Class 05: Viz with ggplot

Carolina Merino PID A14484883

Today we are exploring the ***gggplot*** package and how to make nice figures with R.

There are lots of ways to make figures and plots with R. These include:

- so called "base" R
- and add on packages like **ggplot2**

Here is a simple "base" plot R.

```
head(cars)
```

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

> Key-point: Base R is quick but not so nice looking in some folks eyes.

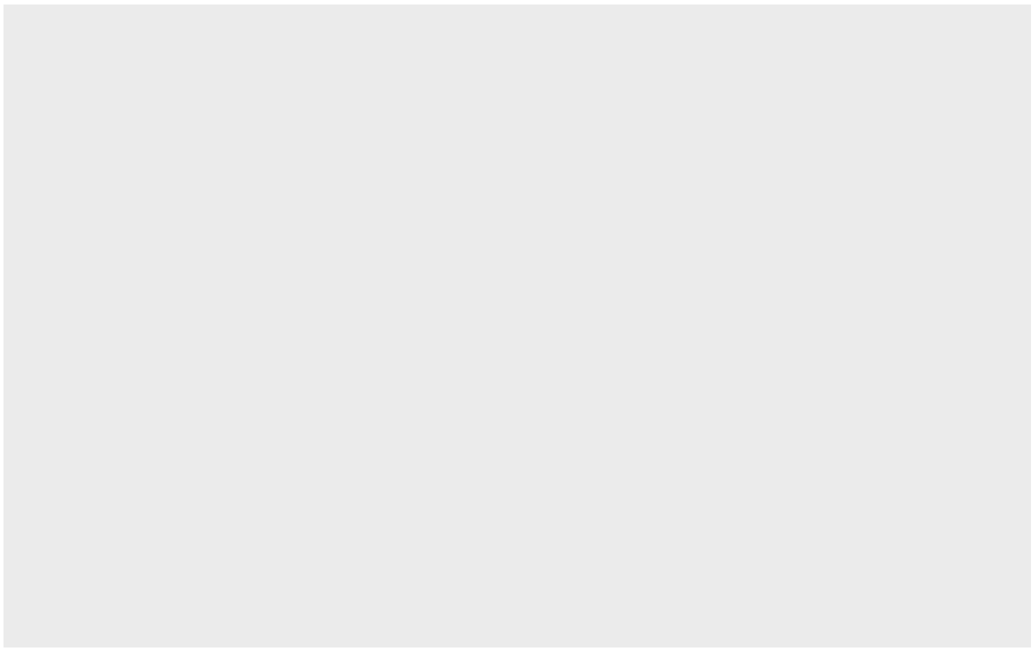
We can simply pass to the `plot()` function

Let's see how we can plot this with **ggplot2**

1st I need to install an add-on package. For this we use `install.packages()` function - **WE DO THIS IN THE CONSOLE, not the report** This is a one time deal.

2nd We need to load the library with the `library()` function everytime we want to use it.

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

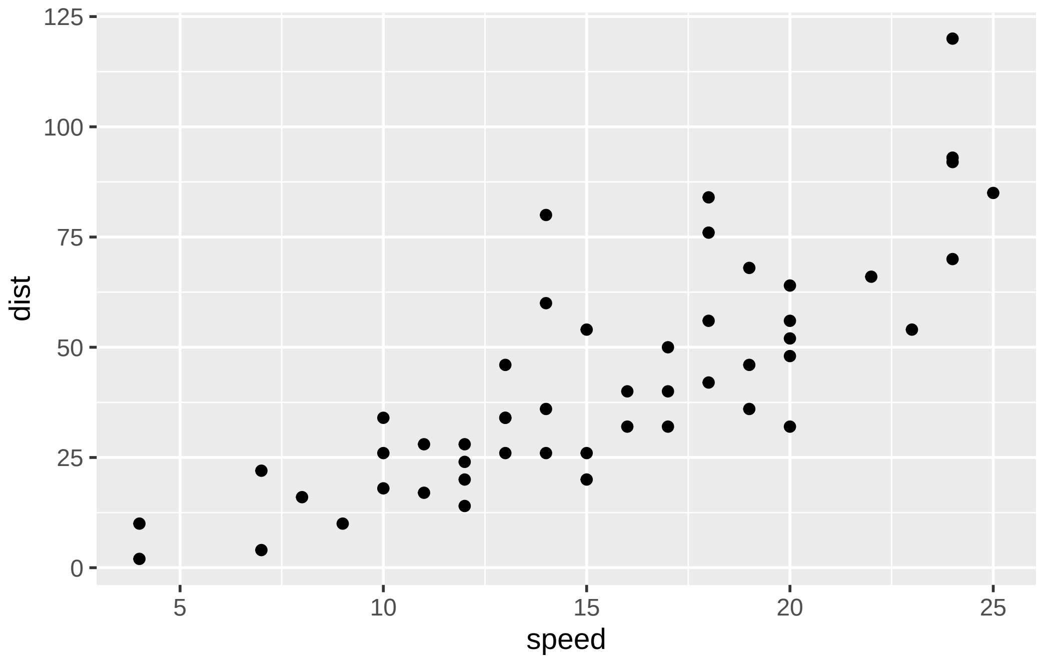


```
#Install the package ggplot2
#Any time I want use this package I need to load
```

> For the graph above: The `ggplot(cars)` code alone doesn't produce output because it lacks specific aesthetic mappings `(aes()) and a geom_*()` function to define the type of plot.

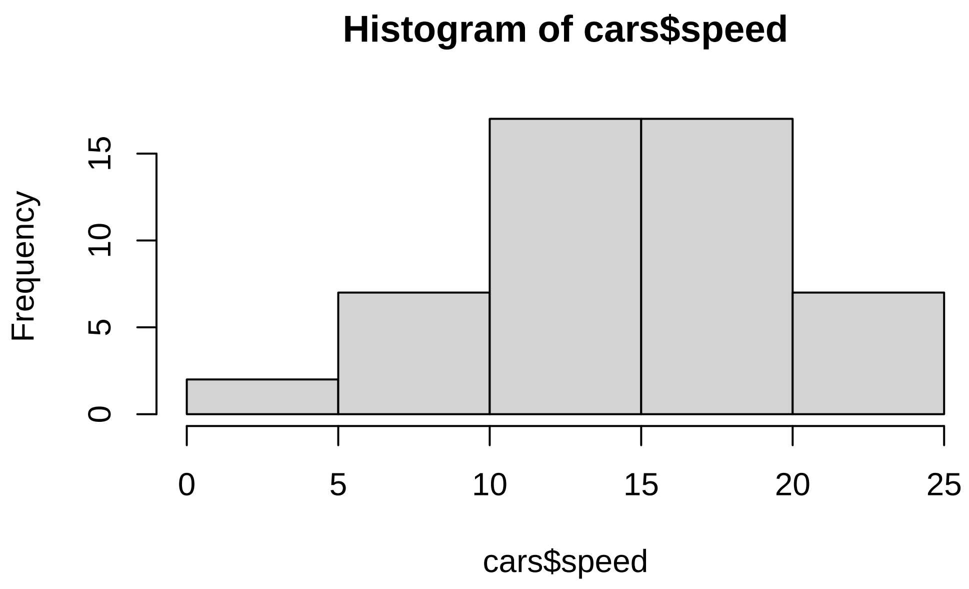
Every ggplot is composed of atleast three layers: - **data**(i.e a data.frame with the thing you want to plot) -aesthitics **aes()** that map the columns of your data to your plot features (i.e aesthitics) -geoms like **geom_point()** that srt how the plot appears

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



```
#Our first ggplot
#we need data +aes +geoms
```

```
hist(cars$speed)
```



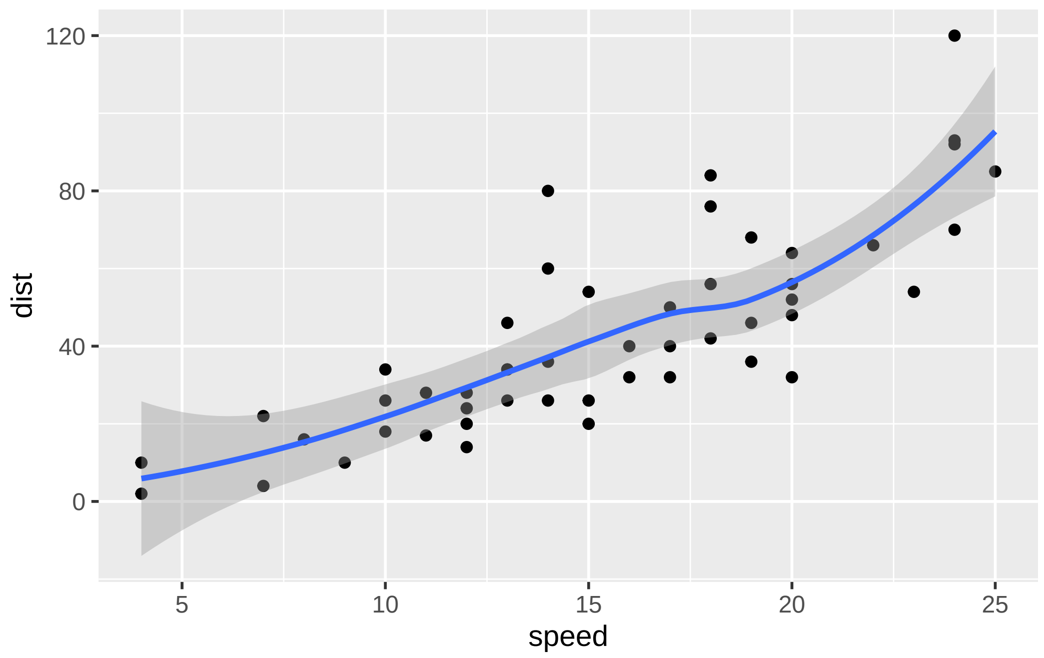
> For simple "canned" graphs base R is quicker and more custom, as things get more elaborate, ggplot wins out…

Let's add more layers to out ggplot

Add a line showing a relationship between x and y

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

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

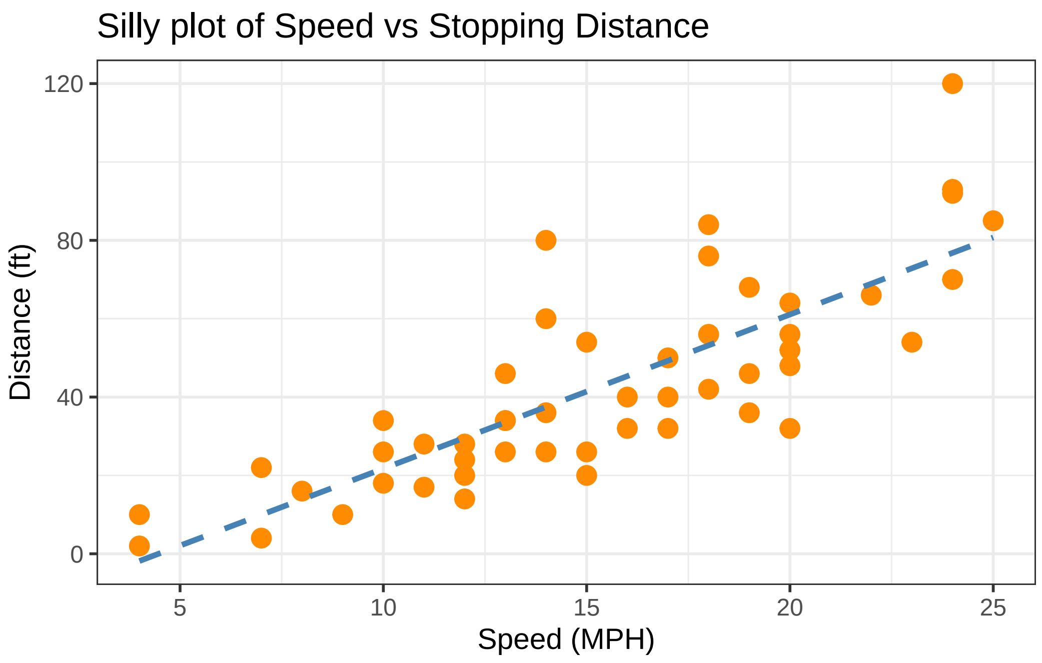


Add a line showing the relationship between x and y Add a title Add custom axis labels “Speed (MPH)” and Distance (ft)”

```r
library(ggplot2)

ggplot(cars, aes(x = speed, y = dist)) +
  geom_point(color = "darkorange", size = 3) +
  geom_smooth(method = "lm", se = FALSE, color = "steelblue", linetype = "dashed", size = 1)
  labs(
    title = "Silly plot of Speed vs Stopping Distance",
    x = "Speed (MPH)",
    y = "Distance (ft)"
  ) +
  theme_bw()
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```

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



- Gene Expression Analysis

Now let's explore gene expression data comparing a control condition to a drug treatment.

We read in a tab-delimited file from the web:

```
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
```

Q1. How many genes are in this dataset?

```
nrow(genes)
```

```
[1] 5196
```

Q2. How many are up-regulated?

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

```

      down unchanging         up 
        72       4997        127 
```

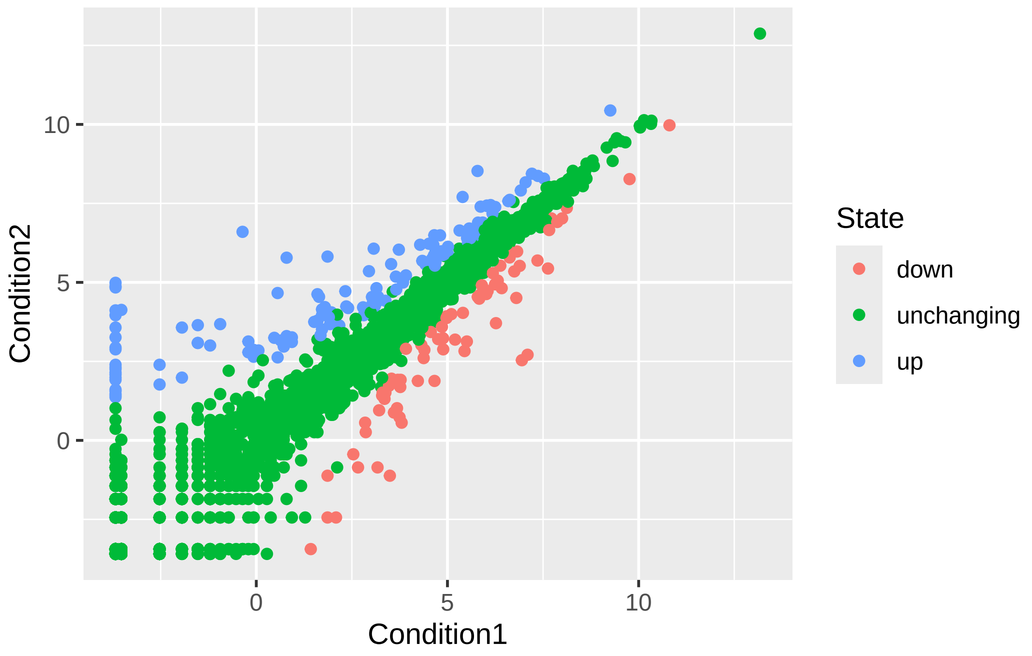
Q3. What fraction of genes are up-regulated?

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

```

      down unchanging         up 
      1.39      96.17       2.44 
```

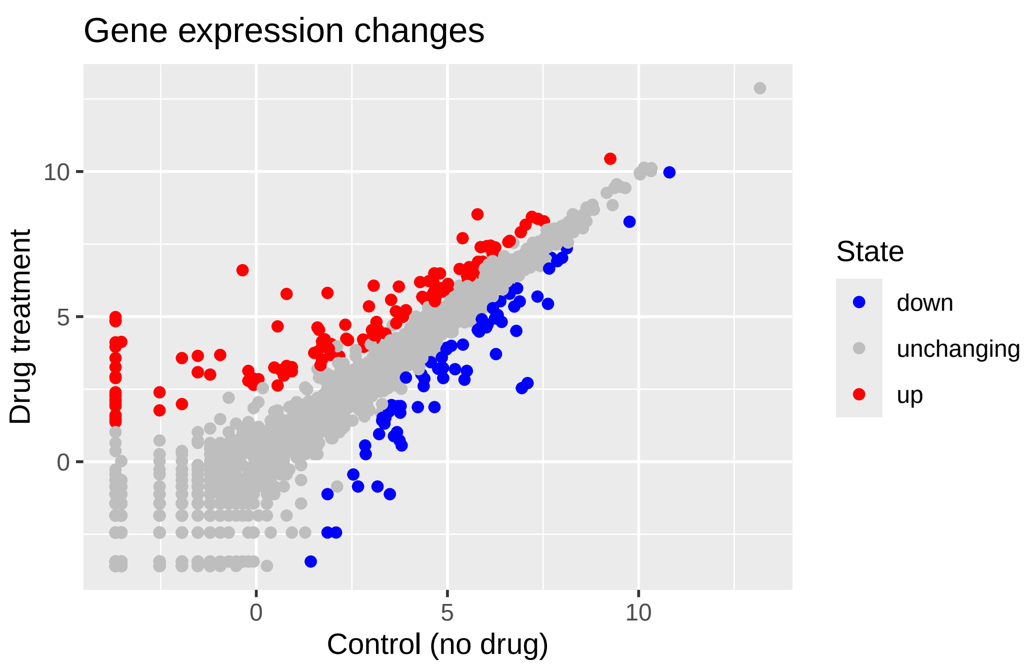
Let's make a first plot attempt

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

g
```



Now, we will be plotting gene expression

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

g +
  scale_color_manual(values = c("blue", "grey", "red")) +
  labs(title = "Gene expression changes",
       x = "Control (no drug)",
       y = "Drug treatment")
```



We will first load our libraries and then take a look a the dataset

```r
library(ggplot2)
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
library(gapminder)

head(gapminder)
```

```
# A tibble: 6 x 6
  country     continent  year lifeExp      pop gdpPercap
  <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
1 Afghanistan Asia       1952    28.8  8425333      779.
2 Afghanistan Asia       1957    30.3  9240934      821.
3 Afghanistan Asia       1962    32.0 10267083      853.
4 Afghanistan Asia       1967    34.0 11537966      836.
5 Afghanistan Asia       1972    36.1 13079460      740.
6 Afghanistan Asia       1977    38.4 14880372      786.
```

Q4. How many different country values are in this dataset?

```r
length( table(gapminder$country))
```

```
[1] 142
```

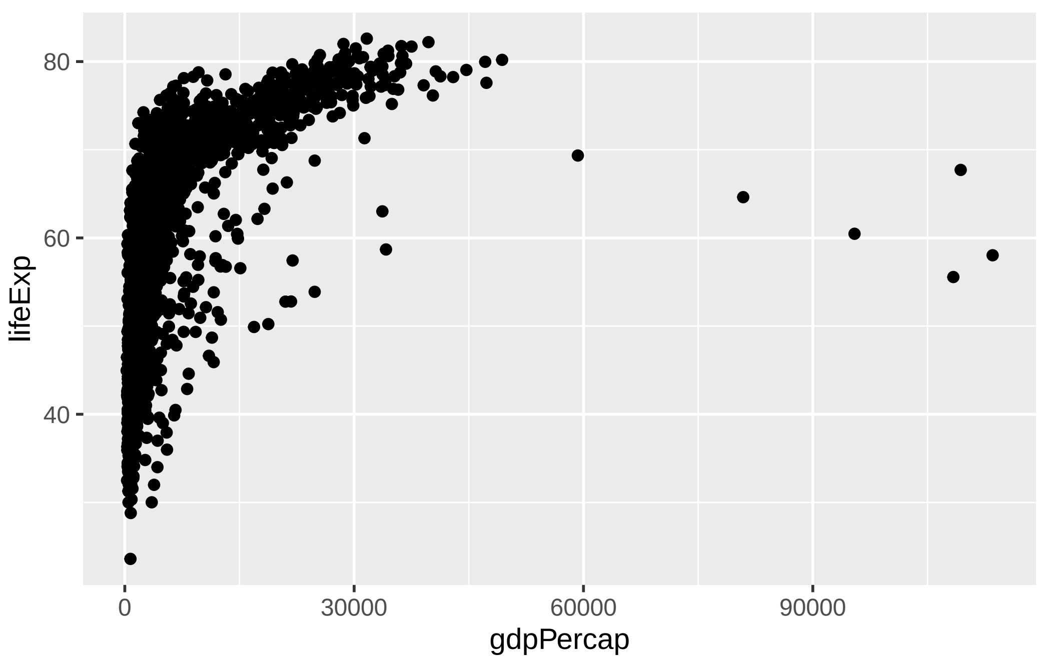
Q5. How many different continent values are in this dataset?

```r
unique(gapminder$continent)
```

```
[1] Asia     Europe   Africa   Americas Oceania
Levels: Africa Americas Asia Europe Oceania
```

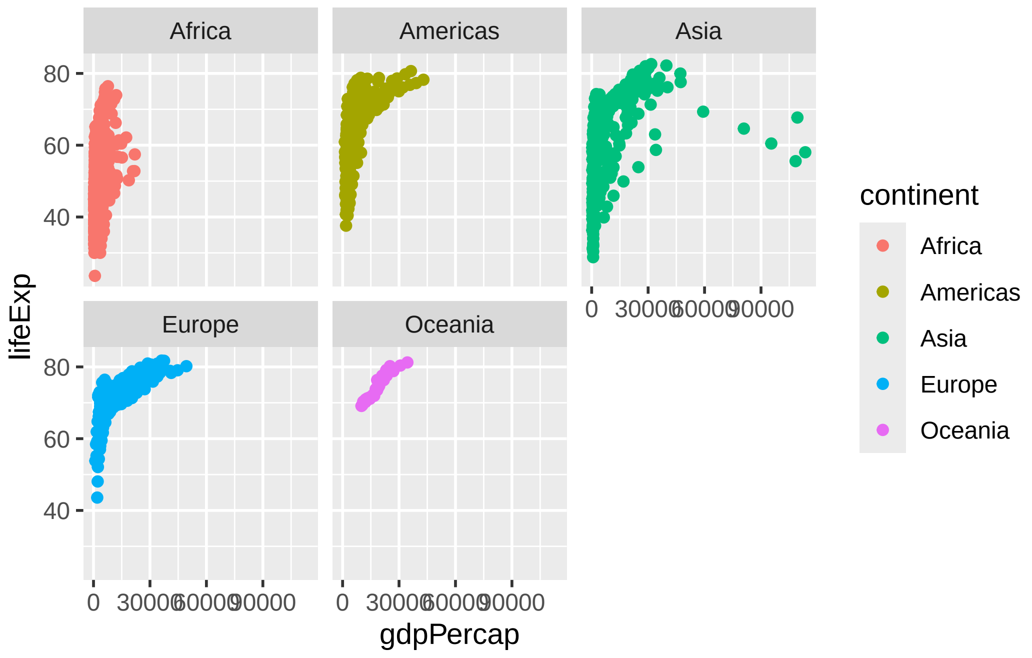
Let's make a plot like the one in the noble prize: Life exp. vs. gdpPerCap - simple graph, including explicit x and y

```r
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point()
```



Let's do a facet wrap

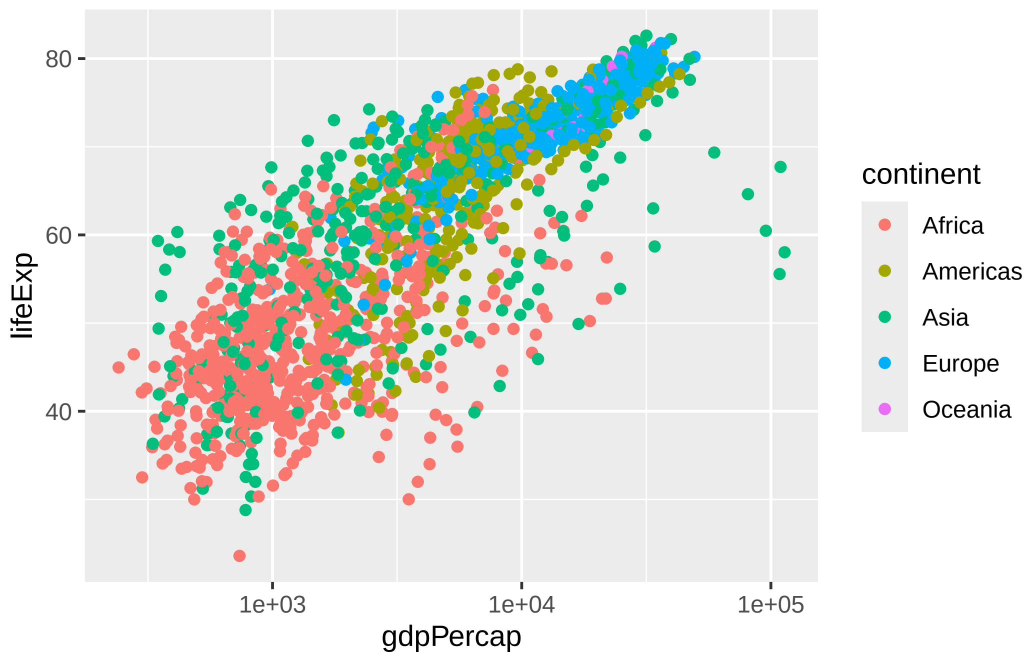
```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent)) +
  geom_point() +
  facet_wrap(~continent)
```



Now, let's add some color for more appeal and easier to distinguish the data

- color by continent
- Log transformed x-axis

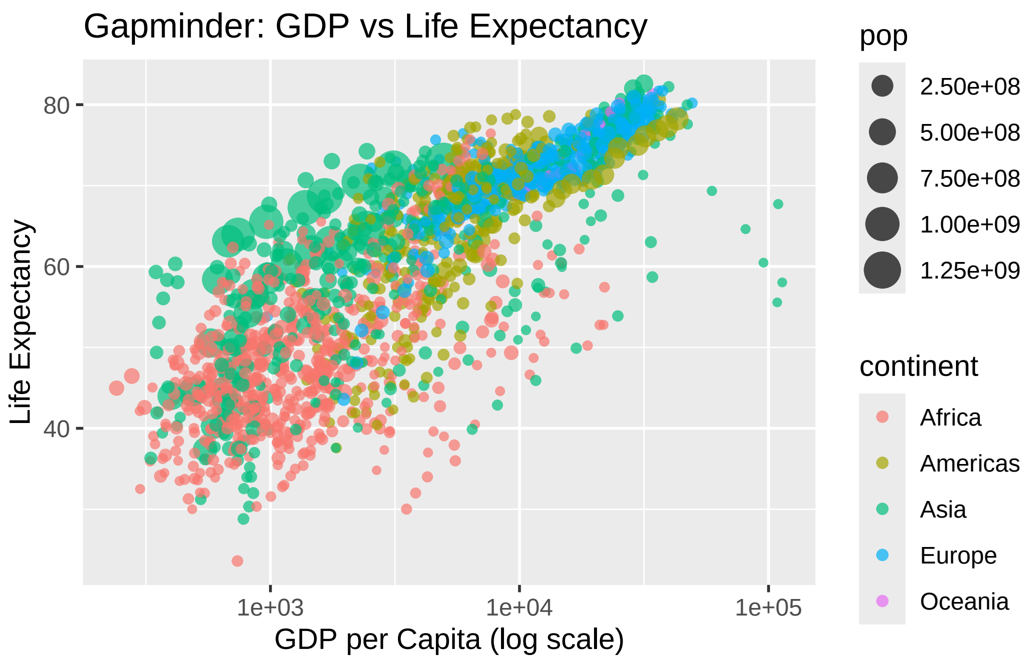
```r
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent)) +
  geom_point() +
  scale_x_log10()
```



Let's properly label our Graph to:

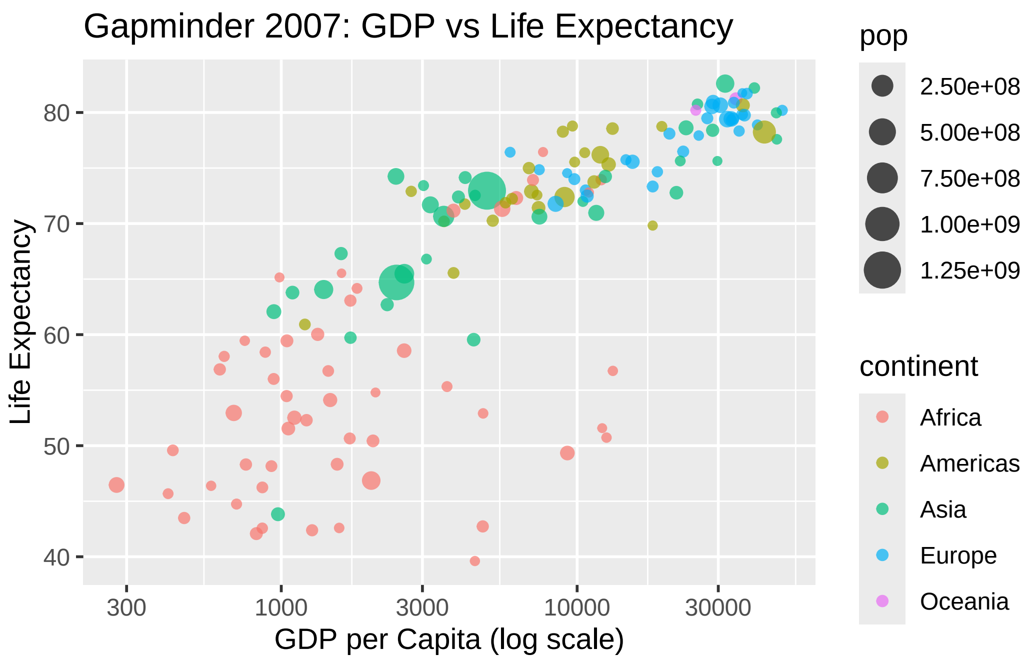
Gapminder: GDP vs Life Expectancy

```r
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp,
                      color = continent, size = pop)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  labs(
    title = "Gapminder: GDP vs Life Expectancy",
    x = "GDP per Capita (log scale)",
    y = "Life Expectancy"
  )
```



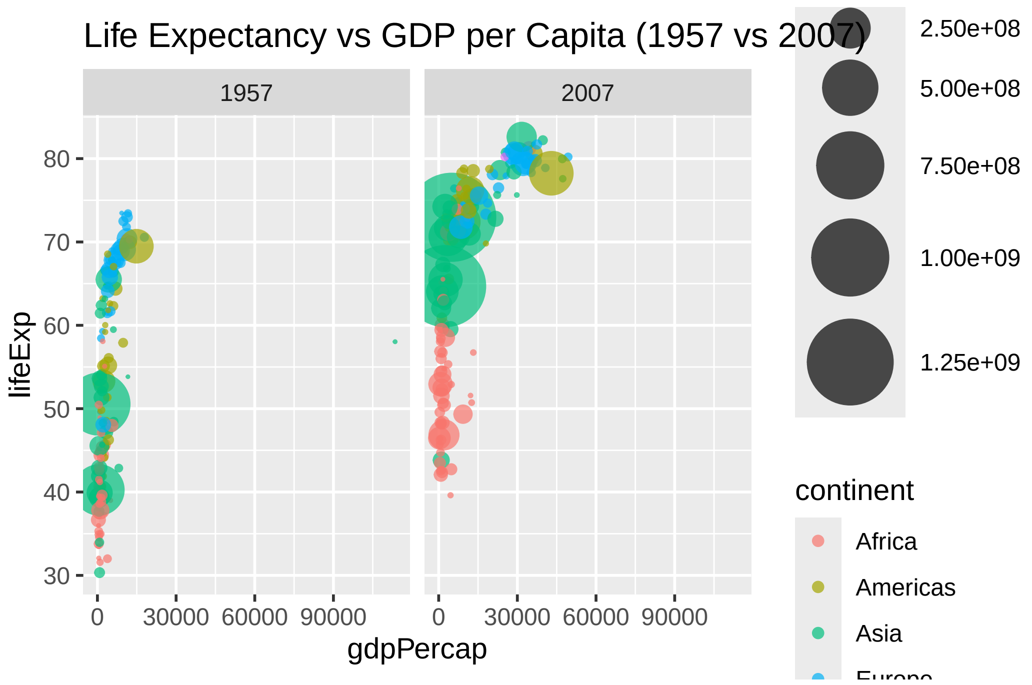
Next, is to filter for the year 2007 only

```r
gap2007 <- gapminder %>% filter(year == 2007)

ggplot(gap2007, aes(x = gdpPercap, y = lifeExp,
                    color = continent, size = pop)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  labs(
    title = "Gapminder 2007: GDP vs Life Expectancy",
    x = "GDP per Capita (log scale)",
    y = "Life Expectancy"
  )
```



Lastly, will be comparing GDP vs Life Expectancy in 1957 and 2007 side by side

```
gap_combo <- gapminder %>% filter(year %in% c(1957, 2007))

ggplot(gap_combo, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +
  geom_point(alpha = 0.7) +
  scale_size_area(max_size = 15) +
  facet_wrap(~ year) +
  labs(title = "Life Expectancy vs GDP per Capita (1957 vs 2007)")
```



Let's Filter top 5 countries by population in 2007

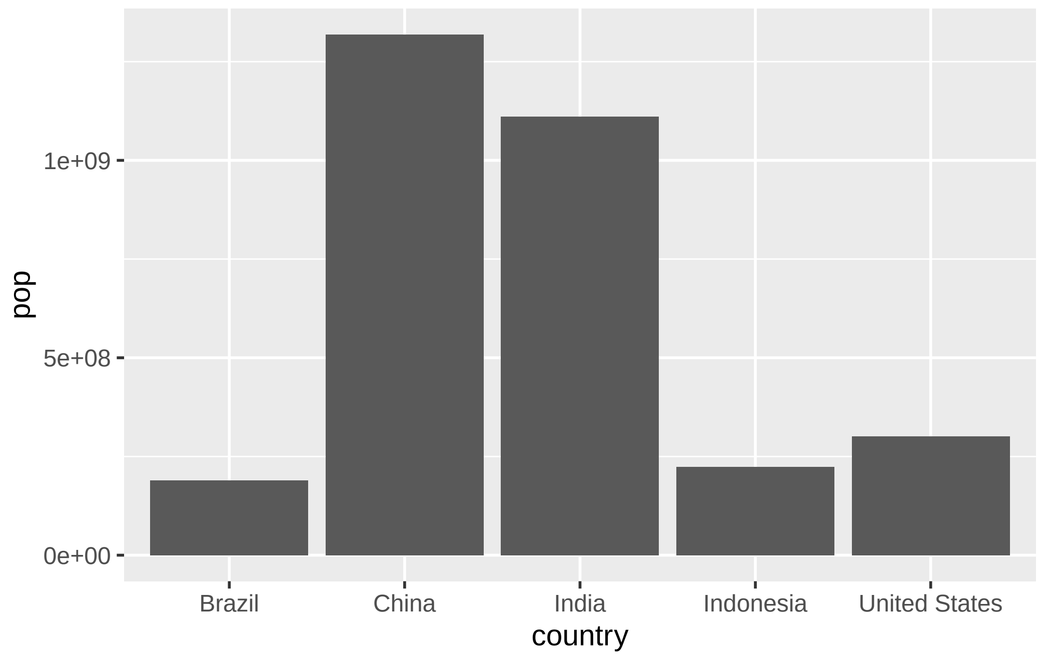
and then view the filtered dataset

```
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.
```

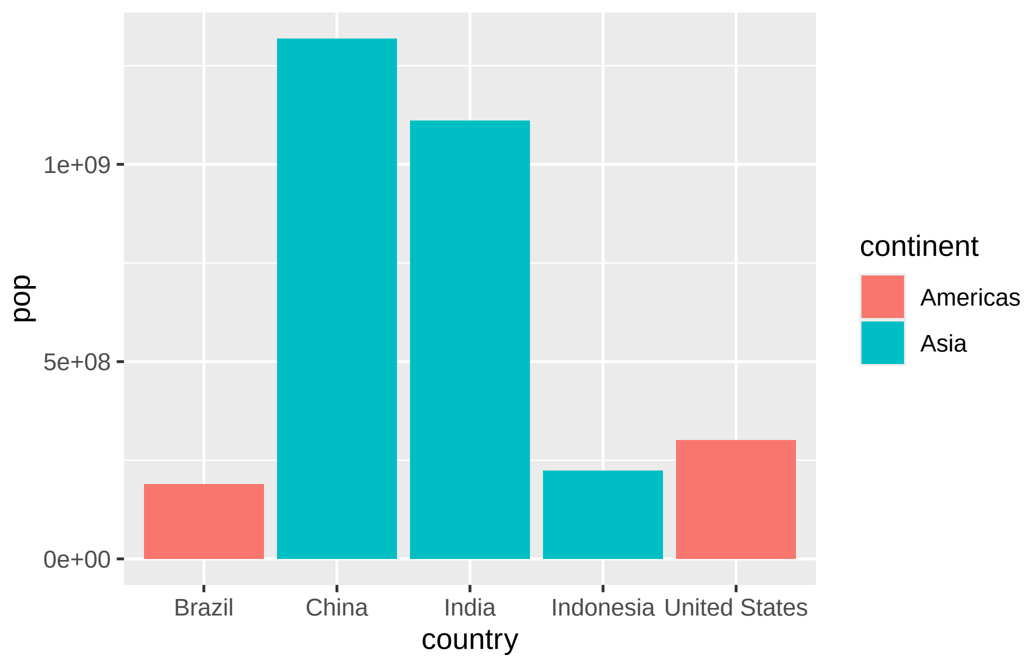
Creating a basic bar chart: population by country

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



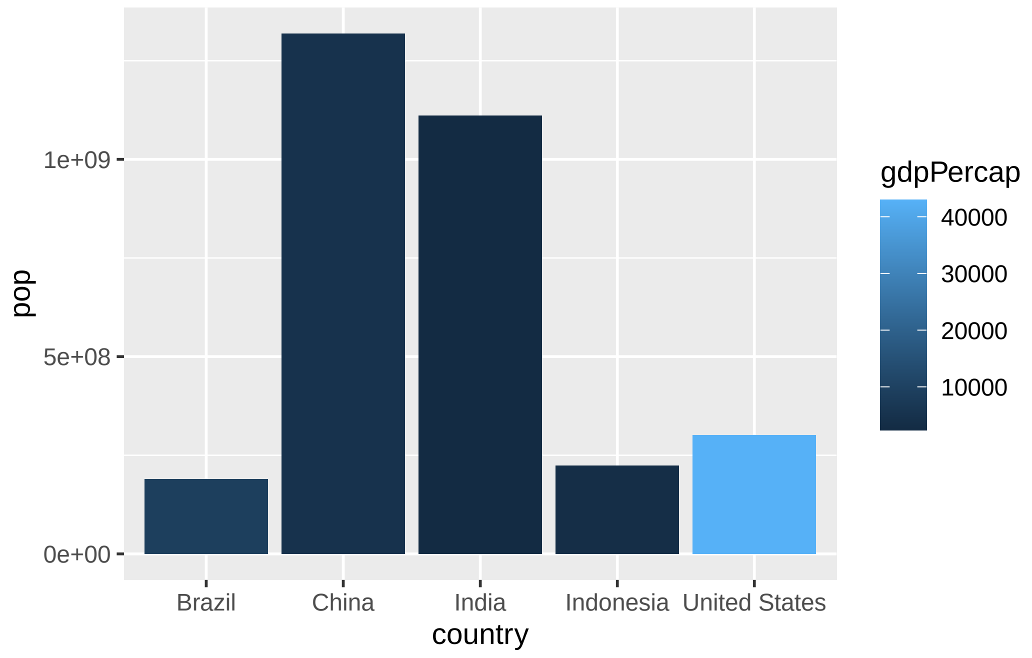
Bar chart filled by continent

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



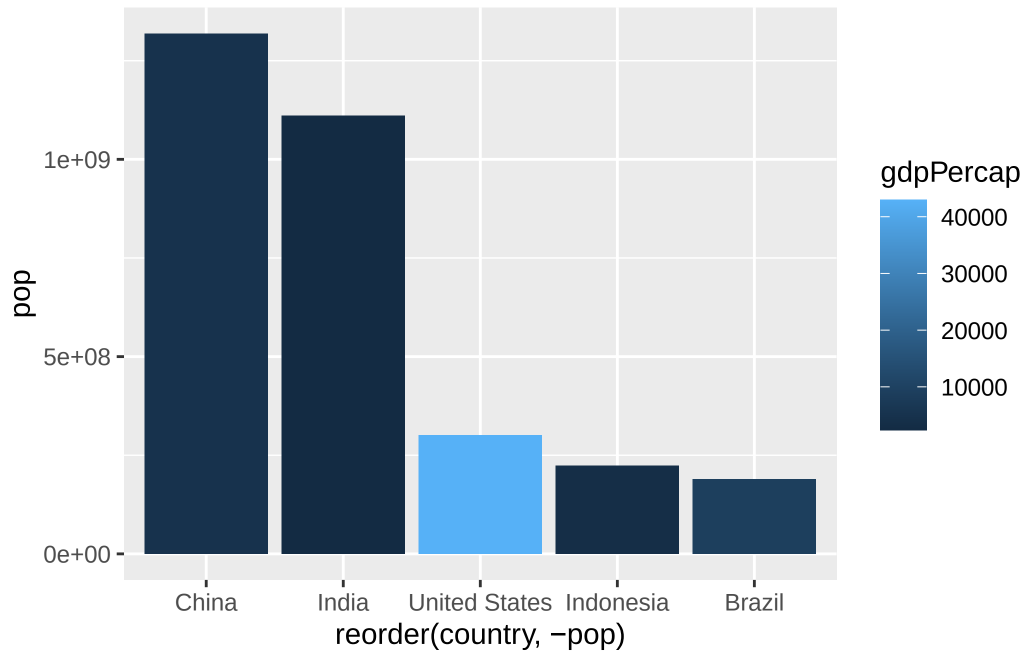
Will set bar chart filled by GDP per capita

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



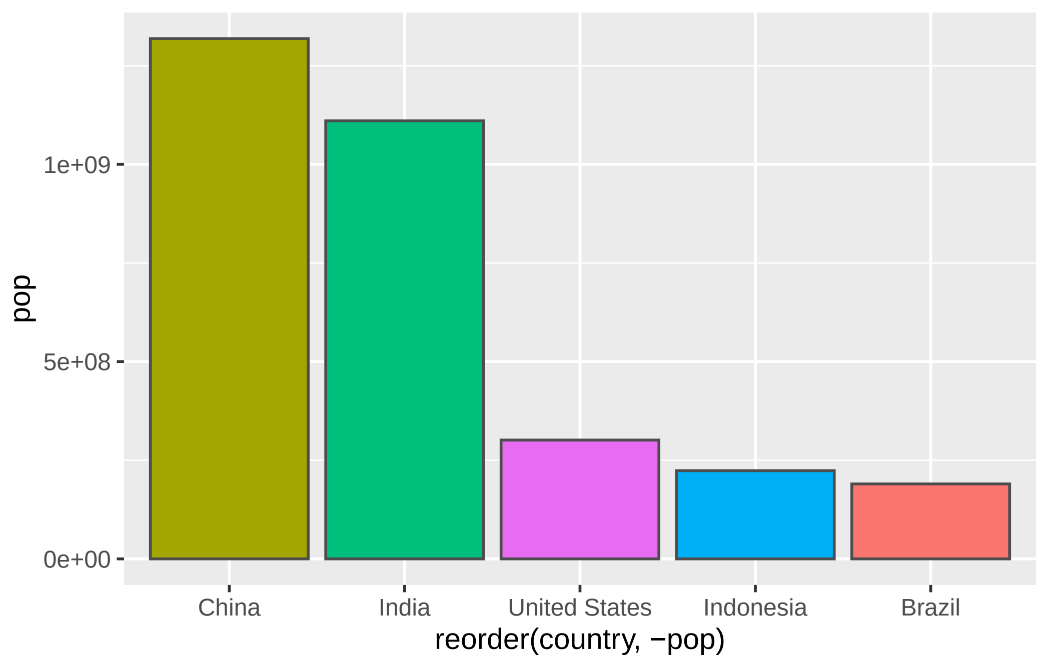
Let's reorder bars by population size descending

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



Now, let's fill bars by country and remove legend

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



Flipping Bar charts

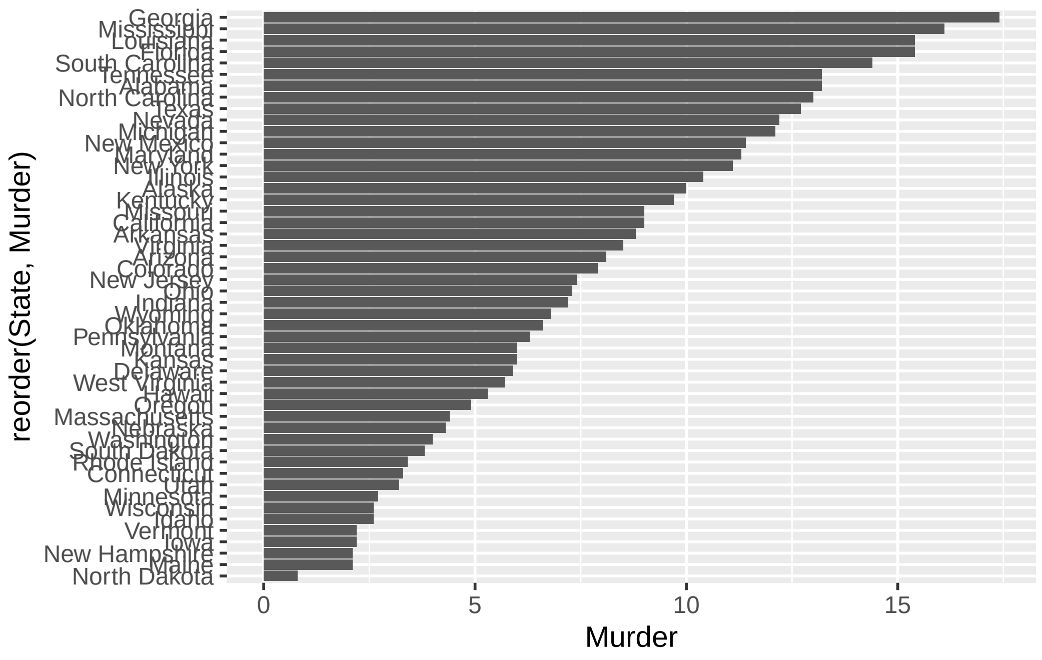
USArrests dataset: rownames are state names; we add them as a new column

```
USArrests$State <- rownames(USArrests)

# Check the first few rows
head(USArrests)
```

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

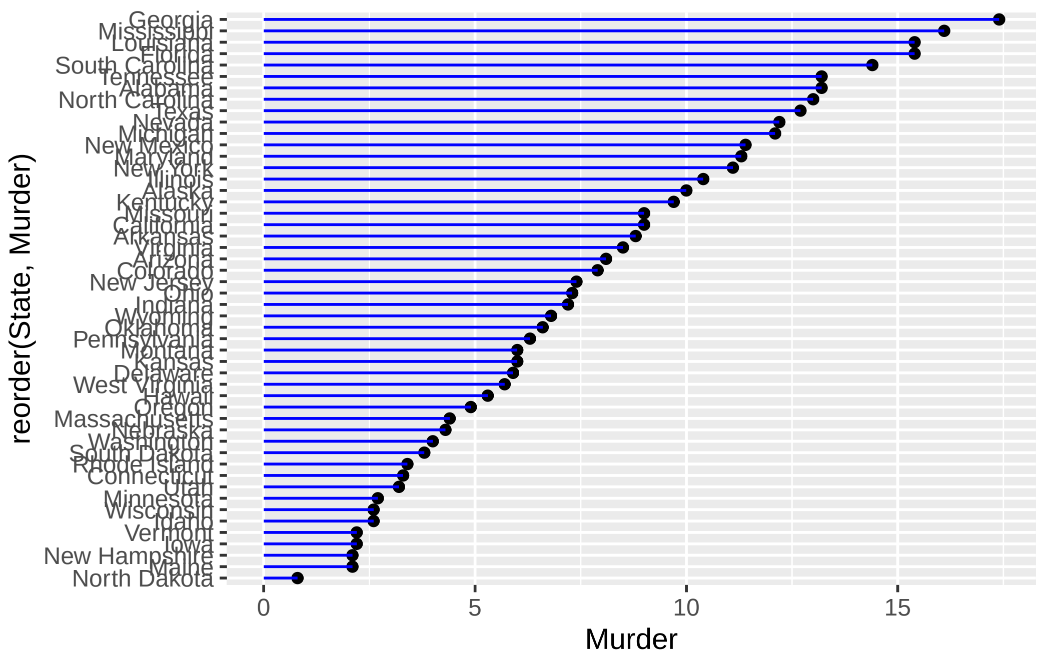
- Basic flipped bar chart

```r
ggplot(USArrests) +
  aes(x = reorder(State, Murder), y = Murder) +
  geom_col() +
  coord_flip()
```



Now, let's make the chart cleaner

```r
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()
```



Let's do an animated graph This has been commented out, due to not working for PDF

#"'{r} # Load necessary libraries library(gapminder) library(gganimate)

Set up the plot with gapminder data

p <- 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_wrap(~continent) + labs(title = 'Year: {frame_time}', x = 'GDP per capita', y = 'Life expectancy') + transition_time(year) + shadow_wake(wake_length = 0.1, alpha = FALSE)

Render the animated graph

p #"'

Summary

Lastly What are the advantages of ggplot over base R plot

1. Grammar of Graphics (Structured Approach)

ggplot2 uses a consistent, layered approach to building plots (based on the Grammar of Graphics).

This makes it easier to build complex visualizations step-by-step using layers (+), rather than relying on scattered commands.

2. Better Aesthetics and Defaults

ggplot2 comes with more visually appealing defaults.

Colors, labels, spacing, legends, and themes look professional out of the box—often ready for publication.

3. Faceting (Small Multiples)

Powerful built-in support for faceting, which means splitting data by variables into subplots.

4. Consistent Syntax

All plot types (scatterplots, histograms, bar charts, etc.) use the same basic syntax and structure.

In base R, plotting functions like `plot()`, `hist()`, `boxplot()` each have different arguments and syntax.

5. Customizability and Theming

`ggplot2` allows deep customization using themes and scale functions.

You can easily control legends, axis labels, titles, color scales, etc.

6. Layering and Reusability

You can store parts of a plot as objects and reuse them or add new layers later.

Encourages modular, clean code.

7. Ecosystem and Extensions

Many powerful extensions available, such as:

ggthemes, gganimate, plotly, ggrepel, patchwork, etc.

These allow for interactive plots, animations, and publication-quality themes.

8. Data-Driven

ggplot2 is more data-aware: it works best with tidy data frames and automatically maps aesthetics to data variables.

Base R often requires more manual work for handling grouped or categorical data.