Codealong: A trendline example using `mtcars` data

Plan

- We're going to use Emacs in Google Cloud Shell to create a linear regression model for the `mtcars` data frame, in order to draw a trendline through the data and gain some insight about the correlation of two features.
- This will show you how simple it is to do fairly ambitious things in R - we can do it in four lines of code. Unlike in Python, no packages have to be loaded.
- To do this, we need:
  1. A dataset (`mtcars`)
  2. A plotting function (`plot`)
  3. A linear regression model function (`lm`)
  4. A way to draw the trendline (`abline`)
- R is not installed in Google Cloud Shell, so we need to do that - fortunately, it is very fast. Unfortunately, we'll have to keep doing it[1].

Loading the dataset with `data`

- The `mtcars` dataset is built into "base R" which comes from CRAN at `cran.r-project.org`, the Comprehensive R Archive Network. To load it you can use the `data` function:

  ```
  data(mtcars)
  ```

- Can you explain the code block header arguments?

  1. `:session *R*` means that the code will be executed in an R console buffer named `*R*`. If it does not exist, it will be created.
  2. `:results none` means that no results will be printed to `stdout` (in this case: the screen) because `data` does not generate output.

[1] There is a way to install R locally but it's more complicated than `sudo apt install r-base`. We'll do this only if we have to.

- `data` produces no output but it loads `Nile` into R's current environment - how can we check what's in the environment?

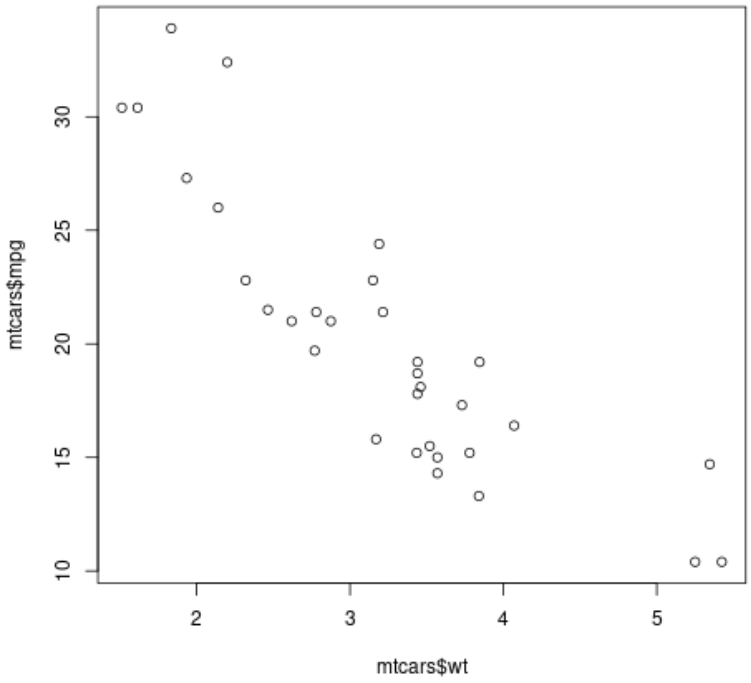
```
ls() # Output: "Nile"
```

```
[1] "mtcars"      "Nile"        "scatterplot" "trend"
```

Plotting `mtcars` data

- We already plotted `mtcars` miles-per-gallon against weight:

```
plot(mtcars$mpg ~ mtcars$wt)
```



- You can open the link with `C-c C-o` in a separate buffer, or in this file with `<F6>`.

- Notice the added header arguments:

    1. `:file mtcars.png` to save graphical output to a PNG file.
    2. `:results file graphics output` to print to file and to `stdout`.

Computing a trendline with `lm`

- How are the two variables correlated and what does that mean?

    Miles-per-gallon and car weight are negatively correlated, which means that they decrease together.

- A trendline is a linear function: how is it computed?

    The slope and intercept of the trendline are computed by fitting a line to the points of the plot. "Fitting" means minimizing a measure of distance between the line and

- In R, the `lm` ("linear model") function from the `stats` package will perform the fit - it needs two vectors of equal length, or (x,y) pairs:

```
trend = lm(mtcars$mpg ~ mtcars$wt)
trend
```

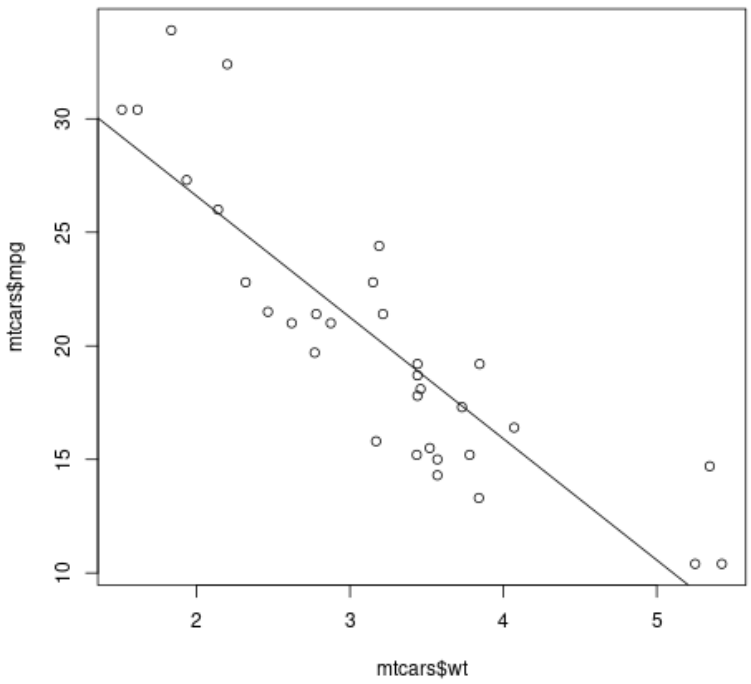
```
Call:
lm(formula = mtcars$mpg ~ mtcars$wt)

Coefficients:
(Intercept)    mtcars$wt
     37.285       -5.344
```

Plotting a trendline with `abline`

- To plot the trendline, we draw the linear graph in the former scatterplot using `abline` ("line with a and slope b") from the `graphics` package.

- For the code block, we need the graphics header arguments again. In R, a graphical `plot` object cannot be stored:

```
plot(mtcars$mpg ~ mtcars$wt)
abline(trend)
```



- As promised, we achieved the desired result in four lines of code (though because `mtcars` is built-in, we only needed three lines):

```
data(mtcars) # load dataset
plot(mtcars$mpg ~ mtcars$wt) # plot data
trend = lm(mtcars$mpg ~ mtcars$wt) # compute linear model
abline(trend)  # plot linear model
```

Further reading

The `mtcars` package is used in this vignette (= essay-like package documentation) of the `explore` package, showing many nice visualizations in the "Tidyverse" style of R rather than base R)