Class 05: Data Visualization with GGPLOT

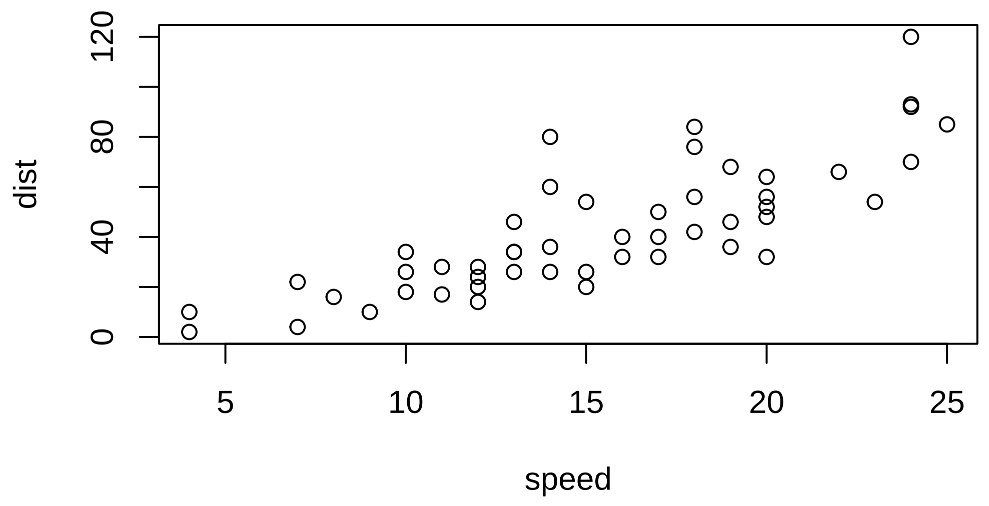
Hannah Kim

4/23/23

Base R Plotting

We are going to start by generating the plot of class 04. This code is plotting the cars dataset

```
plot(cars)
```



GGPLOT2

First, we need to install the package. We do this by using the install.packages command and then comment it because we only need to install once.

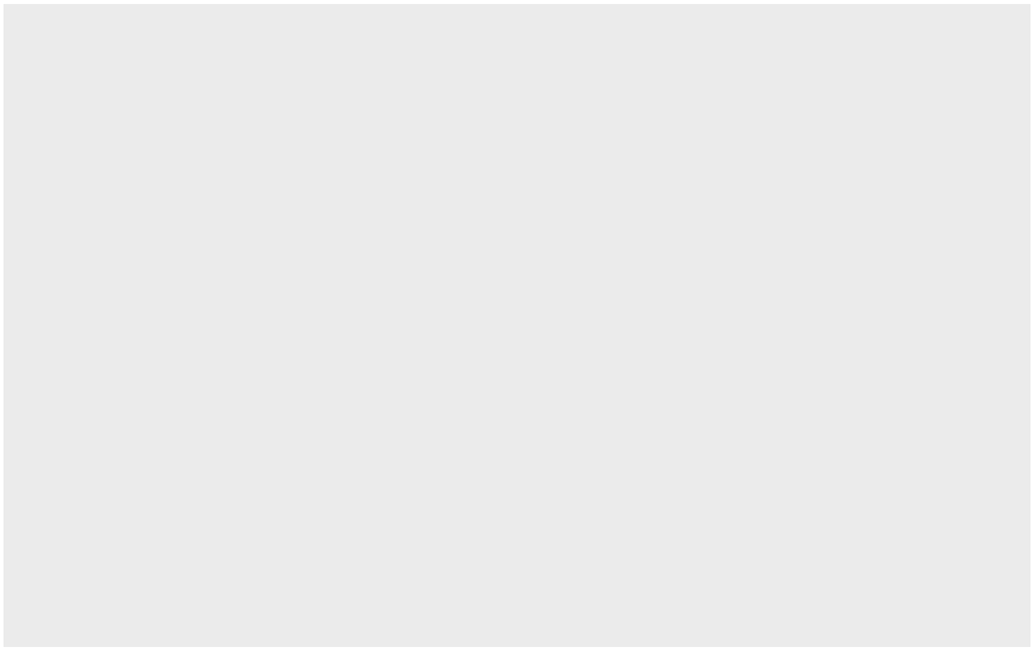
```
##install.packages('ggplot2')
```

After that, we need to load the package.

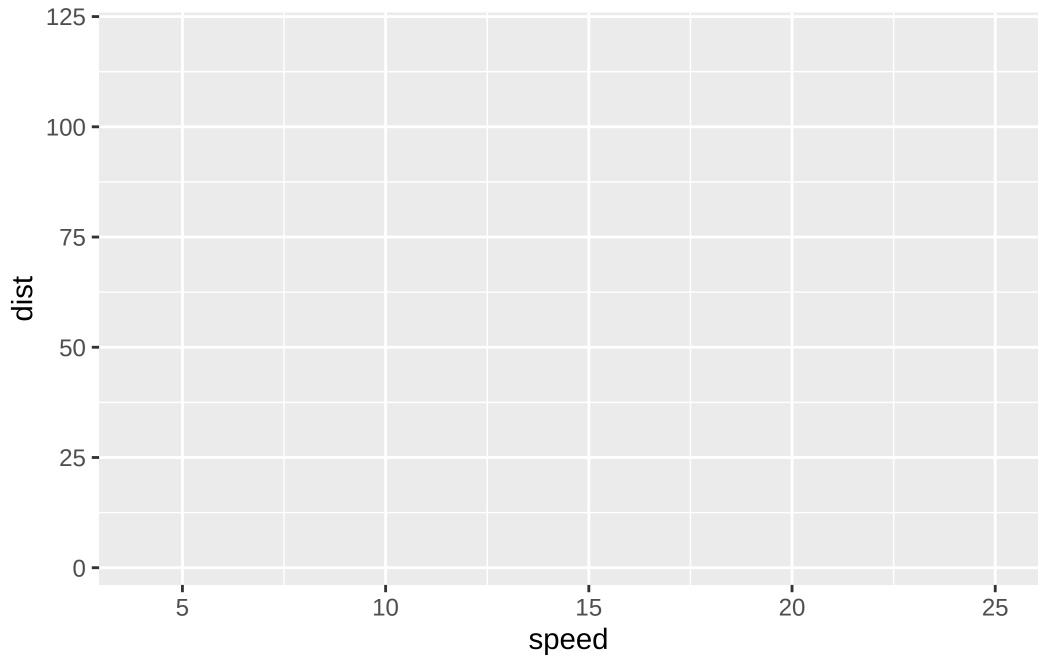
```
library(ggplot2)
```

We are going to build the plot of the cars dataframe by using ggplots

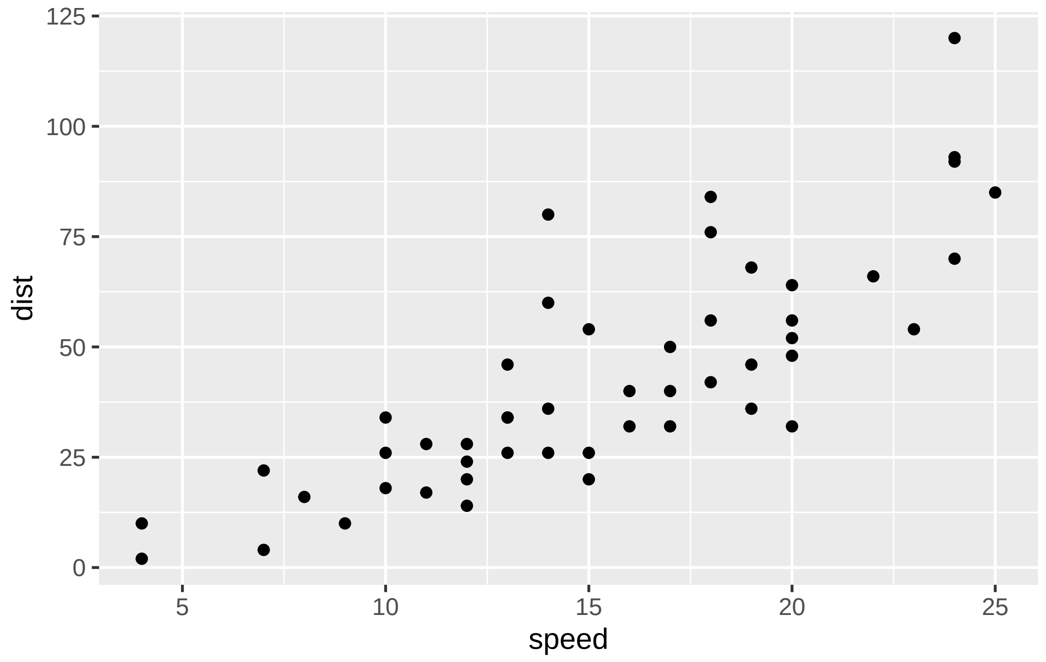
```
##Load data
ggplot(data=cars)
```



```
##Add axis
ggplot(data=cars) + aes(x=speed, y =dist)
```

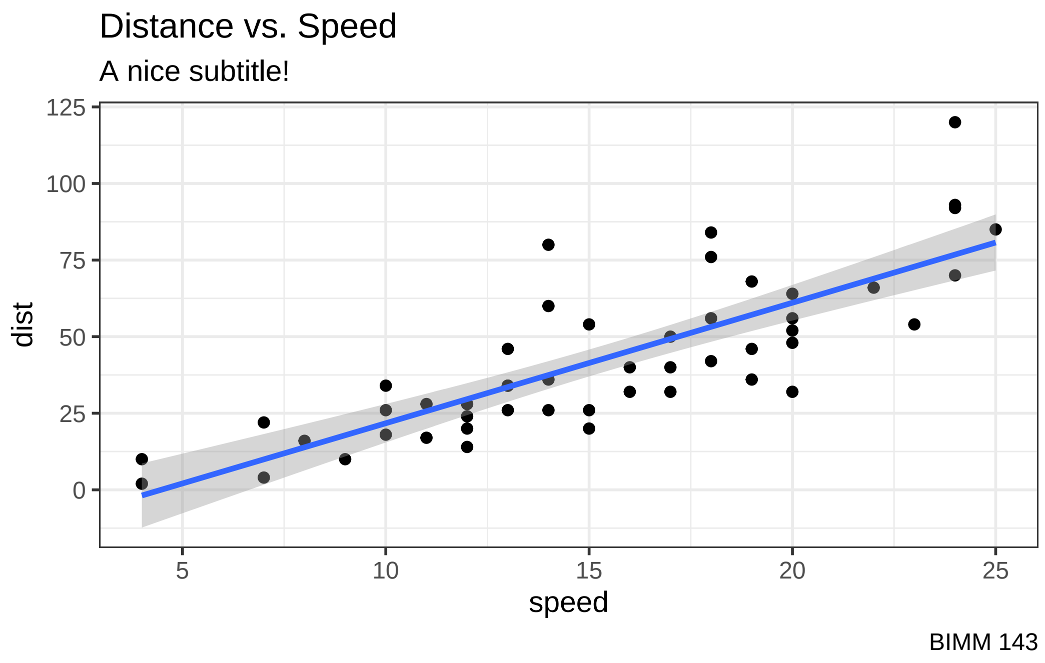


```
#Display points
ggplot(data=cars) + aes(x=speed, y =dist) + geom_point()
```



```
#Add a trend line
ggplot(data=cars) + aes(x=speed, y =dist) + geom_point() +geom_smooth(method = 'lm') + lab
```

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



Q1. For which phases is data visualization important in our scientific workflows?

Communication of Results, Exploratory Data Analysis (EDA), and Detection of outliers

Q2. True or False? The ggplot2 package comes already installed with R?

FALSE

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

Network graphs

Q. Which statement about data visualization with ggplot2 is incorrect?

ggplot2 is the only way to create plots in R

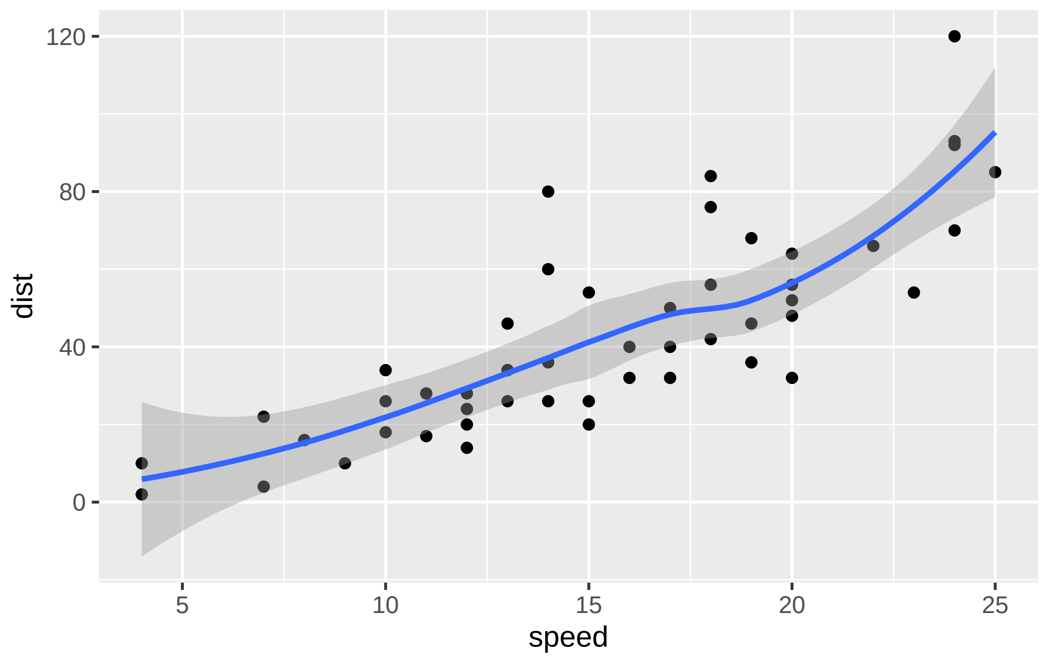
Q. Which geometric layer should be used to create scatter plots in ggplot2?

geom_point()

Q. In your own RStudio can you add a trend line layer to help show the relationship between the plot variables with the `geom_smooth()` function?

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

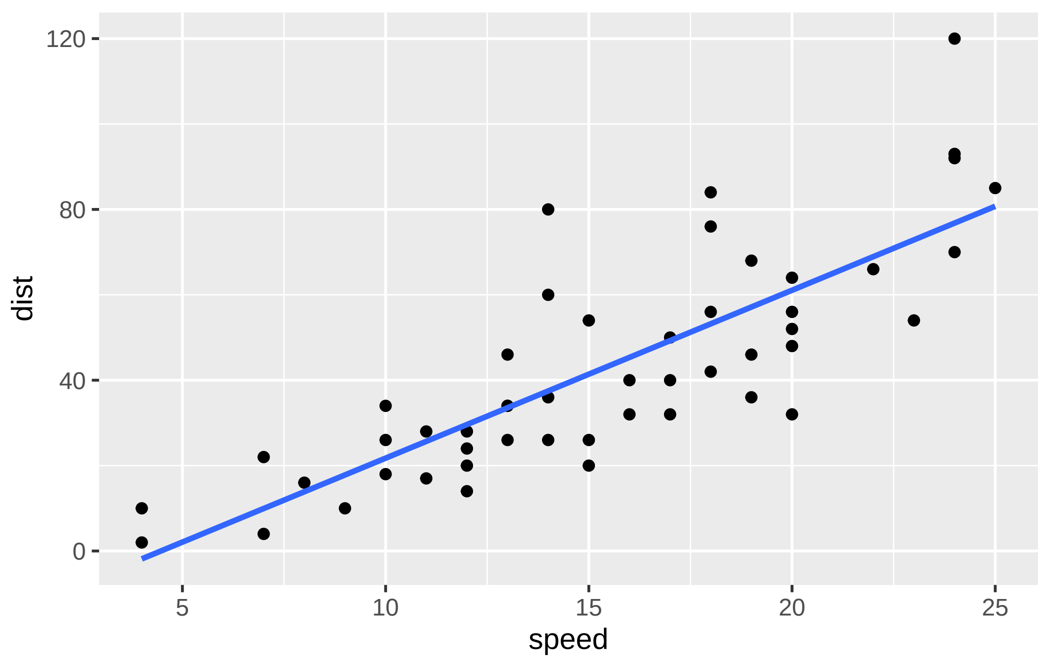
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Q. Argue with `geom_smooth()` to add a straight line from a linear model without the shaded standard error region?

```r
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  geom_smooth(method="lm", se=FALSE)
```

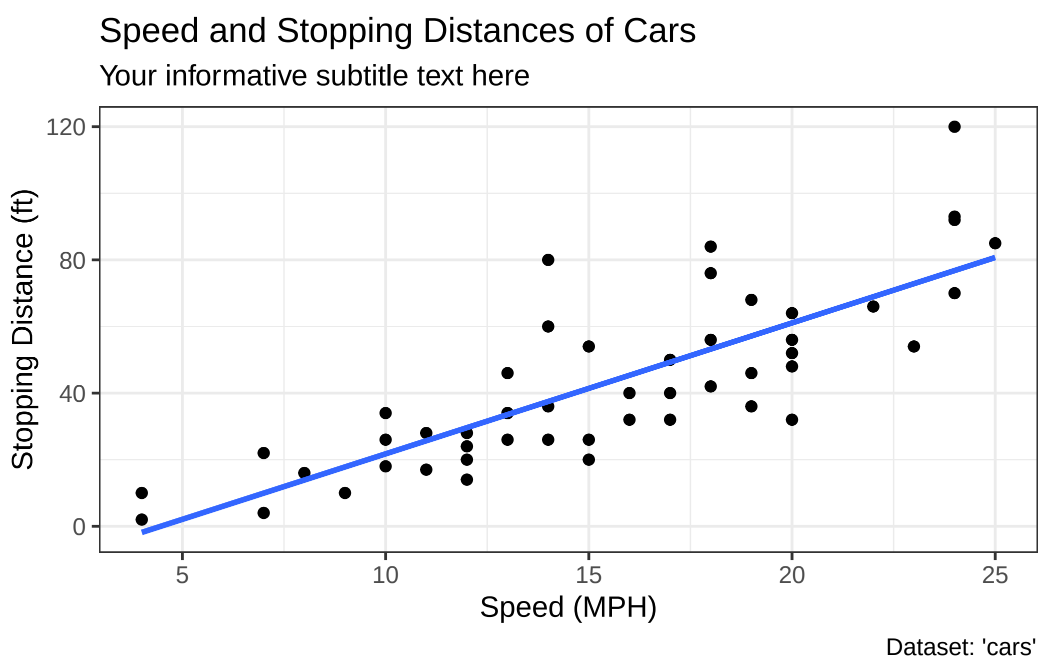
```
`geom_smooth()` using formula = 'y ~ x'
```



Q. Can you finish this plot by adding various label annotations with the `labs()` function and changing the plot look to a more conservative "black & white" theme by adding the `theme_bw()` function:

```r
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  labs(title="Speed and Stopping Distances of Cars",
       x="Speed (MPH)",
       y="Stopping Distance (ft)",
       subtitle = "Your informative subtitle text here",
       caption="Dataset: 'cars'") +
  geom_smooth(method="lm", se=FALSE) +
  theme_bw()
```

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



Plotting Gene Expression Data

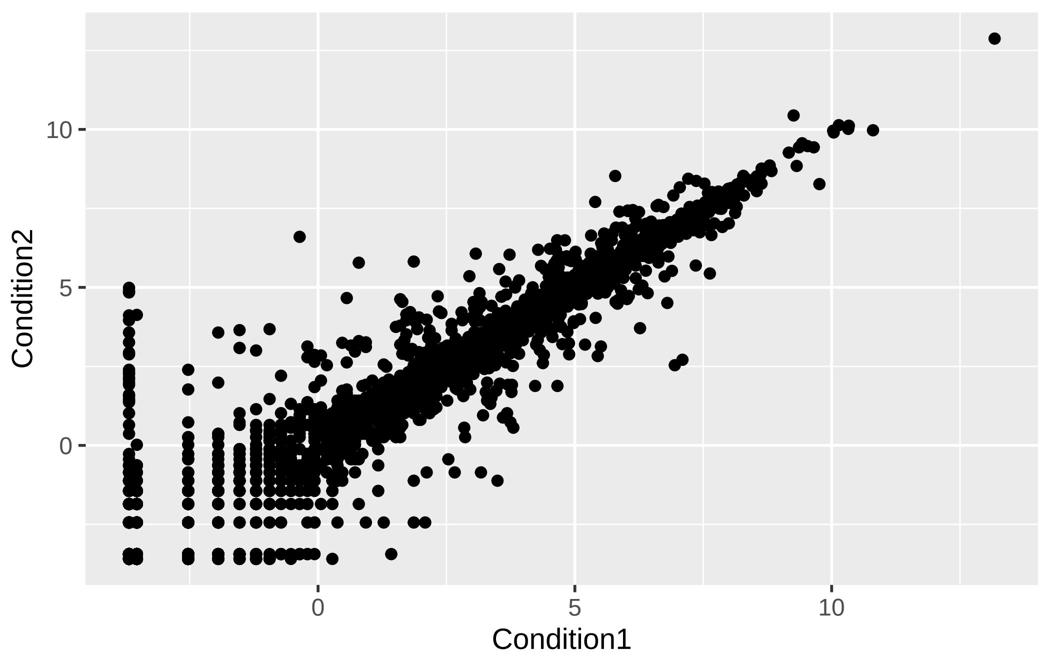
Loading the Data from the URL.

```
##Load data
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
```

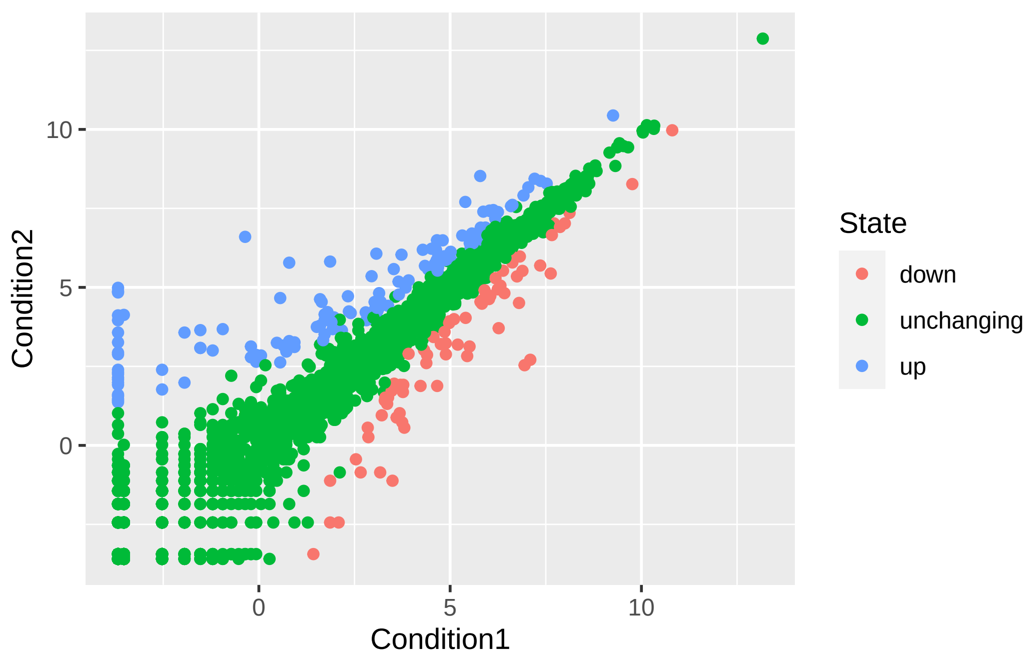
Initial ggplot

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

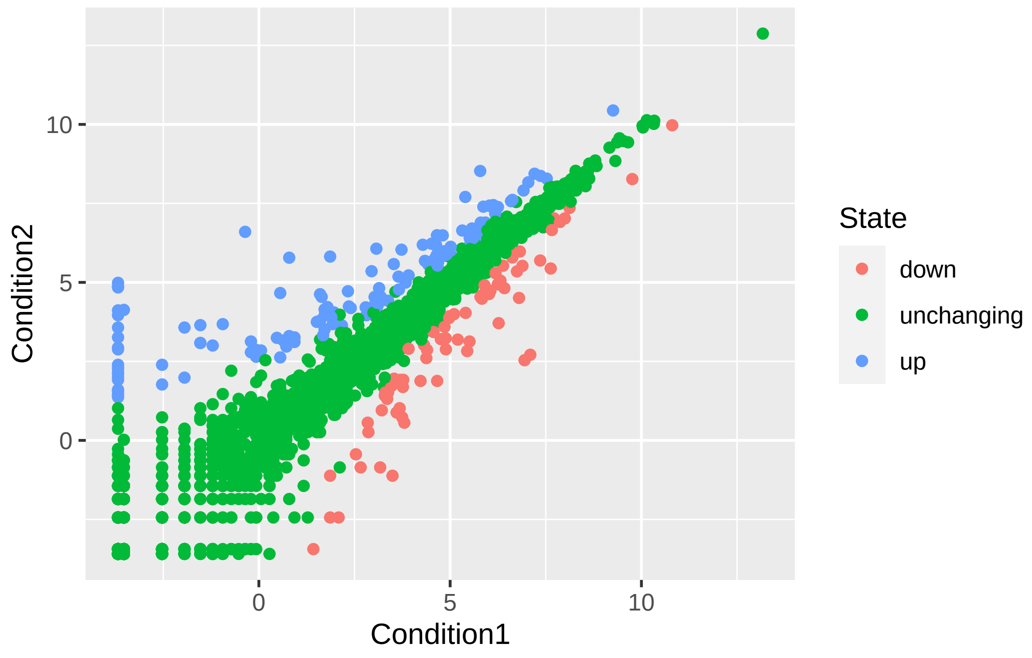


Adding color to the plot

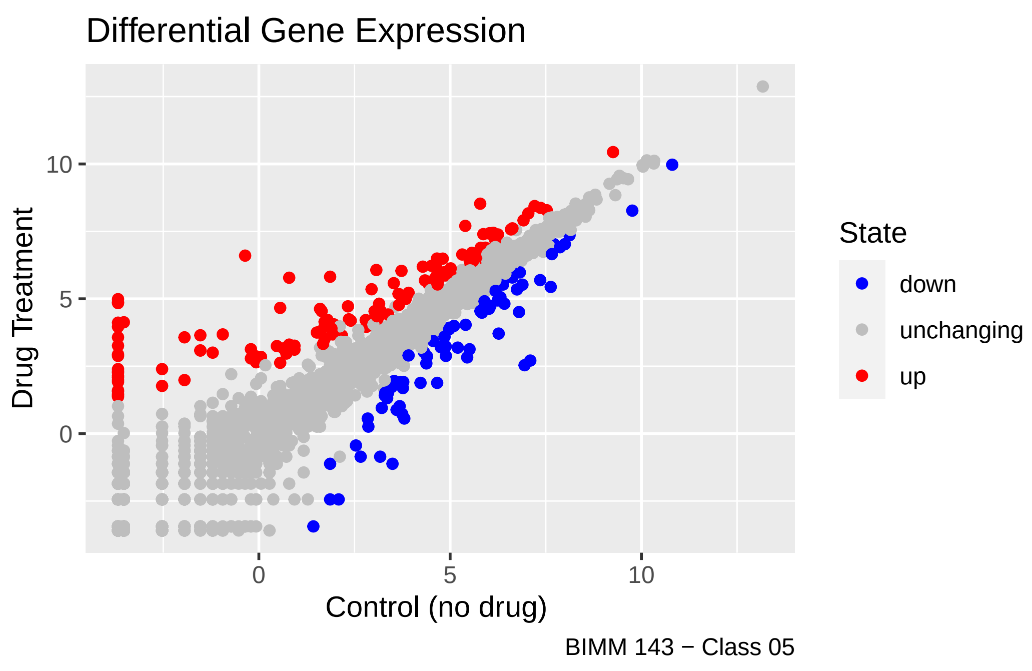
```
ggplot(data = genes) + aes (x = Condition1, y = Condition2, col = State) + geom_point()
```



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



```
p + scale_colour_manual( values=c("blue","gray","red") ) +
    labs(title="Differential Gene Expression",
         x="Control (no drug) ",
         y="Drug Treatment", caption = 'BIMM 143 - Class 05')
```



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

5196

```
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?

4 Columns: Gene, Condition1, Condition2, State

```
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?

2.44

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

```

      down unchanging         up 
      1.39      96.17       2.44 
```

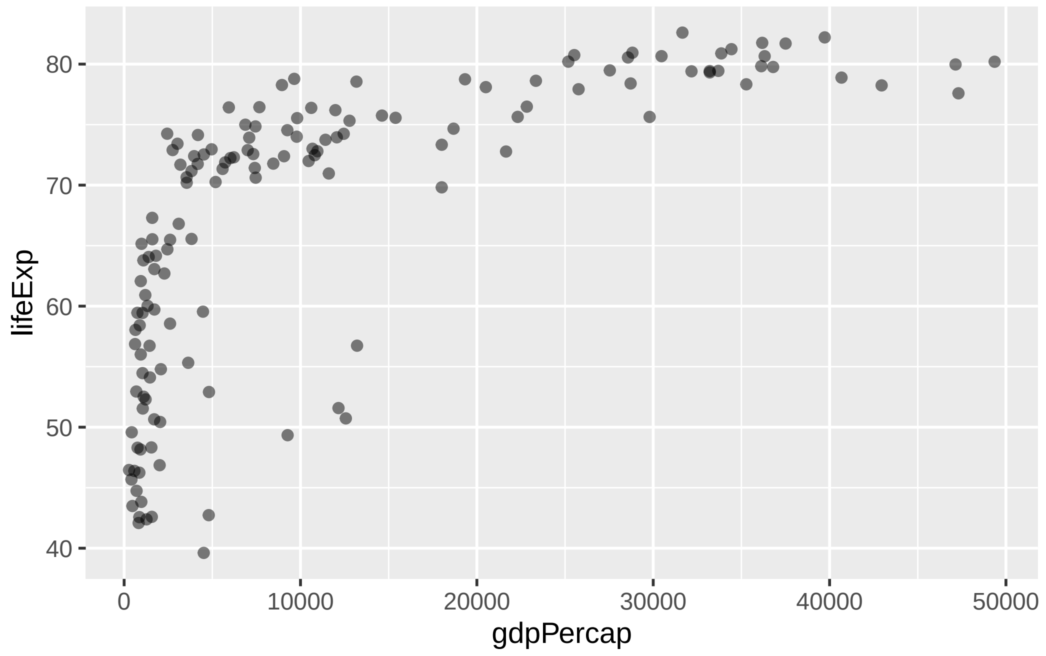
```
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.

gapminder <- read.delim(url)
#install.packages("dplyr")
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)
```

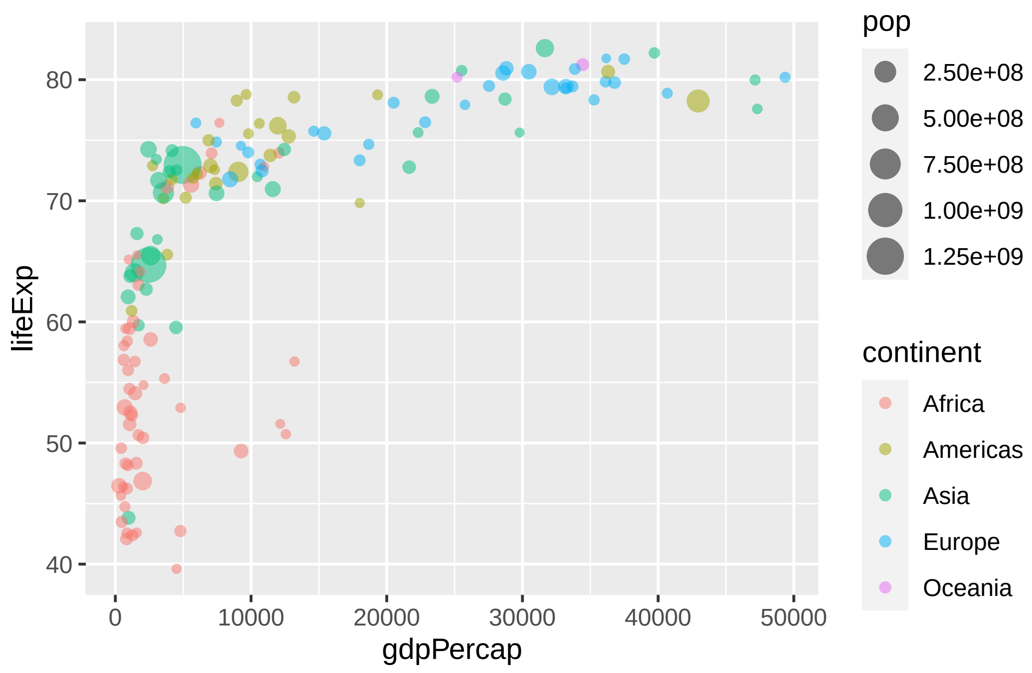
Q. Complete the code below to produce a first basic scater plot of this `gapminder_2007` dataset:

```r
#install.packages("ggplot2")
library(ggplot2)

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



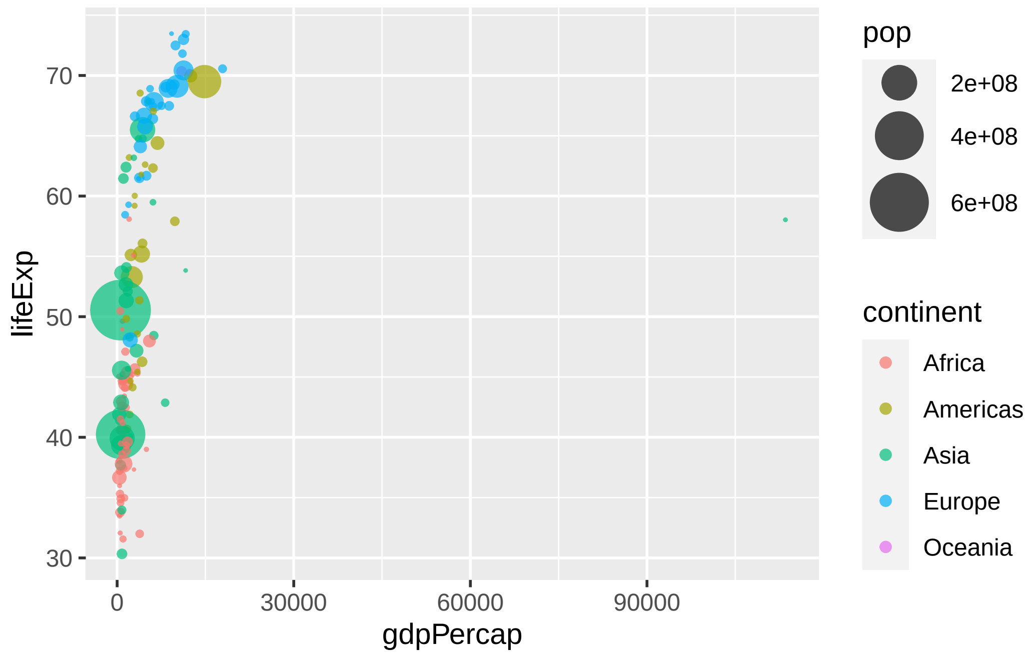
We added color based on continent using ggplot2

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



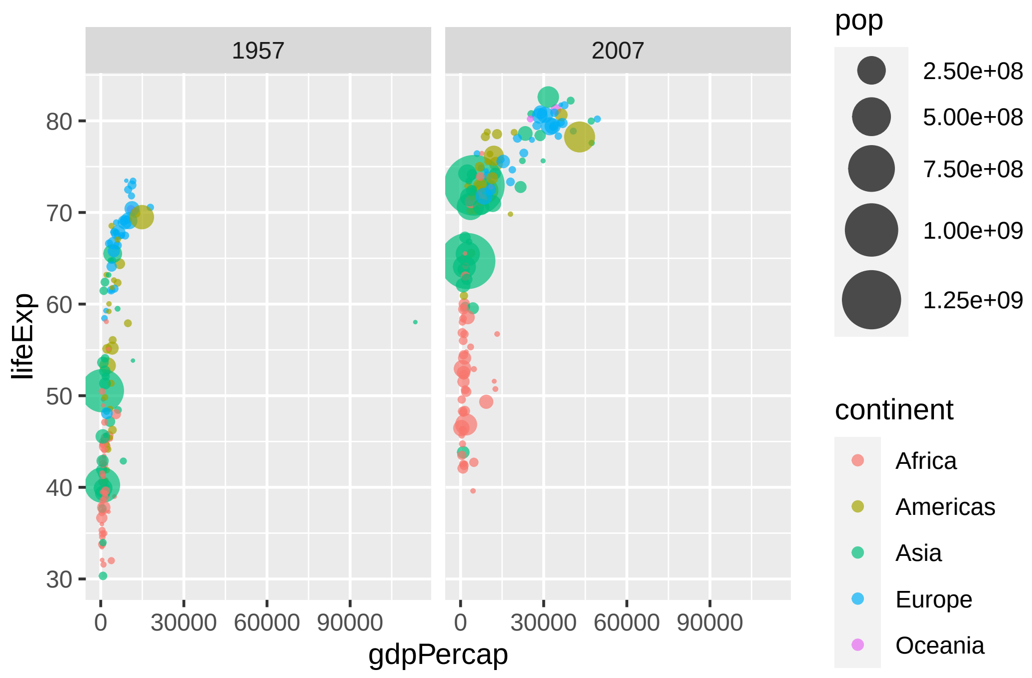
Q. Can you adapt the code you have learned thus far to reproduce our gapminder scatter plot for the year 1957? What do you notice about this plot is it easy to compare with the one for 2007?

```
#filter for only year 1957
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 = 10)
```



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



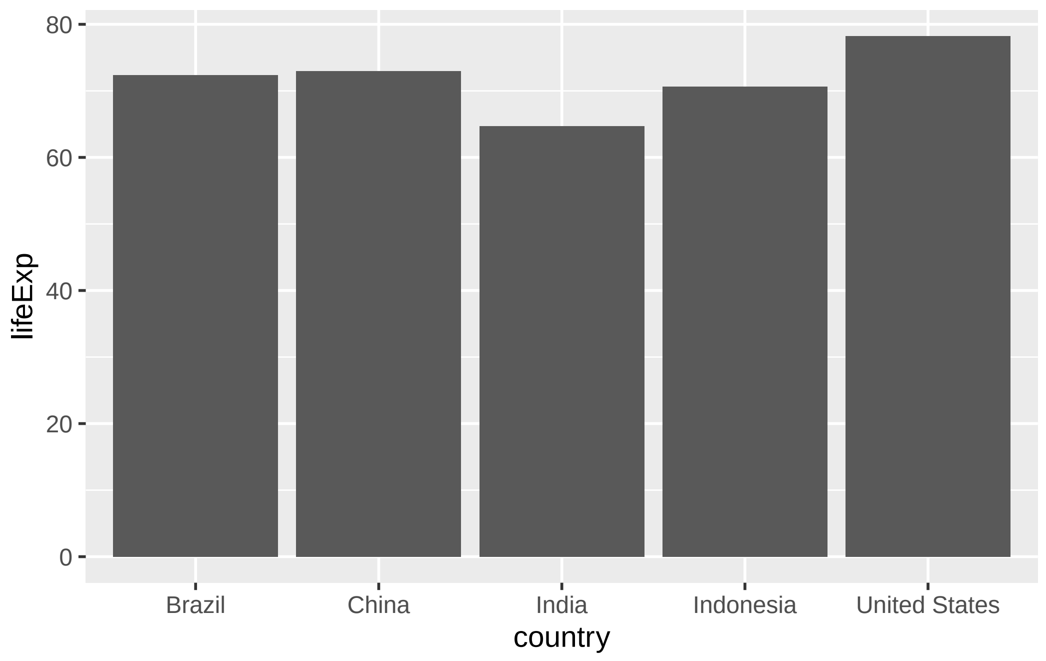
Bar Plots

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

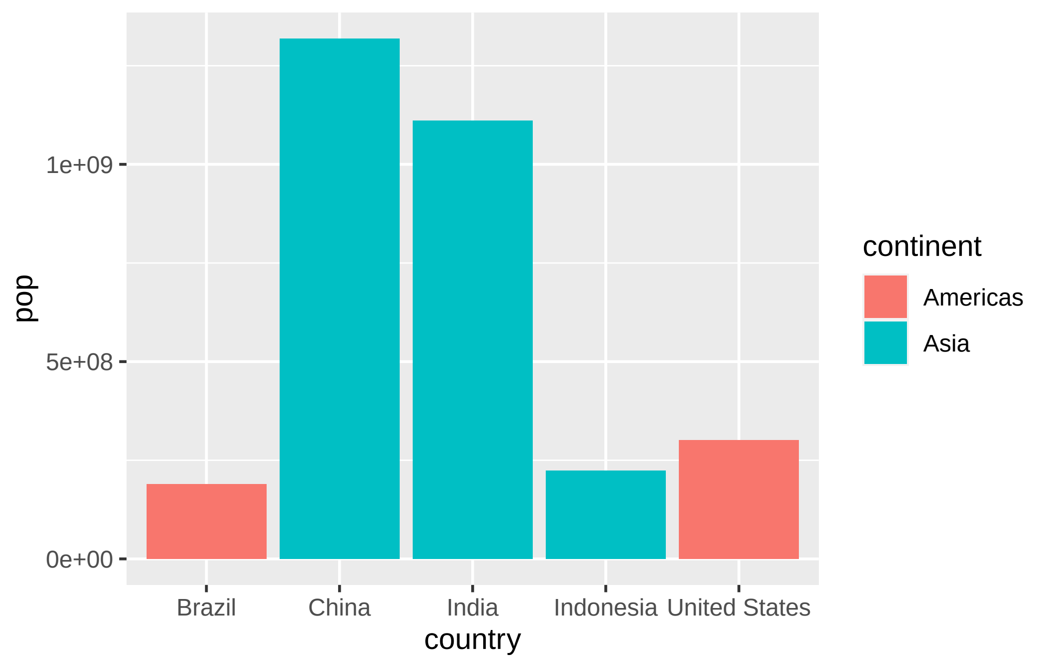
```
gapminder_top5 <- gapminder %>%
  filter(year==2007) %>%
  arrange(desc(pop)) %>%
  top_n(5, pop)

#Creates bar plot of top 5 countries with highest life expectance or lifeExp.
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))
```

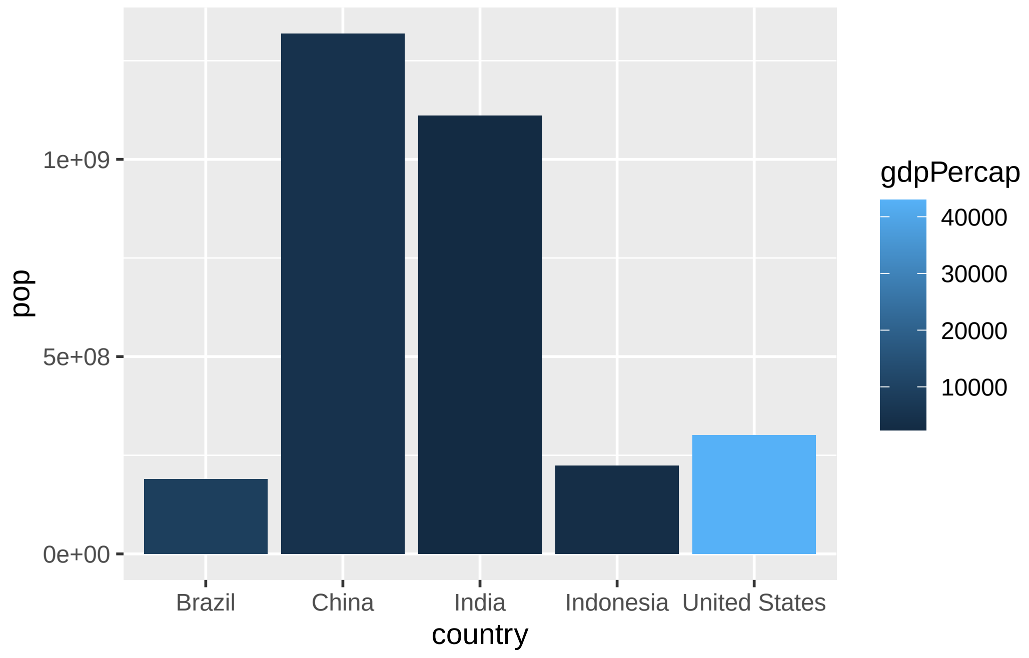


```
#Adding color
ggplot(gapminder_top5) +
  geom_col(aes(x = country, y = pop, fill = continent))
```



Q. 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) +
  aes(x=country, y=pop, fill=gdpPercap) +
  geom_col()
```



gdpPercap
40000
30000
20000
10000
1e+09
5e+08
0e+00
pop
Brazil
China
India
Indonesia
United States
country