

Introduction to Bioinformatics

Alexandros C. Dimopoulos

alexdem@di.uoa.gr

Master of Science
“Data Science and Information Technologies”
Department of Informatics and Telecommunications
National and Kapodistrian University of Athens

2025-26



R

R

R is ‘GNU S’, a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc.

<https://cran.r-project.org/doc/FAQ/R-FAQ.html>

- statistical computation and graphics
- influenced by two existing languages: S (similar appearance) and Scheme (underlying implementation and semantics)
- interpreted
- distributed under a GNU-style copyleft
- Unix-like, Windows and Mac families OS
 - 386, amd64/x86_64, alpha, arm, arm64, hppa, mips/mipsel, powerpc, s390x and sparc CPUs, i386-hurd-gnu, cpu-kfreebsd-gnu for i386 and amd64, i386-pc-solaris, rs6000-ibm-aix, sparc-sun-solaris, x86_64-apple-darwin, x86_64-unknown-freebsd and x86_64-unknown-openbsd

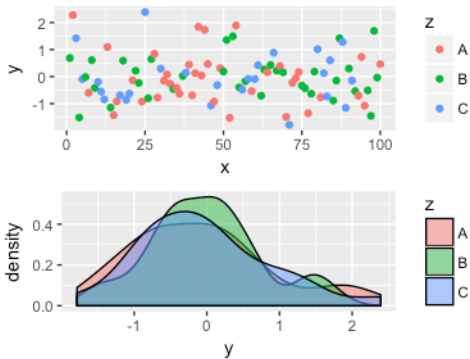


Why R I

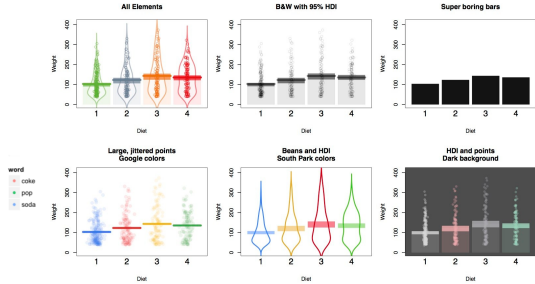
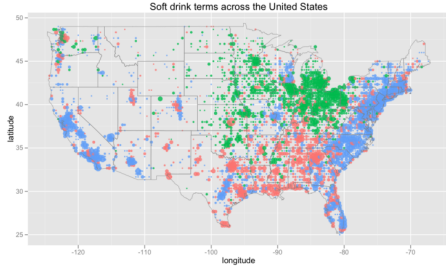
- Free!
- Large user community that contributes packages
- Extremely flexible in abilities
- Graphics capabilities are remarkable
- Fast and efficient
- Interfaces with Microsoft Office Excel
- Can program almost anything AND save and repeat



Why R II



Why R III



Why R IV

- Requires patience
- Somewhat steep learning curve for R
- Somehow different than other typical programming languages



Operators

Operators	Type of operator
<code>+</code> <code>-</code> <code>*</code> <code>/</code> <code>%%</code> <code>%/%</code> <code>^</code>	arithmetic
<code>></code> <code>>=</code> <code><</code> <code><=</code> <code>==</code> <code>!=</code>	Relational
<code>!</code> <code>&</code> <code> </code>	logical
<code><-</code> <code>-></code> <code>=</code>	assignment
<code>\$</code>	reference to list object
<code>:</code>	sequence creation



Basic functions

Function	Explanation
<code>log(x)</code>	log to base e of x
<code>exp(x)</code>	antilog of x (e^x)
<code>log(x,n)</code>	log to base n of x
<code>log10(x)</code>	log to base 10 of x
<code>sqrt(x)</code>	$\sqrt{x}$
<code>factorial(x)</code>	$x!$
<code>floor(x)</code>	$\lfloor x \rfloor$
<code>ceiling(x)</code>	$\lceil x \rceil$
<code>round(x, digits=0)</code>	round the value of x to an integer
<code>signif(x, digits=6)</code>	give x to 6 digits in scientific notation
<code>abs(x)</code>	$ x $
<code>cos(x)</code>	cosine of x in radians
<code>sin(x)</code>	sin of x in radians
<code>tan(x)</code>	tan of x in radians

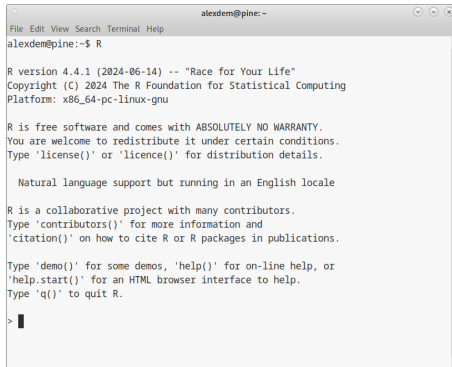


Basic array functions

Function	Explanation
<code>max(x)</code>	maximum value in x
<code>min(x)</code>	minimum value in x
<code>range(x)</code>	vector of <code>min(x)</code> and <code>max(x)</code>
<code>sum(x)</code>	total of all the values in x
<code>mean(x)</code>	arithmetic average of the values in x
<code>median(x)</code>	median value in x
<code>var(x)</code>	sample variance of x
<code>cor(x,y)</code>	correlation between vectors x and y
<code>sort(x)</code>	a sorted version of x
<code>order(x)</code>	an integer vector containing the permutation to sort x into ascending order
<code>quantile(x)</code>	vector containing the minimum, lower quartile, median, upper quartile, and maximum of x
<code>colMeans(x)/rowMeans(x)</code>	column/row means of dataframe or matrix x
<code>colSums(x)/rowSums(x)</code>	column/row totals of dataframe or matrix x



Initial screen I



```
alexdem@pine: ~  
File Edit View Search Terminal Help  
alexdem@pine:~$ R  
  
R version 4.4.1 (2024-06-14) -- "Race for Your Life"  
Copyright (C) 2024 The R Foundation for Statistical Computing  
Platform: x86_64-pc-linux-gnu  
  
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.  
  
Natural language support but running in an English 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.  
  
> █
```

- All commands are given after the “>” symbol



Initial screen II

```
> 4+4
[1] 8
> 3*4
[1] 12
> 5/2
[1] 2.5
> 5%%2 #remainder
[1] 1
> 5%/%2 #quotient
[1] 2
```



Initial screen III

```
> log(10)
[1] 2.302585
> log(10,10)
[1] 1
```



Inf & NaN

Inf (Infinity)

```
> 100/0
```

```
[1] Inf
```

```
> -100/0
```

```
[1] -Inf
```

Not a Number (NaN)

```
> 0/0
```

```
[1] NaN
```

```
> Inf - Inf
```

```
[1] NaN
```



Logic values

```
> 10>1  
[1] TRUE  
> 10<1  
[1] FALSE  
> 100 == 100  
[1] TRUE
```



Vectors I

- Every user input is considered (by default) a vector
- `[1]` refers to the index of the first object of the (first) row
- One-based numbering is used for the indexes of a vector

```
> 1
[1] 1
> 1:5
[1] 1 2 3 4 5
> 1:25
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13
[14] 14 15 16 17 18 19 20 21 22 23 24 25
```



Vectors II

- The `c(...)` function (combine) allows the creation of larger vectors

```
> c(1,3,5,7,9)
```

```
[1] 1 3 5 7 9
```

```
> c(1,3,5,7,9)+c(2,4,6,8,10)
```

```
[1] 3 7 11 15 19
```

```
> c(1, 2, 3, 4) + 1
```

```
[1] 2 3 4 5
```



Vectors III

```
> c(1,3,5,7,9)+c(2,4)
```

```
[1] 3 7 7 11 11
```

Warning message:

```
In c(1, 3, 5, 7, 9) + c(2, 4) :
```

```
longer object length is not a multiple of shorter object  
length
```



Vectors IV

```
> "Hello world."  
[1] "Hello world."  
> c("Hello world", "Hello again")  
[1] "Hello world" "Hello again"
```



Comments

- Whatever follows the # symbol is considered a comment and is ignored

```
> 1 +2 +3
```

```
[1] 6
```

```
> 1 +2 #+3
```

```
[1] 3
```



Variables I

- As an interpreted language, the variables do not have to be declared prior to usage
- Case-sensitive, i.e. x is considered different to X
- Variable names cannot
 - start with digits (e.g. 1variable) or symbols (e.g. %variable)
 - contain spaces, e.g. variable.name and not variable name

```
> x <- 1
> x
[1] 1
> 1 -> x
> x
[1] 1
```



Variables II

```
> x = 1
```

```
> x
```

```
[1] 1
```



Variables III

```
> x = 1
> x
[1] 1
> y <- "a"
> y
[1] "a"
> z=c(x,y)
> z
[1] "1" "a"
```



Variables IV

```
> x=11:20
> x
 [1] 11 12 13 14 15 16 17 18 19 20
> x[4]
 [1] 14
> x[1:4]
 [1] 11 12 13 14
> x[c(4,10)]
 [1] 14 20
> x[-c(1:4)]
 [1] 15 16 17 18 19 20
```



Variables V

```
> x<15
[1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
     FALSE
> x[x<15]
[1] 11 12 13 14
```



Functions

```
> f <- function(x,y) {x + y}
> f(1,2)
[1] 3
> g <- function(x,y) {c(x + y, x * y)}
> g(1,2)
[1] 3 2
```



Factors I

```
> x=c("a","b","a","a","b")  
> x  
[1] "a" "b" "a" "a" "b"  
> x=factor(x)  
> x  
[1] a b a a b  
Levels: a b
```



Factors II

```
> attributes(x)
$levels
[1] "a" "b"
$class
[1] "factor"
```



Factors III

```
> x
[1] a b a a b
Levels: a b
> levels(x)
[1] "a" "b"
> levels(x)=c("0","1")
> x
[1] 0 1 0 0 1
Levels: 0 1
```



Arrays I

```
> a <- array(11:16,dim=c(3,2))
```

```
> a
```

```
      [,1] [,2]  
[1,]    11    14  
[2,]    12    15  
[3,]    13    16
```

```
> a[1,2]
```

```
[1] 14
```



Arrays II

```
> a[1:2,1:2]
      [,1] [,2]
[1,]    11    14
[2,]    12    15
```



Arrays III

```
> a>13
      [,1] [,2]
[1,] FALSE TRUE
[2,] FALSE TRUE
[3,] FALSE TRUE
> a[1]
[1] 11
> a[1,]
[1] 11 14
> a[,1]
[1] 11 12 13
```



Arrays IV

```
> which(a>13)
[1] 4 5 6
> which(a>13,arr.ind=T)
      row col
[1,]    1   2
[2,]    2   2
[3,]    3   2
```



Arrays V

```
> b=array(1:12,dim=c(2,2,3))
```

```
> b
```

```
, , 1
```

```
    [,1] [,2]
```

```
[1,]    1    3
```

```
[2,]    2    4
```

```
, , 2
```

```
    [,1] [,2]
```

```
[1,]    5    7
```

```
[2,]    6    8
```



Arrays VI

```
, , 3  
      [,1] [,2]  
[1,]      9    11  
[2,]     10    12
```



Arrays VII

In general in R :

- vector $\rightarrow$ one dimensional array
- matrix $\rightarrow$ two dimensional array
- array $\rightarrow$ array of any dimensional



Lists I

- Lists can contain objects of different types, e.g. numbers and strings

```
> mylist=list(name="alex",id=1234)
```

```
> mylist
```

```
$name
```

```
[1] "alex"
```

```
$id
```

```
[1] 1234
```

```
> mylist$name
```

```
[1] "alex"
```



Lists II I

```
> mylist2=list(list(name="alex",id=1234),  
  list(name="alex2",id=1234))  
> mylist2  
[[1]]  
[[1]]$name  
[1] "alex"  
[[1]]$id  
[1] 1234
```



Lists II II

```
[[2]]  
[[2]]$name  
[1] "alex2"  
[[2]]$id  
[1] 1234
```



Data frame I

- A list that contains multiple vectors of the same size
- It resembles a spreadsheet

```
> names=c("alex","john","tom")
> ids=c(1,2,3)
> ZipCode=c(5544,2343,1234)
> data=data.frame(names,ids,ZipCode)
> data
```

	names	ids	ZipCode
1	alex	1	5544
2	john	2	2343
3	tom	3	1234



Data frame II

```
> data$ids
[1] 1 2 3
> data$ZipCode[data$names=="alex"]
[1] 5544
> data$names
[1] alex john tom
Levels: alex john tom
> data[data$names=="alex",]
  names ids ZipCode
1  alex   1    5544
```



Data frame III

```
> colnames(data)
[1] "names"      "ids"        "ZipCode"
> data[,2:3]
  ids ZipCode
1   1    5544
2   2    2343
3   3    1234
> colSums(data[,2:3])
  ids ZipCode
    6    9121
> rowSums(data[,2:3])
[1] 5545 2345 1237
```



Classes

```
> class(data)
[1] "data.frame"
> class(names)
[1] "character"
> class(ids)
[1] "numeric"
> class(ZipCode)
[1] "numeric"
> class(g)
[1] "function"
```

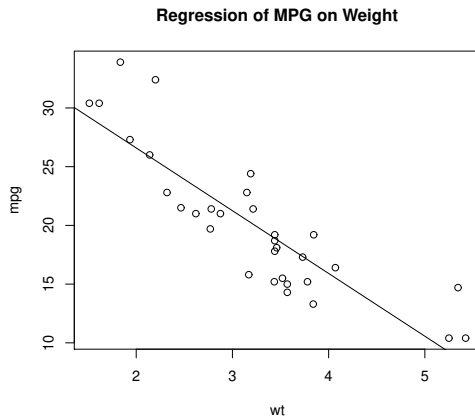


Creating Plots I

```
# Import data extracted from the 1974 Motor  
# Trend US magazine  
# mpg    -->    Miles/(US) gallon  
# wt     -->    Weight (1000 lbs)  
# gear   -->    Number of forward gears  
# examples from http://www.statmethods.net/index.html  
> attach(mtcars)  
> plot(wt, mpg)  
> abline(lm(mpg~wt))  
> title("Regression of MPG on Weight")
```

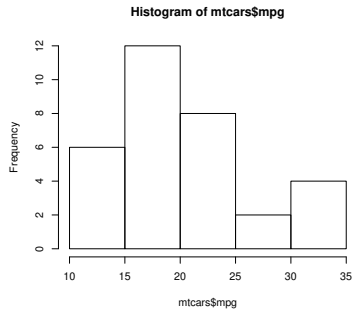


Creating Plots II



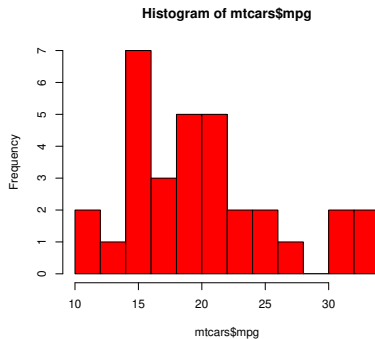
Histograms I

```
> hist(mtcars$mpg)
```



Histograms II

```
> hist(mtcars$mpg, breaks=12, col="red")
```

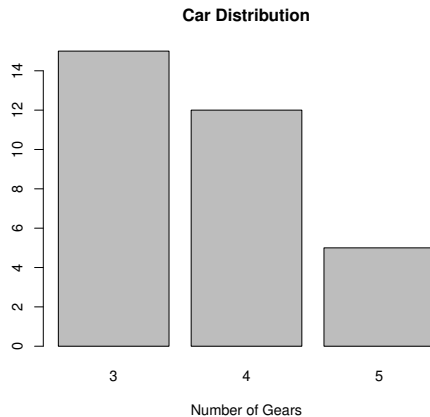


Barplots I

```
> counts <- table(mtcars$gear)
> counts
 3  4  5
15 12  5
> barplot(counts, main="Car Distribution",
          xlab="Number of Gears")
```



Barplots II

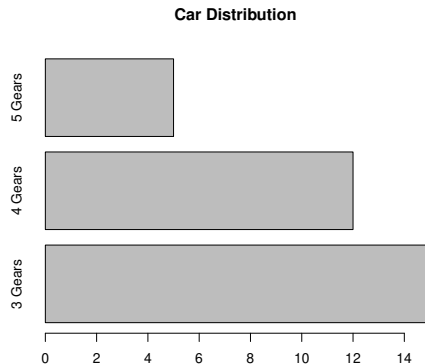


Barplots III

```
> counts <- table(mtcars$gear)
> barplot(counts, main="Car Distribution", horiz=TRUE,
  names.arg=c("3 Gears", "4 Gears", "5 Gears"))
```

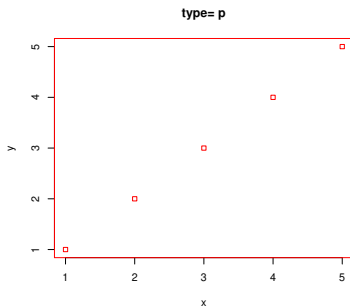


Barplots IV



Line Chart

```
> x <- c(1:5); y <- x # create some data  
> plot(x, y, type="p", main=heading)
```



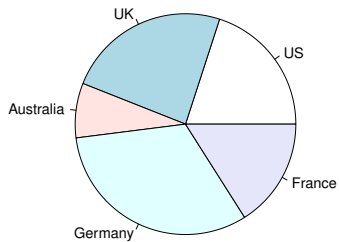
Pie Chart I

```
> slices <- c(10, 12, 4, 16, 8)
> lbls <- c("US", "UK", "Australia", "Germany", "France")
> pie(slices, labels = lbls, main="Pie Chart of Countries")
```



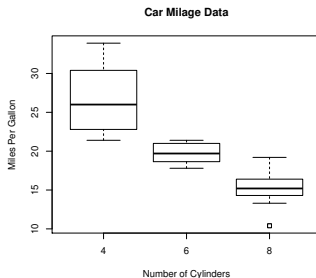
Pie Chart II

Pie Chart of Countries



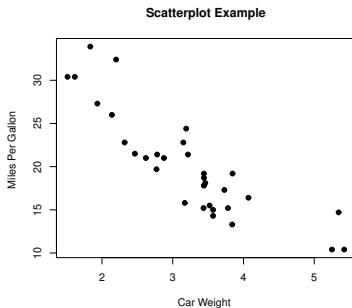
Boxplot

```
# Boxplot of MPG by Car Cylinders  
> boxplot(mpg~cyl,data=mtcars, main="Car Milage Data",  
          xlab="Number of Cylinders", ylab="Miles Per Gallon")
```



Scatterplot

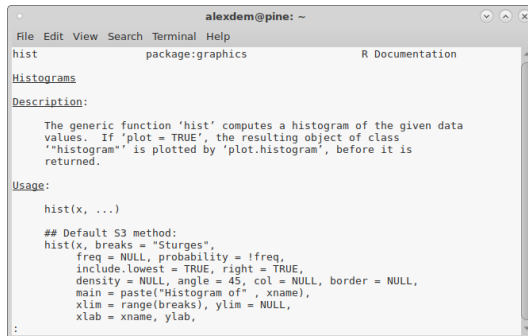
```
> plot(wt, mpg, main="Scatterplot Example", xlab="Car  
Weight ", ylab="Miles Per Gallon ", pch=19)
```



Help I

- R has a help system for built-in functions and installed packages

> ?hist



```
alexdem@pine: ~  
File Edit View Search Terminal Help  
hist                package:graphics                R Documentation  
  
Histograms  
  
Description:  
  
The generic function 'hist' computes a histogram of the given data  
values. If 'plot = TRUE', the resulting object of class  
'"histogram"' is plotted by 'plot.histogram', before it is  
returned.  
  
Usage:  
  
hist(x, ...)  
  
## Default S3 method:  
hist(x, breaks = "Sturges",  
      freq = NULL, probability = !freq,  
      include.lowest = TRUE, right = TRUE,  
      density = NULL, angle = 45, col = NULL, border = NULL,  
      main = paste("Histogram of" , xname),  
      xlim = range(breaks), ylim = NULL,  
      xlab = xname, ylab,  
      :  
      :
```

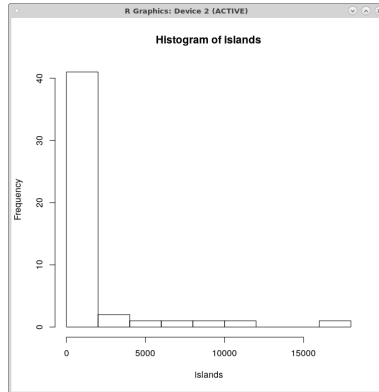


Help II

```
> example(hist)
> op <- par(mfrow = c(2, 2))
> hist(islands)
```



Help III



Help IV

```
> ??hist
```

```
alexdem@pine: ~  
File Edit View Search Terminal Help  
Help files with alias or concept or title matching 'hist' using regular  
expression matching:  
  
base::environment variables  
      Environment Variables  
Aliases: R_HISTFILE, R_HISTSIZE  
car::Blackmore Exercise Histories of Eating-Disordered and  
Control Subjects  
car::hist.boot Generic functions to provide support for 'boot'  
objects  
Aliases: hist.boot  
caret::densityplot.rfe Lattice functions for plotting resampling  
results of recursive feature selection  
Aliases: histogram.rfe  
caret::histogram.train Lattice functions for plotting resampling  
results  
Aliases: histogram.train  
caret::resampleHist Plot the resampling distribution of the model  
statistics  
Aliases: resampleHist  
:
```



Help V

vignette

A Vignette is a free-form document describing a package usage with examples

```
> vignette("affy")
```



Hands on

- Create a vector (A) of 100 elements, containing the values from 1 to 100
- Create a vector (B) of 100 elements, containing the values from 100 to 1
- Create a data frame (DF) with 2 columns, containing the vectors A and B
- Add a new column to DF, containing the sum of the elements of A and B at each row
- Plot $\sin(x)$ for a range of x from -10 to 10 with various steps, e.g. 1, 0.5, 0.01



Exercise 3 - Familiarizing with R

- Filter data
- Create plots
- ...

Submit via e-class assignment

<https://eclass.uoa.gr/modules/work/index.php?course=DI425&id=62670>

OR by email at alexdem@di.uoa.gr

[https://eclass.uoa.gr/modules/document/file.php/DI425/2025-26/
exercises/ITBI2025-exercise4-ACD04112025.zip](https://eclass.uoa.gr/modules/document/file.php/DI425/2025-26/exercises/ITBI2025-exercise4-ACD04112025.zip)

DEADLINE 18/11/25



Questions?

