Class 05: Data Visualization

Trinity (PID: A15766955)

Base R graphics vs ggplot2

There are many graphics systems available in R, including so-called "base" R graphics and the very popular **ggplot2** package.

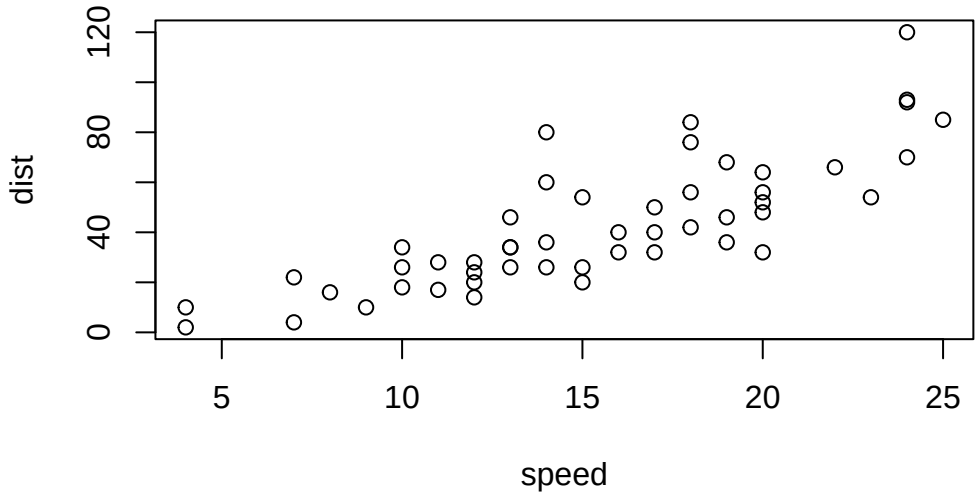
To compare there let's play with the inbuilt `cars` dataset.

```
head(cars)
```

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

To use "base" R I can simply call the `plot()` function:

```
plot(cars)
```



To use `ggplot2` package, I first need to install it with the function`install.packages("ggplot2")`.

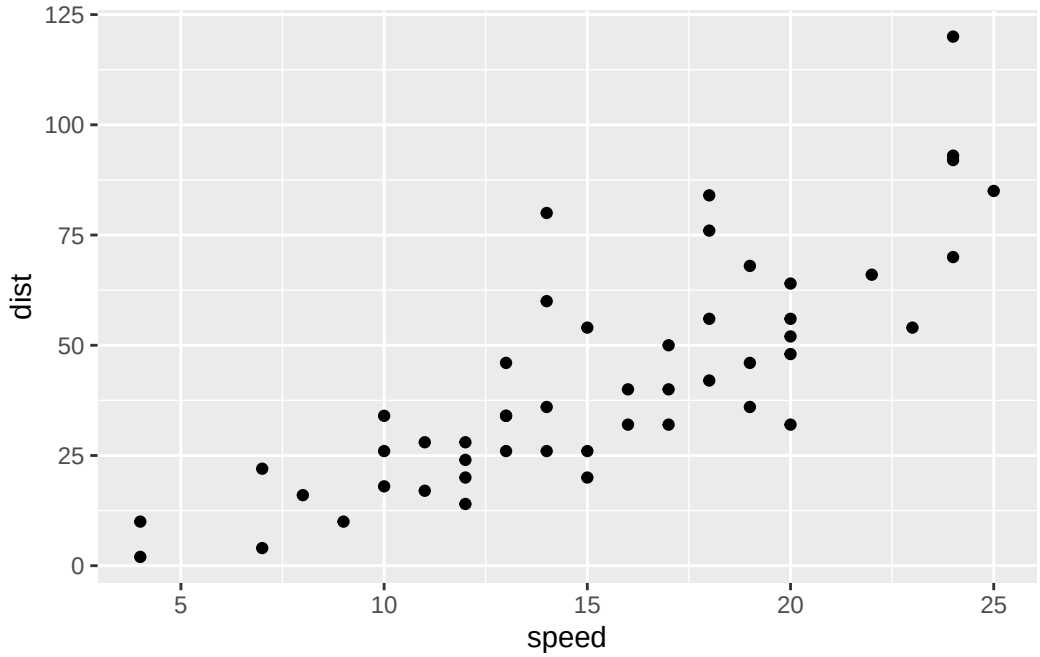
I will run this in my R console (i.e. the R brain) as I do not want to re-install every time I render my report...

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

To make a figure with ggplot, I always need three things:

- **data** (what I want to plot)
- **aes** (the aesthetic mapping of the data)
- **geoms** (how I want to plot the data)

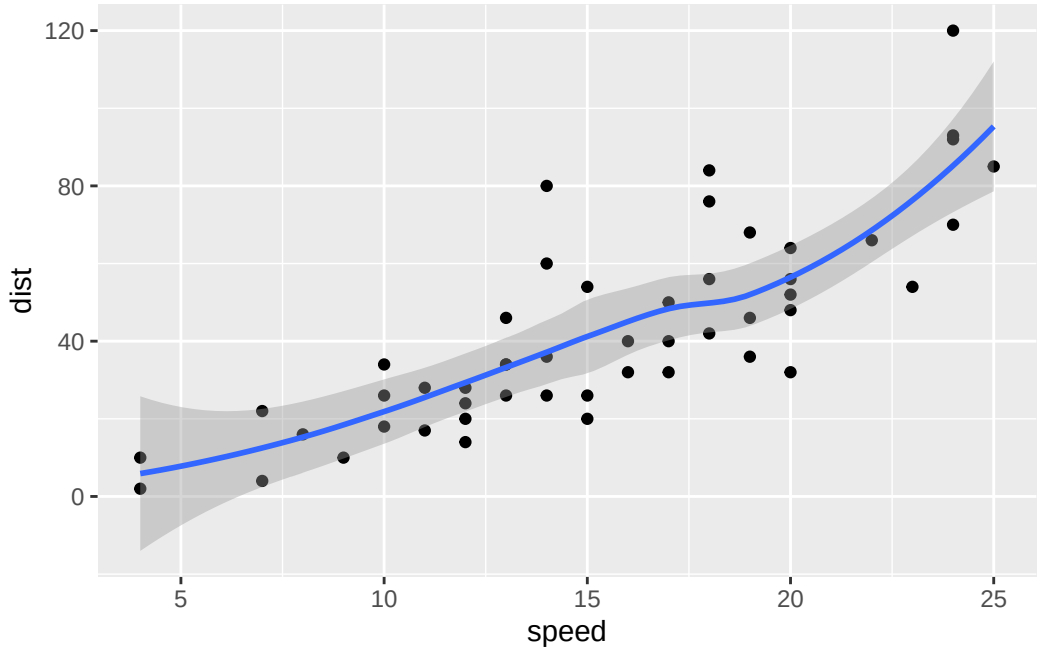
```
ggplot(data=cars) +
         aes(x=speed, y=dist) +
         geom_point()
```



I can just keep adding layers.

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

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

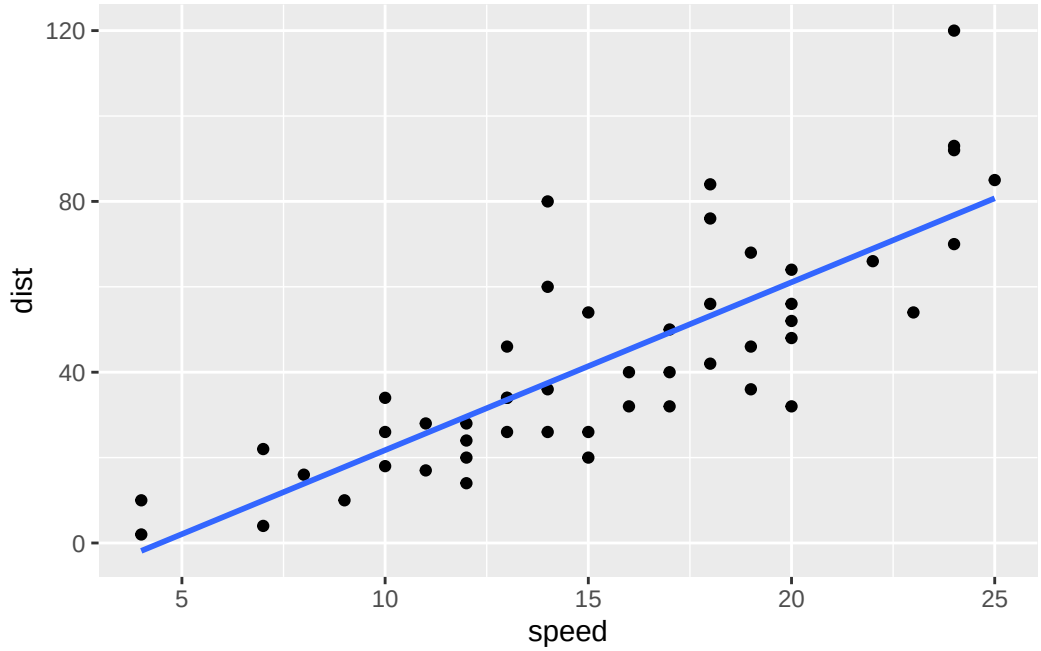


Ggplot is much more verbose than base R plots for standard plots but it has a consistent later system that I can use to make just any plot.

Let's make a plot with a straight line fit - i.e. a linear model and no standard error shown.

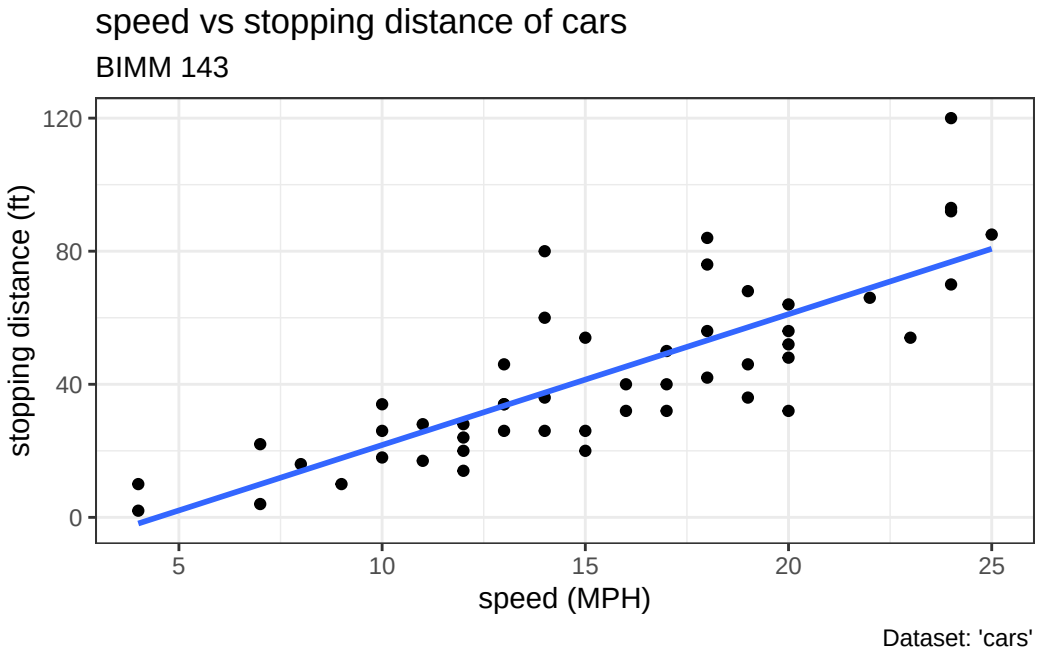
```
ggplot(data=cars) +
         aes(x=speed, y=dist) +
         geom_point() + geom_smooth(se = FALSE, method = lm)
```

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




```
ggplot(data=cars) +
 aes(x=speed, y=dist) +
 geom_point() +
 labs(title="speed vs stopping distance of cars", x ="speed (MPH)", y="stopping distance (
 geom_smooth(se = FALSE, method = "lm") +
 theme_bw()
```

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



A more complicated plot

Let's plot some gene expression data

Let's turn for a moment to more relevant example data set. The code below reads the results of a differential expression analysis where a new anti-viral drug is being tested.

```
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. How many genes are in this dataset?

```
nrow(genes)
```

```
[1] 5196
```

Q. How many columns are there and what are their names?

```
colnames(genes)
```

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

```
ncol(genes)
```

```
[1] 4
```

Q. How can we summarize that last column - the "state" column?

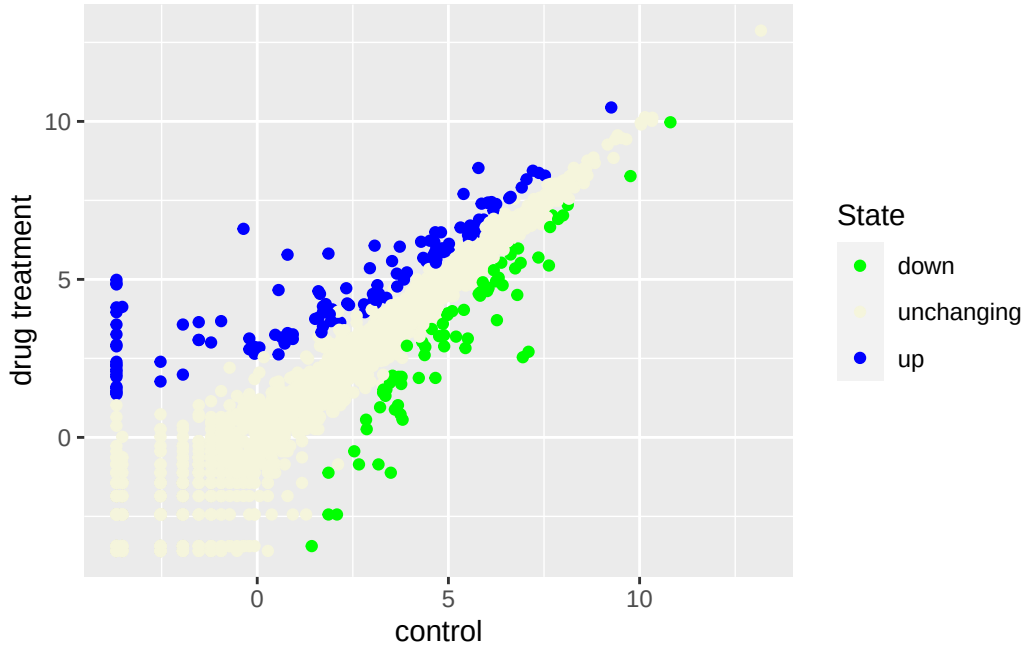
```
table(genes$State)
```

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

Plotting this data:

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

```
p + labs(x="control", y="drug treatment")+ scale_colour_manual(values=c("green", "beige",
```



Going Further

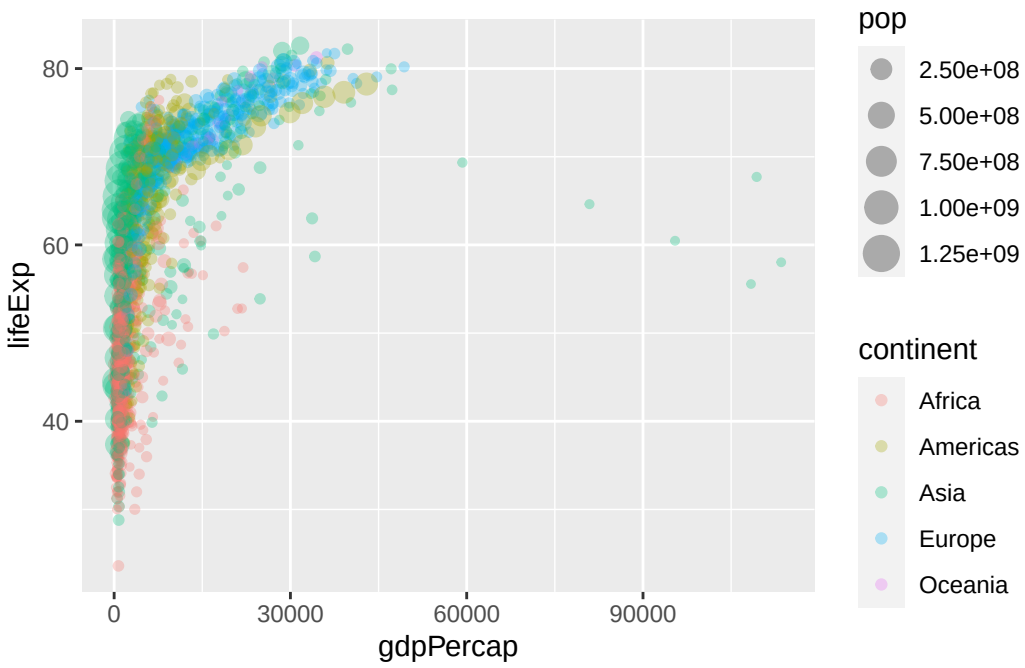
Here I need a slightly larger dataset.

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

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

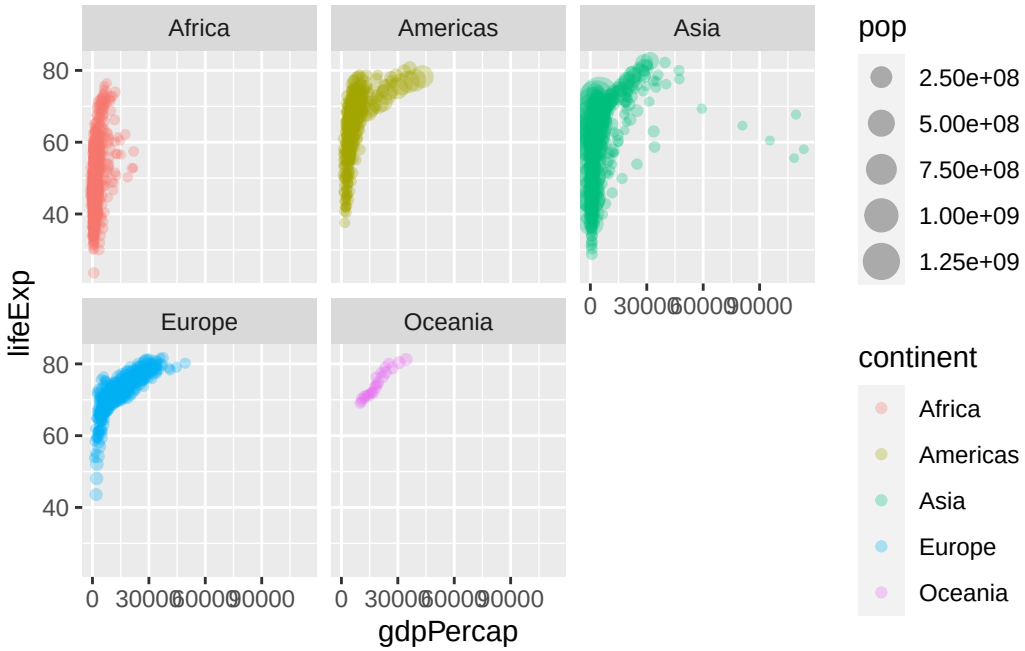
```
      country continent year lifeExp      pop gdpPercap
1 Afghanistan      Asia 1952  28.801  8425333  779.4453
2 Afghanistan      Asia 1957  30.332  9240934  820.8530
3 Afghanistan      Asia 1962  31.997 10267083  853.1007
4 Afghanistan      Asia 1967  34.020 11537966  836.1971
5 Afghanistan      Asia 1972  36.088 13079460  739.9811
6 Afghanistan      Asia 1977  38.438 14880372  786.1134
```

```
ggplot(gapminder) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.3)
```



A very useful layer to add sometimes is "faceting"

```
ggplot(gapminder) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.3) +
  facet_wrap(~continent)
```


Africa
Americas
Asia
Europe
Oceania
lifeExp
80
60
40
0
30000
60000
90000
gdpPercap
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09
continent
Africa
Americas
Asia
Europe
Oceania