

Tutorial 1

Introduction to R



A brief expectations management

- The course site is:
<https://webcourse.cs.technion.ac.il/236523/Spring2021/>
- The course is 2.5 academic points
- Two hours of lecture and one hour of tutorial a week
- The grade consists of 60% HWs + 40% project
- Four homeworks, that will be submitted in pairs. The submission is at the course site.
- The project is submitted in pairs.
- My email for your questions regarding course administration, material taught at the tutorials and the content of HWs:
alonar@technion.ac.il
- All the questions regarding the HW grades should be referred to Mahmoud Yazbek mahmudi@campus.technion.ac.il



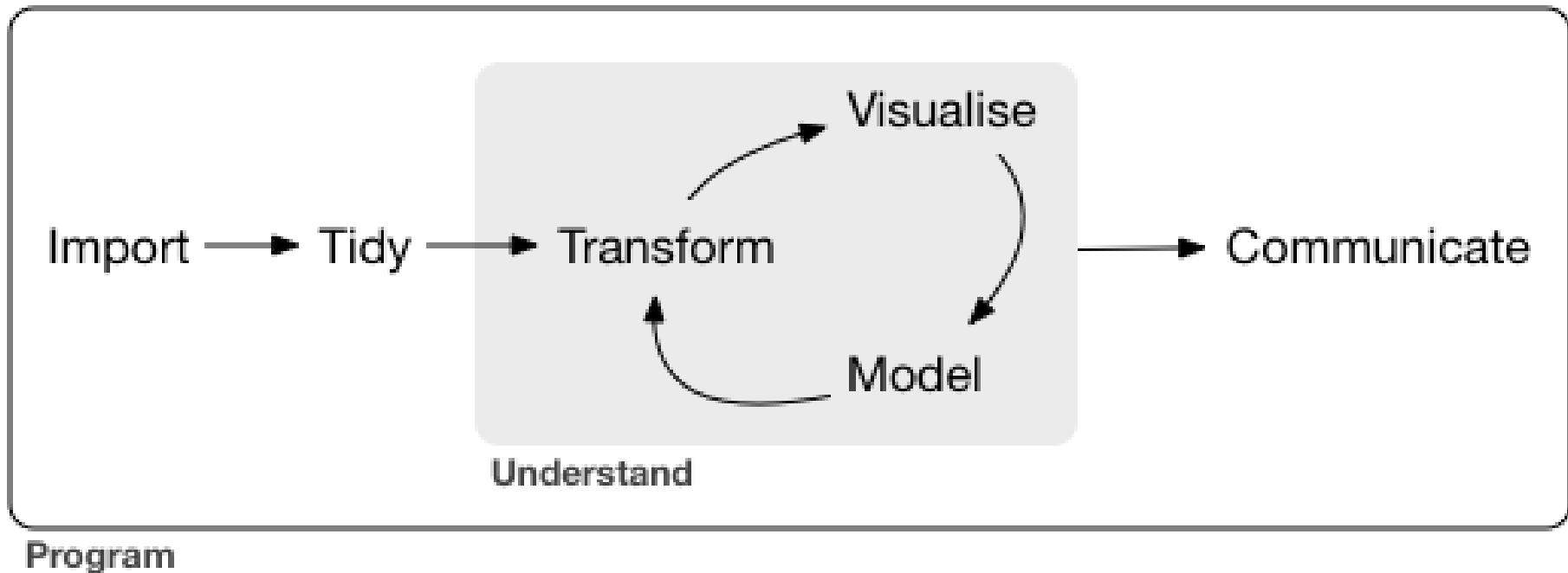
What is R ?

R is a programming language,
like C, C++.

R language can be interpreted
by computer, however almost
everyone uses R with an
Rstudio application.

Current version of R: 4.0.4

What are the main data analysis steps ?



How do we do it ?

The screenshot shows the RStudio interface with the following components:

- Console Pane:** Displays the R startup message and help text. The message includes:
 - R version 4.0.4 (2021-02-15) -- "Lost Library Book"
 - Copyright (C) 2021 The R Foundation for Statistical Computing
 - Platform: x86_64-w64-mingw32/x64 (64-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.
- Environment Pane:** Shows the Global Environment tab. It displays the message "Environment is empty".
- Files Pane:** Shows the Home > R folder. It contains a single folder named "win-library".

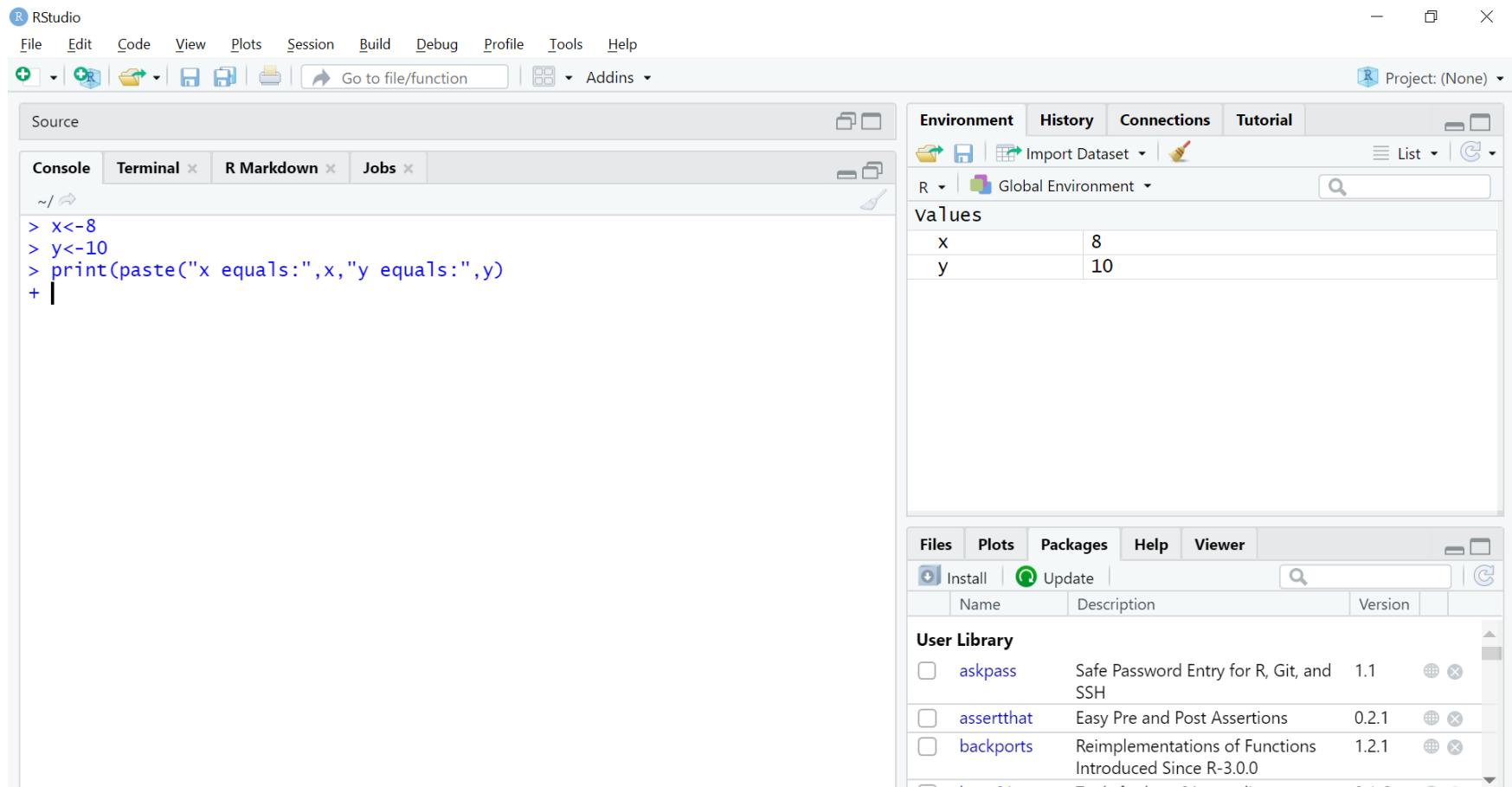
Naming convention

- Must start with letter (A-Z, a-z)
- Can contain letters, digits, underscore, period “.”
- Case sensitive (Cat.Food differs from cat.food)

Assignment

- Assignment is performed by “<-”

object_name <- value



The screenshot shows the RStudio interface with the following details:

- Console Panel:** Displays R code and its output:

```
> x<-8
> y<-10
> print(paste("x equals:",x,"y equals:",y))
+ |
```
- Environment Panel:** Shows the global environment with two variables defined:

values	x	8
y	10	
- Packages Panel:** Shows the user library with three packages installed:

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

RStudio's keyboard shortcut: Alt + - (the minus sign) gives <-



Assignment

The screenshot shows the RStudio interface with the following components:

- Top Bar:** RStudio, File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Toolbar:** Source, Console (selected), Terminal, Jobs, Addins, Project: (None).
- Console Tab:** Displays R code and its output:

```
> x<-8
> y<-10
> print(paste ("x equals:",x,"y equals:",y))
+ )
[1] "x equals: 8 y equals: 10"
> print(paste ("x equals:",x,"y equals:",y),quote = F)
[1] x equals: 8 y equals: 10
> |
```
- Environment Tab:** Shows the current environment variables:

Values	x	8
y	10	
- Packages Tab:** Shows the user library with two packages listed:

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertt...	Easy Pre and Post	0.2.1



Ctrl+L clears the console window

Data structures

Vectors

Matrices

Data frames

Lists

Factors

Data structures

- R has multiple data structures:
- **Vectors**
- Matrices
- Data frames
- Lists
- Factors

Vector is sequence of data elements of the same type (numeric, character or logical). Every column of a table will be represented as a vector.

Vectors

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar below has icons for New Project, Open Project, Save, Print, and Go to file/function. The status bar indicates 'Project: (None)'. The main area consists of several panes: a Source pane (disabled), a Console pane showing R code and its output, a Terminal pane, an R Markdown pane, and a Jobs pane. The Environment pane displays the global environment with objects t, x, y, and z. The Packages pane shows the User Library with packages askpass, assertthat, and backports. The bottom pane is the Viewer.

```
> x<-c(3,2,10,5) #create a vector named x with 4 components
> x
[1] 3 2 10 5
> y<-1:4 #create a vector of sequence of numbers.
> y
[1] 1 2 3 4
> z<-seq(0,100,by=10)
> z
[1] 0 10 20 30 40 50 60 70 80 90 100
> t<-seq(100,0,-1)
> t
[1] 100 99 98 97 96 95 94 93 92 91 90 89 88 87 86
[16] 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71
[31] 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56
[46] 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41
[61] 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26
[76] 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11
[91] 10 9 8 7 6 5 4 3 2 1 0
> t[5]
[1] 96
>
```



Use # to comment



[1] is indicating that the first element of the vector can be accessed with [1]. Subsequent lines show the appropriate index to access the first number in the line.

Some math ...

The screenshot shows the RStudio interface with the following components:

- File Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Toolbar:** Includes icons for New Project, Open Project, Save, Print, Go to file/function, and Addins.
- Source Pane:** Shows the R code entered in the console.
- Console Pane:** Displays the R session history with the following commands and outputs:

```
> x
[1] 3 2 10 5
> x+2 #scalar addition
[1] 5 4 12 7
> x*2 #scalar multiplication
[1] 6 4 20 10
> x^2 #raise each component to the second power
[1] 9 4 100 25
> x
[1] 3 2 10 5
> x<-x^2 #now x changes as we perform assignment
> x
[1] 6 4 20 10
> |
```
- Environment Pane:** Shows the current environment variables and their values:

x	num [1:4]	6	4	20	10
y	num [1:4]	3	2	10	5
- Packages Pane:** Shows the user library with one package listed:

Name	Description	Version
backpo...	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1



Up arrow gives you the last typed command.

Some math ...

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Source

Console Terminal R Markdown Jobs

Go to file/function Addins

Project: (None)

```
> x<-8
> y<-1:5
> y
[1] 1 2 3 4 5
> print (paste(x,y),quote=F)
[1] 8 1 8 2 8 3 8 4 8 5
>
```

Environment History Connections Tutorial

Import Dataset Global Environment

Values

t	num [1:11]	100 90 80 70 60 50 40 ...
x		8
y	int [1:5]	1 2 3 4 5
z	num [1:11]	0 10 20 30 40 50 60 70...

Files Plots Packages Help Viewer

Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

Assignment of mixed data types

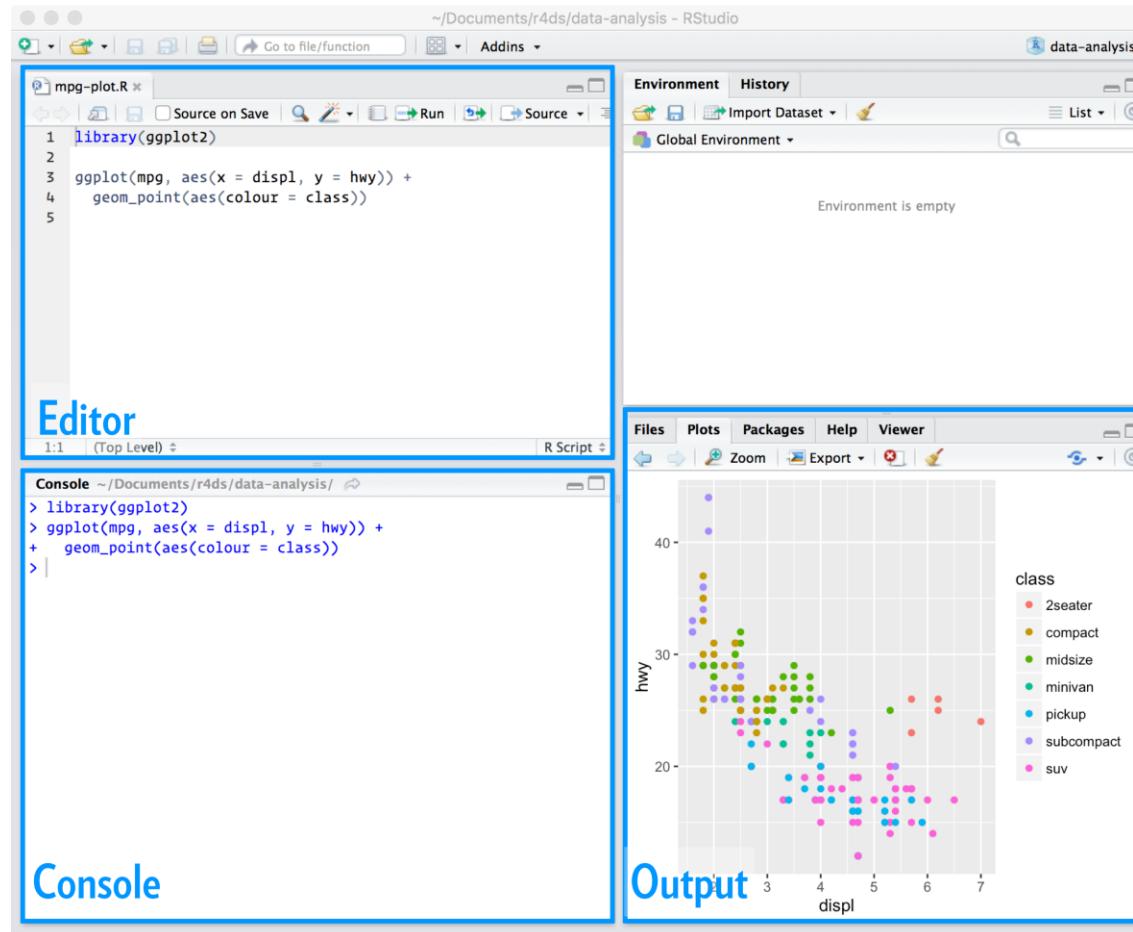
The screenshot shows the RStudio interface with the following details:

- File Menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Toolbar:** Includes icons for New Project, Open Project, Save, Print, Go to file/function, and Addins.
- Source Editor:** A tab labeled "Source" is visible.
- Console Tab:** The "Console" tab is active, showing the following R session:

```
> x<-c(5,"cold",3.2,"a")
> x
[1] "5"     "cold"   "3.2"   "a"
>
```
- Environment Tab:** Shows the variable "x" defined as a character vector [1:4] containing "5", "cold", "3.2", and "a".
- Files Tab:** Shows the "User Library" with the following packages listed:

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

Script editor vs console



-  Ctrl + Enter shortcut allows you to execute every line/code chunk at the script editor.

Script editor vs console



- Instead of executing expression-by-expression, you can also execute the complete script in one step using Ctrl+Shift+Enter.
- Start your script with the packages that you need. If you'll share your code with others, they can easily see what packages they need to install/load.
- You can use a console to execute parts of code, or you can use it observe the results of the script/parts of script execution



Data structures

- R has multiple data structures:
 - Vectors
 - **Matrices**
 - Data frames
 - Lists
 - Factors
-
- A matrix is an array of rows and columns of the same data type. One of the easiest ways to create a matrix is to combine vectors of equal length using `cbind()` function.

Matrix

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Project: (None)

Tutorial_1.R*

```
1 x<-c(6,7,5,4)
2 y<-c(4,13,6,15)
3 z<-c(14,3,5,7)
4 m1<-cbind(x,y,z)
5 m1
6 dim(m1)
```

8:1 (Top Level) R Script

Console

```
> x<-c(6,7,5,4)
> y<-c(4,13,6,15)
> z<-c(14,3,5,7)
> m1<-cbind(x,y,z)
> m1
   x  y  z
[1,] 6  4 14
[2,] 7 13  3
[3,] 5  6  5
[4,] 4 15  7
> dim(m1)
[1] 4 3
> t(m1)
   [,1] [,2] [,3] [,4]
x     6     7     5     4
y     4    13     6    15
z    14     3     5     7
>
```

Environment

Import Dataset Global Environment

Data	m1	num [1:4, 1:3]	6 7 5 4 4 13 6 1...
values	t	num [1:11]	100 90 80 70 60 50 40 ...
x	y	num [1:4]	6 7 5 4
z		num [1:4]	4 13 6 15
		num [1:4]	14 3 5 7

Files

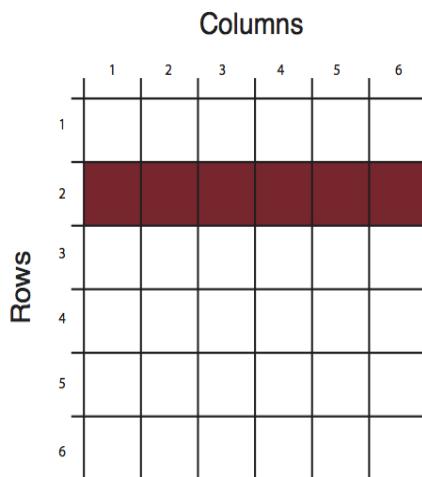
Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

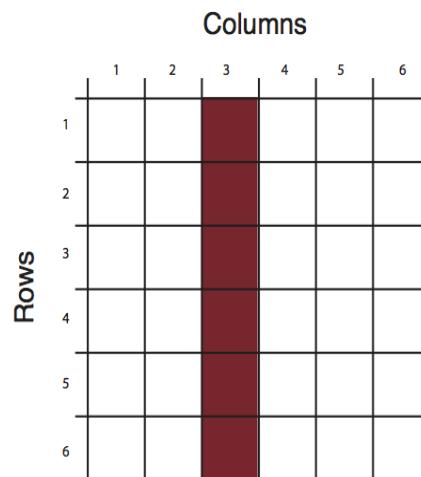
Matrix

Matrices and data frames, are tabular data structures. These are two dimensional datasets in which data in the any specific column is of the same type. You can subset those data structures using [] and providing desired rows and columns to subset.

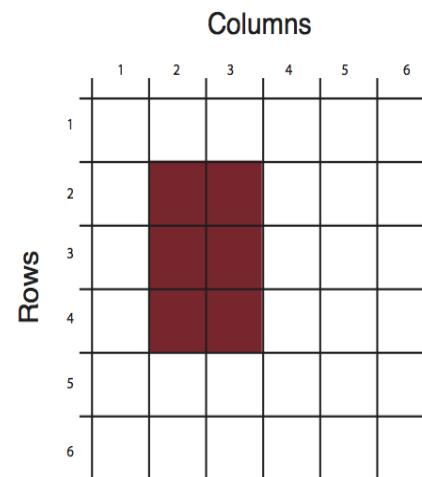
`mat[2,]`



`mat[, 3]`



`mat[2:4, 2:3]`



Matrix

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

+ - Addins ▾ Project: (None)

Tutorial_1.R* | Go to file/function | Addins ▾

1 m1
2 m1[,1]
3 m1[2:4,2:3]

4

4:1 (Top Level) ▾ R Script ▾

Console Terminal R Markdown Jobs

~/

> m1

	x	y	z
[1,]	6	4	14
[2,]	7	13	3
[3,]	5	6	5
[4,]	4	15	7

> m1[,1]

[1]	6	7	5	4
-----	---	---	---	---

> m1[2:4,2:3]

	y	z
[1,]	13	3
[2,]	6	5
[3,]	15	7

>

Environment History Connections Tutorial

Import Dataset Global Environment

Data

m1	num [1:4, 1:3]	6	7	5	4	4	13	6	1...
t	num [1:11]	100	90	80	70	60	50	40	...
x	num [1:4]	6	7	5	4				
y	num [1:4]	4	13	6	15				
z	num [1:4]	14	3	5	7				

Values

Files Plots Packages Help Viewer

Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

User Library

Matrix

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

+ Go to file/function Addins Project: (None)

Tutorial_1.R*

Source on Save Run Source

```
1 m1 <- matrix(1:8, nrow=2)
2 m1
3 m2<- matrix(1:8,nrow=2, byrow=TRUE)
4 m2
```

4:3 (Top Level) R Script

Console Terminal R Markdown Jobs

~/Bioinformatics/RStudio_wdir/

```
> m1 <- matrix(1:8, nrow=2)
> m1
 [,1] [,2] [,3] [,4]
[1,] 1 3 5 7
[2,] 2 4 6 8
> m2<- matrix(1:8,nrow=2, byrow=TRUE)
> m2
 [,1] [,2] [,3] [,4]
[1,] 1 2 3 4
[2,] 5 6 7 8
>
```

Environment History Connections Tutorial

Import Dataset Global Environment

Data

m1	int [1:2, 1:4]	1 2 3 4 5 6 7 8
m2	int [1:2, 1:4]	1 5 2 6 3 7 4 8

Values

res	num [1:4]	64 40 48 56
x		8
y	num [1:4]	8 5 6 7

Files Plots Packages Help Viewer

Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

By default, the matrix is filled column-by-column. If you set `byrow=TRUE` the matrix will be filled row by row



Data structures

- R has multiple data structures:
 - Vectors
 - Matrices
 - **Data frames**
 - Lists
 - Factors
-
- A data frame is a $n \times m$ collection of variables (in the columns) and observations (in the rows). Data frame is more general data structure than matrix, as its columns can have different data types (numeric, character, factor, etc.)

Data frame

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Tutorial_1.R*

```
1 chr <- c("chr1", "chr1", "chr2", "chr2")
2 strand <- c("-", "-", "+", "+")
3 start<- c(200,4000,100,400)
4 end<-c(250,410,200,450)
5 mydata <- data.frame(chr,start,end,strand)
6 mydata
```

6:7 (Top Level) R Script

Console Terminal R Markdown Jobs

```
> chr <- c("chr1", "chr1", "chr2", "chr2")
> strand <- c("-", "-", "+", "+")
> start<- c(200,4000,100,400)
> end<-c(250,410,200,450)
> mydata <- data.frame(chr,start,end,strand)
> mydata
```

	chr	start	end	strand
1	chr1	200	250	-
2	chr1	4000	410	-
3	chr2	100	200	+
4	chr2	400	450	+

Environment History Connections Tutorial

Import Dataset Global Environment

Data

m1	num [1:4, 1:3]	6 7 5 4 4 13 6 1...
mydata	4 obs. of 4 variables	

Values

chr	chr [1:4]	"chr1" "chr1" "chr2" "c...
end	num [1:4]	250 410 200 450
start	num [1:4]	200 4000 100 400
strand	chr [1:4]	"-" "-" "+" "+"
t	num [1:11]	100 90 80 70 60 50 40 ...
x	num [1:4]	6 7 5 4
y	num [1:4]	4 13 6 15
z	num [1:4]	14 3 5 7

Files Plots Packages Help Viewer

Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

User Library

askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

Data frame

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

+ Go to file/function Addins Project: (None)

Tutorial_1.R*

```
1 #change column names
2 names(mydata) <- c("chr","start","foo","bar")
3 mydata
4 mydata[,2:4] # columns 2,3,4 of data frame
```

1:1 (Top Level) R Script

Console Terminal R Markdown Jobs

```
> #change column names
> names(mydata) <- c("chr","start","foo","bar")
> mydata
  chr start foo bar
1 chr1   200 250  -
2 chr1   4000 410  -
3 chr2   100 200  +
4 chr2   400 450  +
> mydata[,2:4] # columns 2,3,4 of data frame
  start foo bar
1     200 250  -
2    4000 410  -
3     100 200  +
4     400 450  +
```

Environment History Connections Tutorial

Import Dataset Global Environment

Data

m1	num [1:4, 1:3]
mydata	4 obs. of 4 variables

values

chr	chr [1:4]
end	num [1:4]
start	num [1:4]
strand	chr [1:4]
t	num [1:11]
x	num [1:4]
y	num [1:4]
z	num [1:4]

Files Plots Packages Help Viewer

Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

User Library

Data frame

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Tutorial_1.R*

Source on Save Run Source

```
1 mydata
2 mydata$start # variable start in the data frame
3 mydata[c(1,3),] # get 1st and 3rd rows
4 mydata[mydata$start>400,] # get all rows where start>400
5
```

5:1 (Top Level) R Script

Console Terminal R Markdown Jobs

~/

```
> mydata
  chr start foo bar
1 chr1 200 250 -
2 chr1 4000 410 -
3 chr2 100 200 +
4 chr2 400 450 +
> mydata$start # variable start in the data frame
[1] 200 4000 100 400
> mydata[c(1,3),] # get 1st and 3rd rows
  chr start foo bar
1 chr1 200 250 -
3 chr2 100 200 +
> mydata[mydata$start>400,] # get all rows where start>400
  chr start foo bar
2 chr1 4000 410 -
```

Environment History Connections Tutorial

Import Dataset Global Environment

Data

m1	num [1:4, 1:3] 6 7 5 4 4 13 6 1...
mydata	4 obs. of 4 variables

Values

chr	chr [1:4] "chr1" "chr1" "chr2" "c...
end	num [1:4] 250 410 200 450
start	num [1:4] 200 4000 100 400
strand	chr [1:4] "-" "-" "+" "+"
t	num [1:11] 100 90 80 70 60 50 40 ...
x	num [1:4] 6 7 5 4
y	num [1:4] 4 13 6 15
z	num [1:4] 14 3 5 7

Files Plots Packages Help Viewer

Install Update

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

User Library

askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

Data structures

- R has multiple data structures:
 - Vectors
 - Matrices
 - Data frames
 - **Lists**
 - Factors
-
- A list is an ordered collection of components. A list allows you to gather a variety of often unrelated objects of different length and type under one name. Each object or element in the list has a numbered position and can have names. The list can be created with the `list()` function.

Lists

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Project: (None)

Tutorial_1.R*

example of a list with 4 components
a string, a numeric vector, a matrix, and a scalar
w <- list(name="Alona",
 mynumbers=c(6,4,3,2,1),
 mymatrix=matrix(1:8,ncol=2),
 age=85)
w

7:2 (Top Level) ▾ R Script

Console Terminal Jobs

~ /
\$name
[1] "Alona"

\$mynumbers
[1] 6 4 3 2 1

\$mymatrix
[,1] [,2]
[1,] 1 5
[2,] 2 6
[3,] 3 7
[4,] 4 8

\$age
[1] 85

Environment History Connections Tutorial

Import Dataset

Global Environment

m1 num [1:4, 1:2] 6 7 5 4 4 13...
mydata 4 obs. of 4 variables
w List of 4

values

chr chr [1:4] "chr1" "chr1" "chr2..."
end num [1:4] 250 410 200 450
i 1L
result NULL
start num [1:4] 200 4000 100 400
strand chr [1:4] "-" "-" "+" "+"
x num [1:4] 6 7 5 4
v num [1:4] 4 13 6 15

Files Plots Packages Help Viewer

Install Update

User Library

askpass Safe Password Entry for R, Git, and SSH 1.1
assertthat Easy Pre and Post Assertions 0.2.1
backports Reimplementations of Functions Introduced Since R-3.0.0 1.2.1

Lists

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Tutorial_1.R*

Source on Save Run Source

```
1 # example of a list with 4 components
2 # a string, a numeric vector, a matrix, and a scalar
3 w <- list(name="Alona",
4           mynumbers=c(6,4,3,2,1),
5           mymatrix=matrix(1:8,ncol=2),
6           age=85)
7 w
```

(Top Level) R Script

Console Terminal Jobs

```
> w[[3]] # 3rd component of the list
 [,1] [,2]
[1,] 1 5
[2,] 2 6
[3,] 3 7
[4,] 4 8
> w[["mynumbers"]] # component named mynumbers in list
[1] 6 4 3 2 1
>
> w$age
[1] 85
> w$
```

name mynumbers mymatrix age

Environment History Connections Tutorial

Import Dataset Global Environment

m1	num [1:4, 1:2]
mydata	4 obs. of 4 variables
w	List of 4

Values

chr	chr [1:4]
end	num [1:4]
i	1L
result	NULL
start	num [1:4]
strand	chr [1:4]
x	num [1:4]
v	num [1:4]

Files Plots Packages Help Viewer

Install Update

Name	Description	Vers...
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

User Library

- askpass
- assertthat
- backports

Data structures

- R has multiple data structures:
- Vectors
- Matrices
- Data frames
- Lists
- **Factors**

- Factors are data structures that are used to store categorical data. We will elaborate on this type of data structure at our next tutorials.

If/else/ifelse

```
if (condition) {  
    # executed when condition is TRUE  
} else {  
    # executed when condition is FALSE  
}
```

An opening curly brace should never go on its own line and should always be followed by a new line. A closing curly brace should always go on its own line, unless it's followed by else. Always indent the code inside curly braces.

If/else/ifelse

Ifelse(test, yes, no)

ifelse returns a value with the same shape as test which is filled with elements selected from either yes or no depending on whether the element of test is TRUE or FALSE.

The screenshot shows the RStudio interface with the following components:

- File Menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Toolbar:** Includes icons for New, Open, Save, Print, Go to file/function, Run, Source, and Addins.
- Project Bar:** Project: (None).
- Code Editor:** Tutorial_1.R contains the following R code:

```
1 x<-1:4
2 ifelse(x=="Blue","Yey!","Neh")
3 ifelse(x!="Blue","Yey!","Neh")
```
- Environment Tab:** Shows the variable x defined as an integer vector [1:4] with values 1, 2, 3, 4.
- Console Tab:** Displays the R session output:

```
> x<-1:4
> ifelse(x=="Blue","Yey!","Neh")
[1] "Neh" "Neh" "Neh" "Neh"
> ifelse(x!="Blue","Yey!","Neh")
[1] "Yey!" "Yey!" "Yey!" "Yey!"
>
```
- Packages Tab:** Shows the User Library with two packages listed: askpass and assertthat.

For

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

+ Project: (None) ▾

Tutorial_1.R*

Source on Save Run Source

```
1 for (i in 1:10) { #number of repetitions
2   print (paste("This is iteration",i)) #the task to be repeated
3 }
```

3:2 (Top Level) R Script

Console Terminal R Markdown Jobs

```
> for (i in 1:10) { #number of repetitions
+   print (paste("This is iteration",i)) #the task to be repeated
+ }
[1] "This is iteration 1"
[1] "This is iteration 2"
[1] "This is iteration 3"
[1] "This is iteration 4"
[1] "This is iteration 5"
[1] "This is iteration 6"
[1] "This is iteration 7"
[1] "This is iteration 8"
[1] "This is iteration 9"
[1] "This is iteration 10"
>
```

Environment History Connections Tutorial

Import Dataset

R Global Environment

values i 10L

Files Plots Packages Help Viewer

Install Update

User Library

Name	Description	Version
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1

Functions

- You should consider writing a function if you used the same block of code more than twice. Function name must be clear. Don't worry about name length as Rstudio has a perfect autocomplete.



Repeatedly copy-paste
same chunks of code
in your script



Write function

Functions

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

+ - Project: (None) X

Tutorial_1.R x sqSum.R x Go to file/function Addins

Source on Save Run Source

```
1 sqSum<-function(x,y){  
2   result=x^2+y^2  
3   return(result)  
4 }  
5
```

5:1 (Top Level) R Script

Console Terminal Jobs

~/ >

Environment History Connections Tutorial

Import Dataset List

R Global Environment Values

x	3
y	2
z	13

Functions

sqSum	function (x, y)
-------	-----------------

Files Plots Packages Help Viewer

Install Update Name Description Version

User Library

askpass	Safe Password Entry for R, Git, and SSH	1.1
assertt...	Easy Pre and Post	0.2.1

This screenshot shows the RStudio interface. The top navigation bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The main menu bar has tabs for Tutorial_1.R and sqSum.R. Below the tabs are buttons for Source on Save, Run, and Source. The code editor contains the following R function:

```
1 sqSum<-function(x,y){  
2   result=x^2+y^2  
3   return(result)  
4 }  
5
```

The environment pane on the right shows variables x, y, and z with values 3, 2, and 13 respectively. The functions pane shows the definition of sqSum. The packages pane at the bottom lists 'askpass' and 'assertt...' in the User Library.

Functions

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Project: (None)

Tutorial_1.R* sqSum.R

Source on Save Run Source

1 z=sqSum(3,2)
2 z

2:2 (Top Level) R Script

Console Terminal Jobs

~/

```
> z=sqSum(3,2)
> z
[1] 13
>
```

Environment History Connections Tutorial

Import Dataset Global Environment

Values

x	3
y	2
z	13

Functions

sqSum	function (x, y)
-------	-----------------

13

Files Plots Packages Help Viewer

Install Update

Name	Description	Ver...
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertt...	Easy Pre and Post	0.2.1

User Library

askpass Safe Password Entry for R, Git, and SSH 1.1
assertt... Easy Pre and Post 0.2.1

Apply functions family

- Try to do your best to replace every “for” by the helper functions from the “apply” family.



Apply functions family

- The apply family of functions in base R (apply(), lapply(), tapply(), etc) consists of functions that eliminate the need for many common “for” loops.
- The difference between these is that they take different types of inputs. For example, apply works on data frames or matrices and applies the function on each row or column of the data structure.



Another option to get rid of “for” loops is to get familiar with the functions in the purrr package
purrr cheatsheet is here:

<https://github.com/rstudio/cheatsheets/blob/master/purrr.pdf>



Apply functions family

R RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

+ | Project: (None) |

Tutorial_1.R* | Go to file/function | Addins |

```
1 m1=cbind(c(3,0,3,3),c(3,0,0,0),c(3,0,0,3),c(1,1,0,0),c(1,1,1,0),c(1,1,1,0))
2 m1
3 #second argument of apply that equals 1,indicates that rows of the matrix/ data frame
4 #will be the input for the function.
5 result<-apply(m1,1,sum)
6 result
7 #second argument of apply that equals 2,indicates that columns of the matrix/ data frame
8 #will be the input for the function.
9 result<-apply(m1,2,sum)
10 result
```

1:1 (Top Level) | R Script

Console Terminal R Markdown Jobs

```
~/
> m1=cbind(c(3,0,3,3),c(3,0,0,0),c(3,0,0,3),c(1,1,0,0),c(1,1,1,0),c(1,1,1,0))
> m1
 [,1] [,2] [,3] [,4] [,5] [,6]
[1,] 3 3 3 1 1 1
[2,] 0 0 0 1 1 1
[3,] 3 0 0 0 1 1
[4,] 3 0 3 0 0 0
> #second argument of apply that equals 1,indicates that rows of the matrix/ data frame
> #will be the input for the function.
> result<-apply(m1,1,sum)
> result
[1] 12 3 5 6
>
```

Environment History Connection

Import Dataset List

Data Global Environment

m1	num [1:4, 1:6]
i	10L
result	num [1:4] 12 3 5...

Files Plots Packages Help V

Install Update

Name	Description
ask...	Safe Password Entry for R, Git, and SSH
ass...	Easy Pre and Post Assertions
har...	Reimplementation 1.2

User Library

Apply functions family

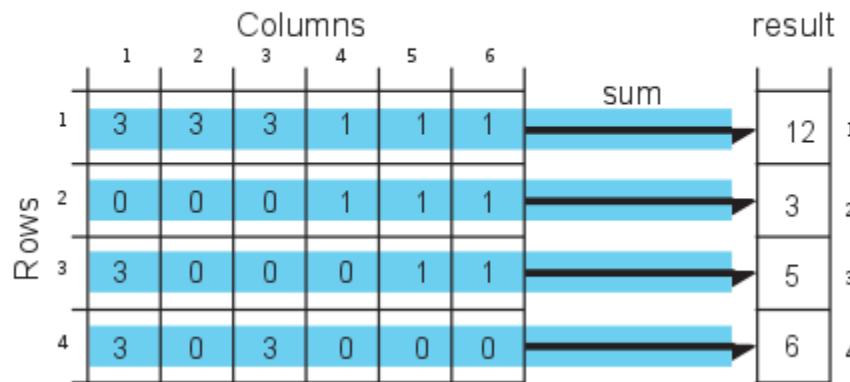
The screenshot shows the RStudio interface with the following components:

- Editor:** Displays the script `Tutorial_1.R*` containing R code. The code creates a matrix `m1` and applies the `sum` function along different dimensions.
- Console:** Shows the output of the R code. It prints the matrix `m1` and its row sums `result`.
- Environment:** Shows the global environment with objects `m1`, `values`, `i`, and `result`.
- Packages:** Shows the user library with packages `ask...`, `ass...`, and `hac`.

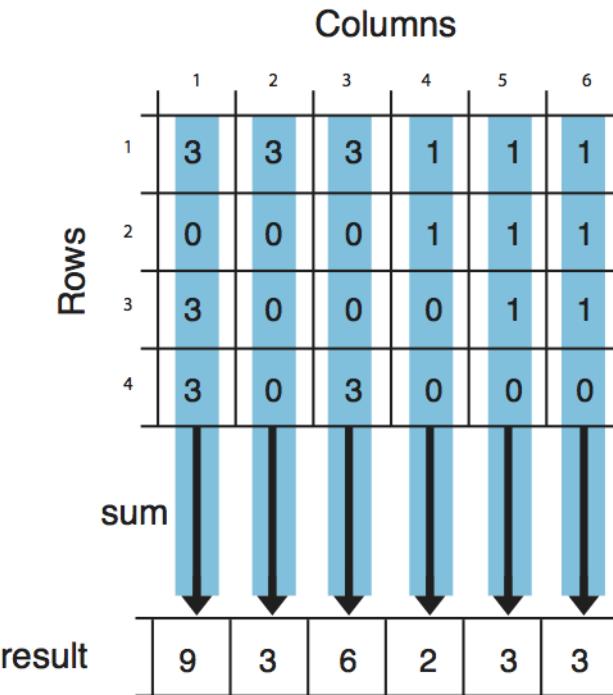
```
1 m1=cbind(c(3,0,3,3),c(3,0,0,0),c(3,0,0,3),c(1,1,0,0),c(1,1,1,0),c(1,1,1,0))
2 m1
3 #second argument of apply that equals 1,indicates that rows of the matrix/ data frame
4 #will be the input for the function.
5 result<-apply(m1,1,sum)
6 result
7 #second argument of apply that equals 2,indicates that columns of the matrix/ data frame
8 #will be the input for the function.
9 result<-apply(m1,2,sum)
10 result
> [1] 9 3 6 2 3 3
```

Apply functions family

```
result<-apply(mat,1,function(x) sum(x) )  
result<-apply(mat,1,sum)
```



```
result<-apply(mat,2,function(x) sum(x) )  
result<-apply(mat,2,sum)
```



More tips...



- Break your code into a well explained blocks

The screenshot shows the RStudio IDE interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The main window displays a script editor titled "Tutorial_1.R*" containing the following R code:

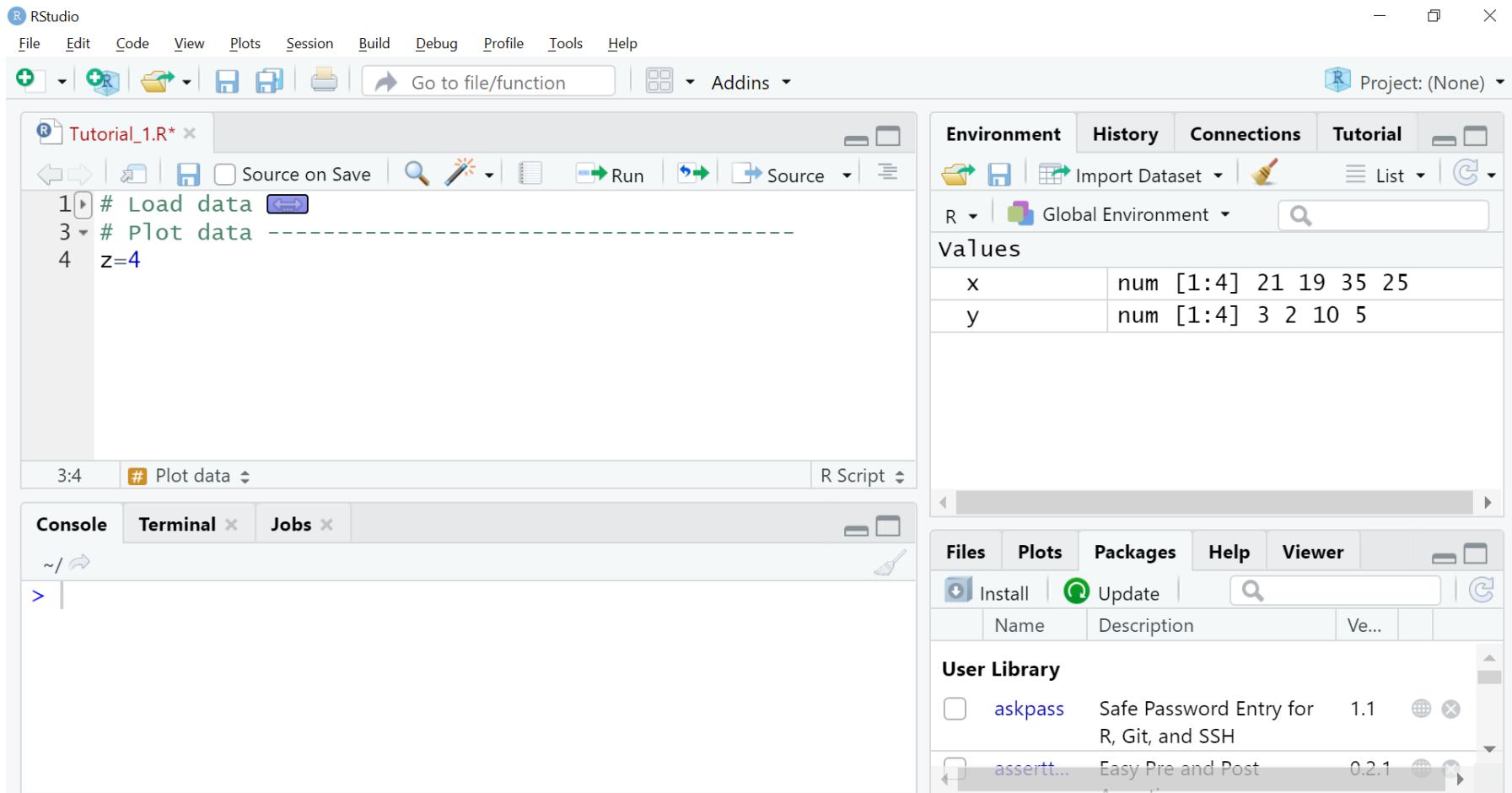
```
1 # Load data
2 t=c(1,2,3)
3 # Plot data
4 z=4
```

Below the script editor is a toolbar with icons for Load data and Plot data. The bottom status bar shows "R Script".

The right side of the interface features several panes:

- Environment** pane: Shows the Global Environment with variables x and y. Variable x is a numeric vector [1:4] with values 21, 19, 35, 25. Variable y is a numeric vector [1:4] with values 3, 2, 10, 5.
- Tutorial** pane: A tabbed pane showing "Import Dataset" and "List".
- Plots** pane: A small preview area.
- Packages** pane: A tabbed pane showing "Install" and "Update". It lists the "User Library" with packages "askpass" (version 1.1) and "assertt..." (version 0.2.1).

More tips...



Error messages in the editor window

A screenshot of the RStudio interface. On the left, the code editor shows a script named 'Tutorial_1.R' with the following content:

```
1 print(x, ...)  
2 y=10  
3 print("X equals to:",x,"y equals to:"y)  
unexpected token 'y'
```

A red arrow points to the line 'unexpected token 'y'' in the code editor. In the bottom-left corner, the console window shows the following session history:

```
> x<-8  
> y=10  
>
```

The top right panel displays the Global Environment, showing variables x and y with values 8 and 10 respectively. The bottom right panel shows the User Library with two packages listed: 'askpass' and 'assertthat'.

What's next ?

- At our next tutorial we'll discover the wonders of the plotting and we'll make statistics our dedicated partner for the rest of the semester.