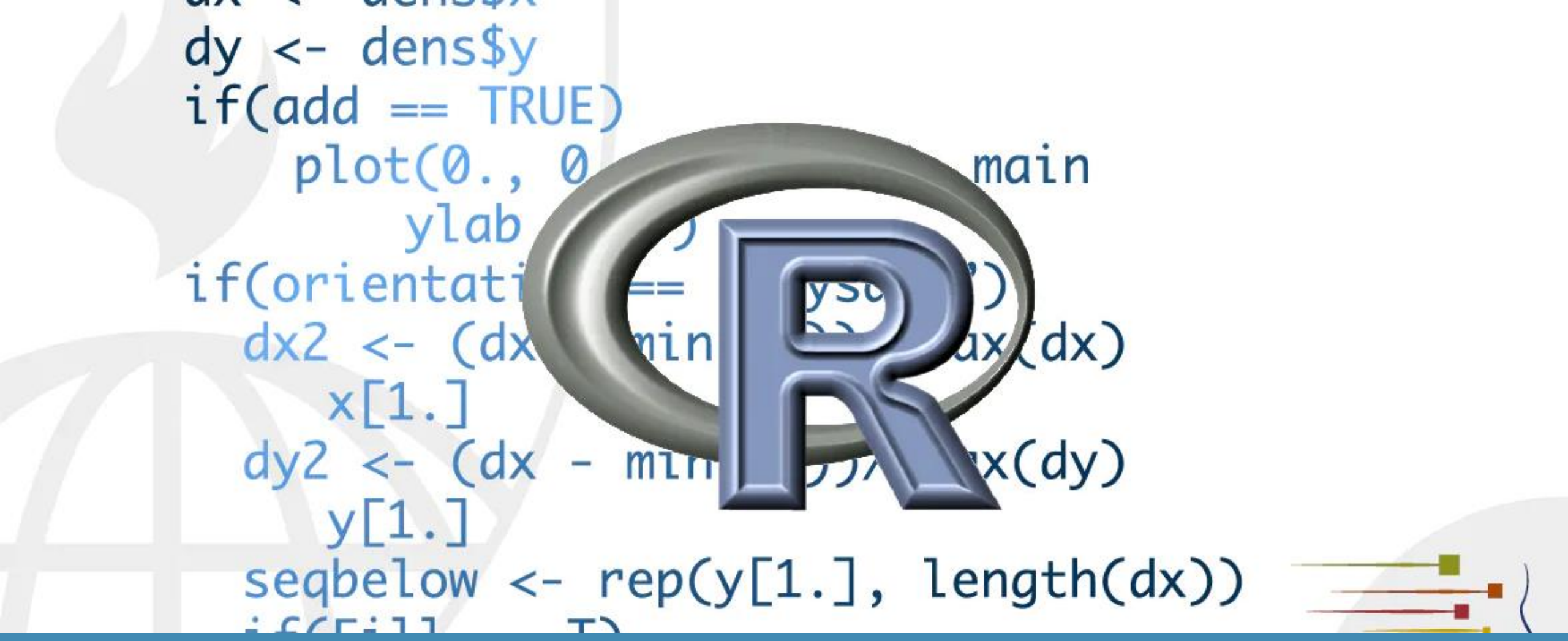


Introduction to R

Eroteida Sánchez García
esanchezg@aemet.es

Agencia Estatal de Meteorología

Contents

References:

- **An Introduction to R (**W. N. Venables, D. M. Smith and the R Core Team)

  http://cran.r-project.org

- **An Introduction to R (***Alex Douglas, Deon Roos, Francesca Mancini, Ana Couto & David Lusseau*)

  https://intro2r.com/

Connecting with server ‘medcoftr.aemet.es’

1. Open the System Window

2. Run this command:

   $ ssh –X trYYY@medcoftr.aemet.es

   (trYYY is the username)

3. Introduce your password

First session in R: console mode

1. Change to our working directory for this session ‘Rintro’:
   $ cd Rintro

2. Start the R program with the command:
   $ R

3. At this point R commands may be issued.

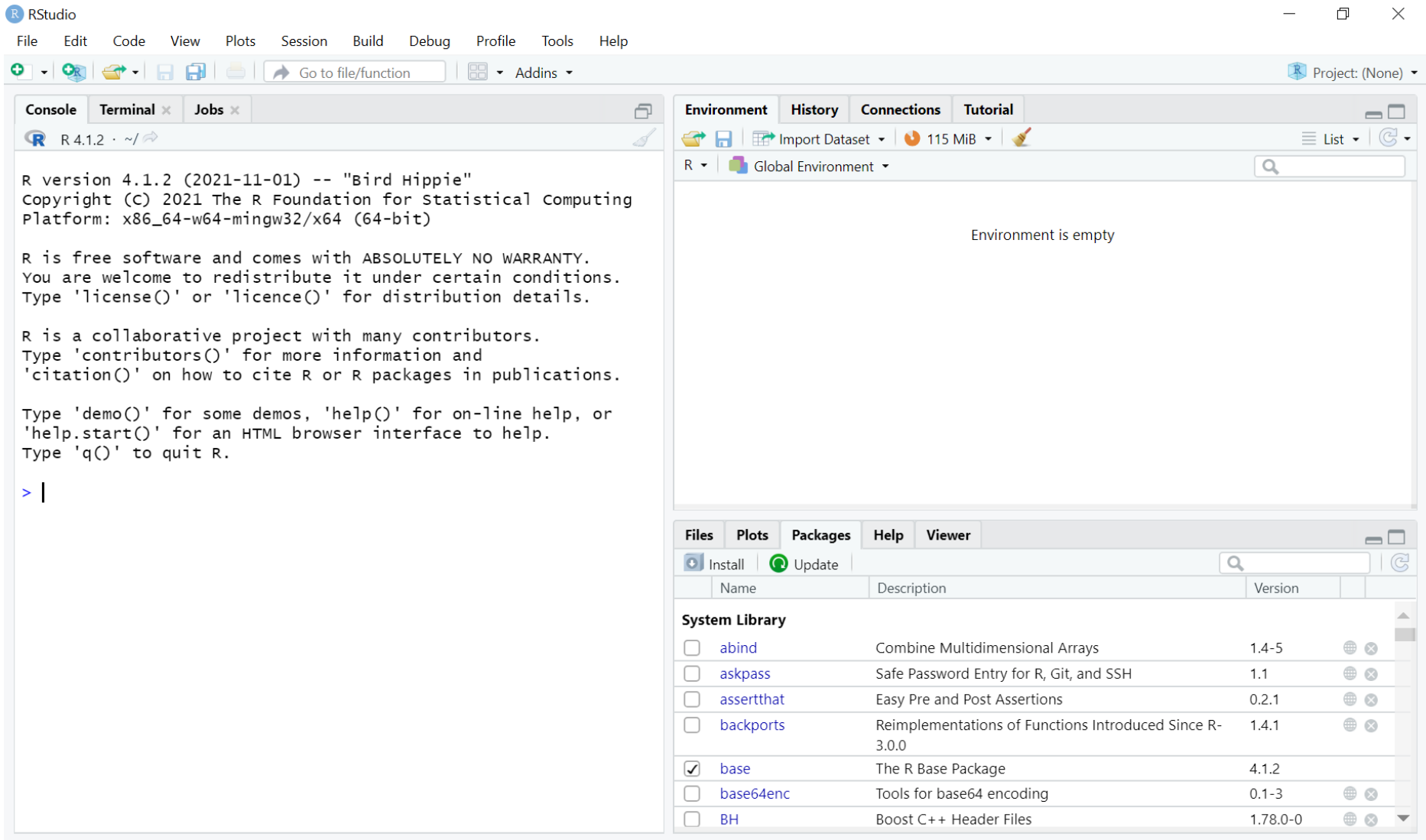
4. To quit the R program the command is:
   > q()

Open a session in R with RStudio

1. Start the RStudio with the command:

   $ rstudio

If you have done the connection to the server with Windows System, you can't open Rstudio.



Getting help with R

R HAS AN EXCELLENT HELP SYSTEM.

IT CAN BE INVOKED IN SEVERAL WAYS:

- help('lm'), ?lm
- help.search('regression'), ??regression
- RSiteSearch('regression') (http://search.r-project.org/)

OUTPUT EXAMPLE: ?median or help('median')

Median Value

Description

Compute the sample median.

Usage

```
median(x, na.rm = FALSE, ...)
```

Arguments

x	an object for which a method has been defined, or a numeric vector containing the values whose median is to be computed.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	potentially further arguments for methods; not used in the default method.

Details

This is a generic function for which methods can be written. However, the default method makes use of `is.na`, `sort` and `mean` from package **base** all of which are generic, and so the default method will work for most classes (e.g., "Date") for which a median is a reasonable concept.

Value

The default method returns a length-one object of the same type as `x`, except when `x` is logical or integer of even length, when the result will be double.

If there are no values or if `na.rm = FALSE` and there are `NA` values the result is `NA` of the same type as `x` (or more generally the result of `x[FALSE][NA]`).

References

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

See Also

`quantile` for general quantiles.

Examples

```
median(1:4)              # = 2.5 [even number]
median(c(1:3, 100, 1000))  # = 3 [odd, robust]
```

[Package *stats* version 4.0.2 Index]

Objects and Classes in R

OBJECT : X

- An object is a store of information: It has an identifier and a structure to store the information.
- We act on objects by applying functions to them.
- Generally we create an object by using the designation operator (<-):
  - x <- 4
- Everything in R is an object (data, analysis results, and even functions).
- The objects are stored in memory for later use.
  - At any time we can obtain a list of the objects stored in memory:
    - ls()
  - We can determine whether or not an object exists in memory:
    - exists(x)
  - Objects can be removed from memory:
    - rm(x)

Objects and Classes in R

CLASS(X)

- Any data in R must belong to one of the following fundamental types or classes: logical, integer, numeric, complex or character.
- The ***logical*** type allows to store logical truth values (T or TRUE, F or FALSE):
- The ***integer*** type allows storing integers (x <- as.integer(2))
- The ***numeric*** type allows storing real numbers (x <- 2.5)
- The ***complex*** type allows you to store complex numbers (x <- 1i)
- The ***character*** type allows storing text strings; quotes (", ') are used to delimit character strings: x<- 'hello, Dolly'
- Some special cases:
  - missing data (NA): x <- c(1:4,NA,6:10)
  - infinite value (Inf) : x <- 5/0
  - Not a Number (NaN): x<- Inf –Inf

Operators

ARITHMETIC:

+	addition	5 + 2
-	subtraction	5 - 2
*	multiplication	5 * 2
/	division	5 / 2
^	power	5 ^2
%%	module	5 %% 2
%/%	division of integers	5 %/% 2

COMPARATIVES

<	less than	5 < 2
>	greater than	5 > 2
<=	less than or equal to	5 <= 2
>=	greater than or equal	5 >= 2
==	equal to	5 == 2
!=	different from	5 =! 2

LOGICAL:

!	NO logical	! x
&, &&	AND logical	x & y
|, ||	OR logical	x | y
xor()	OR exclusive	xor(x, y)

Vectors

- Vectors (vector) are sequences of elements of the same type.
- We can refer to specific elements of the vector by means of an index

```
x <- c('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec')
x
is.vector(x)
length(x)
x[1]
a <- 1; x[a]
x[2:4]
x[c(1,3,3,5)]
x[-1]
x[x=='Mar']
x[x!='Mar']
```

Vectors

- Sort and rank:
  - rev(x)
  - sort(x)
  - rank(x)
  - order(x)
- Search:
  - which(x==max(x))
  - w <- which(x==max(x)); x[w]
- Vector combination:
  - y <- c('Dec','Jan','Feb').
  - union(x,y)
  - intersect(x,y)
  - setdiff(x,y)
- Extreme values:
  - max(x)
  - min(x)
  - range(x)
- Sampling:
  - sample(x, 7)
  - sample(x, 7, replace=T)

Matrices and Arrays

- Matrices (matrix) allow data to be stored in two-dimensional tables.

- The data must be of the same type, and all columns and rows must be of the same size.

- A matrix can be understood as a table. Also, as a concatenation by columns of vectors of the same length and type.

  - y <- matrix(1:9, ncol=3)
  - y
  - is.matrix(y)

- Concatenation of vectors by columns and by rows:

  - y <- cbind(c(1:7), c(8:14))
  - cbind(y, y[,1])
  - rbind(y, y[1,])

Matrices and Arrays

- Dimensions of a matrix:
  - dim(y)

- Selection of specific parts of a matrix:
  - y[2,1]
  - y[,1]
  - y[1,]
  - y[,1:2]

- Matrix transpose:
  - t(y)

- Arrays are matrices of more than two dimensions:
  - z <- array(1:42, dim=c(7,3,2))
  - dim(z)
  - z[,,1]

- Names for rows and columns:
  - rownames(y) <- c('A','B','C')
  - colnames(y) <- c('a','b','c')

- Diagonal element:
  - y_diag <- diag(y)

- Matrix operations
  - mat1 <- matrix(1:4, nrow=2)
  - mat2 <- matrix(2:5, nrow=2)
  - mat1 + mat2
  - mat1 * mat2 # element by element
  - mat1 %*% mat2 # matrix multiplication

Factors, Dataframes and Lists

- Factors are vectors of categorical variables. In addition values themselves, they contain the information of the different possible levels of the factor:

```
x <- factor(c('A','C','C','C','A','A','C'), levels = c('A','B','C'))
x
x[1]
levels(x)
```

- Data frames are tables made up of one or more vectors and/or factors of the same length, but which can be of different types:

```
y <- data.frame(a=c('x','y','z'), b=1:3, c=c(T,T,F))
is.data.frame(y)
y[,1]
names(y)
y$a
y$a[2]
```

Factors, Dataframes and Lists

- Every R installation contains several predefined datasets that are used in various examples (e.g. iris dataset as data frame:

  - iris
  - head(iris)
  - summary(iris)

- Lists can contain any type of object (including other lists). The objects do not have to be of the same type, nor do they have to be of the same length.

  - z <- list(estation = c(‘Paris’, ‘Madrid’), months = c(“jan”, “feb”, “mar”), data=c(1,2,3,4,5))
  - is.list(z)
  - names(z)
  - z$months
  - z$months[1]
  - z[[2]]
  - z[[2]][1]

Functions

MATHEMATICAL FUNCTIONS

Function	Description
abs(x)	Absolute value
sqrt(x)	Square root
ceiling(x)	First integer value greater than x
floor(x)	First integer value less than x
trunc(x)	Truncated integer value of x
round(x,n)	Rounding of x to n decimal digits
signif(x,n)	Rounding of x to n significant digits
sin(x), cos(x), tan(x)	Sine, cosine and tangent of x
asin(x), acos(x), atan(x)	Sine arc, cosine arc and tangent arc of x
log(x,n)	Logarithm of x in base n
log(x)	Neperian logarithm (base e) of x
log10(x)	Decimal logarithm of x
exp(x)	Exponential function

Functions

STATISTICAL FUNCTIONS

Function	Description
mean(x)	Average of x
sd(x), var(x)	Standard deviation and variance of x
range(x)	Range of x
min(x), max(x)	Minimum and maximum values of
median(x)	Median (50th percentile) of x
quantile(x,p)	Quantile or probability quantiles p of x
summary(x)	Generates a summary of the object x
sum(x)	Sum of the values of xdiff(x,n
scale(x)	Centring or standardising

Functions

GRAPHICAL FUNCTIONS

Function	Description
plot(x)	Creates a graph from the object x
plot(x,y), plot(y~x)	Creates an x-y scatter plot
points(x), lines(x)	Adds a point or line plot to the previous graph
barplot(x)	Creates a barplot from the frequency table x
dotchart(x)	Creates a dot plot from the x table
pie(x)	Creates a pie chart
hist(x)	Creates a frequency histogram of x
boxplot(x)	Creates a boxplot of x

Functions

PROBABILITY FUNCTIONS

Prefix	Description
d	Probability density
p	Distribution (cumulative probability)
q	Quantiles
r	Random number generation

Function	Description
norm	Normal or Gaussian
lnorm	Lognormal
chisq	Chi-square
f	F
logis	Logistic
binom	Binomial
multinom	Multinomial
……	and much more

Functions

FUNCTIONS FOR CHARACTER STRINGS

Function	Description
nchar(x)	Number of characters in x
substr(x,a,b)	Extracts the characters of x between positions a and b.
strsplit(x,a)	Splits the string x at pattern a
grep(a,x)	Searches for the pattern a in the string x
sub(a,b,x)	Find the pattern a in the string x and replace it with the string b
paste(..., sep='a')	Concatenates the elements of ..., using the string a as a separator
toupper(x)	Convert the string x to upper case
tolower(x)	Convert the string x to lower case

Functions

AUXILIAR FUNCTIONS

Function	Description
cat(...), c(...)	Concatenate the elements of ... into a vector
length(x)	Length of x
dim(x)	Dimensions of x
seq(i,j,n)	Creates a sequence from i to j, n by n
cut(x,n)	Divides the variable x into a factor with n levels
pretty(x,n)	Divides the variable x into n intervals
t(x)	Transpose the matrix x
tolower(x)	Convert the string x to lower case
apply(x,d,f)	Applies the function f to the object x, along the dimension or dimensions d
tapply(x,i,f)	Applies the function f to the segmented x object from the levels of the factor i
aggregate(x,l,f)	Splits the object x into subsets from a list of elements l, by applying the function f to the a list of elements l

Functions

AUXILIAR FUNCTIONS

Function	Description
factor(x,l,b)	Creates or redefines a factor out of x, with the specified levels l and the labels b
names(x,c), colnames(y,c)	Assigns names to a vector x or to the columns of a matrix or data frame y, and columns of a matrix or data frame y, designated by the character vector c
rbind(x,y)	Joins the x and y data frames by rows
cbind(x,y)	Binds the x and y data frames by columns
sort(x)	Sorts the vector x
order(x)	Vector of indices of the elements of x sorted

Packages (collections of R functions)

INSTALLING:

- install.packages('ggplot2')

LOADING:

- library(ggplot2)

Once the package is installed, can you start using their functions?
NO, it has to be loaded in the session

EXAMPLE:

- ‘ggplot2’ package
- 'Orange' dataset

```
install.packages(ggplot2)  # installing the package
library(ggplot2) # loading the package
head('Orange')  # viewing the first elements of the dataframe
qplot(x=age, y=circumference, data=Orange, geom=c('point', 'line'), colour=Tree) # plotting data
```

Reading and Writting Files

DATA STORED IN TEXT TABLES:

- est <- read.table(‘data/atmosphere.txt', header=T, sep="‘;", dec=".")
- write.table(est[,c(5,3)], file='altitudes.csv', sep=';', row.names=FALSE)

DATA SEQUENTIAL:

- pre <- scan('Prec_1956-2005.dat')
- pre <- array(pre, dim=c(12,50,10))

BINARY FILES:

- zz <- file('prec_1956-2005.bin', 'rb').
- pre_2 <- matrix(readBin(zz,double(),6000,4),ncol=10)
- close(zz)

Reading and Writting Files

The save() and load() functions allow you to save and read objects from R:

- save(est, pre, file='data.Rdata')
- rm(est, pre)
- ls()
- load('data.Rdata')

The source() function is used to read and execute a file with R code:

- source('codigo.R')

EXERCISE:

- Load the package RNetCDF to read a NetCDF file.
- Create an object of class "NetCDF“, named ‘ncfile’, which points to the NetCDF dataset ‘/data/Spain_orog.nc’ (Hint: use the ‘open.nc’ function).
- Read all data from the NetCDF dataset, obtaining an object named ‘data’, a list with the list elements containing an array for each variable or a (possibly nested) list for each group in the NetCDF dataset. (Hint: use the ‘read.nc’ function)
- Exploring our object ‘data’ using the function ‘str’.
- Search the altitude for the point with longitude equal to 0.6 and latitude equal to 40.6.

EXERCISE: SOLUTION

```
> library(RNetCDF)
> ncfile <- open.nc('data/Spain_orog.nc')
> data <- read.nc(ncfile)

> str(data)

> which(abs(data$lon - 0.6) < 10e-5)

> which(abs(data$lat - 40.6) < 10e-5)

> z <- data$orog[100, 47]
```

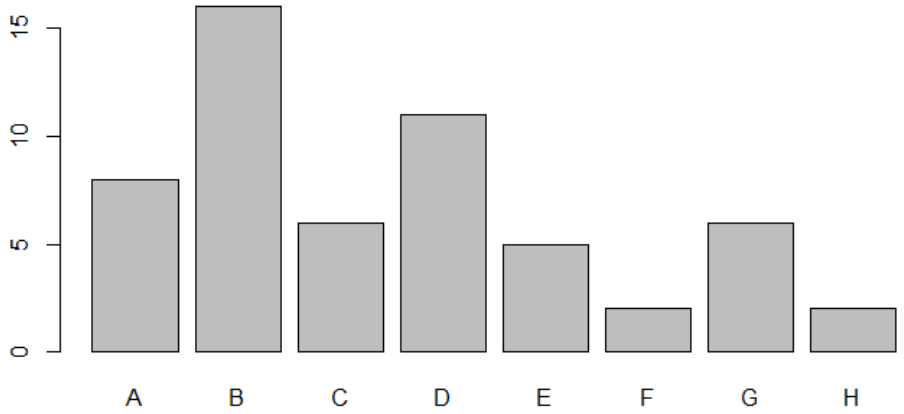
ANNEX

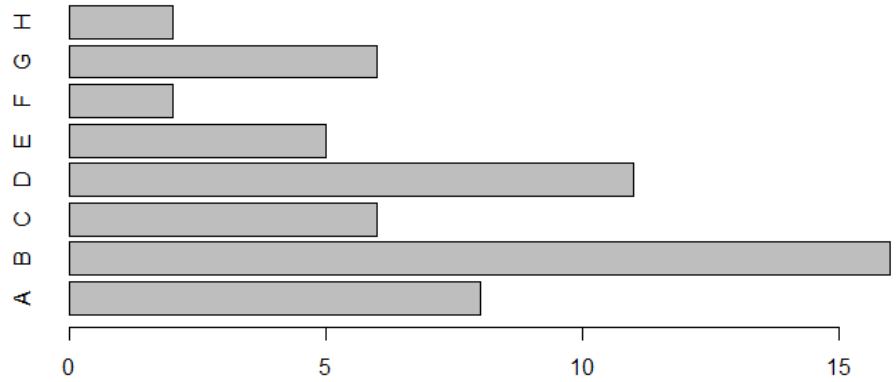
Basic Graphics

BAR CHARTS

Frequencies of a categorical variable by means of vertical or horizontal bars:

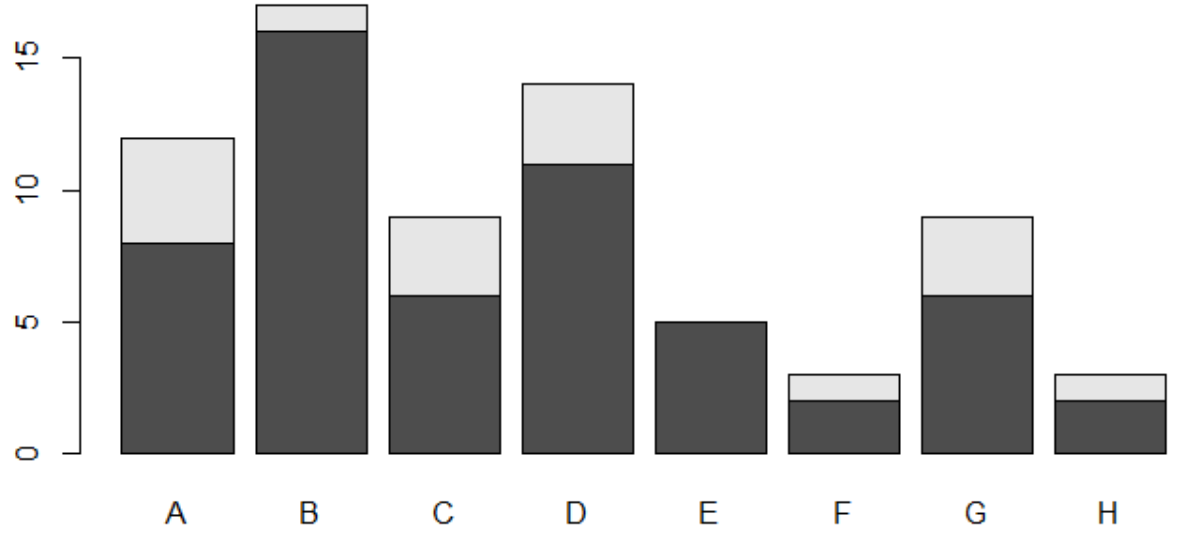
- x <- c(8,16,6,11,5,2,6,2)
- names(x) <- c('A','B','C','D','E','F','G','H')
- barplot(x)
- barplot(x,horiz=TRUE)





Basic Graphics

STACKED BARS

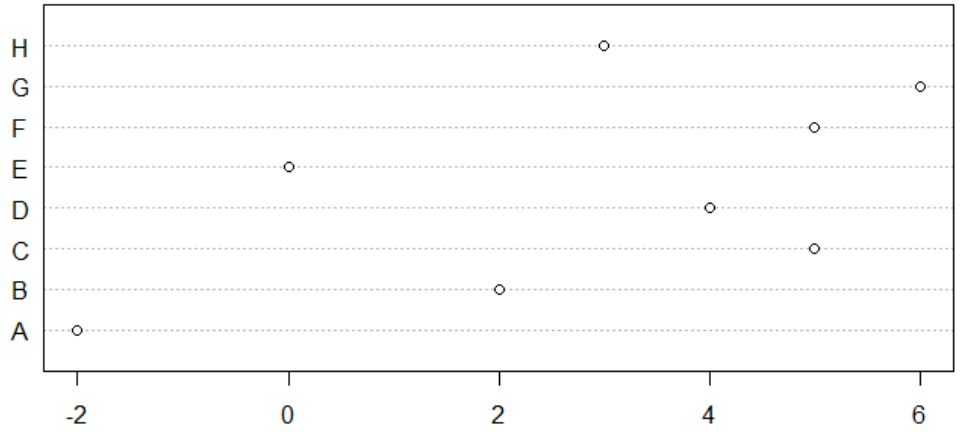


- x <- c(8,16,6,11,5,2,6,2)
- names(x) <- c('A','B','C','D','E','F','G','H')
- y <- c(4,1,3,3,0,1,3,1)
- xy <- rbind(x,y)
- barplot(xy)

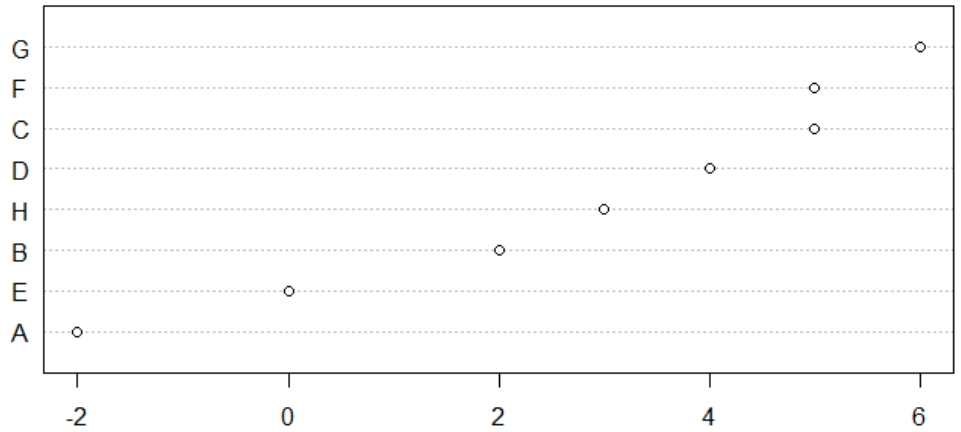
Basic Graphics

DOT PLOTS

- x <- c(-2, 2, 5, 4, 0, 5, 6, 3)
- names(x) <- c('A','B','C','D','E','F','G','H')
- dotchart(x)

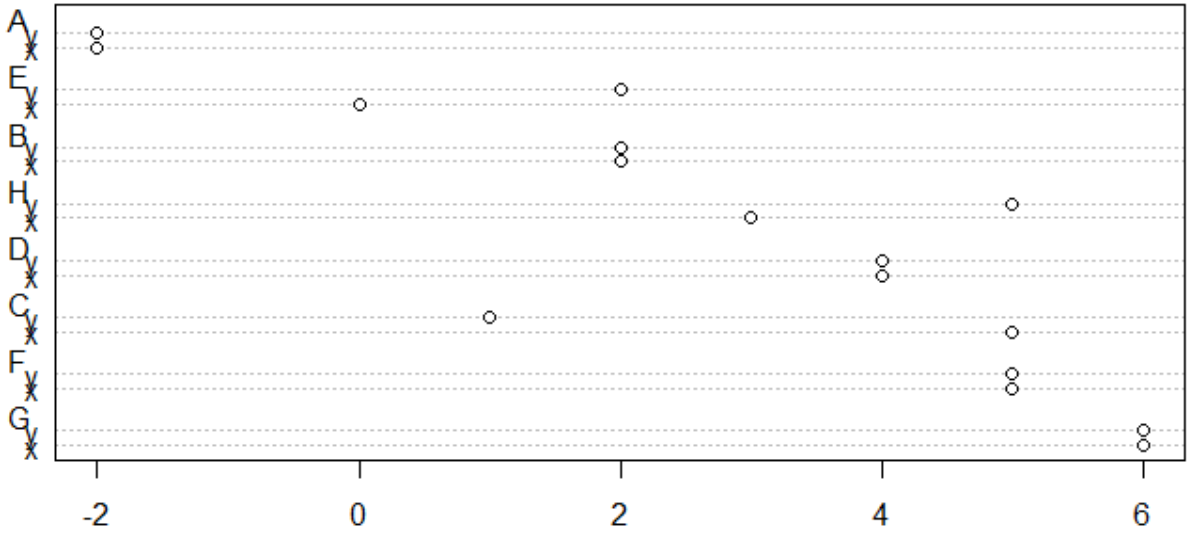


- orden <- order(x)
- dotchart(x[orden]) x <- c(8,16,6,11,5,2,6,2)



Basic Graphics

DOT PLOTS

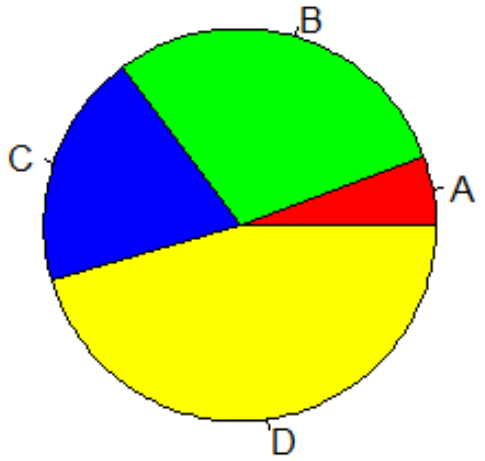


- x <- c(-2, 2, 5, 4, 0, 5, 6, 3)
- names(x) <- c('A','B','C','D','E','F','G','H')
- y <- c(-2, 2, 1, 4, 2, 5, 6, 5)
- xy <- rbind(x,y)
- dotchart(xy[,orden])

Basic Graphics

PIE CHARTS

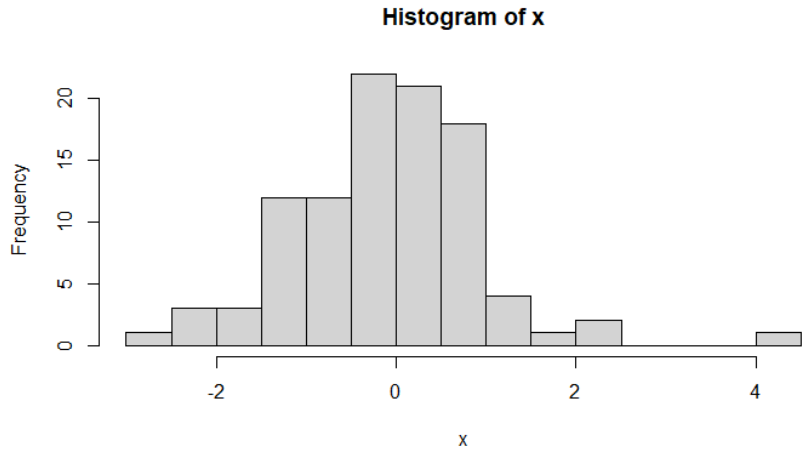
- x <- c(5,26,17,40)
- names(x) <- c('A','B','C','D')
- pie(x)
- orden <- order(x)
- pie(x[orden])
- pie(x, labels=c('A, 'B', 'C', 'D'), col=c('red','green','blue','yellow'))

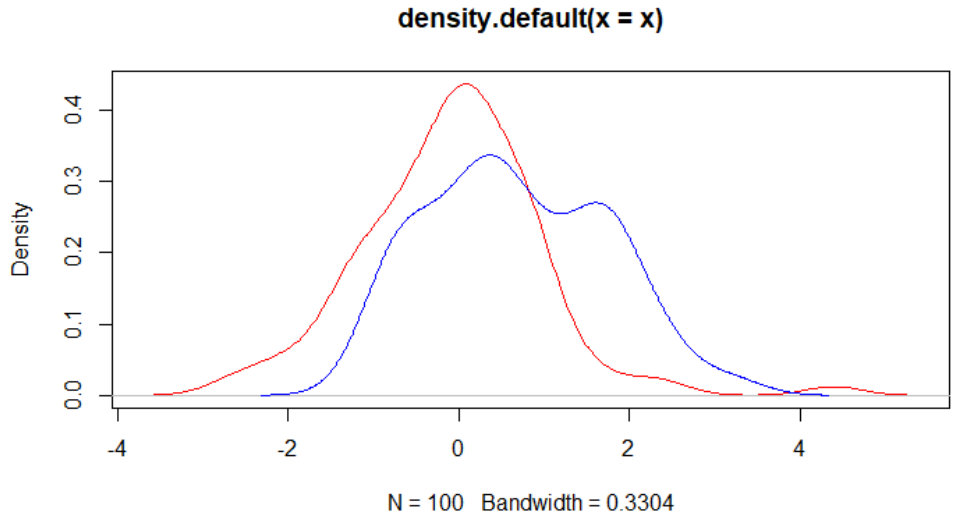


Basic Graphics

HISTOGRAMS

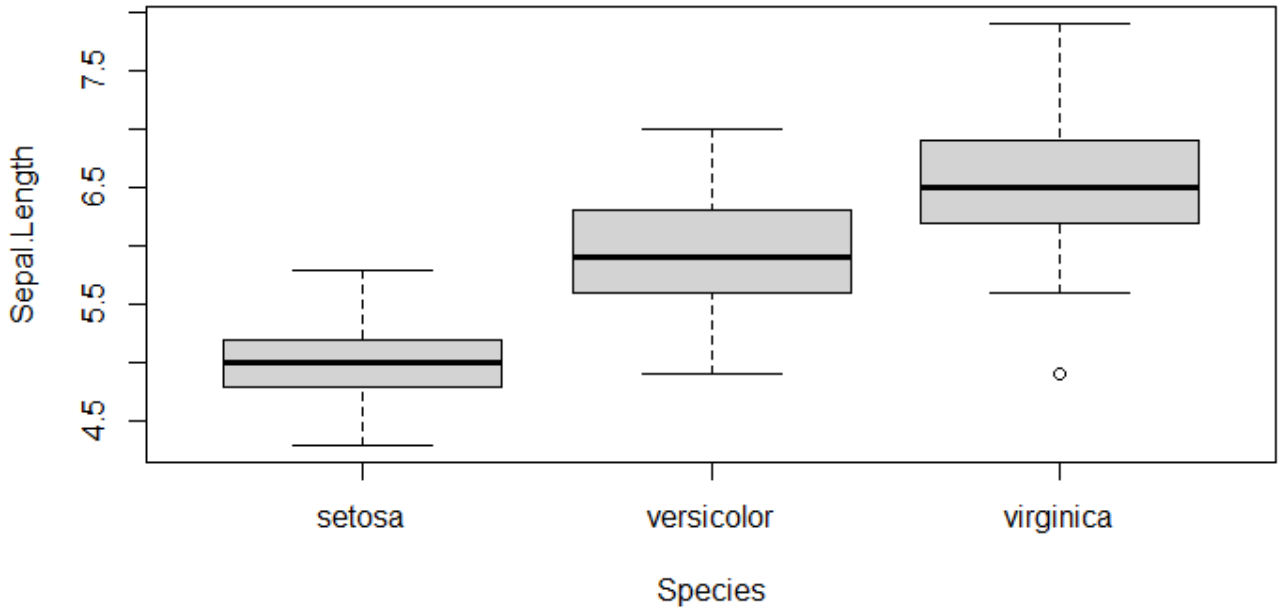
- x <- rnorm(100)
- hist(x)
- hist(x,breaks=20)
- hist(x,freq=FALSE)
- lines(density(x),col='red')
- rug(x)
- plot(density(x),col='red')
- y <- rnorm(100,mean=0.7)
- lines(density(y),col='blue')





Basic Graphics

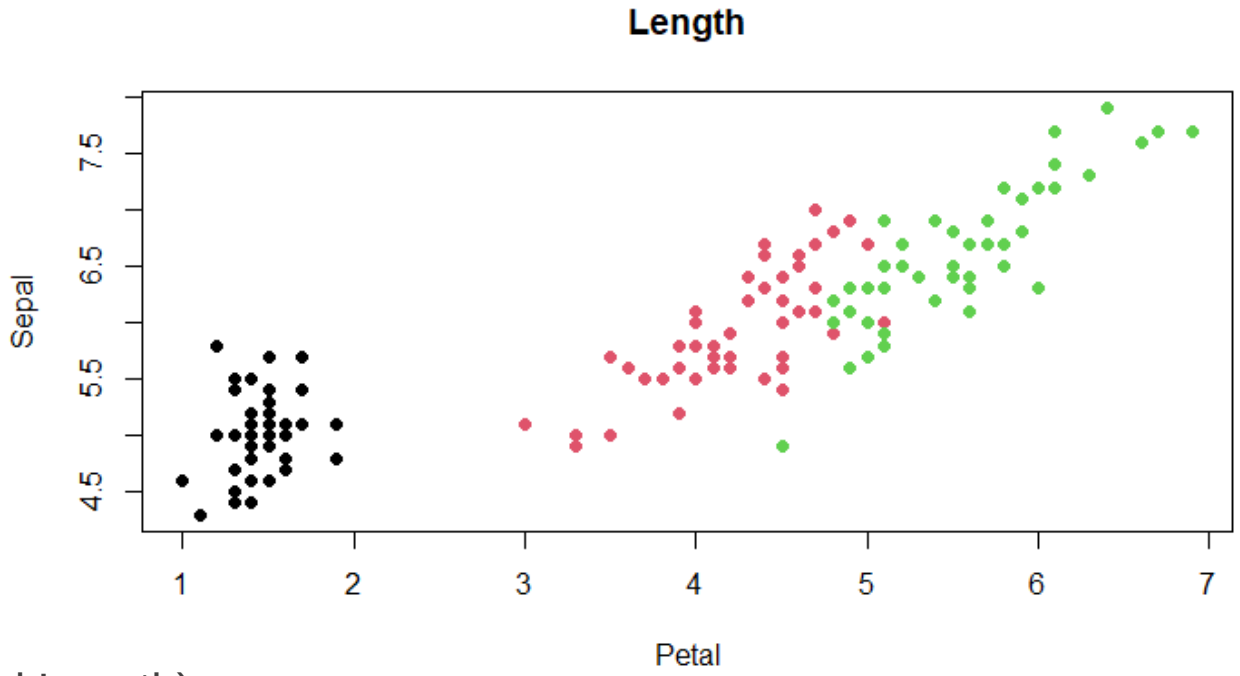
BOX PLOTS



- head(iris)
- boxplot(Sepal.Length~Species, data=iris)

Basic Graphics

SCATTER PLOTS



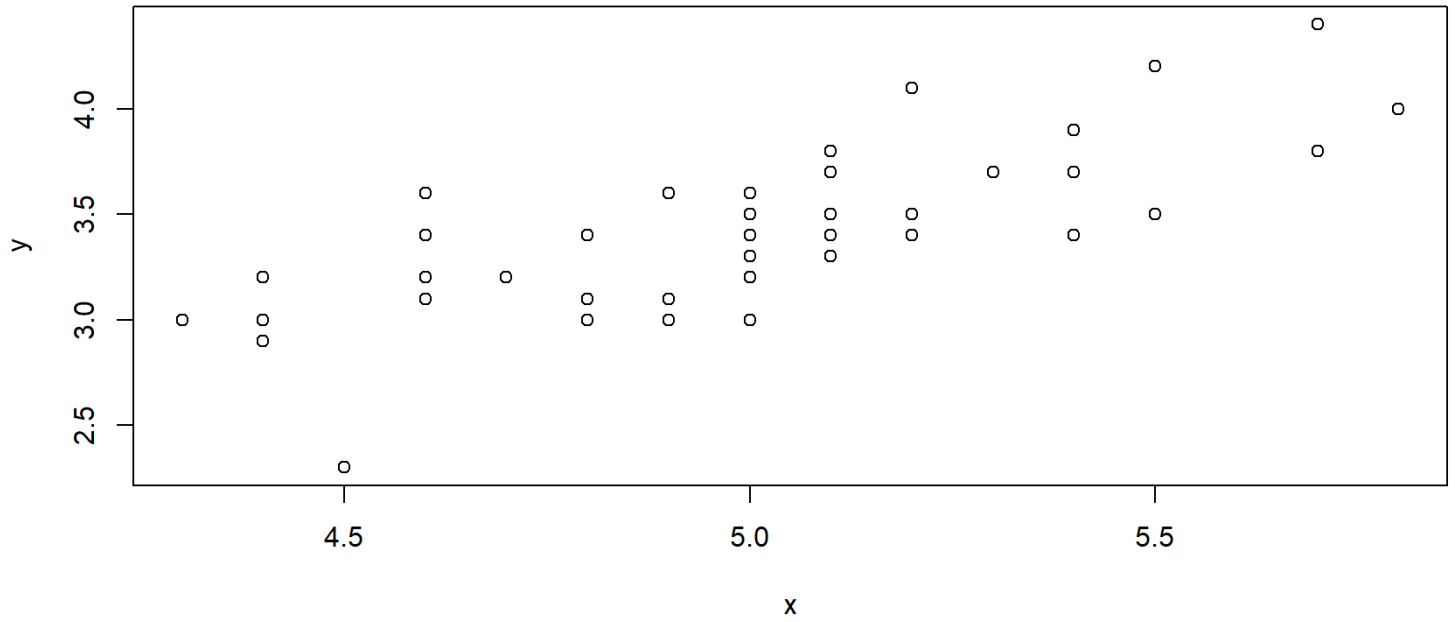
- head(iris)
- plot(x=iris$Sepal.Length, y=iris$Petal.Length)
- plot(iris$Sepal.Length~iris$Petal.Length)
- plot(iris$Sepal.Length~iris$Petal.Length, pch=19, col=iris$Species, main='Length', ylab='Sepal', xlab='Petal')

Regression analysis

CORRELATION

Correlation analysis allows us to know the magnitude, sign and significance of the relationship between two continuous variables.

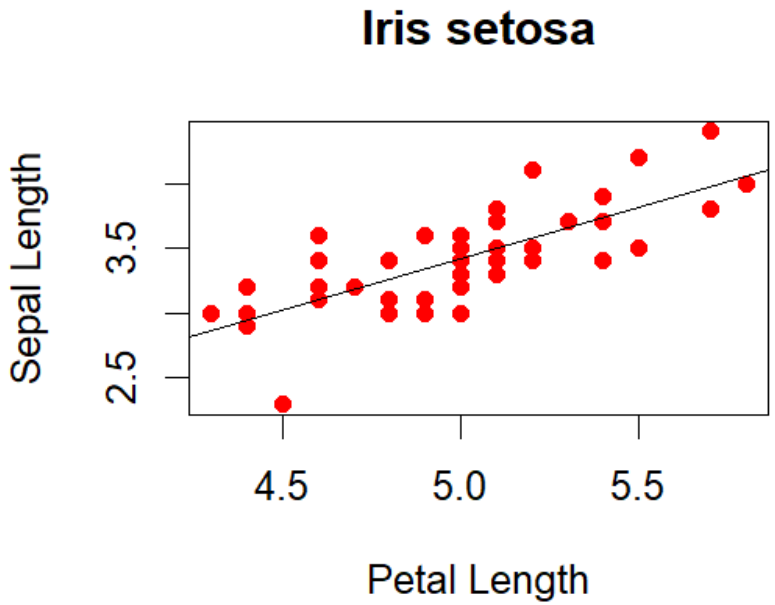
- w <- iris$Species=='setosa'
- x <- iris$Sepal.Length[w]
- y <- iris$Sepal.Width[w]
- plot(x,y)



Regression analysis

CORRELATION

- To perform a correlation analysis cor():
  - **cor(x, y)**
  - cor.test(x, y)
  - cor(x, y, method='pearson')
  - cor.test(x, y, method='pearson')



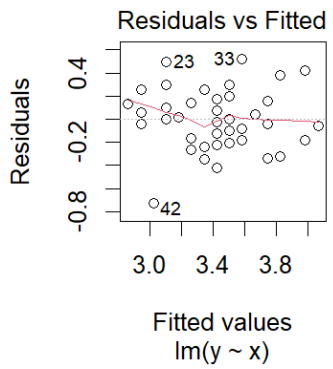
- To perform a basic regression analysis lm():
  -
  - w <- iris$Species=='setosa'
  - x <- iris$Sepal.Length[w]
  - y <- iris$Sepal.Width[w]
  - plot(y~x, pch=19, col='red', main='Iris setosa', xlab='Petal Length', ylab='Sepal Length')
  - **m <- lm(y~x)**
  - abline(m)

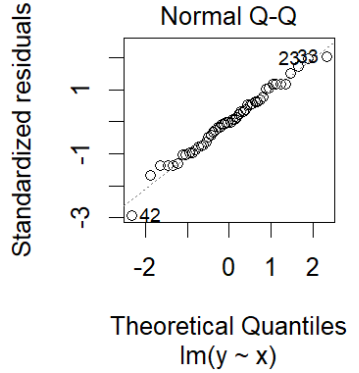
Regression analysis

- summary(m)

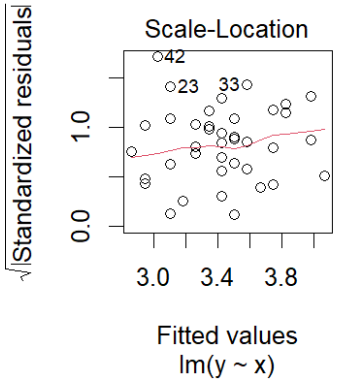
```
Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-0.72394 -0.18273 -0.00306  0.15738  0.51709

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.5694     0.5217  -1.091    0.281
x             0.7985     0.1040   7.681 6.71e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2565 on 48 degrees of freedom
Multiple R-squared:  0.5514,                    Adjusted R-squared:
0.542
F-statistic: 58.99 on 1 and 48 DF,  p-value: 6.71e-10
```

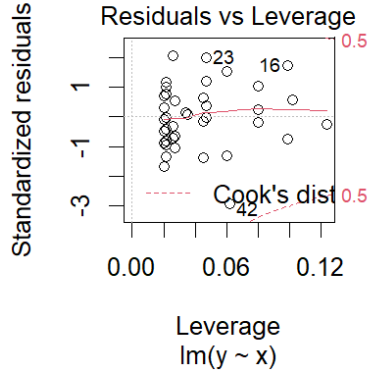
- plot(m)





- attributes(m)
- predict(m,interval='confidence')
- coef(m)
- residuals(m)
- deviance(m)



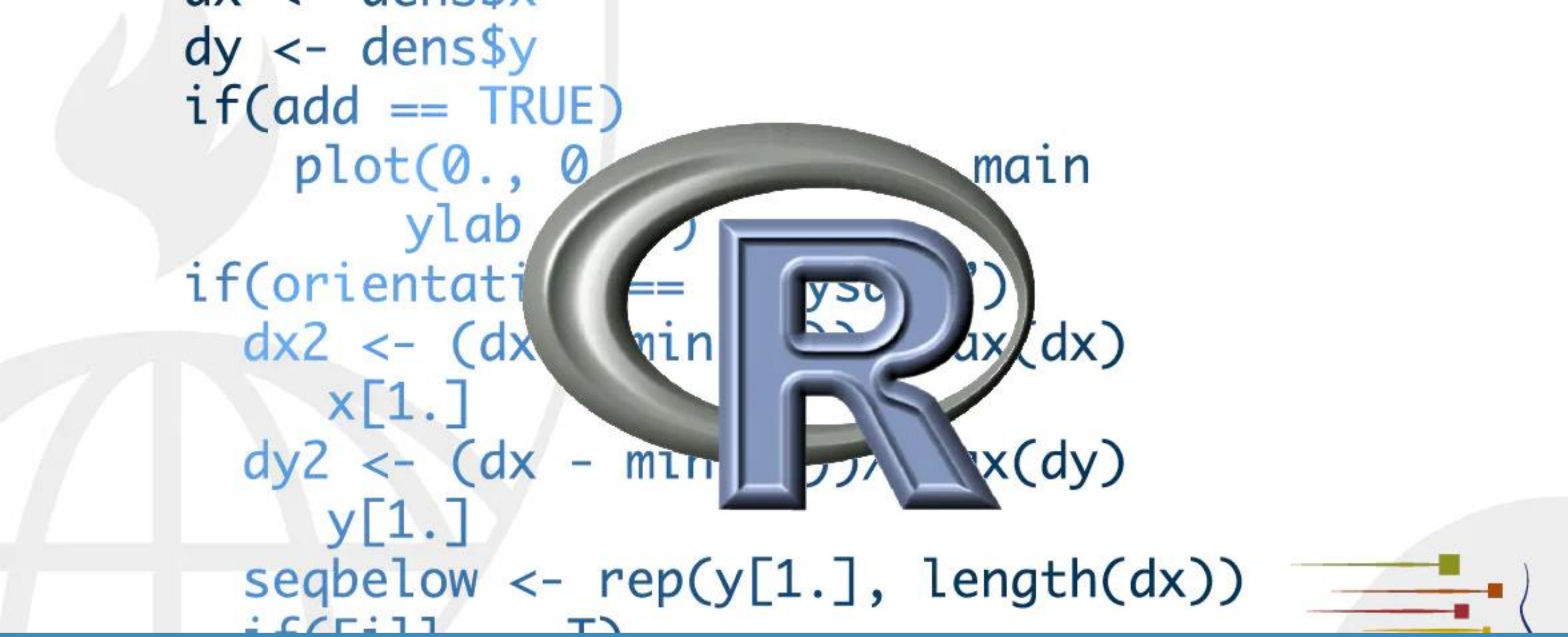


Regression analysis

FUNCTIONS AND LIBRARIES FOR REGRESSION ANALYSIS

http://cran.r-project.org/doc/contrib/Ricci-refcard-regression.pdf

- Simple linear, multivariate, polynomial linear regression by the methodleast squares (OLS): lm()
- Generalised least squares (GLS) regression: gls()
- General linear models (GLM): glm()
- Additive models (GAM)
- Mixed-effects models: lme(), nlme()
- Robust regression
- Principal Component Regression (PCR) and Partial Regression(PLSR)
- ……



THANKS!

AEMet
Agencia Estatal de Meteorología