Outline

day 2 – July 13th

- Two-dimensional data structures:
  - Matrix
  - Data Frames
- Library / packages
- More on Input and Output

Data Structures

Vectors

Matrices

Data frames

Matrices

- A matrix is a vector of **2 dimensions**

- All columns in a matrix must have :
  - the same **type** (numeric, character, logical)
  - the same **length**

```
b <- matrix(c(1, 0, 34, 44, 12, 4),
            nrow=3,
            ncol=2)
```

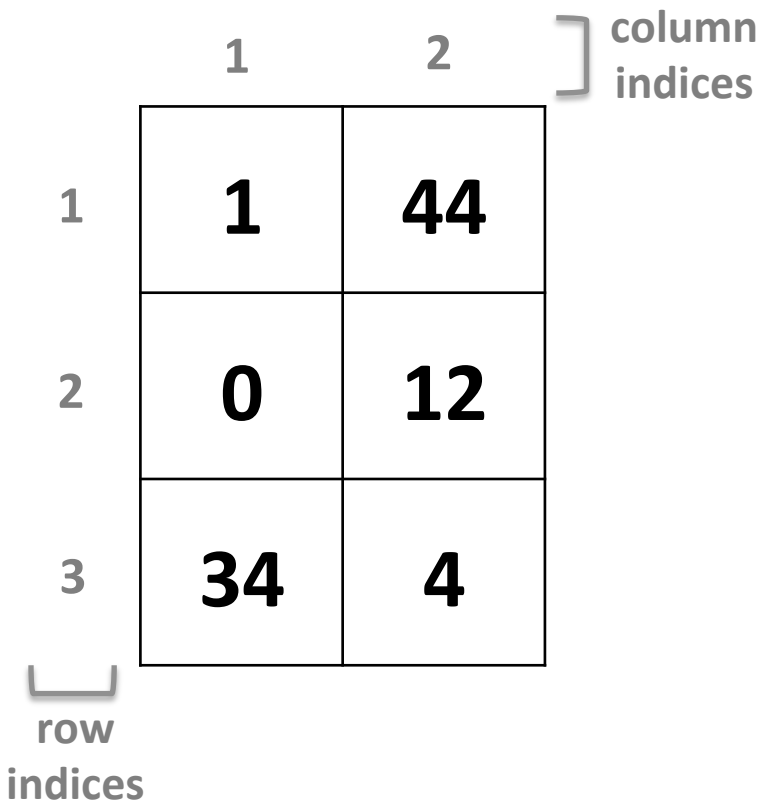
Matrices

b

1	**44**
0	**12**
34	**4**

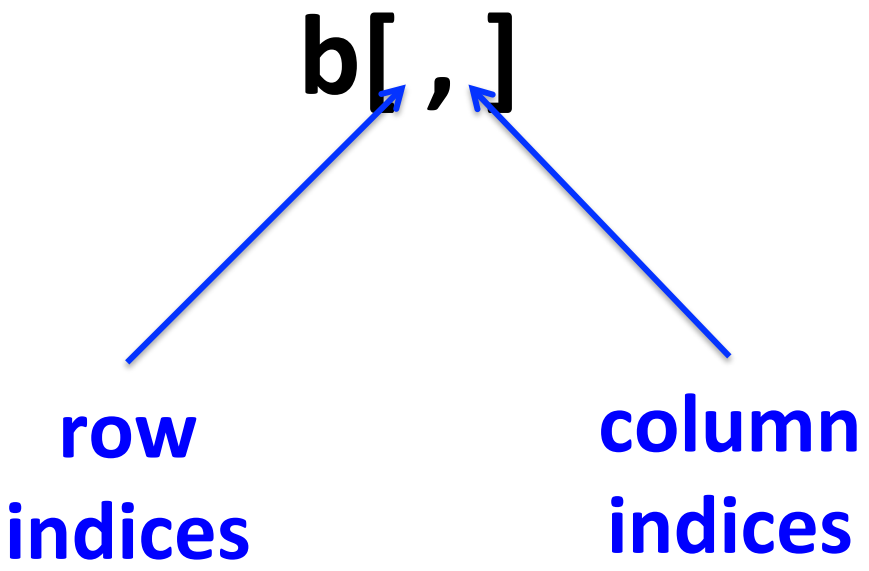
Matrices

b



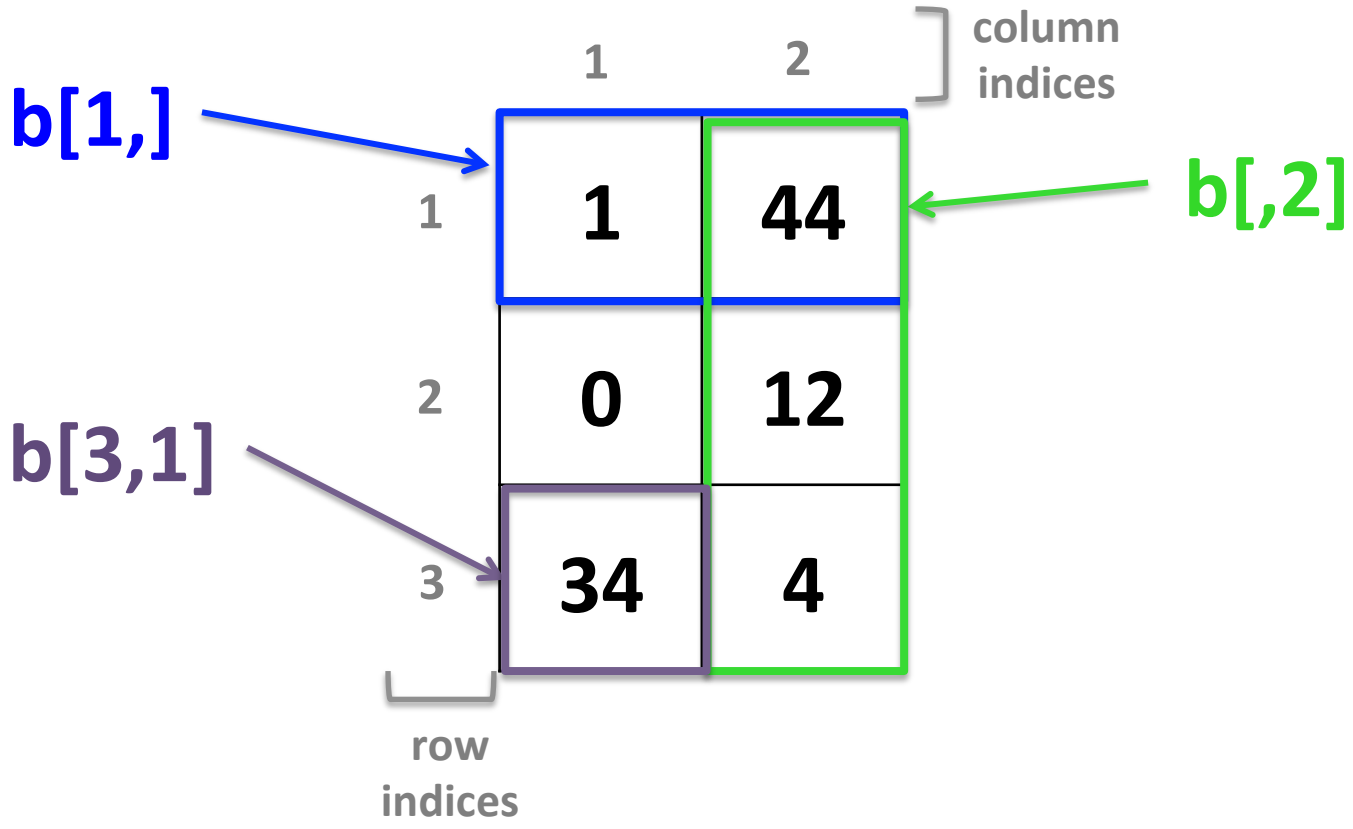
Matrices

- Fetch rows, columns or single elements of a matrix:



Matrices

b



Manipulate matrices

- **add 1** to all elements of a matrix

```
b <- b + 1
```

- **multiply by 3** all elements of a matrix

```
b <- b * 3
```

- **subtract 2** to each element of the first row of a matrix

```
b[1, ] <- b[1, ] – 2
```

- **replace** elements that comply a condition:

```
b[ b>3 ] <- 4
```

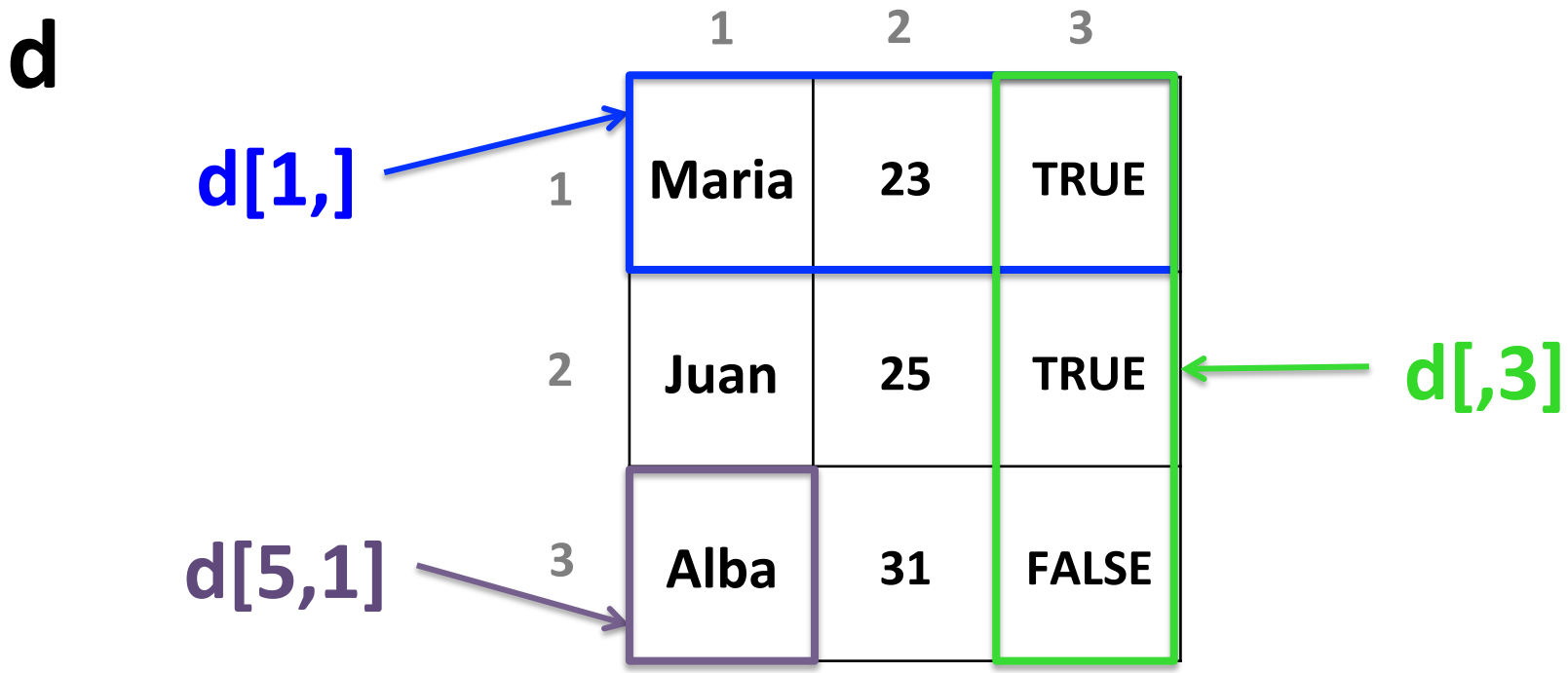
Data frames

- Structures of 2 dimensions.
- More general than matrices.
- Different columns can have **different types** but must have the **same length.**

```
d <- data.frame(c("Maria", "Juan", "Alba"),
                c(23, 25, 31),
                c(TRUE, TRUE, FALSE))
```

Data frames

- Fetch rows, columns or single elements of a data frame:



Dimensions

- Get dimension of data frame or matrix

- **dim**(b)

  first element: number of rows

  second element: number of columns
- **nrow**(b)
- **ncol**(b)

Dimension names

- Column and / or row names can be added to matrices and data frames:

```
colnames(d) <- c("Name", "Age", "Vegetarian")
rownames(d) <- c("Patient1", "Patient2", "Patient3")
```

Dimension names

- Instead of indices, you can use dimension names to subset a matrix or a data frame:

d

		Name	Age	Vegetarian
		1	2	3
Patient1	1	Maria	23	TRUE
Patient2	2	Juan	25	TRUE
Patient3	3	Alba	31	FALSE

column names

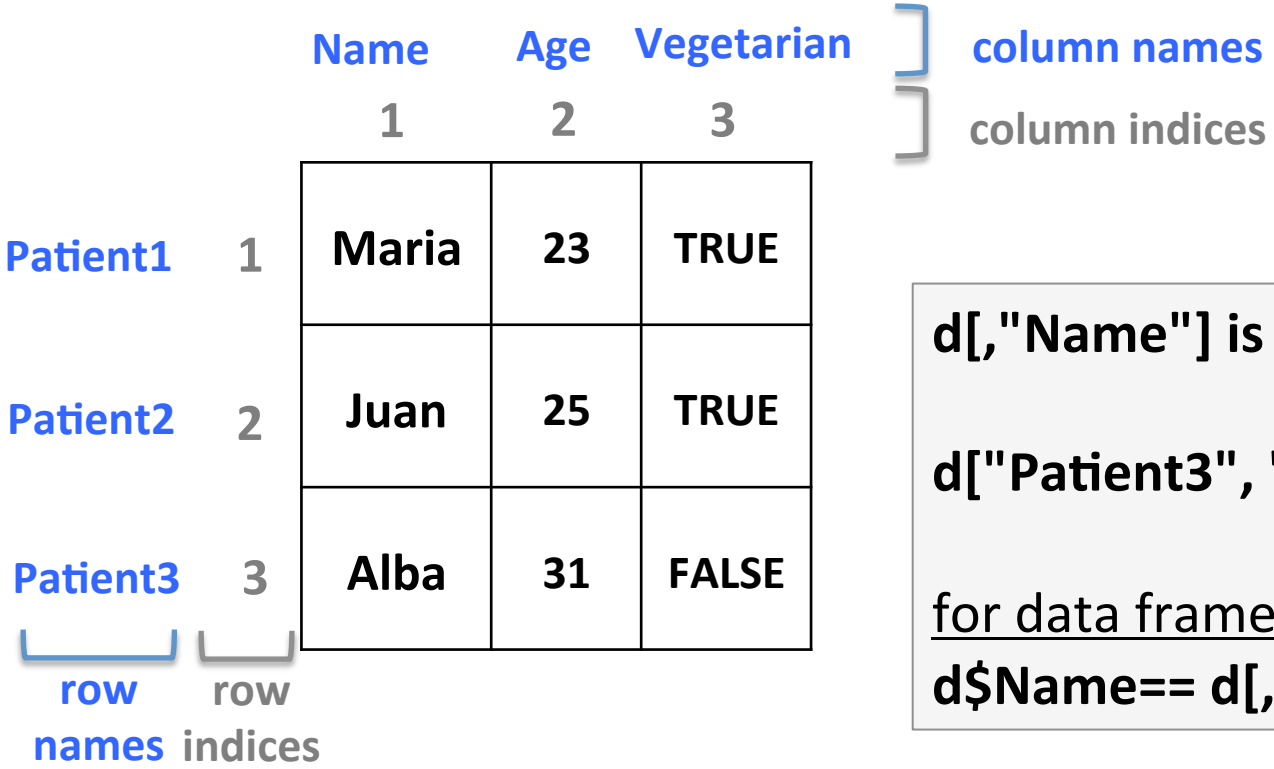
column indices

row names

row indices

Dimension names

- Instead of indices, you can use dimension names to subset a matrix or a data frame:



d[,"Name"] is d[,1]

d["Patient3", "Age"] == d[3,2]

for data frames:

d$Name== d[,1] == d[, "Name"]

Create matrices and data frames with names

- **Matrix**

```
m <- matrix(1:4, ncol=2,
    dimnames=list(c("row1", "row2"), c("col1", "col2")))
```

- **Data frame**

```
df <- data.frame(col1=1:2, col2=1:2,
    row.names=c("row1", "row2"))
```

Manipulate 2-dimensional objects

- Same principle as for vectors… but 2-dimensional.

- Select columns of b if elements in **row 3 are inferior or equal to 4:**

b

1	**44**
0	**12**
34	**4**

b[, b[3 ,] <= 4]

44
12
4

Manipulate 2-dimensional objects

- Select rows of b if elements in **column 2 are superior to 24:**

		Name	Age	Vegetarian
		1	2	3
Patient1	1	Maria	23	TRUE
Patient2	2	Juan	25	TRUE
Patient3	3	Alba	31	FALSE

d[d[,2] > 24,]

or

d[d[,"Age"] > 24,]

Juan	25	TRUE
Alba	31	FALSE

Manipulate 2-dimensional objects

- Select rows of d if elements in **column 3 are TRUE:**

		Name	Age	Vegetarian
		1	2	3
Patient1	1	Maria	23	TRUE
Patient2	2	Juan	25	TRUE
Patient3	3	Alba	31	FALSE

d[d[, 3] == TRUE,]

Maria	23	TRUE
Juan	25	TRUE

Some useful commands for matrices and data frames

- **cbind**: adds a column
  cbind(d, 1:3)

- **rbind**: adds a row
  rbind(d, 4:6)

- **rowSums/colSums**: processes the sum of rows/columns of a matrix
  rowSums(b); colSums(b)

Some useful commands

- **head**: check the first rows of an object:

  head(d)

- **tail**: check the last rows of an object:

  tail(d)

- By default, the 6 first/last rows are displayed

  change the **n** argument:

  head(d, n=10)

Exercise 4:
matrix manipulation

the **apply** function

- Powerful tool to **apply a command** to all rows or all columns of a data frame or a matrix

d

Maria	**23**	**TRUE**
Juan	**25**	**TRUE**
Alba	**31**	**FALSE**

apply(d, 2, table)

$Name
Alba Juan Maria
1 1 1

$Age
23 25 31
1 1 1

$Vegetarian
TRUE FALSE
2 1

Input / Output

- a <- **read.table**("file.txt",
  sep="\t", **# input column field separator**
  header=TRUE **# first row of file considered as header / column names vector**
)

- **write.table**(a, "file.txt",
  sep="\t", **# output column field separator**
  col.names=TRUE, **# col names written in output file**
  row.names=TRUE, **# row names written in output file**
  quote=FALSE) **# quote around character elements not written in output file**

Exercise 5:
Data frame manipulation

Libraries / packages

Library/packages

- **Packages** are collections of R functions, data, and compiled code in a well-defined format.

- The directory where packages are stored is called the **library**.

Definitions from http://www.statmethods.net/interface/packages.html

R-base: standard packages

- **R-base**:

  about 25 standard packages supplied with R by default (example: base, stats, graphics).

R-contrib: all other packages

- 2 main repositories:

  – CRAN: Comprehensive R Archive Network

  11038 packages available

  https://cran.r-project.org

  – Bioconductor:

  1383 packages available

  https://www.bioconductor.org/

Find packages

- R packages in CRAN:

https://cran.r-project.org/web/packages/

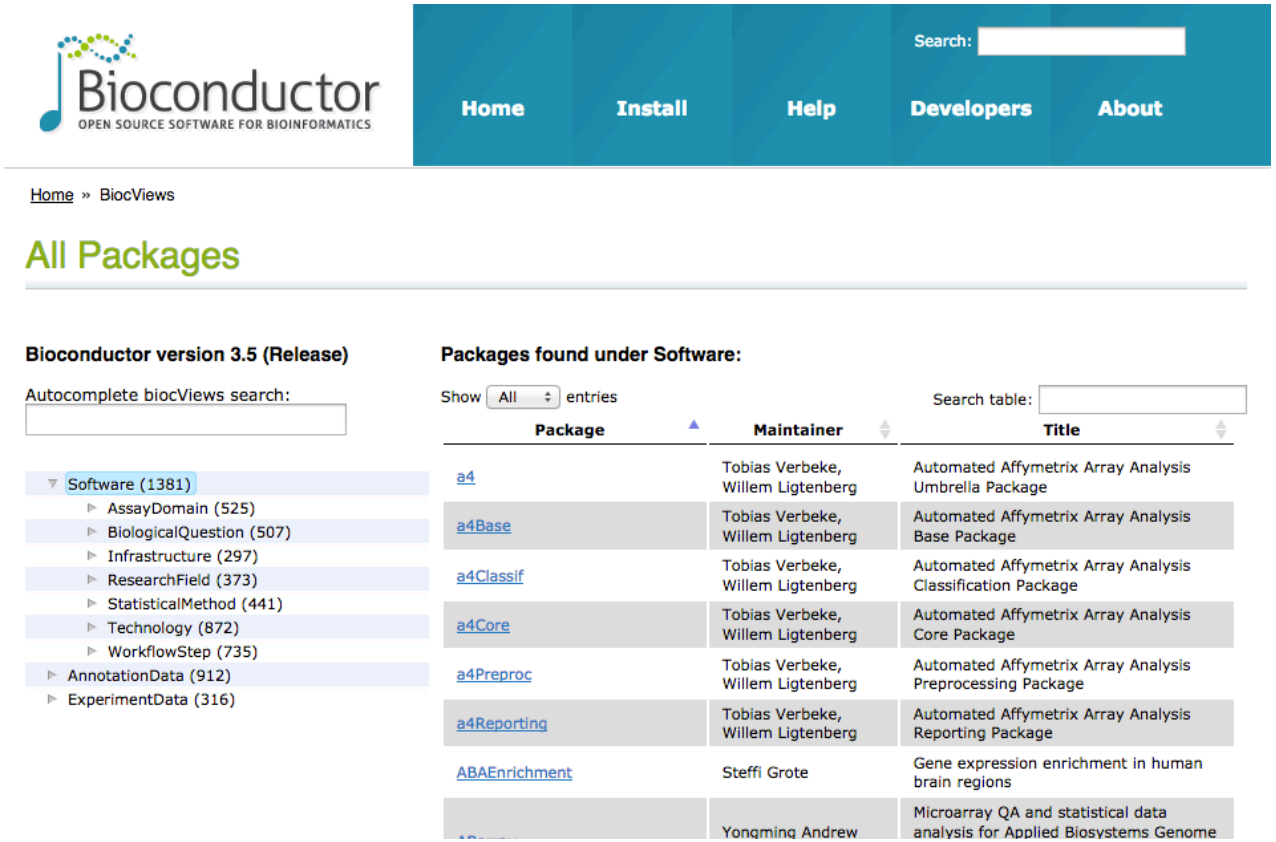
Available CRAN Packages By Date of Publication

Date	Package	Title
2017-05-21	aSPU	Adaptive Sum of Powered Score Test
2017-05-21	d3r	'd3.js' Utilities for R
2017-05-21	dynr	Dynamic Modeling in R
2017-05-21	edfReader	Reading EDF(+) and BDF(+) Files
2017-05-21	GlobalOptions	Generate Functions to Get or Set Global Options
2017-05-21	GWmodel	Geographically-Weighted Models
2017-05-21	IgorR	Read Binary Files Saved by 'Igor Pro' (Including 'Neuromatic' Data)
2017-05-21	imputeTS	Time Series Missing Value Imputation
2017-05-21	lagged	Classes and Methods for Lagged Objects
2017-05-21	mclust	Gaussian Mixture Modelling for Model-Based Clustering, Classification, and Density Estimation
2017-05-21	NPMOD	Non Parametric Module
2017-05-21	numGen	Number Series Generator
2017-05-21	penRvine	Flexible R-Vines Estimation Using Bivariate Penalized Splines
2017-05-21	PhenotypeSimulator	Flexible Phenotype Simulation from Different Genetic and Noise Models
2017-05-21	plfMA	A GUI to View, Design and Export Various Graphs of Data
2017-05-21	RANN	Fast Nearest Neighbour Search (Wraps ANN Library) Using L2 Metric
2017-05-21	regnet	Network-Based Regularization for Generalized Linear Models
2017-05-21	timereg	Flexible Regression Models for Survival Data
2017-05-20	AIG	Automatic Item Generator

Find packages

- R packages in Bioconductor:

https://bioconductor.org/packages



Bioconductor

- R packages specialized in bioinformatics analysis
  - Supports most types of genomics and NGS data (e.g. limma, DESeq2, BayesPeak)
  - Specific data classes (e.g. Granges from GenomicRanges)
  - Integrates command line tools (e.g Rsamtools)
  - Annotation tools (e.g. biomaRt)

Types of Bioconductor packages

- **Software**: set of functions

  e.g. DESeq2(NGS data analysis)

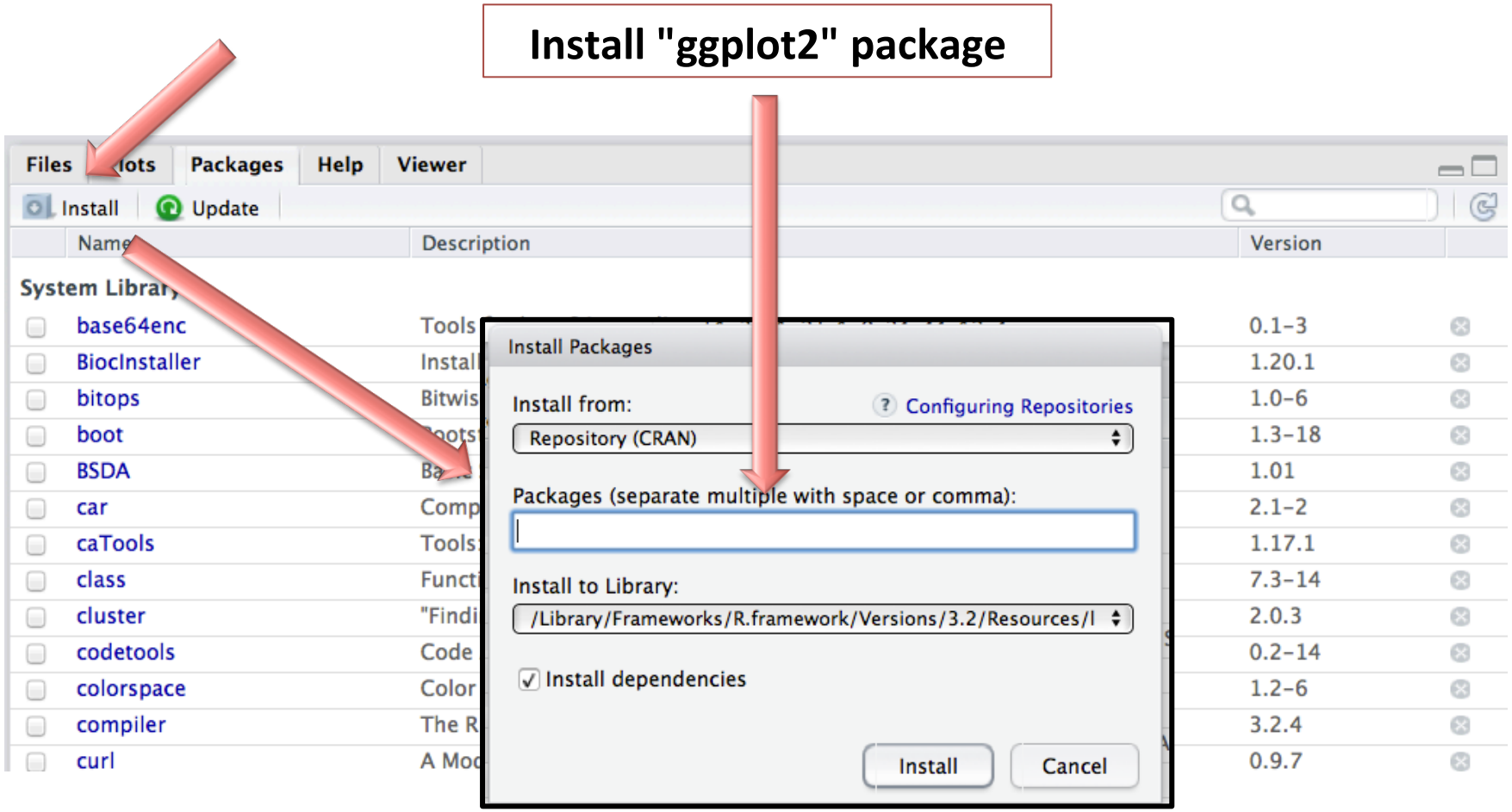
- **Annotation**: annotation of specific arrays, organisms, events, etc.

  e.g. BSgenome.Hsapiens.UCSC.hg38

- **Experiment**: data that can be loaded and used

  e.g. ALL (acute lymphoblastic leukemia dataset)

Install a package with R studio

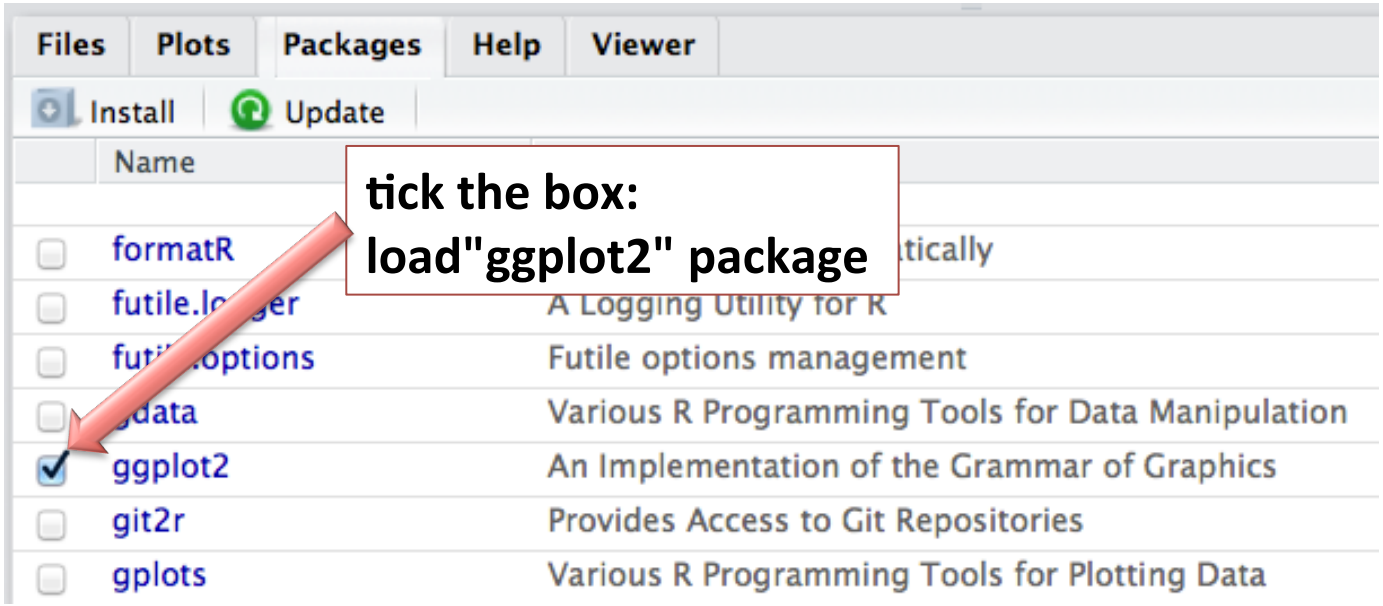


Load a package

- **from console**

  library(ggplot2**)**

- **with RStudio**



Listing functions from packages

- **ls(**"package:ggplot2"**)**

```
 [1] "%+%"                  "%+replace%"
 [3] "Coord"                "CoordCartesian"
 [5] "CoordFixed"             "CoordFlip"
 [7] "CoordMap"               "CoordPolar"
 [9] "CoordQuickmap"            "CoordTrans"
[11] "Geom"                  "GeomAbline"
[13] "GeomAnnotationMap"       "GeomArea"
[15] "GeomBar"               "GeomBlank"
…
```

R Studio server at CRG

- You need to specify a writeable directory to install packages into!
- Follow these steps:

```
setwd("~") # go to home directory
dir.create("R_packages") # create directory where to store packages
.libPaths("~/R_packages/") # add path to library path
```

Exercise 6:
Packages