Data Science and Statistics in Research: unlocking the power of your data

Session 1.7: Is your data ready to analyse?

Introduction

In this practical we will introduce graphics in R. One of the more popular methods of creating graphics in R is to use the ggplot language for graphics which is implemented in the ggplot2 package. This way of creating graphs is flexible and modular, as you build a plot by layering different plot components. We will see examples of graphics using base R and the ggplot2 package.

Preliminaries

We need the following packages

- ggplot2 - Package to implement the ggplot language for graphics in R.
- raster - Package to do geographic data analysis and modelling

Make sure that these packages are downloaded and installed in R. We use the require() function to load them into the R library.

```
# Loading packages
require(ggplot2)
require(raster)
```

Data

To create plots, we will use the mtcars dataset available within R. This dataset consists of fuel consumption and other aspects of automobile design and performance for 32 cars from the 1973-74 Motor Trend US magazine. We load this dataset, using the data() function.

```
# Loading mtcars dataset
data(mtcars)
```

Make sure you are familiar with the contents of this dataset before continuing on with the rest of this practical, by typing ?mtcars into R.

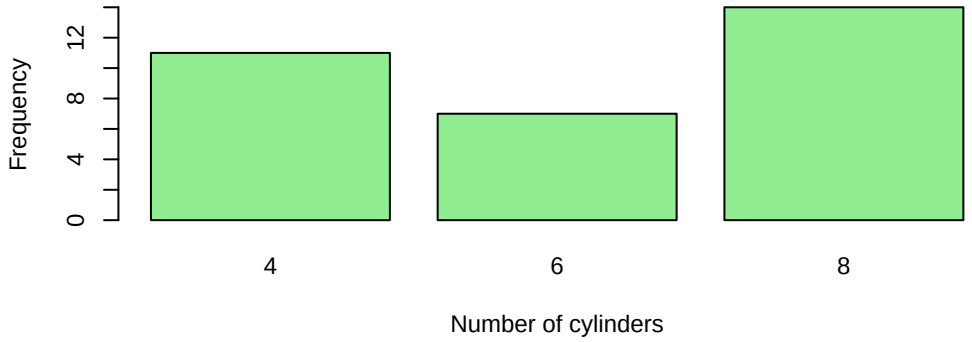
Bar Charts

Bar charts can be used to display frequencies. We are interested in the number of cars that have 4, 6 and 8 cylinders being tested in the mtcars dataset. Let's display this information as a bar chart.

Using base R graphics, we use the barplot() function. This function requires heights of each of the bars. We can find these by creating a frequency table using the table() function.

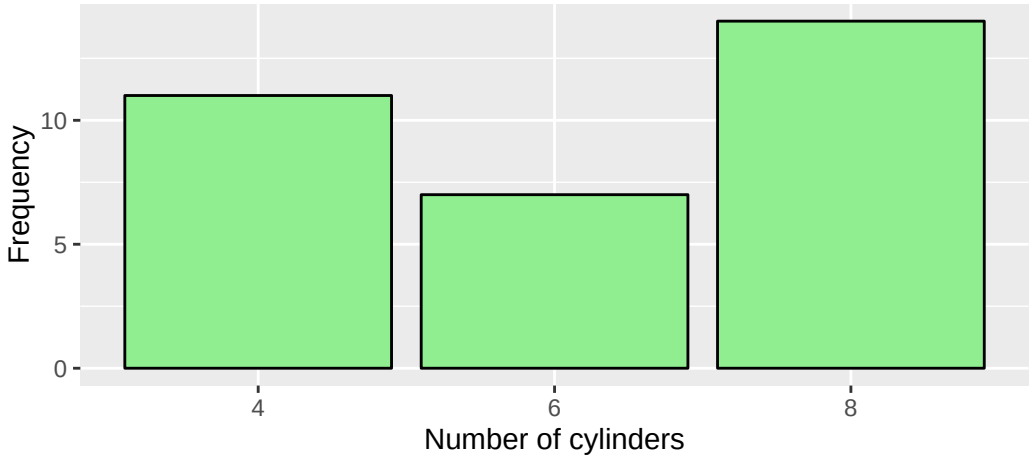
```
# Bar chart of cars by number of cylinders
barplot(table(mtcars$cyl), # Frequency table
        col='lightgreen', # Colour of bars
```

```
        xlab="Number of cylinders", # Label for the x-axis
        ylab="Frequency") # Label for the y-axis
```



Alternatively, we could use the `ggplot2` package. We use the `ggplot()` function to select the data that we want to display, then we use the `geom_bar()` to tell `R` we want to display the data as a bar chart.

```
# Bar chart of cars by number of cylinders using ggplot
ggplot(mtcars, aes(x = factor(cyl))) + # ggplot with the desired data
  geom_bar(fill='lightgreen',colour='black') + # Specifying a bar chart
  labs(x="Number of cylinders", y="Frequency") # Axes labels
```

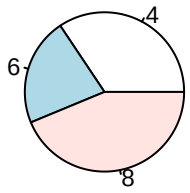


Pie Charts

Pie charts can be used to display variables where proportions are important. We are interested in the proportion of cars that have 4, 6 and 8 cylinders being tested in the `mtcars` dataset. Let's display this information as a pie chart.

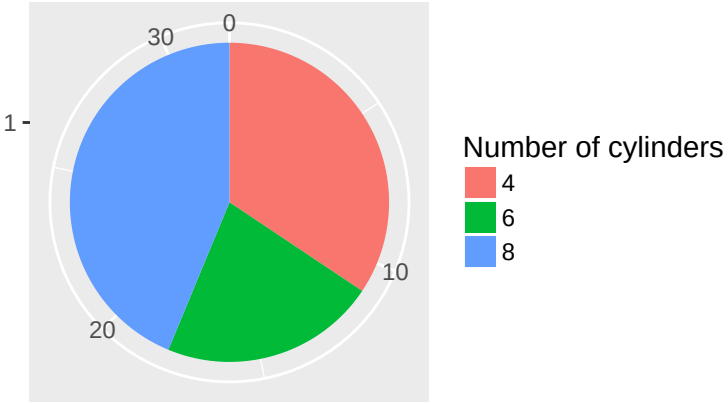
Using base `R` graphics, we use the `pie()` function. This function requires heights of each of the bars. We can find these by creating a frequency table using the `table()` function.

```
# Pie chart of cars by number of cylinders
pie(table(mtcars$cyl))
```



Alternatively, we could use the `ggplot2` package. To do this we create a bar chart as before similar to the one above but we add an extra option to say we want a pie chart.

```
# Pie chart of cars by number of cylinders using ggplot
ggplot(mtcars, aes(x = factor(1), fill=factor(cyl))) + # ggplot with the desired data
  geom_bar(width = 1) + # A bar chart
  coord_polar(theta = "y") + # Specifying a pie chart
  labs(x="",y="",fill='Number of cylinders') # Blank Axes labels
```

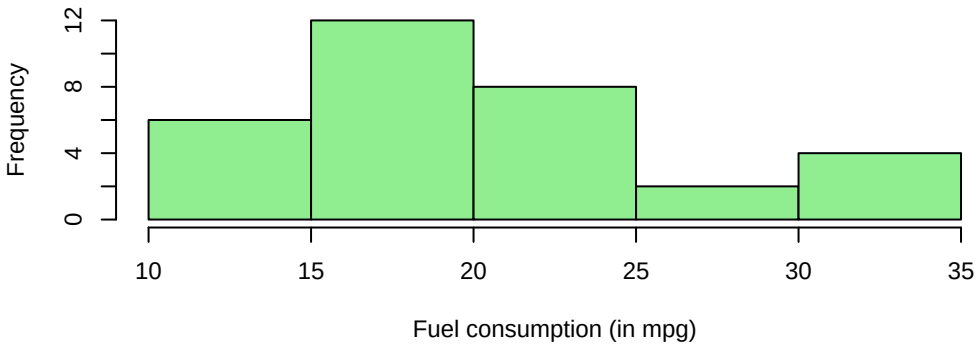


Histograms

Histograms can be used to display frequencies of quantitative variable using frequencies. We are interested in the distribution of fuel consumption (in miles per gallon) of cars tested in the `mtcars` dataset. Let's display this information as a histogram.

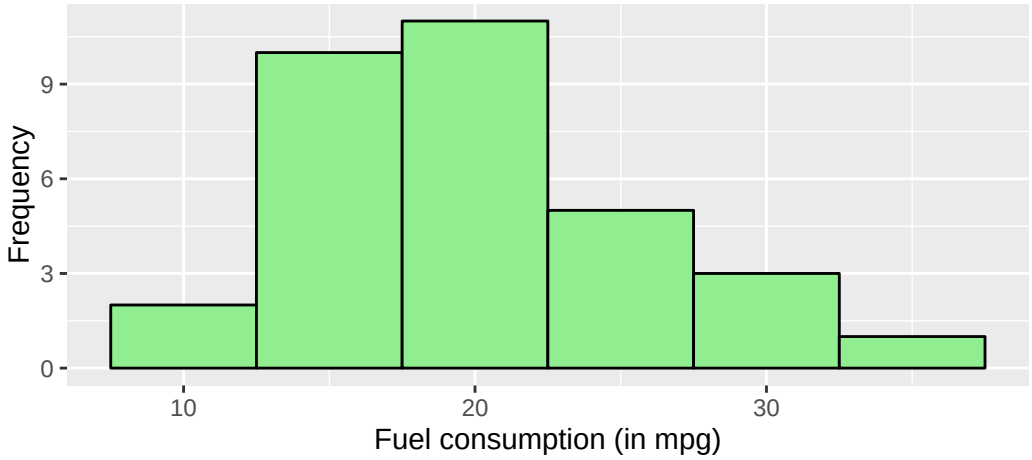
Using base `R` graphics, we use the `hist()` function. We simply enter the variable that we want to display in the histogram.

```
# Histogram of fuel consumption
hist(mtcars$mpg, # Fuel consumption
     col='lightgreen', # Colour of the bars
     xlab = 'Fuel consumption (in mpg)',  # x-axis label
     main = '')  # No plot title
```



Alternatively, we could use the `ggplot2` package. We use the `ggplot()` function to select the data that we want to display, then we use the `geom_histogram()` to tell `R` we want to display the data as a scatter plot.

```
# Histogram of fuel consumption using ggplot
ggplot(mtcars, aes(x=mpg)) +  # ggplot with the desired data
  geom_histogram(binwidth=5, fill='lightgreen',colour='black') + # Specifying bar chart
  labs(x="Fuel consumption (in mpg)", y="Frequency") # Axes labels
```



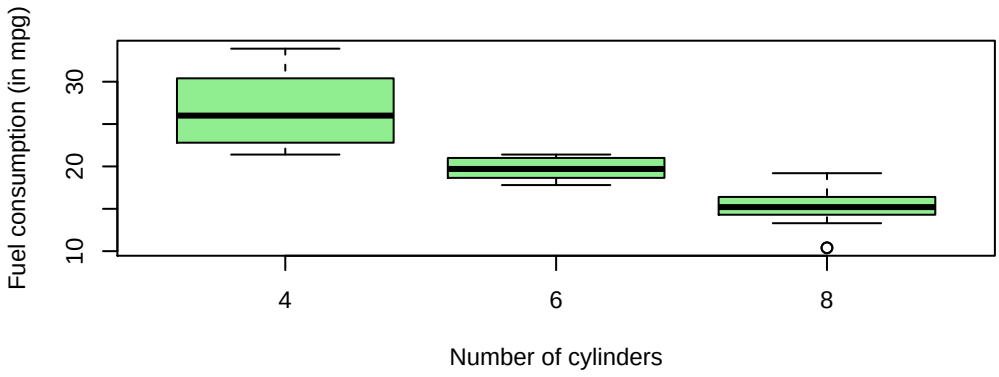
Box Plots

Box plots can be used to display median and variability in data. The central line is drawn at the median, and the box extends from the lower quartile to the upper quartile. The lines that extend from the box indicate three interquartile ranges, with any data outside this range shown using dots.

We are interested in the distribution of fuel consumption (in miles per gallon) of cars with 4, 6 and 8 cylinders in the `mtcars` dataset separately. Let's display this information as a boxplot.

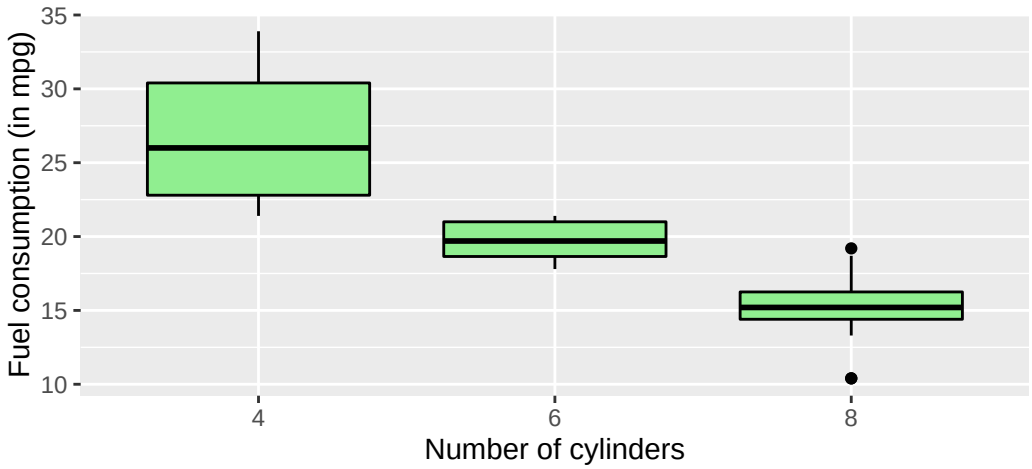
Using base `R` graphics, we use the `boxplot()` function. We simply enter two variables that we want to plot into the function.

```
# Boxplot of fuel consumption
boxplot(mtcars$mpg ~ mtcars$cyl, # Data
        col='lightgreen',# Colour of the boxplot
        xlab = 'Number of cylinders', # x-axis label
        ylab = 'Fuel consumption (in mpg)')  # y-axis label
```



Alternatively, we could use the `ggplot2` package. We use the `ggplot()` function to select the data that we want to display, then we use the `geom_boxplot()` to tell `R` we want to display the data as a box plot.

```r
# Boxplot of fuel consumption using ggplot
ggplot(mtcars, aes(x=factor(cyl),y=mpg)) + # ggplot with the desired data
  geom_boxplot(fill='lightgreen',colour='black') + # Specifying boxplot
  labs(x="Number of cylinders",y="Fuel consumption (in mpg)") # Axes labels
```

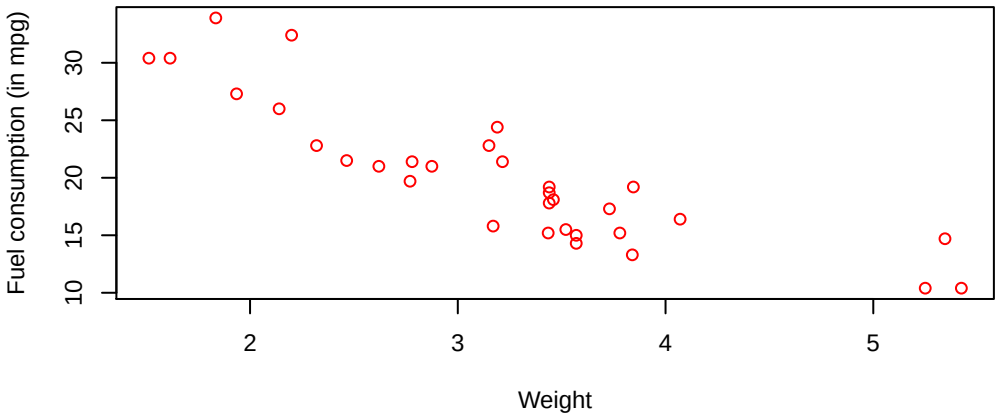


Scatter plots

Scatter plots can be used to display pairs of values of two quantitative variables, often to test for a correlation or association of the variables. We are interested in seeing the relationship between weight and fuel consumption (in miles per gallon) of cars in the `mtcars` dataset. Let's display this information as a scatter plot.

Using base `R` graphics, we use the `plot()` function. We specify the two variables to plot.

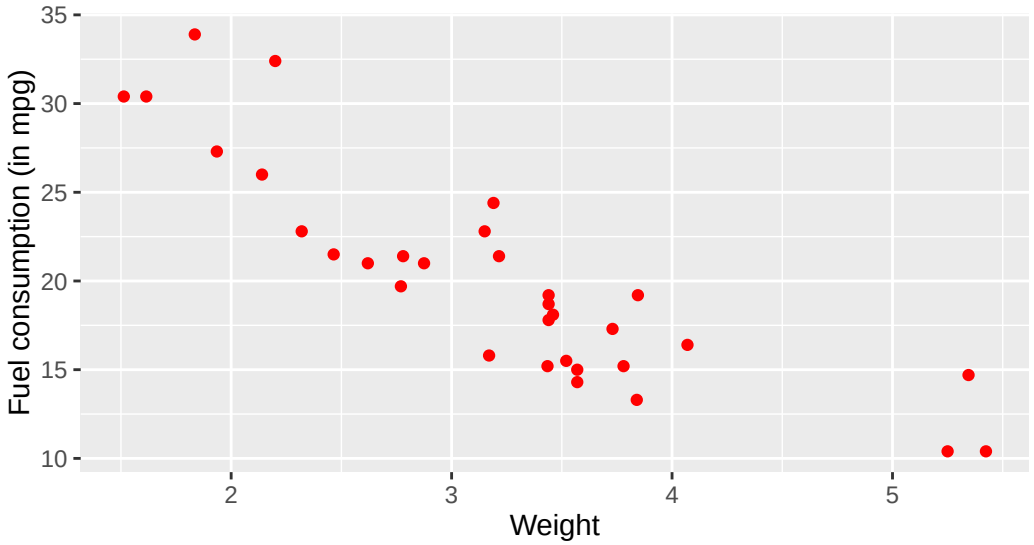
```r
# Scatter plot of cars weight by fuel consumption
plot(mtcars$wt, # Car weights
     mtcars$mpg, # Fuel consumption
     col='red', # Colour of the points
     xlab = 'Weight', # x-axis label
     ylab = 'Fuel consumption (in mpg)') # y-axis label
```



Alternatively, we could use the `ggplot2` package. We use the `ggplot()` function to select the data that we want to display, then we use the `geom_point()` to tell `R` we want to display the data as a scatter plot.

```r
# Scatter plot of cars weight by fuel consumption using ggplot
ggplot(data = mtcars, aes(x=wt,y=mpg)) + # ggplot with the desired data
```

```
  geom_point(colour='red') + # Specifying a scatter plot
  labs(x="Weight", y="Fuel consumption (in mpg)") # Axes labels
```

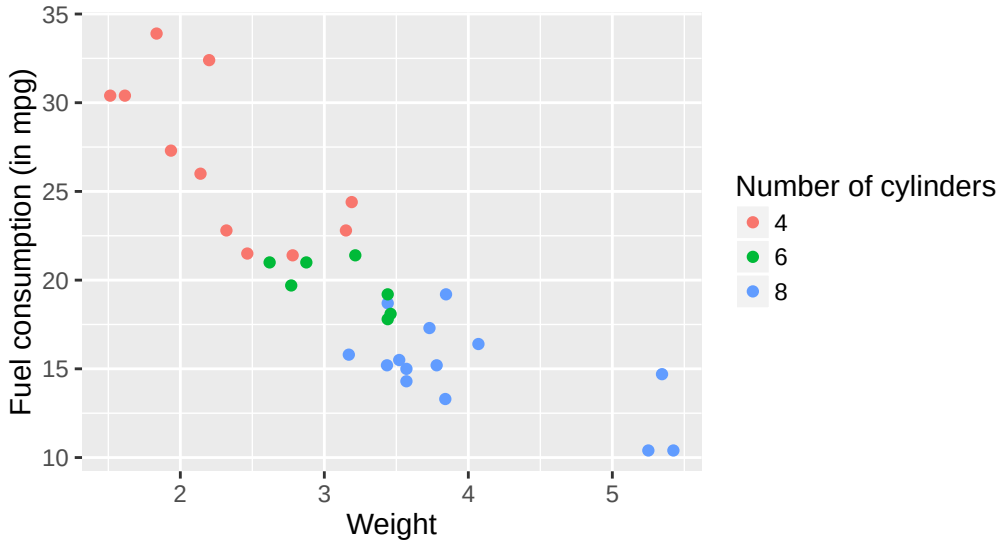


We can see that there is a negative relationship between weight and fuel consumption (in miles per gallon), so as weight increases, the fuel consumption decreasess.

Customising your plot

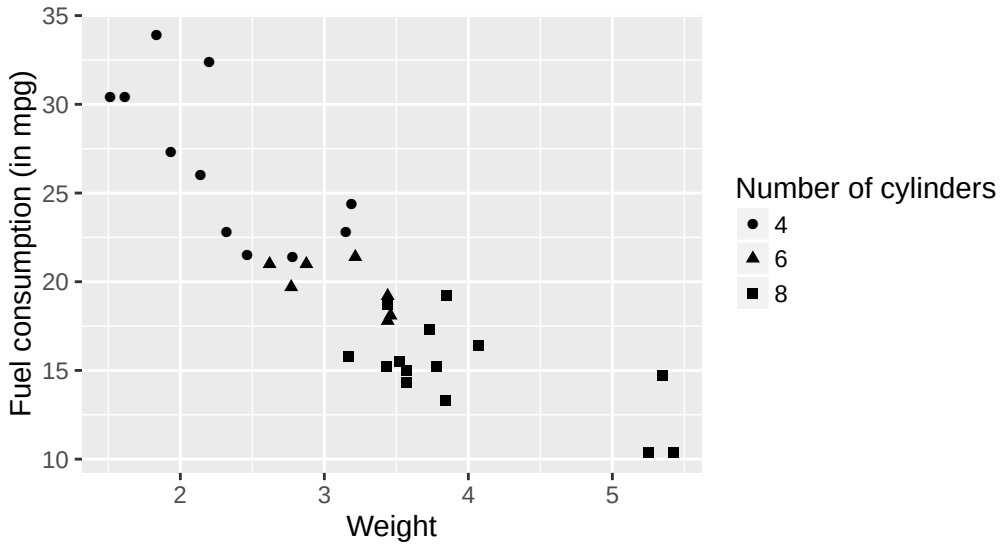
We created a scatter plot of weight against fuel consumption (in miles per gallon) and saw that there is a negative relationship between weight and fuel consumption. We are interested in seeing if this relationship is different between cars with 4, 6, and 8 cylinders. We can colour points in a scatter plot by a variable in our dataset by using the `colour` option in the aesthetics of the `ggplot()` function.

```
# Scatter plot of cars weight by fuel consumption using ggplot
ggplot(data = mtcars, aes(x=wt,y=mpg,colour=factor(cyl))) +
  # ggplot with the desired data
  geom_point() + # Specifying that we want it to be a scatter plot
  labs(x="Weight", y="Fuel consumption (in mpg)",colour='Number of cylinders') # Axes labels
```



Some journals restrict the use of colour in plots. Instead of adding colour to a scatter plot, we can change the style of the points in a scatter plot by a variable in our dataset. We do this using the `shape` option in the aesthetics of the `ggplot()` function.

```
# Scatter plot of cars weight by fuel consumption using ggplot
ggplot(data = mtcars, aes(x=wt,y=mpg,shape=factor(cyl))) +
  # ggplot with the desired data
  geom_point(aes(x=wt,y=mpg)) + # Specifying that we want it to be a scatter plot
  labs(x="Weight", y="Fuel consumption (in mpg)",shape='Number of cylinders') # Axes labels
```



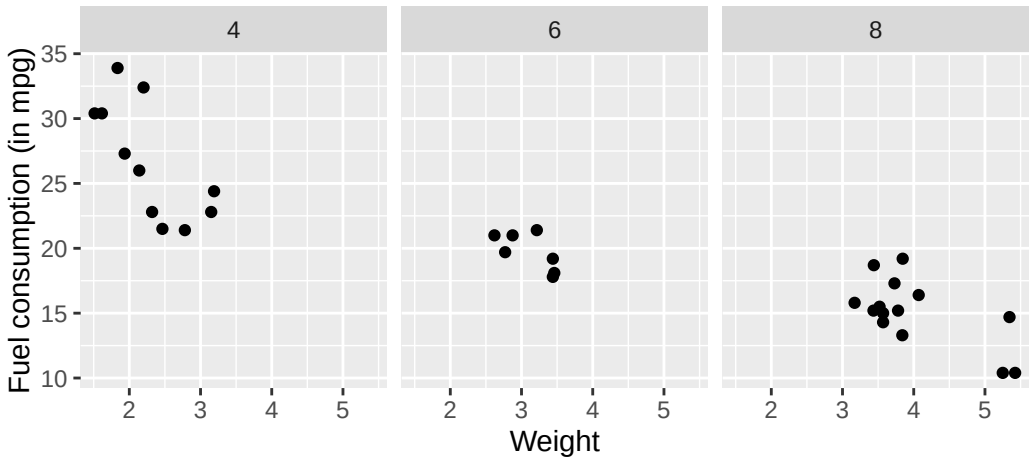
We see that while there is a relationship between weight and fuel consumption, it differs between 4, 6, and 8 cylinder cars.

Facetting

There may be times where you cannot distinguish patterns or relationships by simply plotting the entire dataset in one plot. It may be useful when trying to find these patterns if different the data from different groups are displayed in separate panels.

We can do this in a straightforward fashion using `ggplot2` package, but not using base `R` graphics. We created a scatter plot of weight against fuel consumption (in miles per gallon) and saw that there is a negative relationship between weight and fuel consumption. We want to see if the pattern differs between cars with 4, 6 and 8 cylinders. We add a call to the `facet_grid()` function to split the plot into different facets (or panels)

```
# Scatter plot of cars weight by fuel consumption using ggplot
ggplot(data = mtcars, aes(x=wt,y=mpg)) + # ggplot with the desired data
  geom_point(aes(x=wt,y=mpg)) + # Specifying that we want it to be a scatter plot
  labs(x="Weight", y="Fuel consumption (in mpg)") + # Axes labels
  facet_grid(. ~ cyl) # Facet split by columns
```



We see that while there is a relationship between weight and fuel consumption, it differs between 4, 6, and 8 cylinder cars.

Line Plots

To create plots, we will use the `mtcars` dataset available within `R`. The `economics` dataset contains information about the US economy across time. We load this dataset, using the `data()` function.

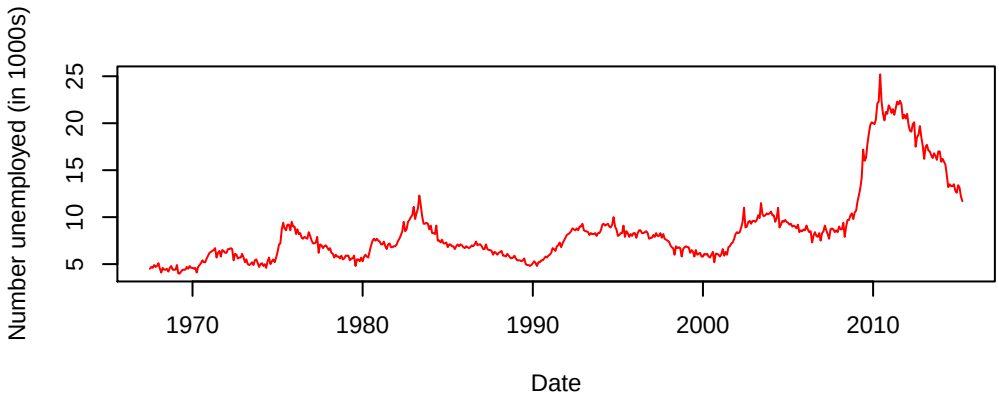
```
# Loading economics dataset
data(economics)
```

Make sure you are familiar with the contents of this dataset before continuing on with the rest of this practical, by typing `?economics` into `R`.

Line plots can be used to show values of one or more variables measured over time, connected by a line. We are interested in seeing the number of people unemployed changing over time. The dataset `economics` in `R` contains this information. Let's display this information as a line plot.

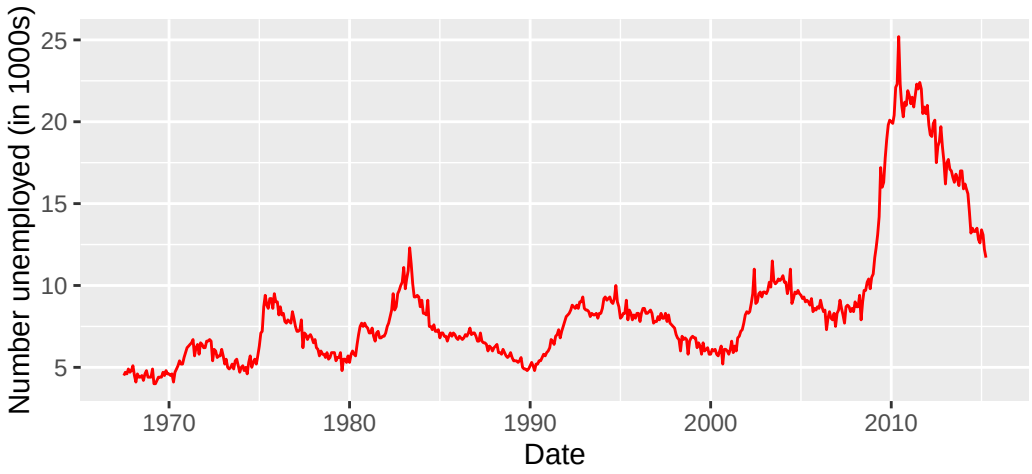
Using base `R` graphics, we use the `plot()` function. We specify two variables (i) the time variable and (ii) the variable we want to plot over time. We also need to specify that we want a line plot not a scatter plot.

```
# Line plot of unemployment
plot(economics$date, # Date
     economics$uempmed, # Number unemployed
     type='l', # Line plot
     col='red',
     xlab = 'Date', # x-axis label
     ylab = 'Number unemployed (in 1000s)') # y-axis label
```



Alternatively, we could use the `ggplot2` package. We use the `ggplot()` function to select the data that we want to display, then we use the `geom_line()` to tell `R` we want to display the data as a line plot.

```
# Line plot of unemployment using ggplot2
ggplot(data=economics, aes(x=date,y=uempmed)) + # ggplot with the desired data
  geom_line(colour='red') + # Specifying a bar chart
  labs(x='Date',y='Number unemployed (in 1000s)') # Axes labels
```



Maps

To create maps, we use something called 'shapefiles'. Shapefiles contain location, shape, and attributes of geographic features such as country borders. A website called GADM is a repository for shapefiles. These can be downloaded into `R` using the `getData()` function. This function needs three inputs

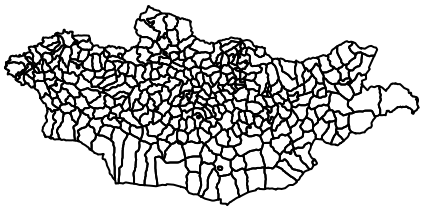
- the website with the files you wantt to download (Here it's GADM)
- the country ISO3 code, (for example, Mongolia is MNG)
- the type of borders (0 - Gives a country's external borders, 1 - gives a country split into regions and 2 - gives a country split into smaller sub-regions).

```
# Downloading shapefiles for Mongolia
Mongolia <- getData(name='GADM',    # Download from GADM
                    country='MNG', # Country ISO3 code
                    level=2)       # Level of borders
```

More information on this function can be found by typing `?getData` into `R`.

Maps can be used to display information and variation over space. We are interested in makign maps of Mongolia. We can plot the boundaries of Mongolia using the `plot()` function.

```
# Plotting borders for Mongolia
plot(Mongolia)
```



This plot does not provide us with any information so we use simulated data to allocate a number to an area which we will use to create a colour scheme. If you have your own data, you can also use that. To plot this simulated data we use the `spplot()` function.

```
# Adding random numbers to each Mongolia region
Mongolia@data$rnum <- rnorm(nrow(Mongolia@data),0,2)

# Cut off points for legend of plot
range <- seq(min(Mongolia@data$rnum)-0.01, max(Mongolia@data$rnum)+0.01, length.out=5)

# Creating Map
spplot(obj  = Mongolia,              # Spatial object to be plotted
       zcol = c("rnum"),            # Choice of the column  the object you are plotting.
       xlab = "Easting",            # Label of the x column
       ylab = "Northing",           # Label of the y column
       at   = range,                # Break points for colours
       col.regions = hsv(0.6, seq(0.2, 1, length.out=10), 1)) # Create a set of colours
```