
[illegible]

- [illegible]

[illegible]

- [illegible]

[illegible][illegible]

Navigating an Online Code-Along Workshop

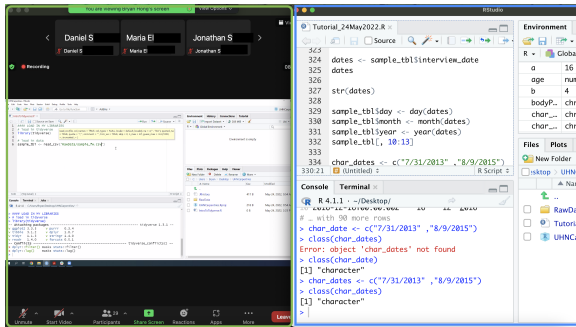


Figure: It is recommended that windows are resized so that both the user RStudio window and Instructor Zoom window (with RStudio) is visible at the same time. User may collapse panels of their RStudio not in current use.

Time	Topic
10.00 -10.20 am	Introduction
10.20 - 10.50 am	Setup and RStudio
10.50 - 11.00 am	Short Break
11.00 - 12.15 pm	Basics + Vectors
12.15 - 1.00 pm	Lunch
1.00 - 2.00 pm	Matrices, Lists, Data Frames
2.00 - 2.10 pm	Short Break
2.10 - 3.10 pm	Data Import/Export; Functions
3.10-3.20 pm	Short Break
3.20 - 3.45 pm	Graphics; Next Steps

Any questions?

R and RStudio



Top skills: Data Scientist

Rank	Skill	Percent of jobs
1	python	79%
2	machine learning	72%
3	r	64%
4	sql	53%
5	hadoop	29%
6	spark	28%
7	java	25%
8	sas	24%
9	tableau	21%
10	deep learning	20%



[illegible]

- [illegible]

[illegible]

- R is a scripting language, thus an interpreter executes commands one line at a time.
- A Free software under the terms of the GNU General Public License.
- R home page: <https://www.R-project.org/>
- How can R be obtained?
 - Via CRAN, the “Comprehensive R Archive Network”.
 - <https://cran.r-project.org/>



- How can R be installed?
 - Unix
 - https://cran.r-project.org/doc/FAQ/R-FAQ.html#How-can-R-be-installed-_0028Unix_002dlike_0029
 - Windows
 - <https://cran.r-project.org/bin/windows/base/>
 - Mac
 - <https://cran.r-project.org/bin/macosx/>



R continue...

R GUI

R Console

```
R version 2.11.0 (2010-04-23)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

> 1+2
[1] 3
> |
```

R Console

```
R version 3.3.3 (2017-03-06) -- "Another Canoe"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (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.

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.

[R.app GUI 1.69 (7328) x86_64-apple-darwin13.4.0]

|

executing: load("/Users/julienassouline/.RData")
```


- Ihaka,R. and Gentleman,R. (1996) R: a language for data analysis and graphics. *J. Comput. Graph. Statist.*, 5, 299–314.



- RStudio is an integrated development environment R.
- Not only for R, but can also use Python.
- Has:
 - A console
 - Syntax-highlighting editor for code execution
 - Tools for plotting, viewing history, debugging and workspace management

Why learn R?

- Not only is R free, but it is also open-source and cross-platform.
- R code is great for reproducibility.
- R is interdisciplinary and extensible.
- R works on data of all shapes and sizes.
- R produces high-quality graphics.
- R has a large and welcoming community.

Documentation for R

- Online documentation for functions and variables in R exists.
- Obtained by typing *help(FunctionName)* or *?FunctionName* at the R prompt, where FunctionName is name of function.
- E.g., if 'sum' is the function then:

```
> help(sum)
> ?sum
```

RStudio

- RStudio contains many features that make the development process easier and faster.

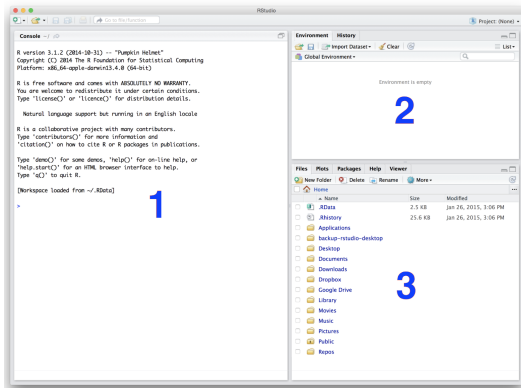


Figure: Anatomy of default RStudio. 1. This is the Console. 2. Environment and History. 3. Files, Plots, Packages, Help and Viewer. If a script is opened up, it will appear on top of Console.

Options To Work With R

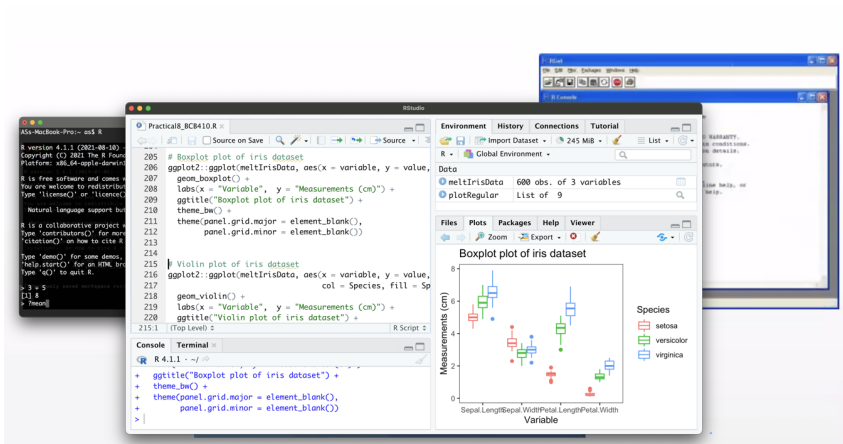


Figure: Some options to work with R. Several other options are present including the Jupyter Notebook.

Any questions?

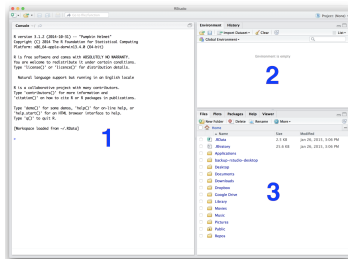
Practical - Setup

- «В-РКР АНН / ИИ V 6 & C-99S»

Practical - Explore RStudio

RStudio

- By now, you should have RStudio installed.



- There are two main ways of interacting with R:
 - Using the console
 - By using script files
- Click on 'Tools' → 'Keyboard Shortcuts Help' for shortcuts.

Interacting with R

- Console:
 - Type commands directly into the console and press 'Enter' to execute.
- Script:
 - Put cursor at the end of the line to execute OR highlight the section.
 - Press 'Ctrl' + 'Enter' on Windows, Mac OR 'Cmd' + 'Return' on Mac.
- Clear console with 'Ctrl' + 'L'.
- If R is still waiting for you to enter more text, the console will show a + prompt.

R Project

- Good to keep data, analyses, and text in a single folder.
- RStudio interface for this is Projects.
 - File → New project; choose New directory → New project
- Enter a name for this new folder (“directory”) and choose a convenient location for it. This will be your working directory.
 - On Desktop, save as ‘DSI_IntroR’
- Click on ‘Create’ project.
- Create a new file where we will type our scripts.
 - Go to File → New File → R script. Click the save icon on your toolbar and save your script as “script.R”.

[illegible]

- [illegible]

[illegible]

- [illegible]

[illegible][illegible][illegible][illegible]

Any questions?

Some Basics

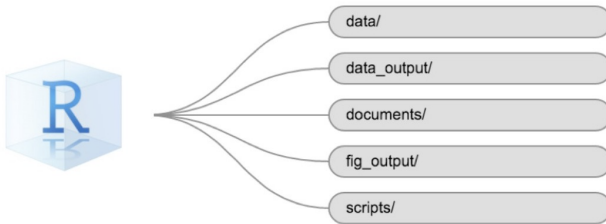


Figure: Examples of suggested directories within working directory or R Project.
Figure from: <https://datacarpentery.org/r-socialsci/00-intro/index.html>

R Coding Style

- Limit yourself to 80 characters per line.
- Use comments. Don't describe what the code does, but explain why you wrote it that way.
- Use only \leftarrow for assignment, not $=$.
- Never reassign reserved words.
- You may read more:
 - <https://google.github.io/styleguide/Rguide.html>
 - http://steipe.biochemistry.utoronto.ca/abc/index.php/RPR-Coding_style

[illegible]

- [illegible]

Assignment and Commenting

- The \leftarrow symbol is the assignment operator.
- To assign a value to a variable or object called 'test1'

```
test1 <- 123  
test1
```

- Comment using # character

```
test1 <- 123 # This is a comment  
test1 # This is called auto-printing
```

[illegible]

- [illegible]

- ```
library(help = "base")
library(help = "stats")
```

\_\_\_\_\_

[illegible]

- [illegible]

[illegible]

- [illegible]

[illegible]

- [illegible]

[illegible]

10. *Journal of the American Medical Association*, 2000; 284: 2689-2694.

1. 3. (11.1.11)  $\mu_1 = 3$

Any questions?

# R Data Types

- Numeric: floating types (double precision).
- Logicals: booleans = TRUE/FALSE or T/F.
- Character strings.
- Examples:

```
xValue <- 100
xValue
```

```
yVariable <- FALSE
yVariable
```

```
zVariable <- "hello"
zVariable
```

\_\_\_\_\_

\_\_\_\_\_

NaN # not a number



\_\_\_\_\_

```
x <- c(0.5, 0.6)
```

```
class(x) # "numeric"
```

```
x <- c("a", "b", "c")
```

```
class(x) # "character"
```

```
x <- c(TRUE, FALSE)
```

```
class(x) # "logical"
```

\_\_\_\_\_

Variable / 'hello'

```
if Variable < "hello"
```

```
negate(Variable)
```

\_\_\_\_\_

1. *Journal of the American Medical Association*, 1997; 277: 1001-1005.

## Missing Values

- Missing values are denoted by NA (Not Available) or NaN (Not a Number).

```
x <- c(1, 3, NA, 4, 5)
class(x) # "numeric"
```

```
y <- c(1, 3, NaN, 4, 5)
class(y) # "numeric"
```

```
is.na() is used to test objects if they are NA
is.nan() is used to test for NaN
```

```
is.na(x) # FALSE FALSE TRUE FALSE FALSE
is.nan(x) # FALSE FALSE FALSE FALSE FALSE
```

Any questions?

Question 1: How can you indicate that something is a comment, and not a command that should be run?

Question 2: What is the difference between NA and NaN in R?

Question 3: Use the variable/object name 'distanceSchoolKM' and assign it numeric value two. Note, this is two kilometers. How can you convert this to meters?

Make a new variable/object called 'distanceSchoolMeters' with distance in meters. Note,  $1\text{km} = 1000\text{m}$ .



Question 4: What is the final value of 'areaHectares' as per below code?

```
areaHectares <- 2.5
2.47 * areaHectares
```

Question 5: Where are the objects created in RStudio kept and displayed?

1. in the Viewer
2. in the Files
3. in the Environment pane
4. in the console

1. logical variables
2. character variables
3. numerical variables
4. complex numbers

\_\_\_\_\_

- Copy and paste the entire **exact** error message into Google.
  - Someone else may have gotten this same error and has asked a question.
- Copy and paste the entire error message into Google, followed by 'r'.
- Google the name of the function with term 'tutorial r' to see tutorials.
- If struggling with code for a plot, Google 'r plot plotname', then click on Images.
- If errors with reading files, ensure path is correct. Check using `getwd()`.

\_\_\_\_\_

# Data Structures - Vectors

\_\_\_\_\_

\_\_\_\_\_

```
vector using paste() function
y <- paste("A", 1:5, sep = "")
is.vector(y)
```

```
vector using rep() function
y <- rep(letters[1:5], 3)
is.vector(y)
```



# Vectors

- There are multiple ways to create a vector:

```
?vector # vector using vector() function
vector(mode, length)
```

- The atomic modes are "logical", "integer", "numeric" (synonym "double"), "complex", "character" and "raw":

```
Initialize vector of certain length
x <- vector(mode = "numeric", length = 5)
x # 0 0 0 0 0
```

```
Initialize vector of certain length
x <- vector(mode = "character", length = 10)
x # "" "" "" "" "" "" "" "" "" ""
```

\_\_\_\_\_

# Vectors

- Sometimes objects can be coerced from one class to another using the `as.*` functions:

```
x <- 1L:10L
x # 1 2 3 4 5 6 7 8 9 10
class(x) # "integer"

z <- as.character(x)
z # "1" "2" "3" "4" "5" "6" "7" "8" "9" "10"
class(z) # "character"
```

\_\_\_\_\_

```
w <- c("a", "b", "c")
w
class(w)
q <- as.numeric(w)
q # NA NA NA
```

Question 7:  
Vector index in R starts from .....

\_\_\_\_\_

```
x <- 20:30 # vector
x # 20 21 22 23 24 25 26 27 28 29 30
length(x) # 11
x[1] # 20
x[15] # NA
```

```
x[c(1, 2, 4)] # 20 21 23
```

$$x[c(-2, -4)]$$

Can you mix positive and negative integers when accessing elements of a vector?

```
x <- 20:30 # vector
x # 20 21 22 23 24 25 26 27 28 29 30

x[c(2, -4)] # ?
```



## Content of Vectors

- There are several ways to modify vectors:

```
x <- 20:30 # vector
x # 20 21 22 23 24 25 26 27 28 29 30
```

```
x[1] <- 10
x # 10 21 22 23 24 25 26 27 28 29 30
```

```
x[1:3] <- 10
x # 10 10 10 23 24 25 26 27 28 29 30
```

```
x[x < 25] <- 5
x # 5 5 5 5 5 25 26 27 28 29 30
```

Any questions?

Create a numeric vector called 'testVector' containing numbers 1:10. Replace every second element of the vector with 0.

```
testVector <- c(1:10)
```

## Matrices & Lists

\_\_\_\_\_

|   |   |   |
|---|---|---|
| 1 | 3 | 5 |
| 2 | 4 |   |

\_\_\_\_\_

\_\_\_\_\_

```
a <- 1:4
```

```
b <- 5:8
```

```
c <- cbind(a, b)
```

C

```
dim(c) # 4 2
```

```
d <- rbind(a, b)
```

d

dim(d) # 2 4

How would you bind the two matrices below?

```
matOne <- matrix(data = 1:6, nrow = 2, ncol = 3)
matTwo <- matrix(data = 7:12, nrow = 2, ncol = 3)
```



Can you bind the following two matrices? Explain.

```
matCharacter <- matrix(data =
c("a", "b", "c", "d", "e", "f"),
nrow = 2, ncol = 3)
```

In the below matrix, alter position of first row, second column to numeric value of 10?

```
matNumeric <- matrix(data = 1:6, nrow = 2, ncol = 3)
```

|   |   |   |
|---|---|---|
| 1 | 3 | 5 |
| 2 | 4 | 6 |

### Question 13:

How would you generate the following matrix in R?

Hint: see help documentation for matrix using `?matrix`.

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |

\_\_\_\_\_

```
listOne <- list(16, "abc", TRUE, 5 + 4i)
listOne
length(listOne) # 4
typeof(listOne) # "list"
class(listOne) # "list"

access the contents of the list
listOne[[1]] # 16
listOne[[2]] # "abc"
listOne[[4]] # "5+4i"
```

# Lists

- Empty lists can be created using `vector()` function:

```
listTwo <- vector(mode = "list", length = 5)
listTwo
length(listTwo) # 5
```

- Lists can have names:

```
listDestinations1 <- list(1, 2, 3)
listDestinations1
names(listDestinations1) # NULL
names(listDestinations1) <- c("Canada", "Alaska",
"England")
listDestinations1
```

Any questions?