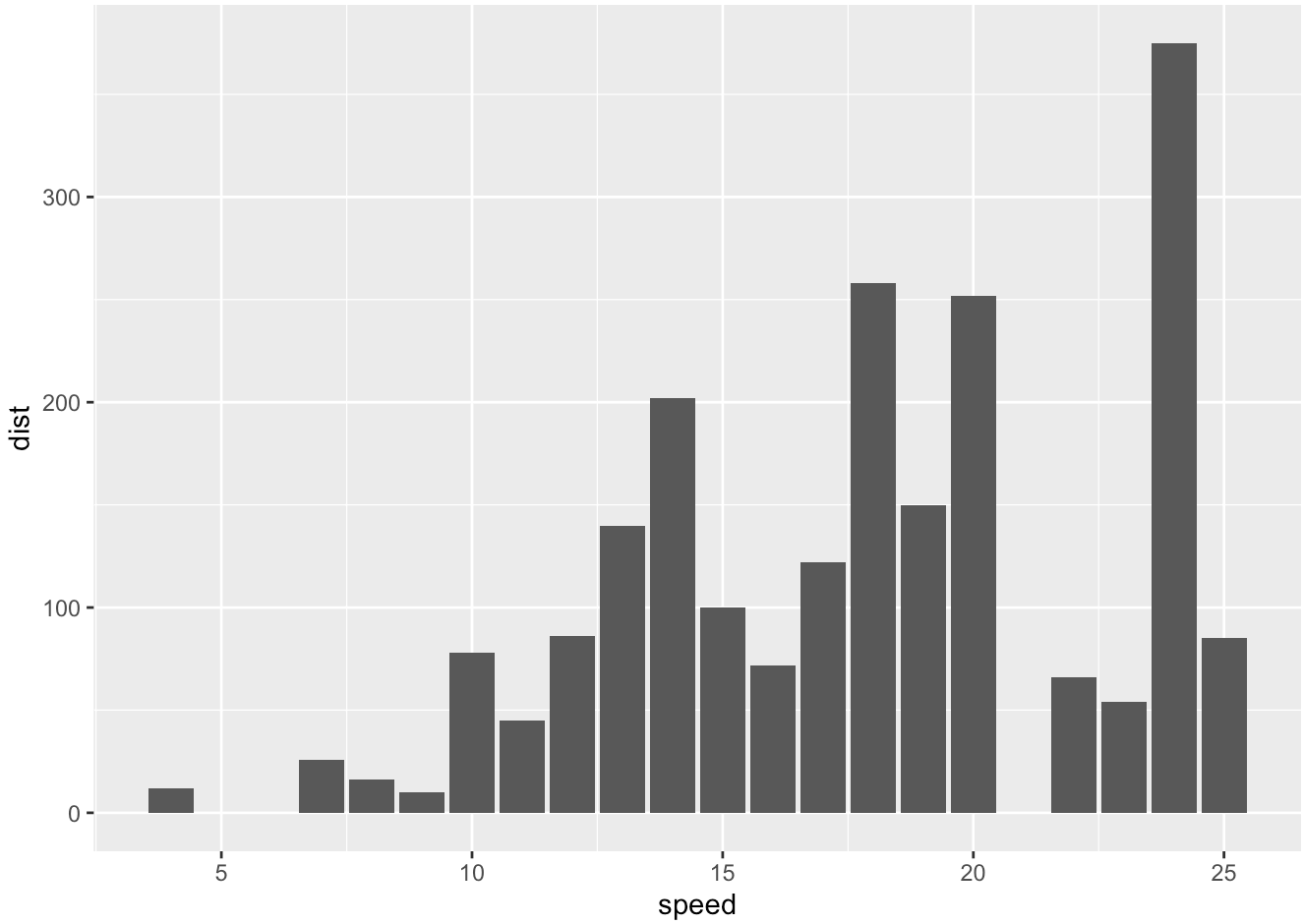
```
ggplot(cars) +
  aes(x = speed, y = dist) +
  geom_point()
```




```
ggplot(cars) +
  aes(x = speed, y = dist) +
  geom_line()
```

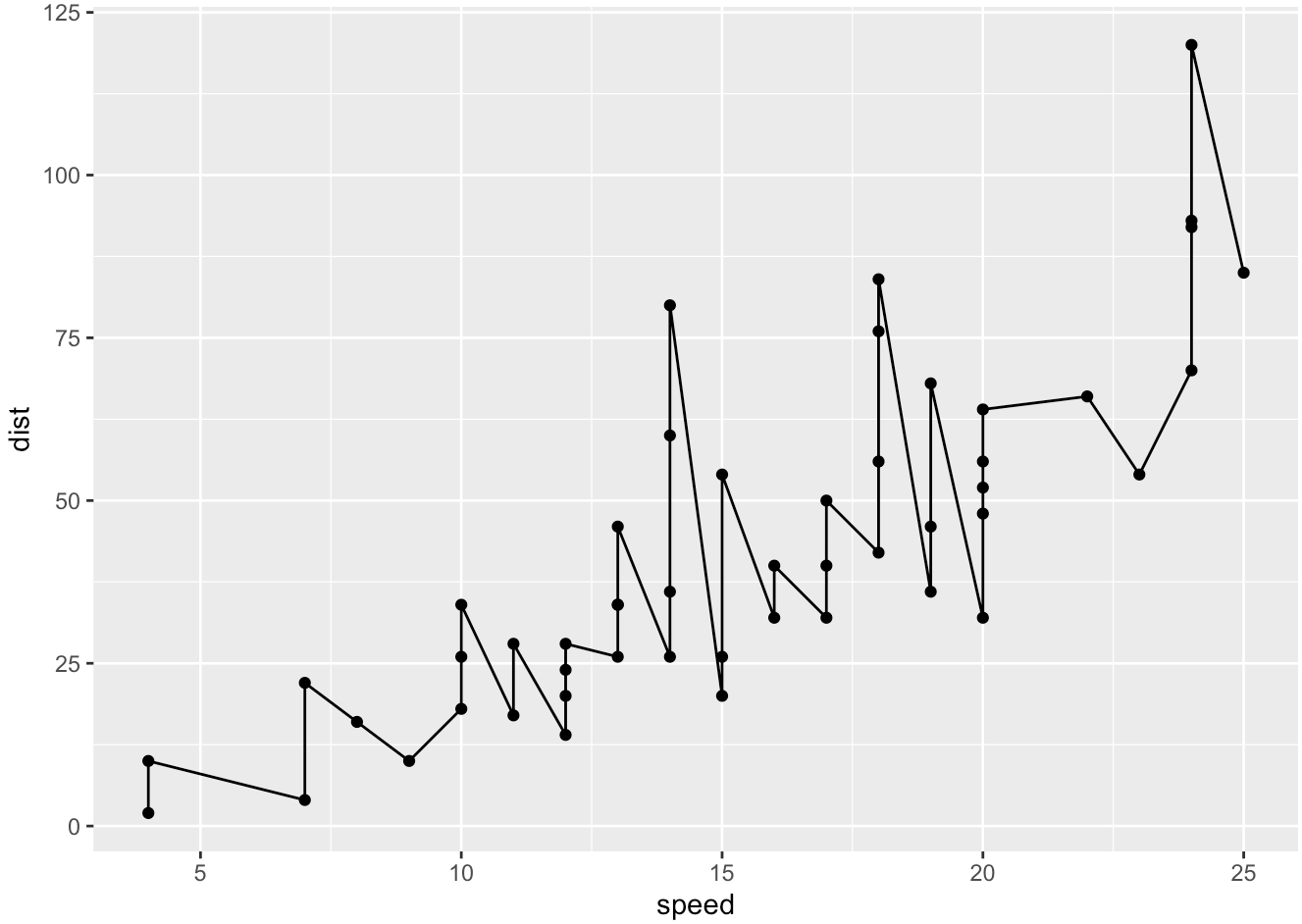



```
ggplot(cars) +
  aes(x = speed, y = dist) +
  geom_col()
```



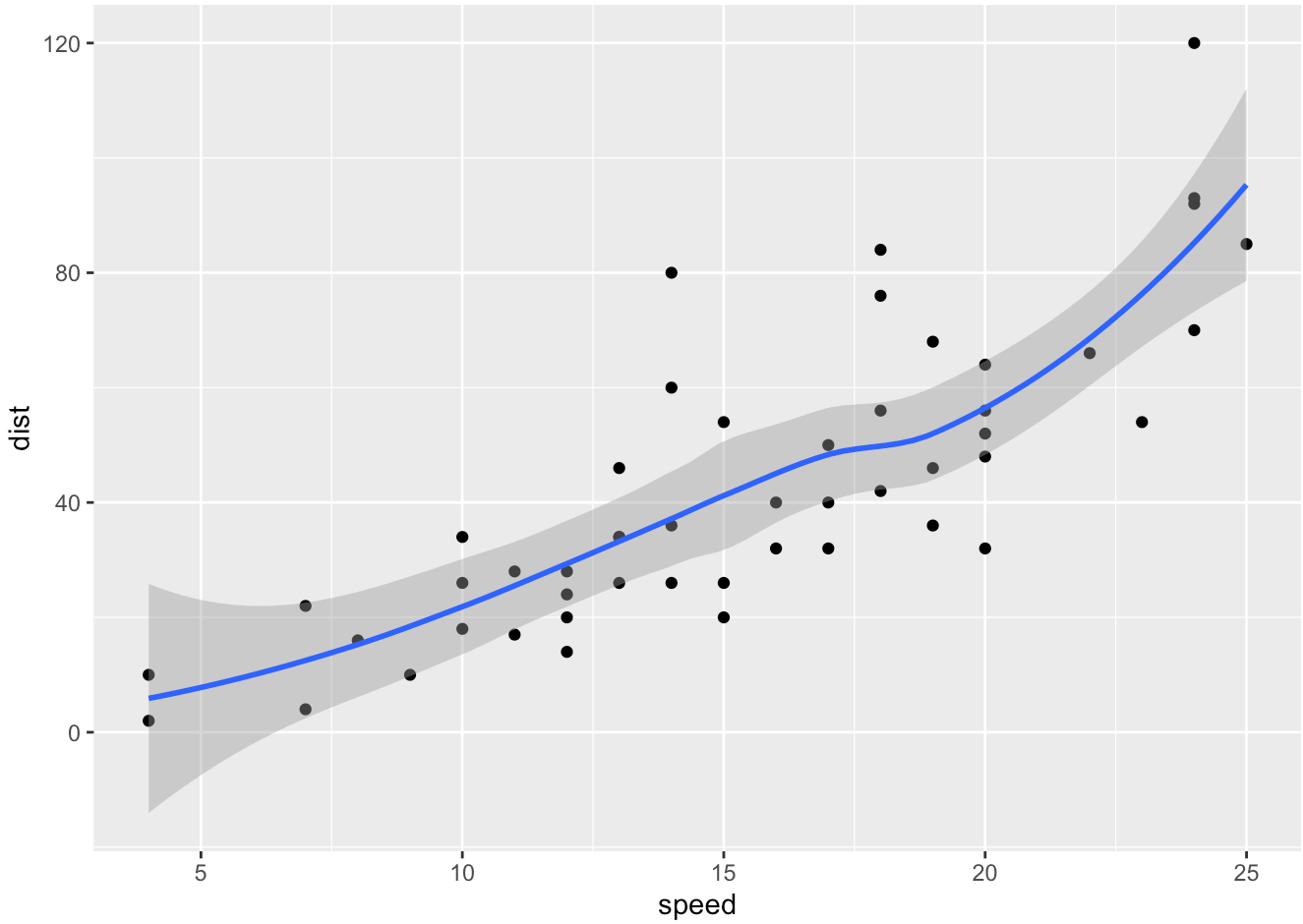
For “simple” plots like this one base R code will be much shorter than ggplot code. Let’s fit a model and show it on my plot:

```
ggplot(cars) +
  aes(x = speed, y = dist) +
  geom_point() +
  geom_line()
```




```
ggplot(cars) +
  aes(x = speed, y = dist) +
  geom_point() +
  geom_smooth()
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Step back

Every ggplot has at least 3 layers - data (data.frame with the numbers and stuff you want to plot) - aes (aesthetics, mapping of your data columns to your plot, position, size, line type, line width, color, shape) - geom (geom_point(), geom_line(), geom_col())

Little exercise

ggplot of the `mtcars` data set using `mpg` vs `disp`. Set the size of the point to the `hp`. And set color to `am`.

```
mtcars
```

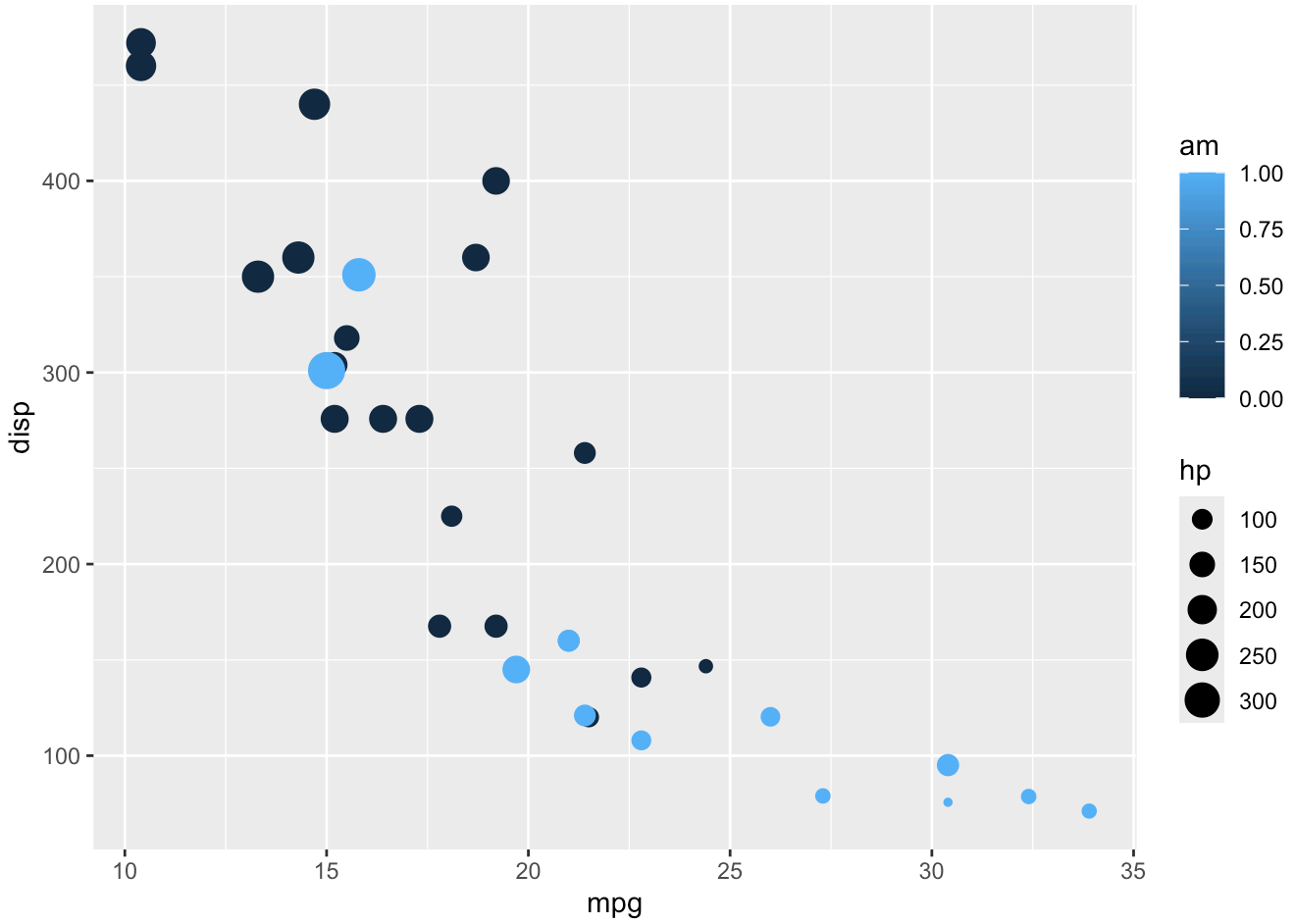
	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	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

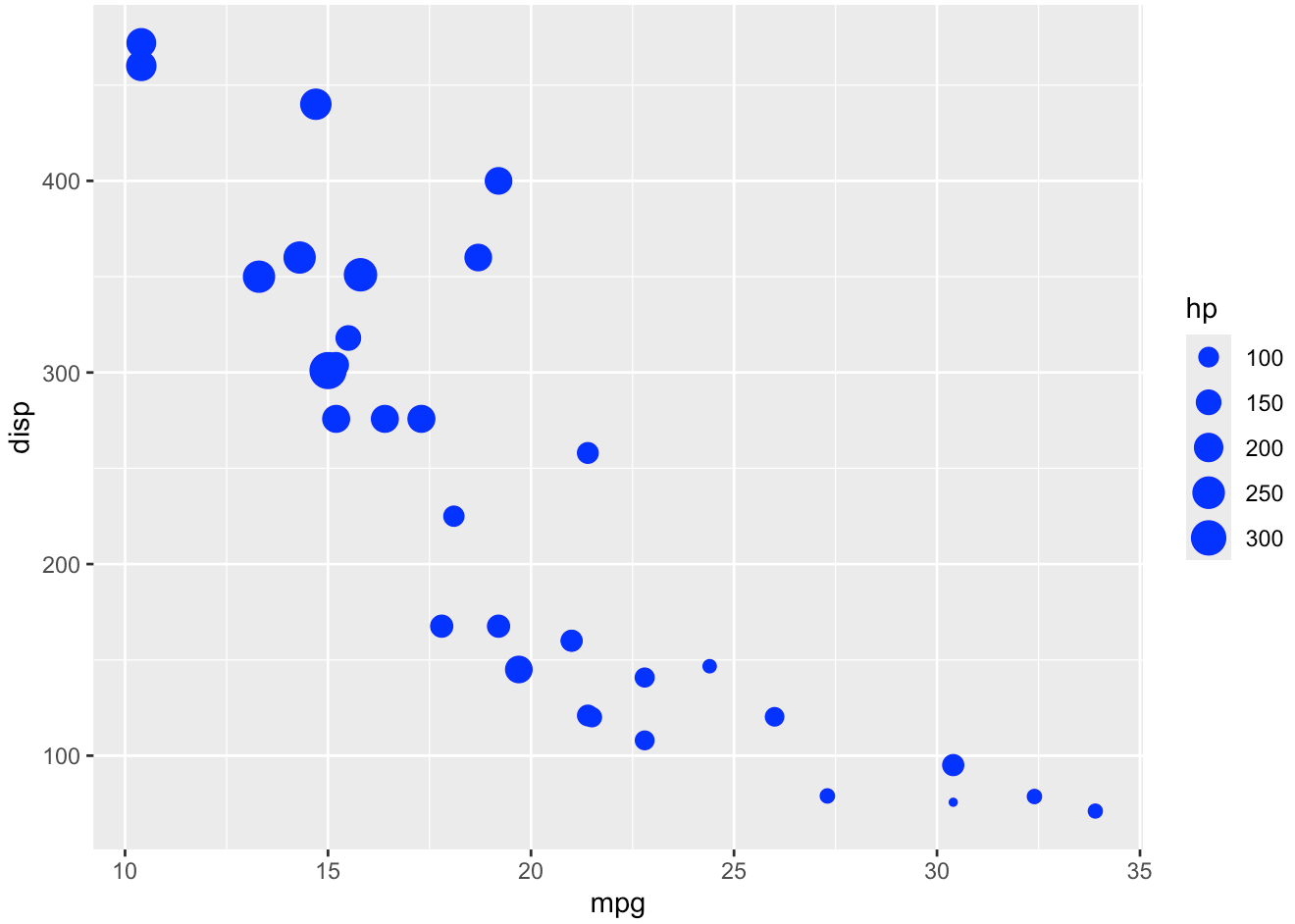
```
head(mtcars)
```

	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

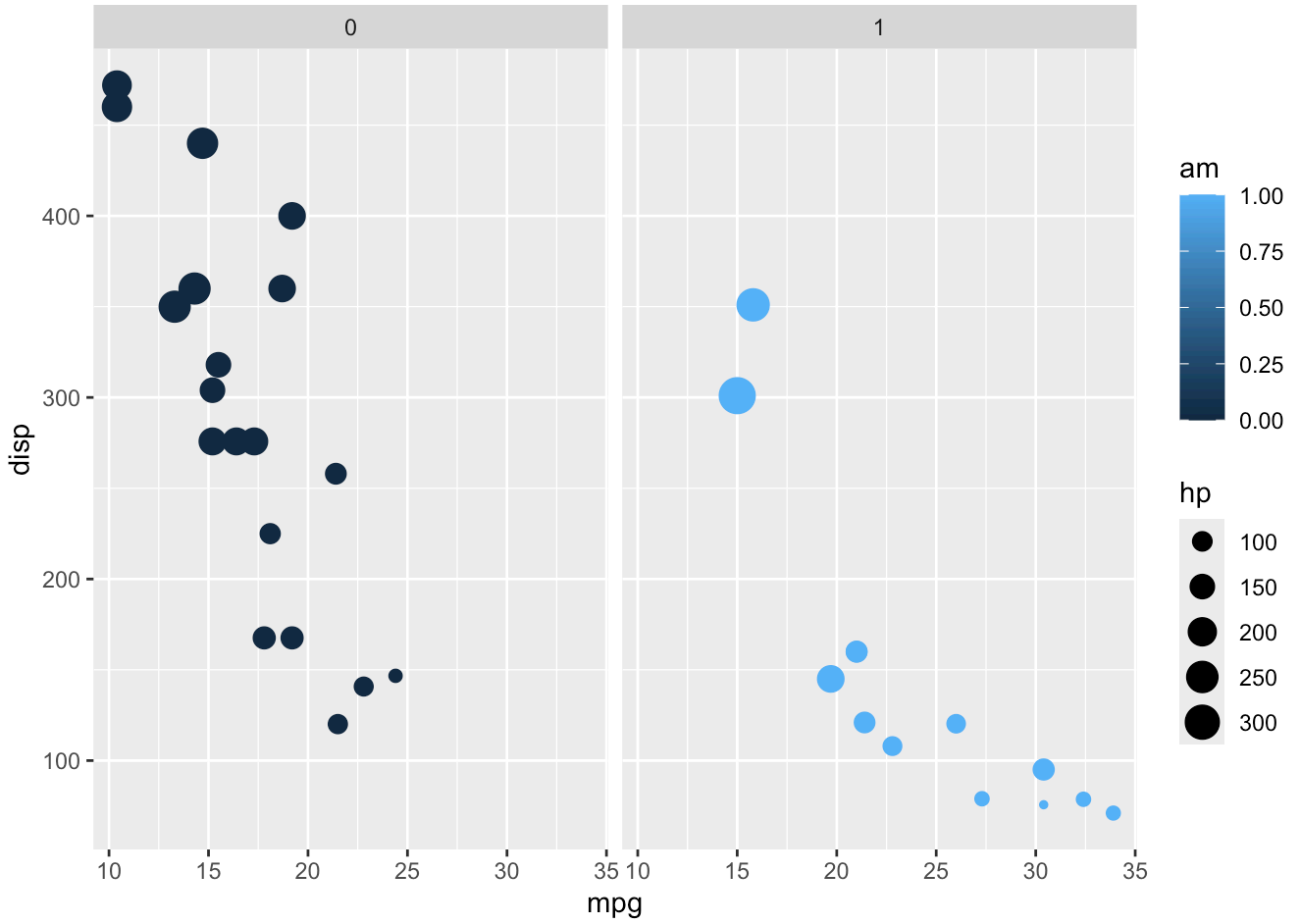
```
ggplot(mtcars) +
  aes(x = mpg, y = disp, size = hp, color = am) +
  geom_point()
```



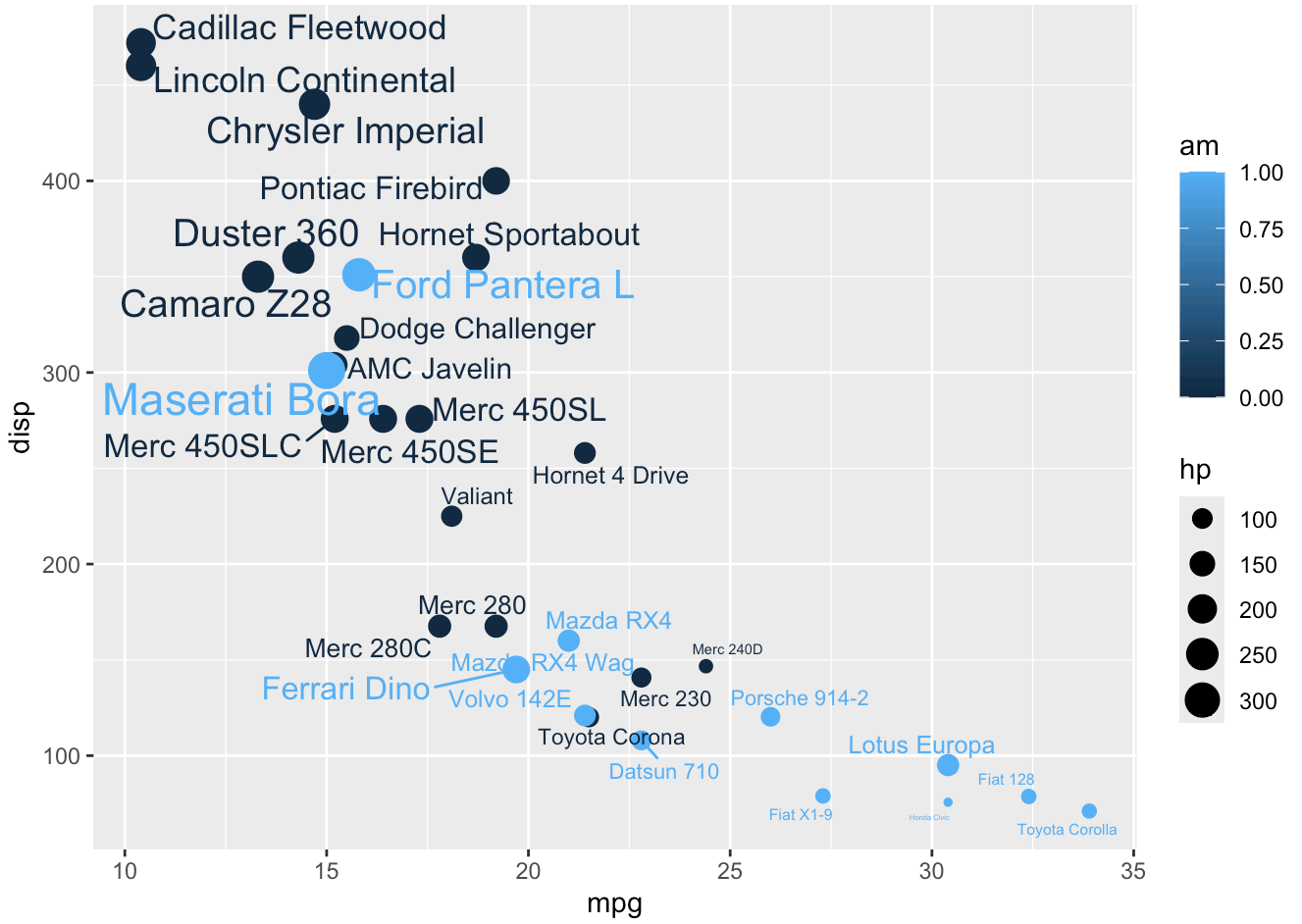

```
# Now everything blue
ggplot(mtcars) +
  aes(x = mpg, y = disp, size = hp) +
  geom_point(color = "blue")
```




```
# Facet
ggplot(mtcars) +
  aes(x = mpg, y = disp, size = hp, color = am) +
  geom_point() +
  facet_wrap(~am)
```




```
# Label
library(ggrepel)
ggplot(mtcars) +
  aes(x = mpg, y = disp, size = hp, color = am, label=rownames(mtcars)) +
  geom_point() +
  geom_text_repel()
```



From this moment, we work on our own:

Adding more plot aesthetics through aes()

```
# Load the data
url <- "https://bioboot.github.io/bimm143_S20/class-material/up_down_expression.t
genes <- read.delim(url)

# Display the first few rows
head(genes)
```

```
        Gene Condition1 Condition2      State
1      A4GNT -3.6808610 -3.4401355 unchanging
2       AAAS  4.5479580  4.3864126 unchanging
3      AASDH  3.7190695  3.4787276 unchanging
4       AATF  5.0784720  5.0151916 unchanging
5       AATK  0.4711421  0.5598642 unchanging
6 AB015752.4 -3.6808610 -3.5921390 unchanging
```

```
# Count the number of rows and check column names
nrow(genes)         # answer = 5196
```

```
[1] 5196
```

```
colnames(genes)     # answer = gene, condition1, condition2, state
```

```
[1] "Gene"       "Condition1" "Condition2" "State"
```

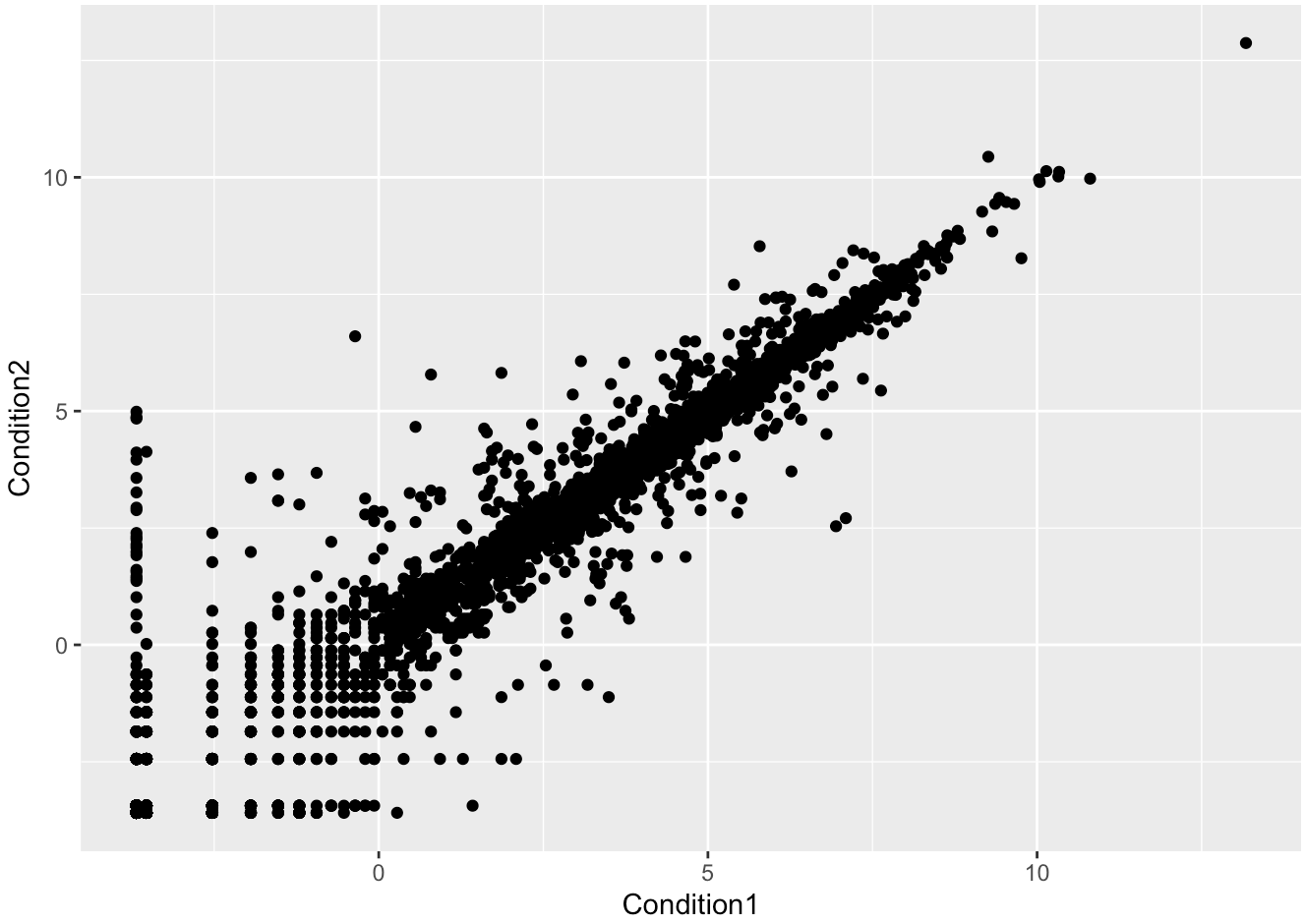
```
# Calculate how many up regulated genes there are
table(genes$State)
```

```
      down unchanging         up
        72       4997        127
```

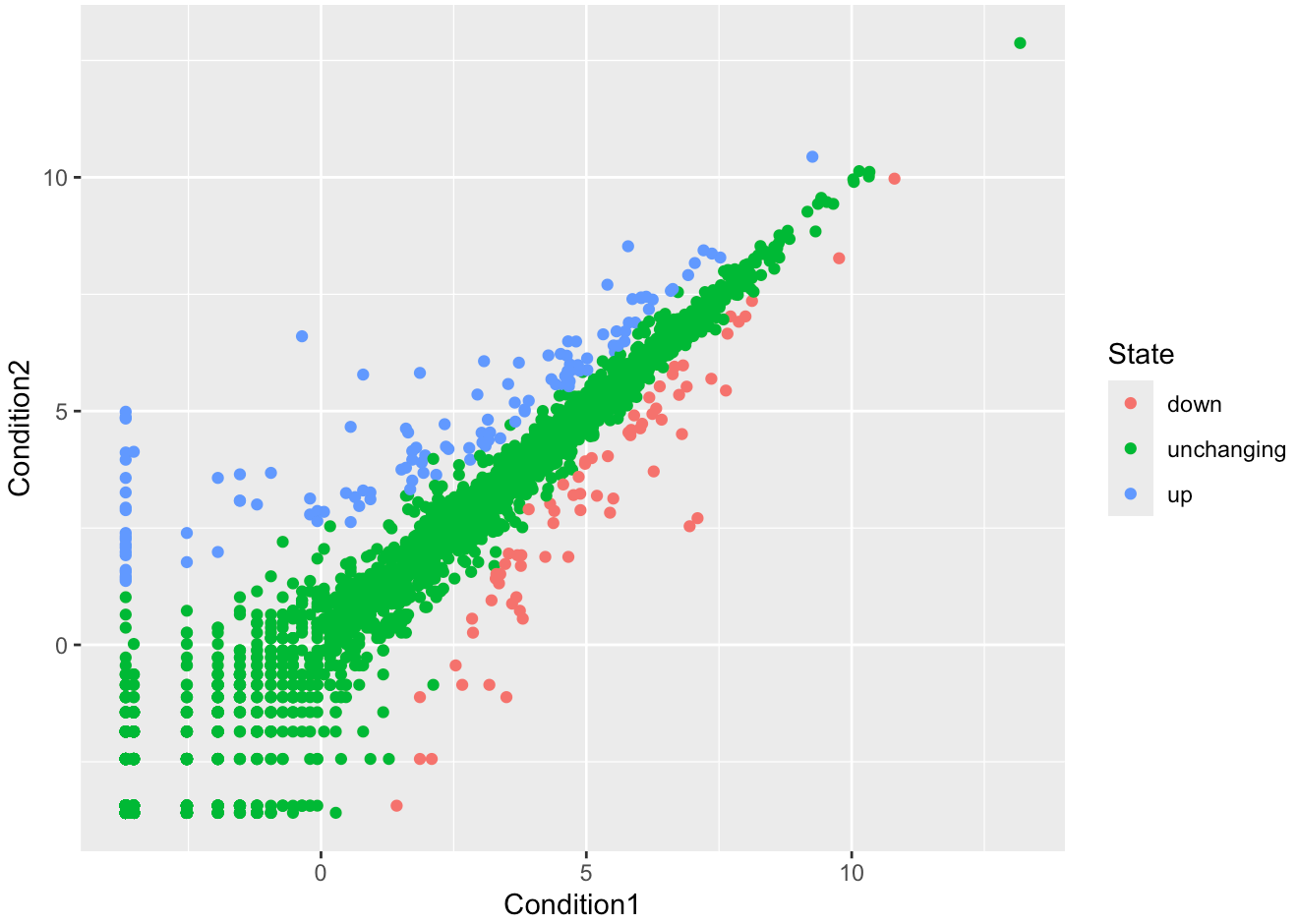
```
# What fraction of total genes is up-regulated in this dataset?
round( table(genes$State)/nrow(genes) * 100, 2 )
```

```
      down unchanging         up
      1.39      96.17       2.44
```

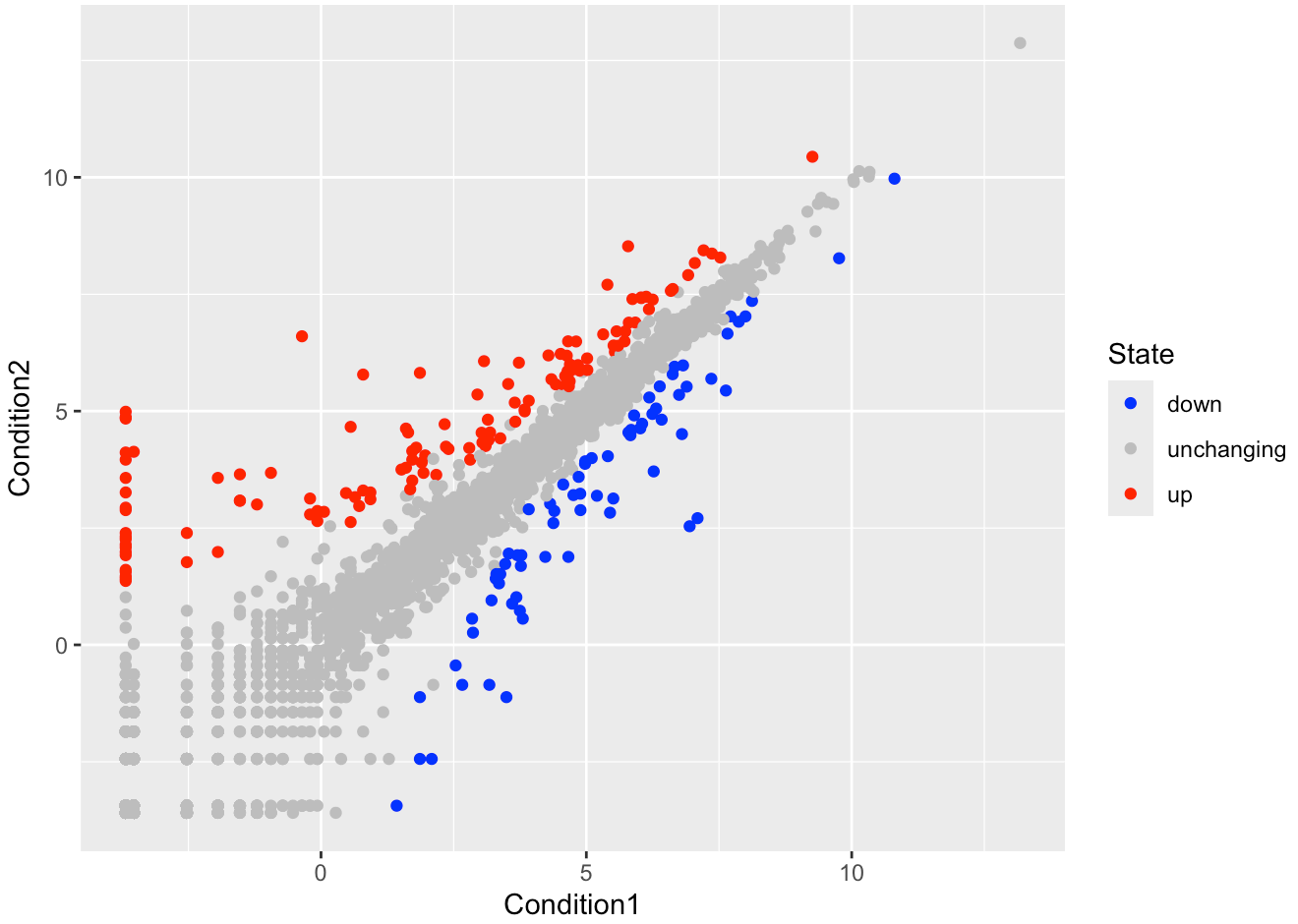
```
# Make graph
ggplot(genes) +
  aes(x=Condition1, y=Condition2) +
  geom_point()
```



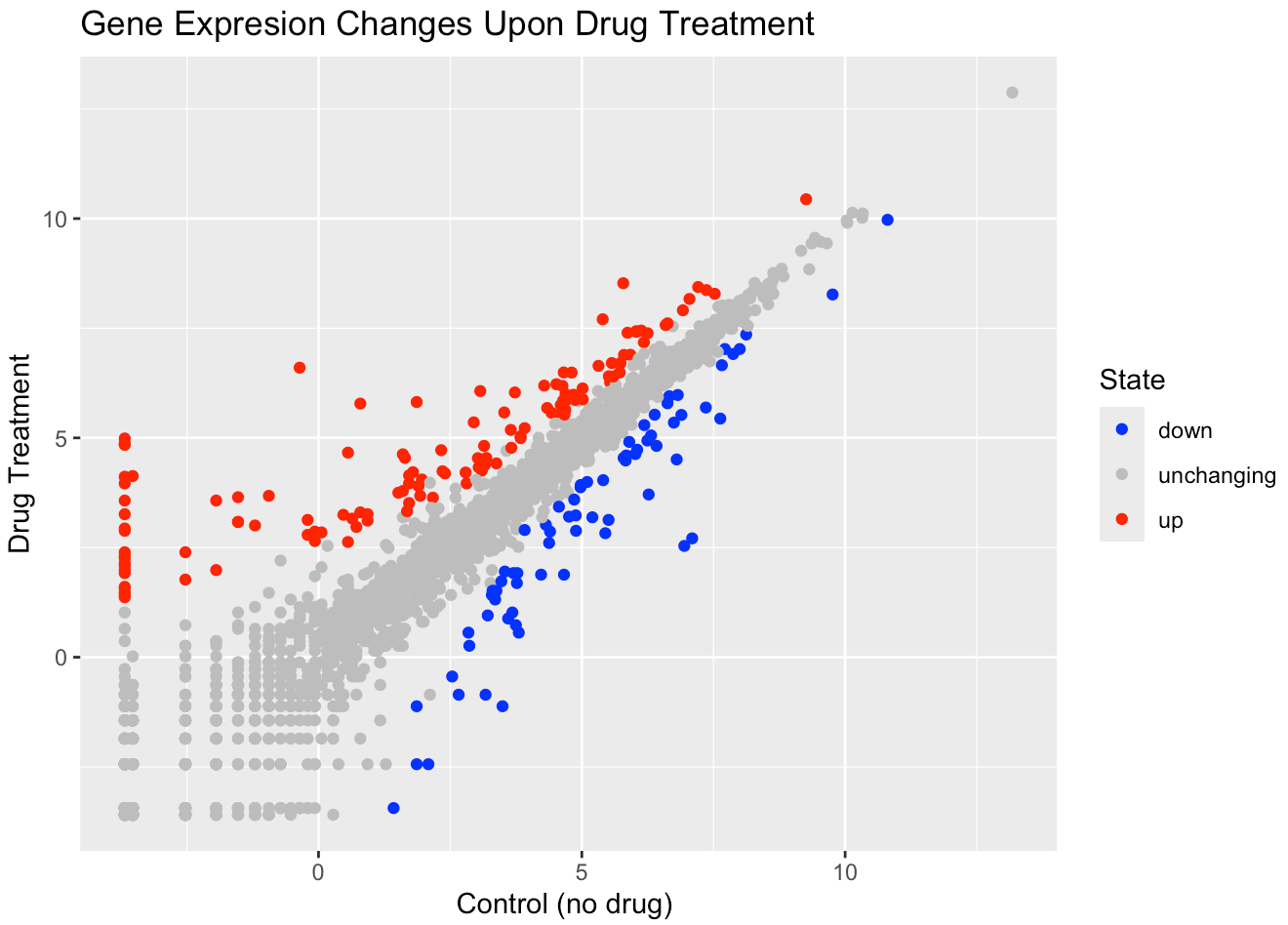

```
# Extra Info
p <- ggplot(genes) +
    aes(x=Condition1, y=Condition2, col=State) +
    geom_point()
p
```




```
# Other colors
p + scale_colour_manual( values=c("blue","gray","red") )
```




```
# Labels
p + scale_colour_manual(values=c("blue","gray","red")) +
    labs(title="Gene Expresion Changes Upon Drug Treatment",
         x="Control (no drug) ",
         y="Drug Treatment")
```



Going Further

```
# Set the CRAN mirror
options(repos = c(CRAN = "https://cran.r-project.org"))

# Install
install.packages("gapminder")
```

```
The downloaded binary packages are in
    /var/folders/wc/y60y10bj5jz0zzxkrq739z580000gn/T//RtmpRfONWV/downloaded_packages
```

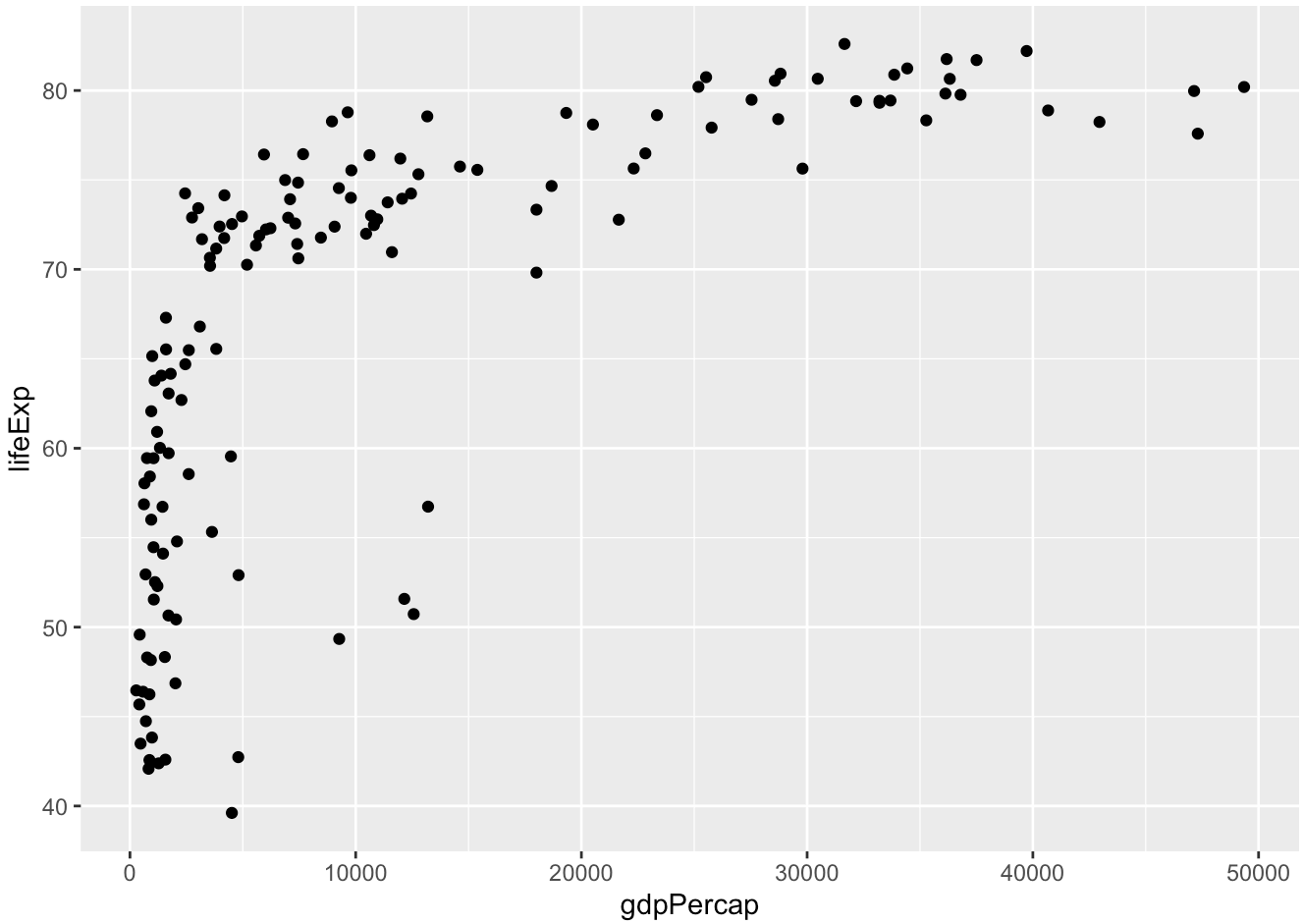
```
library(gapminder)

# Extra, will talk about this next week
install.packages("dplyr")
```

```
The downloaded binary packages are in
    /var/folders/wc/y60y10bj5jz0zzxkrq739z580000gn/T//RtmpRfONWV/downloaded_packages
```

```
library(dplyr)
```

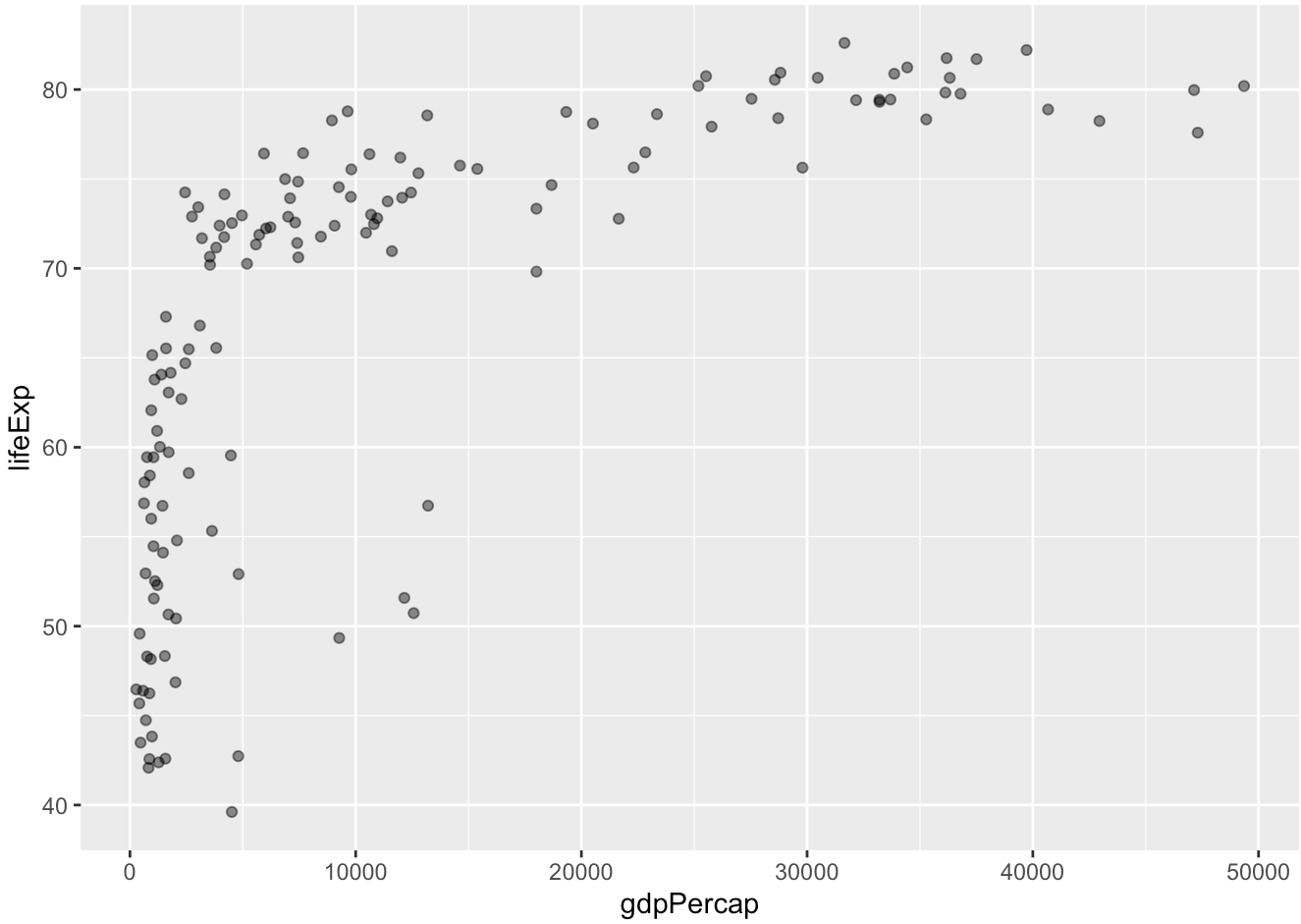
```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
gapminder_2007 <- gapminder %>% filter(year==2007)

# Plot
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point()
```

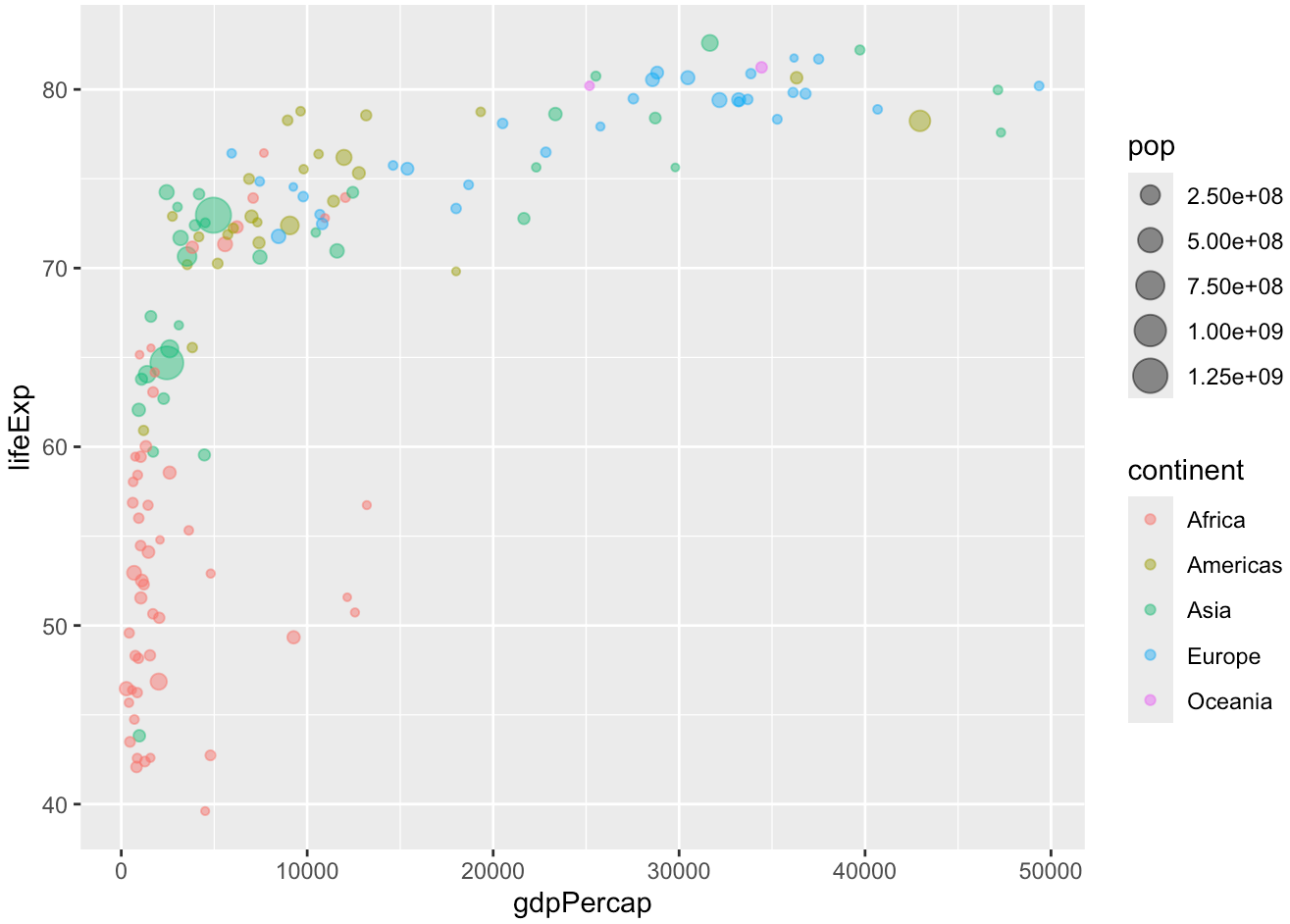



```
# Plot to have less points on top of eachother
ggplot(gapminder_2007) +
```

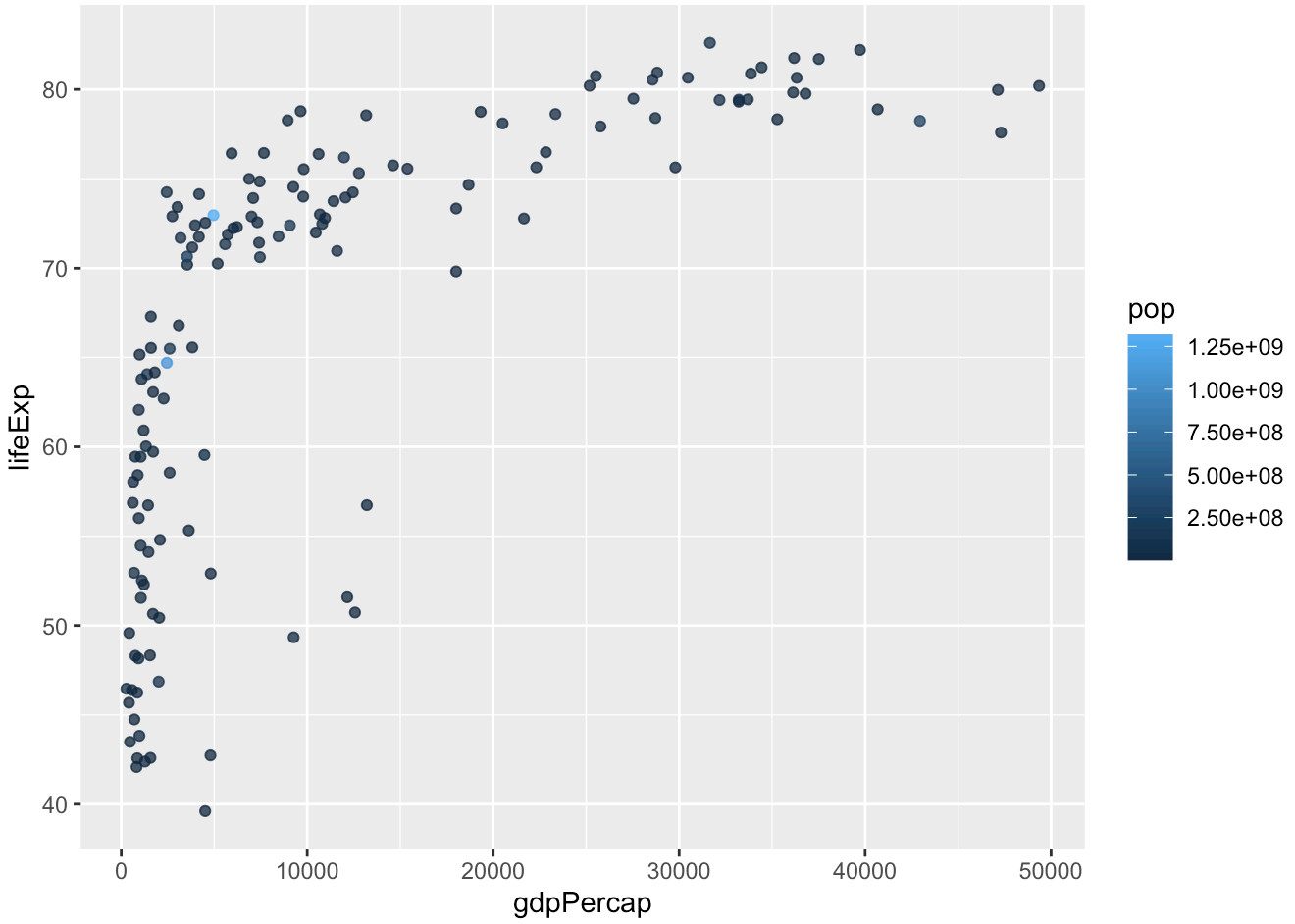
```
  aes(x=gdpPercap, y=lifeExp) +
  geom_point(alpha=0.5)
```



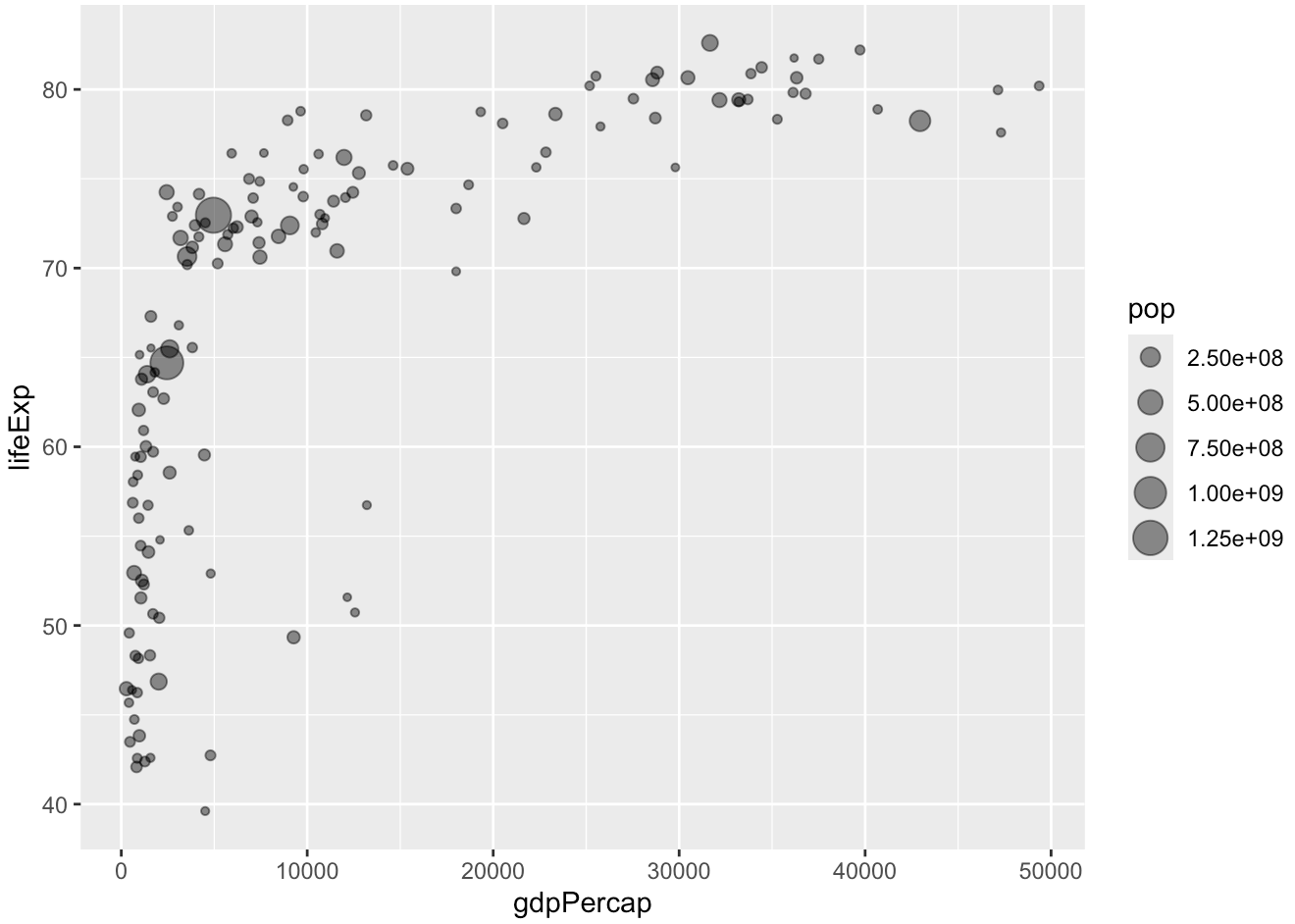

```
# Adding more variables to aes()
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.5)
```



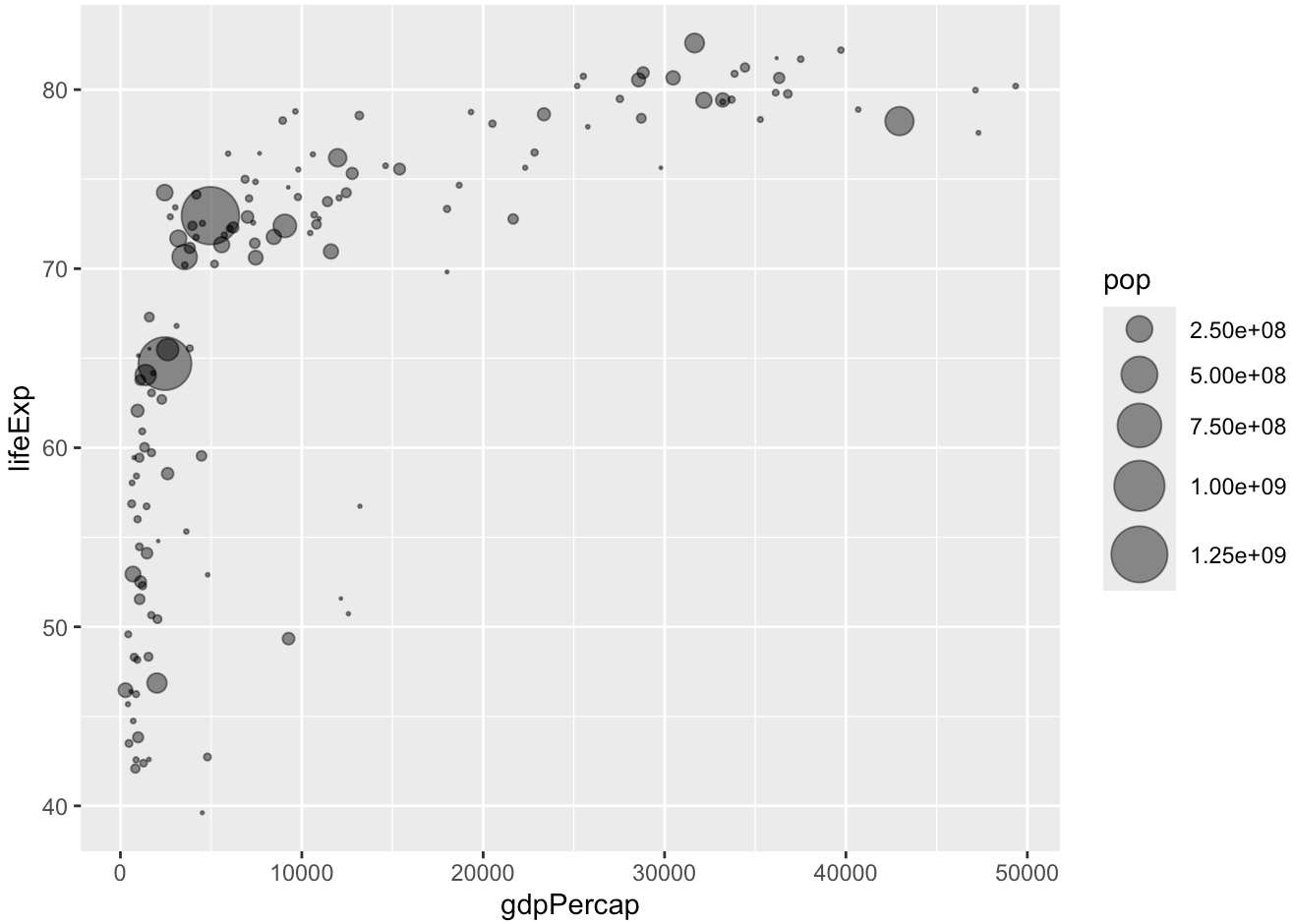

```
# This is how it looks like if we color the points by the numeric variable popula
ggplot(gapminder_2007) +
  aes(x = gdpPercap, y = lifeExp, color = pop) +
  geom_point(alpha=0.8)
```




```
# Adjusting point size
ggplot(gapminder_2007) +
  aes(x = gdpPercap, y = lifeExp, size = pop) +
  geom_point(alpha=0.5)
```

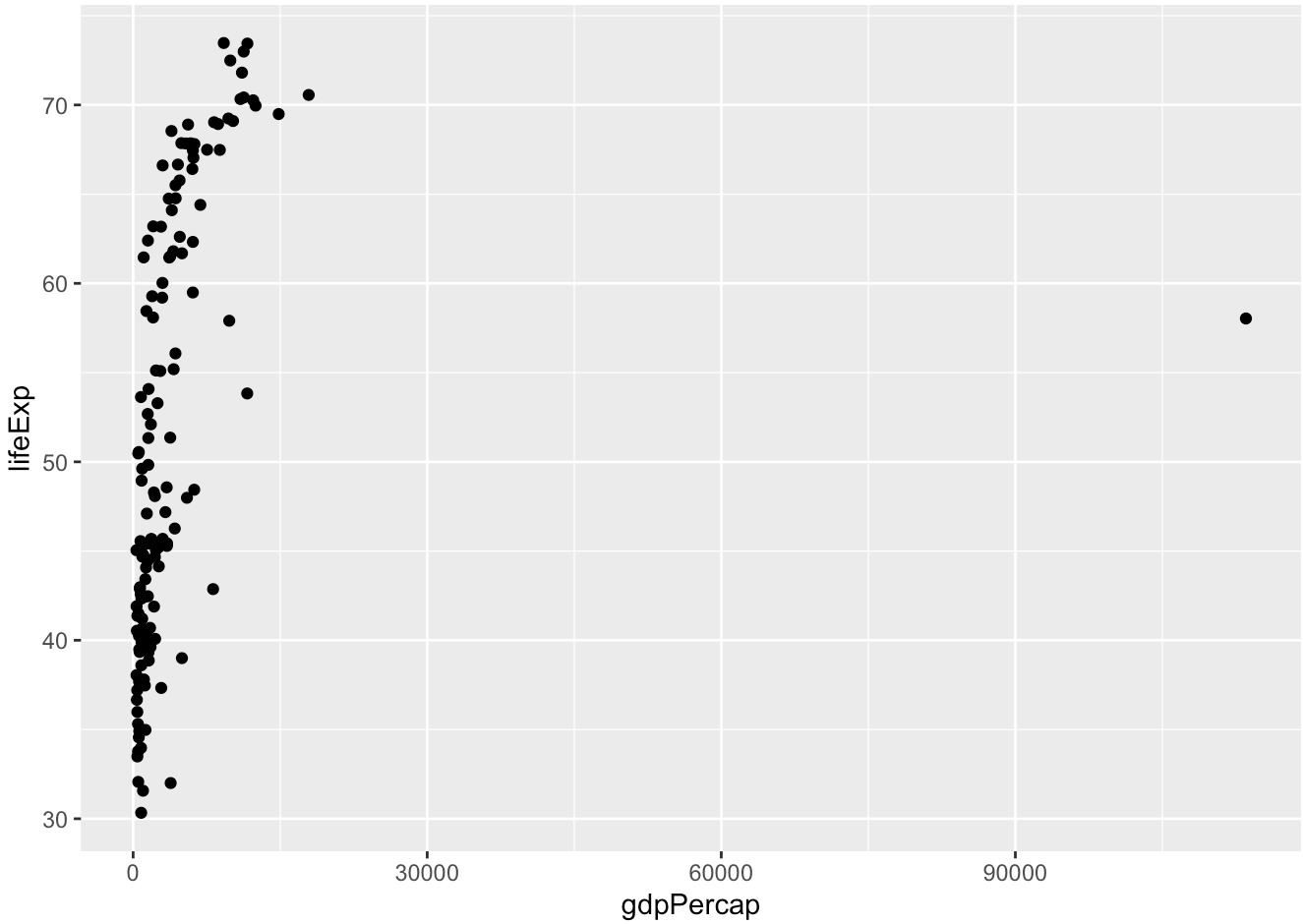



```
# Add scaling information, to reflect the actual population differences by the pc
ggplot(gapminder_2007) +
  geom_point(aes(x = gdpPercap, y = lifeExp,
                 size = pop), alpha=0.5) +
  scale_size_area(max_size = 10)
```

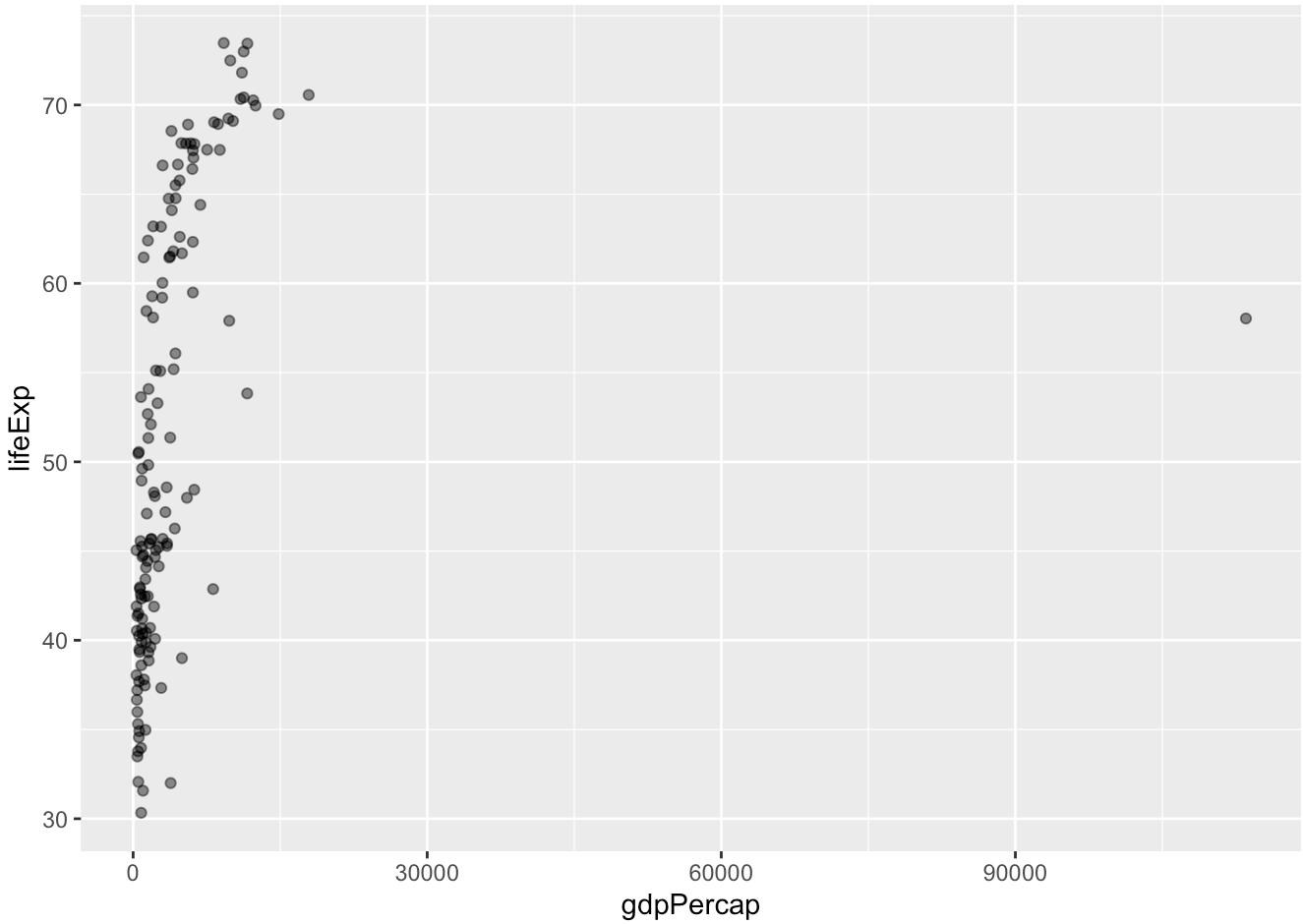


Now make the plot for year 1957

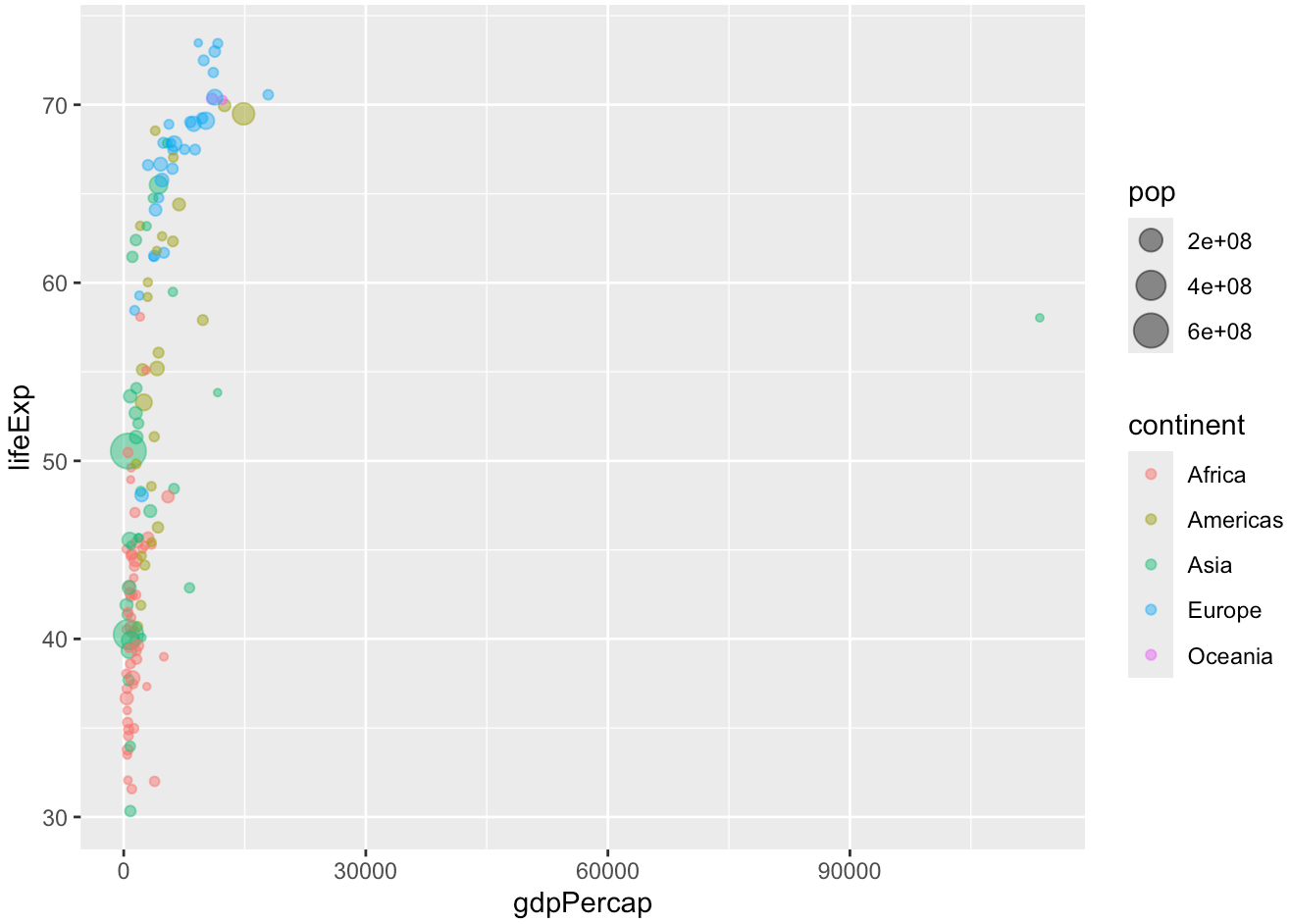
```
# Install
library(gapminder)

# Extra, will talk about this next week
library(dplyr)
gapminder_1957 <- gapminder %>% filter(year==1957)

# Plot
ggplot(gapminder_1957) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point()
```



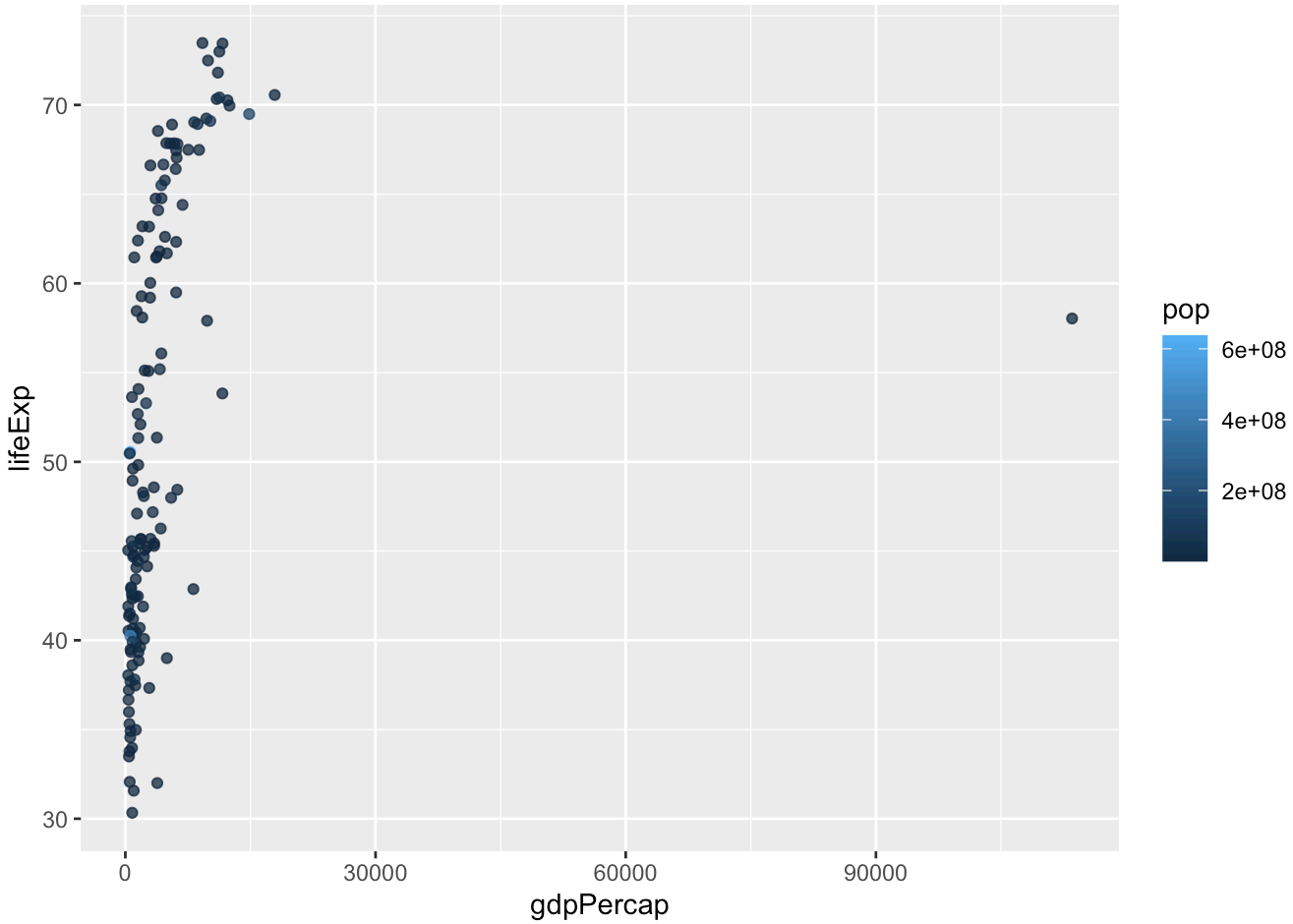

```
# Plot to have less points on top of eachother
ggplot(gapminder_1957) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point(alpha=0.5)
```



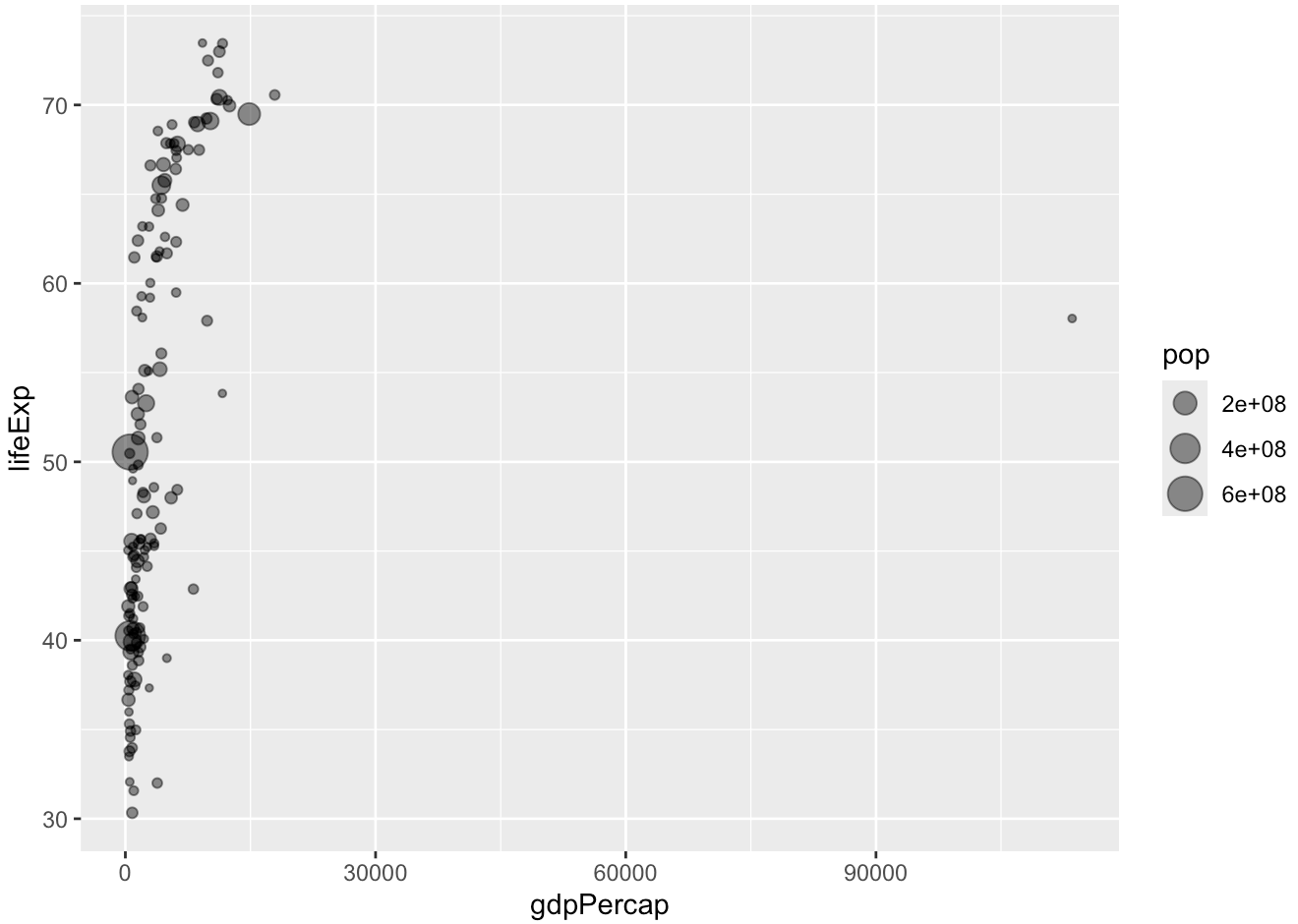

```
# Adding more variables to aes()
ggplot(gapminder_1957) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.5)
```



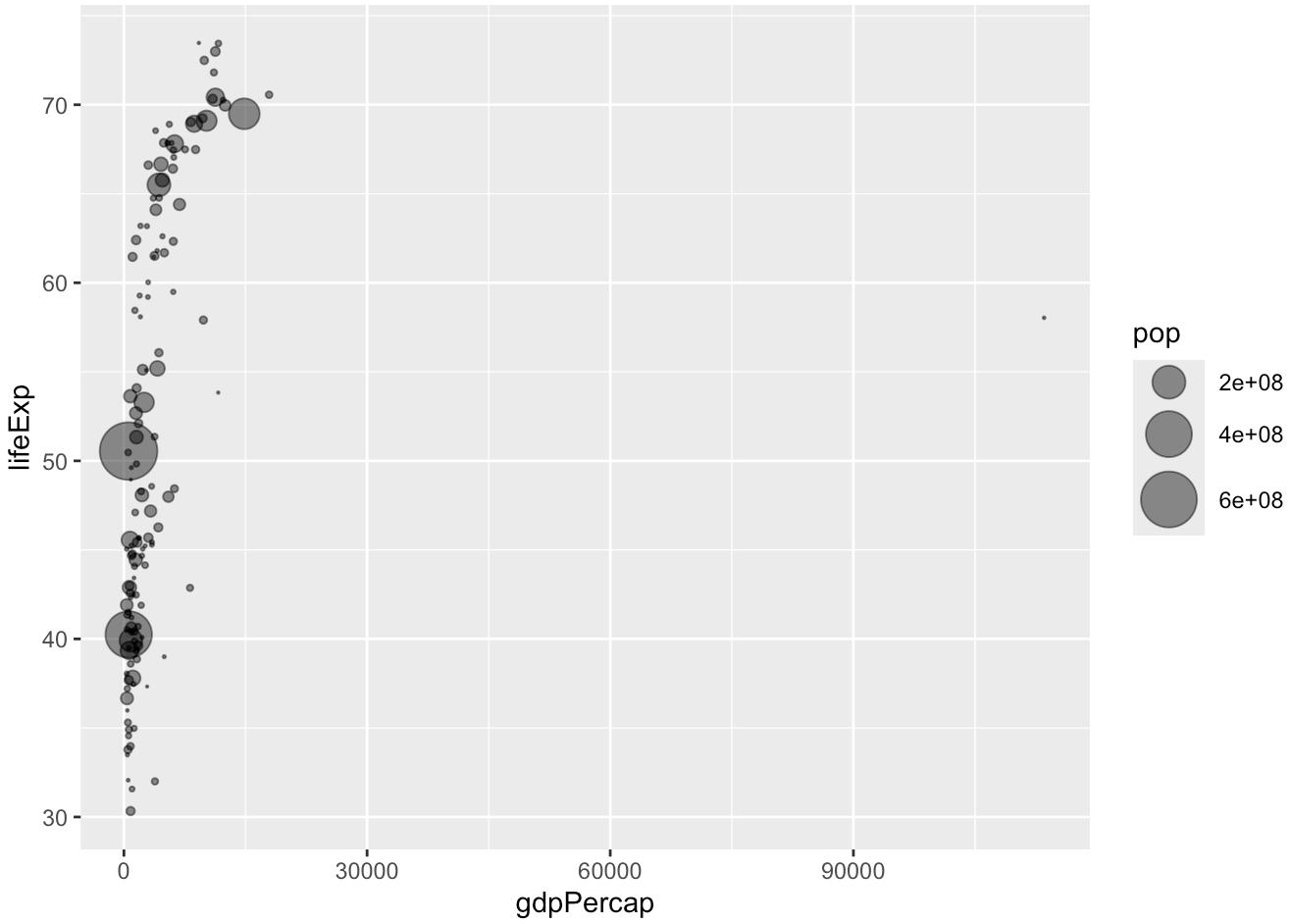

```
# This is how it looks like if we color the points by the numeric variable popula
ggplot(gapminder_1957) +
  aes(x = gdpPercap, y = lifeExp, color = pop) +
  geom_point(alpha=0.8)
```




```
# Adjusting point size
ggplot(gapminder_1957) +
  aes(x = gdpPercap, y = lifeExp, size = pop) +
  geom_point(alpha=0.5)
```

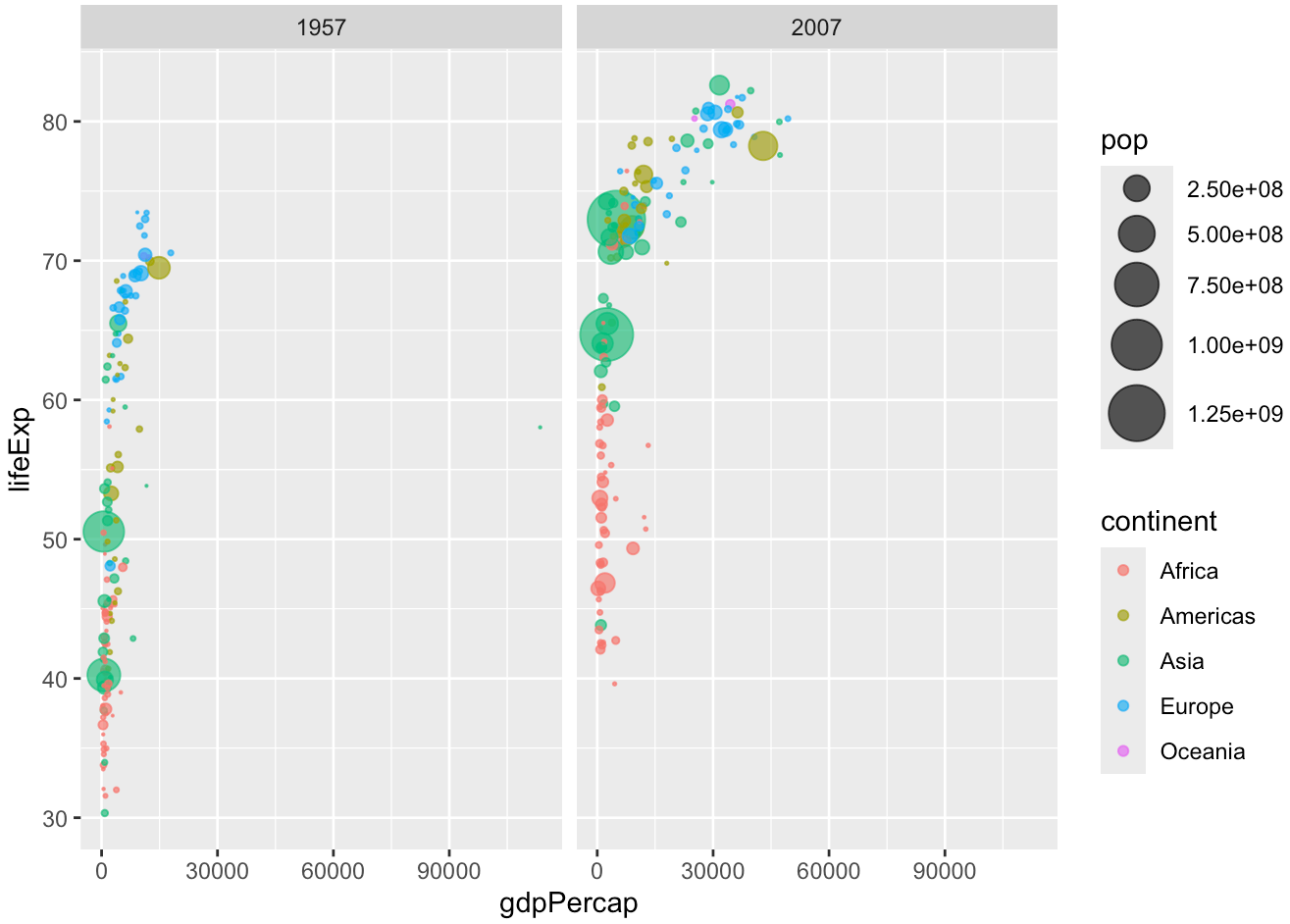



```
# Add scaling information, to reflect the actual population differences by the pc
ggplot(gapminder_1957) +
  geom_point(aes(x = gdpPercap, y = lifeExp,
                 size = pop), alpha=0.5) +
  scale_size_area(max_size = 10)
```



Now make a summarizing plot

```
gapminder_1957 <- gapminder %>% filter(year==1957 | year==2007)

ggplot(gapminder_1957) +
  geom_point(aes(x = gdpPercap, y = lifeExp, color=continent,
                 size = pop), alpha=0.7) +
  scale_size_area(max_size = 10) +
  facet_wrap(~year)
```

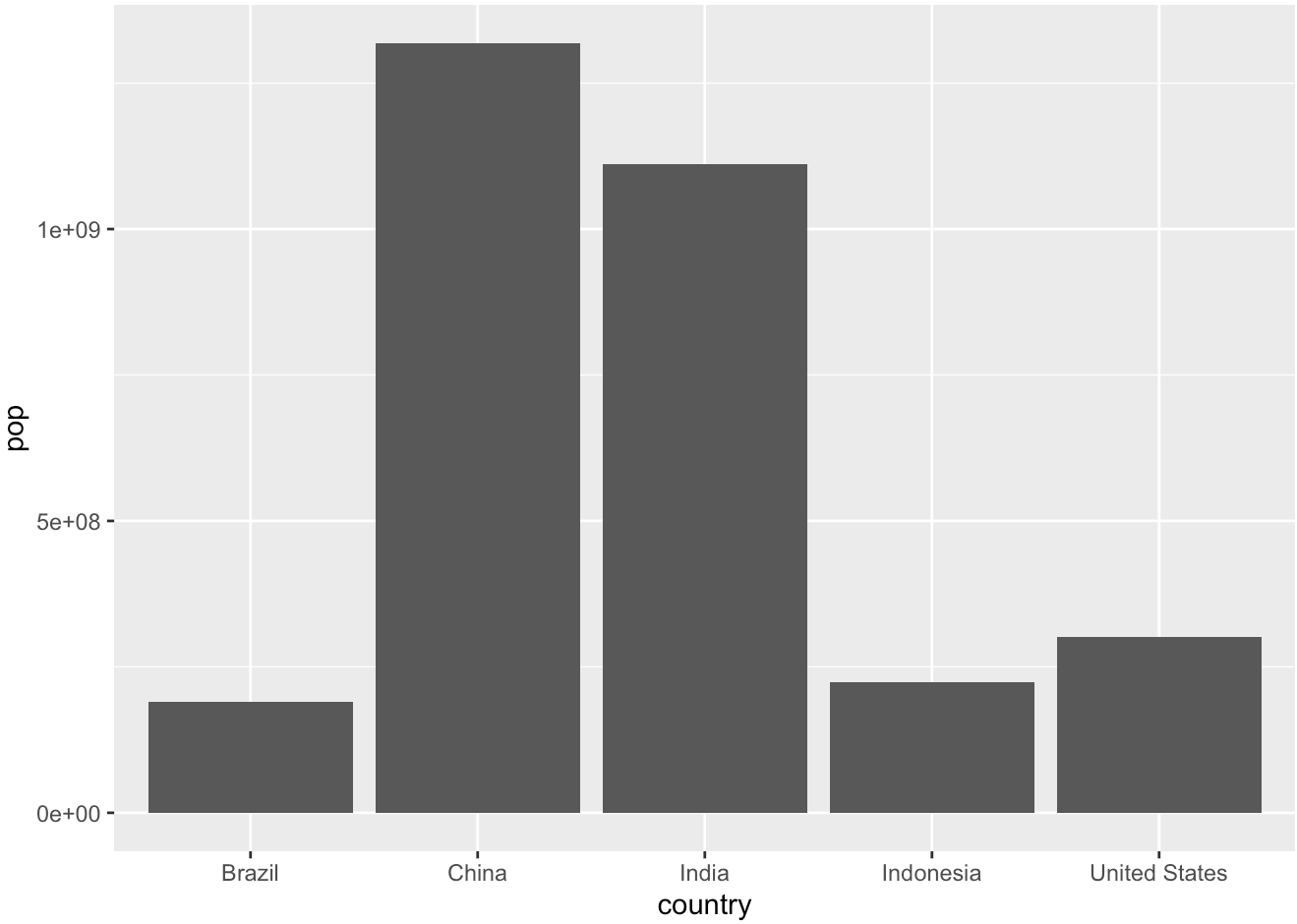


Optional: Bar Charts

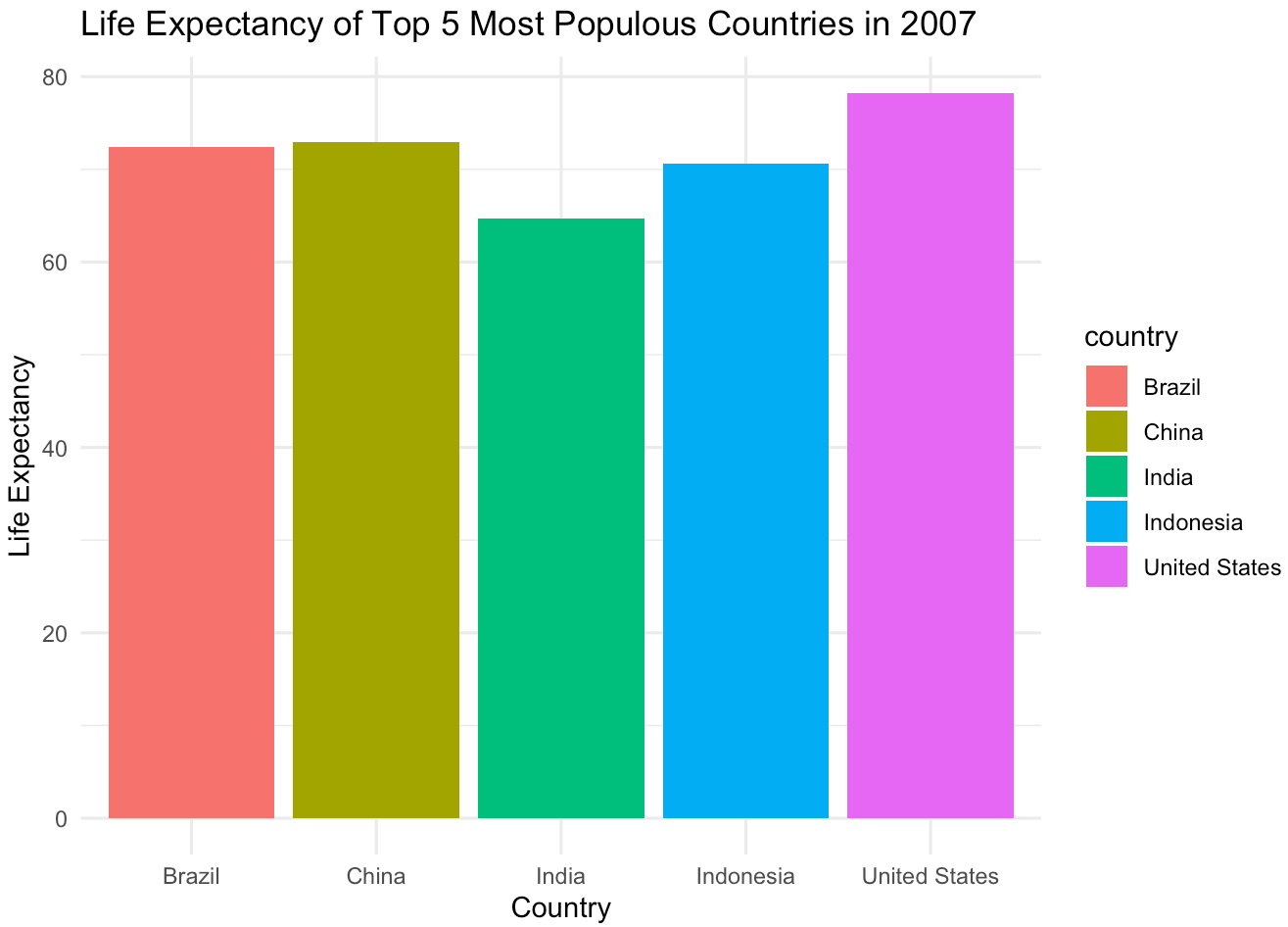
```
gapminder_top5 <- gapminder %>%
  filter(year==2007) %>%
  arrange(desc(pop)) %>%
  top_n(5, pop)

gapminder_top5
```

```
# A tibble: 5 × 6
  country       continent  year lifeExp        pop gdpPercap
  <fct>         <fct>     <int>   <dbl>      <int>     <dbl>
1 China         Asia       2007    73.0 1318683096     4959.
2 India         Asia       2007    64.7 1110396331     2452.
3 United States Americas   2007    78.2  301139947    42952.
4 Indonesia     Asia       2007    70.6  223547000     3541.
5 Brazil        Americas   2007    72.4  190010647     9066.
```

```
# Creating a simple bar chart
ggplot(gapminder_top5) +
  geom_col(aes(x = country, y = pop))
```

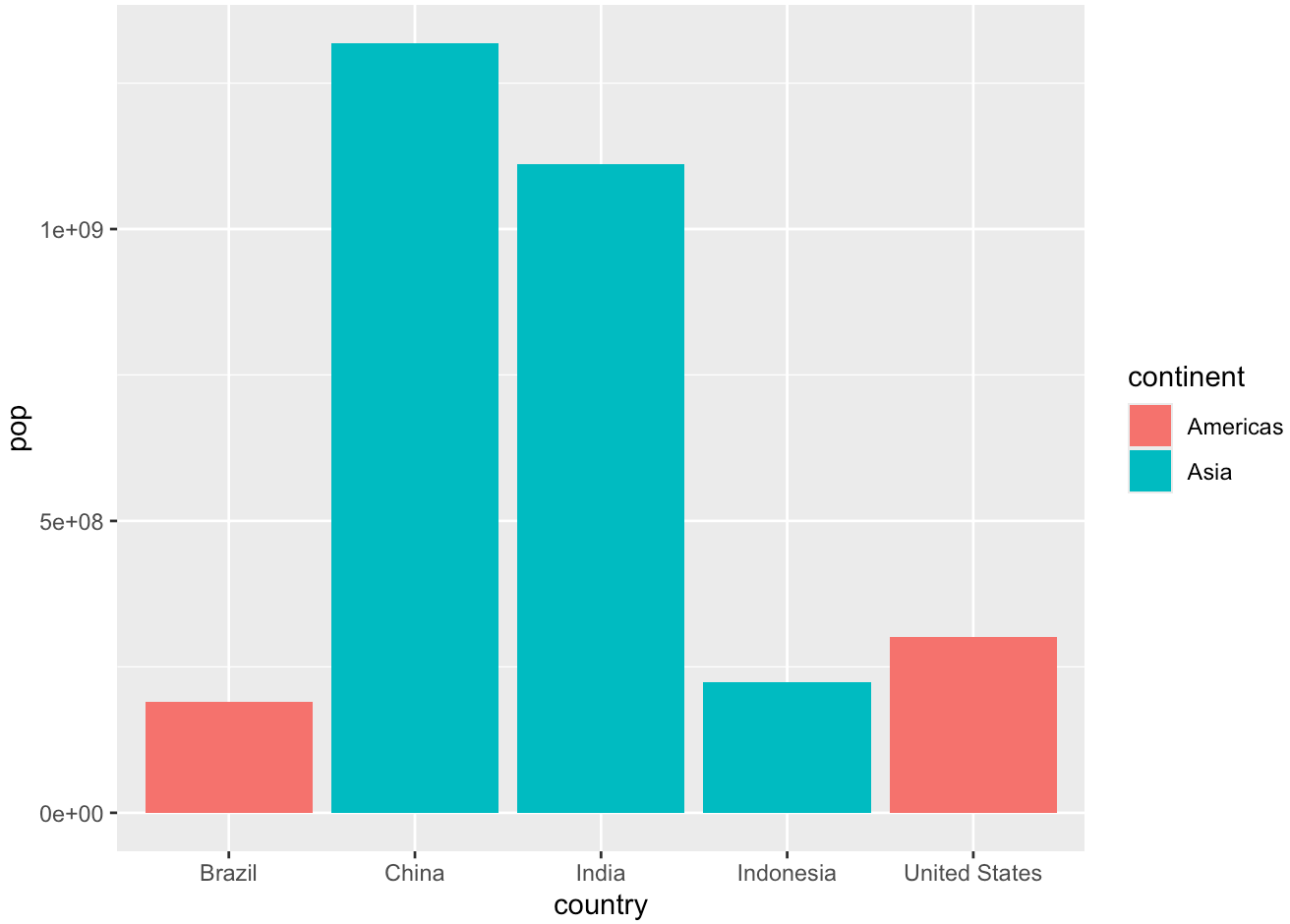



```
# Create a bar chart showing the life expectancy of the five biggest countries by
ggplot(gapminder_top5) +
  geom_col(aes(x = country, y = lifeExp, fill = country)) +
  labs(title = "Life Expectancy of Top 5 Most Populous Countries in 2007",
       x = "Country",
       y = "Life Expectancy") +
  theme_minimal()
```

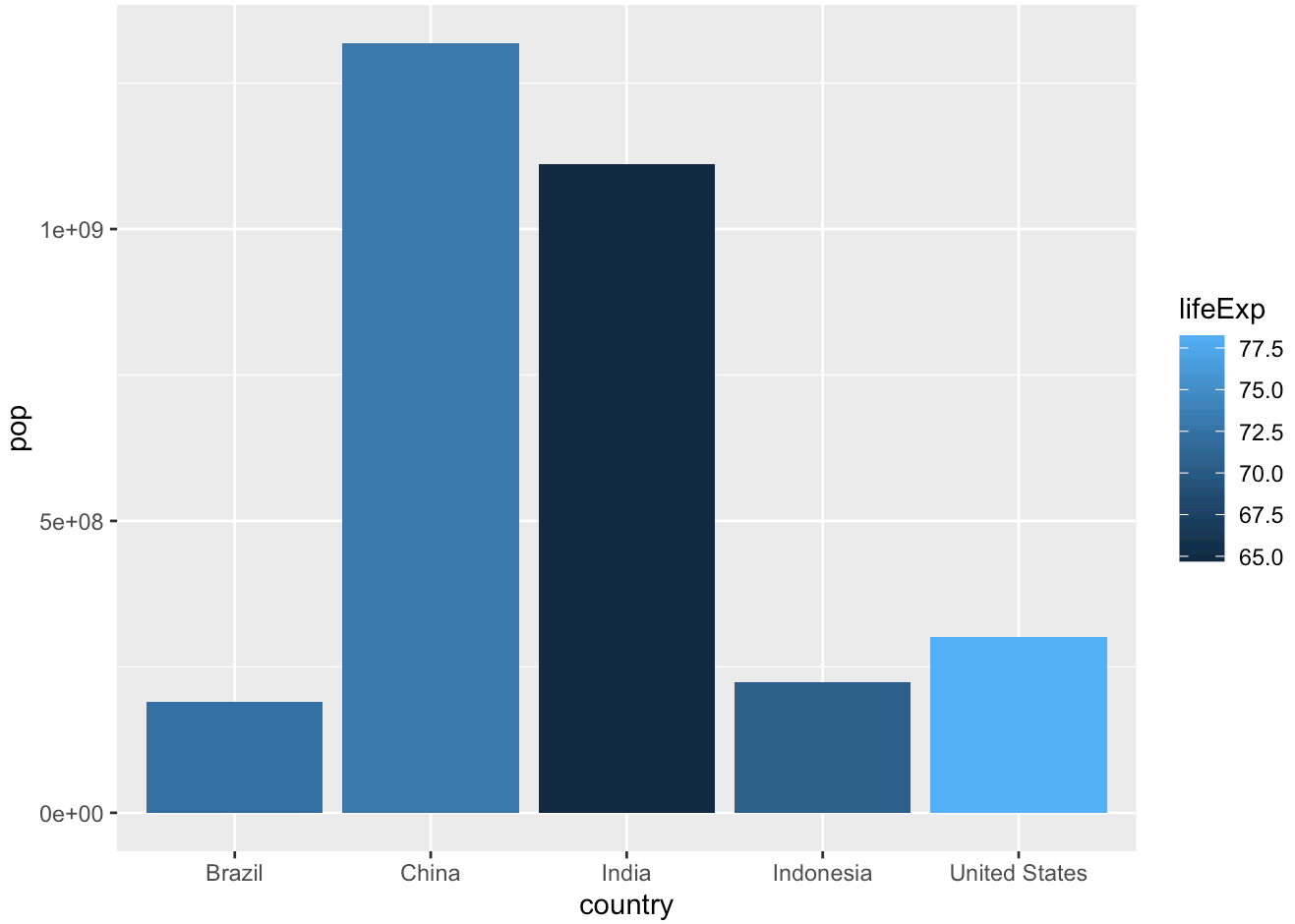


Filling bars with color

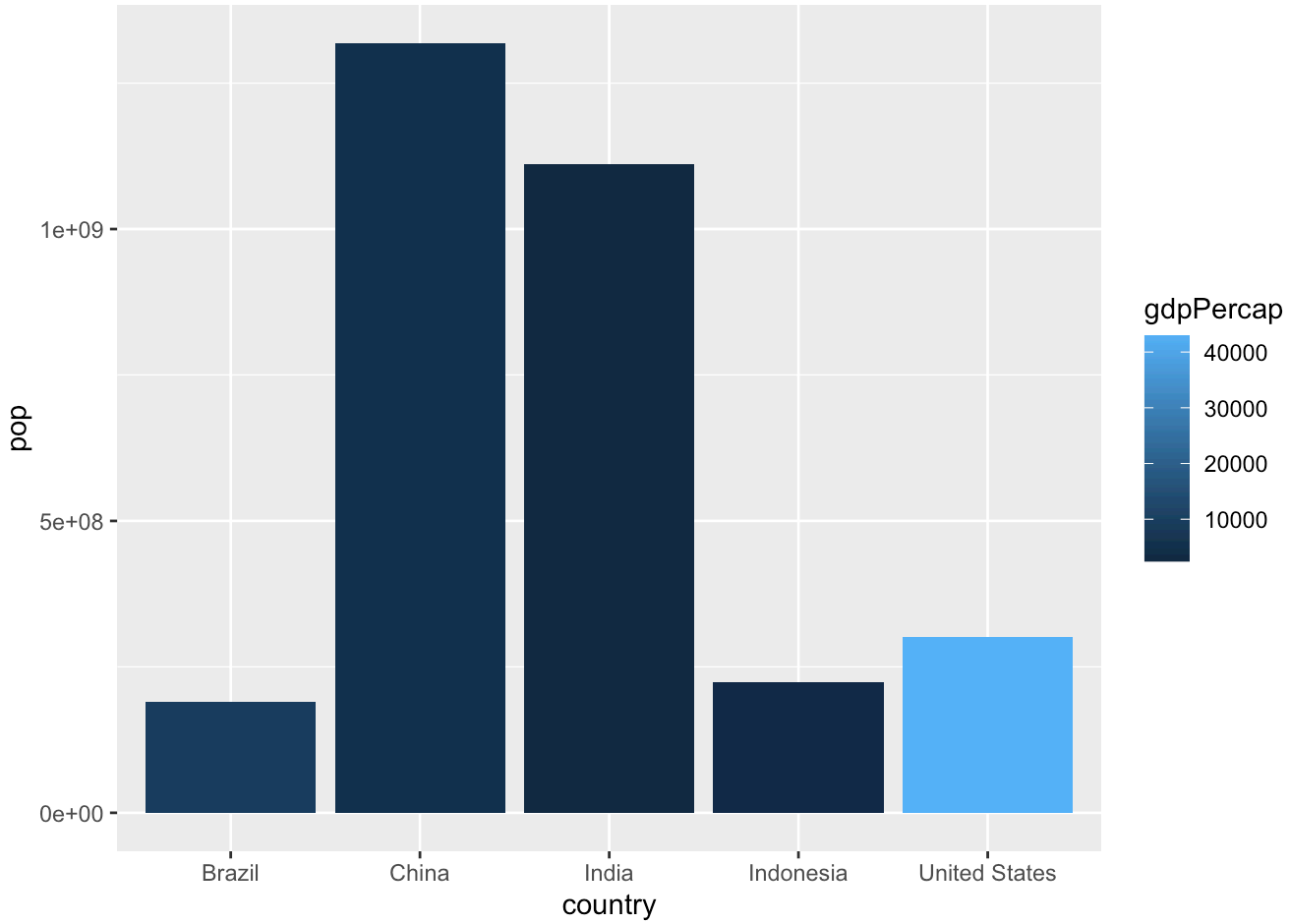
```
ggplot(gapminder_top5) +
  geom_col(aes(x = country, y = pop, fill = continent))
```



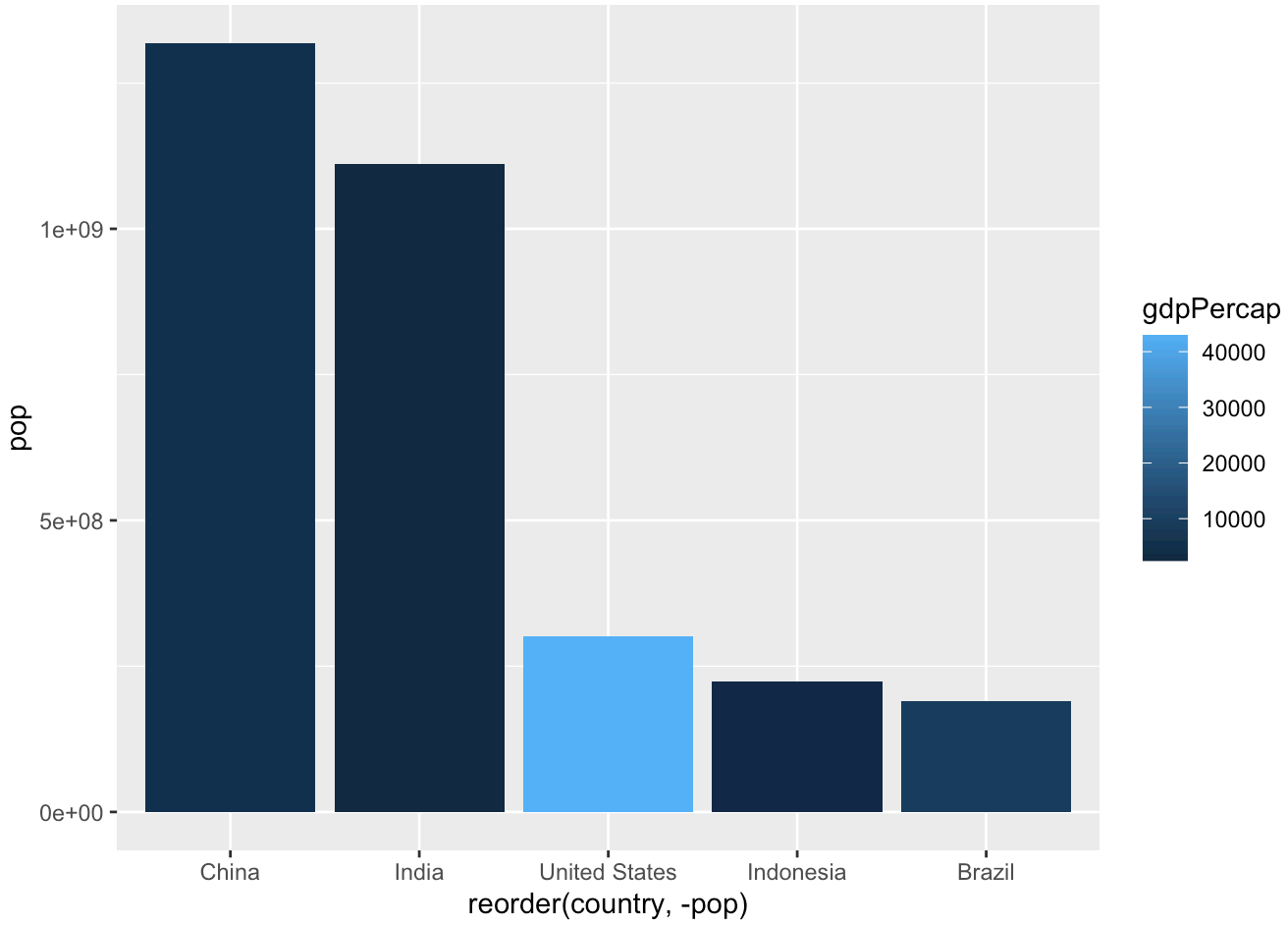

```
ggplot(gapminder_top5) +
  geom_col(aes(x = country, y = pop, fill = lifeExp))
```



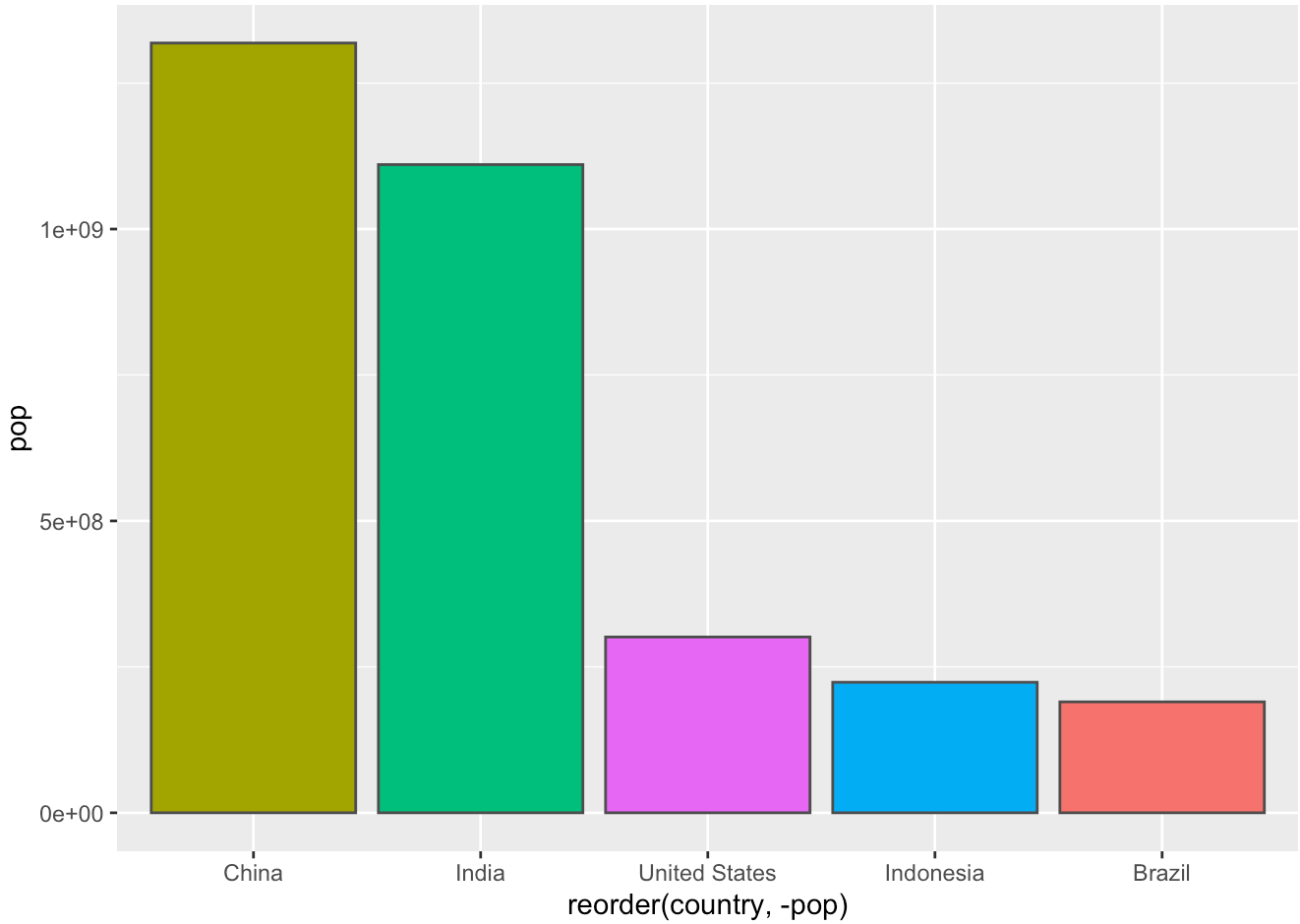

```
ggplot(gapminder_top5) +
  aes(x=country, y=pop, fill=gdpPercap) +
  geom_col()
```




```
# Change order of the bars
ggplot(gapminder_top5) +
  aes(x=reorder(country, -pop), y=pop, fill=gdpPercap) +
  geom_col()
```




```
# Fill by country
ggplot(gapminder_top5) +
  aes(x=reorder(country, -pop), y=pop, fill=country) +
  geom_col(col="gray30") +
  guides(fill="none")
```

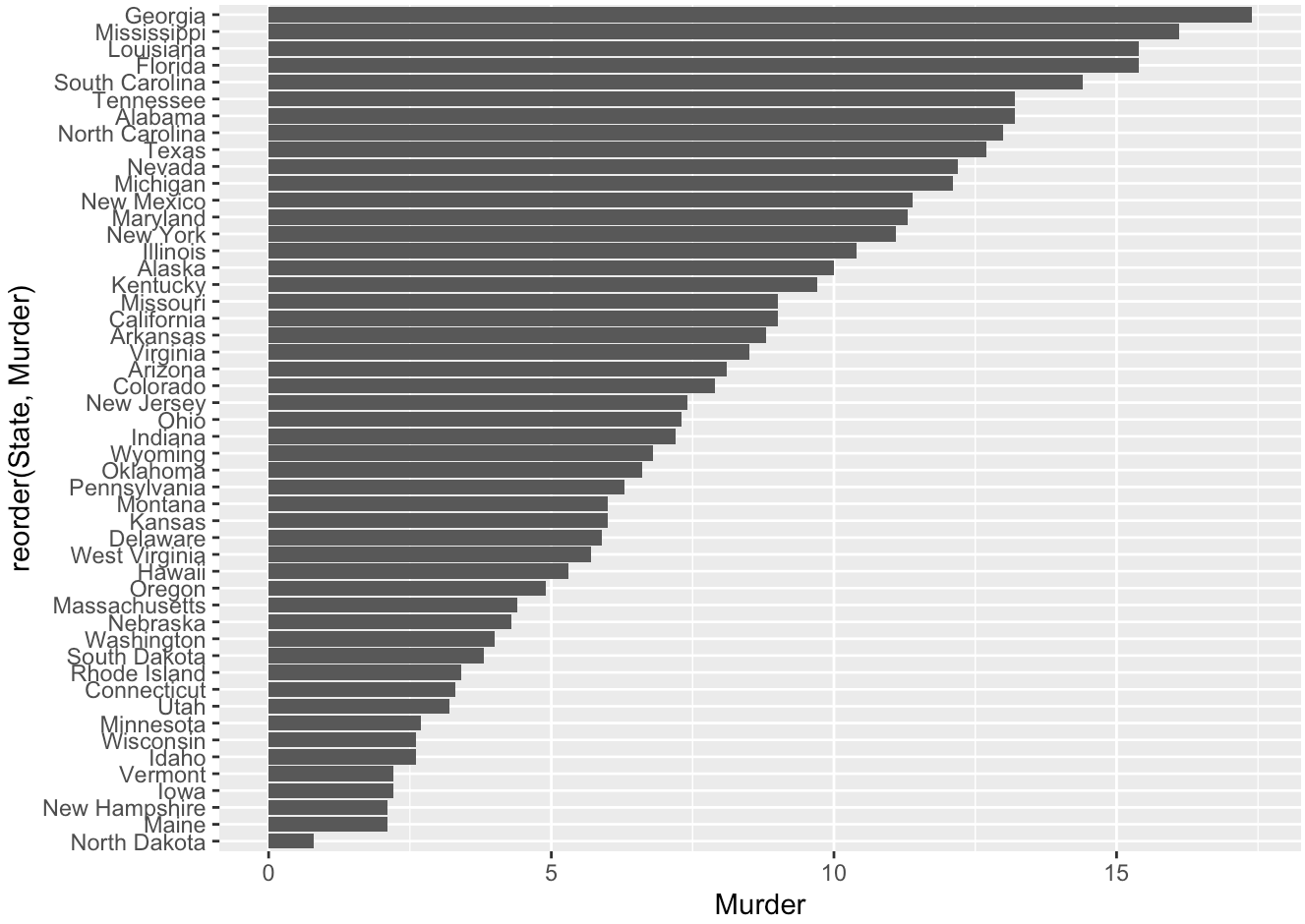


Flipping bar charts

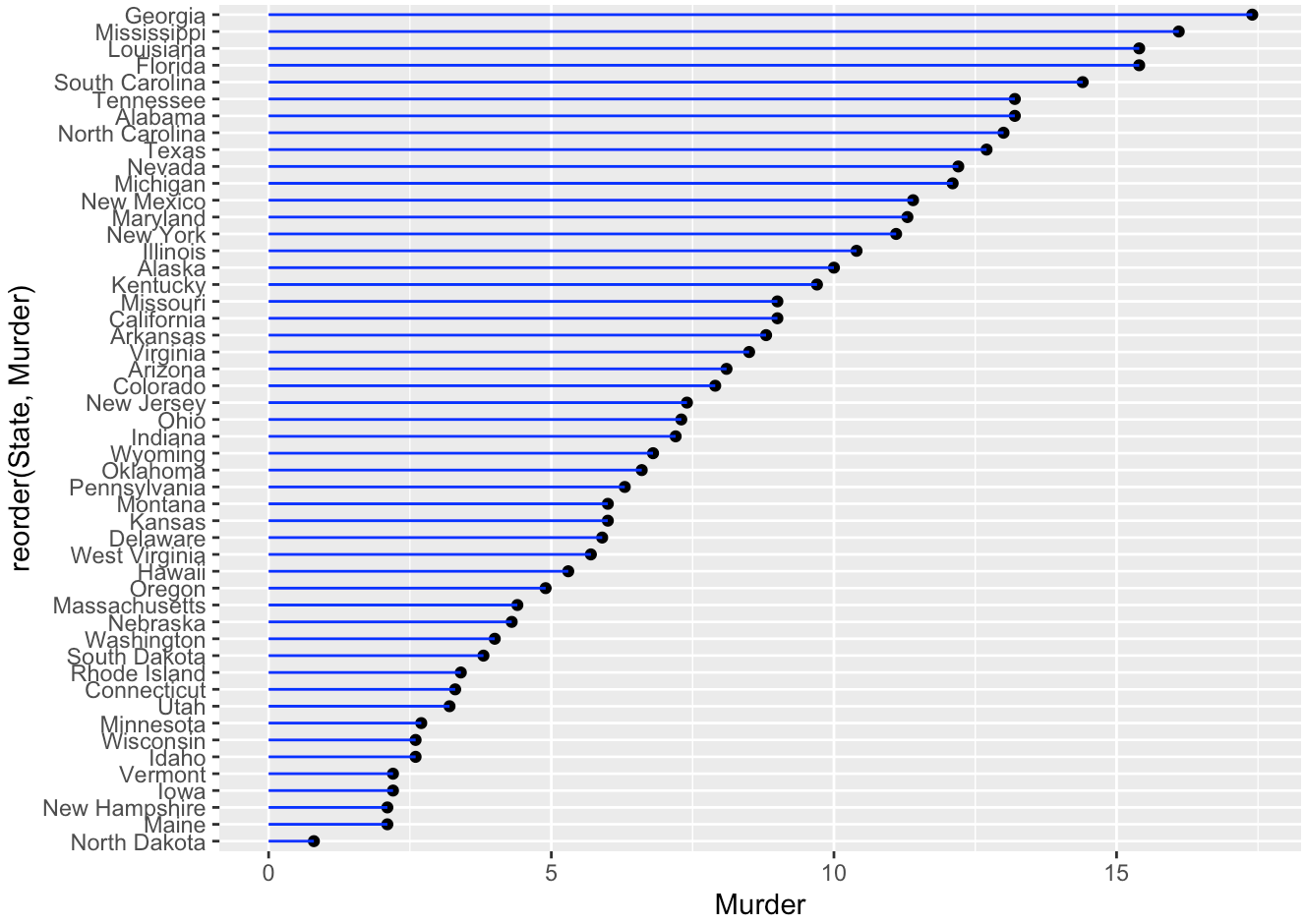
```
head(USArrests)
```

```
           Murder Assault UrbanPop Rape
Alabama      13.2     236       58 21.2
Alaska       10.0     263       48 44.5
Arizona       8.1     294       80 31.0
Arkansas      8.8     190       50 19.5
California    9.0     276       91 40.6
Colorado      7.9     204       78 38.7
```

```
USArrests$State <- rownames(USArrests)
ggplot(USArrests) +
  aes(x=reorder(State,Murder), y=Murder) +
  geom_col() +
  coord_flip()
```




```
# This looks crowded
ggplot(USArrests) +
  aes(x=reorder(State,Murder), y=Murder) +
  geom_point() +
  geom_segment(aes(x=State,
                   xend=State,
                   y=0,
                   yend=Murder), color="blue") +
  coord_flip()
```



Extensions: Animation

```
install.packages("gifski")
```

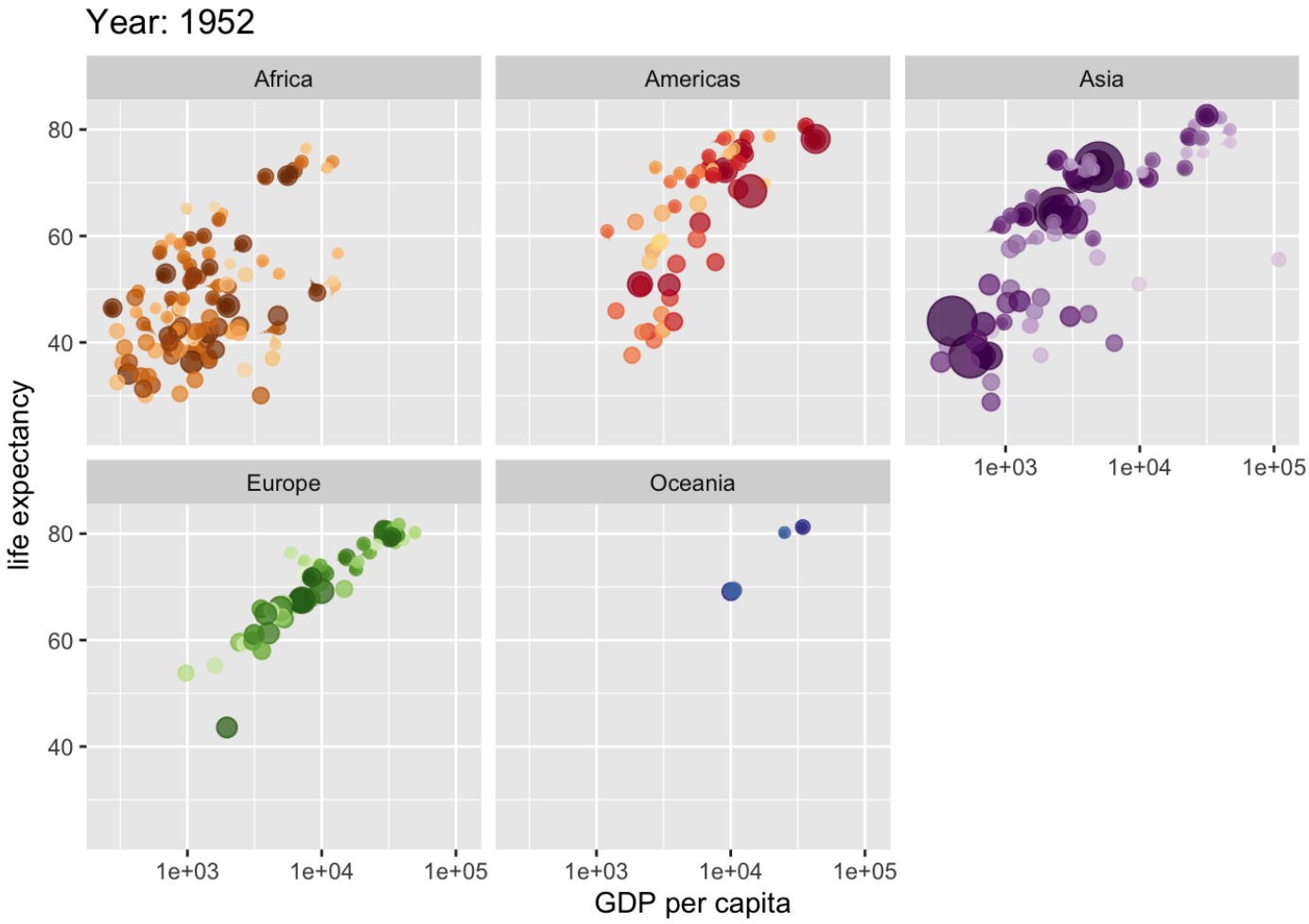
```
The downloaded binary packages are in
    /var/folders/wc/y60y10bj5jz0zzxkrq739z580000gn/T//RtmpRfONWV/downloaded_packages
```

```
install.packages("gganimate")
```

```
The downloaded binary packages are in
    /var/folders/wc/y60y10bj5jz0zzxkrq739z580000gn/T//RtmpRfONWV/downloaded_packages
```

```
library(gapminder)
library(gganimate)

# Setup nice regular ggplot of the gapminder data
ggplot(gapminder, aes(gdpPercap, lifeExp, size = pop, colour = country)) +
  geom_point(alpha = 0.7, show.legend = FALSE) +
  scale_colour_manual(values = country_colors) +
```

```
  scale_size(range = c(2, 12)) +
  scale_x_log10() +
  # Facet by continent
  facet_wrap(~continent) +
  # Here comes the gganimate specific bits
  labs(title = 'Year: {frame_time}', x = 'GDP per capita', y = 'life expectancy') +
  transition_time(year) +
  shadow_wake(wake_length = 0.1, alpha = FALSE)
```

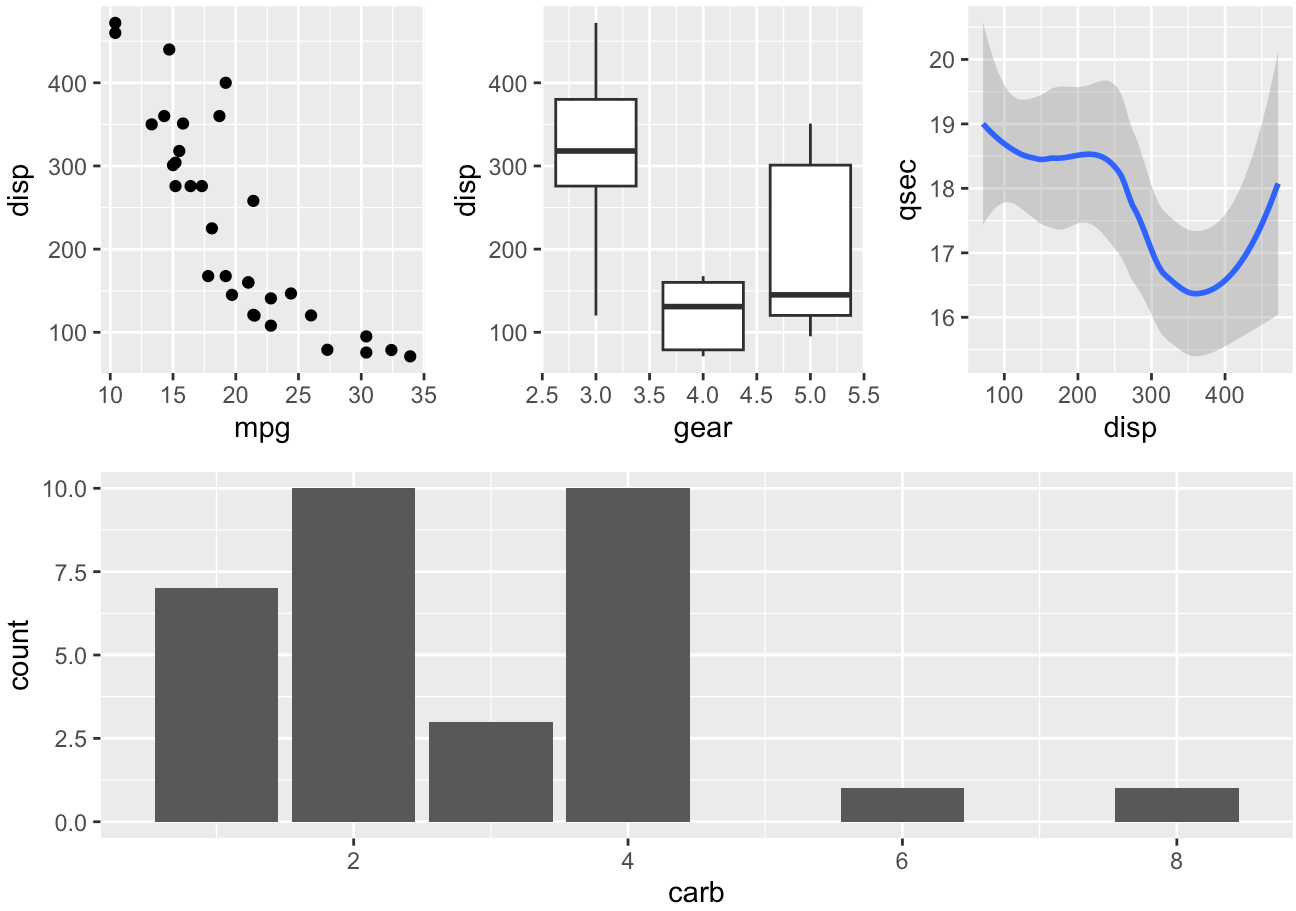


Combining plots

```
library(patchwork)

# Setup some example plots
p1 <- ggplot(mtcars) + geom_point(aes(mpg, disp))
p2 <- ggplot(mtcars) + geom_boxplot(aes(gear, disp, group = gear))
p3 <- ggplot(mtcars) + geom_smooth(aes(disp, qsec))
p4 <- ggplot(mtcars) + geom_bar(aes(carb))

# Use patchwork to combine them here:
```

```
        (p1 | p2 | p3) /
              p4
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



SessionInfo

```
        sessionInfo()
```

```
R version 4.3.2 (2023-10-31)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS 15.0.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-
arm64/Resources/lib/libRlapack.dylib;  LAPACK version 3.11.0

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

time zone: America/Los_Angeles
```

```
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] patchwork_1.2.0 gganimate_1.0.9 dplyr_1.1.4     gapminder_1.0.0
[5] ggrepel_0.9.5   ggplot2_3.5.1

loaded via a namespace (and not attached):
 [1] Matrix_1.6-5      gtable_0.3.5      jsonlite_1.8.8    crayon_1.5.3
 [5] compiler_4.3.2    tidyselect_1.2.1  Rcpp_1.0.13       progress_1.2.3
 [9] splines_4.3.2     scales_1.3.0      yaml_2.3.10       fastmap_1.2.0
[13] lattice_0.21-9    R6_2.5.1          labeling_0.4.3    generics_0.1.3
[17] knitr_1.48        htmlwidgets_1.6.4 tibble_3.2.1      munsell_0.5.1
[21] pillar_1.9.0      rlang_1.1.4       utf8_1.2.4        stringi_1.8.4
[25] xfun_0.47         cli_3.6.3         tweenr_2.0.3      withr_3.0.1
[29] magrittr_2.0.3    mgcv_1.9-1        digest_0.6.37     grid_4.3.2
[33] rstudioapi_0.16.0 hms_1.1.3         lifecycle_1.0.4   nlme_3.1-163
[37] prettyunits_1.2.0 vctrs_0.6.5       evaluate_0.24.0   glue_1.7.0
[41] farver_2.1.2      gifski_1.32.0-1   fansi_1.0.6       colorspace_2.1-1
[45] rmarkdown_2.28    tools_4.3.2       pkgconfig_2.0.3   htmltools_0.5.8.1
```