Class 05: Data Visualization with GGPLOT

Ashley Mazon (PID:A17478903)

Today we are exploring the **ggplot** package and how to make nice figures in R.

There are lots of ways to make figures and plot in R. These include:

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

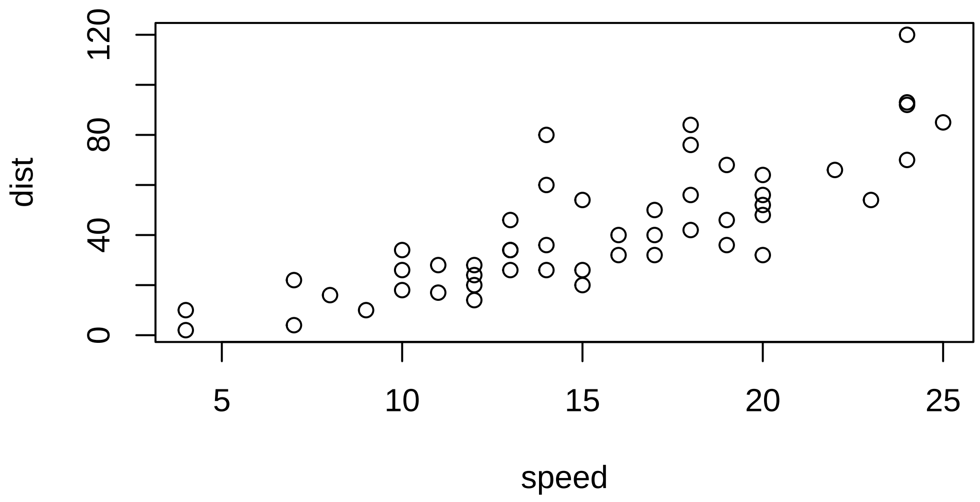
Here is a simple "base" R plot.

```
head(cars)
```

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

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

```
plot(cars)
```



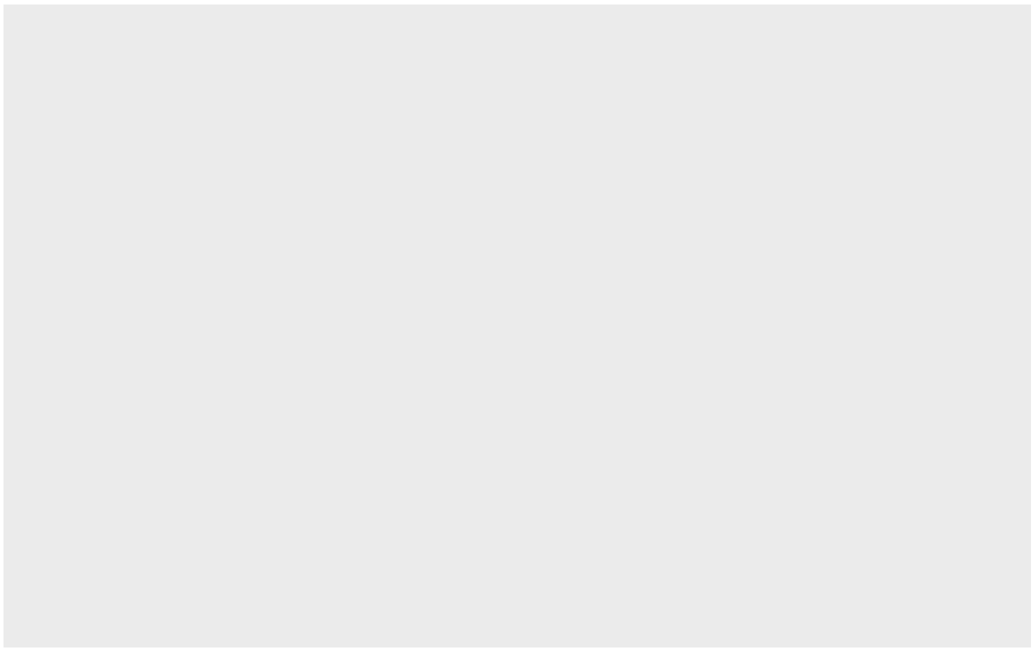
Key point: Base R is quick but not so nice looking in some folks eyes

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

1st I would need to install this add-on package. For this we use the `install.packages()` function - **WE DO THIS IN THE CONSOLE**. This is a one time only deal

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

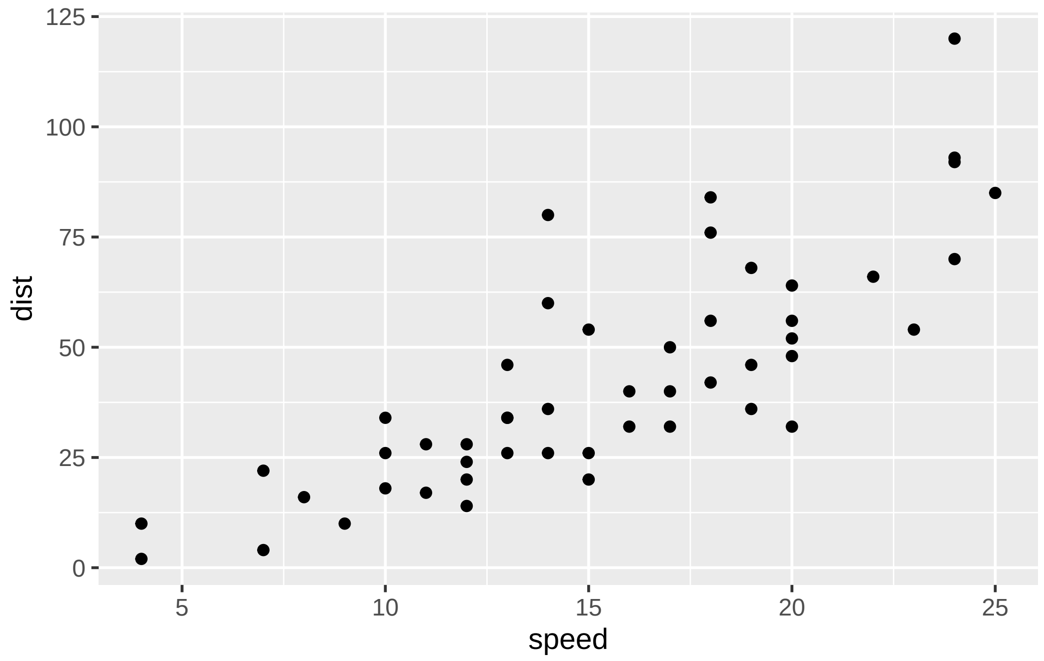
```
library(ggplot2)
ggplot(cars)
```



Every ggplot is composed of at least 3 layers:

- **data** (i.e a data.frame with the things you want to plot)
- aesthetics **aes()** that map th ecoumns of data to you plot features (i.e aesthetics)
- geoms like **geom_point** that srt how the plot appears

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



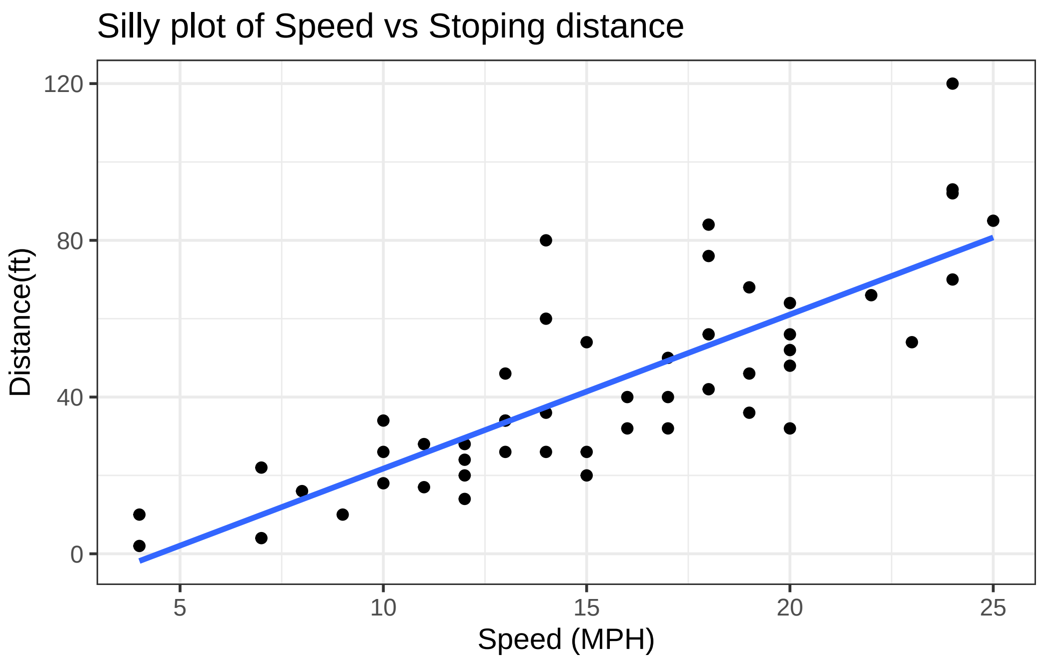
> Key Point: For simple "canned" graphs base R is quicker but as things get more custom and elaborate then ggplot wins out…

Let's add more layers to our ggplot

Add a line showing the relationship between x and y Add a title Add custom axis labels "Speed(MPH)" and "Distance(ft)" Change the theme…

```
ggplot(cars)+
  aes(x=speed, y=dist)+
  geom_point()+
  geom_smooth(method=lm, se=FALSE)+
  labs(title= "Silly plot of Speed vs Stoping distance", x="Speed (MPH)", y="Distance(ft)")+
  theme_bw()
```

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



Going Further

Read some gene expression data

```r
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?

```r
nrow(genes)
```

```
[1] 5196
```

Q2. How many columns are there?

```r
ncol(genes)
```

```
[1] 4
```

Q3. How many upregulated genes are there?

```r
sum(genes$State == "up")
```

```
[1] 127
```

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

```

      down unchanging         up 
        72       4997        127 
```

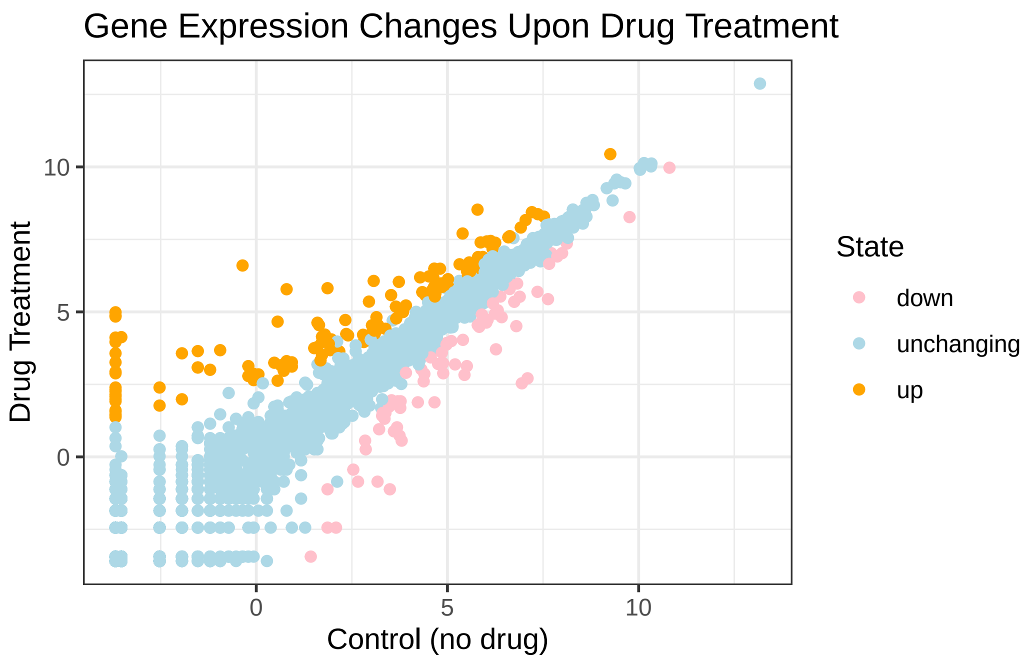
Q4. What fraction of the total genes is unregulated?

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

```

      down unchanging         up 
      1.39      96.17       2.44 
```

```r
p <- ggplot(genes) +
  aes(x=Condition1, y=Condition2, col=State)+
  geom_point()+
  scale_colour_manual( values=c("pink","lightblue","orange") )+
  labs(title= "Gene Expression Changes Upon Drug Treatment", x="Control (no drug)", y= "Drug
  theme_bw()
p
```



More Plotting

Read in the gapminder dataset

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

gapminder <- read.delim(url)
```

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

```
nrow(gapminder)
```

```
[1] 1704
```

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

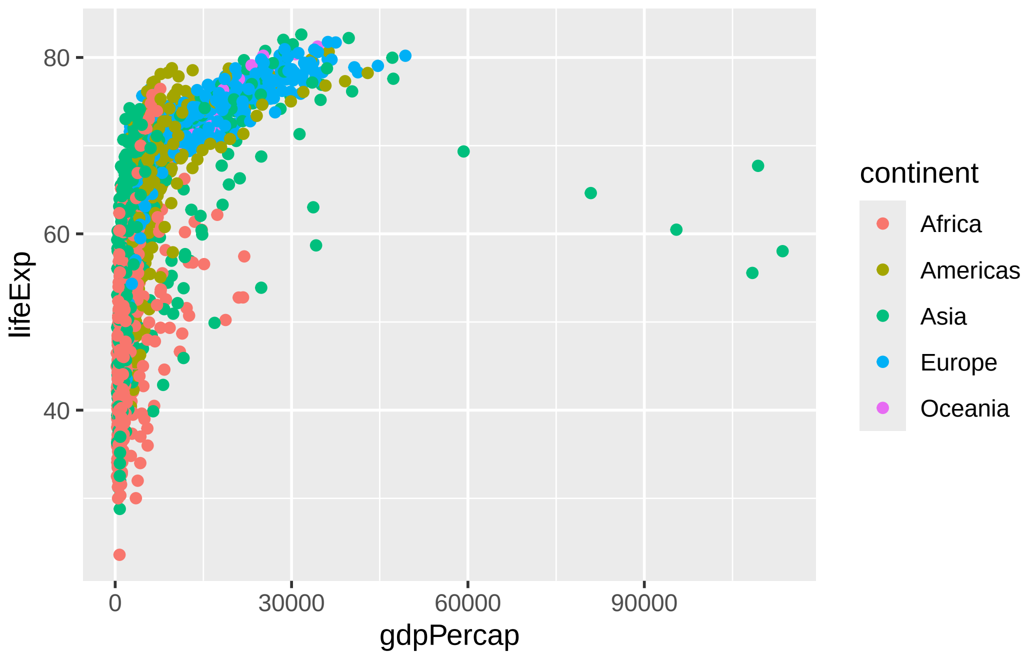
```
[1] 142
```

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

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

```
[1] "Asia"     "Europe"   "Africa"   "Americas" "Oceania"
```

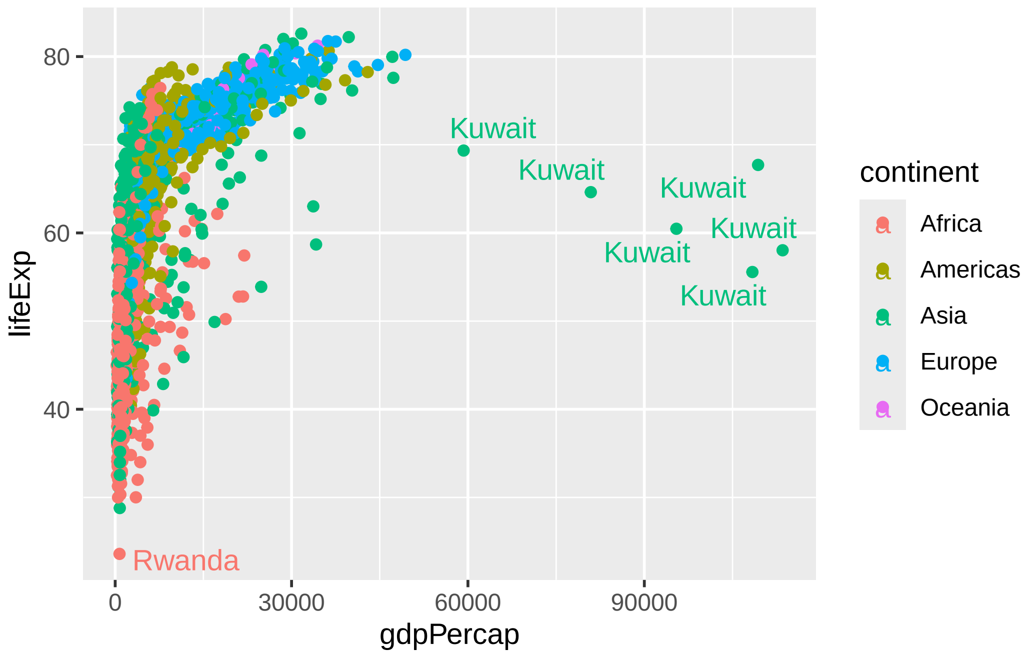
```r
ggplot(gapminder)+
  aes(gdpPercap, lifeExp, col=continent)+
  geom_point()
```



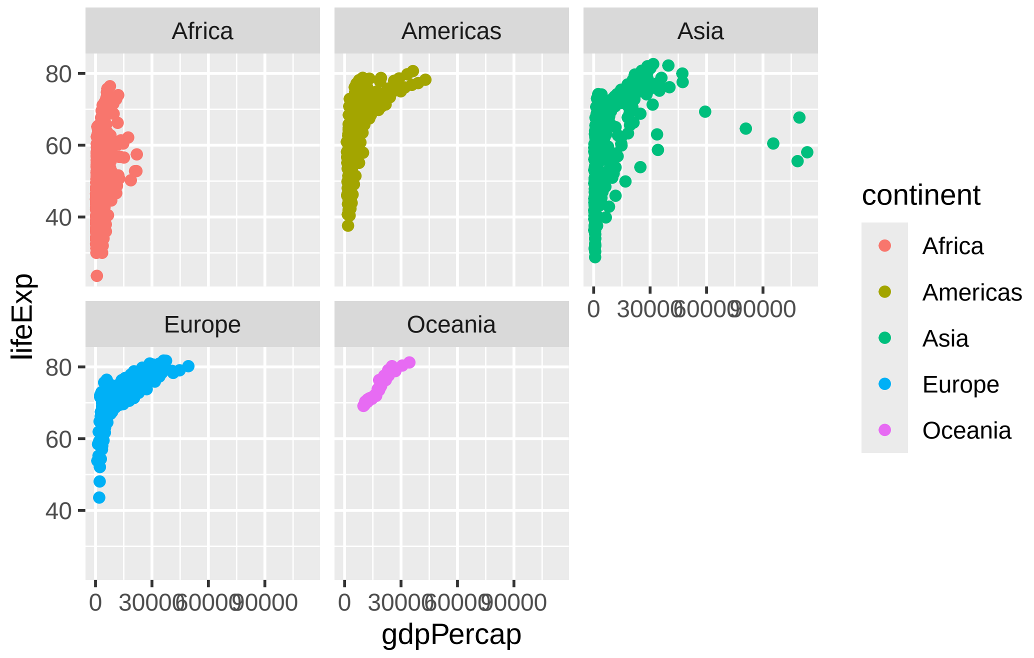
I can use the **ggrepel** package to make more sensible labels here

```r
library(ggrepel)
ggplot(gapminder)+
  aes(gdpPercap, lifeExp, col=continent, label=country)+
  geom_point()+
  geom_text_repel()
```

```
Warning: ggrepel: 1697 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```



```
ggplot(gapminder)+
  aes(gdpPercap, lifeExp, col=continent, label=country)+
  geom_point()+
  facet_wrap(~continent)
```



The main advantages of ggplot vs Base R plots:

- Consistency: ggplot2 uses a unified grammar for building plots, so you specify data, aesthetics, and geometric layers in a consistent way for all plot types. Base R requires different functions and arguments for each plot type, which can be less intuitive and harder to remember
- Layering: ggplot2 lets you build plots by adding layers (e.g., points, lines, labels) with the + operator, making complex plots easier to construct and modify. Base R plots require manual additions and can be more cumbersome for multi-layered figures
- Defaults and Appearance: ggplot2 produces attractive, publication-quality graphics by default, while Base R plots often need extensive tweaking to look polished
- Customization: ggplot2 makes it easier to customize aesthetics (color, shape, size, etc.) and add features like legends, themes, and annotations. Base R offers fine control but can be fiddly and time-consuming for newcomers
- Data Handling: ggplot2 works directly with data frames and can map multiple variables to different aesthetics in a single call. Base R often requires manual data manipulation before plotting