Summer/Winter/Semester2	Introduction to R	2016

This self-study tutorial is an introduction to R.
It should be completed at home before your first tutorial lab.

Why learn R/R Studio?

Throughout MATH1005 we use a versatile statistical language called **R**, which provides a wide and ever-increasing suite of statistical and graphical techiques.

R is a programming language, which means it is not menu-driven. All commands are case sensitive and are written and executed in the console window at the prompt. However, there are certain tasks which can be implemented through the menus, like installing new packages. Data in R are organised as named structures. We will mainly deal with the simplest such structures: vectors and matrices. They can be numerical data (like height and weight) or categorical factors (like gender and type of diet). R treats factors and numerical data differently, and can combine them in a data frame. Each vector must contain elements of only one type, while a data frame can contain columns of different types.

R Studio is an integrated user interface for R. When you open up R Studio, it automatically runs R.

1. Overview of R

To get an overview of how R works, complete this excellent free online tutorial: tryr.codeschool.com/ It takes about an hour but will give you an excellent introduction to R.

Another good sumamry is: https://learnxinyminutes.com/docs/r/

2. Download R/R Studio for home usage (free)

R and R Studio are available in the Carslaw computer labs. However we recommend you download both R and R Studio onto your home computer so that you can do your Reports at home, and for use in other subjects.

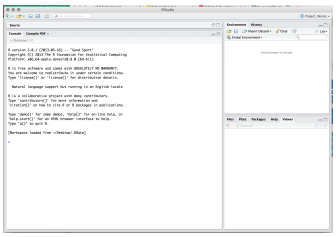
• Download R from the CRAN (Comprehensive R Archive Network) website:
PC: cran.r-project.org/bin/windows/base/
Mac: cran.r-project.org/bin/macosx/

• Download RStudio: crstudio.com/products/rstudio/download/

Alternatively, you can use RFiddle: http://www.r-fiddle.org/

3. The layout of R Studio

Type commands into the main console window. Note the 'Help' Tab which allows you to look up commands.



4. Uploading data into RStudio

There are many ways to upload data into RStudio, depending on the size of the data and in what form it is stored.

Method1: Enter data manually (for small datasets)

```
x=c(1.1,2.3,4.5,6.7,3.2)
```

Note that the vector `x` is now listed in the Environment.

To see what is stored inside `x`, type the name of the vector

```
x

## [1] 1.1 2.3 4.5 6.7 3.2
```

Method2: Copy and paste the data from a file

- At the R prompt enter `y=scan()` (the prompt changes to "`1:`" indicating that it is waiting for the 1st data point).

```
y=scan()
```

- Right click to copy and paste these numbers: 1 3 5 7 8 8
- Click next to the "`1:`" prompt, paste the numbers and hit Enter twice.

Method3: Import data from the Internet

- For `.csv` file:

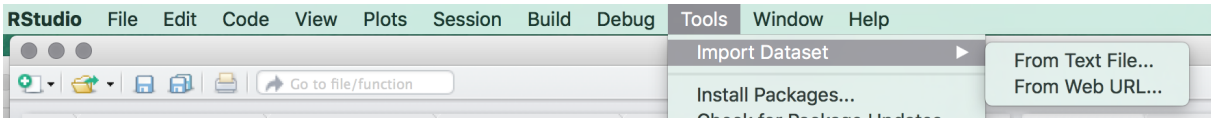
```
Road <- read.csv("http://www.maths.usyd.edu.au/u/UG/JM/MATH1005/r/StatsData/
                 2016Fatalities.csv")
```

- For `.txt` file:

```
Mice <- read.table("http://www.maths.usyd.edu.au/u/UG/JM/MATH1005/r/StatsData/Mice.txt")
```

```
Mice=scan(file=url("www.maths.usyd.edu.au/u/UG/JM/MATH1005/r/StatsData/Mice.txt"))
```

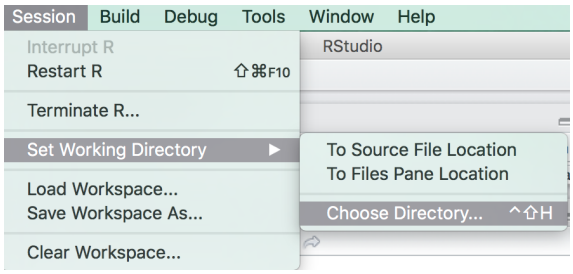
Method4: Import a file from a directory



Method5: Import a file from a directory

- Download the `2016Fatalities.csv` file into a directory.
  The data files are found here: http://www.maths.usyd.edu.au/u/UG/JM/StatsData.html

- Change the RStudio Working Directory to where your file is stored, by clicking on Session/Set Working Directory/Choose Directory and choosing where the file is stored.



- Alternatively, use the command

```
setwd()
```

  to specify the directoryaddress.

- Upload the file into R.

```
Road <- read.csv("2016Fatalities.csv",header=T)
```

  Note: You can check what the Working Directory is by using

```
getwd()
```

5. Snapshot of Multivariate Data

- The `dim` command gives the dimension of the matrix.

```
dim(Road)
## [1] 442  18
```

- The `names` command lists the variables.

```
names(Road)
##  [1] "Crash.ID"                          "State"
##  [3] "Date"                              "Day"
##  [5] "Month"                             "Year"
##  [7] "Dayweek"                           "Time"
##  [9] "Hour"                              "Min"
## [11] "Crash.Type"                        "BusInvolvement"
## [13] "RigidTruck..Involvement"           "Articulated.Truck..Involvement."
## [15] "SpeedLimit"                        "RoadUser"
## [17] "Gender"                            "Age"
```

- The `head` command lists the top of the dataset, where `1` specifies the 1st row.

```
head(Road,1)
##     Crash.ID State     Date Day   Month Year Dayweek  Time Hour Min
```

```
## 1 2.2016e+12   VIC 1-Jan-16   1 January 2016  Friday 20:30   20  30
##       Crash.Type BusInvolvement RigidTruck..Involvement
## 1 Single vehicle             No                       No
##   Articulated.Truck..Involvement. SpeedLimit         RoadUser Gender Age
## 1                              No         80 Motorcycle rider   Male  25
```

- The `str` command classifies each variable.

```
str(Road)
## 'data.frame': 442 obs. of  18 variables:
##  $ Crash.ID                     : num  2.2e+12 4.2e+12 1.2e+12 5.2e+12 6.2e+12 ...
##  $ State                        : Factor w/ 8 levels "ACT","NSW","NT",..: 7 5 2 8 6 6 4 6 2 2 ...
##  $ Date                         : Factor w/ 113 levels "1-Apr-16","1-Feb-16",..: 3 3 44 44 44 44 86 86 95 95 ...
##  $ Day                          : int  1 1 2 2 2 2 3 3 4 4 ...
##  $ Month                        : Factor w/ 4 levels "April","February",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Year                         : int  2016 2016 2016 2016 2016 2016 2016 2016 2016 2016 ...
##  $ Dayweek                      : Factor w/ 7 levels "Friday","Monday",..: 1 1 3 3 3 3 4 4 2 2 ...
##  $ Time                         : Factor w/ 225 levels "0:00","0:12",..: 150 10 5 109 135 135 64 37 149 164 ...
##  $ Hour                         : int  20 1 0 17 19 19 14 11 20 21 ...
##  $ Min                          : int  30 0 30 20 58 58 0 55 25 45 ...
##  $ Crash.Type                   : Factor w/ 3 levels "Multiple vehicle",..: 3 3 3 1 1 1 2 1 3 3 ...
##  $ BusInvolvement               : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ RigidTruck..Involvement      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Articulated.Truck..Involvement.: Factor w/ 2 levels "No","Yes": 1 1 1 2 1 1 1 1 1 1 ...
##  $ SpeedLimit                   : int  80 110 100 110 80 80 60 100 100 60 ...
##  $ RoadUser                     : Factor w/ 6 levels "Bicyclist (includes pillion passengers)",..: 4 2 5 2 4 3 6 4 2 2 ...
##  $ Gender                       : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 1 ...
##  $ Age                          : int  25 40 18 53 17 31 70 51 59 17 ...
```

- To choose a particular variable, select `dataname$variablename`

```
SpeedLimit <- Road$SpeedLimit
```

- To classify a particular variable, use the `class` command.

```
class(SpeedLimit)
## [1] "integer"
```

- A factor can be re-classified as numerical by using the `as.numeric` command.

- Most commands are easy to guess.

```
mean(SpeedLimit)
hist(SpeedLimit)
x+1
1/2*(exp(x))
```

6. Saving Results

It is good practise to make a summary of your work in each tutorial. The easiest way is to:
(1) Open an RScript file [Click on File/New File/R Script]
(2) Copy and paste useful commands.
(3) Save the file.
(4) In RStudio, you can reopen this file at any time, and press Run to perform the commands again.