

class 5: data viz with ggplot

Abraham PID A16735986

Table of contents

Background	1
Using different aes and geoms	6
Using different geoms	8
Running Code	11

Background

There are many graphics systems available in R. These include “base” R and tons of add on packages like **ggplot2**.

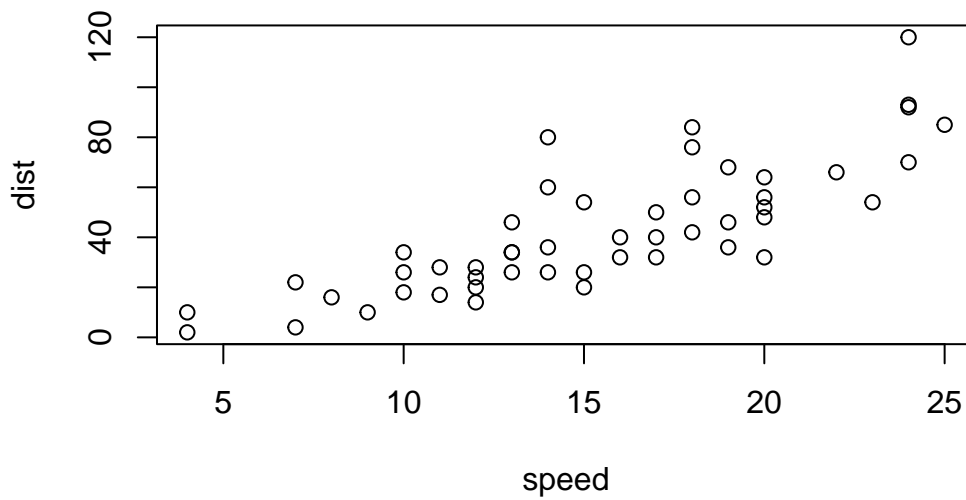
Let’s compare “base” and **ggplot2** briefly: We can use some example data that is built-in with R called `cars`:

```
head(cars)
```

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

In base R I can just call `plot()`

```
plot(cars)
```



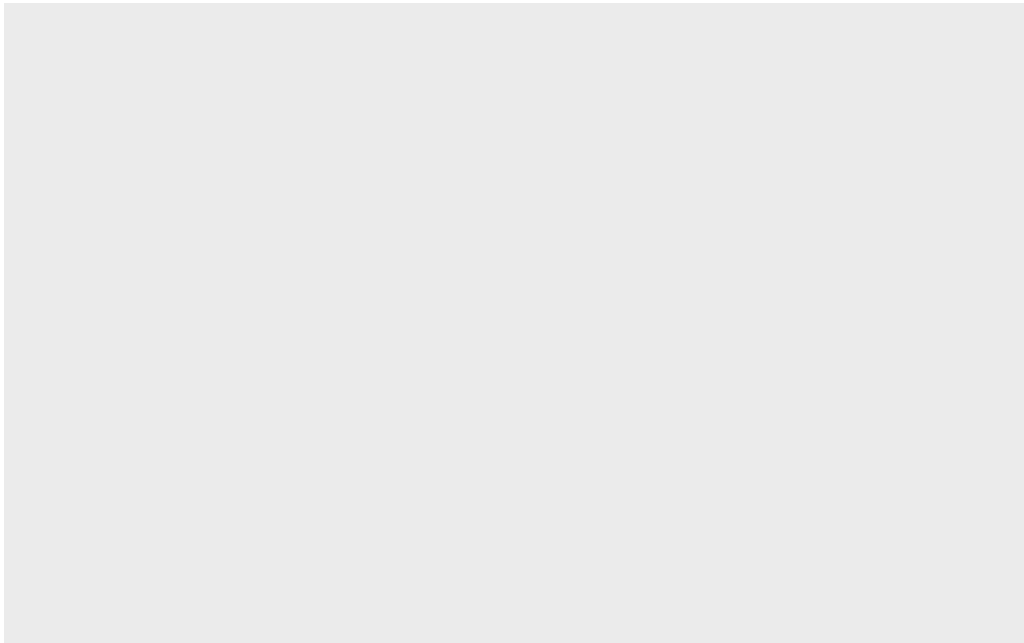
How can we do this with **ggplot2**.

First we need to install the packages. We do this `install.packages("ggplot2")`. I only need to do this once and then it will be available on my computer from then on.

Key point: I only install packages in the R console not within quarto docs or R scripts.

Before I use any add-on packages I must load it up with a call to `library()`

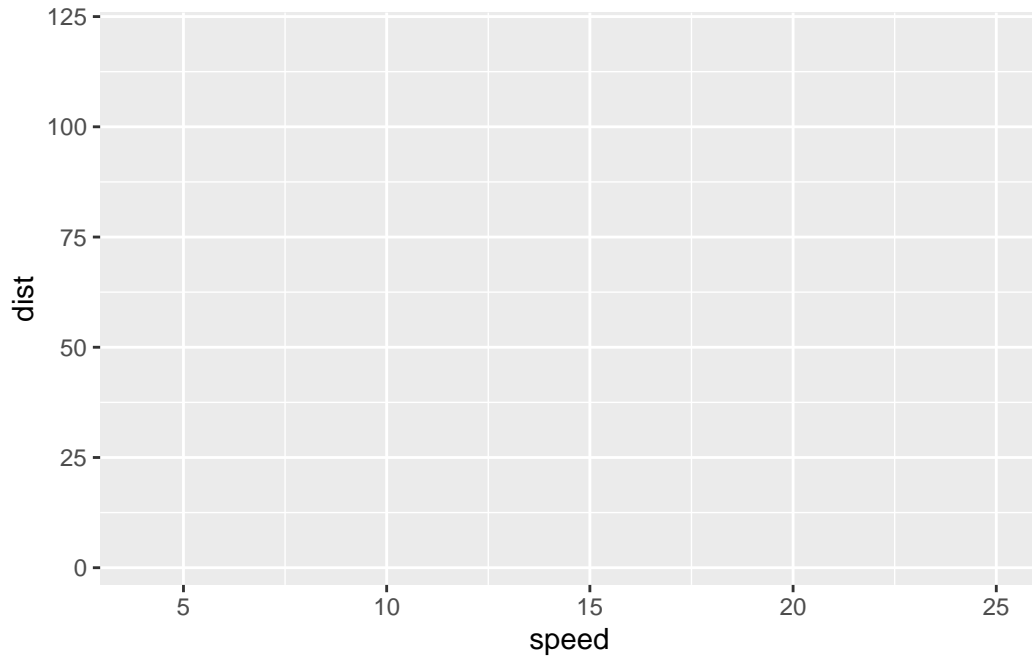
```
library(ggplot2)
ggplot(cars)
```



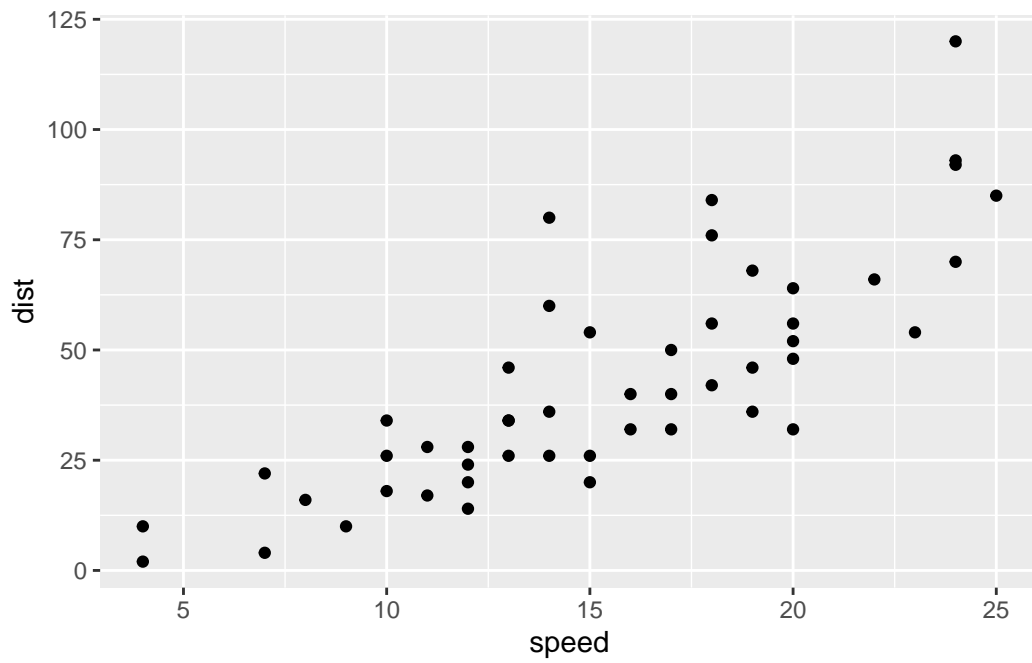
Every ggplot has at least 3 things:

- the **data** (in our case **cars**)
- the **aesthetics** (how the data map to the plot)
- the **geoms** that determine how the plot is drawn (lines, points, columns, etc.)

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



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



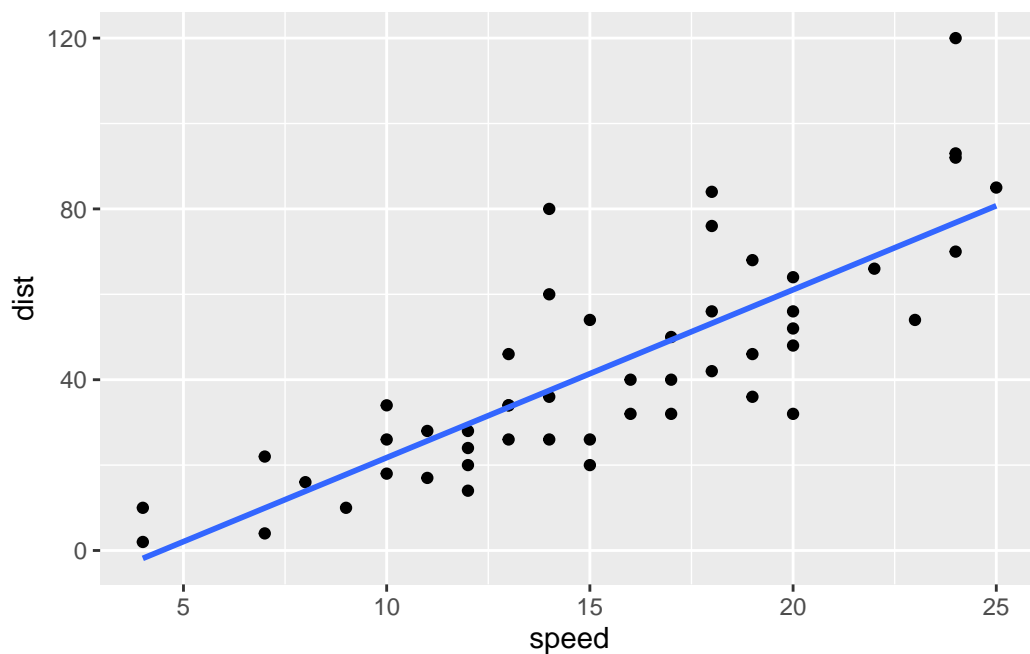
For “simple” plots ggplot is much more verbose than base R but the defaults are nicer and for complicated plots it becomes much more efficient and structured.

Q. Add a line to show the relationship of speed to stopping distance (i.e. add another “layer”)

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

p

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



I can always save any ggplot object (i.e. plot) and then use it for later for adding more layers
>Q. Add a title and subtitle to the plot

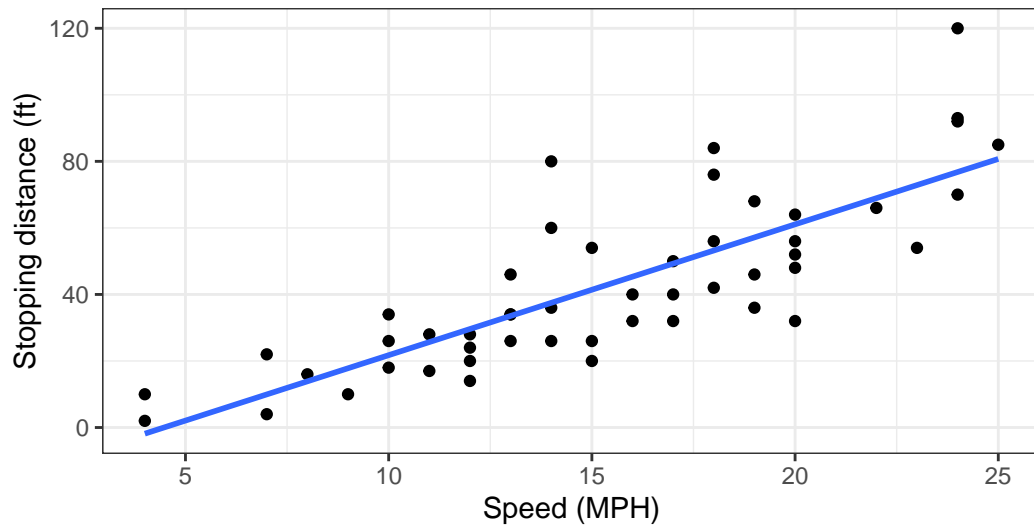
```
p + labs(title="My first ggplot",  
  subtitle = "stopping distance of old cars",  
  caption = "BIMM143",  
  x="Speed (MPH)",
```

```
y="Stopping distance (ft)" +  
theme_bw()
```

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

My first ggplot

stopping distance of old cars



BIMM143

Using different aes and geoms

```
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 wee dataset?

```
nrow(genes)
```

```
[1] 5196
```

Q. How many columns are there?

```
ncol(genes)
```

```
[1] 4
```

Q. What are the column names?

```
colnames(genes)
```

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

Q. How many “up” and “down” regulated genes are there?

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

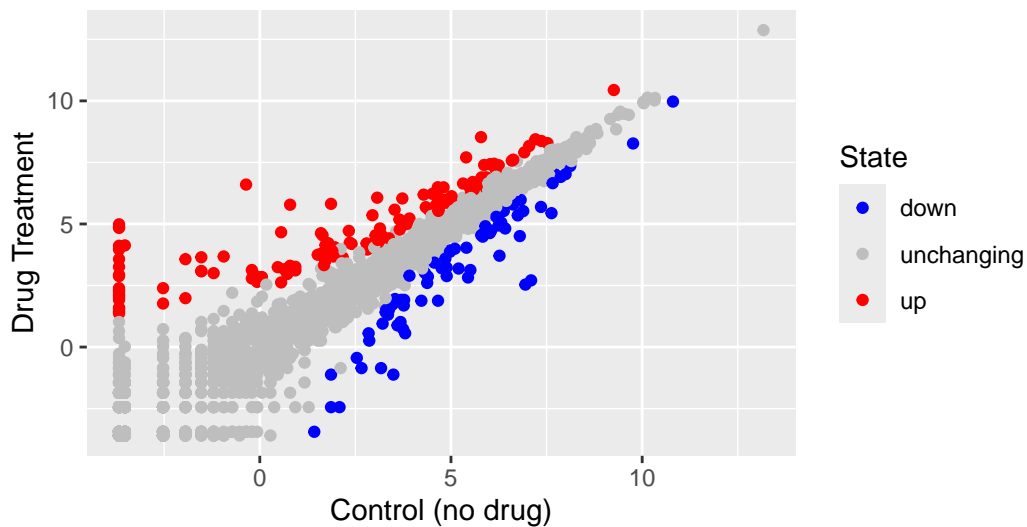
down	unchanging	up
72	4997	127

Q. Make a first plot of this.

```
ggplot(genes) +  
  aes(x=Condition1, y=Condition2, colour = State) +  
  scale_color_manual(values=c("blue","gray","red")) +  
  geom_point() +  
  labs(title="Gene Expression Changes Upon Drug Treatment",  
        subtitle = "Just another scatter plot made with ggplot",  
        caption="BIMM143",  
        x="Control (no drug)",  
        y="Drug Treatment")
```

Gene Expression Changes Upon Drug Treatment

Just another scatter plot made with ggplot



BIMM143

Using different geoms

Let's plot some aspects of the in-built `mtcars` dataset.

```
head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Q. Scatter plot of `mpg` vs `disp`

```
p1 <- ggplot(mtcars) +  
  aes(x=mpg, y=disp) +  
  geom_point()
```

Q. Boxplot of `gear` vs `disp`


```
p2 <- ggplot(mtcars) +
  aes(gear, disp, group=gear) +
  geom_boxplot()
```

Q. Barplot of carb

```
p3 <- ggplot(mtcars) +
  geom_bar(aes(x=carb))
```

Q. Smooth of disp vs qsec

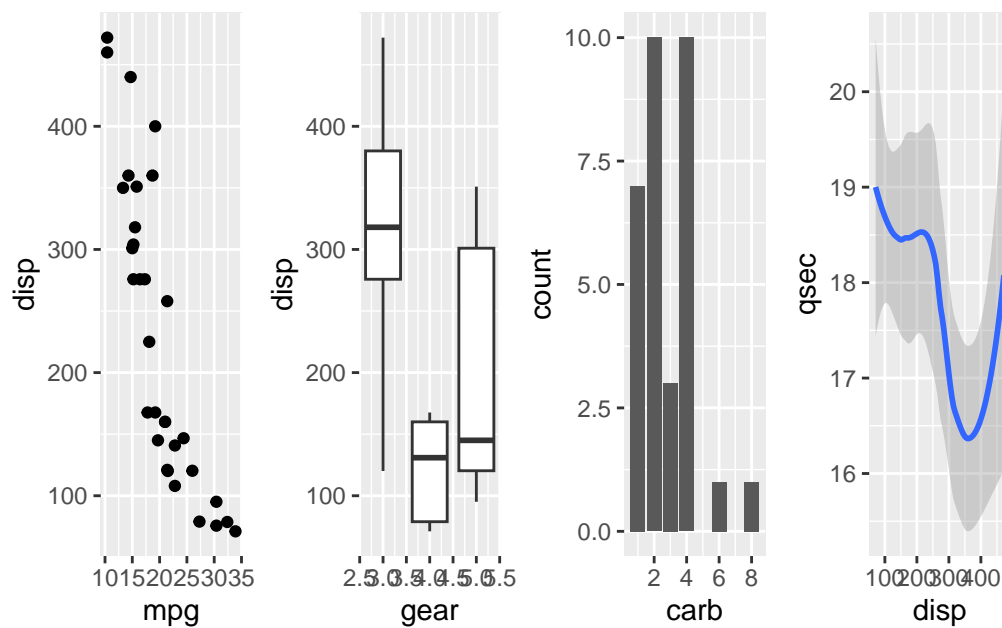
```
p4 <- ggplot(mtcars) +
  aes(dis, qsec) +
  geom_smooth()
```

I want to combine all these plots into one figure with multiple pannels. We can use the **patchwork** package to do this.

```
library(patchwork)
```

```
(p1 | p2 | p3 | p4)
```

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



```
ggsave(filename = "myplot.png", width=5, height=2)
```

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

```
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.tsv"
gapminder <- read.delim(url)
```

And a wee peak

```
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

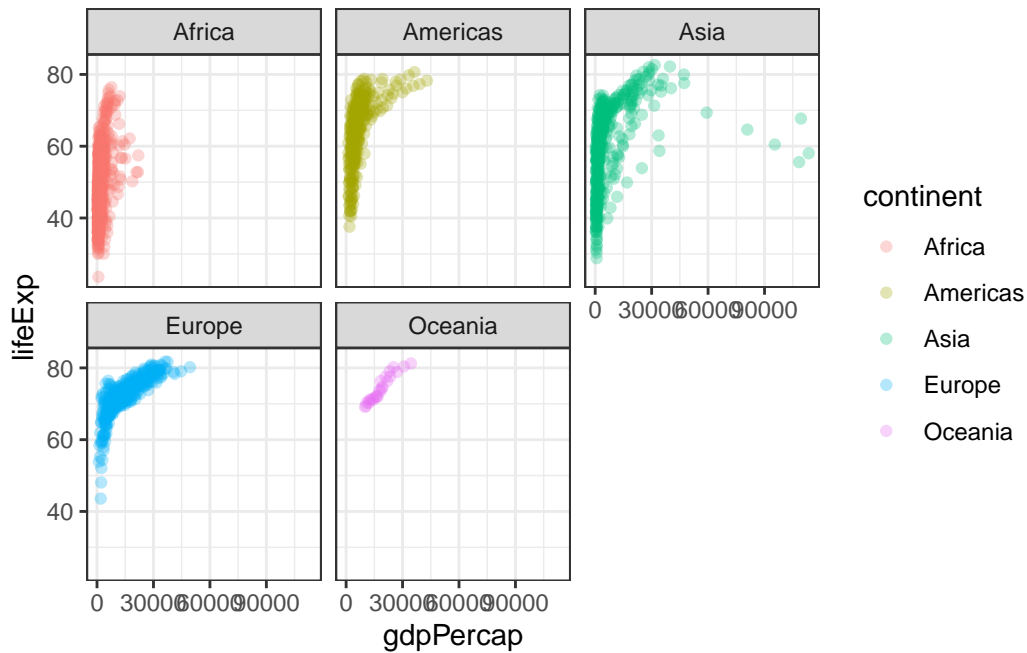
Q. How many countries are in this data set?

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

```
[1] 142
```

Q. Plot gdpPercap vs Life expectancy, color by continent

```
ggplot(gapminder) +
  aes(x=gdpPercap, y=lifeExp, colour =continent) +
  geom_point(alpha=0.3) +
  facet_wrap(~continent) +
  theme_bw()
```



Quarto *enables you to weave* together content and executable code into a finished document. To learn more about Quarto see <https://quarto.org>.

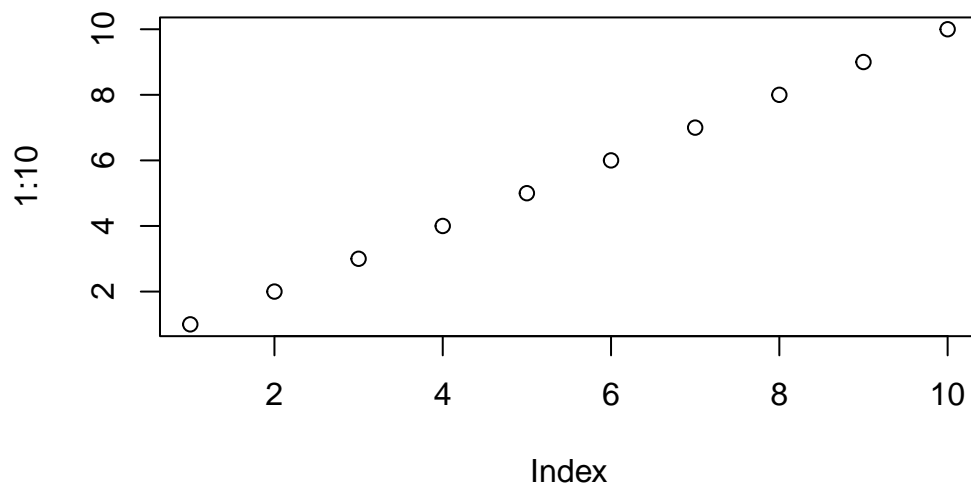
```
log(100)
```

```
[1] 4.60517
```

Running Code

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(1:10)
```



You can add options to executable code like this

```
[1] 4
```

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