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

Session 1.5: Initial data analysis

## Introduction

In this session we will perform an initial data analysis. We will initially use the cost of drugs data, to show how commonly used descriptive statistics needed for an initial data analysis are caclulated. We will then apply these techniques to an additional dataset containing car attributes.

## Descriptive Statistics

Descriptive statistics are calculated from a sample. Various statistics are routinely used, for both qualitative and quantitative data.

### Summarising quantitative data

Quantitative data are numeric values and can be of two types: (i) discrete, or (ii) continuous. To demonstrate how to summarise quantitative data, we use the data set `drugcosts` that contains the total costs of drugs (in Burundi francs) received by 84 adults aged 20-29 visiting five different health centers in the Myinga province of Burundi in 1991-2.

```
# Drug Costs in a vector
drugcost <- c(2.0,4.0,6.2,8.0,8.0,8.0,12.0,15.0,15.0,18.0,18.2,20.0,20.0,20.0,21.0,22.0,
              24.2,27.0,27.0,27.0,28.0,29.7,29.7,29.7,29.7,29.7,30.0,30.0,31.0,41.4,42.3,
              45.4,45.4,45.4,45.4,45.4,45.4,45.4,45.4,45.4,49.4,50.8,51.4,53.4,56.0,57.0,
              57.4,59.0,59.4,60.0,61.5,65.4,65.4,65.4,65.4,67.0,90.8,92.0,94.0,94.0,94.0,
              94.0,94.0,94.0,94.7,105.0,125.0,126.0,130.0,130.0,130.0,151.2,160.0,177.0,
              187.0,187.0,194.4,194.4,194.4,212.4,213.0,233.0,267.0,320.4)
```

We will calculate the following descriptive statistics for this data:

- sample size
- mean
- median
- variance
- standard deviation
- minimum
- maximum
- range
- lower quartile
- upper quartile
- inter quartile range.

To find the number of different drug costs we can calculate the length of `data` using the `length()` function.

```
# Sample size
length(drugcost)
[1] 84
```

To find the mean of different drug costs within the data set we can use the `mean()` function.

```
# Calculate the mean drug cost.
mean(drugcost)
[1] 75.05952
```

We can also use the `sum()` and `length()` function to calculate the mean and verify the results obtained using the `mean()` function.

```
# Calculate the mean drug cost.
sum(drugcost)/length(drugcost)
[1] 75.05952
```

To find the median of different drug costs within the data set we can use the `median()` function.

```
# Calculate the median drug cost.
median(drugcost)
[1] 51.1
```

To find the variance of different drug costs within the data set we can use the `var()` function.

```
# Calculate the variance of drug costs.
var(drugcost)
[1] 4494.914
```

To find the standard deviation of the different drug costs within the data set we can either take the square root of the variance or use the `sd()` function.

```
# Calculate the standard deviation of drug costs.
sqrt(var(drugcost))
[1] 67.04412
sd(drugcost)
[1] 67.04412
```

To find the least expensive drug we can use the `min()` function.

```
# Minimum drug cost.
min(drugcost)
[1] 2
```

To find the most expensive drug we can use the `max()` function.

```
# Maximum drug cost.
max(drugcost)
[1] 320.4
```

To find the range of the drug costs we can subtract the least expensive drug from most expensive.

```
# Drug cost range
max(drugcost) - min(drugcost)
[1] 318.4
```

To find the lower quartile we can use the `quartile()` function. Here we input the data set followed by the quartile as a decimal.

```
# Lower quartile of the drug costs.
quantile(drugcost, probs = 0.25)
   25%
29.275
```

Similarly, to find the upper quartile.

```
# Upper quartile of the drug costs.
quantile(drugcost, probs = 0.75)
75%
 94
```

To find the inter quartile range (IQR), we can use the `IQR()` function.

```
# Interquartile range of the drug costs.
IQR(drugcost)
[1] 64.725
```

Some of these statistics can be calculated the `summary()` function.

```
# Summarising the drug costs.
summary(drugcost)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.00   29.28   51.10   75.06   94.00  320.40
```

This function calculates the minimum, lower quartile, median, mean, upper quartile and the maximum, but not variance or standard deviation.

More information on any functions used here can be found by using the help operator `?` or function `help()`.

## Summarising qualitative data

Qualitative data take on distinct values or classes and are of three types; (1) categorical, (2) ordered categorical and (3) binary. To demonstrate how to summarise qualitative data, we use another vector of data, which we call the drug class. This gives us the type of drug that is being sold; A - pain relievers, B - antibiotics and C - antiviral.

```
# Drug Class in a vector
drugclass <- c("A","C","C","C","A","A","A","B","A","B","B","B","B","A","A","A","B",
               "A","A","A","C","A","B","B","C","A","C","A","C","A","B","C","C","B",
               "B","A","C","B","A","B","B","A","A","C","B","B","A","B","C","C","B",
               "A","C","B","C","A","B","C","C","C","A","A","A","C","B","A","A","B",
               "C","A","B","A","A","B","C","C","A","A","C","C","A","B","C","C")
```

The elements of `drugclass` corresponds to the elements of `drugcost`, for example, the first element of `drugclass` corresponds to the first element of `drugcost`.

In Session 1.4, we summarised qualitative data by counting the numbers in each category and calculating proportions in each group. We can use the `table()` function to calculate the numbers in each group. We then enter the numbers into the `prop.table()` function to calculate proportions, and multiply by 100 to get percentages.

```
# Calculating the number of each drug type
table(drugclass)
drugclass
 A  B  C
33 25 26

# Number of each drug type
prop.table(table(drugclass))
drugclass
        A         B         C
0.3928571 0.2976190 0.3095238

# Number of each drug type as a percentage
100*prop.table(table(drugclass))
drugclass
       A        B        C
39.28571 29.76190 30.95238
```

## Categorising quantitative data

Quantitative variables can have their values grouped into classes and presented as categorical variables. It is often useful to do so to highlight important groups or clusters in your data. In `R` we can use the `cut()` function to categorise the drug costs into three groups: (i) 1-100, (ii) 101-200 and (iii) 201+.

```
# Categorising the drugcost: (i) Low (1-50), (ii) Medium (50-100),
# (iii) High (100-200) and (iii) Very high (+201).
drugcostcat <- cut(drugcost, # Data to be categorised
                   breaks = c(0,50,100,200,max(drugcost)), # Breaks for the categories
                   labels = c('Low','Medium','High','Very High')) # Category names
```

More information about the `cut()` function can be found by typing `?cut` into `R`. Now that we have created categories we can summarise them in the same way that we would qualitative data

```
# Calculating the number in each group
table(drugcostcat)
drugcostcat
      Low    Medium      High Very High
       41        24        14         5

# Proportion of each drug type
prop.table(table(drugcostcat))
drugcostcat
       Low     Medium       High  Very High
0.48809524 0.28571429 0.16666667 0.05952381
```

```
# Proportion of each drug type as percentage
100*prop.table(table(drugcostcat))
drugcostcat
      Low    Medium      High Very High
48.809524 28.571429 16.666667  5.952381
```

**Activities**

- Use the `data.frame()` function to create a dataset containing drugcosts, drugclass and drugcostcat.
- Recalculate the statistics used above using this dataframe rather than vectors.

## Initial data analysis on `mtcars` dataset

To demonstrate how to perform an initial data analysis, we will use to the 'Motor Trend Car Roads Tests' dataset. This dataset was extracted from the 1974 Motor Trend US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973–74 models). We can load this dataset, which is stored in `R`, using the `data()` function.

```
# Call the mtcars dataset into R
data(mtcars)
```

To determine what types of variables are in the dataset we need to study the dataframe. We can view the variables in the dataset by using the `str()` function and view the first six observations of the dataset by using the `head()` function

```
# Viewing the structure of the dataset
str(mtcars)
'data.frame':   32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
 $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...

# Viewing the first 6 rows of the 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
```

We can see that this dataset contains the following variables:

- `mpg` - Miles/(US) gallon
- `cyl` - Number of cylinders
- `disp` - Displacement (cu.in.)
- `hp` - Gross horsepower
- `drat` - Rear axle ratio
- `wt` - Weight (1000 lbs)
- `qsec` - 1/4 mile time
- `vs` - Engine type (0 = v engine, 1 = straight engine)
- `am` - Transmission (0 = automatic, 1 = manual)
- `gear` - Number of forward gears
- `carb` - Number of carburetors.

More information about the `str()` and `head()` functions can be found by typing `?str` or `?head` into `R`. More information on the `mtcars` dataset can be found by typing `?mtcars` into `R`.

There are five types of data; (i) categorical, (ii) ordered categorical, (iii) binary, (iv) discrete or (v) continuous.

Let's look at the second variable, `cyl`. From inspection, we see that this might be either a discrete variable (as it only takes integer values) or an ordered categorical variable (as it has a natural order and takes distinct classes).

Let's extract all of the data to be sure.

```
# Extracting the number of cylinders
mtcars$cyl
 [1] 6 6 4 6 8 6 8 4 4 6 6 8 8 8 8 8 8 4 4 4 4 8 8 8 8 4 4 4 8 6 8 4
```

So we can see the number of cylinders can only take values of '4', '6' or '8'. This means that `cyl` is an ordered categorical variable

**Activity**

- Go through each of the remaining variables in the `mtcars` datasets and determine whether each of the variables are either (i) categorical, (ii) ordered categorical, (iii) binary, (iv) discrete or (v) continuous. You should use the output from the `str()` and `head()` functions above and extract the data from the dataset to aid your decision.

Suppose a car company wants to see if there is a difference in the fuel consumption (in miles per gallon) between automatic and manual cars. We need to decide the type of descriptive statistics to use. The choice is down to you, and depends on the information that you want your readers to see.

For continuous variables, let's calculate the mean, median and standard deviation. The continuous variable we are interested in is fuel consumption in miles per gallon (`mpg`), so we calculate these statistics.

```
# Calculating the mean of miles per gallon
mean(mtcars$mpg)
[1] 20.09062

# Calculating the median of miles per gallon
median(mtcars$mpg)
[1] 19.2

# Calculating the standard deviation of miles per gallon
sd(mtcars$mpg)
[1] 6.026948
```

For categorical, ordered categorical and binary data, let's calculate the proportions of cars in each group. Proportions are useful to see if the numbers in each group are even or uneven. The categorical variable we are interested in is whether the car is automatic or not (`am`), so we calculate the proportion of cars (as a percentage) in each of these classes.

```
# Proportion of automatics and manual cars as percentage
100*prop.table(table(mtcars$am))

     0      1
59.375 40.625
```

As we are interested to see if there is a difference in the fuel consumption (in miles per gallon) between automatic and maunal cars, let's calculate the mean and median fuel consumption in both automatic and manual cars separately. We do this by using the `aggregate()` function.

```
# Calculating the mean miles per gallon
# for automatic and manual cars separately
aggregate(mpg ~ am,    # Splitting mpg by automatic/manual
          data=mtcars, # Dataset to summarise
          FUN = mean)  # Summary statistic to use
  am      mpg
1  0 17.14737
2  1 24.39231

# Calculating the median miles per gallon
# for automatic and manual cars separately
aggregate(mpg ~ am,    # Splitting mpg by automatic/manual
          data=mtcars, # Dataset to summarise
          FUN = median)  # Summary statistic to use
  am  mpg
1  0 17.3
2  1 22.8
```

We can see that automatic cars have achieve less miles per gallon to that of the manual.

**Activity**

- Suppose the same car company wants to know if the quarter mile time of the cars (`qsec`) are different depending on the number of cylinders (`cyl`) the car might have. Choose whichever descriptive statistics you wish to use and perform your own initial data analysis.