Class 5 Data Visualization with ggplot2

Alvin Cheng (PID A16840171)

Using GGPLOT

The ggplot2 package needs to be installed as it does not come with R "out of the box."

We use the `install.packages()` function to do this.

```
head(cars)
```

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

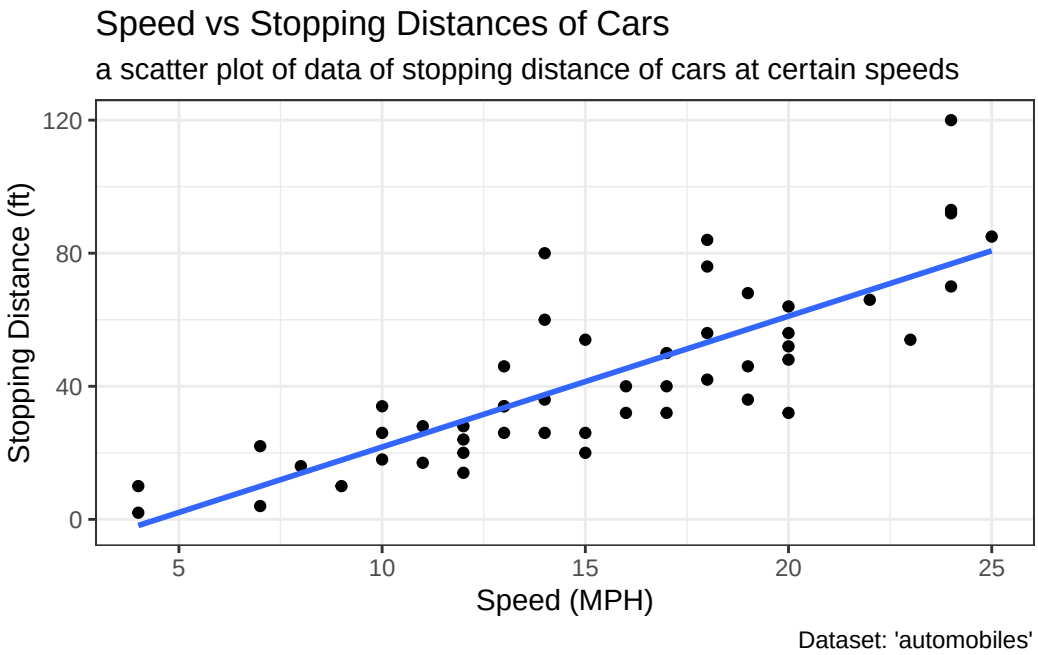
to use ggplot I need to load it up before I can call any of the functions in the package. I do this with the `library()` function.

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

all ggplot figures have at least 3 things: - data (the stuff we want to plot) - aesthetic mapping (aes values) - geoms

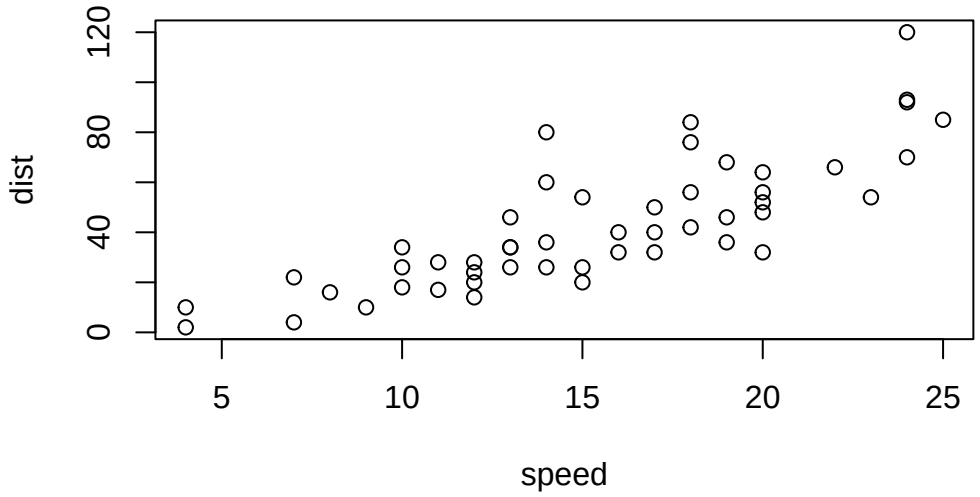
```
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  labs (title = "Speed vs Stopping Distances of Cars", x ="Speed (MPH)", y = "Stopping Dis
theme_bw()
```

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



ggplot is not the only graphing system in R. There are lots of others. There is even "base R" graphics.

```
plot(cars)
```



Gene Expression 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
```

```
genes_in_set <- nrow(genes)
genes_in_set
```

```
[1] 5196
```

```r
colnames(genes)
```

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

```r
ncol(genes)
```

```
[1] 4
```

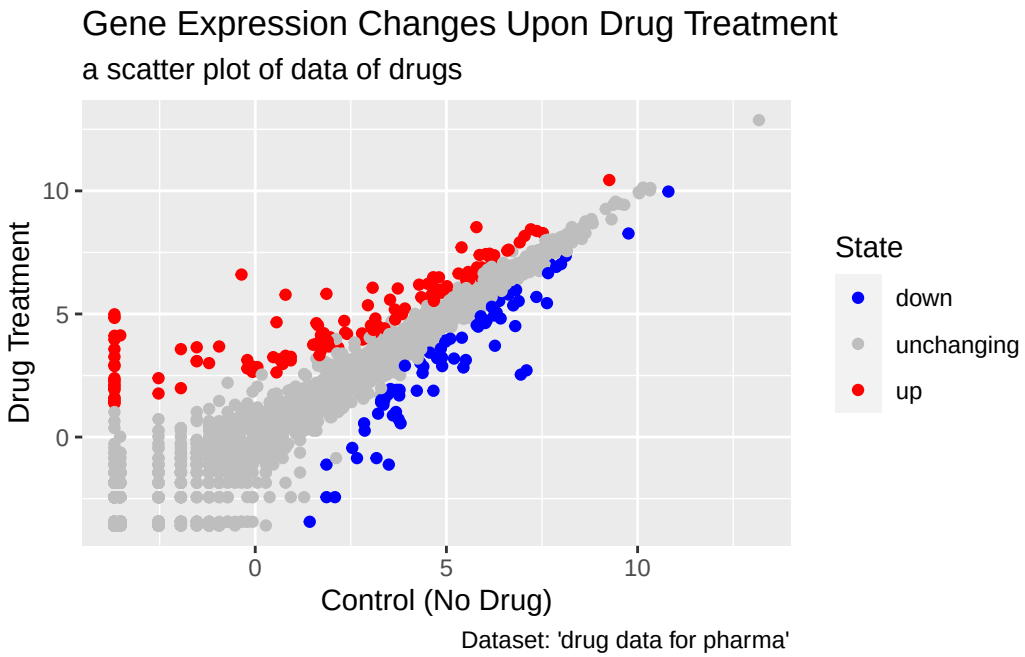
```r
table(genes[,"State"])
```

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

```r
round(table(genes[,"State"])/nrow(genes) * 100, 2)
```

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

```r
#From this dataset, we are going to make a scatter plot
p = ggplot(genes) +
  aes(x=Condition1, y=Condition2, col=State) +
  geom_point() +
    labs(title = "Gene Expression Changes Upon Drug Treatment",
         x ="Control (No Drug)",
         y = "Drug Treatment",
         subtitle = "a scatter plot of data of drugs",
         caption = "Dataset: 'drug data for pharma'")
#+  geom_smooth(method="lm", se=FALSE)

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

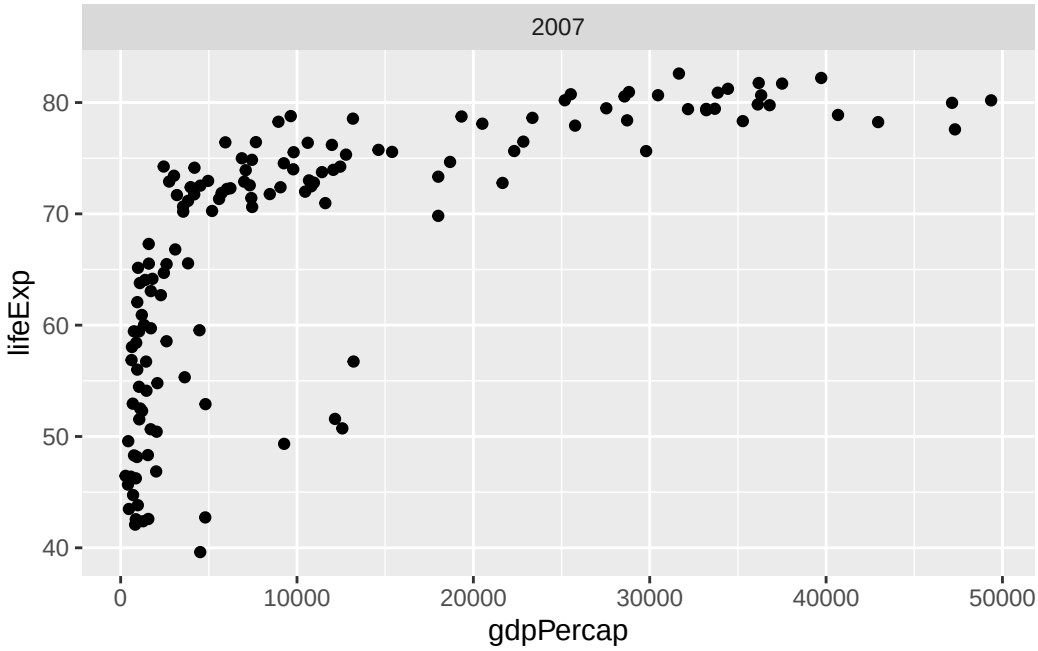


Going Further

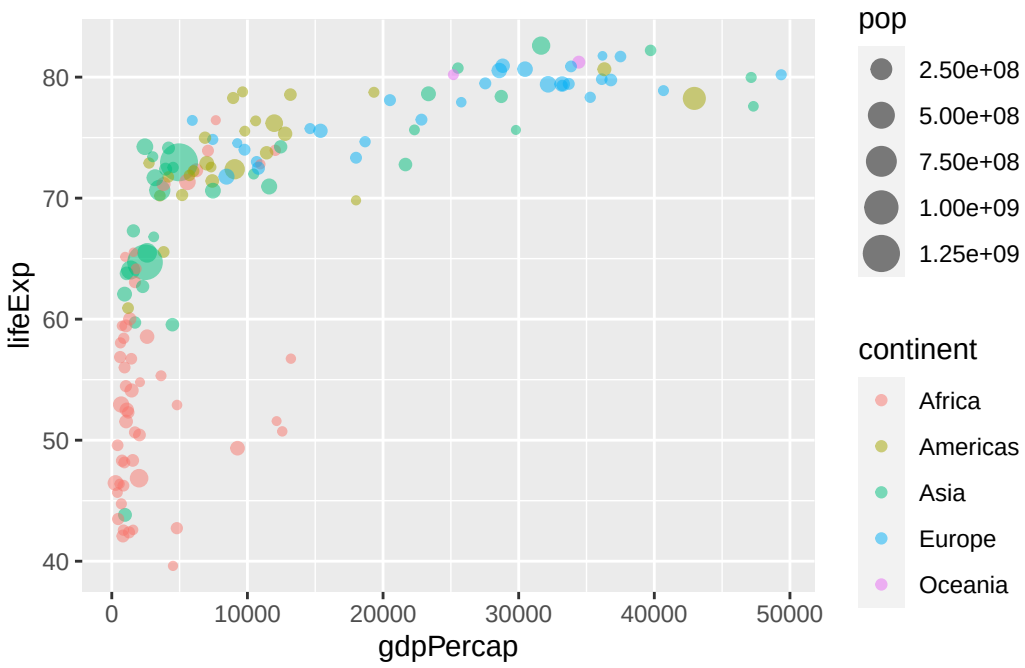
```
# File location online
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.
gapminder <- read.delim(url)
library(dplyr)
```

```
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
```

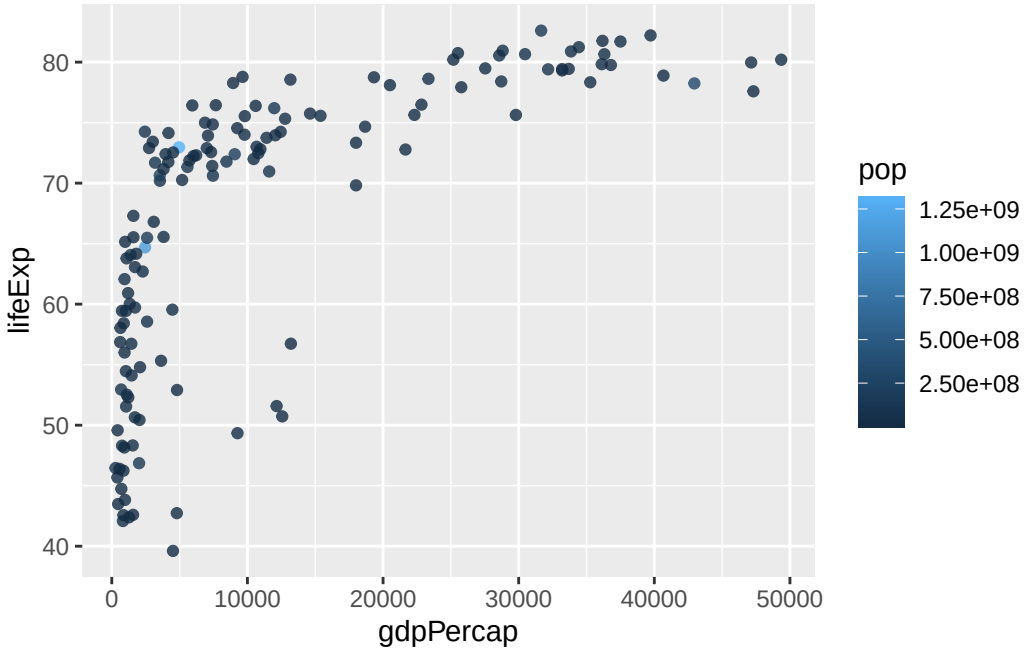
```
gapminder_2007 <- gapminder %>% filter(year==2007)
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point() + facet_wrap(~year)
```



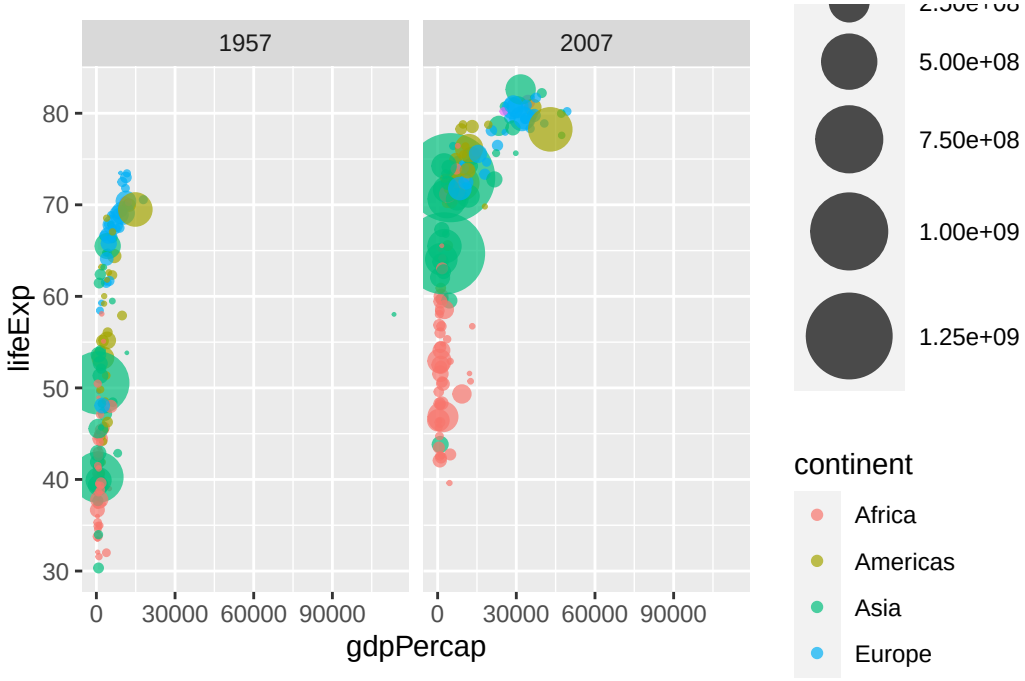

```
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.5)
```




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




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

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

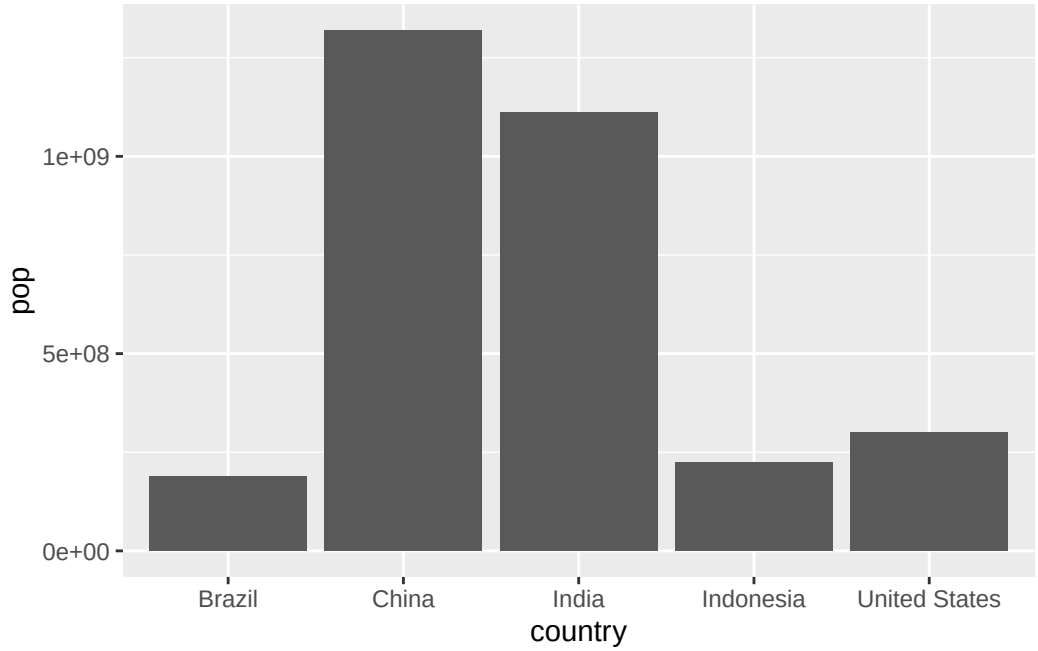


Bar Charts

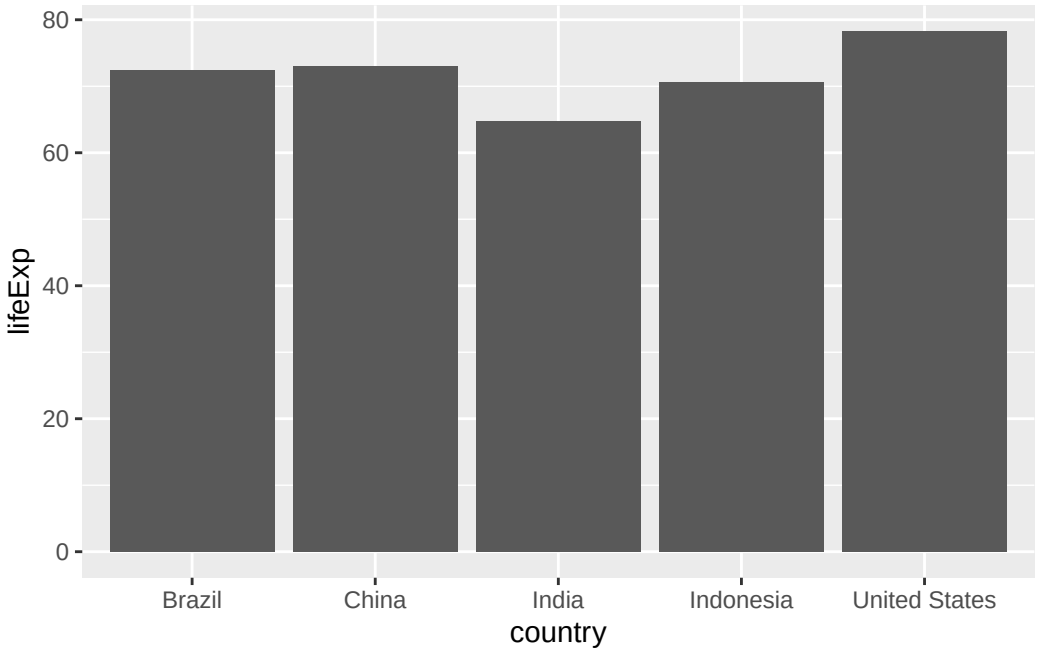
```
gapminder_top5 <- gapminder %>%
  filter(year==2007) %>%
  arrange(desc(pop)) %>%
  top_n(5, pop)

gapminder_top5
```

```
        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
```

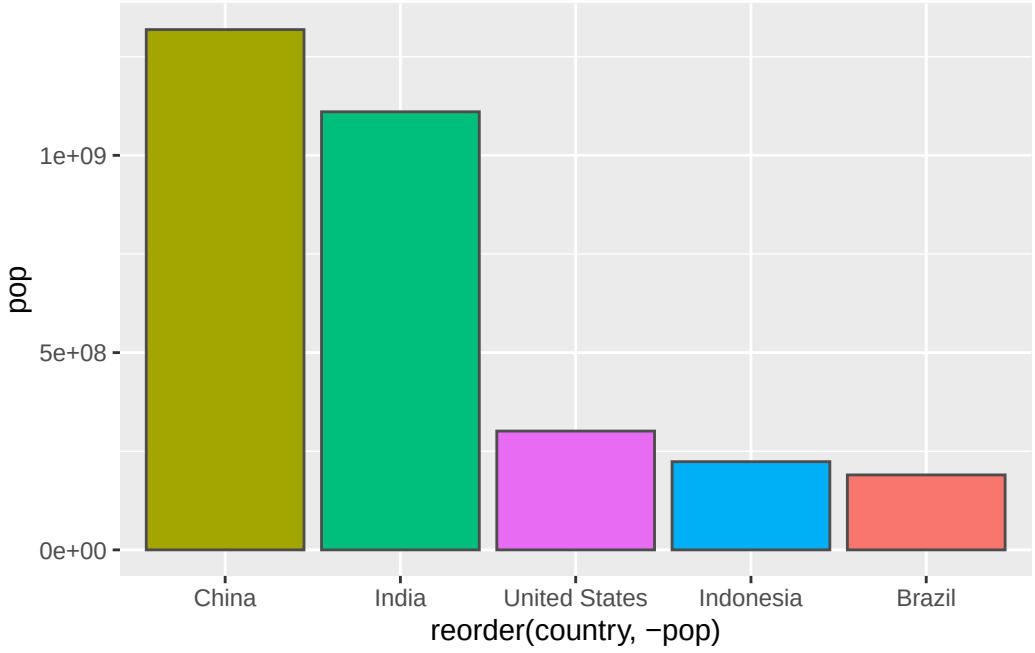
```
ggplot(gapminder_top5) +
  geom_col(aes(x = country, y = pop))
```




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



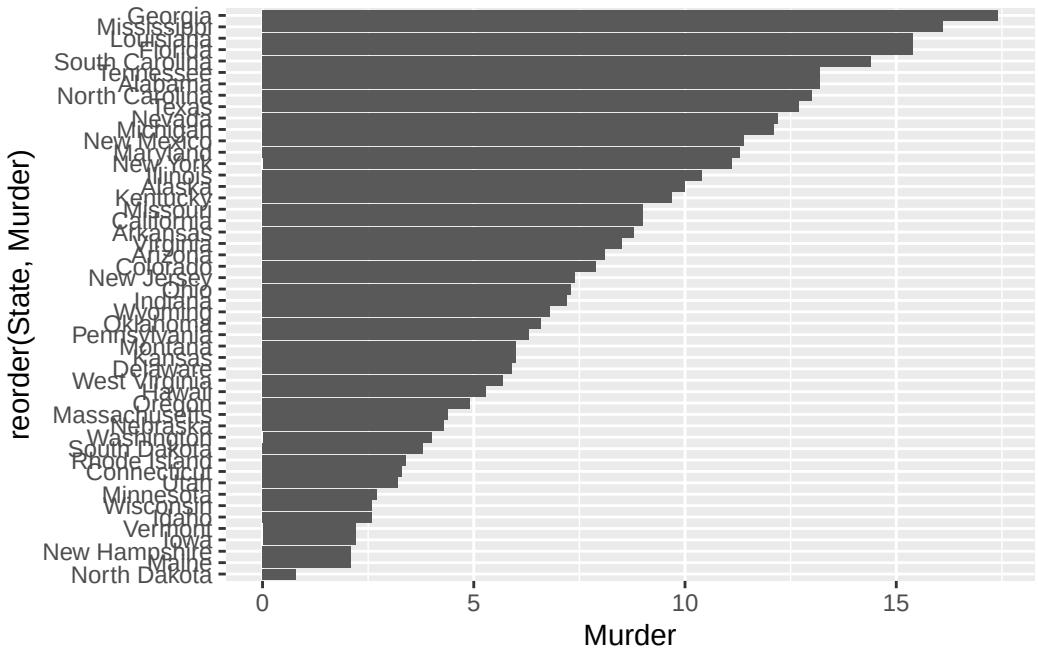
```
ggplot(gapminder_top5) +
  aes(x=reorder(country, -pop), y=pop, fill=country) +
  geom_col(col="gray30") +
  guides(fill="none")
```



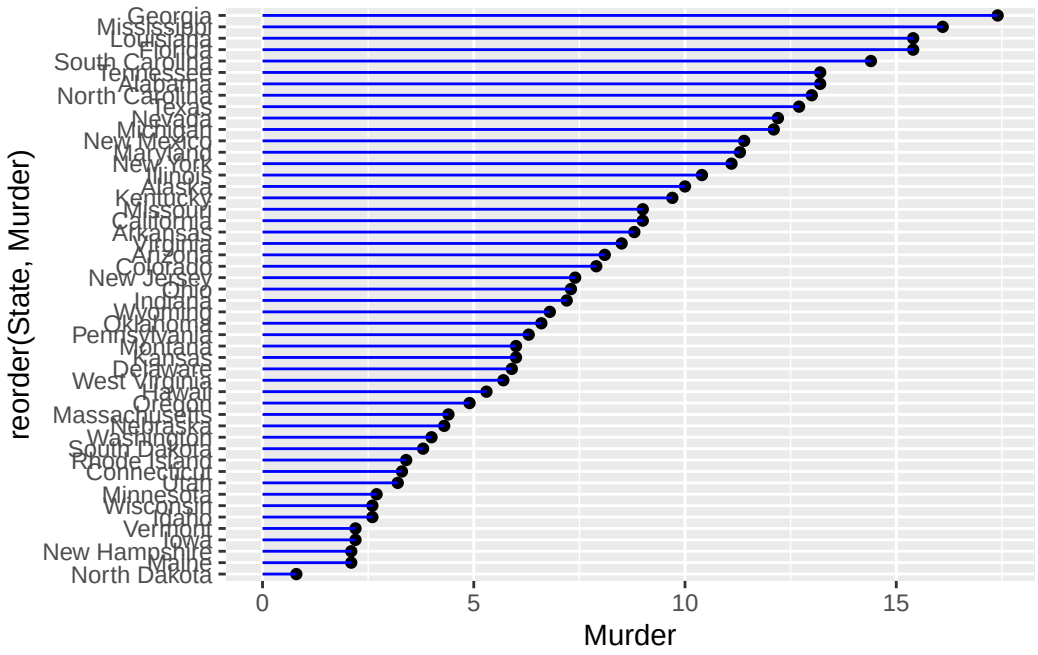

```
head(USArrests)
```

```
           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()
```




```
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()
```



Below is some old stuff

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

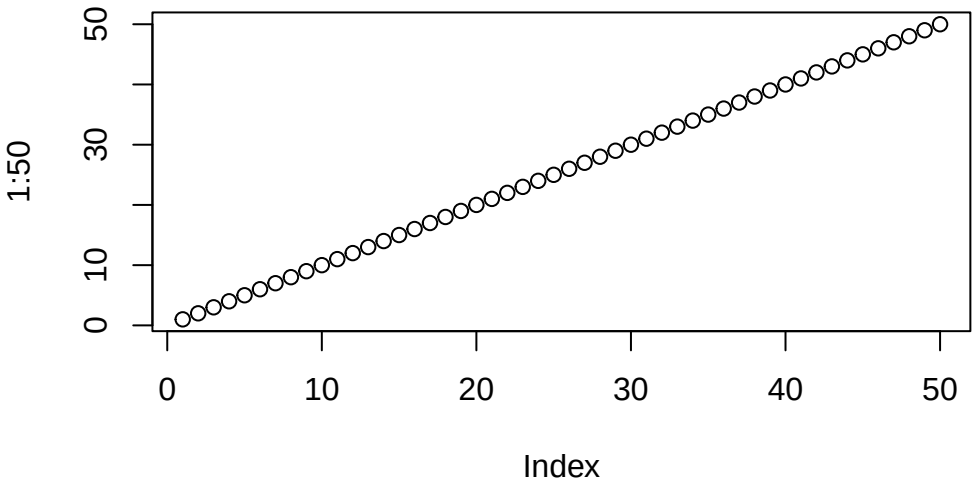
This is some of my text.

```
log(100)
```

```
[1] 4.60517
```

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(1:50)
```



You can add options to executable code like this

```
[1] 4
```

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