# Introduction to R: Basics

Dr Cameron Hurst
cphurst@gmail.com

DAMASAC and CEU, Khon Kaen University

17th August, 2558

## Conventions: Reminder

**Note:.....**

Things to note will occur in a green box

**Pitfalls:.....**

Common mistakes and things to watch out for will occur in a red box

**R SYNTAX:....**

Most (important) R syntax will be in purple boxes and be in `courier` font. This will help you find it easily when you have to refer back to these notes.

## "Learning" syntax

Before we move onto R, I would just like to make one important point:

**You are not expected to learn R syntax!!!!**

You are just expected to remember which session you covered a particular topic, and refer back to those notes (or other resources....e.g. R help files).

**Hint: Learning R**

USING (not memorizing) syntax is the best way to learn it

## Introduction
What we cover today.....

## About R.....

- R is freely available open-source software created under the GNU agreement (i.e. Open source)
- It brings together the basic functionality of the creators of R (core packages) and work from others such as you and I (contributed packages)
- In this respect R is at the cutting edge in a way that the expensive competitors (i.e. SAS, Stata and SPSS) can't be

## What can R do?

R can do pretty much all statistical analysis:
The basic tests

- Pearson's Correlation Coefficient
- Independent and paired t-test
- $\chi^2$ test of independence
- etc etc....

These can be a little cumbersome. Where it's real strength is is in the intermediate to advanced methods.

## Intermediate and advanced methods
### The linear models

- General linear models (ANOVA, linear regression)
- Generalized linear models (Logistic regression, Poisson Regression etc)
- Cox proportional hazard regression (Survival analysis)
- Mixed models (Linear mixed models, Generalized linear mixed models and GEEs)

The linear models are R's real strength because of the approach R uses makes it very easy (if you can specify a Linear regression you can specify anything).

**Hint:**

Don't worry you will be experts in these techniques soon

## Intermediate and advanced methods
### Other methods

- Generalized additive models (good for epidemiological studies of ecological datasets)
- Multivariate methods (Factor analysis, Principal component analysis, Partial least squares, clustering and many more)
- Time series analysis methods
- Specialized analyses(e.g. Genetic/genomic studies)

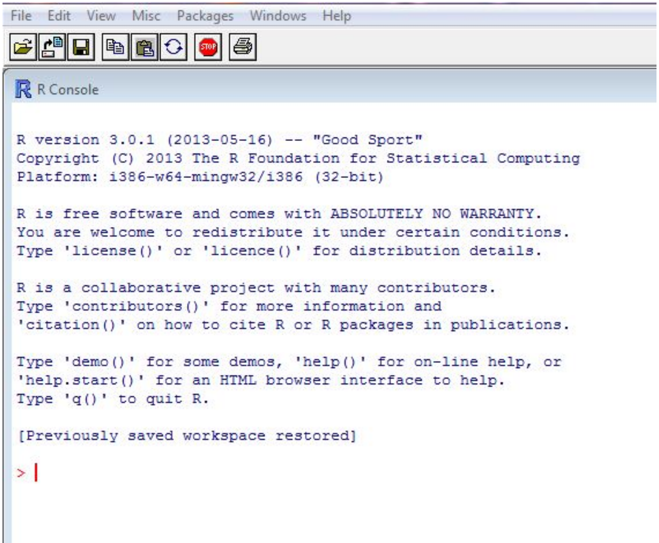
## Graphics

- R also produces excellent publication quality graphics
- It provides a wide range of graphs; and
- (once your R coding is good enough) all statistical method and graphs can be easily extended or enhanced

## Where can I get R?

- R can be downloaded from `http://cran.r-project.org/`
- This web site provides both the base and contributed (written by others) packages.
- It is available on Windows, Linux/Unix and Mac platforms

## Opening R



```
R version 3.0.1 (2013-05-16) -- "Good Sport"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

>
```

We can use R interactively, where we type in one line of code at a time

```
> x <- 1.3
> y <- 2.5
> x+y
```

```
[1] 3.8
```

## Interactive or script file?

Using R interactively (typing in individual lines) is generally not the best way to use it.
As you get better you will want to start collecting your syntax into syntax files ("R files"). For example, I might save my code into
`myanalysis.R`
R comes with a (VERY) basic script file editor, but there are a few much better ones (freely) available. The ones I have used are:

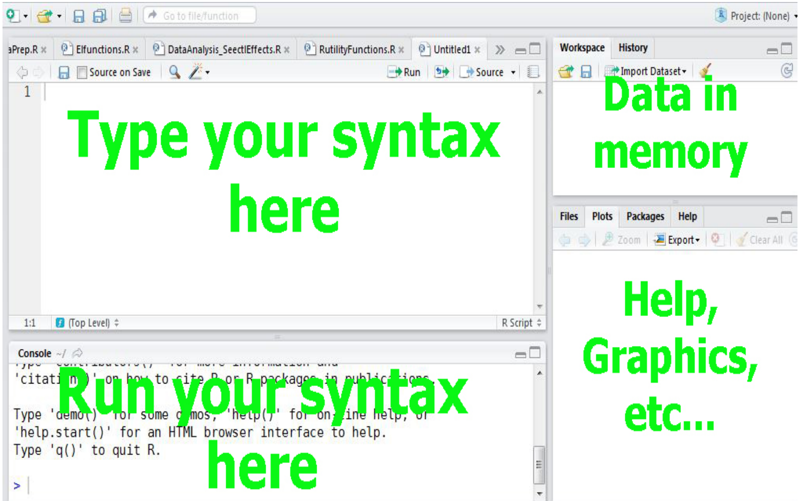
- tinn-R (version 1.17.2.4 later versions are unstable)
- notepad++
- R-studio (**Recommended)**

**Hint: Downloading R studio**

R studio can be downloaded from `www.rstudio.com/ide/`.
Also cross-platform (Windows/Linux and Mac versions available)

## Opening Rstudio

Main advantage of R studio: Keeps everything on the same screen



```
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

## Objects

R works on **objects**. These include:

- single number or character
- vector or 1-D array of numbers, characters, logical values, or factors
- matrices (table of values of a **single** type)
- data frames (table allowed to contain variables of different types) = same as SAS, Stata or SPSS dataset
- lists (data frames as a special case of a list)
- functions and user defined objects

**Hint:**

It is a good idea to include type in object name e.g. `mydataframe.df` and `mymatrix.mat`

## Naming objects

Object names tend to use letters, numbers and periods
Values (or functions) are assigned to an object using the $<-$ operator. For example: `my.val <- 7` stores the value of 7 into the object `my.val`

**Aside:**

The character $=$ can also be used to assign values to an object, but $<-$ is much more widely used.

**Warning/Pitfall:**

- Avoid underscores in names. While technically legal, it can cause problems
- Note that the operator for "equals to" is $==$, not $=$

## Some basic data structures

```
> A.val<-7

> A.vec<-c(1:6)

> A.mat<-matrix(A.vec, nrow=2, byrow=TRUE)
```

## To find out what is in each object, just type name at prompt

```
> A.val
[1] 7

> A.vec
[1] 1 2 3 4 5 6

> A.mat
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

## Data input and data frames

- The most commonly used data structure you will use in R is the data frame
- The data frame represents a matrix-like object where the columns represent variables and the rows observations
- Rarely will we manually enter the data directly into R, it is much more common that we will read it in via a data file

**Aside: Data frames**

Unlike matrices, data frames allow different columns to be of different data types. E.g. Column 1 could represent **sex** (Categorical: **M**, **F**) and column 2 **age** (Continuous: **any value between 15 and 75**)

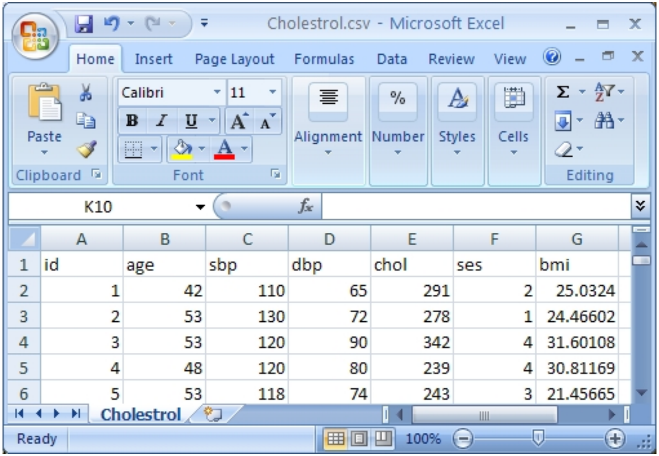
## Data input and data frames
### Motivating example

Consider a dataset containing 200 patients on which the following variables are measured:

1. id (Patient ID)
2. age
3. sbp (Systolic blood pressure)
4. dbp (diastolic blood pressure)
5. chol (LDL-cholesterol)
6. ses (Socio-economic status <- education and income)
7. bmi (body mass index)

These data are currently stored in an excel comma delimited (csv) file. Note that a few other variables (e.g. categorizations of BMI) have been included for your convenience.

## Cholesterol data in csv file
Motivating example


Cholestrol.csv - Microsoft Excel
Home Insert Page Layout Formulas Data Review View
Calibri 11
Paste Clipboard Font Alignment Number Styles Cells Editing
K10
Cholestrol
Ready 100%

|   | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | id | age | sbp | dbp | chol | ses | bmi |
| 2 | 1 | 42 | 110 | 65 | 291 | 2 | 25.0324 |
| 3 | 2 | 53 | 130 | 72 | 278 | 1 | 24.46602 |
| 4 | 3 | 53 | 120 | 90 | 342 | 4 | 31.60108 |
| 5 | 4 | 48 | 120 | 80 | 239 | 4 | 30.81169 |
| 6 | 5 | 53 | 118 | 74 | 243 | 3 | 21.45665 |

## Cholesterol data in csv file
### Motivating example

To read in the csv file and dump it to a dataframe:

```
> mychol.df<-read.csv("E:/Introduction to R/Basics/Cholesterol.csv")
```

**Warning/Pitfall:**

Note the use of the forward slash "/" rather than the traditional backslash. Note that a double back slash (\\) can be used instead of the forward slash

Alternatively we can set up a working directory:

**R SYNTAX: Reading in a comma delimited text file**

```
> setwd("E:/Introduction to R/Basics")
> mychol.df<-read.csv("Cholesterol.csv")
```

## Reading in data from other stats packages

We can also read data in SPSS (.sav), Stata (.dta) data files (and other stats packages) using the **foreign** R library.
E.G. To read in the same data above from a Stata data file:

### R SYNTAX: Reading in a Stata datafile

```
library(foreign)
setwd("E:/Introduction to R/Basics")
mychol.dta.df<-read.dta("Cholesterol.dta")
```

### Warning/Pitfall:

Avoid using this approach. OK for simple datasets, but can be unstable for other types of data (e.g. data formatted for survival analysis). Also can be problems with SPSS's value labels. Best practice to export data from your stats package into a comma delimited (.csv) format first

## Libraries

- Most of the analysis you will perform when you start with R will use methods available in the libraries that come with the base distribution of R
- For example, most statistical analysis will be from the libraries `stats` or `MASS`, and most graphics will employ the `graphics` library
- These standard libraries load into to memory when we start up R

## Libraries

- As you use more specialized and advanced analysis you will have to load a library into memory before you can call its methods. We do this with the `library()` statement (usually at the start of our R file)
- Some libraries (e.g. `foreign`) are provided with the base distribution of R, but still need to be loaded into memory (e.g. `library(foreign)`
- Other libraries need to be downloaded from repositories (e.g. CRAN) first

For example, if you wanted to analyse data arising from a longitudinal study, you might use Linear Mixed Models from the library `lme4` In this case you would have to download this library from CRAN and then start your script file with the command:

```
library(lme4)
```

## Viewing data in data frames and matrices

Many ways to view your data in R, I will just mention two:

1. Just type the name of your data frame (or data object) at the prompt
   ```
   mychol.df
   ```
2. Click the data frame (or object) in the workspace area in Rstudio (top right hand corner), alternatively, type:
   ```
   > View(mychol.df)
   ```

**Hint: Viewing the dataset**

Viewing the data is very easy in R studio, just click the data frame in ”Workspace” (Top right hand window).

## Indexing

One of the best features in R is how object indices can be used to subset the data. For example:

- If we only want the first three columns:
  `subset.chol.df<-mychol.df[,c(1:3)])`
- If we only want the rows 10 to 20:
  `subset.chol.df<-mychol.df[c(10:20),])`
- If we only want all the rows **except** 10 to 20:
  `subset.chol.df<-mychol.df[-c(10:20),])`
- If we want all patients in the 2nd socio-economic group:
  `subset.chol.df<-mychol.df[mychol.df[,6]==2,])`
  or...
  `subset.chol.df<-mychol.df[mychol.df$ses==2,])`

Hint:

- Columns can be referenced by col number, or name
- The absence of something before(after) the "," means all rows(columns)

## Documenting your code: The importance of comments

If you want to leave yourself "notes" about your syntax use #

**R SYNTAX: Reading in a data (with comments)**

```
# I can write anything here blah blah

# Set my working directory
setwd("E:/Introduction to R/Basics")

# Read in comma delimited chol data
mychol.df<-read.csv("Cholesterol.csv")
```

**Hint: Documenting your code**

Documenting your code has two MAJOR advantages:

1. You (and other people) can understand what you have done
2. If you do it well, it IS your methods section

## Data type vs. Data representation

Before we start seeing how R graphics work, I would just like to remind you about the relationship between the types (scales) of variables we have, and how we (graphically) represent it. For univariate relationships:

Table: **Data representation(univariate)**

| **Variable type** | **Method** |
|---|---|
| Continuous | Histogram<br>Boxplot |
| Categorical | Histogram†<br>Frequency table |

†*Only if categorical variable is binary or ordinal(NOT nominal)*

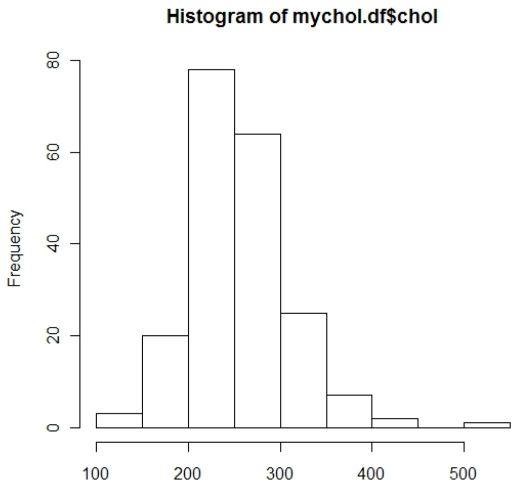
## Generating plots

- R is very strong regarding plots (with thousands available from the various libraries)
- Let's start with a basic histogram using the cholesterol data

**R syntax: A (very) basic histogram**

```
# Histogram of cholesterol
hist(mychol.df$chol)
```

## A very basic plot

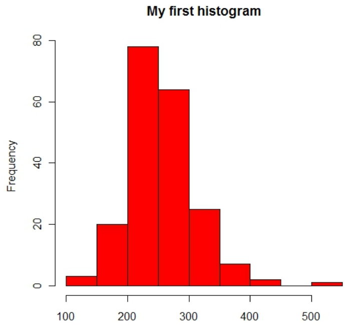


## A bit better histogram

Now let's put some more features in the graph

**R syntax: Histogram**

```
hist(mychol.df$chol, main="My first histogram",
xlab="Cholesterol", c="Red")
```

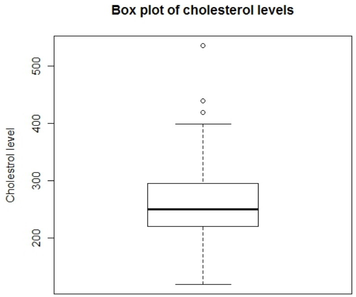


## A simple boxplot

To generate a simple boxplot..

**R syntax: Boxplots**

```
# Generating a boxplot
boxplot(mychol.df$chol, main="Box plot of cholestrol levels",
ylab="Cholesterol level")
```



## Summarizing categorical data
### Univariate

Frequency table for a categorical variable:

**R syntax: Frequency tables**

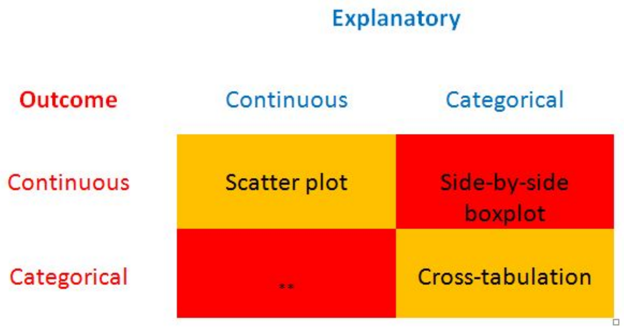
```
table(mychol.df$bmi.class)
```

Gives:

```
 Underweight   Normal   Overweight/obese
      2          93            105
```

## Data type vs. Data representation
### Bivariate relationships

For **Bivariate relationships** it is important to note (if relevant) which is the **outcome variable** and which is the **explanatory variable**



|  | Explanatory |  |
|---|---|---|
| Outcome | Continuous | Categorical |
| Continuous | Scatter plot | Side-by-side boxplot |
| Categorical | ** | Cross-tabulation |

*** Not really a simple graph for this situation. If we collapse the continuous variables into categories we can use cross-tabulation*

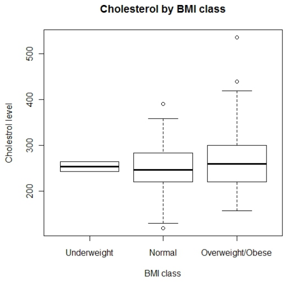
## Bivariate plots: Side-by-side boxplots

Here, I collapse BMI into classes (underweight, normal, overweight/obese)

### R syntax: Side-by-side boxplots

```
boxplot(chol~bmi.class, data=mychol.df, main="Cholesterol by BMI class", ylab="Cholestrol level", xlab="BMI class")
```

Note the use of:
`chol∼ bmi.class`
This is a very important feautre in R. It is the `formula` format and you will see **much** more of it.



## A basic scatter plot

**R syntax: Scatter plots**

```
plot(x=age, y=chol, data=mychol.df, main="My first scatterplot",
sub="Cholestrol vs Age", pch=2, col="Blue", xlab="Age",
ylab="Cholestrol")
```

**Hint:**

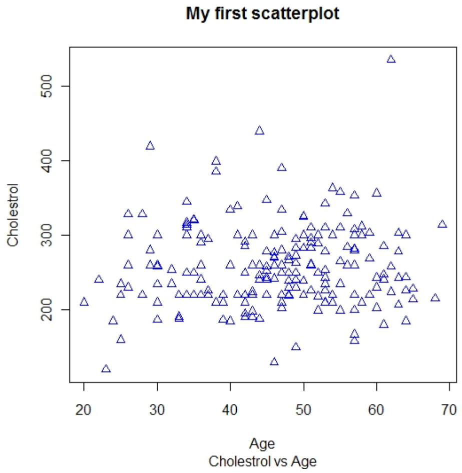
- R doesn't care what order you specify function parameters
- if you don't specify a parameter, R will use the default
- `pch` stands for plot character

**Aside:**

By using `data=mychol.df` no need to prefix names with `chol.df$`

The above R code chunk gives

## A basic scatter plot



## Summarizing categorical data
### Bivariate

Now to generate a cross-tabulation:

**R syntax: Cross tabulation**

```
 # generate cross-tabulation
table(mychol.df$bmi.class, mychol.df$ses.class)
```

Gives:

```
    BMI class         low  low-mid  mid  high-mid  high
   Underweight         0      1      1      0        0
     Normal           10     15     55      5        8
 Overweight/obese     12     19     47     13       14
```

## Some summary statistics for quantitative data

**Centre**

```
mean(mychol.df$age)
46.065
median(mychol.df$age)
47
```

**Variability**

```
sd(mychol.df$age)
10.81274
range(mychol.df$age)
20 69
```

**Five number summary (min Q25, Median, Q75, Max)**

```
fivenum(mychol.df$age)

20 38 47 54 69
```

## More summary statistics

To get summary statistics by group....

```
# Mean age underweight
mean(mychol.df$age[bmi.class==1])
# Mean age normal weight
mean(mychol.df$age[bmi.class==2])
# Mean age overweight/obese
mean(mychol.df$age[bmi.class==3])
```

**Hint:**

As age only represents a single column (vector) we only need a single (row) index

## Using the psych library to get summary stats

### R syntax: Generating summary stats using the psych library

```
 # U have to download the psych library first
# Now load the psych library into memory
library(psych)

# Generate summary stats
describe(mychol.df$ chol)
```

### Downloading a (non-standard) library vs 'loading' it into memory

- For libraries that don't come with R, have to installed **downloaded** first (can do this from inside R or R studio)
- This has to only be done once
- Once installed, **load** it using `library()`
- You have to load a library for every R session

## Using psych library to get summary stats BY groups

Again, using the library psych is a much friendlier way of getting summary statistics BY groups (i.e. a set of summary stats for a continuous variable for each level of a categorical variable)

**R syntax: Generating summary stats using the psych library**

```
 # Load the psych library into memory
library(psych)

# Generate summary stats
describeBy(mychol.df$chol, group=mychol.df$
bmi.class)
```

## Saving an R sesssion (workspace)

- After reading in data, generating graphs and analyses, we can save all of this "R session" into a 'workspace'
- Allows us to start of where we left off next time we start R.
- This information is stored in an `.RData` file, and generally you would have one for each project you are doing. E.G.
    1. `Cholesterol.RData` for my cholesterol analysis
    2. `T2DM.RData` for my Type 2 diabetes study
    3. `Sleep.RData` for my analysis of sleep data

**R workspaces (`.RData` files)**

Workspaces (`.RData` files) allow us to save data, graphs, analyses etc; a major advantages of R's object oriented approach

## Saving and loading an R workspace (`.RData` files)

To save an R workspace:

**Saving a session**

```
# Make sure you have set your working directory
save.image("Blah.RData")
```

To open a session that you have previously saved....

**Loading a (previously saved) session**

```
load("Blah.RData")
```

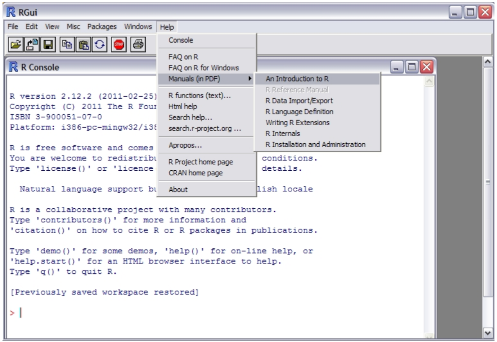
**Alternatively, you can just save and load the sessions in the 'workspace' pane in RStudio (top right-hand window)**

## R for beginners

- R can be quite tricky when you first begin
- One of the ways to make this much easier is to understand and use the help
- I will talk about three main aspects of the R help system:
  1. free introductory text
  2. searching help
  3. HTML package help

  I will also discuss resources on the internet that make it MUCH easier

## Free introductory text



```
R version 2.12.2 (2011-02-25)
Copyright (C) 2011 The R Four
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i38

R is free software and comes                          
You are welcome to redistribu                        conditions.
Type 'license()' or 'licence                          details.

  Natural language support bu                        lish locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> |
```

## Searching help

Searching the help will identify methods associated with a key word. Three ways of using:

1. *Help -> Search help....->type in keyword*
2. type in `help.search("t test")` This will identify libraries with this (or similarly named) functions

## Searching help

Finally, if we know the function name (e.g. `t.test`) and want help in its properties use:

- `help("myfunction")` or `?("myfunction)`
  For example: `help("t.test")`
- If we know what package the function is, we can go to: *Help -> HTML help -> packages*, and then we would choose the appropriate package

## HTML help

- The help I most commonly use is the HTML help pages
- Go to *Help*→*html help*, and then hit the "packages' hyperlink
- This lists all of the packages (and their functions and datasets)

In R studio, it is even easier to get to these pages. Go to the help window (bottom left window) and hit the "help" pane

## R resources on the internet

- Being open source, a lot of resources on the internet
- Also a lot of wikibooks and free textbooks available on the web (e.g. "The R book" is about 1300 pages)
- One of the best websites for beginners is the Quick R site: `www.statmethods.net`
- There are also HEAPS of youtube walk-thrus. There is a really good set listed on "R function of the day" site `rfunction.com`. This provides about 20 5-minute you tube walk-thrus which will get you well on the way
- Alternatively, just type "R for beginners" or "Introduction to R" in youtube's search engine.
- Help pages (more advanced users). I rarely encounter a problem in R someone else hasn't come across (and solved). I usually just google my problem.

## A final word...

- R is a comprehensive and comparatively 'low-level' language
- For those of you not used to coding/programming, mastering R will take a little while
- However, it is free (open source), extendible, and IS the strongest and most versatile statistical packages available
- For this reason it is the package of choice among many statisticians (few people look back after converting over to R)
- Once you have put this work in, you won't ever look back either

**Number one hint for learning R**

USE IT!!!!!!!

## R code for this lecture

I have provided the data and the R script file for all code snippets provided in this lecture in the files:

Syntax: Rsyntax_ IntroR.R
Data: Cholesterol.csv

Note:

- You will have to change the working directory for where **YOU** save your data
- Hint: Keep the location simple (e.g. `c:\myRdata`)
- Note all R syntax file have the ".r" extension

## Any questions??????

Thank-you!!!!!!

YOUR TURN!!!!

## The Diabetes and Hypertension dataset

For our exercise we will use the Diabetes and Hypertension dataset:

- National Thai study documenting cases of Type 2 Diabetes Mellitus and Hypertension from 605 hospitals across Thailand (2554-2556)
- We will consider a very 'trimmed down' version of the dataset:
    - 2500 Diabetics (who may or may not also be hypertensive)
    - Of the hundreds of variables measured in the study we will only consider a small subset (next page)
- Today we are just go to use this data to practice reading data in, basic graphics and some summary statistics

## The Diabetes and Hypertension dataset
### Variables measured

| Outcomes | Study effect | Covariates |
|---|---|---|
| **a1cyn** | **ht** | **sex** |
| **bpyn** | | age |
| **ldlcyn** | | **religion** |
| **all3yn** | | duradm |
| **any3yn** | | **smoke** |
| a1c | | bmi |
| ldlc | | **bmigroup** |

**Blue** variables are categorical (all variables are binary) and grey variables are continuous

## Variable description: Outcome

The outcomes we will consider represent the "ABC" clinical goals often used to assess the quality of diabetes care:

A `a1cyn`$\rightarrow$ Hemoglobin A1c (yes: $< 7\%$; no $\geq 7\%$)

B `bpyn`$\rightarrow$ Bloop pressure(yes: $< \frac{130}{80}$mmHG; no $\geq \frac{130}{80}$ mmHg)

C `ldlcyn`$\rightarrow$ Low density lipoprotien-Cholesterol(yes: $<$ 100mg/dL; no $\geq$ 100mg/dL)

Hemoglobin A1C (`a1c`) and LDL cholesterol `ldlc` are also in the dataset in their continuous form

We will also consider the 'collective' quality performance outcomes:

- `all3yn`: **All** three (ABC) clinical targets are met
- `any3yn`: **Any** of the three (ABC) clinical targets is met

## The study effect

For our purpose, we are interested whether there is any difference in the achievement of clinical targets between patients with Type 2 DM alone (T2DM), and those that have the hypertension comorbidity (DMHT). We will use the `ht` variable to represent this (`0:T2DM; 1:DMHT`).

**Note: Study effects**

Remember a **study effect** is an explanatory variable that is of primary interest (in terms of our research hypothesis)

## Exercise:

We are primarily interested (today) in whether there is a relationship between **A1C** and the hypertension comorbidity, but also whether the duration of T2DM relates to A1c. So we will consider four variables:

1. `a1c`:Hemoglobin A1c in it's continuous form
2. `a1cyn`:Hemoglobin A1c in it's categorical form
3. `ht`: The absence (0) or presence (1) of the hypertension comorbidity
4. `duradm`: Time (years) since T2DM diagnosis

### Datafile

You will find the data in the file: `DMHT2500.csv` (text file) and the R workspace (`DMHT2500.RData` )

## Exercise:

So we will consider three (sets of) analyses:

- `a1c` vs `ht`: Continuous outcome and categorical explanatory variable
- `a1cyn` vs `ht`: Categorical outcome and categorical explanatory variable
- `a1c` vs `duradm`: Continuous outcome and continuous explanatory variable

So for the exercises:

1. For univariate analysis:
    - Consider the distribution of `a1c` (graphs/summary stats)
    - Consider the numbers of T2DMs and DMHTs (`ht`)
2. Bivariate analysis: Consider the 3 relationships outlined above

**Hint:**

Think first about analyses, and only then about R