Class 05: Data Viz with ggplot

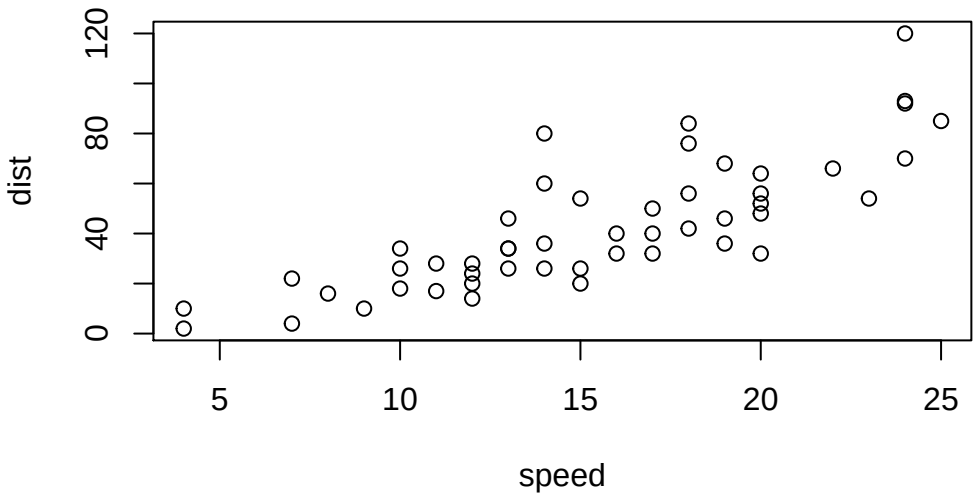
Mina Wu (PID: A59013200)

2024-01-24

#Graphics systems in R

There are many graphics systems for R. These include so-called *"base R"* and those in add-on packages like `ggplot2`.

```
plot(cars)
```



How can we make this with `ggplot2`.

This is an add-on package and I first need to install it on my computer.This install is a one time only deal.

To install any package, I use the `install.packages` function.

To use it, we need to load up the package from our library of install packages.For this, I use library `(ggplot2)`

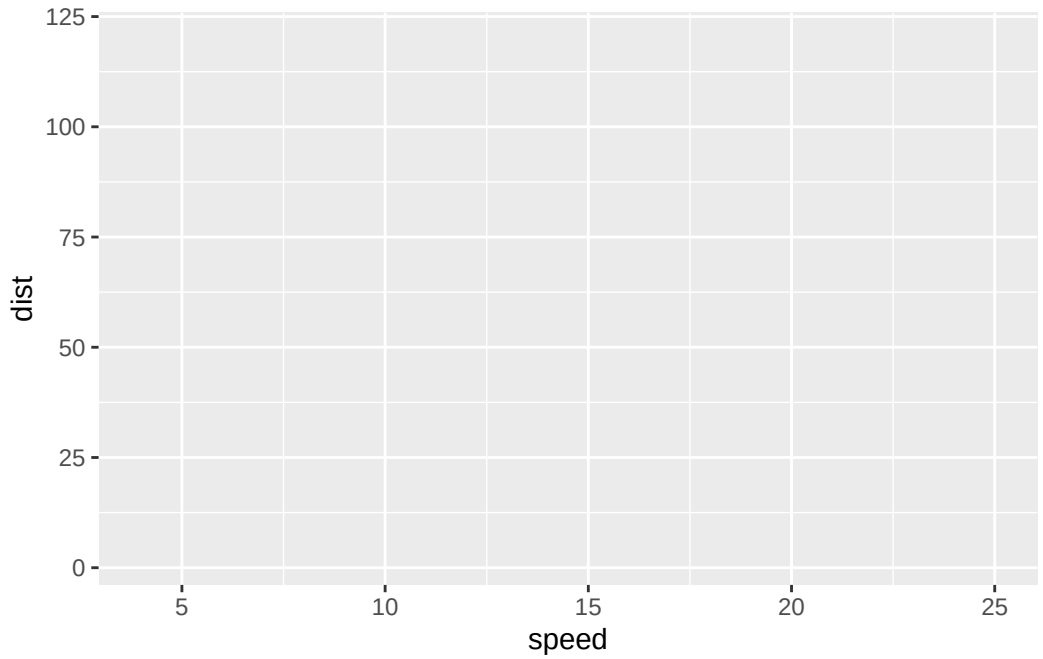
```
library (ggplot2)
ggplot(cars)
```


Using ggplot is not as strightforward as base R plot for basic plots. I have some more typing to do.

Every ggplot has at least 3 things (layers):

-data (data.frame) -aes(how the data map to the lpot) -geoms (think of this as the type of plot, e.g. points, lines, etc.)

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



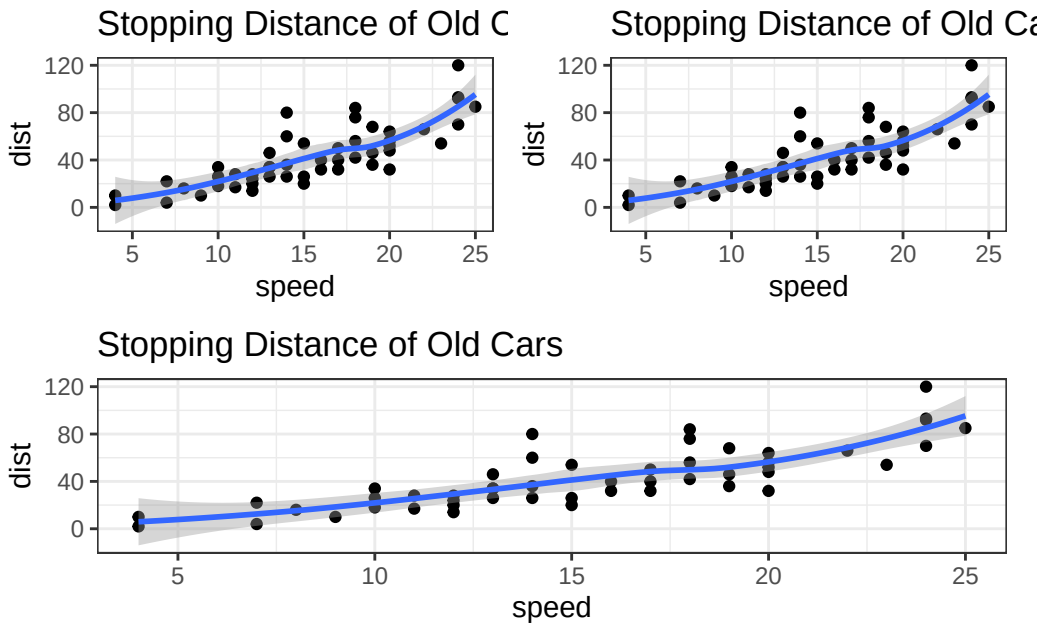
Here ggplot was more verbose- i.e. I had more typing to do- than base R. However, I can add more layers to make nicer and more complicated plots in an easy way with ggplot.

```
p <- ggplot(cars)+
  aes(x=speed, y=dist)+
  geom_point()+
  geom_smooth()+
  labs(title= "Stopping Distance of Old Cars") +
  theme_bw()
```

Use patchwork for combining plots to make an all-in-one multi-panel figure. install.packages("patchwork")

```
library (patchwork)
(p|p)/p
```

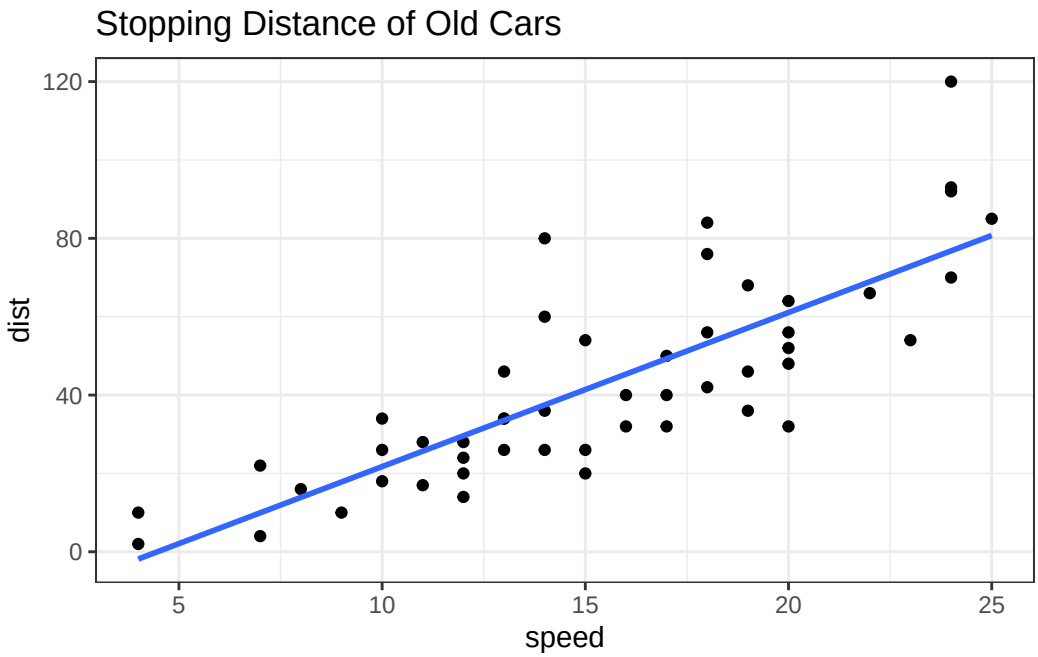
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Make it a smooth line without the shade

```
ggplot(cars)+
 aes(x=speed, y=dist)+
 geom_point()+
 geom_smooth (method= "lm", se= FALSE)+
 labs(title= "Stopping Distance of Old Cars")+
 theme_bw()
```

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



6. Creating Scatter Plots

```
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
```

Q. Use the nrow() function to find out how many genes are in this dataset. What is your answer?

```
nrow(genes)
```

```
[1] 5196
```

Q. Use the colnames() function and the ncol() function on the genes data frame to find out what the column names are (we will need these later) and how many columns there are. How many columns did you find?

```
colnames(genes)
```

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

```
ncol(genes)
```

```
[1] 4
```

Q.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? 127

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

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

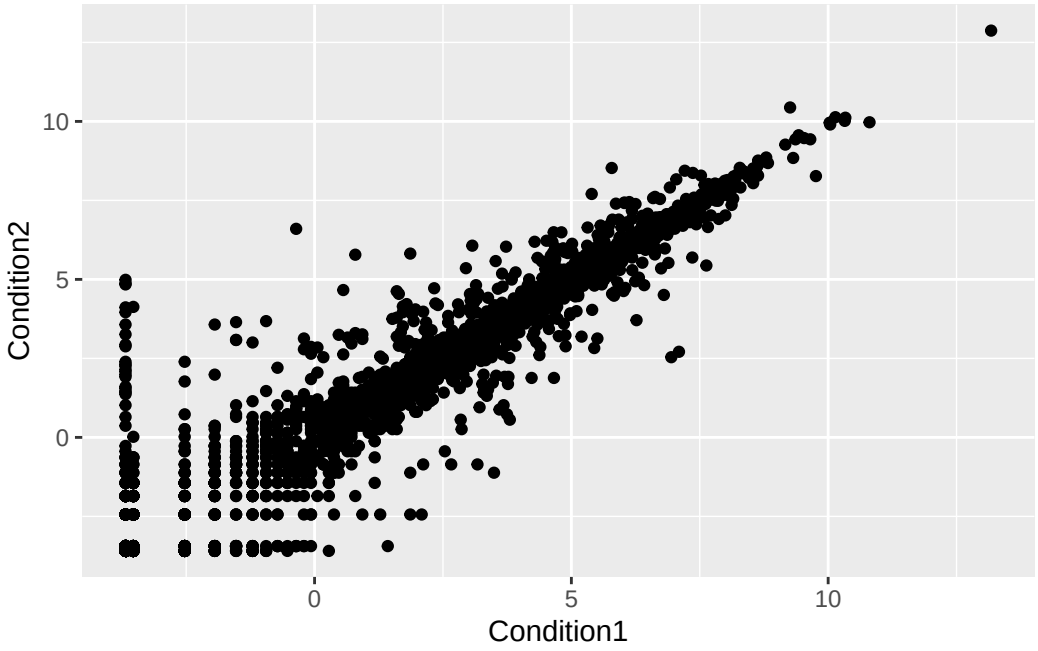
Q. Using your values above and 2 significant figures. What fraction of total genes is up-regulated in this dataset?

```
round (table(genes$State)/nrow(genes)*100,2)
```

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

Plot the Genes dataset

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



Add State column to the dataset to differentiate the genes condition (down, unchanging, and up) and save the plot as p.

```
p2 <- ggplot(genes)+
  aes(x=Condition1, y= Condition2, col=State)+
  geom_point()
```

Change the state color, add title, and change the x and y axis.

```
p3 <- p2 + scale_colour_manual(values=c("blue","gray","red"))+
  labs (title= "Gene Expression Changes Upon Drug Treatment", x= "Control (no drug)", y= "
```

Add an interactive version with plotly.install.packages("plotly")

```
library(plotly)
#ggplotly(p3)
```

7. Going Further. install.packages("dplyr"); install.packages("gapminder")

```
library(dplyr)
```

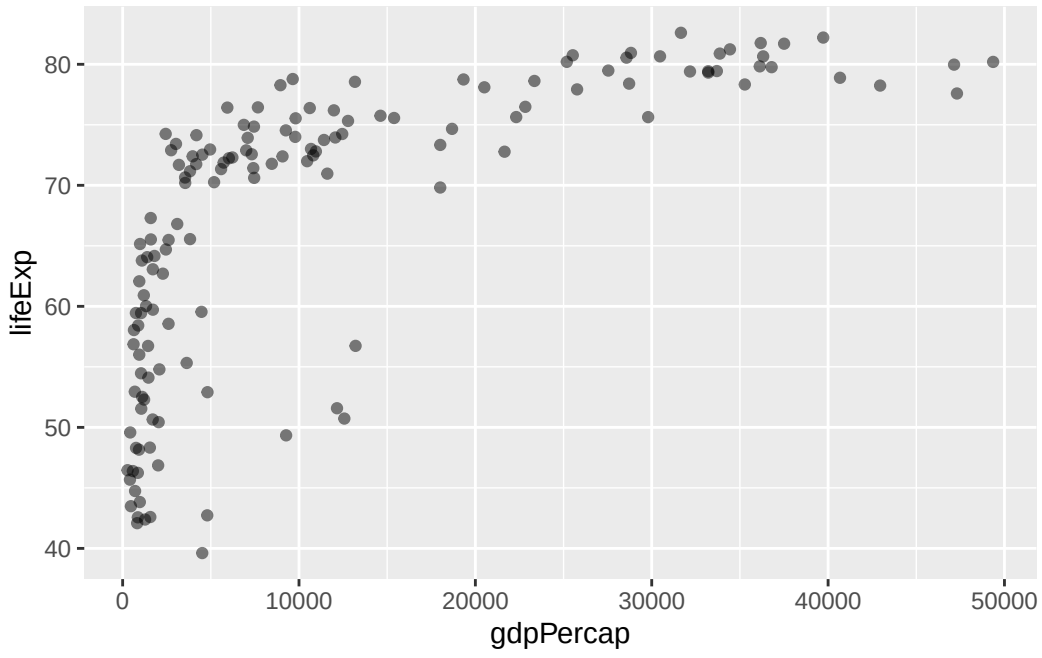
```
Warning: package 'dplyr' was built under R version 4.0.5
```

```
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
```

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

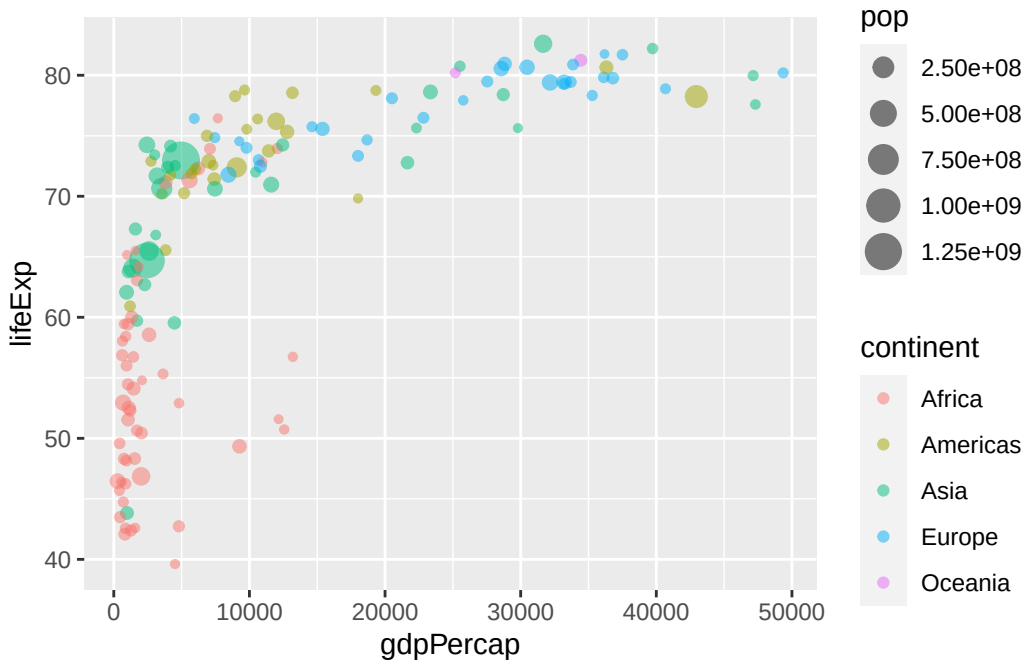
Q. Complete the code below to produce a first basic scatter plot of this gapminder_2007 dataset:

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



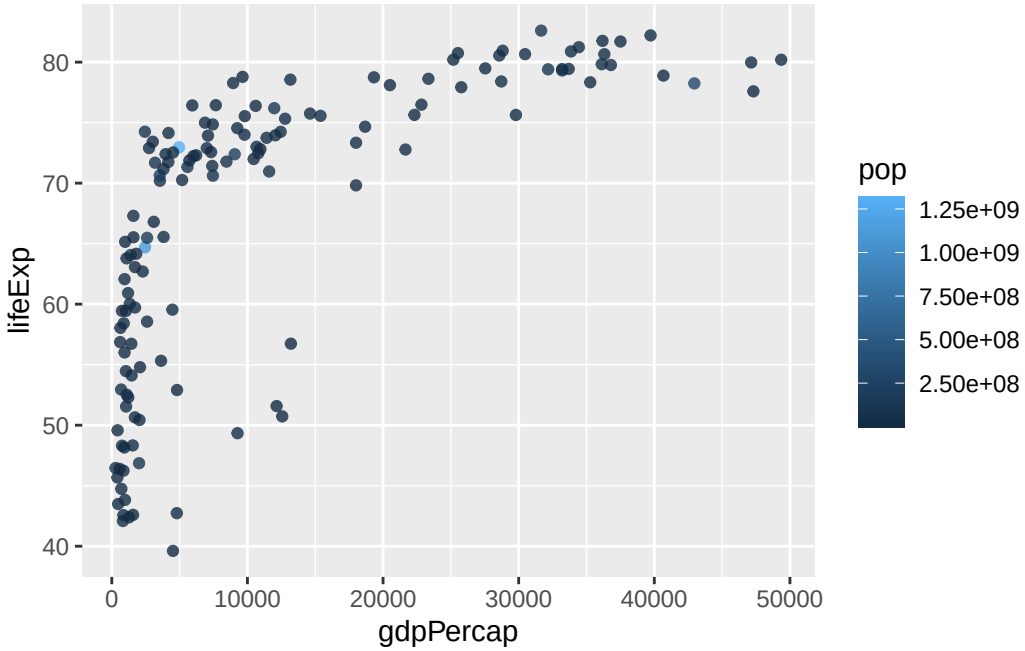
Adding more varables to aes() by adding color to the continent and size to the population

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



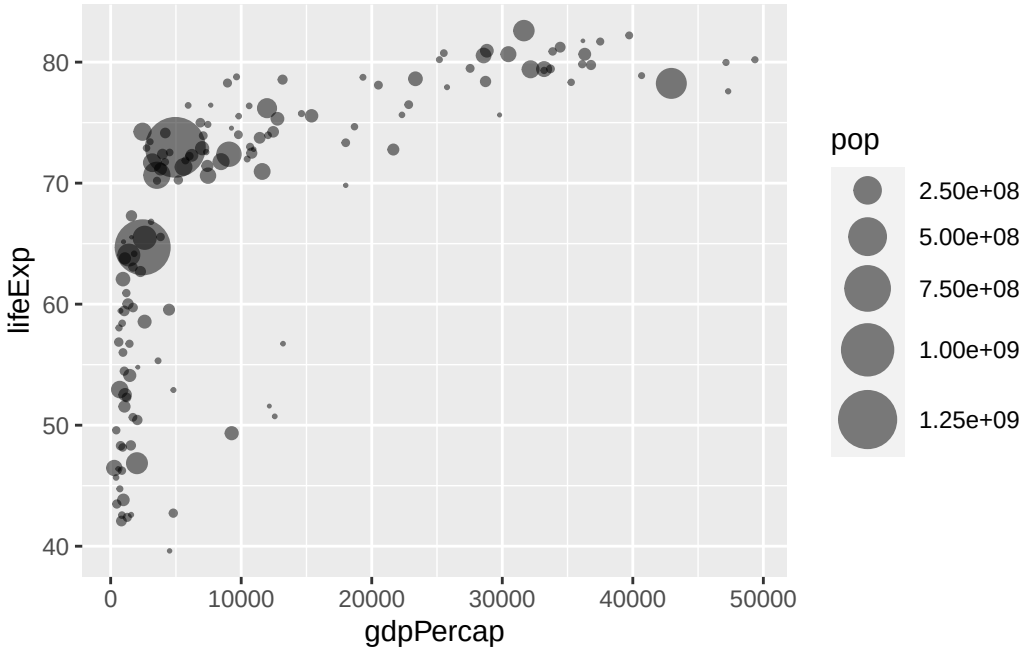
color points by the numeric variable population

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



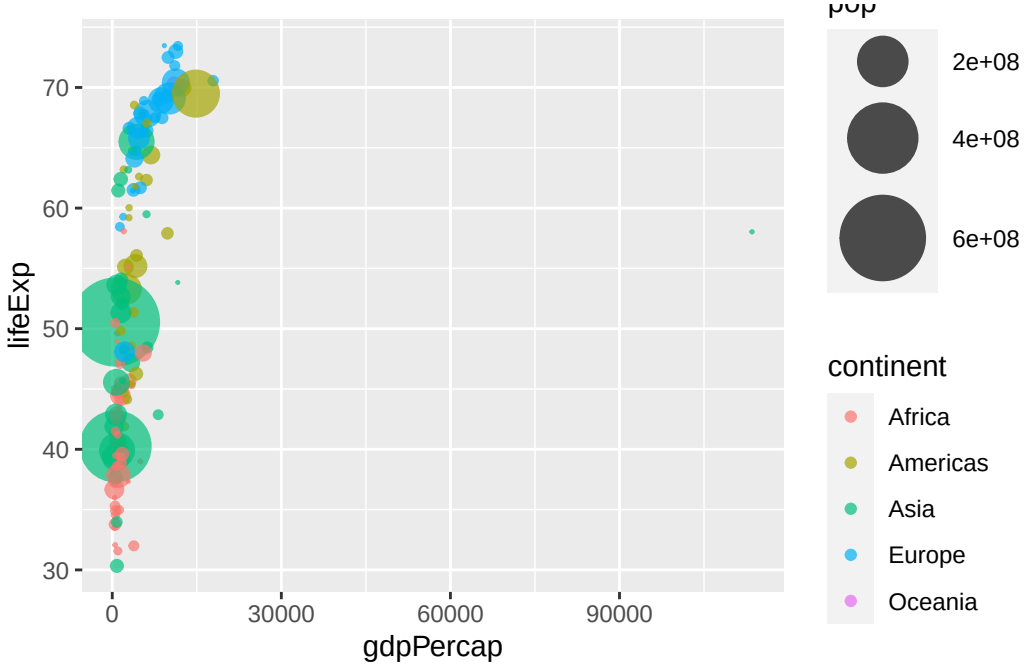
Adjust the point size of the data

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



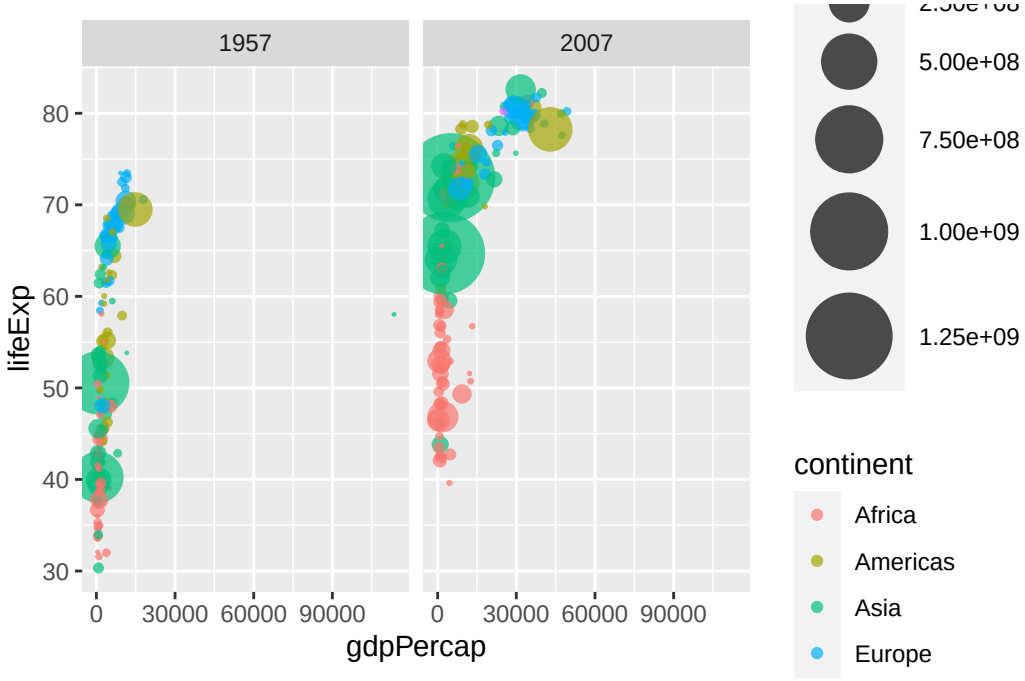
-Q. Can you adapt the code you have learned thus far to reproduce our gapminder scatter plot for the year 1957? What do you notice about this plot is it easy to compare with the one for 2007? Including the color for each continent is easier to interpret and visualize

```
gapminder_1957 <- gapminder %>% filter(year==1957)
ggplot(gapminder_1957)+
  aes(x=gdpPercap, y= lifeExp, color=continent, size= pop)+
  geom_point(alpha= 0.7)+
  scale_size_area(max_size =15)
```



Include the 1957 and 2007 plots together with facet_wrap function

```
gapminder_1957 <- gapminder %>% filter(year==1957| year==2007)
ggplot(gapminder_1957)+
  aes(x=gdpPercap, y= lifeExp, color=continent, size= pop)+
  geom_point(alpha= 0.7)+
  scale_size_area(max_size =15)+
  facet_wrap(~year)
```



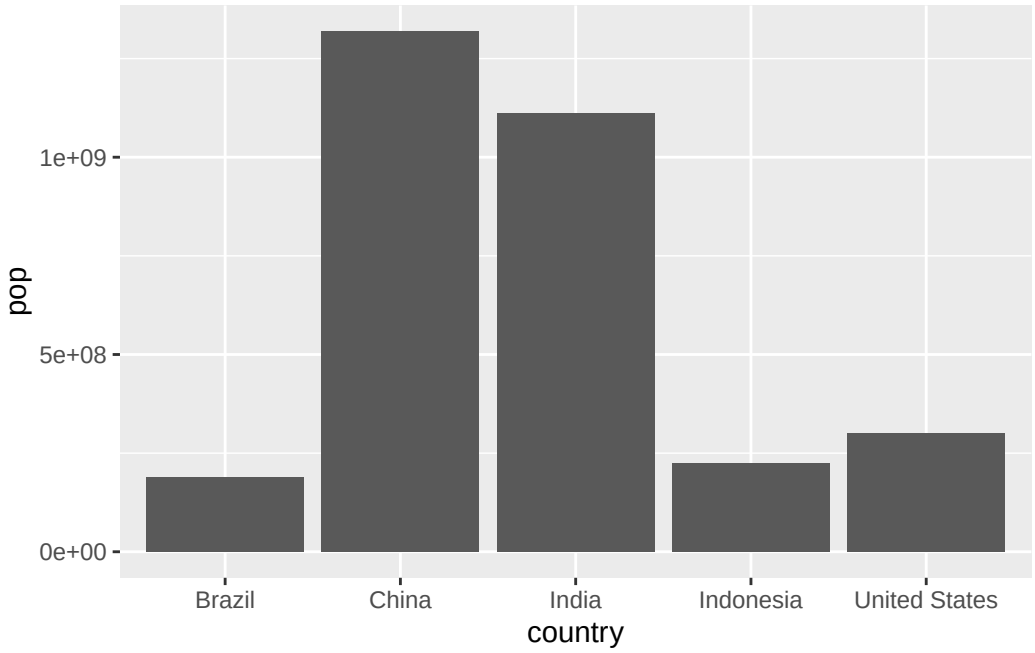
Bar Charts

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

gapminder_top5
```

```
# A tibble: 5 x 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.
```

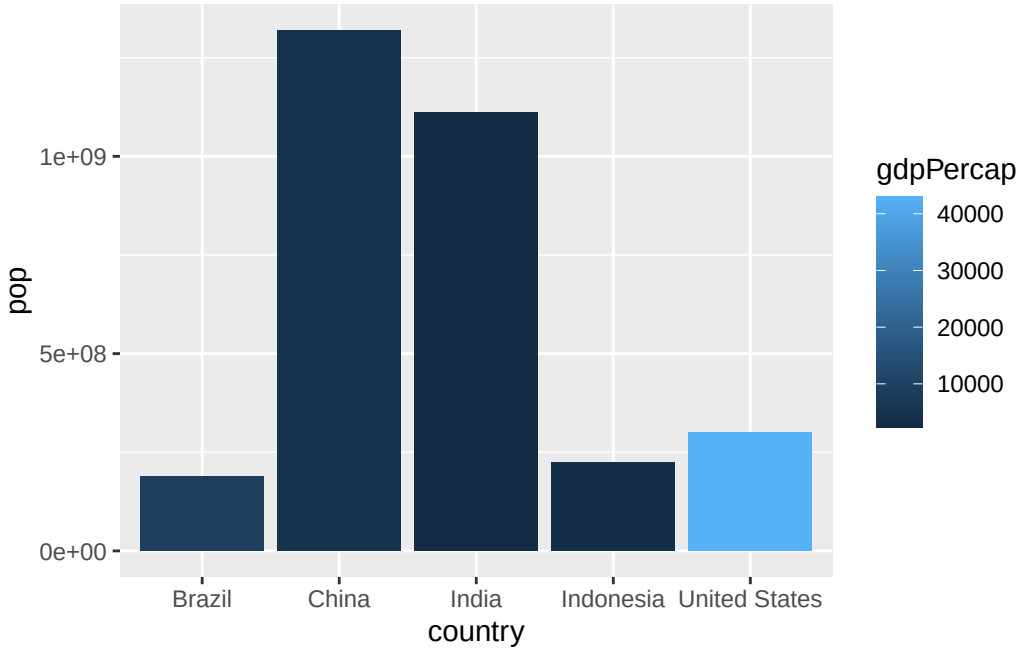
create a simple bar charts

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



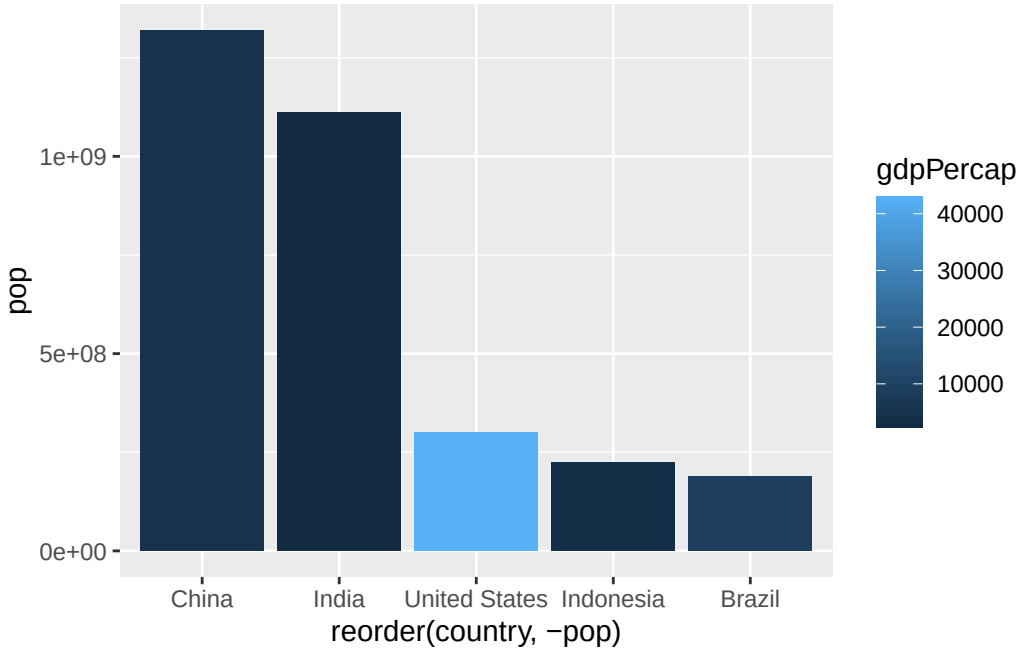
Plot the top five biggest countries by population in 2007 using Bar graph

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



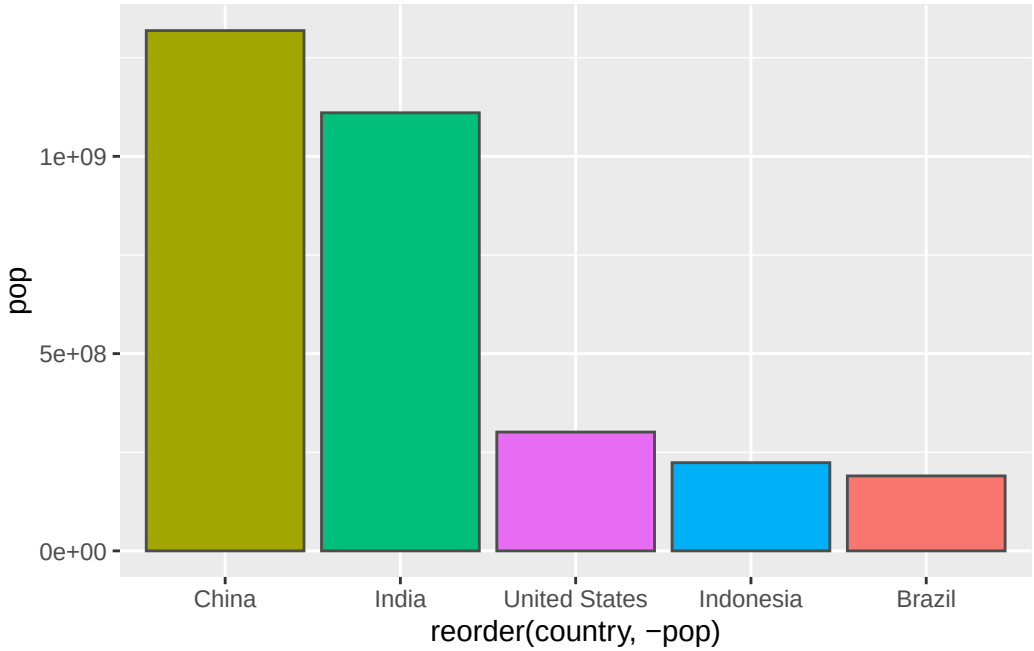
Change the order of the bars

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



and fill by country

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



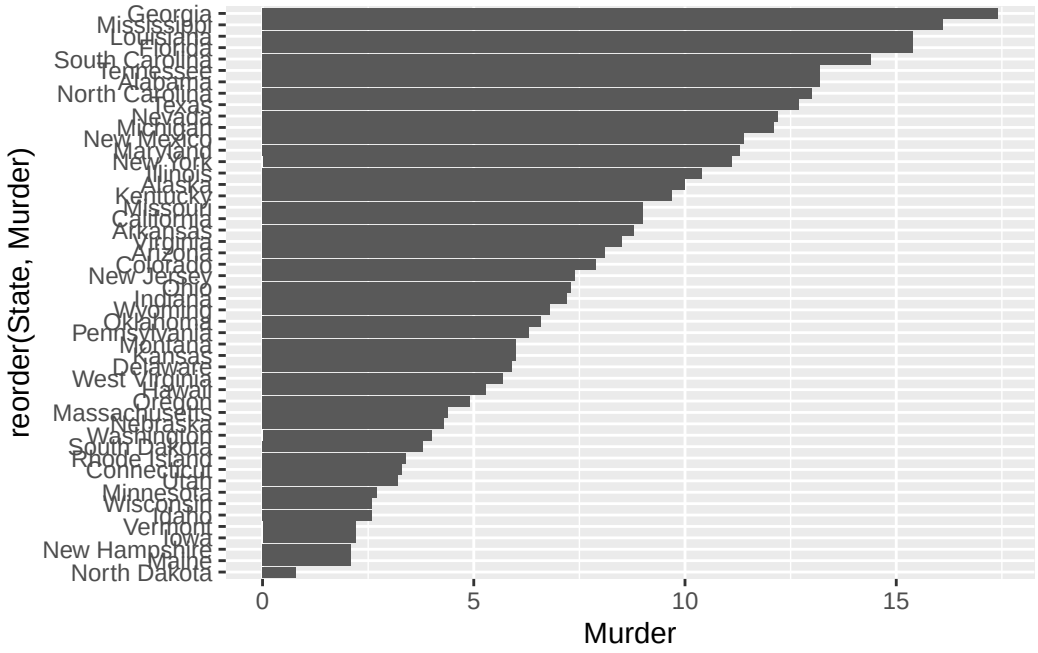
Flipping (rotate) bar chars with coord_flip() function on USArrests dataset

```
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
```

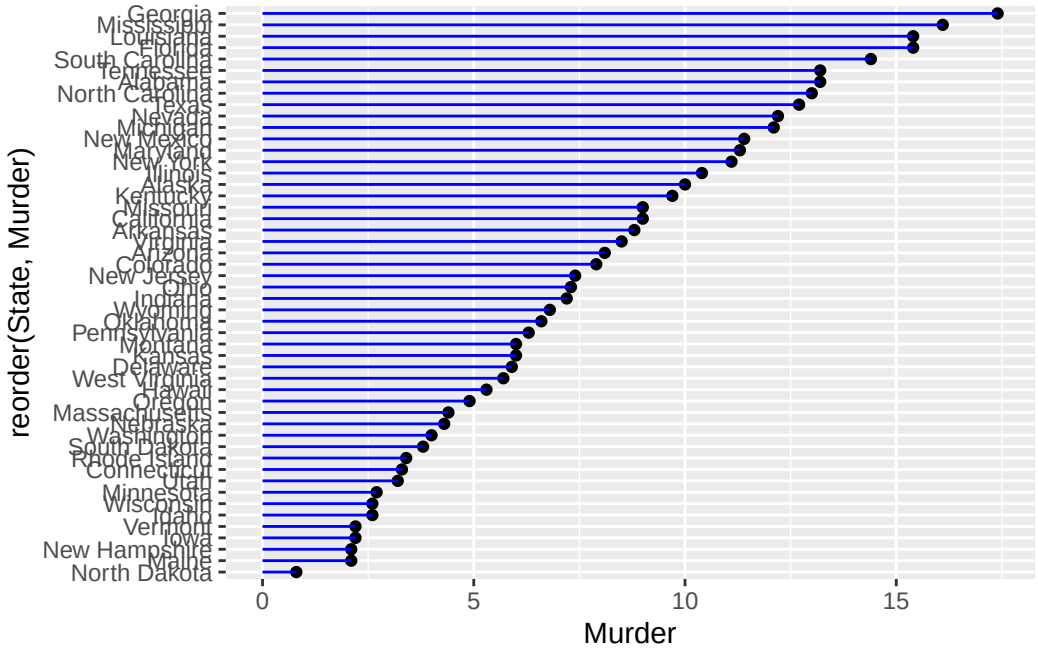
Use corrd_flip function

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



Using geom_point and geom_segment function to for better visualization

```
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