Presentation of R and R Studio

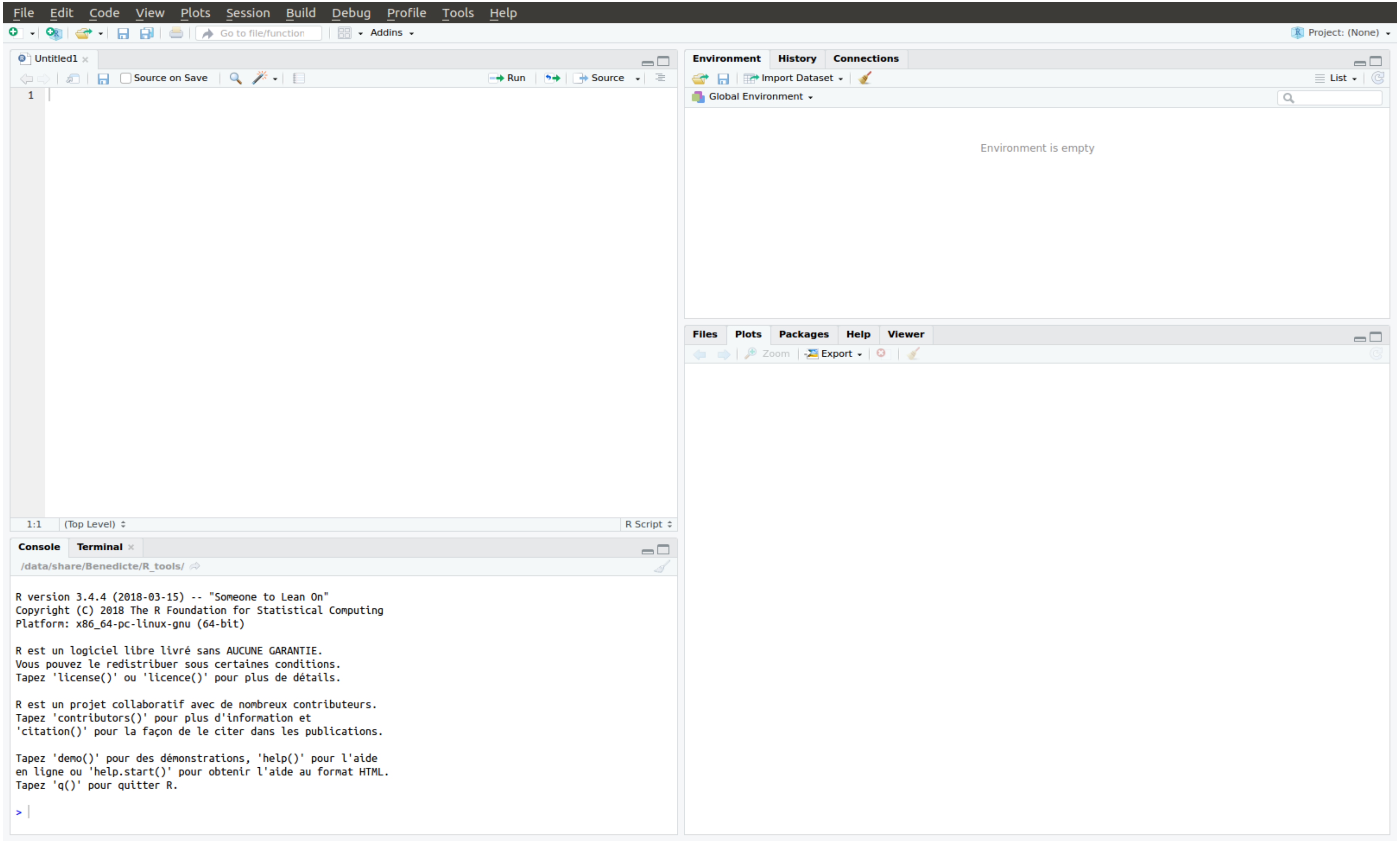
Bénédicte Lefèvre – Club BioInfo
08/11/2018

What are R and R studio?

- R is a programming language created in 1993
- R and R studio are free, open source, software environment
- CRAN project: https://cran.r-project.org/

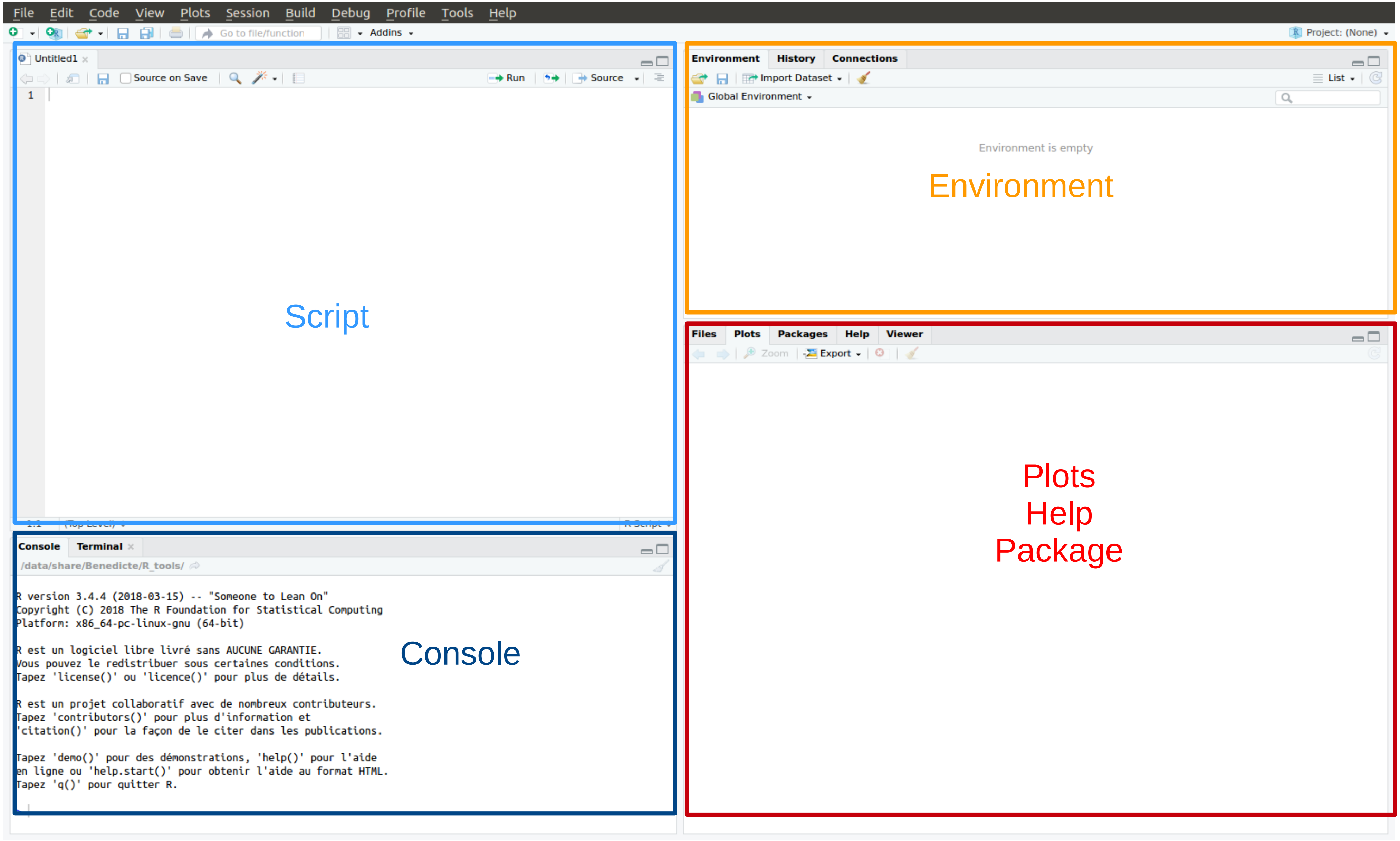
- R is useful to analyse data:
    - sorting complexe data frame
    - statisics
    - graphics
    - automatisation of repetitive tasks on datasets
    - and plenty of packages in function of your needs...

R studio



```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

> 
```

R studio

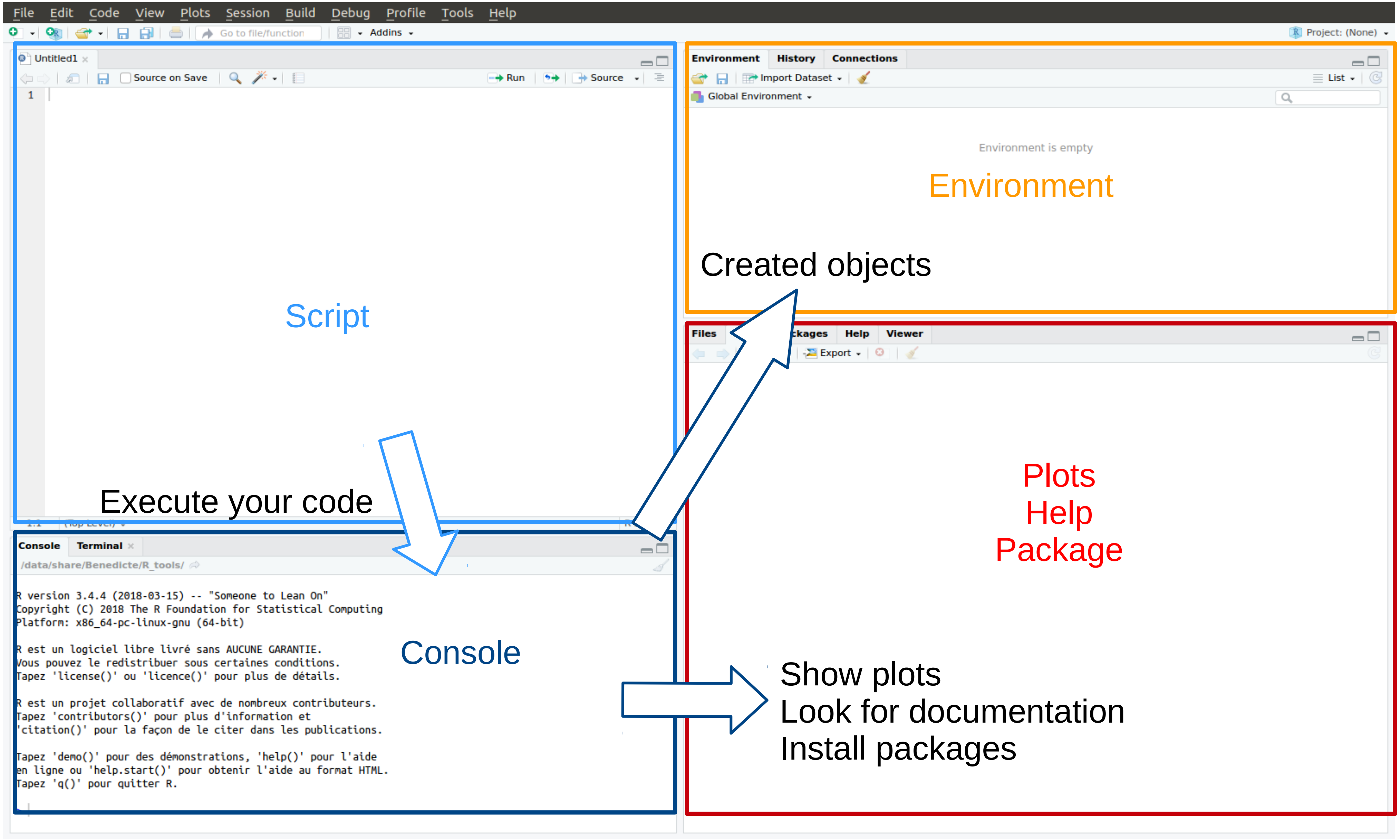


Environment is empty

/data/share/Benedicte/R_tools/

```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

> 
```

R studio



```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.
```

Script

Environment

Created objects

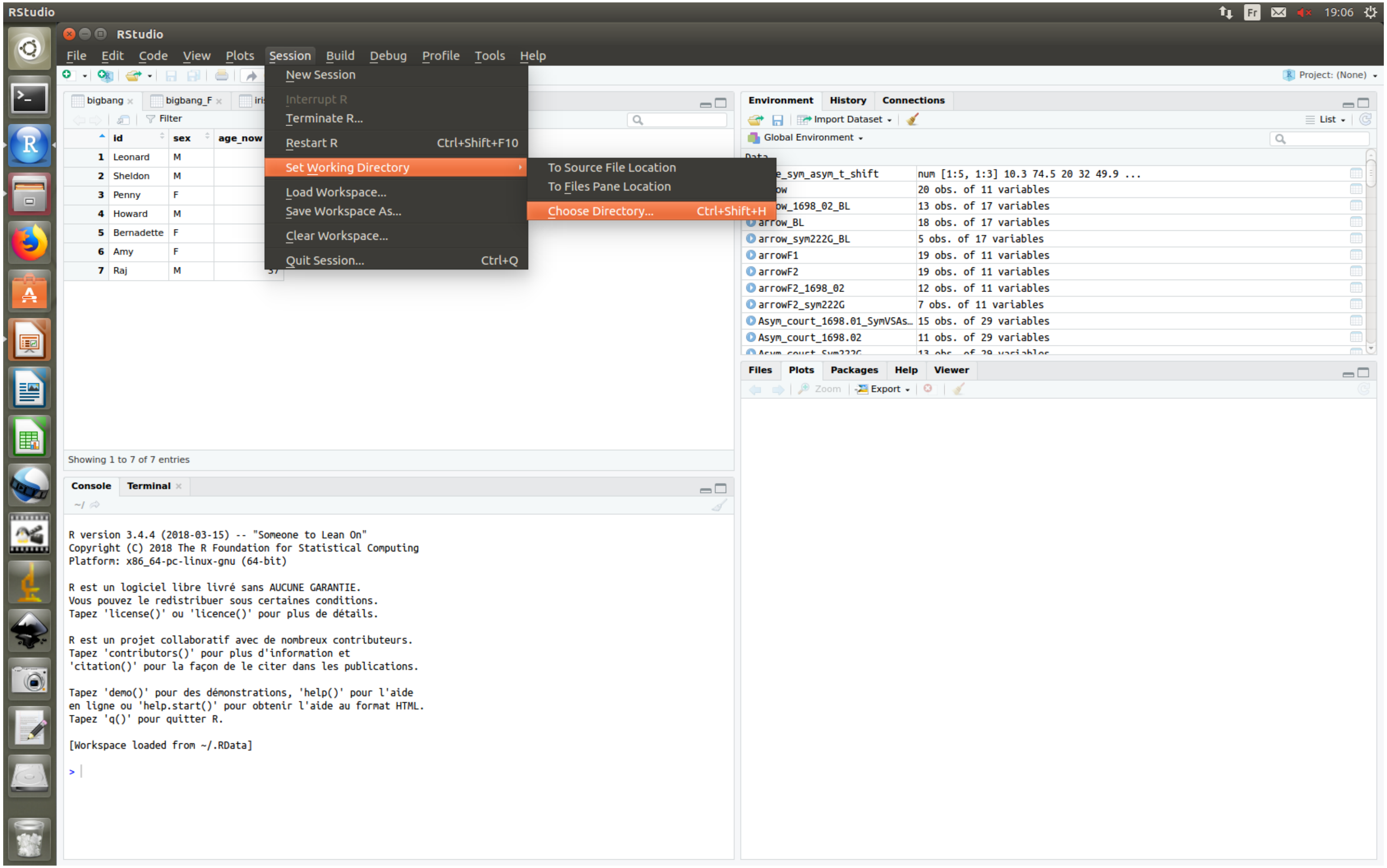
Execute your code

Console

Plots
Help
Package

Show plots
Look for documentation
Install packages

Before to start: let’s set our working directory



	id	sex	age_now
1	Leonard	M	
2	Sheldon	M	
3	Penny	F	
4	Howard	M	
5	Bernadette	F	
6	Amy	F	
7	Raj	M	37

Showing 1 to 7 of 7 entries

Data

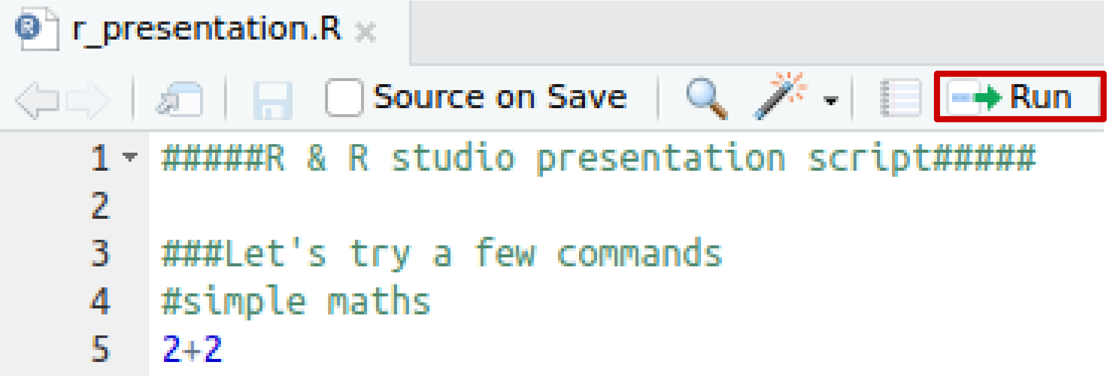
e_sym_asym_t_shift	num [1:5, 1:3] 10.3 74.5 20 32 49.9 ...
ow	20 obs. of 11 variables
ow_1698_02_BL	13 obs. of 17 variables
arrow_BL	18 obs. of 17 variables
arrow_sym222G_BL	5 obs. of 17 variables
arrowF1	19 obs. of 11 variables
arrowF2	19 obs. of 11 variables
arrowF2_1698_02	12 obs. of 11 variables
arrowF2_sym222G	7 obs. of 11 variables
Asym_court_1698.01_SymVSAs...	15 obs. of 29 variables
Asym_court_1698.02	11 obs. of 29 variables
Asym_court_Sym222G	13 obs. of 29 variables

```
~/

R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

[Workspace loaded from ~/.RData]

> 
```

- Please, select your desktop as working directory and copy the iris.csv file in your desktop

How to write a command and run it?

- Let’s try a few commands:



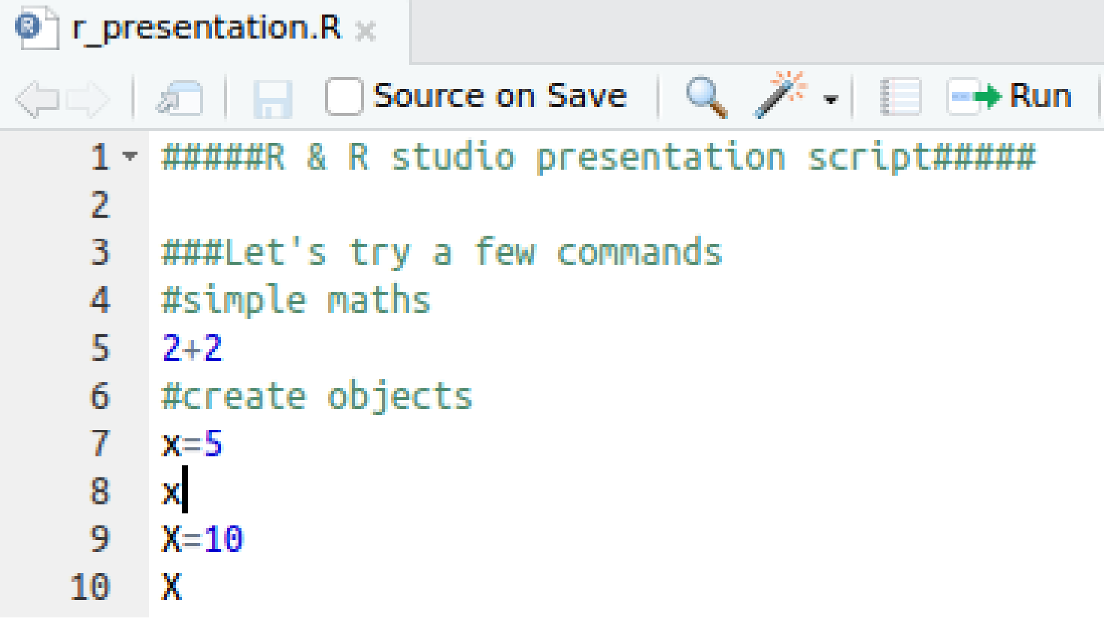
```
#####R & R studio presentation script#####

###Let's try a few commands
#simple maths
2+2
```

To run a command:
- highlight it and click on run
or
- put your cursor in the line and press CTRL+enter

Lines that starts by # are not recognized as code but as **comments**

How to create an object?

- Let’s try a few commands:



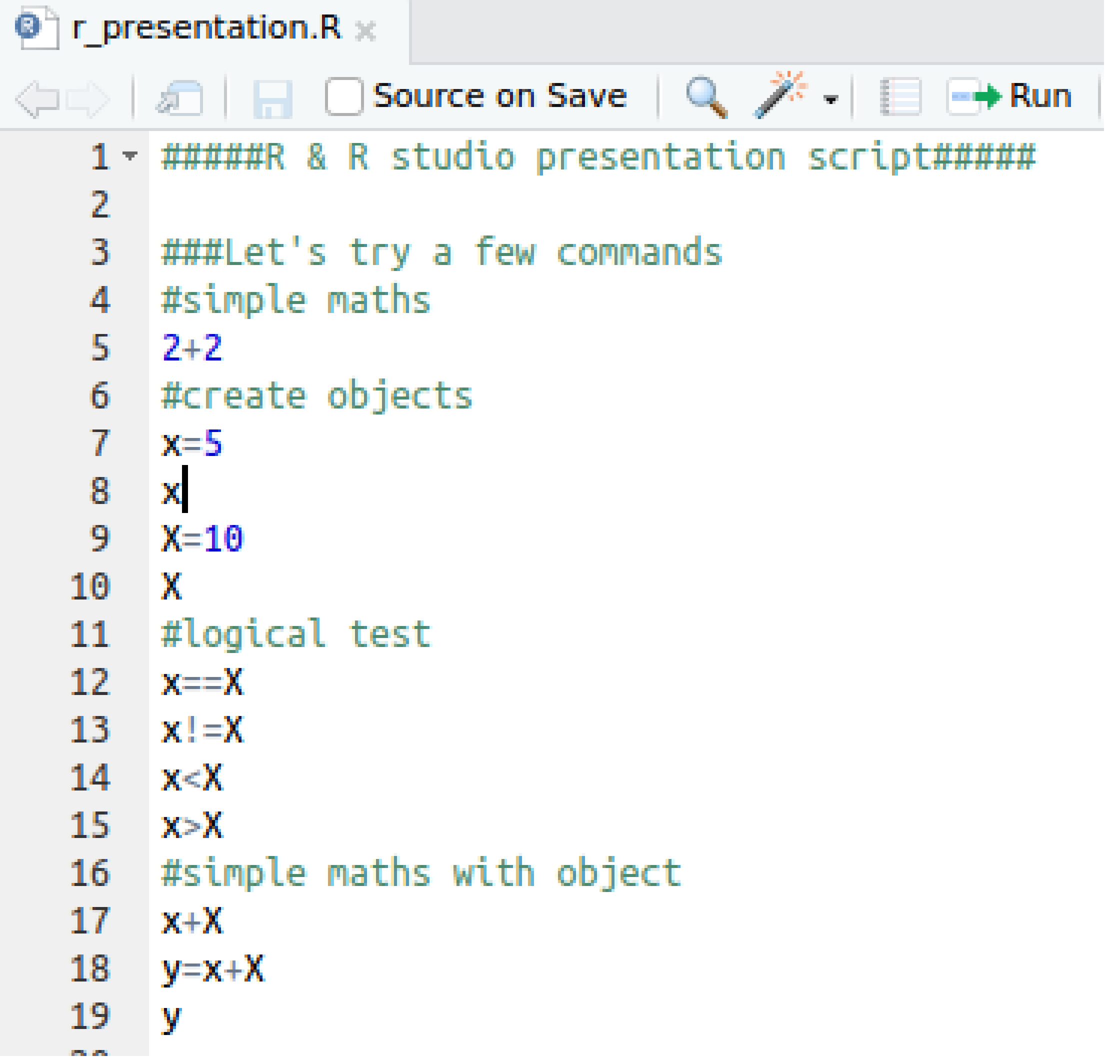
```
#####R & R studio presentation script#####

###Let's try a few commands
#simple maths
2+2
#create objects
x=5
x
X=10
X
```

To run a command:
- highlight it and click on run
or
- put your cursor in the line and press CTRL+enter

If you reassign a value to an existing object, the previous one is removed

What can we do with the created objects?

- Let’s try a few commands:



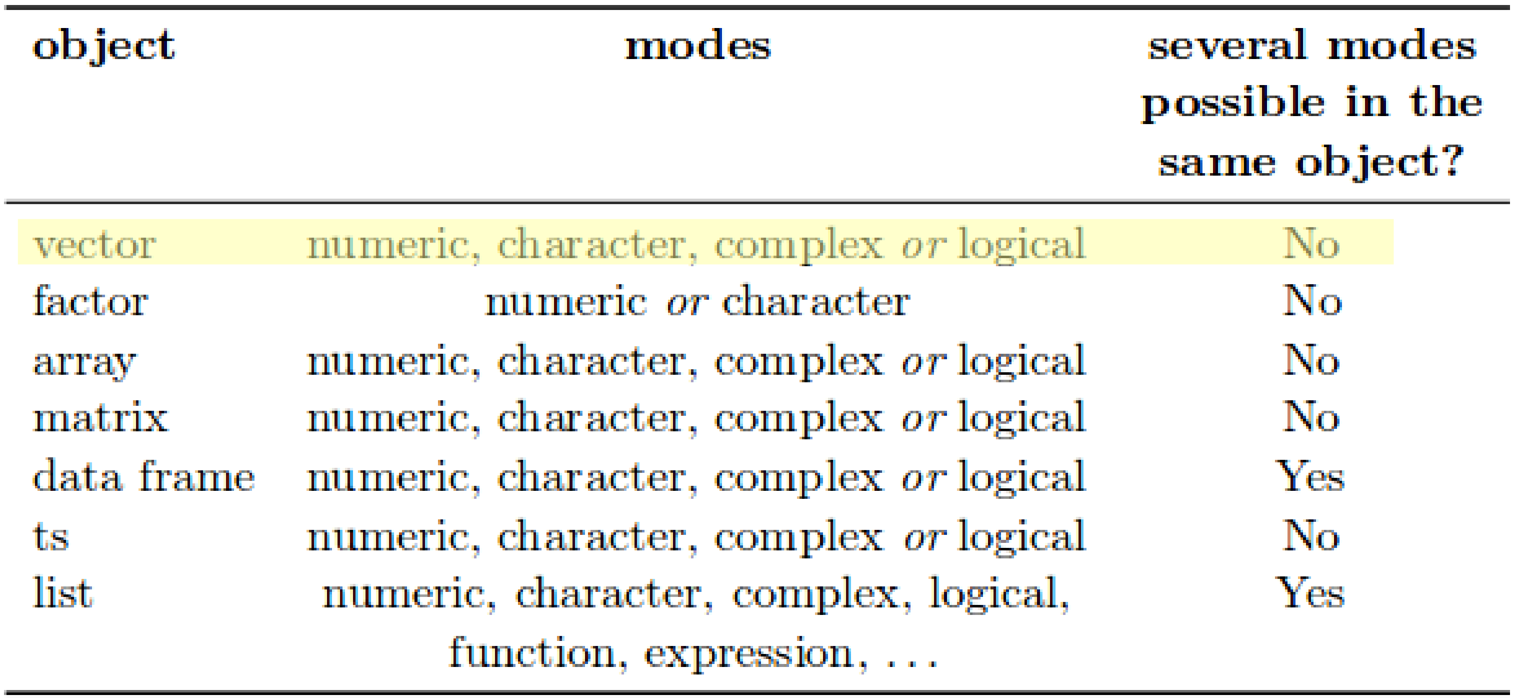
```
#####R & R studio presentation script#####

###Let's try a few commands
#simple maths
2+2
#create objects
x=5
x
X=10
X
#logical test
x==X
x!=X
x<X
x>X
#simple maths with object
x+X
y=x+X
y
```

To run a command:
- highlight it and click on run
or
- put your cursor in the line and press CTRL+enter

R is case sensitive !
x ≠ X

A very important concept in R: vectors

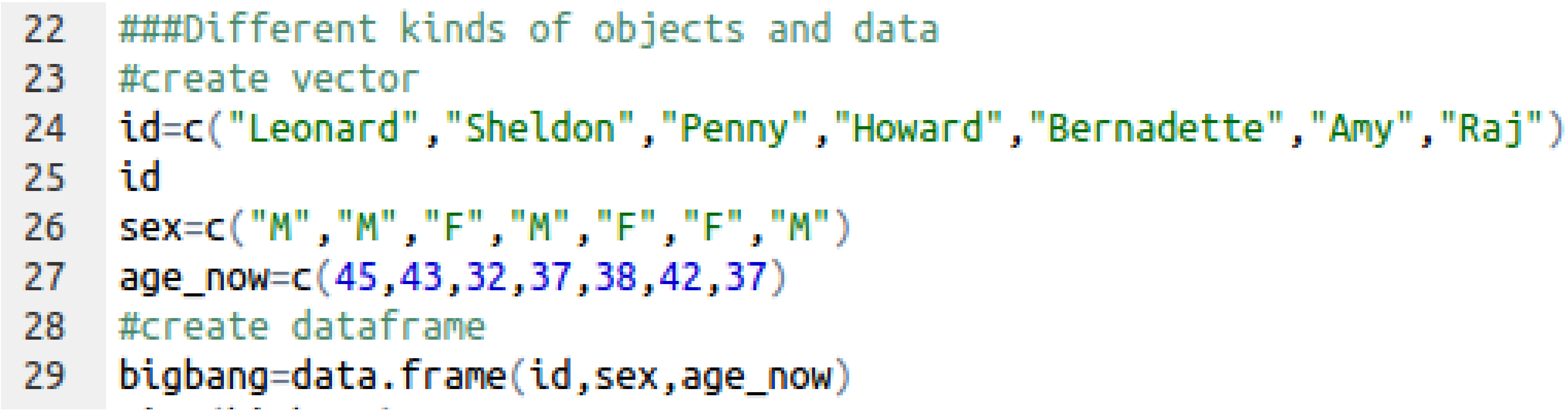
- A vector is an object that contain a serie of variables of a same type, in a precise order
- c() is used to create vectors



object	**modes**	**several modes possible in the same object?**
vector	numeric, character, complex *or* logical	No
factor	numeric *or* character	No
array	numeric, character, complex *or* logical	No
matrix	numeric, character, complex *or* logical	No
data frame	numeric, character, complex *or* logical	Yes
ts	numeric, character, complex *or* logical	No
list	numeric, character, complex, logical, function, expression, ...	Yes

E.Paradis, R for beginners, 2005

Let’s create vectors and data frame: example with BigBang Theory

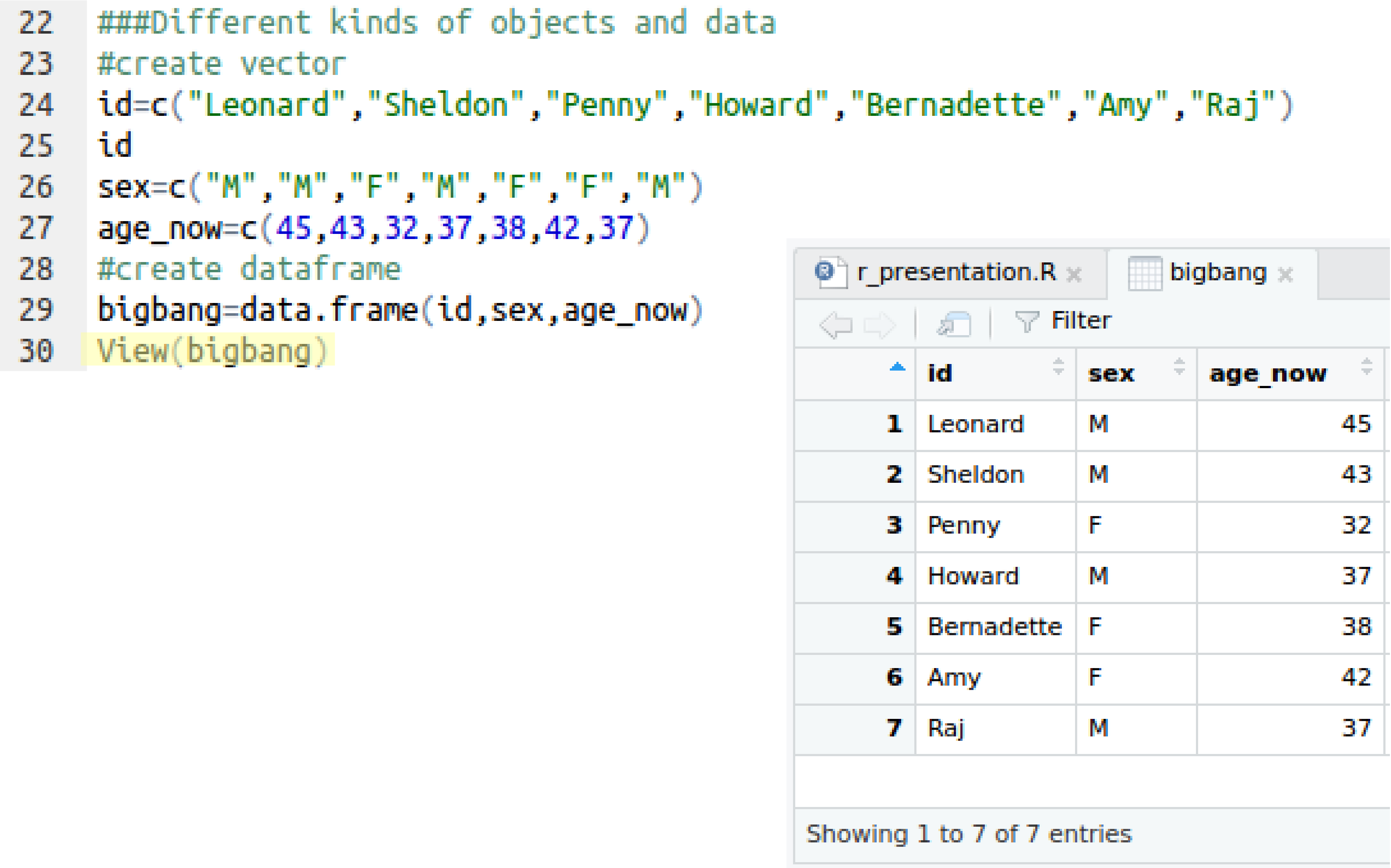


```
###Different kinds of objects and data
#create vector
id=c("Leonard","Sheldon","Penny","Howard","Bernadette","Amy","Raj")
id
sex=c("M","M","F","M","F","F","M")
age_now=c(45,43,32,37,38,42,37)
#create dataframe
bigbang=data.frame(id,sex,age_now)
```

characters are written as ’’character’’ and appear in green

numbers appear in blue

Visualisation of bigbang with View()

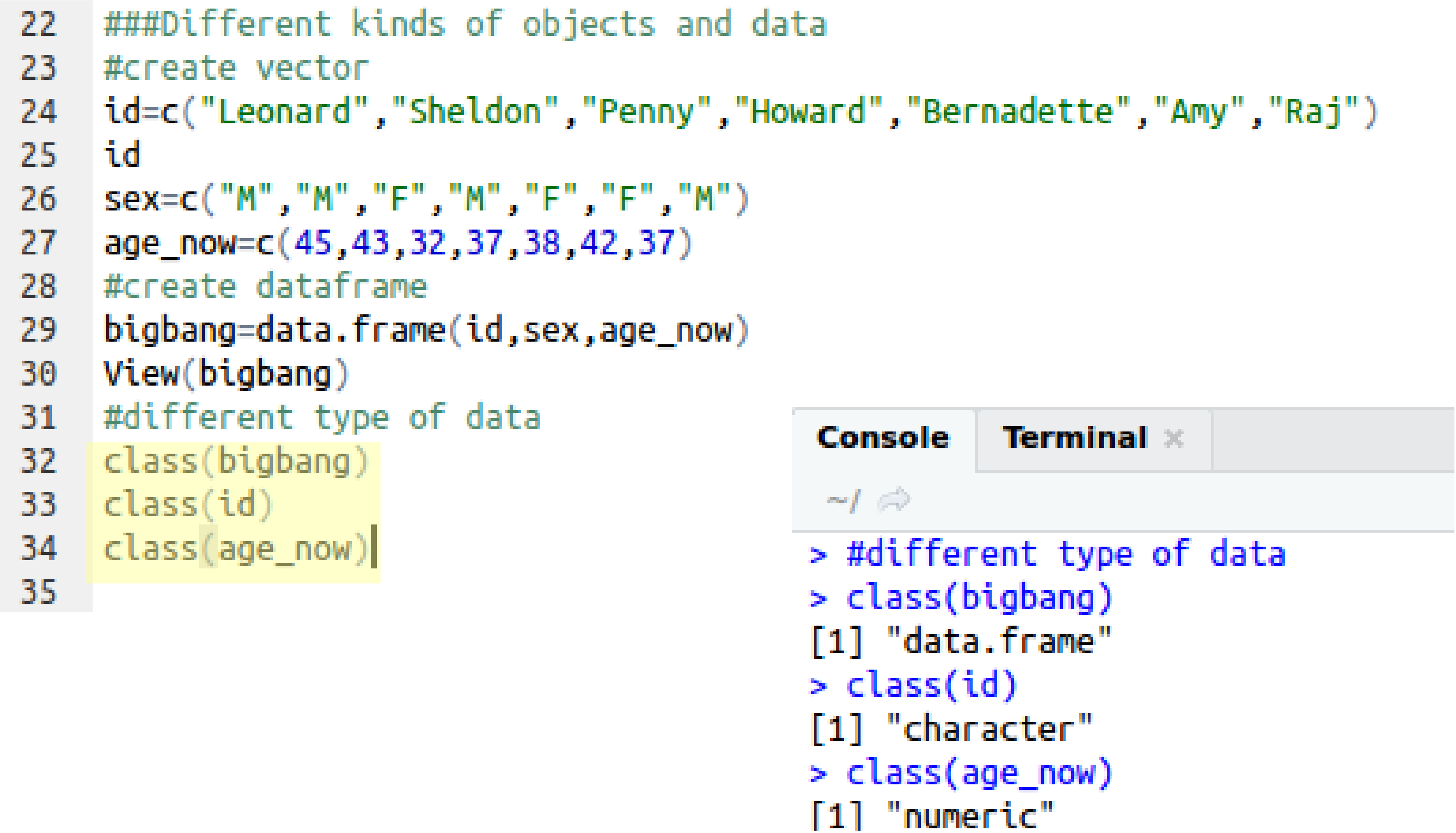


```r
###Different kinds of objects and data
#create vector
id=c("Leonard","Sheldon","Penny","Howard","Bernadette","Amy","Raj")
id
sex=c("M","M","F","M","F","F","M")
age_now=c(45,43,32,37,38,42,37)
#create dataframe
bigbang=data.frame(id,sex,age_now)
View(bigbang)
```

	id	sex	age_now
1	Leonard	M	45
2	Sheldon	M	43
3	Penny	F	32
4	Howard	M	37
5	Bernadette	F	38
6	Amy	F	42
7	Raj	M	37

Showing 1 to 7 of 7 entries

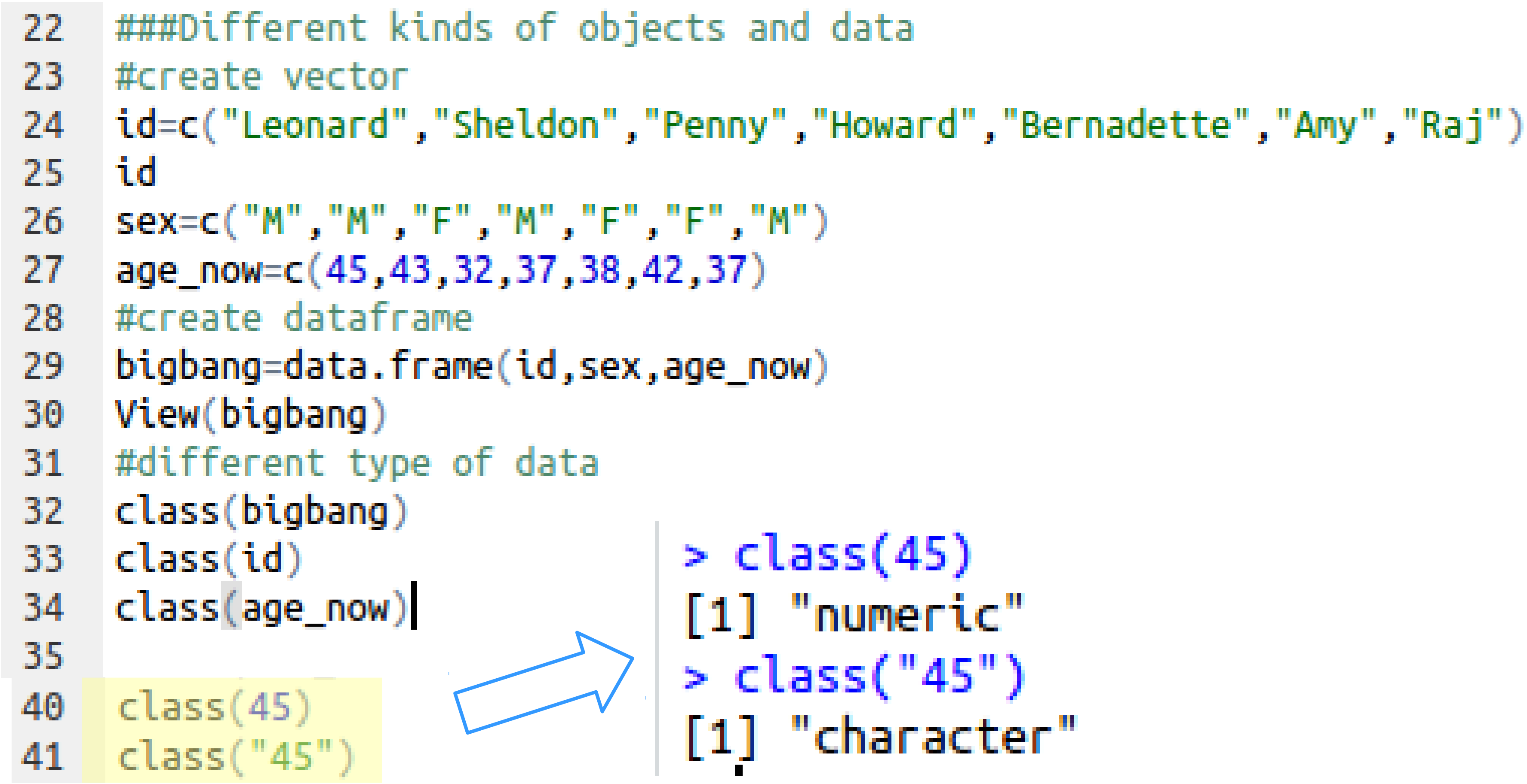
Different kinds of objects and data



```
###Different kinds of objects and data
#create vector
id=c("Leonard","Sheldon","Penny","Howard","Bernadette","Amy","Raj")
id
sex=c("M","M","F","M","F","F","M")
age_now=c(45,43,32,37,38,42,37)
#create dataframe
bigbang=data.frame(id,sex,age_now)
View(bigbang)
#different type of data
class(bigbang)
class(id)
class(age_now)
```

```
> #different type of data
> class(bigbang)
[1] "data.frame"
> class(id)
[1] "character"
> class(age_now)
[1] "numeric"
```

Different kinds of objects and data

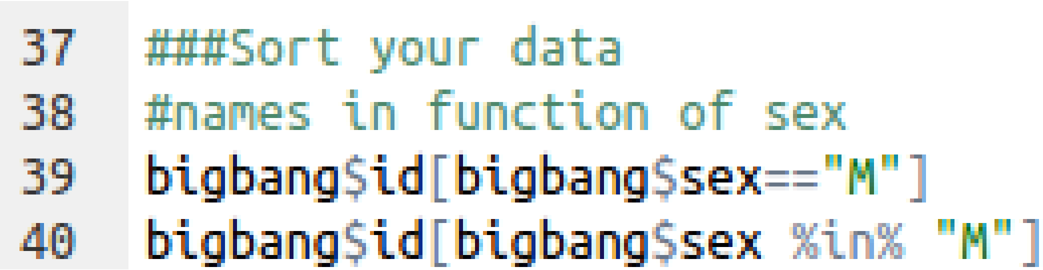


```
###Different kinds of objects and data
#create vector
id=c("Leonard","Sheldon","Penny","Howard","Bernadette","Amy","Raj")
id
sex=c("M","M","F","M","F","F","M")
age_now=c(45,43,32,37,38,42,37)
#create dataframe
bigbang=data.frame(id,sex,age_now)
View(bigbang)
#different type of data
class(bigbang)
class(id)
class(age_now)

class(45)
class("45")
```

```
> class(45)
[1] "numeric"
> class("45")
[1] "character"
```

characters are written as ”character” so ‘‘45’’ is recognized as a character and not as a number

Sort your data

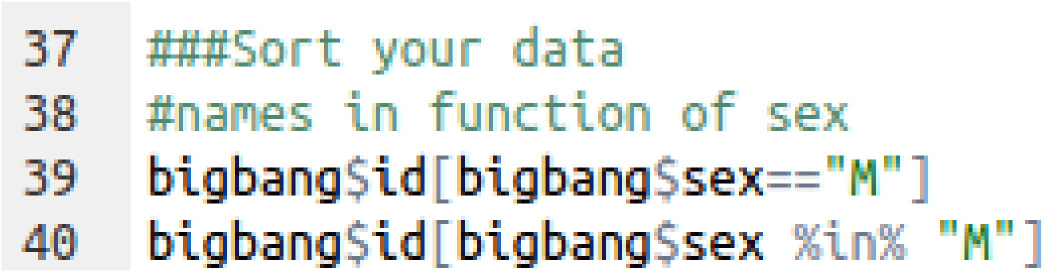


```
###Sort your data
#names in function of sex
bigbang$id[bigbang$sex=="M"]
bigbang$id[bigbang$sex %in% "M"]
```

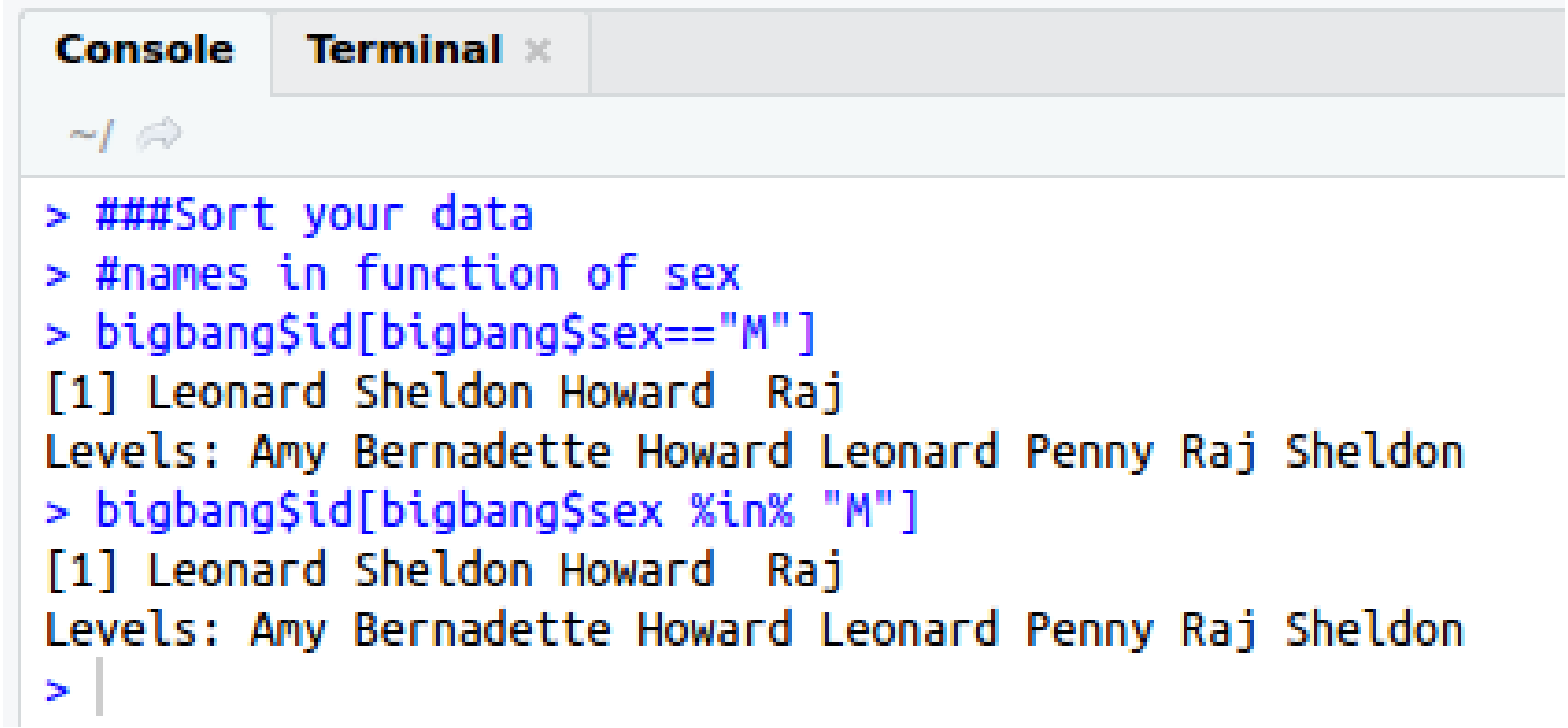
data$col refers to the column named col of the table named data

sorting is done on the elements between brackets [ ]

Sort your data

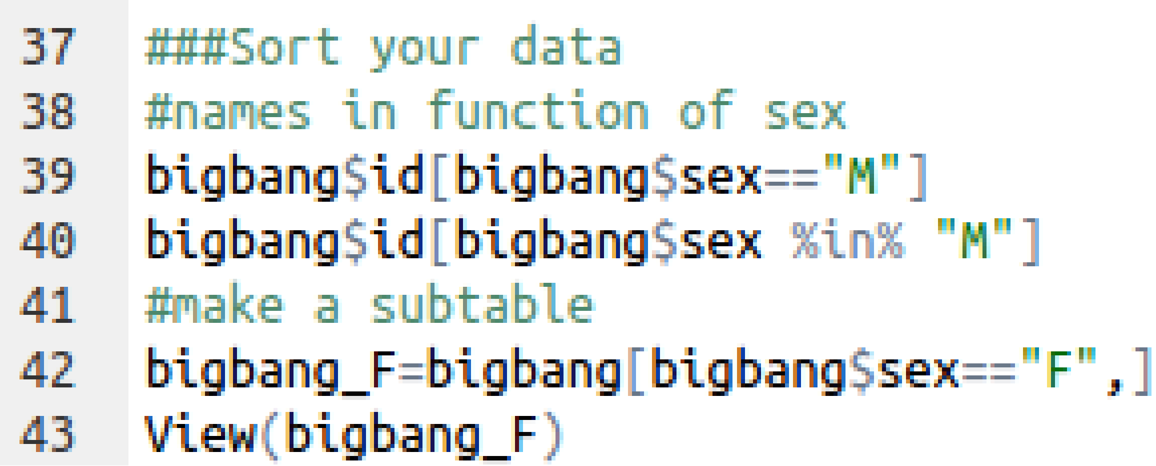


```
###Sort your data
#names in function of sex
bigbang$id[bigbang$sex=="M"]
bigbang$id[bigbang$sex %in% "M"]
```



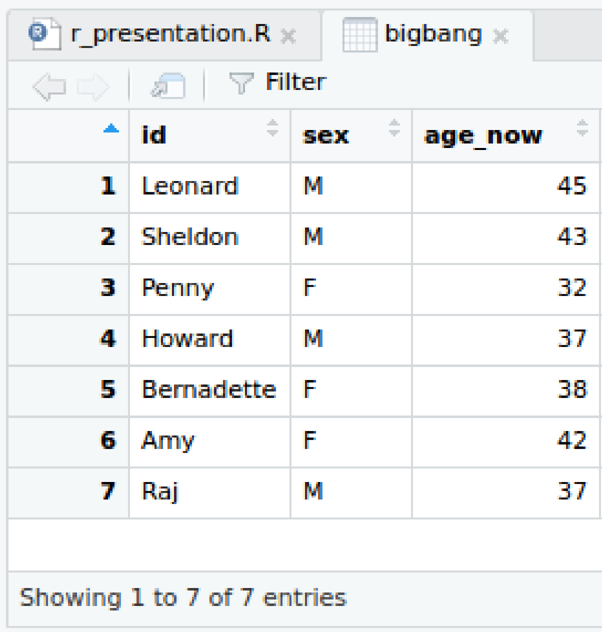
```
> ###Sort your data
> #names in function of sex
> bigbang$id[bigbang$sex=="M"]
[1] Leonard Sheldon Howard  Raj
Levels: Amy Bernadette Howard Leonard Penny Raj Sheldon
> bigbang$id[bigbang$sex %in% "M"]
[1] Leonard Sheldon Howard  Raj
Levels: Amy Bernadette Howard Leonard Penny Raj Sheldon
>
```

Make a subtable



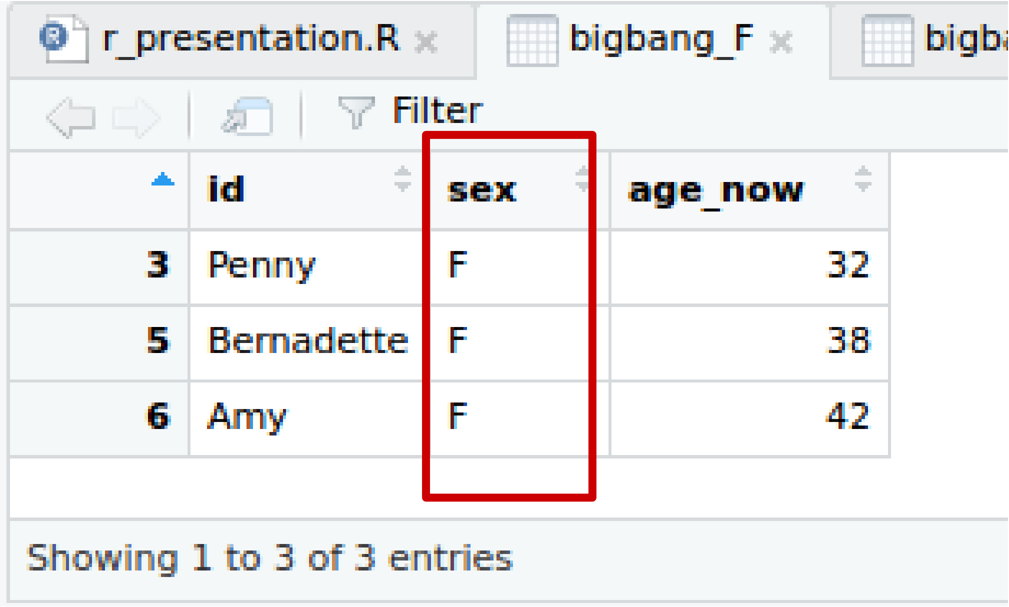
```
###Sort your data
#names in function of sex
bigbang$id[bigbang$sex=="M"]
bigbang$id[bigbang$sex %in% "M"]
#make a subtable
bigbang_F=bigbang[bigbang$sex=="F",]
View(bigbang_F)
```

As we want a subtable with 2 dimensions, it is very important **not to forget the comma in the brackets**



	id	sex	age_now
1	Leonard	M	45
2	Sheldon	M	43
3	Penny	F	32
4	Howard	M	37
5	Bernadette	F	38
6	Amy	F	42
7	Raj	M	37

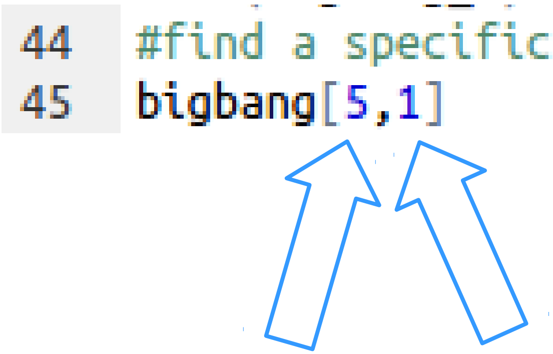
Showing 1 to 7 of 7 entries



	id	sex	age_now
3	Penny	F	32
5	Bernadette	F	38
6	Amy	F	42

Showing 1 to 3 of 3 entries

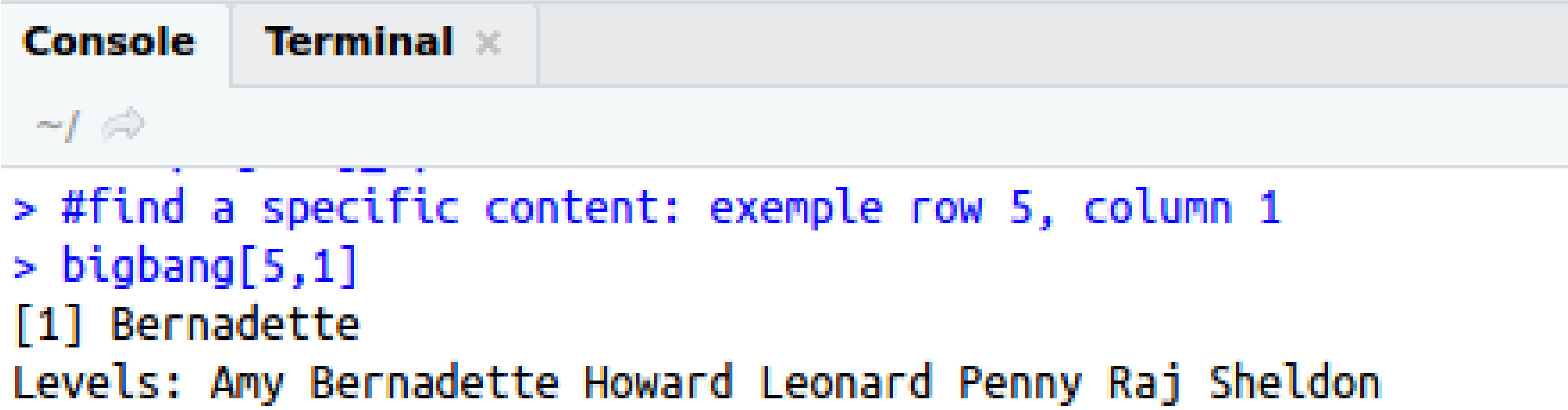
Navigate into your table: subscript



```
#find a specific content: exemple row 5, column 1
bigbang[5,1]
```

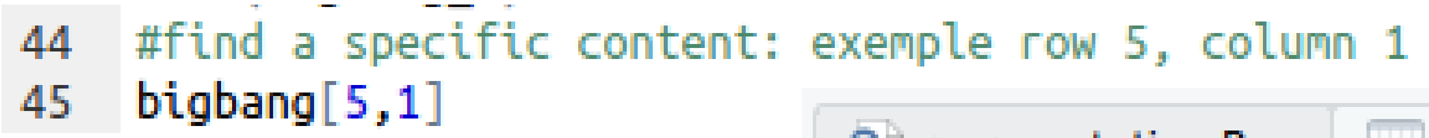
row number

column number

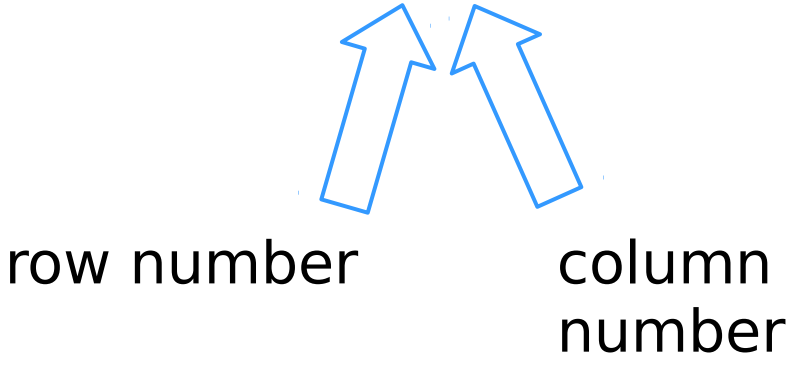


```
> #find a specific content: exemple row 5, column 1
> bigbang[5,1]
[1] Bernadette
Levels: Amy Bernadette Howard Leonard Penny Raj Sheldon
```

Navigate into your table

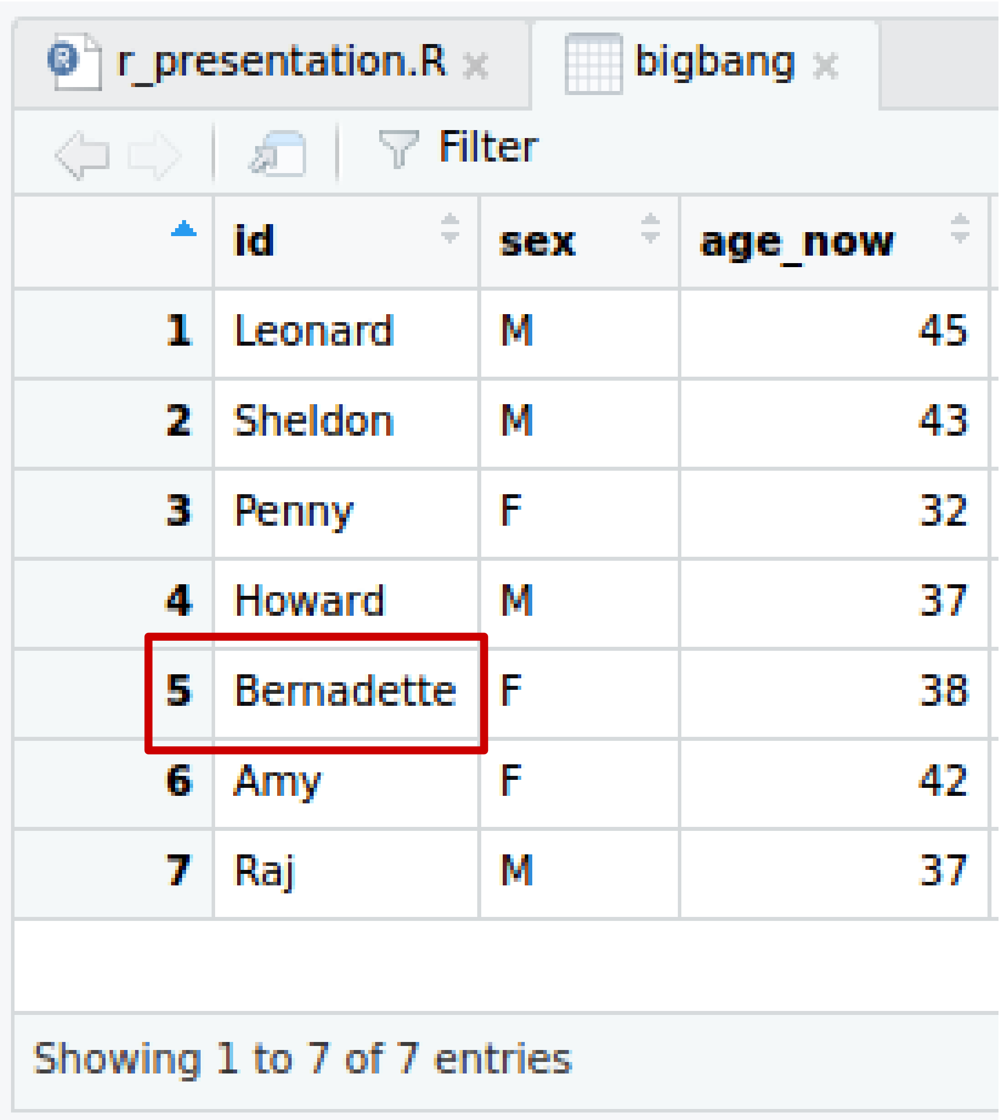


```
#find a specific content: exemple row 5, column 1
bigbang[5,1]
```



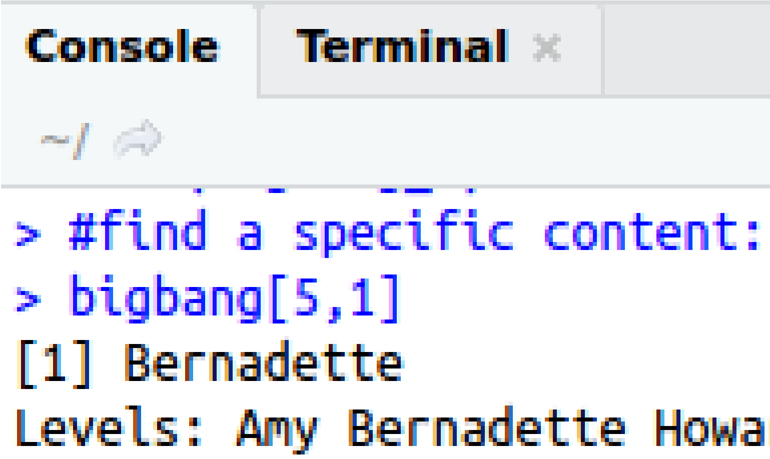
row number

column number



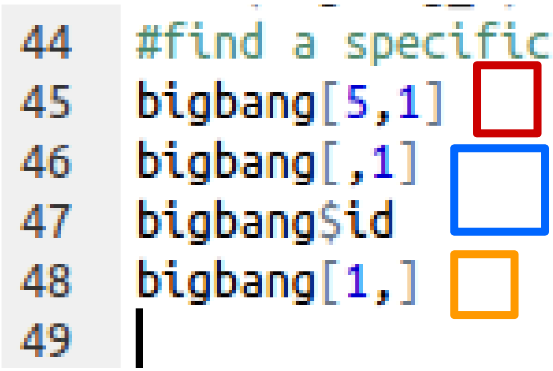
	id	sex	age_now
1	Leonard	M	45
2	Sheldon	M	43
3	Penny	F	32
4	Howard	M	37
5	Bernadette	F	38
6	Amy	F	42
7	Raj	M	37

Showing 1 to 7 of 7 entries

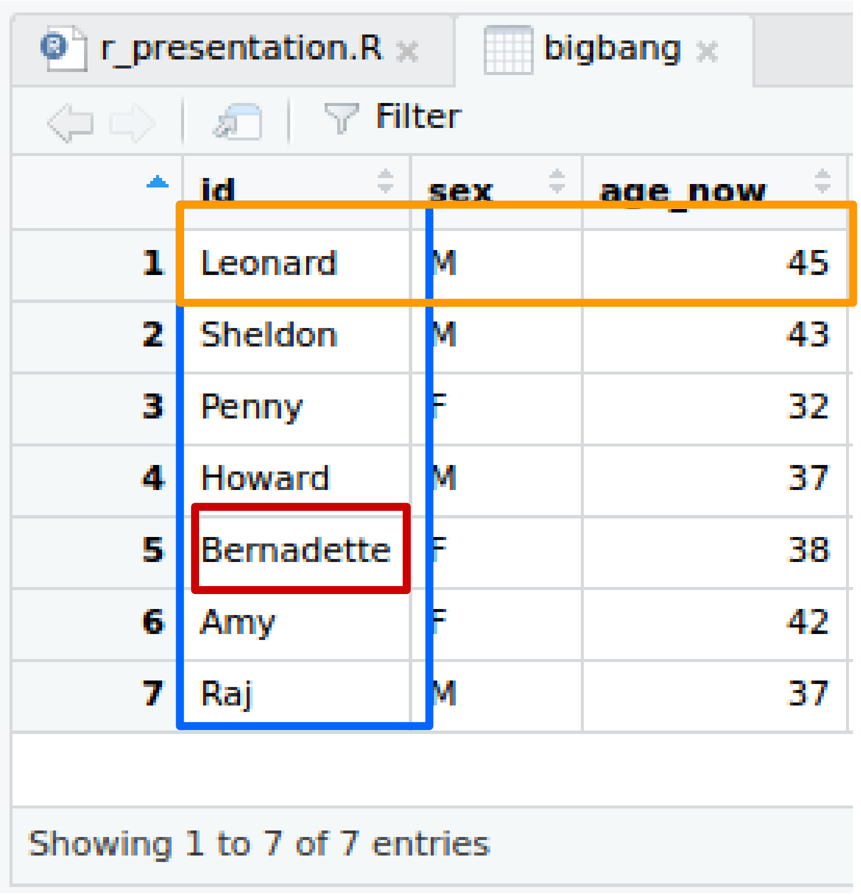


```
> #find a specific content:
> bigbang[5,1]
[1] Bernadette
Levels: Amy Bernadette Howa
```

Navigate into your table



```
#find a specific content: exemple row 5, column 1
bigbang[5,1]
bigbang[,1]
bigbang$id
bigbang[1,]
```

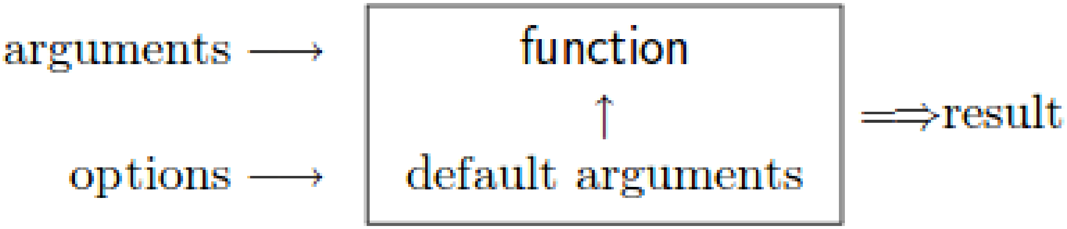


	id	sex	age_now
1	Leonard	M	45
2	Sheldon	M	43
3	Penny	F	32
4	Howard	M	37
5	Bernadette	F	38
6	Amy	F	42
7	Raj	M	37

Showing 1 to 7 of 7 entries

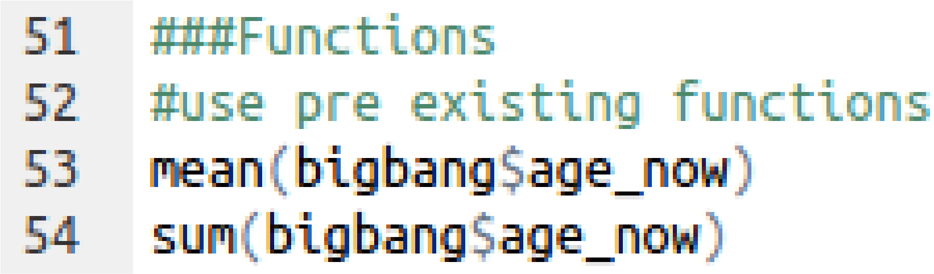
Let’s now use functions

- Functions in R are written as following: function()
- You can specify options and arguments in function of your needs
- The arguments you do not specify will be set by default



E.Paradis, R for beginners, 2005

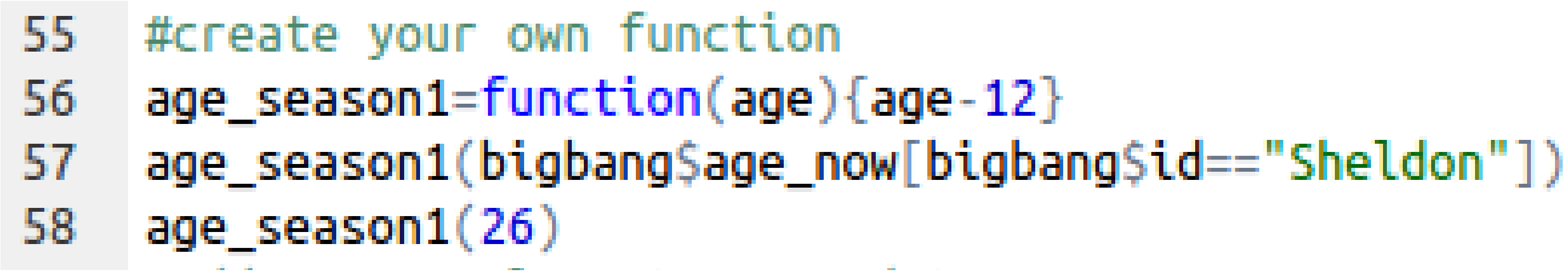
Let’s now use functions



```
###Functions
#use pre existing functions
mean(bigbang$age_now)
sum(bigbang$age_now)
```

You can also create your own function

- Let’s write a function to calculate the age of the actors at the beginning of the serie, so twelve years ago

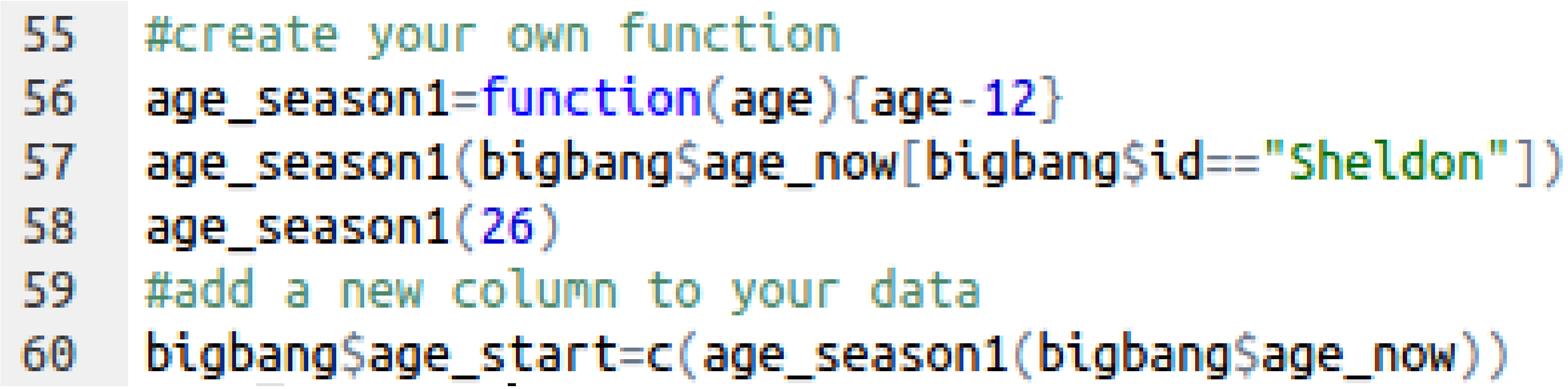


```
#create your own function
age_season1=function(age){age-12}
age_season1(bigbang$age_now[bigbang$id=="Sheldon"])
age_season1(26)
```

Syntax to write your function:
myfunction=function(x) {what to do}

You can also create your own function

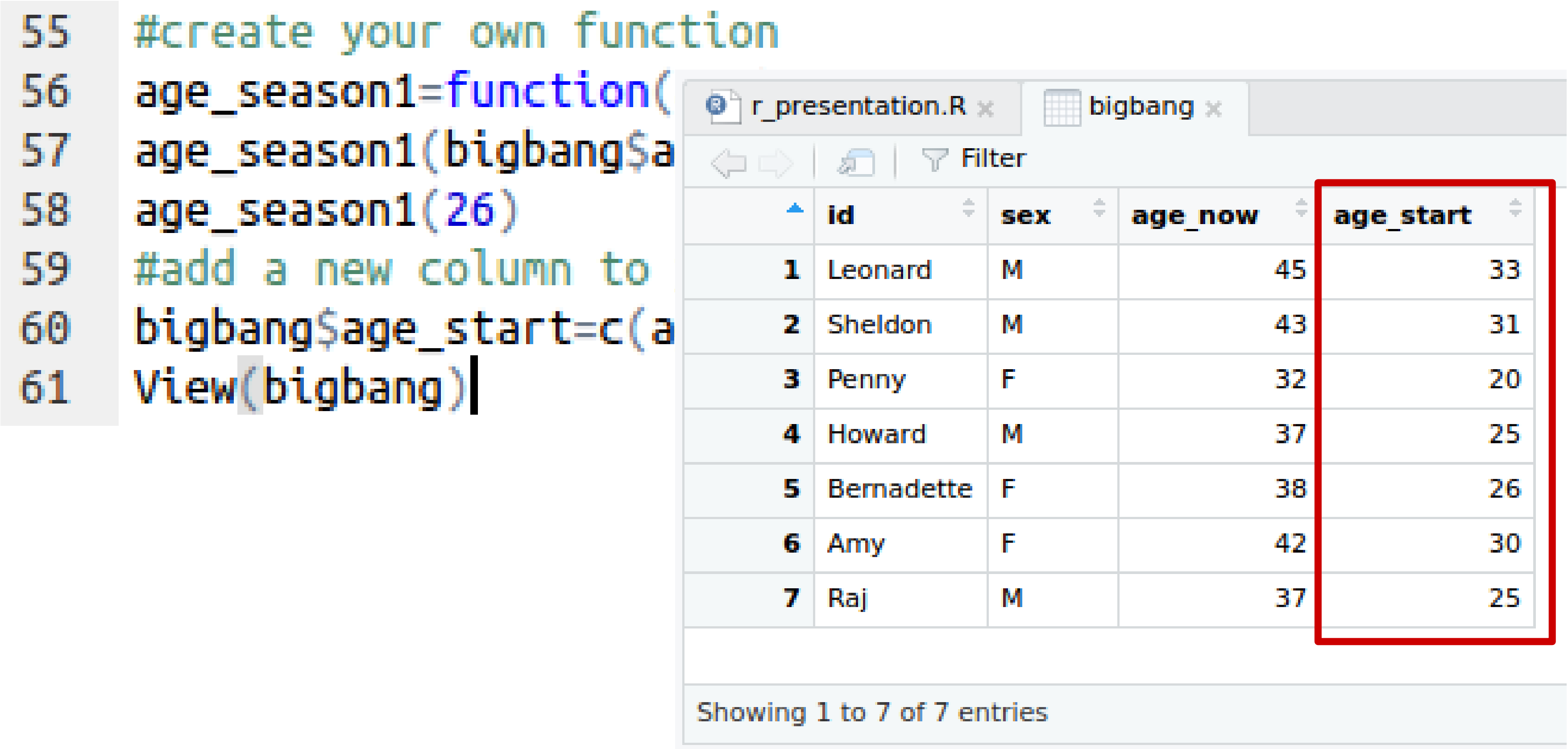
- Let’s write a function to calculate the age of the actors at the beginning of the serie, so twelve years ago



```
#create your own function
age_season1=function(age){age-12}
age_season1(bigbang$age_now[bigbang$id=="Sheldon"])
age_season1(26)
#add a new column to your data
bigbang$age_start=c(age_season1(bigbang$age_now))
```

You can also create your own function

- Let’s write a function to calculate the age of the actors at the beginning of the serie, so twelve years ago

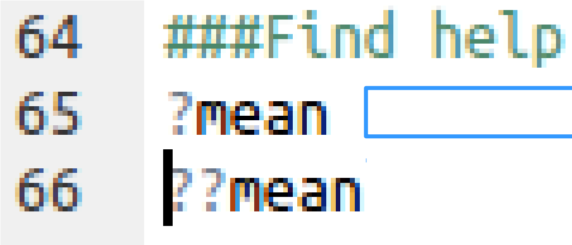


```
#create your own function
age_season1=function(
age_season1(bigbang$a
age_season1(26)
#add a new column to
bigbang$age_start=c(a
View(bigbang)
```

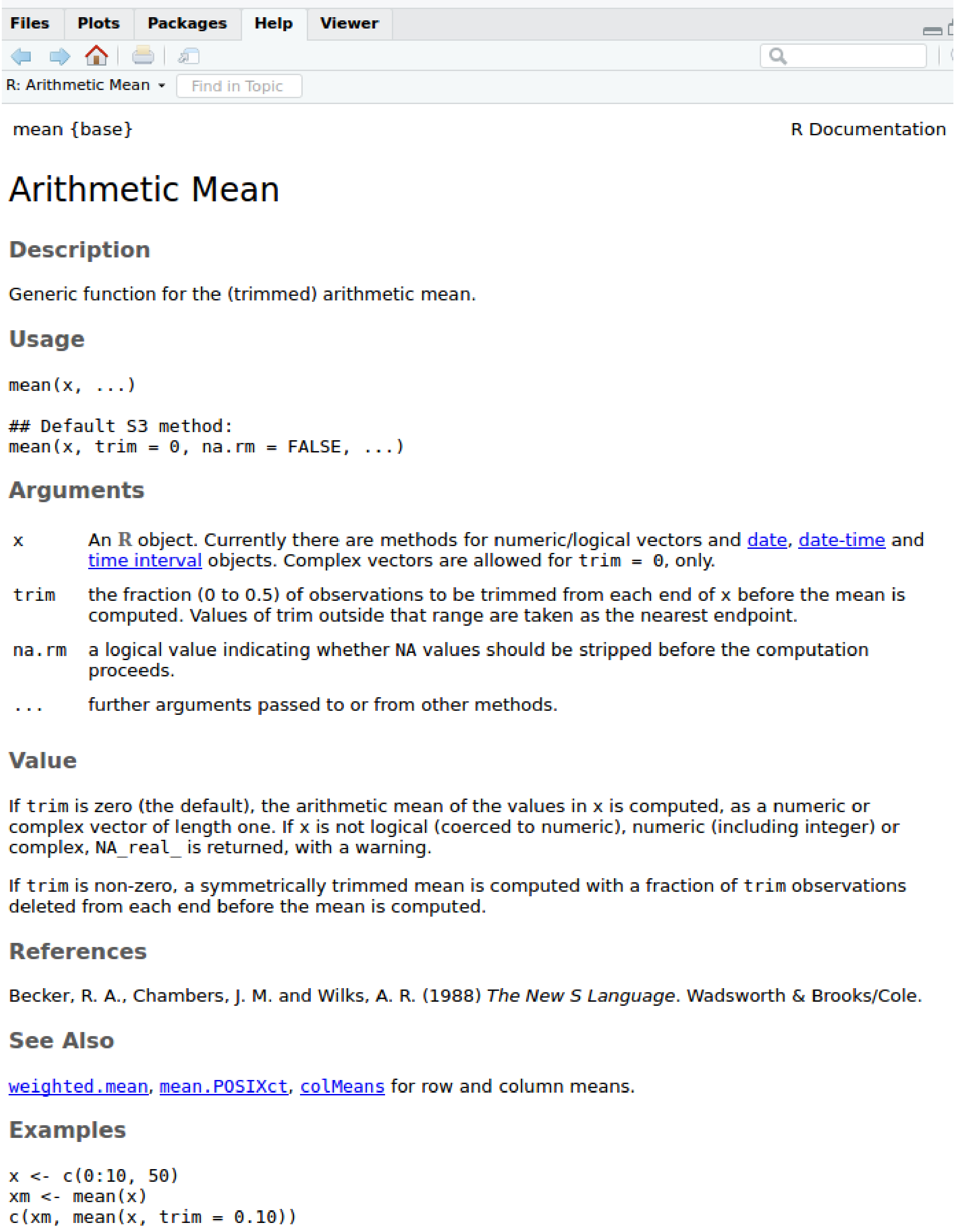
	id	sex	age_now	age_start
1	Leonard	M	45	33
2	Sheldon	M	43	31
3	Penny	F	32	20
4	Howard	M	37	25
5	Bernadette	F	38	26
6	Amy	F	42	30
7	Raj	M	37	25

Showing 1 to 7 of 7 entries

Finding help in R



```
###Find help
?mean
??mean
```



mean {base}

R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

`x` An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.

`trim` the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.

`na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.

`...` further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in x is computed, as a numeric or complex vector of length one. If x is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

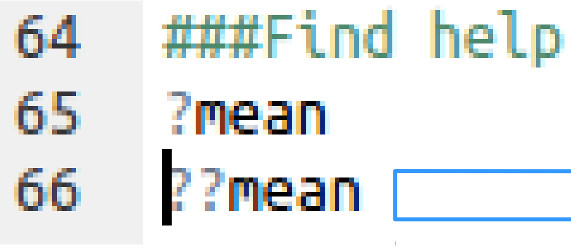
See Also

`weighted.mean`, `mean.POSIXct`, `colMeans` for row and column means.

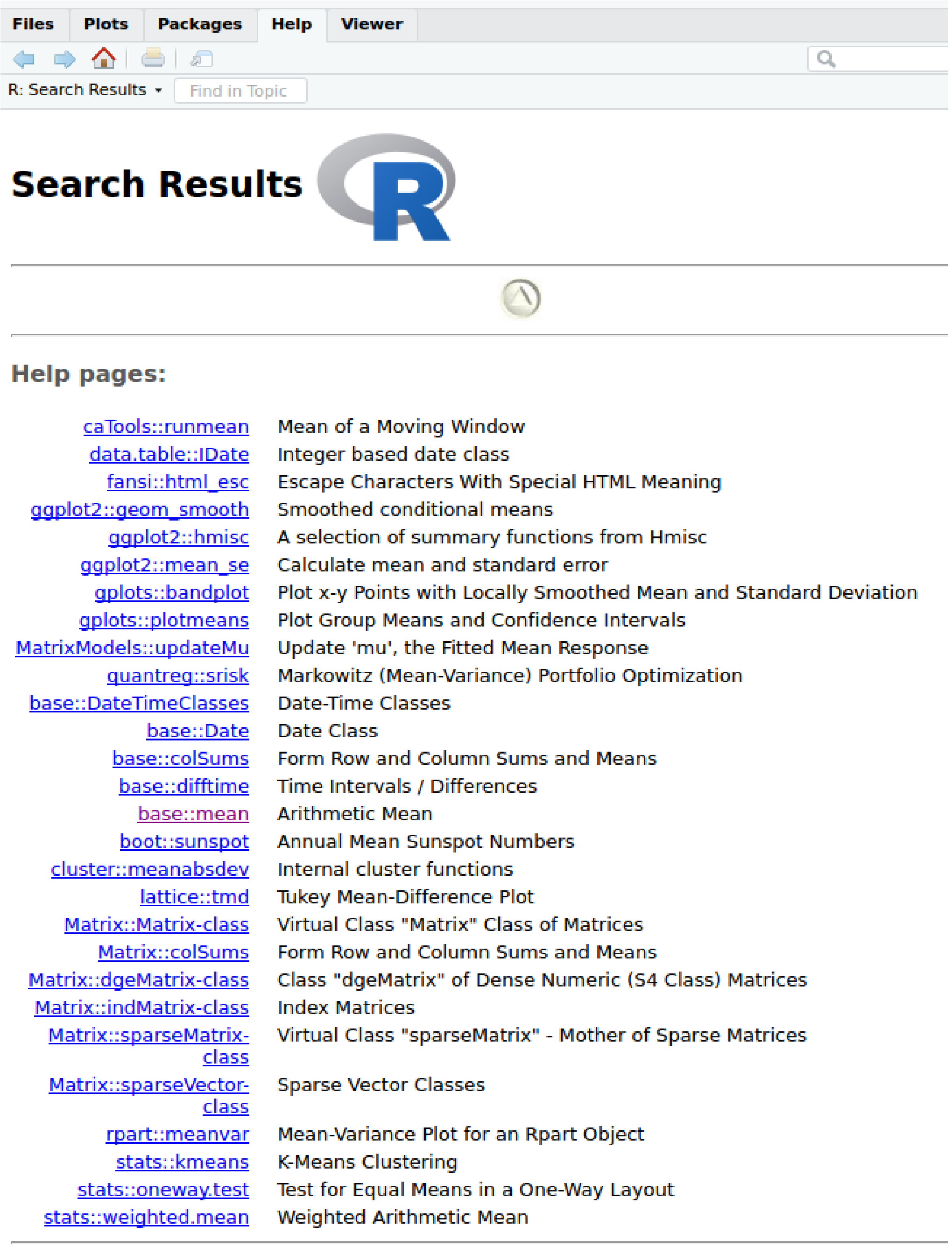
Examples

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
```

Finding help in R



```
###Find help
?mean
??mean
```



Search Results

Help pages:

caTools::runmean	Mean of a Moving Window
data.table::IDate	Integer based date class
fansi::html_esc	Escape Characters With Special HTML Meaning
ggplot2::geom_smooth	Smoothed conditional means
ggplot2::hmisc	A selection of summary functions from Hmisc
ggplot2::mean_se	Calculate mean and standard error
gplots::bandplot	Plot x-y Points with Locally Smoothed Mean and Standard Deviation
gplots::plotmeans	Plot Group Means and Confidence Intervals
MatrixModels::updateMu	Update 'mu', the Fitted Mean Response
quantreg::srisk	Markowitz (Mean-Variance) Portfolio Optimization
base::DateTimeClasses	Date-Time Classes
base::Date	Date Class
base::colSums	Form Row and Column Sums and Means
base::difftime	Time Intervals / Differences
base::mean	Arithmetic Mean
boot::sunspot	Annual Mean Sunspot Numbers
cluster::meanabsdev	Internal cluster functions
lattice::tmd	Tukey Mean-Difference Plot
Matrix::Matrix-class	Virtual Class "Matrix" Class of Matrices
Matrix::colSums	Form Row and Column Sums and Means
Matrix::dgeMatrix-class	Class "dgeMatrix" of Dense Numeric (S4 Class) Matrices
Matrix::indMatrix-class	Index Matrices
Matrix::sparseMatrix-class	Virtual Class "sparseMatrix" - Mother of Sparse Matrices
Matrix::sparseVector-class	Sparse Vector Classes
rpart::meanvar	Mean-Variance Plot for an Rpart Object
stats::kmeans	K-Means Clustering
stats::oneway.test	Test for Equal Means in a One-Way Layout
stats::weighted.mean	Weighted Arithmetic Mean

If you do not know the exact name of the function, use ??

Finding help on the web

- Books/ebooks
  - Emmanuel Paradis – R for beginners
  - Michael Crawley - The R book
  - Andrie de Vries & Joris Meys - R for dummies

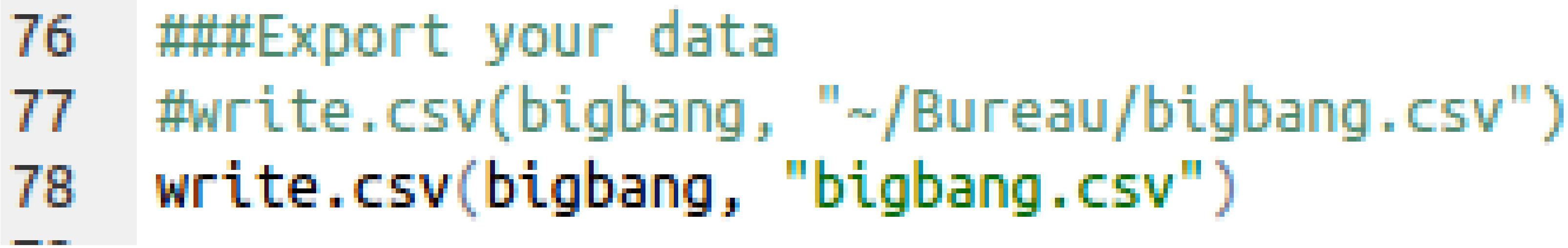
- Websites
  - Statistical Tools for High-Throughput Data Analysis (www.sthda.com)
  - Stackoverflow (http://stackoverflow.com)

- Forums
  - R-bloggers (www.r-bloggers.com)

Do not ask questions on forums without doing preliminary research

Give a sample of data and script to illustrate your problem

Export the data created on R in csv file



```
###Export your data
#write.csv(bigbang, "~/Bureau/bigbang.csv")
write.csv(bigbang, "bigbang.csv")
```

To easily find the pathway, you can right-clic on a document located in the place of interest and clic on properties, the pathway will be indicated

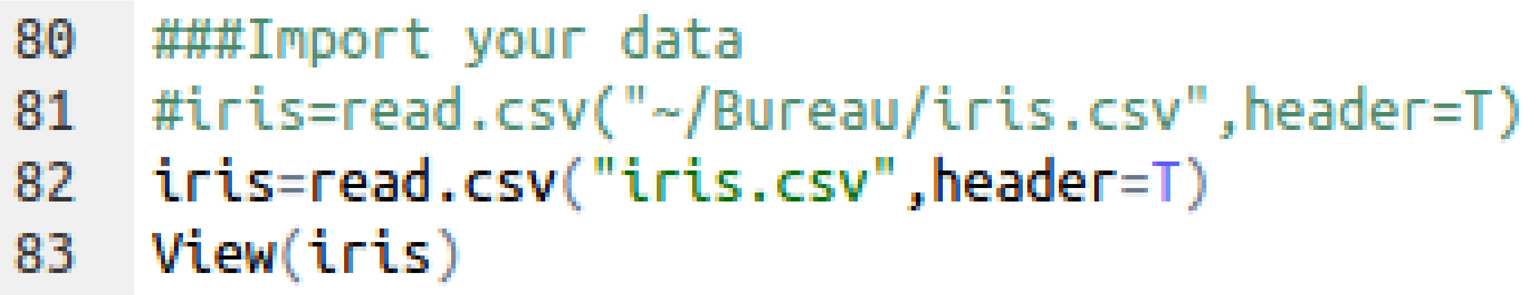
Alternatively to write the pathway, you can set your working directory manually:
Session>Set working directory>choose location

Import your own data in R studio

- A few precautions before to import a document
  - Decimal numbers have to be written using a dot and not a comma: 2.5 instead of 2,5
  - Do not mix empty cases and cases with NA in a same column
  - Replace #VALUE ! (that happens in excell sheet when you apply function on Nas) by NA
  - Convert your excell sheet in .csv format

Import your own data in R studio

- Iris dataset is a free dataset avaiable in R and regularly used in courses and examples

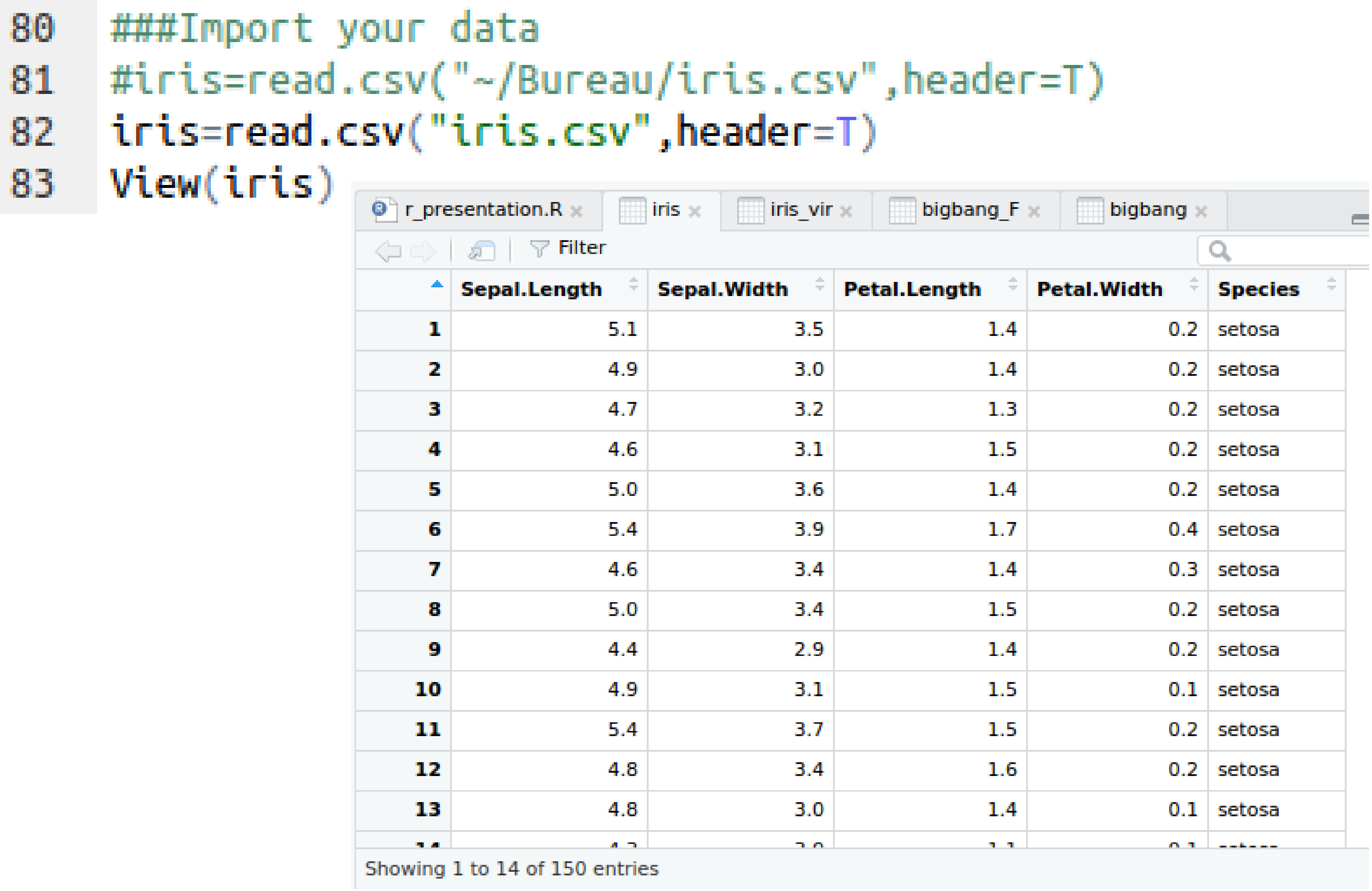


```
###Import your data
#iris=read.csv("~/Bureau/iris.csv",header=T)
iris=read.csv("iris.csv",header=T)
View(iris)
```

To easily find the pathway, you can right-clic on a document located in the place of interest and clic on properties, the pathway will be indicated

If you specified that the column separator of your csv file is tab, add the following argument:
read.csv(‘’path/filename.csv’’, sep= ‘’\t’’)

Import your own data in R studio

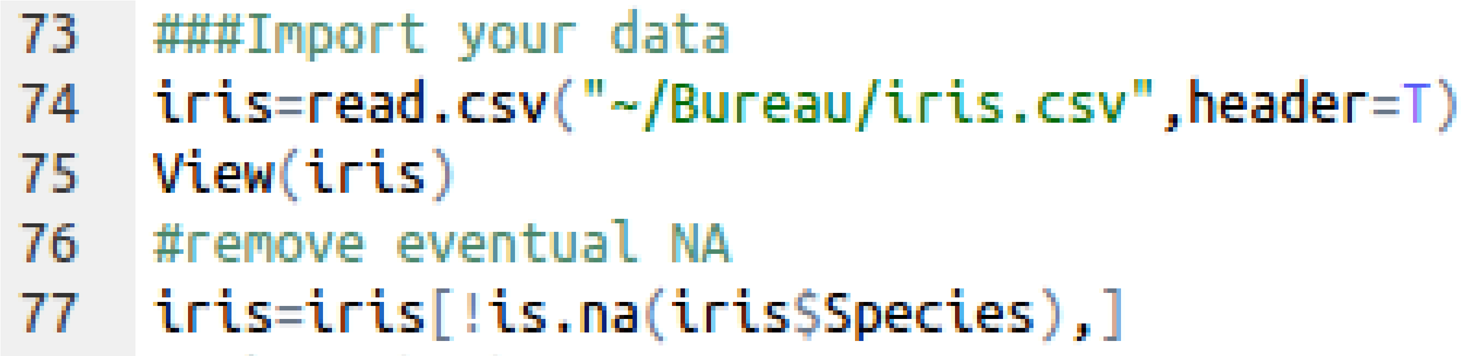


```
###Import your data
#iris=read.csv("~/Bureau/iris.csv",header=T)
iris=read.csv("iris.csv",header=T)
View(iris)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa

Showing 1 to 14 of 150 entries

Remove the eventual NAs

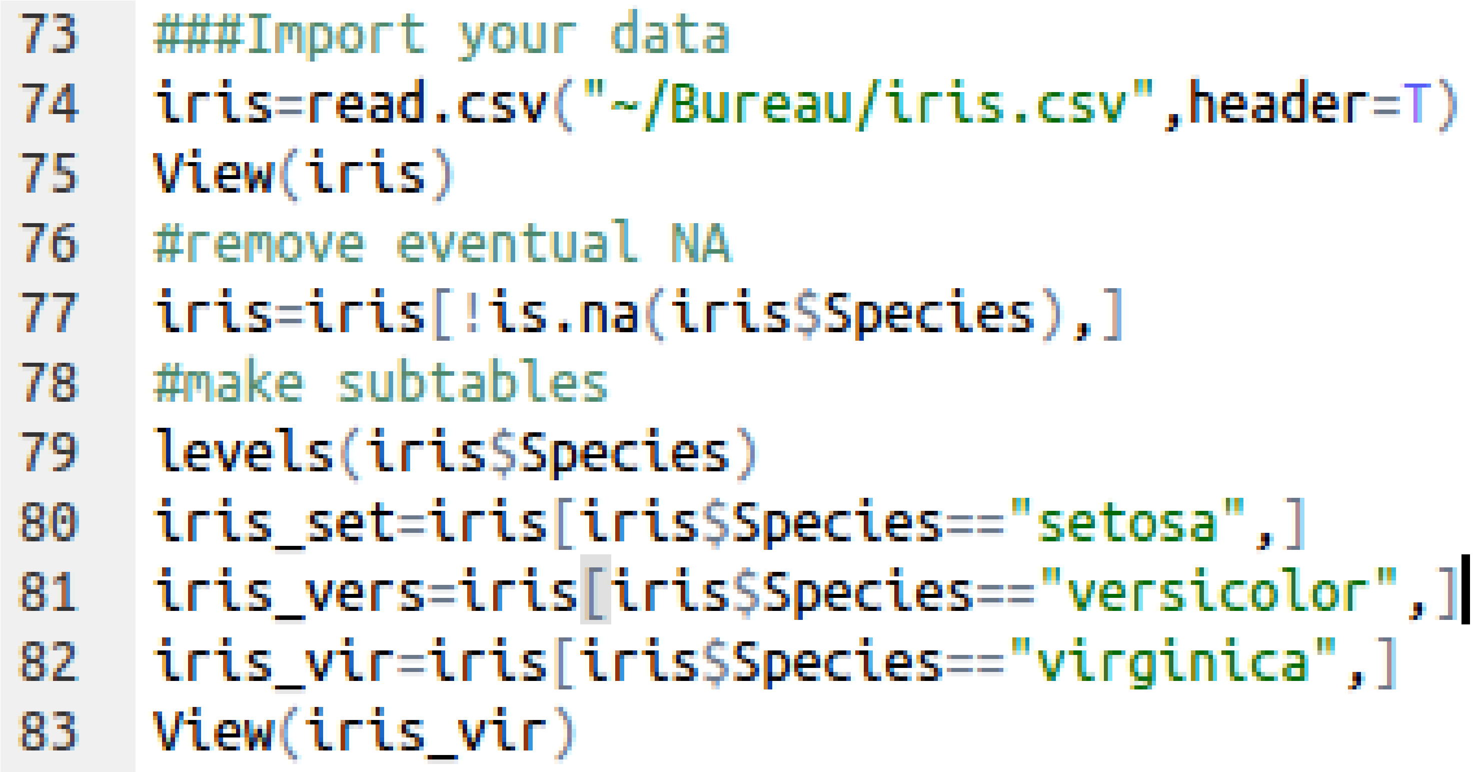


```
###Import your data
iris=read.csv("~/Bureau/iris.csv",header=T)
View(iris)
#remove eventual NA
iris=iris[!is.na(iris$Species),]
```

is.na() return the cases that contain Nas

!is.na() make the contrary: it gives you the cases where there is something else than NA

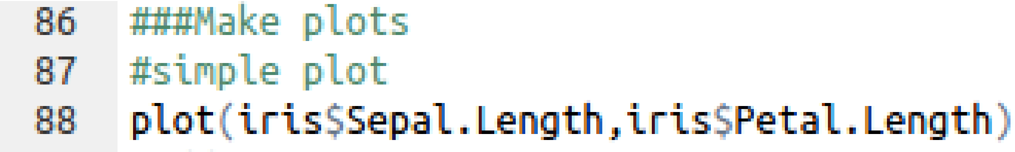
Create sub-tables in function of the specie



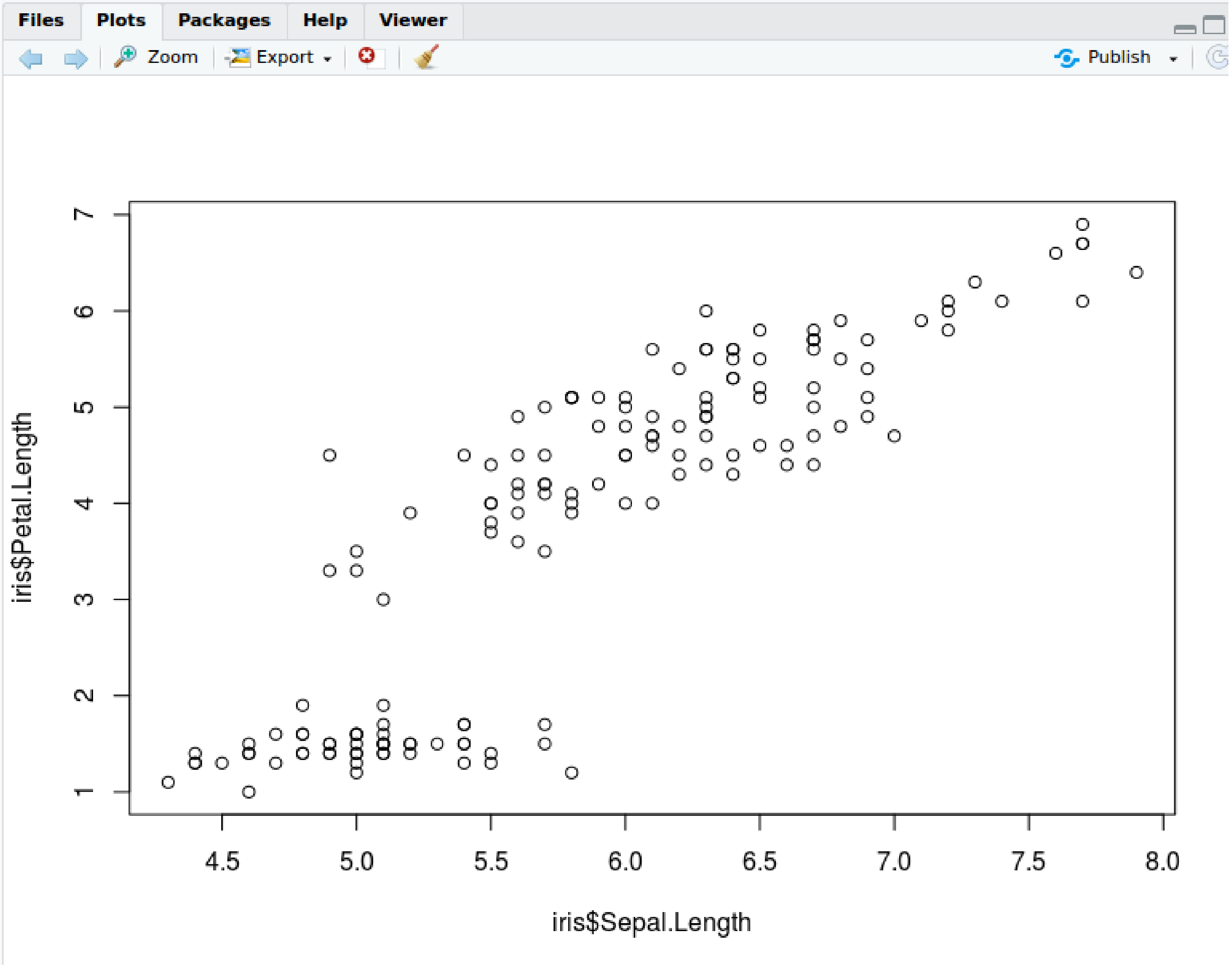
```
###Import your data
iris=read.csv("~/Bureau/iris.csv",header=T)
View(iris)
#remove eventual NA
iris=iris[!is.na(iris$Species),]
#make subtables
levels(iris$Species)
iris_set=iris[iris$Species=="setosa",]
iris_vers=iris[iris$Species=="versicolor",]
iris_vir=iris[iris$Species=="virginica",]
View(iris_vir)
```

As you want a subtable and not a vector, do not forget the comma after the the specie

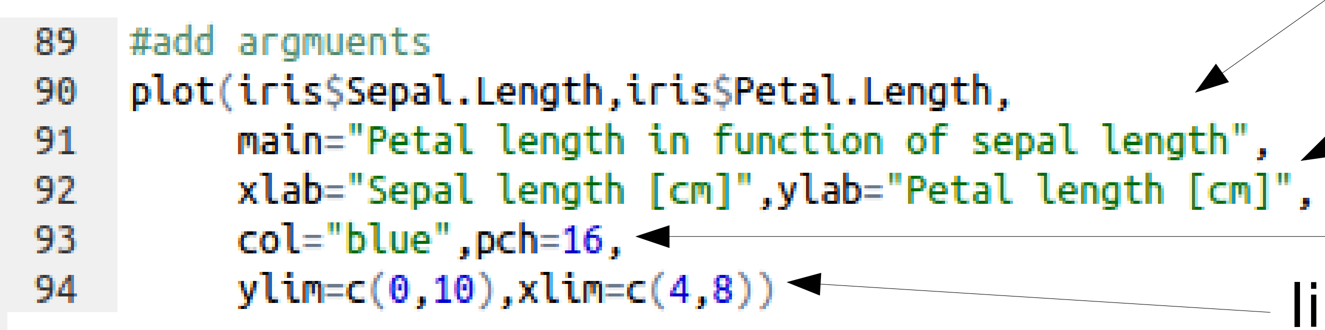
Draw a plot



```
###Make plots
#simple plot
plot(iris$Sepal.Length,iris$Petal.Length)
```



Draw a nicer plot



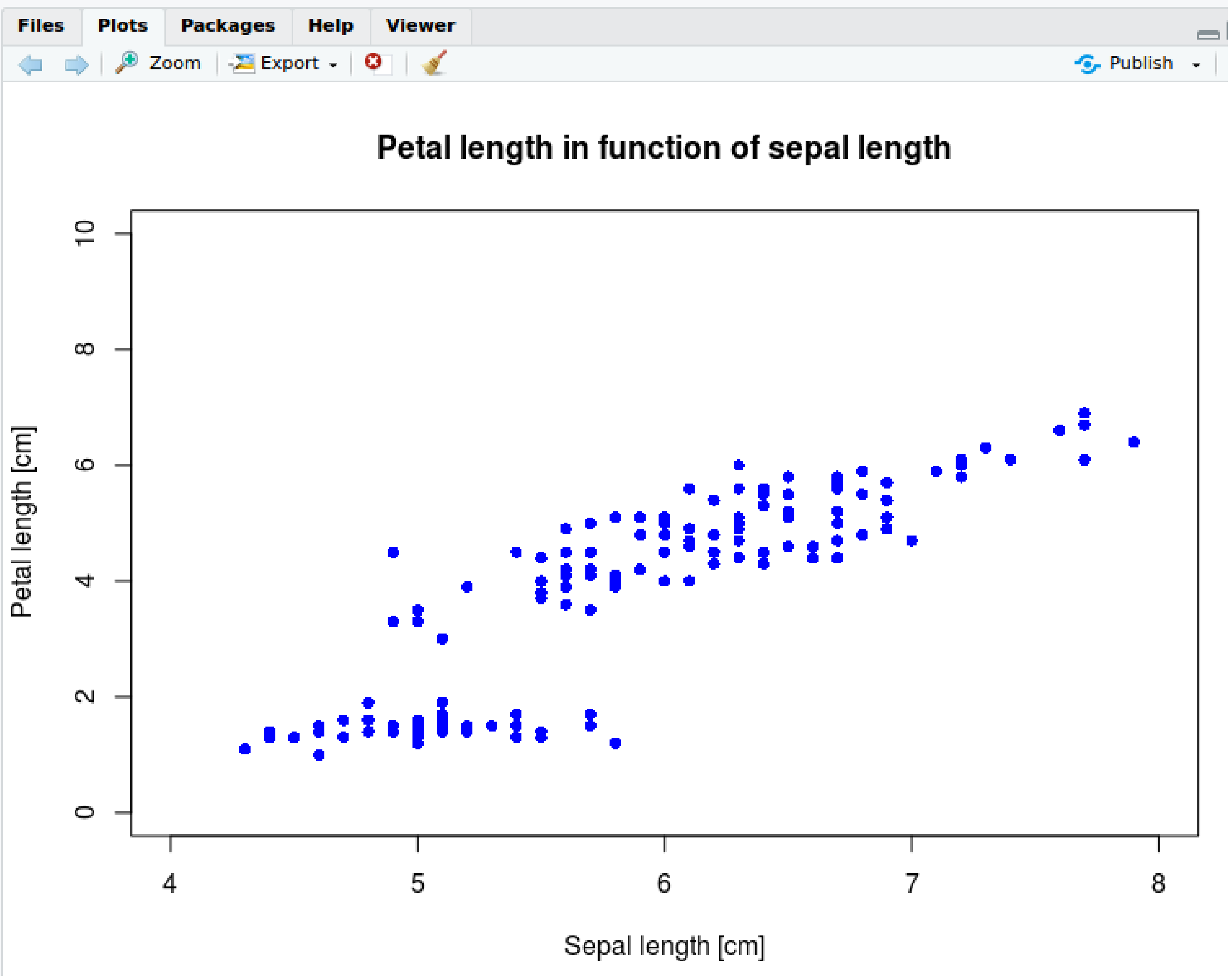
```
#add argmuents
plot(iris$Sepal.Length,iris$Petal.Length,
     main="Petal length in function of sepal length",
     xlab="Sepal length [cm]",ylab="Petal length [cm]",
     col="blue",pch=16,
     ylim=c(0,10),xlim=c(4,8))
```

title

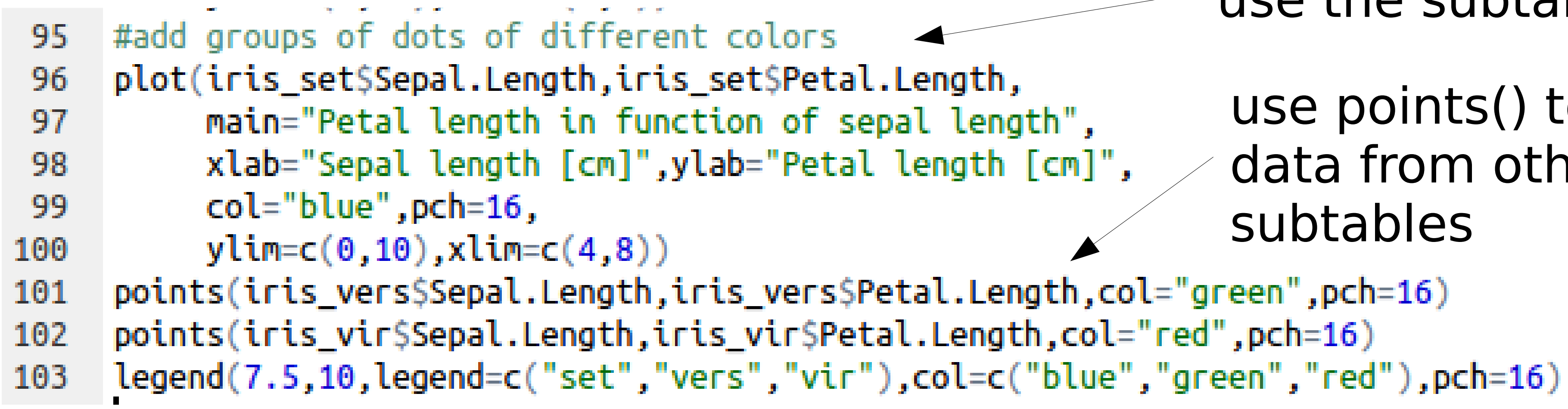
y and x subtitles

color and type of dot

limits of the y and x axis



Draw the nicest plot

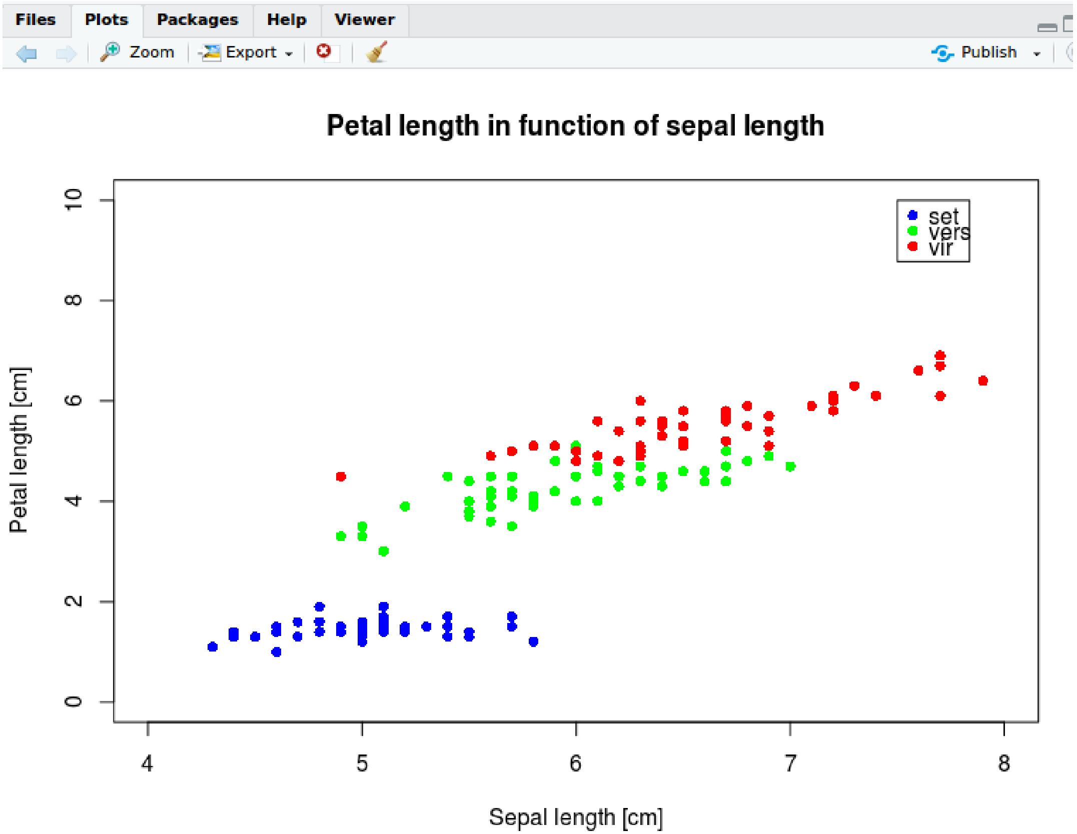


```
#add groups of dots of different colors
plot(iris_set$Sepal.Length,iris_set$Petal.Length,
     main="Petal length in function of sepal length",
     xlab="Sepal length [cm]",ylab="Petal length [cm]",
     col="blue",pch=16,
     ylim=c(0,10),xlim=c(4,8))
points(iris_vers$Sepal.Length,iris_vers$Petal.Length,col="green",pch=16)
points(iris_vir$Sepal.Length,iris_vir$Petal.Length,col="red",pch=16)
legend(7.5,10,legend=c("set","vers","vir"),col=c("blue","green","red"),pch=16)
```

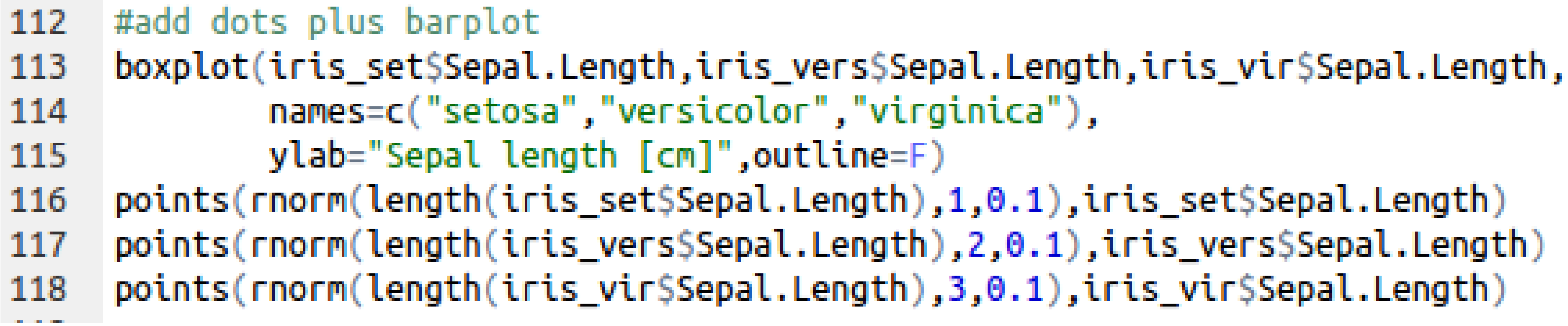
use the subtable

use points() to add data from other subtables

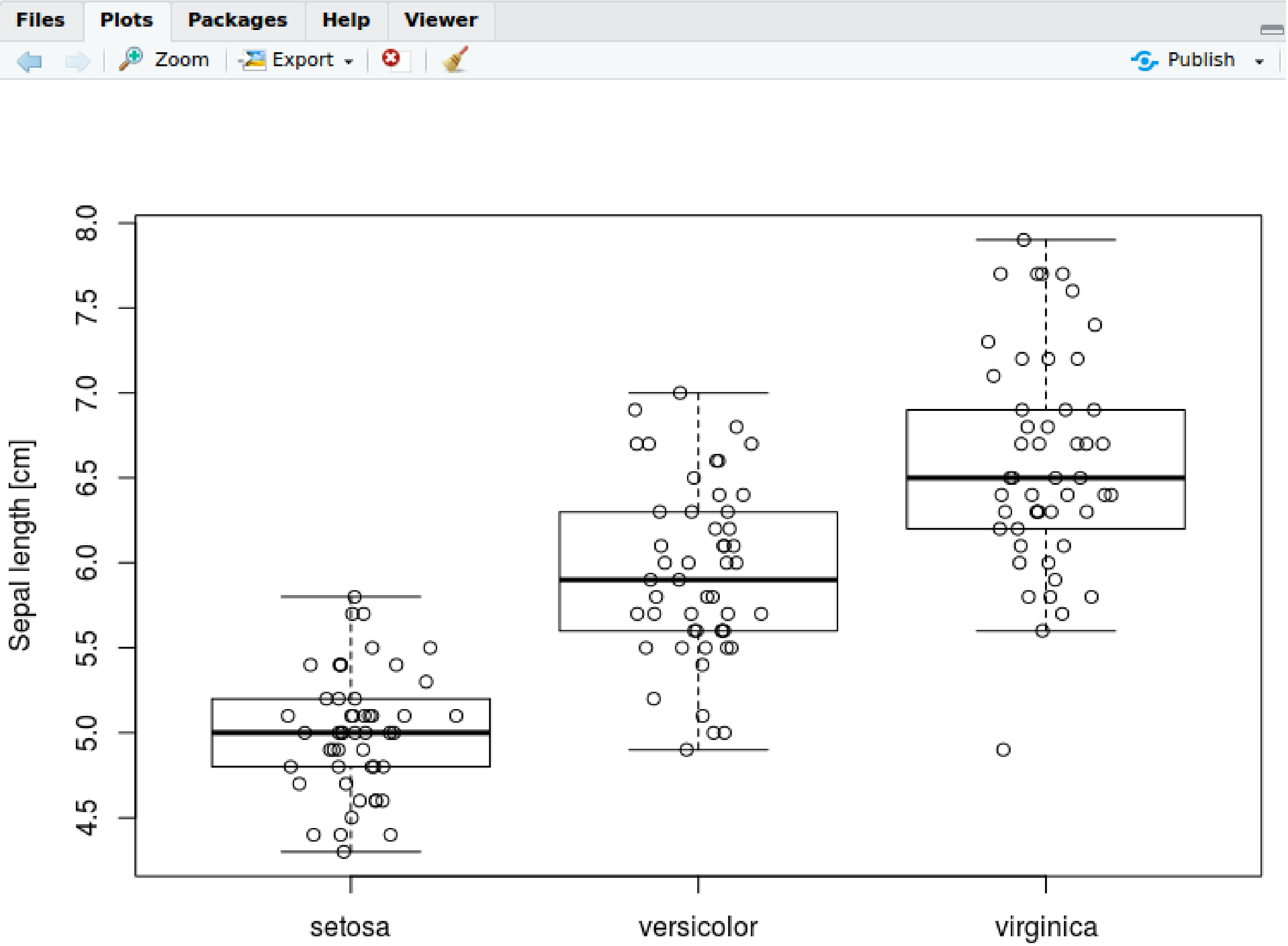
eventually add legend



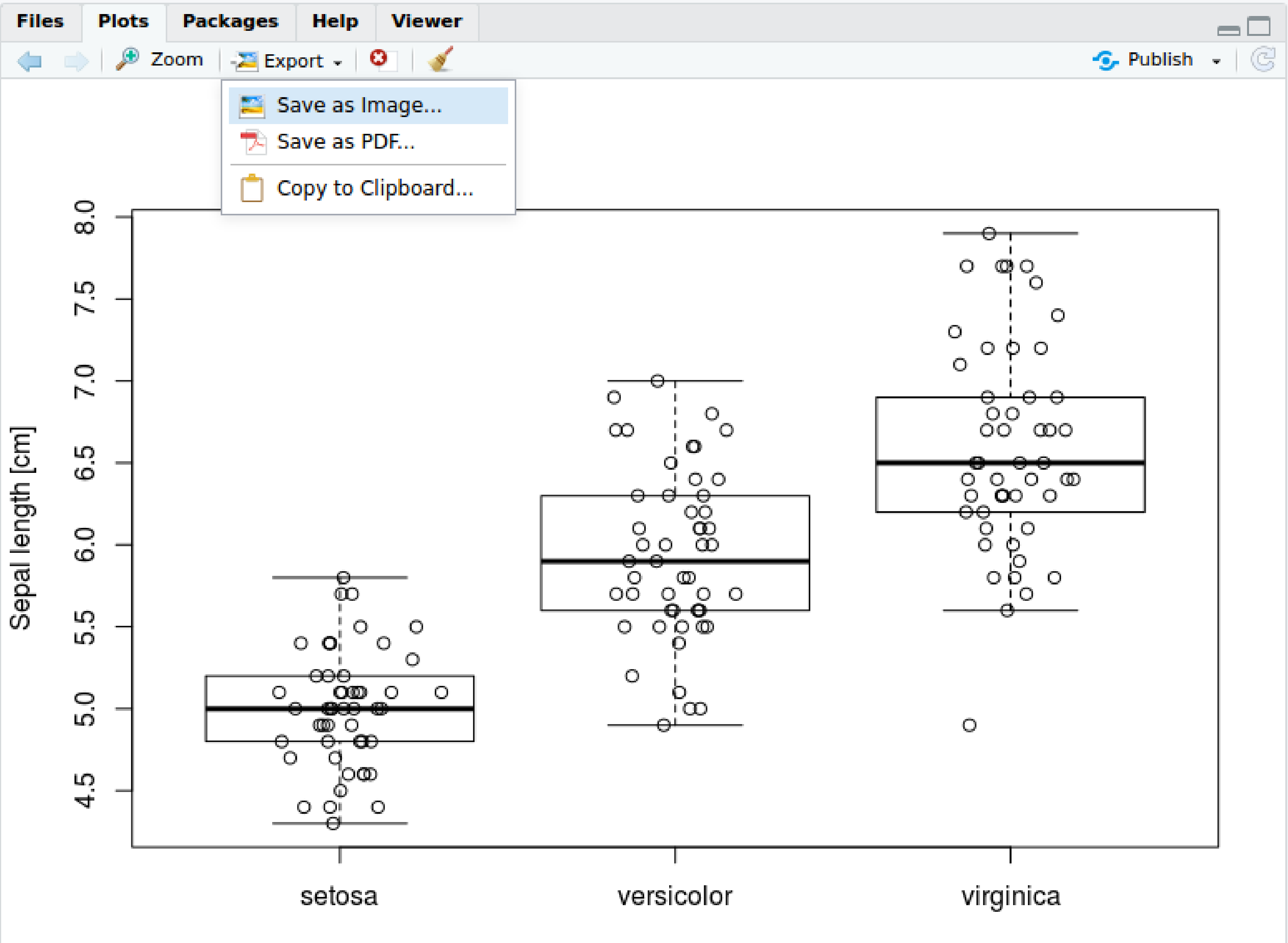
What about a boxplot?



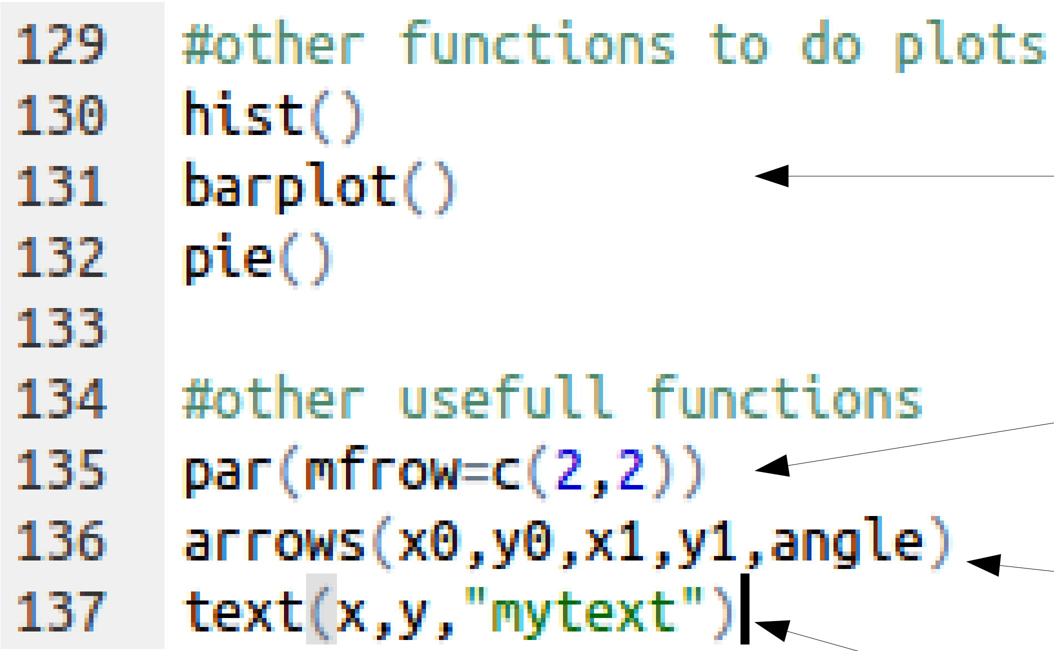
```
#add dots plus barplot
boxplot(iris_set$Sepal.Length,iris_vers$Sepal.Length,iris_vir$Sepal.Length,
        names=c("setosa","versicolor","virginica"),
        ylab="Sepal length [cm]",outline=F)
points(rnorm(length(iris_set$Sepal.Length),1,0.1),iris_set$Sepal.Length)
points(rnorm(length(iris_vers$Sepal.Length),2,0.1),iris_vers$Sepal.Length)
points(rnorm(length(iris_vir$Sepal.Length),3,0.1),iris_vir$Sepal.Length)
```



Save your plot



Other usefull function linked to plots



```
#other functions to do plots
hist()
barplot()
pie()

#other usefull functions
par(mfrow=c(2,2))
arrows(x0,y0,x1,y1,angle)
text(x,y,"mytext")
```

other kinds of plots

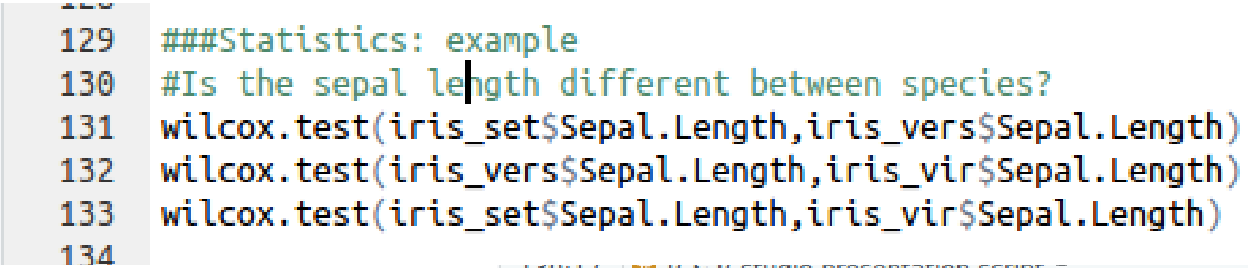
split your plot window to see several plots (here 2x2)

add arrows or lines (angle=0) on your plots

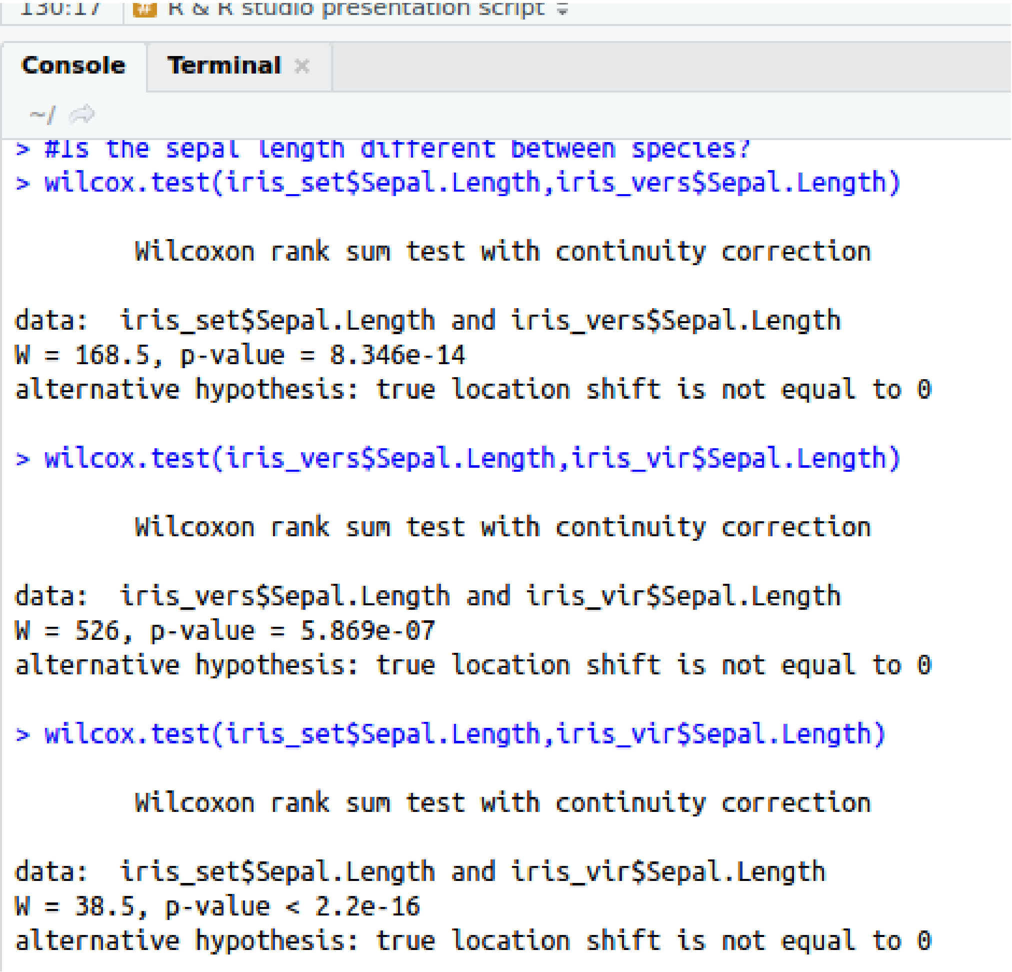
add text on your plot

When you add elements to your plot, that are not between the axis, add the argument xpd=T to see it

A few lines of statistics: is the sepal length different between species?



```
###Statistics: example
#Is the sepal length different between species?
wilcox.test(iris_set$Sepal.Length,iris_vers$Sepal.Length)
wilcox.test(iris_vers$Sepal.Length,iris_vir$Sepal.Length)
wilcox.test(iris_set$Sepal.Length,iris_vir$Sepal.Length)
```



```
> #Is the sepal length different between species?
> wilcox.test(iris_set$Sepal.Length,iris_vers$Sepal.Length)

        Wilcoxon rank sum test with continuity correction

data:  iris_set$Sepal.Length and iris_vers$Sepal.Length
W = 168.5, p-value = 8.346e-14
alternative hypothesis: true location shift is not equal to 0

> wilcox.test(iris_vers$Sepal.Length,iris_vir$Sepal.Length)

        Wilcoxon rank sum test with continuity correction

data:  iris_vers$Sepal.Length and iris_vir$Sepal.Length
W = 526, p-value = 5.869e-07
alternative hypothesis: true location shift is not equal to 0

> wilcox.test(iris_set$Sepal.Length,iris_vir$Sepal.Length)

        Wilcoxon rank sum test with continuity correction

data:  iris_set$Sepal.Length and iris_vir$Sepal.Length
W = 38.5, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0
```

To conclude...

- Basic tools
  - Create, use, visualise vectors and data frames
  - Use and create functions
  - Find help
  - Draw and save plots and boxplots
  - Import and export your data
  - A few lines of statistics

- Many more possibilities in function of your needs
  - automatisation of repetitive tasks: for boucle and apply()
  - specific packages

- Practise !