Class 5: Data Visualization with ggplot

Josefina O'Toole (PID: A16978557)

Intro to ggplot

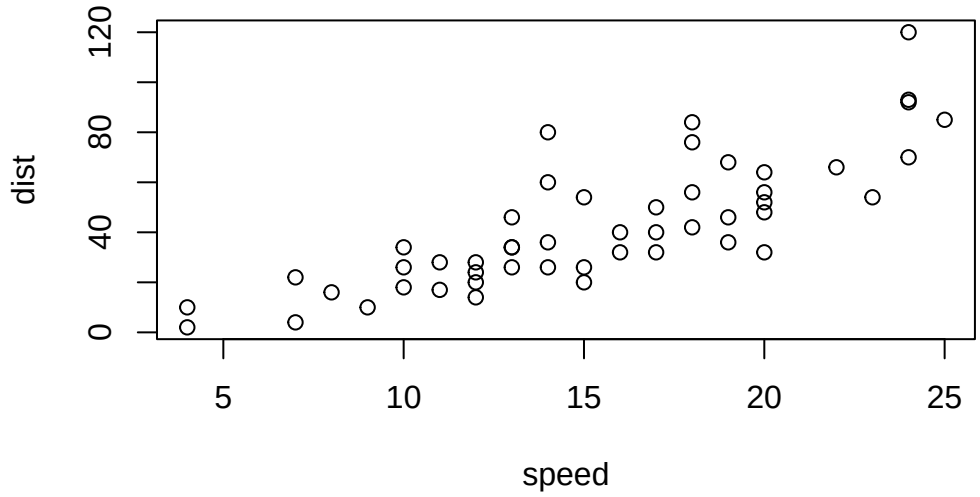
There are many graphic systems in R (ways to make plots and figures). These include "base" R plots. Today we will focus mostly on the **ggplot2** package.

Let's start with a plot of a simple in-built dataset called `cars`.

```
head(cars)
```

```
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

```
plot(cars)
```



Let's see how we can make this figure using **ggplot**. First I need to install this package on my computer. To install any R package I use the function `install.packages()`.

I will run 'install.packages("ggplot2") in my R console not this quarto document!

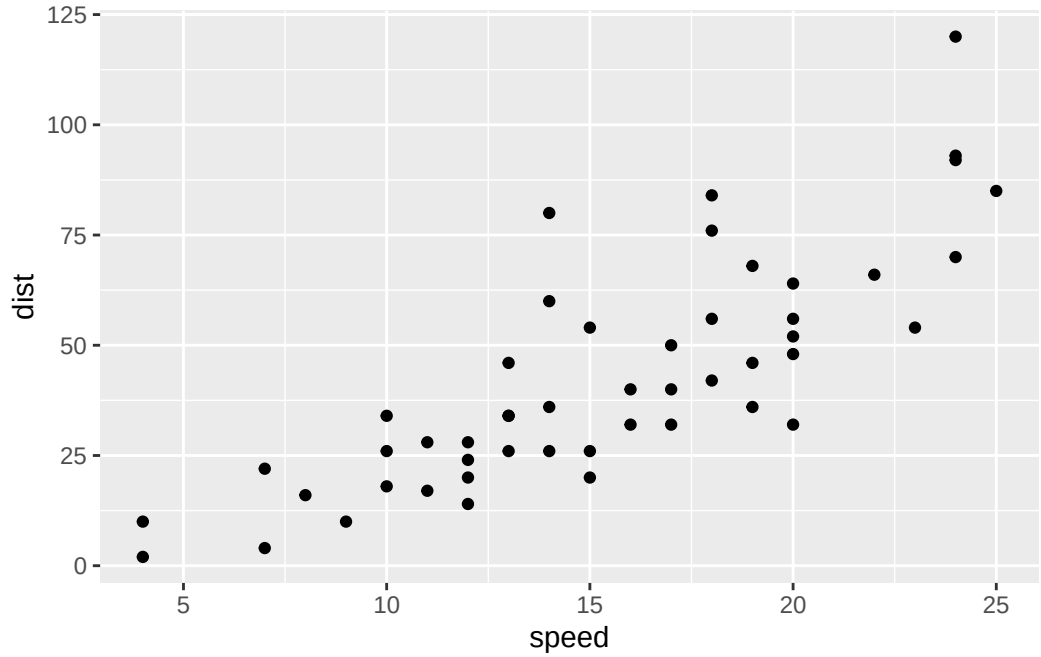
Before I can use any functions from add on packages I need to load the package from "library()" with the 'library(ggplot2) call.

```
library(ggplot2)
ggplot(cars)
```

All ggplot figures have at least 3 things (called layers). These include:

- **data** (the input dataset I want to plot from),
- **aes** (the aesthetic mapping of the data to my plot),
- **geoms** (the geom_point(), geom_line(), etc. that I want to draw).

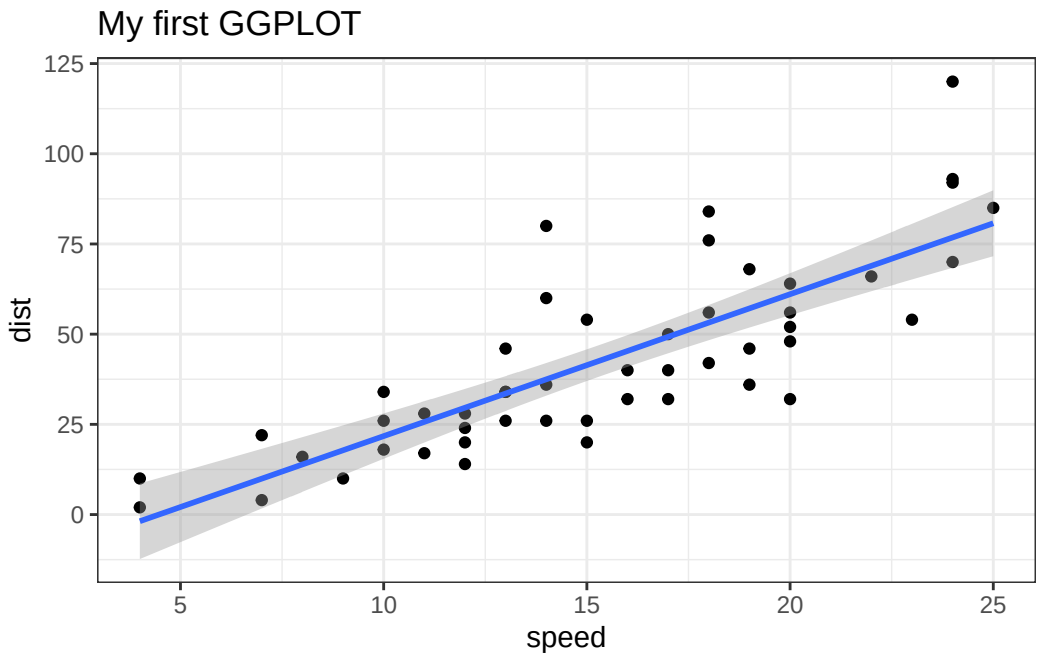
```
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point()
```



Let's add a line to show the relationship here:

```
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  geom_smooth(method="lm") +
  theme_bw() +
  labs(title="My first GGPLOT")
```

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



For which phases is data visualization important in our scientific workflows?

All of the above

True or False? The ggplot2 package comes already installed with R?

FALSE

Which plot types are typically NOT used to compare distributions of numeric variables?

Network graphs

Which statement about data visualization with ggplot2 is incorrect?

ggplot2 is the only way to create plots in R

Which geometric layer should be used to create scatter plots in ggplot2?

geom_point()

Gene expression figure

The code to read the dataset

```
url <- "https://bioboot.github.io/bimm143_S20/class-material/up_down_expression.txt"
genes <- read.delim(url)
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
```

How many genes are in this dataset?

```
nrow(genes)
```

```
[1] 5196
```

How many columns did you find?

```
colnames(genes)
```

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

```
ncol(genes)
```

```
[1] 4
```

Use the table() function on the State column of this data.frame to find out how many 'up' regulated genes there are. What is your answer?

```
table(genes$State)
```

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

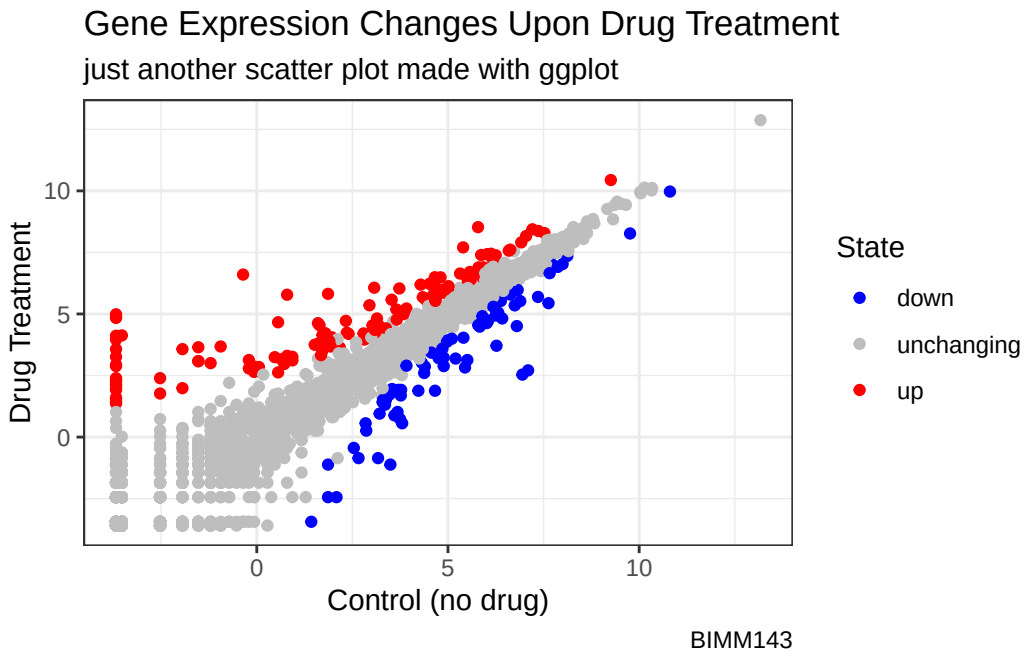
Using your values above and 2 significant figures. What fraction of total genes is up-regulated in this dataset?

```
n.tot <- nrow(genes)
vals <- table(genes$State)

vals.percent <- vals/n.tot * 100
round(vals.percent, 2)
```

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

Polishing our figure:

```
ggplot(genes) +
  aes(x=Condition1, y=Condition2, col=State) +
  geom_point() +
  theme_bw() +
  labs(title="Gene Expression Changes Upon Drug Treatment",
      subtitle="just another scatter plot made with ggplot",
      caption="BIMM143",
      x="Control (no drug)",
      y="Drug Treatment") +
  scale_colour_manual( values = c("blue", "gray", "red") )
```



Automated Quarto information:

Quarto

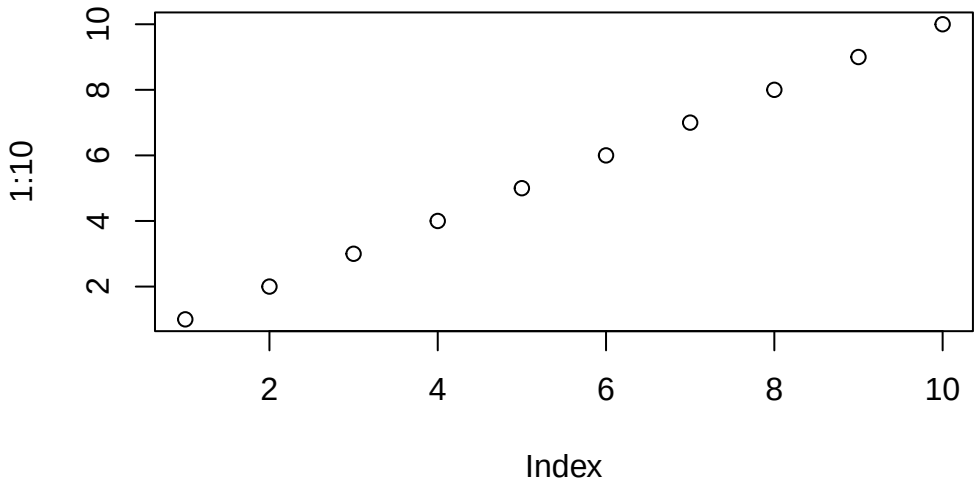
Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see https://quarto.org.

Something **else** here

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:

```
plot(1:10)
```



You can add options to executable code like this

```
[1] 4
```

The `echo: false` option disables the printing of code (only output is displayed).

7. Going Further

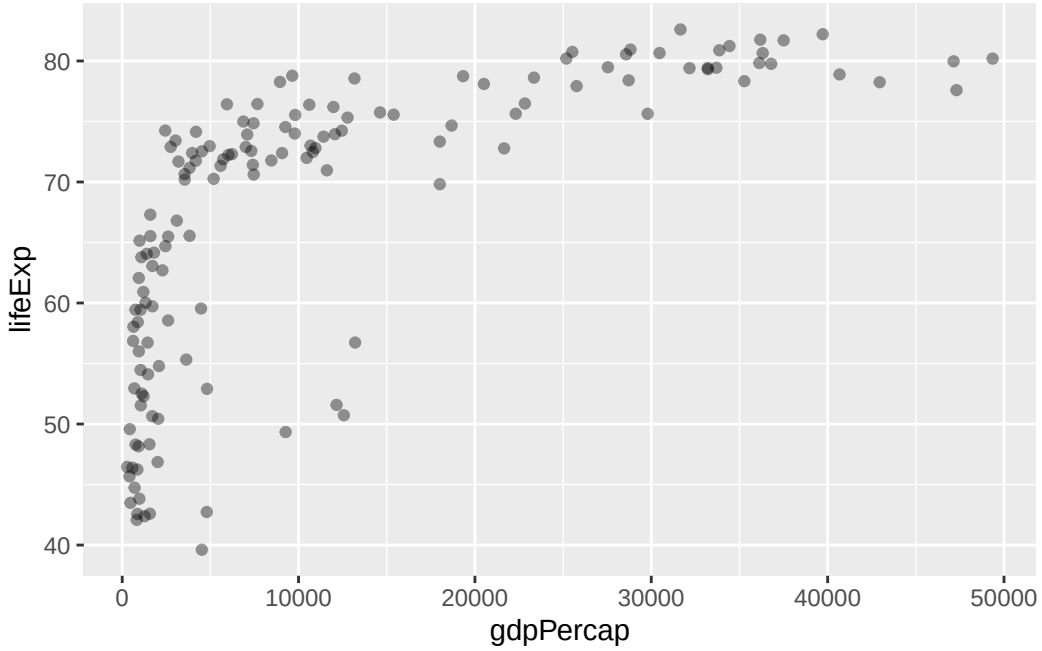
Using the gapminder and dplyr packages:

```
# File location online
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.tsv

gapminder <- read.delim(url)


# install.packages("dplyr")  ## un-comment to install if needed
library(dplyr)
```

```
Attaching package: 'dplyr'
```

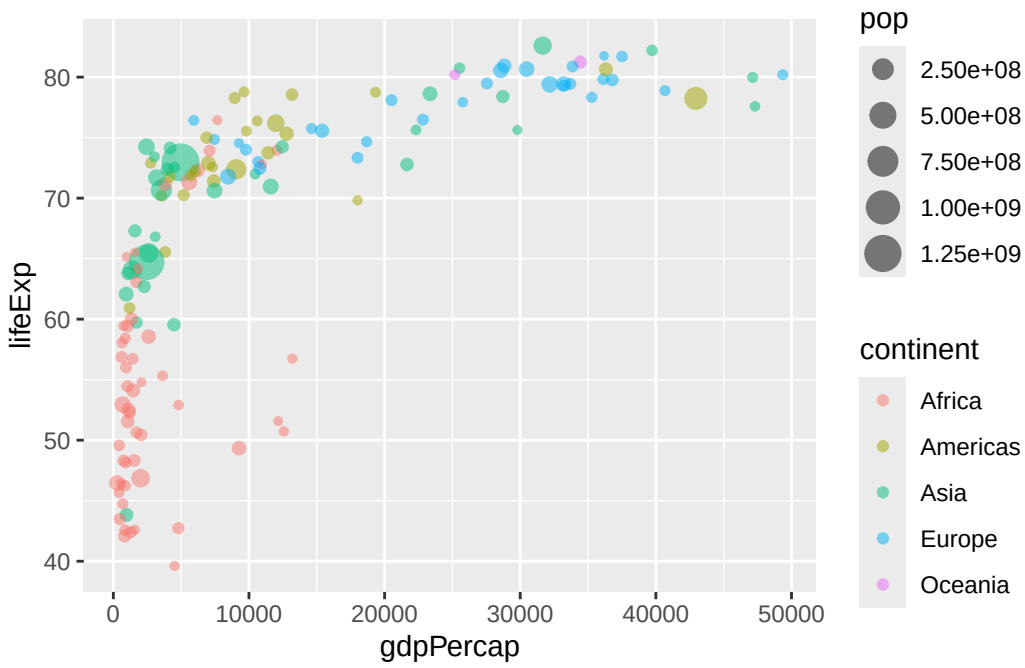
```
The following objects are masked from 'package:stats':

    filter, lag

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

    intersect, setdiff, setequal, union
```

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

ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point(alpha=0.4)
```



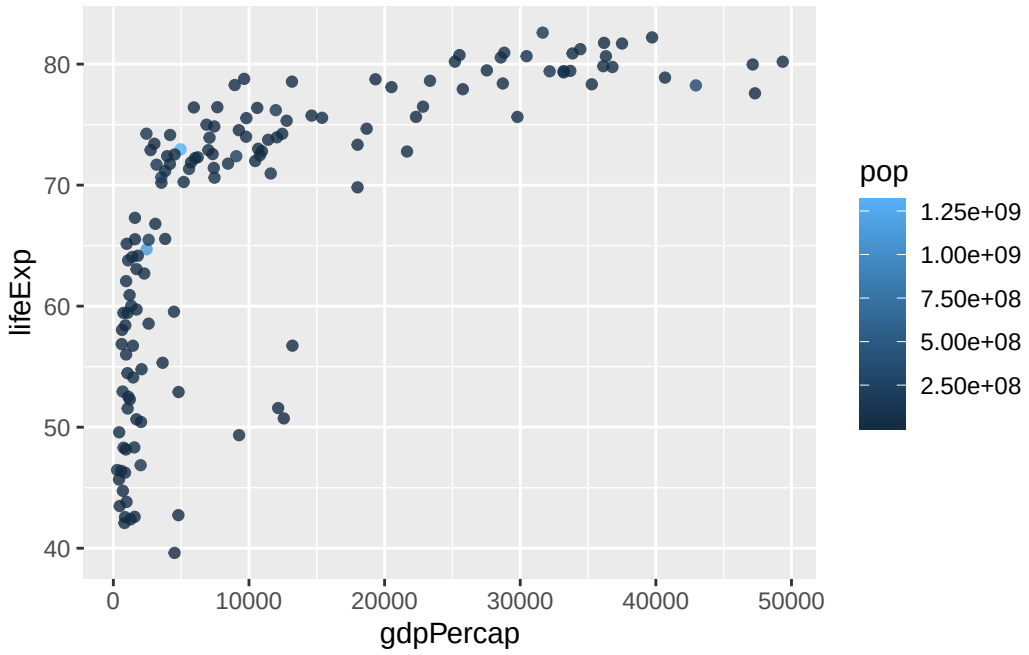
Adding more variables to aes():

```r
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.5)
```



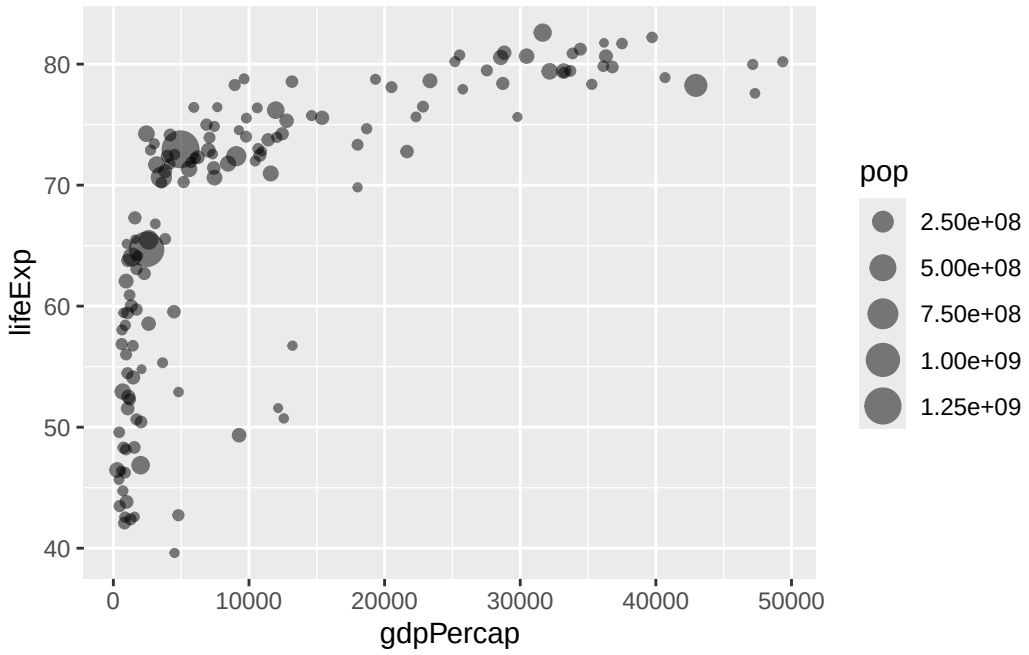
Coloring the points by the numeric variable population (pop):

```
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp, color=pop) +
  geom_point(alpha=0.8)
```



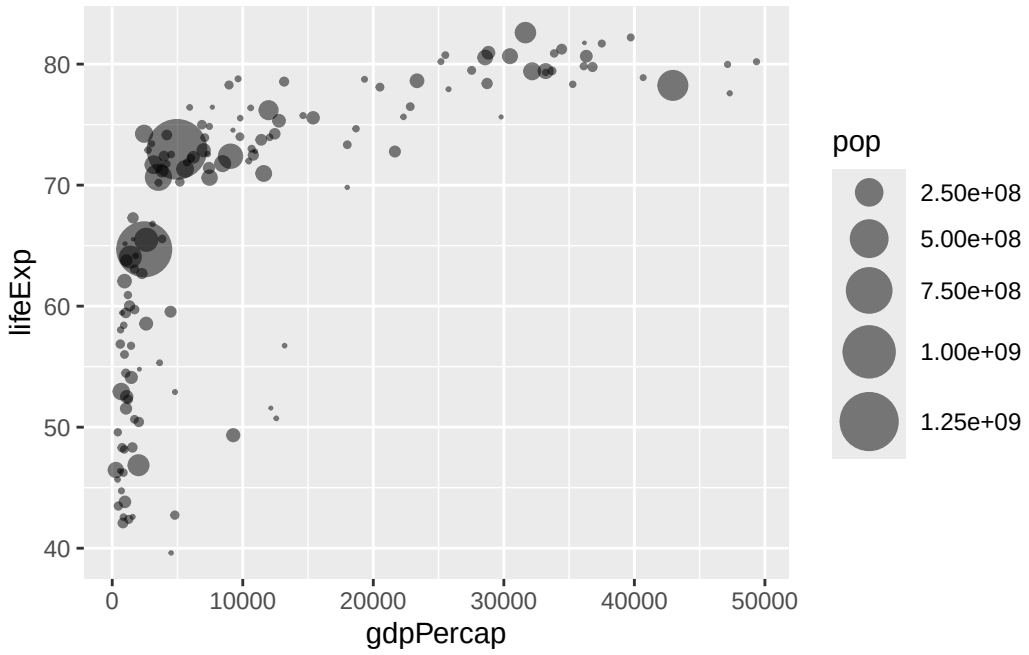
Adjusting point size:

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



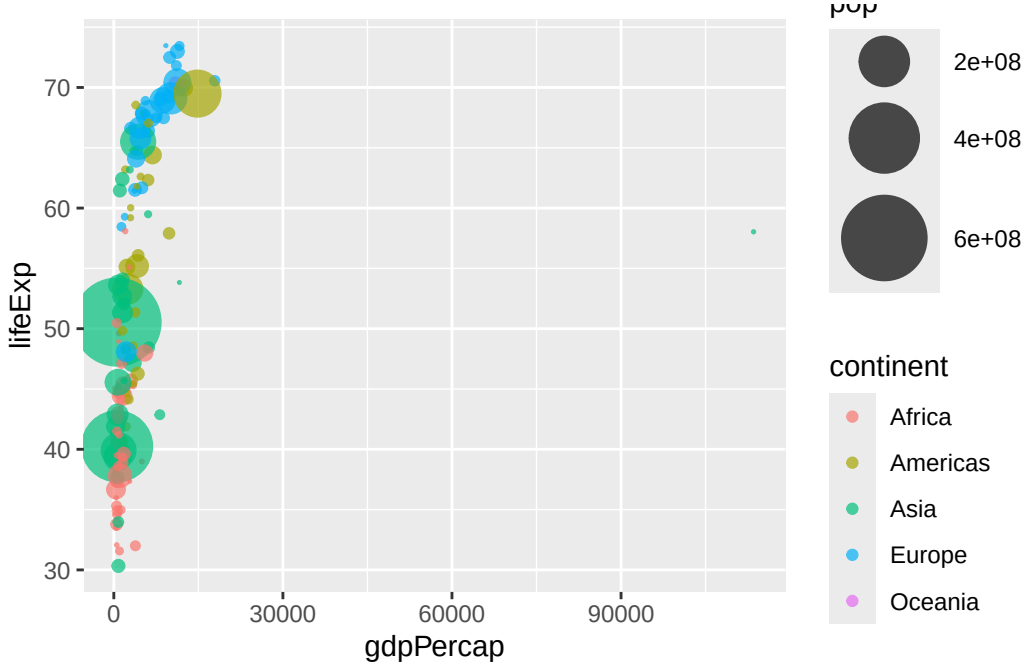
Scaling the point size information:

```
ggplot(gapminder_2007) +
  geom_point(aes(x=gdpPercap, y=lifeExp,
                 size=pop), alpha=0.5) +
  scale_size_area(max_size=10)
```



Can you adapt the code you have learned thus far to reproduce our gapminder scatter plot for the year 1957? What do you notice about this plot is it easy to compare with the one for 2007?

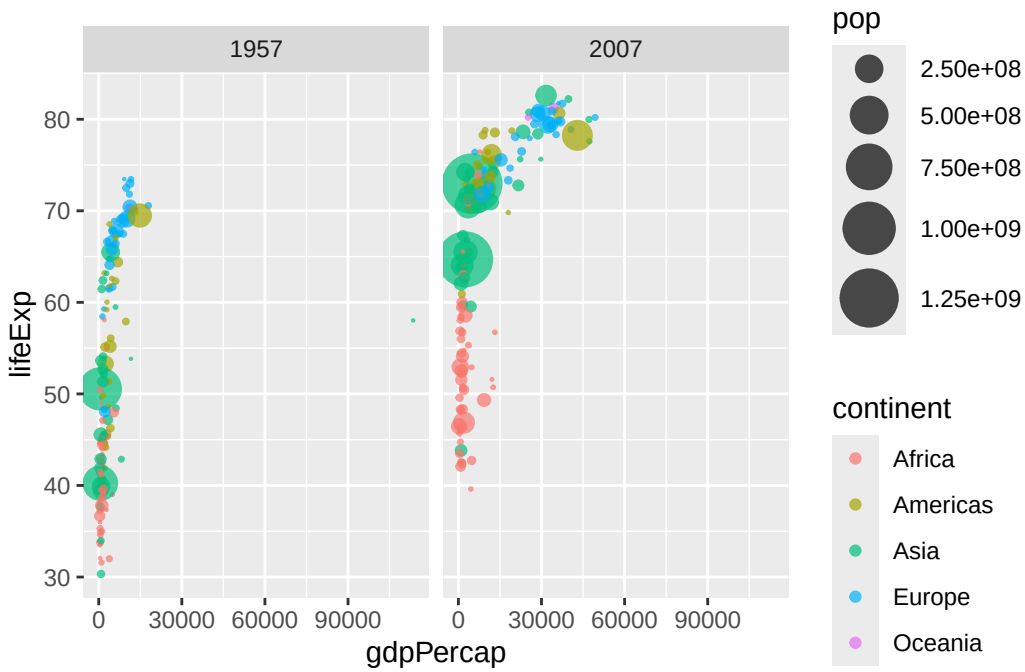
```
gapminder_1957 <- gapminder %>% filter(year==1957)

ggplot(gapminder_1957) +
  geom_point(alpha=0.7) +
  aes(x=gdpPercap, y=lifeExp, col=continent, size=pop) +
  scale_size_area(max_size=15)
```



This plot is not easy to compare because the graphs are not positioned next to each other.

Do the same steps above but include 1957 and 2007 in your input dataset for ggplot(). You should now include the layer facet_wrap(~year) to produce the following plot:

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

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

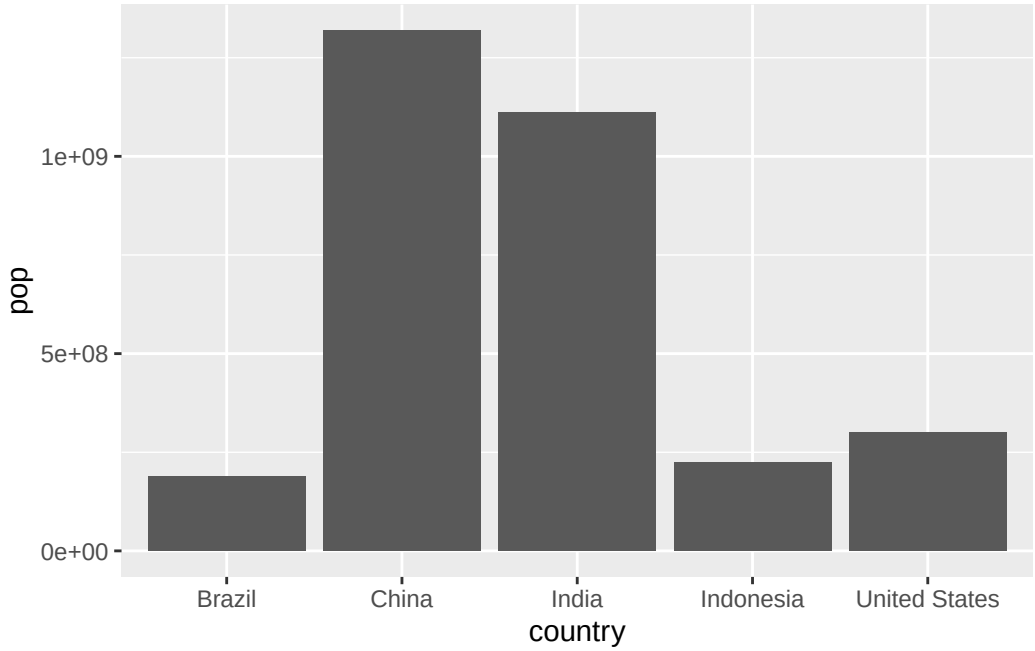


Introduction to bar charts

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

gapminder_top5
```

```
        country continent year lifeExp        pop gdpPercap
1         China      Asia 2007  72.961 1318683096  4959.115
2         India      Asia 2007  64.698 1110396331  2452.210
3 United States  Americas 2007  78.242  301139947 42951.653
4     Indonesia      Asia 2007  70.650  223547000  3540.652
5        Brazil  Americas 2007  72.390  190010647  9065.801
```

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

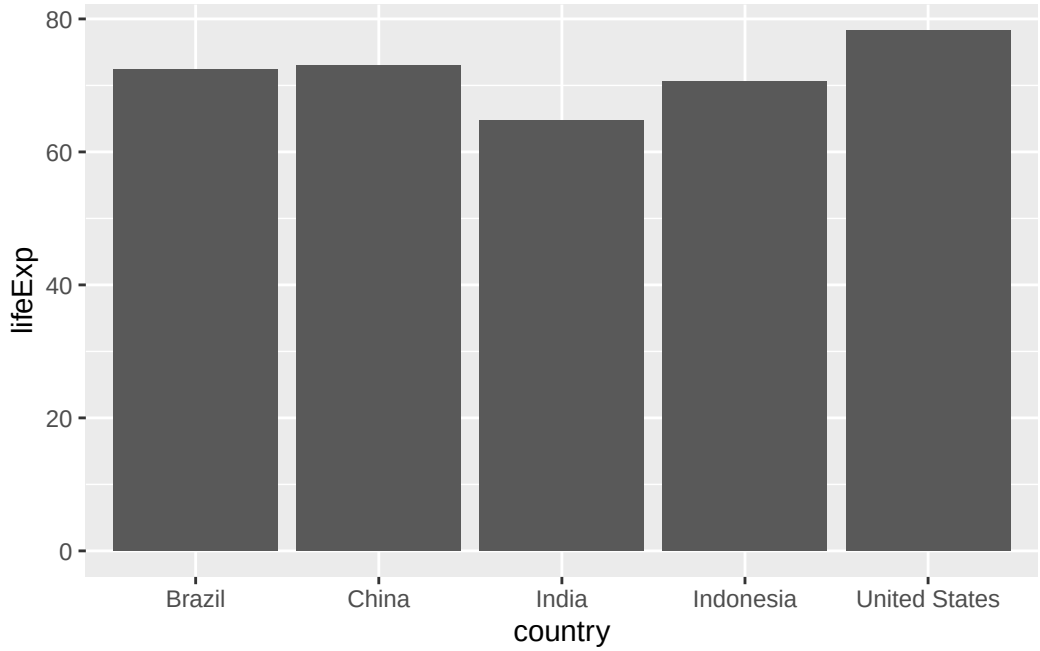


Create a bar chart showing the life expectancy of the five biggest countries by population in 2007.

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

gapminder_top5
```

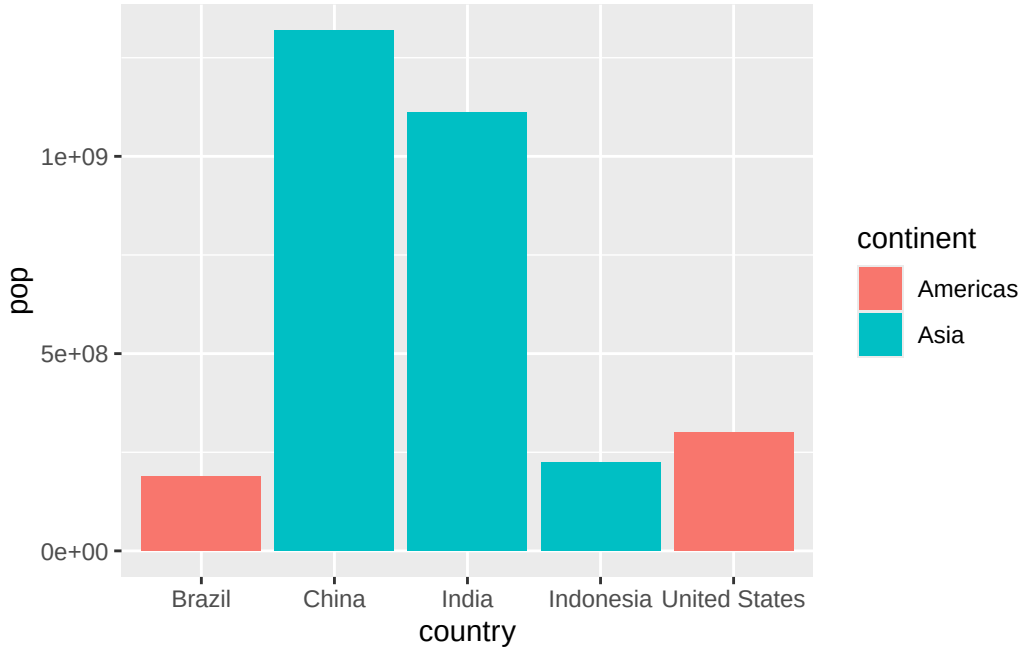
```
        country continent year lifeExp        pop gdpPercap
1         China      Asia 2007  72.961 1318683096  4959.115
2         India      Asia 2007  64.698 1110396331  2452.210
3 United States  Americas 2007  78.242  301139947 42951.653
4     Indonesia      Asia 2007  70.650  223547000  3540.652
5        Brazil  Americas 2007  72.390  190010647  9065.801
```

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



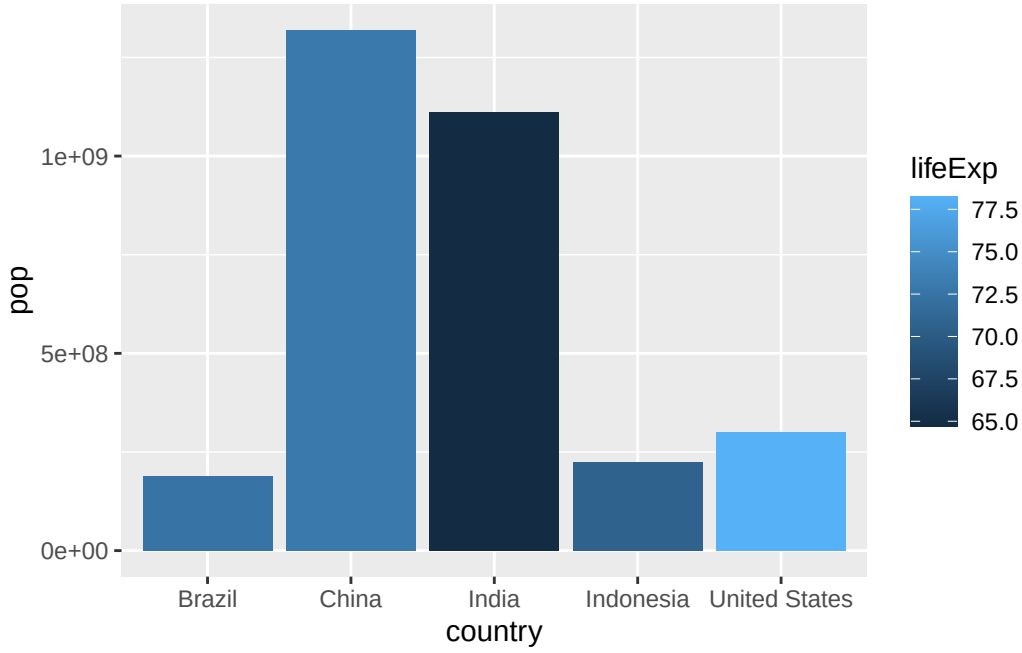
Filling bars with color

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



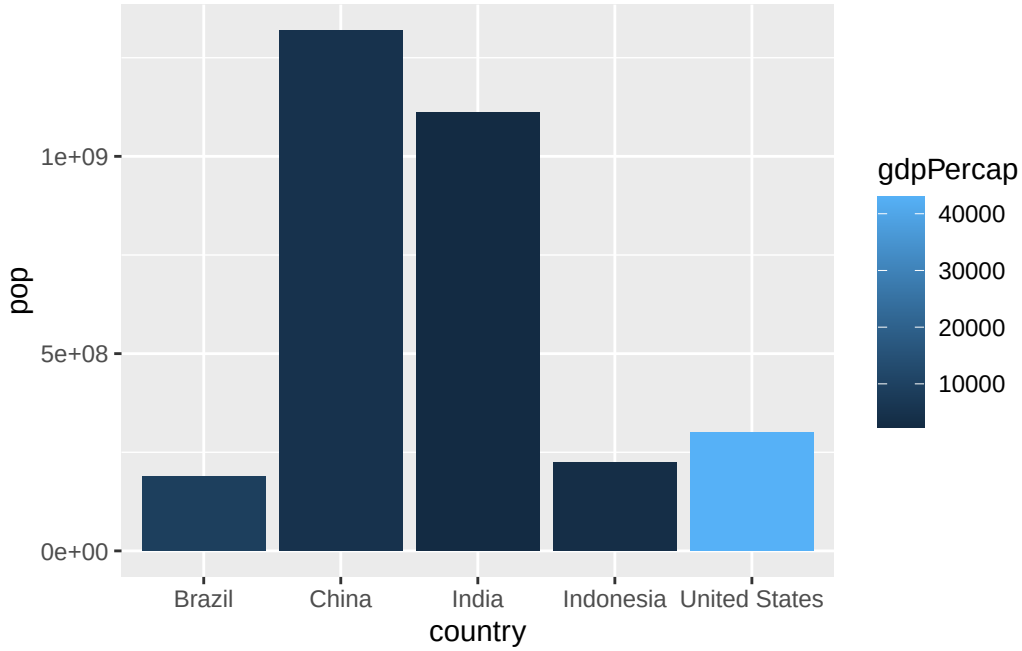
Using a numeric variable:

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



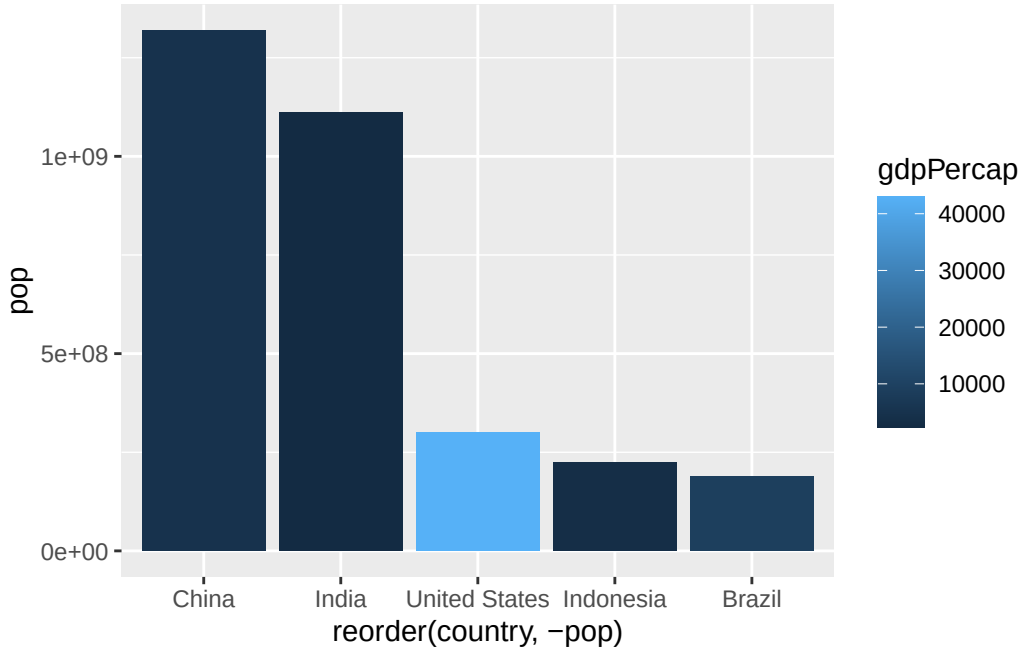
Plot population size by country. Create a bar chart showing the population (in millions) of the five biggest countries by population in 2007.

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



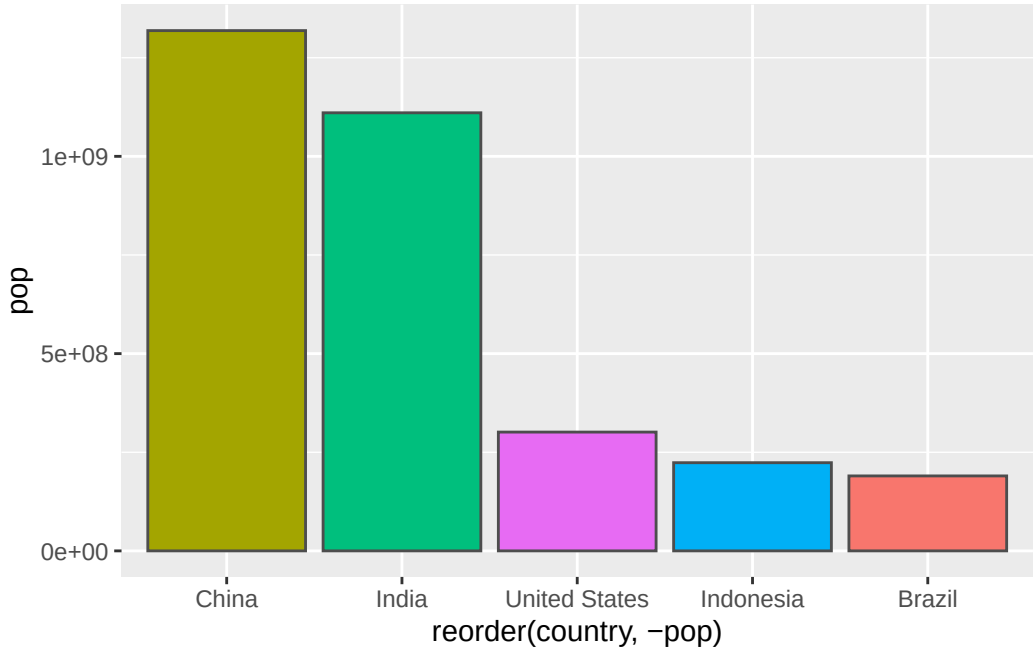
Re-order bars:

```
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")
```

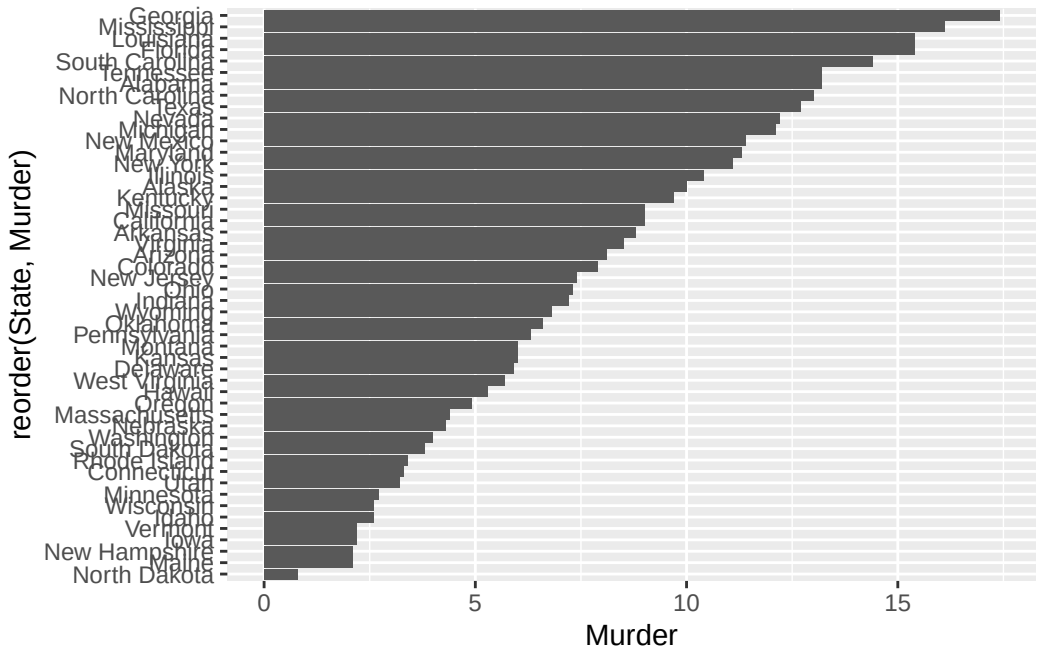


Flipping bar charts

```
head(USArrests)
```

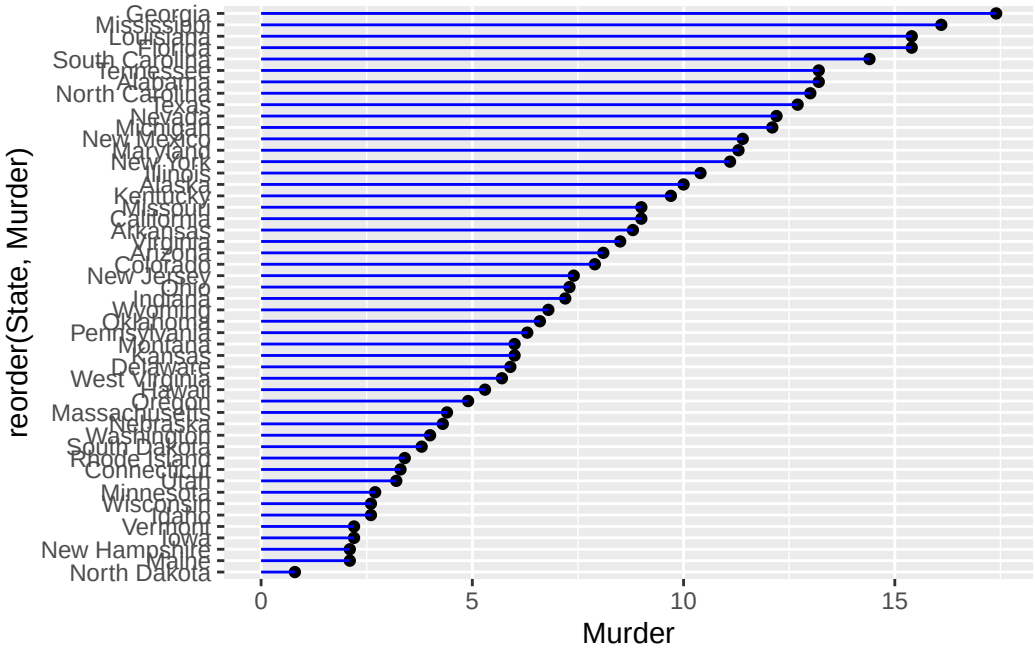
```
           Murder Assault UrbanPop Rape
Alabama      13.2     236       58 21.2
Alaska       10.0     263       48 44.5
Arizona       8.1     294       80 31.0
Arkansas      8.8     190       50 19.5
California    9.0     276       91 40.6
Colorado      7.9     204       78 38.7
```

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



Combining geom_point() & geom_segment():

```
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()
```


reorder(State, Murder)
Georgia
Mississippi
Louisiana
Florida
South Carolina
Tennessee
Alabama
North Carolina
Texas
Nevada
Michigan
New Mexico
Maryland
New York
Illinois
Alaska
Kentucky
Missouri
California
Arkansas
Virginia
Arizona
Colorado
New Jersey
Ohio
Indiana
Wyoming
Oklahoma
Pennsylvania
Montana
Kansas
Delaware
West Virginia
Hawaii
Oregon
Massachusetts
Nebraska
Washington
South Dakota
Rhode Island
Connecticut
Utah
Minnesota
Wisconsin
Idaho
Vermont
Iowa
New Hampshire
Maine
North Dakota
0
5
10
15
Murder