

# R Notes

Mehmet Tevfik DORAK

<http://www.dorak.info/r>

These notes are aimed at absolute beginners of learners of R program who want to "use R" rather than write codes for R (those are developers and they know where to go). If you just want to use the existing capabilities of R programme and currently (as of 2019) more than 16,000 packages associated with it, go ahead. If you want to be a developer to add to the existing capabilities, this site is not for you (but only after a while; if you are an absolute beginner, you may still benefit from the basics in these notes). Please go through these notes and associated presentations at <http://www.dorak.info/r> at your own pace. Don't skip the links provided to see what options are available to develop your R skills. Make sure you run the R scripts provided (as prompted) and start writing your own right away. Good luck!

The **R Project for Statistical Computing** (or just **R** for short) is a powerful data analysis tool. It is both a programming language and a computational and graphical environment.

## Selected links relevant to R and learning R

- R Packages: <https://rdrr.io> (CRAN - R Packages by Name; [CRAN Task View - CTVs](#))
- Quick-R: <http://www.statmethods.net> (BOOK: [R in Action](#))
- Instant R: <http://www.instantr.com> (BOOKs: [InstantR \(PDF\)](#) > [Using R for Statistics](#); [GitHub](#))
- Cookbook for R: <http://www.cookbook-r.com> (BOOK: [R Graphics Cookbook](#))
- R Cookbook: <http://shop.oreilly.com/product/9780596809164.do> (PDF)
- Impatient R (Some Hints for the R Beginner): <https://www.burns-stat.com/documents/tutorials/impatient-r>
- Learn R (Data Science Made Simple): <http://www.datasciencemadesimple.com/learn-r-what-is-r>
- How To In R: <http://howtoinr.weebly.com>
- Statistical Tools for High-throughput Data Analysis (STHDA) – Basic R Training: <http://www.sthda.com/english>
- Kickstarting R: <https://cran.r-project.org/doc/contrib/Lemon-kickstart>
- A Tutorial Introduction to R: [https://kingaa.github.io/R\\_Tutorial](https://kingaa.github.io/R_Tutorial)
- R Tutorials on R-bloggers: <http://rtutorialseries.blogspot.co.uk/search/label/tutorial>
  - Introduction Tutorials:
    - Part 1: <http://www.r-bloggers.com/r-tutorial-series-introduction-to-the-r-project-for-statistical-computing-part-1>
    - Part 2: <http://rtutorialseries.blogspot.co.uk/2009/10/r-tutorial-series-introduction-to-r.html>
- R Tutorials @ EndMemo: <http://www.endmemo.com/program/R>
- R Tutorials @ ListenData: <https://www.listendata.com/p/r-programming-tutorials.html>
- R Tutorials @ TutorialPoint: <https://www.tutorialspoint.com/r> (Quick Guide)
- R Tutorials @ DataMentor: <https://www.datamentor.io/r-programming>
- R Tutorials @ R-Chart: <http://www.r-chart.com> (emphasis on [graphics](#))
- R Tutorials @ UC Business Analytics R Programming Guide: <http://uc-r.github.io>
- R Tutorials by Kelly Black (with case studies): <https://www.cyclismo.org/tutorial/R/index.html>
- R Tutorials @ DataScienceMadeSimple: <http://www.datasciencemadesimple.com/learn-r-what-is-r>
- R Tutorials @ R Tutorial: <http://www.r-tutor.com> (Statistics: <http://www.r-tutor.com/elementary-statistics>)
- R Tutorials @ R-Statistics: <http://r-statistics.co/R-Tutorial.html>
- R Tutorials @ finnstats: <https://finnstats.com/index.php/category/data-analysis-in-r>
- R Tutorials @ R Complete Guide: <https://datasciencetut.com/category/r>
- R flow (R Workflow): <http://hbiostat.org/rflow>
- faster: Fast lane to learning R (online tutorials): <https://github.com/matloff/fasterR>
- YaRrr! The Pirate's Guide to R: <https://bookdown.org/ndphillips/YaRrr>
- Introduction to R by [StatEducation](#): <http://statseducation.com/Introduction-to-R>
- A Sufficient Introduction to R (eBook): <https://dereksonderregger.github.io/570L>
- UCLA Statistics/R pages: <http://statistics.ats.ucla.edu/stat/r>

- Learning Statistics with R: A tutorial for psychology students and other beginners @ <https://learningstatisticswithr.com/book>
- Practical R courses at [Data Novia](#)
- R Blog (Open Source Automation): <http://theautomatic.net/category/r>
- Cheat Sheets: [R-Graph Gallery](#); [RStudio](#); [DataCamp](#) (see also: [R Reference Card](#) & [v.2](#))
- R in Action (live online book): <https://livebook.manning.com/#!/book/r-in-action-second-edition/about-this-book>
- Practical Data Science with R (live online book): <https://livebook.manning.com/#!/book/practical-data-science-with-r>
- R for Data Science (live online book): <https://r4ds.had.co.nz> (tidyverse-oriented)
- Learn R (Online R Books): <http://ucanalytics.com/blogs/learn-r-12-books-and-online-resources>
- R Programming Wiki Book: [https://en.wikibooks.org/wiki/R\\_Programming](https://en.wikibooks.org/wiki/R_Programming)
- Learning R- The Ultimate Introduction (incl. Machine Learning): <http://blog.ephorie.de/learning-r-the-ultimate-introduction-incl-machine-learning>
- Data Science Live Book: <https://livebook.datascienceheroes.com>
- CRAN Task Views: <https://cran.r-project.org/web/views>
- R-specific internet search engines: <https://rseek.org> / [http://www.dangoldstein.com/search\\_r.html](http://www.dangoldstein.com/search_r.html)  
**SEE ALSO** the list of free, open-access R books at the end of this document.

## R syntax and related tips

- R is a case-sensitive language: `str()` is different `Str()`
- Variable names for (names for R objects) cannot start with a number
- Spacing is variable, one-space or two-spaces will not matter. Even splitting a line and having an indentation in the second line is fine.
- Single quote (') or double quote (") can be used for the same purpose as long as they are used consistently
- Beware of MS WORD replacing straight quotes (" ... ") with smart quotes (" ... "). Turn this feature off at AutoCorrect options, and never use MS WORD as a text editor
- String variables must be written within quotes, numeric variables are without quotes
- Dollar sign (\$) signifies the column name within a dataset/dataframe (like `iris$Species`)
- The up and down arrow keys can be used to recall and scroll through past commands, which can save typing when fixing typos or modifying a command
- Entering a letter and then hitting the Tab key twice will list the commands and objects starting with that letter
- Material can be cut and pasted into or from the R window. This allows you to save and print R results as part of MS Word documents, or save the text of your R session as a record of your work. R text is generally formatted as Courier font, and using Courier 9 point font works well for R output.
- Every function ends with a pair of brackets (even if they are empty as in `getwd()` or `quit()` )
- Each function may have multiple arguments/options/parameters separated by commas (and usually with default values) within brackets (like: `plot(x, col = "red")` )
- A function may be typed (in a script) with lots of space, including hard return, between its elements and even comments (beginning with #) inserted between elements. For example, the following command:
 

```
boxplot(iris[1:4],
        boxwex = 0.2,
        pch = 14)
# you can even have a comment at the end of first or second line above
```

 is equivalent to:
 

```
boxplot(iris[1:4], boxwex = 0.2, pch = 14)
```
- Multiple functions can be written in a single line as long as separated by semi-colons like: `d <- density(x); plot(d)`
- Scripts can be saved as text files with file extension ( .R ) and run from R after highlighting the script and pressing CTRL + R (or right click and "Run line or selection")
- R function names with more than one word have a dot between the words (like `install.packages()` or `shapiro.test(x)` )
- Double column ( :: ) separates a function name from its package name (like `fBasics::basicStats`). This is important if the same function name is used in multiple packages. If you see double colons like

in the following: `DescTools::Cstat()`, it means the `Cstat()` function is from the package named `DescTools`

- **CTRL + L:** Clears the console (without unloading any variable, library, object etc). There is no "clear screen" function in R for Windows, but one can try the following: `cat(rep("\n", 50))` or `clc <- function() cat(rep("\n", 50)); clc()`
- When generating a set of numbers, setting a seed number generator with `set.seed()` is not necessary, but it will make your results reproducible
- R uses forward slashes in directory addresses. It is not "C:\R", but "C:/R"; it is not `H:\DATA` but `H:/DATA`
- If you want to print an object after creating it (and without asking for a print on a separate line), put the assignment line within brackets, like: `(x <- (3 * 4))`. This function will create the "x" object and will print its value.
- In a user-defined function, elements between braces (`{...}`) are the body of the function (equivalent to indented bits in a Python function)
- If you see a line starting with a plus sign (+) after trying to execute a function, it means something is missing at the end of the line (usually a closing bracket)
- If the graphics window does not appear after asking for a plot and with no error messages, it may be hiding behind the maximised R console
- In R, the coordinates are 1-based (NOT 0-based); i.e., the first nucleotide on a chromosome is numbered 1, rather than 0
- Each R function returns a "value" which can be saved as an R object and recalled for downstream applications (like the survival analysis result being used for forest plot development)
- Most R packages can be downloaded from CRAN repository, but some need to be downloaded from GitHub using "devtools" or "remotes" packages
- Escape sequences in R: <http://127.0.0.1:20926/library/base/html/Quotes.html> (or type `?Quotes` in R). The most commonly used ones are `\n` for new line (linefeed) and `\r` for carriage return. For example in a ggplot graphic the if you want the title to appear in two lines: `ggtitle("This is the Title \n This is the Subtitle")`
- SEE ALSO: [Good Practices in R Programming](#); [Common Uncommon Notations that Confuse New R Coders](#); [Avoiding Pitfalls in R](#).

### Most common sources of error messages

- Typos (comma instead of dot; pound sign or ampersand instead of dollar sign; leaving out the dot in multi-word function names like `read.delimit()` or `as.factor()`); case sensitivity (including WORD changing the initial to a capital letter if you have copied from WORD)
- Parentheses (type, unequal opening and closing parentheses)
- Quotes; including WORD changing your straight quotes (" ... ") with smart quotes (" ... "); inconsistent use of single and double quotes
- Missing commas (e.g., between function arguments), or missing semi-colons between functions on the same line
- Your grouping variable is NOT a factor (but numerical or string/character variable; check with `class()`)
- Missing data is causing trouble
- The function you use exists in two different libraries in use (loaded)
- The library you intend to use is not loaded
- In path definition, R uses forward slash like `C:/R` as in web addresses (not back slash as in Windows Explorer like `C:\R`)

### The assignment operator `<-`

You will make lots of assignments using the operator `<-` (aRrow). Do not use `=` for the same purpose. Even if it works, it may cause some confusion (especially for developers).

Notice that RStudio automatically surrounds `<-` with spaces, which demonstrates a useful code formatting practice. It is good practice to use spaces on both sides of all operators.

RStudio offers many handy [keyboard shortcuts](#). Also, Alt (+) Shift (+) K brings up a keyboard shortcut reference card (in RStudio). In RStudio, the keyboard shortcut for the assignment operator `<-` is Alt (+) - (Windows).

If you get stuck with some syntax (usually, mismatched parentheses or quotes), the R Console will change from the `>` at the beginning of the line (which means it is waiting for a new command) to the `+` at the beginning of the line (which means it is waiting for you to finish a command). To get out, hit the **Escape** key.

## R language

The screen prompt, `>`, is the R prompt that waits for commands. The output following the entry of a command is prefixed by `[1]`, which indicates that this is item 1 in a vector of output, and makes the output results easier to be identified. Braces `{ }` are used to group together one or more expressions (like indented blocks in Python). R is a strictly case-sensitive language.

## Running R online

If you cannot install R, you can get at least some of the functionality from different providers of online R:

- <https://posit.cloud> (posit/RStudio Cloud)
- <https://rdr.io/snippets> (with 19K R packages pre-loaded)
- <https://webr.r-wasm.org/latest> (WebR)
- [https://www.tutorialspoint.com/execute\\_r\\_online.php](https://www.tutorialspoint.com/execute_r_online.php)
- [http://www.compileonline.com/execute\\_r\\_online.php](http://www.compileonline.com/execute_r_online.php)
- <https://app.displayr.com> (for making graphs online from data frames; NOT free; 14-day free trial available)

## Start-up screen

On the start-up screen, a function or a command can be typed, or R can be used to perform basic calculations. R uses the usual symbols for addition (+), subtraction (-), multiplication (\*), division (/), and exponentiation (^). R uses %% for taking the modulus and %/% for integer division. Comments in R begin with the character # (everything after # up to the end of the line will be ignored by R). If you do not want the lines beginning with # to be interpreted as comment, you can disable this feature by specifying `comment.char = ""`. To clear the standard information from the screen, use CTRL + L.

## What to do at the start of a session

- You may want to start a new session on R by the following command:  
`rm(list=ls())`, which removes (rm) all variables already in the memory.
- If you are going to download lots of packages, start with:  
`chooseCRANmirror()` OR select "Set CRAN Mirror..." in the Packages tab in the menu, and select the closest mirror to you for all downloads in the current session.
- Get your working directory (default folder):  
`getwd()` # which returns something like: "H:/DATA or C:/R".  
Note that R uses forward slashes (as in web addresses but not in Windows locations)
- If you need to keep things simple, change your working directory to something simple:  
`setwd("C:/R")` which will change the working directory only for the current session.
- Check the version of R if you like:  
`R.Version()`  
# OR: Help tab and About
- If you want to make sure that all of your installed packages are up-to-date, run: `update.packages()` and select a CRAN mirror to get all missing updates.

## If you must run R on a mobile device

There are applications for running R on your iOS or Android device with limited functionality. You can explore that, for instance, when waiting for the bus. If you must use a mobile device, try R online, for example: [DataCamp Light \(GitHub\)](#).

To run small snippets of R code online, try: <https://rdr.io/snippets>

## Demos and help

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.

As a first step with R, start the R help browser by typing `help.start()` in the R command window. For help on any function, e.g. the “mean” function, type `?mean`. Try also the Help tab in the menu on top of R console.

## Help manuals

A series of manuals are available in the **Help** section of the R menu (Help > Manuals (in [PDF](#))). For example, the manual [An Introduction to R](#) contains an introduction to the R language and environment (and includes a Sample R Session in [Appendix A](#)). More advanced users may like the [Writing R Extensions](#) manual, which focusses on creation of R packages. See also [R-FAQs](#).

## The working directory ([R-bloggers](#))

One of the initial things that you want to do when you launch R is to set its working directory. This is the default location on your hard disk that R will look to read and write files. The working directory is comparable to what is called the "default folder" in many other applications. It is important to select a location that is easy to find and remember, so you can access your files when you need them.

To display the current working directory, use the function `getwd()`.

```
> getwd()
[1] "/Users/Admin/Documents/R"
```

To display the contents of the default folder:

```
list.files(path=".")
```

To change the current working directory, use the function `setwd('PATH')`, replacing PATH with the directory path that you would like to use.

```
> setwd('/Users/Admin/Documents/R/newProject')
```

Use `getwd()` again to verify that the change took place.

```
> getwd()
[1] "/Users/Admin/Documents/R/newProject"
```

This change is only for the current R session.

Note that you have the option to set the working directory at any time. Do this when you want to access files in a new location, such as when you are working on multiple projects at the same time or at the start of a new project. (See also: [http://stat545-ubc.github.io/block002\\_hello-r-workspace-wd-project.html](http://stat545-ubc.github.io/block002_hello-r-workspace-wd-project.html).)

While reading a file into R, if you are not sure about the location, you can use the `file.choose()` function to browse the computer. For example, to read a CSV file (with a header row):

```
> read.csv(file.choose(), header = TRUE)
```

## Packages ([Quick-R](#))

**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**. **Base R** comes with a standard set of packages (like the graphics package `lattice`). Others are available for download and installation. Once installed, they have to be loaded into the session to be used:



## Package/library management

```
> chooseCRANmirror()          # To choose the CRAN mirror (or via Packages tab in the menu)
> install.packages("packagename") # to install a specific package
> install.packages(c("packagename1", "packagename2", "packagename3"))
    # to install multiple packages
    # If you add dependencies = TRUE as an option to
    install.packages() function, you will also get all the dependencies
    (additional libraries needed) installed if you do not already have them
> require("packagename")     # checks if the required package installed, and installs if not
> installed.packages()       # details of all packages installed in the specified libraries
> library("packagename")     # to load a specific package in the workspace
    # no need for quotes, but it is good practice
> search()                   # see packages currently loaded
> library()                  # list of currently installed packages
> (.packages())              # list of currently loaded packages; you need to have brackets as shown
> detach(package:lattice, unload = TRUE) # to unload a loaded package
> remove.packages(pkgs, lib) # removes an installed specific package (see link)
> .libPaths()                # get library location (there is a dot in the beginning)
```

Alternatively, for a group of related packages installation (FROM: [http://phytools.org/eqg/Exercise\\_3.2](http://phytools.org/eqg/Exercise_3.2)): Use the package "ctv" (i.e., CRAN Task View) to automatically install & update all the packages for R "phylogenetic analysis" that are available and listed in the Task View.

```
> install.packages('ctv')          # install CRAN Task View
> library('ctv') install.views('Phylogenetics') # install all phylogenetics-related packages
> update.views('Phylogenetics')
```

Before you run `install.packages`, go to your R packages directory (`C:/Users/mtd/R/win-library/3.5`) and delete any zero-length files that may be present. These are artefacts of failed install attempts and will break `install.packages()` if present.

## How to Install Packages ([Quick-R](#)) via the menu

1. Choose **Install Packages** from the **Packages** menu
2. Select a **CRAN Mirror** (e.g. Norway)
3. Select a package (e.g. `fBasics`)
4. Then use the `library("package")` function to load it for use (e.g. `library("ggplot2")`; quotes are optional)

Alternatively, a package can be installed by typing the following command:

```
> install.packages("packagename")
```

SEE a complete [list of contributed packages](#) available from **CRAN**.

(the installation via the CRAN repository should be preferred over manual installation. This way, the dependency of the package will also be downloaded automatically, which otherwise would require manual installation.)

<http://www.cran.r-project.org/web/views> Packages organized by task

R-Forge Software Map: [https://r-forge.r-project.org/softwaremap/trove\\_list.php](https://r-forge.r-project.org/softwaremap/trove_list.php)

<http://www.rseek.org> Function finder

## R objects

Everything that exists in R like variables, vectors, matrices/arrays, lists and data frames are R objects (matrices contain the same type of data; data frames may contain different types of data). In R, an object is anything that can be assigned to a variable. This includes constants, data structures, functions, and even graphs. An object has a mode (which describes how the object is stored) and a class (which tells generic functions like `print` how to handle it). To name any R object, use the assignment operator (`<-`). See this [note](#) about "objects" in R for details. R objects can be saved with `save()` function in RData format.

## R commands/functions

Most of the R commands (or functions) end with a couple of parentheses. The parentheses indicate that you are using a function, but not an object. Within the parentheses, the arguments (or options) of the function are specified (for example, the function `boxplot()` has arguments like `col = "red"`, `pch = 21`, etc). If the function doesn't need any argument or default values are fine, no need to include any argument within the parentheses (it can be left blank as in `quit()`, `windows()`, `ls()`).

## User-defined functions

<https://www.statmethods.net/management/userfunctions.html>

<https://www.datamentor.io/r-programming/function>

## Reading files into the R environment

```
read.table()
read.csv()      # For arguments of read.csv(),
                SEE: https://swcarpentry.github.io/r-novice-inflammation/11-supp-read-write-csv
read.csv2()     # In some European countries the delimiter in a .csv is a semicolon ";" and not a comma.
                If you are using such files use read.csv2() instead of read.csv().
```

If you are not sure what your working directory is, and you want to read a CSV file into R, use the following command:

```
x <- read.csv(file.choose(), header=TRUE)      # if no header, use FALSE
```

This will allow you to browse your computer and locate the file like you do on Windows Explorer.

Alternatively, you can copy your spreadsheet on the clipboard, and import (paste) into R using the following command:

```
x <- read.delim("clipboard", header=TRUE)      # if no header, use FALSE
```

`read.csv()` automatically interprets nonnumeric data as a factor (categorical variable). if you want your non numerical variable interpreted as strings, not as factors, set the `as.is` parameter to TRUE (`as.is = TRUE`). This indicates that R should not interpret nonnumeric data as a factor.

## Reading CSV files into the R environment in countries (including Turkey) using comma as the decimal separator ( `read.csv2()` )

In some European and most Asian countries, a comma (8,52) is used as the decimal separator (instead of a period/dot like 8.52). Because of that, Excel in those countries actually uses a semicolon (;) instead of a comma (,) as the delimiter (between the Excel columns) in csv files.

If the standard `read.csv()` function is used to read those csv files, each row appears in a single column with cell contents separated by semicolons.

`read.csv2()` solves this problem and reads semicolons as the separator and reads those types of csv files correctly.

`read.csv2()` is, therefore, equivalent to `read.csv("filename.csv", sep = ";")` (See:

<https://stackoverflow.com/questions/35360847/r-import-csv-file-all-data-fall-into-one-the-first-column>;

<https://hotware.wordpress.com/2009/12/16/trouble-with-opening-csv-files-with-excel-the-comma-and-semicolon-issue-in-excel-due-to-regional-settings-for-europe>)

For further information (similar problem in a different setting well explained):

[https://www.ibm.com/support/knowledgecenter/en/SSWU4L/Data/imc\\_Data/Data\\_q\\_a\\_watson\\_assistant/How\\_To\\_Remove\\_a\\_Semicolon\\_in\\_a\\_CSV\\_File197.html](https://www.ibm.com/support/knowledgecenter/en/SSWU4L/Data/imc_Data/Data_q_a_watson_assistant/How_To_Remove_a_Semicolon_in_a_CSV_File197.html)

See [csv2.read.txt](#) for details.

## Reading a PDF file:

<https://datascienceplus.com/extracting-tables-from-pdfs-in-r-using-the-tabulizer-package>

<https://stackoverflow.com/questions/44141160/recognize-pdf-table-using-r>

## Reading from a ZIP file on the computer:

```
data <- read.table(unz("zipfile.zip", "zipfile.dat"), nrows=10, header=T,
quote="\\"", sep=",")
```

```
myfile <- read.csv(zip.file.extract("~/files/test.csv", "myzip.zip")) # Where the file
test.csv is actually located in the: ~/files/myzip.zip/test.csv
```

**Read zipped file into R:** <https://www.r-bloggers.com/read-zipped-file-into-r> OR  
<https://kariert.wordpress.com/2011/05/15/read-zipped-file-into-r>

### Reading from a ZIP file on the internet:

<https://stackoverflow.com/questions/3053833/using-r-to-download-zipped-data-file-extract-and-import-data>:

Zip archives are actually more a 'filesystem' with content metadata etc. See `help(unzip)` for details. To do what you sketch out above you need to:

Create a temp. file name (eg `tempfile()`)

Use `download.file()` to fetch the file into the temp file

Use `unz()` to extract the target file from temp. file

Remove the temp file via `unlink()`

which in code looks like:

```
temp <- tempfile()
download.file("https://www.zip.com/zippedfile.zip", temp)
data <- read.table(unz(temp, "filename.txt"))
unlink(temp)
```

Compressed (.z) or gzipped (.gz) or bzip2ed (.bz2) files can also be downloaded, unzipped and read by R (see this [link](#)).

# Reading a table from the internet:

CSV file:

```
myfile <- read.csv("http://www.dorak.info/r/ensembl-export.csv", header=TRUE)
```

TSV file:

```
assets <-
read.table("https://www.portfolioprobe.com/R/blog/xassetCountrySector.txt",
sep="\t", header=TRUE)
```

**For more on data import, see:**

- **Data Import Cheat Sheet:** <https://www.rstudio.com/wp-content/uploads/2018/08/data-import-600x464.png>
- **DataCamp Tutorial:** <https://www.datacamp.com/community/tutorials/r-data-import-tutorial>
- **HoningDS Tutorial:** <https://honingds.com/blog/import-data-into-r/>
- **Data Manipulation in R- Online Course (Data Novia):** <https://www.datanovia.com/en/coursecategory/data-manipulation>

### Graphing in R:

- R Tips (Graphs/Tables): <https://www.zoology.ubc.ca/~schluter/R/display>
- Producing Simple Graphs with R: <https://www.harding.edu/fmccown/r>
- Styling Base R Graphics: <https://www.jumpingrivers.com/blog/styling-base-r-graphics>
- Quick-R (Graphs): <https://www.statmethods.net/graphs/index.html>
- R Tutorials @ R-Chart: <http://www.r-chart.com> (emphasis on [graphics](#))
- R Graphics Cookbook\_BBC Visual and Data Journalism: <https://bbc.github.io/rcookbook>
- R Graph Gallery: <https://www.r-graph-gallery.com/the-r-graph-gallery> ([cheatsheets](#))
- R Graph Gallery (Art from Data): <https://www.r-graph-gallery.com/portfolio/data-art>
- R Graph Catalog (with scripts): <http://shinyapps.stat.ubc.ca/r-graph-catalog> (see [here](#))
- R Graphics for Clinical Trials @ Gersonides (with scripts): <http://www.gersonides.com/r>
- Data Visualization in R (complete course with scripts): <http://www.datavis.ca/courses/RGraphics> (lecture presentations: [Lecture 1](#); [Lecture 2](#); [Lecture 3](#); [Lecture 4](#))
- DataCamp project: [Phyllotaxis: Draw Flowers Using Mathematics](#) (free)



- DataCamp project: [Visualizing Inequalities in Life Expectancy](#) (paid)
- Graphics ([StatEducation](#)): <http://statseducation.com/Introduction-to-R/modules/graphics/ggplot2>
- Graphics ([Stats and R](#)): <https://www.statsandr.com/blog/graphics-in-r-with-ggplot2>
- Statistical charts: <https://www.r-graph-gallery.com/author/holtz>
- 10 tips for making your R graphics look their best: <https://blog.revolutionanalytics.com/2009/01/10-tips-for-making-your-r-graphics-look-their-best.html>
- Playing Around with Phyllotactic Spirals: <https://chichacha.netlify.com/2019/01/29/playing-around-with-phyllotactic-spirals>
- R for Beginners: Some simple code to produce informative graphs, part I: <https://dmwiig.net/2017/01/11/r-for-beginners-some-simple-code-to-produce-informative-graphs-part-one>
- R for Beginners: Basic graphics code to produce informative graphs, part II (working with big data): <https://dmwiig.net/2017/01/16/r-for-beginners-basic-graphics-code-to-produce-informative-graphs-part-two-working-with-big-data>
- FlowingData Visualisation by R Tutorials: <http://flowingdata.com/category/tutorials>
- FlowingData Visualisation by R Courses: <https://flowingdata.com/courses>
- Get Started. Make Your Own Plot and More: [Learn how to plot a graph in R](#)
- Graphical Parameters (Quick-R): <https://www.statmethods.net/advgraphs/parameters.html> ; <https://stat.ethz.ch/R-manual/R-devel/library/graphics/html/par.html>

R can produce plots in a standard format and allows controlling of the plot appearance. Three main packages for data representation are:

- **graphics**: This is already available in the base R and is loaded by default (SEE: [Styling Base R Graphics](#))
- **lattice**: This is already available with the basic installation of R, but needs to be loaded using the `library()` function.
- **ggplot2**: This is not included in the basic installation of R, but can be installed using the function: `install.packages("ggplot2")`.
- **ggplot2** Reference (functions): <https://ggplot2.tidyverse.org/reference>
- **ggplot2** Cheat Sheet: <http://zevross.com/blog/2014/08/04/beautiful-plotting-in-r-a-ggplot2-cheatsheet-3>
- R-Statistics: ggplot short tutorial: <http://r-statistics.co/ggplot2-Tutorial-With-R.html>
- R-Statistics- ggplot2 quickref: <http://r-statistics.co/ggplot2-cheatsheet.html>
- STHDA: **ggplot2** essentials: <http://www.sthda.com/english/wiki/ggplot2-essentials>
- STHDA: Histograms with **ggplot2** (tutorial): <http://www.sthda.com/english/wiki/ggplot2-histogram-easy-histogram-graph-with-ggplot2-r-package>
- STHDA: Dot plots with **ggplot2** (tutorial): <http://www.sthda.com/english/wiki/ggplot2-dot-plot-quick-start-guide-r-software-and-data-visualization>
- STHDA: **ggplot2** scatter plots: Quick start guide - R software and data visualization: <http://www.sthda.com/english/wiki/print.php?id=188>
- sape **ggplot2** Quick Reference: <http://sape.inf.usi.ch/quick-reference/ggplot2>
- Graphics tutorials by Michael Toth (<https://michaeltOTH.me>):
  - Detailed Guide to the Bar Chart in R with **ggplot**: <https://michaeltOTH.me/detailed-guide-to-the-bar-chart-in-r-with-ggplot.html>
  - A Detailed Guide to Plotting Line Graphs in R using **ggplot** `geom_line`: [https://michaeltOTH.me/a-detailed-guide-to-plotting-line-graphs-in-r-using-ggplot-geom\\_line.html](https://michaeltOTH.me/a-detailed-guide-to-plotting-line-graphs-in-r-using-ggplot-geom_line.html)
  - A Detailed Guide to the **ggplot** Scatter Plot in R: <https://michaeltOTH.me/a-detailed-guide-to-the-ggplot-scatter-plot-in-r.html>
  - How to Create a Bar Chart Race in R (Animated Bar Chart) - Mapping United States City Population 1790-2010: <https://michaeltOTH.me/how-to-create-a-bar-chart-race-in-r-mapping-united-states-city-population-1790-2010.html>
  - A Detailed Guide to **ggplot** colors: <https://michaeltOTH.me/a-detailed-guide-to-ggplot-colors.html>
  - One Step to Quickly Improve the Readability and Visual Appeal of **ggplot** Graphs: <https://michaeltOTH.me/one-step-to-quickly-improve-the-readability-and-visual-appeal-of-ggplot-graphs.html>
- How to combine Multiple **ggplot** Plots to make Publication-ready Plots: <https://datascienceplus.com/how-to-combine-multiple-ggplot-plots-to-make-publication-ready-plots>

- Data Visualization with R (live book by Rob Kabacoff): <https://rkabacoff.github.io/datavis>
- R-Statistics: Top 50 **ggplot2** Visualizations - The Master List (with Full R Code): <http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html>
- Examples of ggplot()
  - <https://www.neonscience.org/dc-time-series-plot-ggplot-r>
  - <http://t-redactyl.io/blog/2016/02/creating-plots-in-r-using-ggplot2-part-6-weighted-scatterplots.html> (bubble plot)
  - <https://stackoverflow.com/questions/22598517/how-to-have-multiple-labels-in-ggplot2-for-bubble-plot> (multiple labels)
  - <https://towardsdatascience.com/hands-on-guidance-of-data-visualization-in-r-package-ggplot2-of-nba-players-stats-d812ed272d66>

**R Color Chart:** R has 657 built-in color names. The function `colors()` will show all of them. All these color names can be used in plot parameters like `col = "..."`. The function `col2rgb()` can convert all these colors into RGB numbers (try, for example `col2rgb("orchid1")`).

**PCH Symbols Chart** lists PCH symbols used in R plot. When the PCH is 21-25, the parameter `col = "..."` and `bg = "..."` should be specified. PCH can also be in characters, such as "#", "%", "A", "a", and the character will be plotted.

See also [R Graph Gallery cheat sheets](#) for charts related to graphs.

HSV (HSV Color Specification): <https://stat.ethz.ch/R-manual/R-devel/library/grDevices/html/hsv.html>

R Color Styles: [http://bioinfo.umassmed.edu/bootstrappers/bootstrappers-courses/pastCourses/rCourse\\_2016-04/Additional\\_Resources/Rcolorstyle.html](http://bioinfo.umassmed.edu/bootstrappers/bootstrappers-courses/pastCourses/rCourse_2016-04/Additional_Resources/Rcolorstyle.html)

#### *Command options for base R plots*

Many (but not all -- try them) of the basic plotting commands in base R will accept the same options to control axis limits, labeling, print a title, change the plotting symbol, change the size of the plotting symbols and text, and change the line types. Here are some of the most frequently modified options. Use them inside the parentheses of a plotting command to have their effect. If you are not sure whether a given option works in your case, try it. The worst that could happen is you get an error message, or R ignores you.

```
main = "Eureka"      # add a title above the graph
pch = 16             # set plot symbol to a filled circle
color = "red"        # set the item colour
xlim = c(-10,10)    # set limits of the x-axis (horizontal axis)
ylim = c(0,100)     # set limits of the y-axis (vertical axis)
lty = 2              # set line type to dashed
las = 2              # rotate axis labels to be perpendicular to axis
cex = 1.5            # magnify the plotting symbols 1.5-fold
cex.lab = 1.5        # magnify the axis labels 1.5-fold
cex.axis = 1.3       # magnify the axis annotation 1.3-fold
xlab = "Body size"   # label for the x-axis
ylab = "Frequency"   # label for the y-axis
```

#### **Scripts:**

# To generate a scatterplot:

```
a <- rnorm(50) # rnorm(50) generates 50 random numbers in standard normal
               # distribution
b <- rnorm(50) # Another 50 random numbers into vector b
plot(a, b)     # This plot should have quite a scatter centered around the center
               # Try to generate a plot with rnorm(10000) to see a centralised
               # scatter
```

# To change the shape of the dot, TRY "pch" argument (pch: plotting character):

```

plot(rnorm(1000), pch=21) # TRY also pch=22, pch=23, pch=24 etc (between 0 and 25)
OR pch="*", pch="#"
# To read from a table and plot data in two columns
# File name: table.txt; column headers: Body_mass and Brain_mass
dataset <- read.table("table.txt") # table.txt is in the working directory
# reads the table as a dataframe called "dataset"
plot(dataset$Body_mass, dataset$Brain_mass)
# OR: first use the attach function: attach(dataset)
# then: plot(Body_mass, Brain_mass)

# To save a graph as an image file format such as PNG, we can use the png() command:
png("scatterplot.png")
plot(rnorm(1000))
dev.off()
# The preceding command will save the graph as scatterplot.png in the current working directory. Similarly, if we
wish to save the graph as JPEG, BMP or TIFF we can use the jpeg(), bmp(), or tiff() commands
respectively.
# SEE: Cookbook for R - Output to a File: http://www.cookbook-r.com/Graphs/Output\_to\_a\_file

# You can specify a number of arguments to adjust the graph as per your needs. The simplest one that we've
already used is the filename. You can also adjust the height and width settings of the graph:
png("scatterplot.png",
height=600,
width=600)
# The default units for height and width are pixels but you can also specify the units in inches, cm or mm:
png("scatterplot.png", height=4, width=4, units="in")
# The resolution of the saved image can be specified in dots per inch (dpi) using the res argument:
png("scatterplot.png", res=600)

# To set the plot background color to gray, use the bg argument in the par() command (not with ggplot):
par(bg="gray") # TRY: yellow, blue, pink, orchid2; OR: 2, 3, 4 (but not 1!)
plot(a,b) # if a and b are vectors of equal lengths, OR plot(rnorm(5000))
# SEE: http://127.0.0.1:19842/library/graphics/html/par.html about par function

# Setting colours for text elements: axis annotations, labels, plot titles, and legends:
# If you want to make the axis value annotations black, the labels of the axes gray, and the plot title dark blue, you
should do the following:
plot(rnorm(100),
main = "Plot Title",
col.axis = "blue",
col.lab = "red",
col.main = "darkblue")

# Full command:
a <- rnorm(50)
b <- rnorm(50)
plot(a, b, pch = 16, col = "orchid1", xlim = c(-3, 3), ylim = c(-3, 3), main
= "MY PLOT", xlab = "X VARIABLE" , ylab = "Y VARIABLE" , col.axis="blue",
col.lab="red", col.main="darkblue")

# Boxplot:
boxplot(iris$Sepal.Width) # for the whole dataframe
boxplot(iris$Sepal.Width~iris$Species) # Three boxplots for each of the three species

boxplot(rnorm(1000), boxwex=0.2, las = 1) # TRY various values (<1) for boxwex, and 0
and 1 for las (labels are parallel (=0) or perpendicular(=1) to axis)

# Boxplot without outliers:
# TRY the following two boxplot commands:
boxplot(rnorm(100000))
windows()
boxplot(rnorm(100000), outline=FALSE)

```

# Histogram basics: <https://www.r-bloggers.com/basics-of-histograms/> <https://www.r-bloggers.com/basics-of-histograms/>

# Histogram by a grouping factor using lattice included in Base R:

```
data(infert)
library(lattice)
histogram(~ parity | case, data = infert) # parity is a count variable, and case is
                                         the case-control status variable
histogram(~ spontaneous | case, data = infert) # spontaneous is a count variable, and
                                              case is the case-control status variable
```

# Scatter plot by a grouping factor (in different colours):

```
library("ggplot2")
data(iris)
qplot(Petal.Width, Petal.Length, color = Species, data = iris)
```

# Multiple histograms in the same panel:

```
library("lattice")
data(infert)
histogram(~ parity + spontaneous + induced, data = infert)
```

# For a full list of graphical parameters, see (1):

<https://www.statmethods.net/advgraphs/parameters.html>

# For a full list of graphical parameters, see (2): <https://www.statmethods.net/advgraphs/axes.html>

# Pie chart using Base R:

```
v <- c(88, 76, 52, 26, 12)
pie(v)
pie(v, main = "Number of e-mails received", col = rainbow(length(v)),
    labels=c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday"))
```

# Barplot example (with labels and colours):

```
v <- c(88, 76, 52, 26, 12)
barplot(v)
barplot(v, main = "Number of e-mails received", xlab = "Days", ylab =
"Count", col = rainbow(length(v)), names.arg = c("Monday", "Tuesday",
"Wednesday", "Thursday", "Friday"), border = "blue")
```

# 3D piechart:

```
install.packages("plotrix") # installs the package (called "plotrix" in this case)
library(plotrix)           # calls the package (called "plotrix")
v <- c(2, 6, 4, 9, 10, 1, 12, 3, 5) # generates a vector called "v"
pie3D(v)                   # generates a 3D piechart of "v" (D = capital letter!)
```

# Adding a line and a legend to a graph:

```
# Generate some data
x <- 1:10; y1 = x*x; y2 = 2*y1
plot(x, y1, type = "b", pch = 19, col = "red", xlab = "x", ylab = "y")
# Add a line
lines(x, y2, pch = 18, col = "blue", type = "b", lty = 2)
# Add a legend
legend(1, 95, legend = c("Line 1", "Line 2"), col = c("red", "blue"),
lty = 1:2, cex = 0.8)
```

# Adding a line to the boxplot:

```
data(iris)
boxplot(iris[-5])
abline(h=2.5, col = "blue")
```

# Overlaying density line over a histogram:

```
data(iris)
hist(iris$Petal.Width, prob=TRUE) # prob=TRUE uses probability not counts
```

```

lines(density(iris$Petal.Width)) # overlays the density plot curve on histogram
# Options for lines(): Set line type (lty) to solid (lty=1), color to red (col="red"), and line width (lwd) to 4 (lwd=4).
# SEE:
https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781849513067/6/ch06lv1sec07/overlying-density-line-over-a-histogram

# Creating Heat Maps and Contour Plots (Chapter 8 of R Graph Cookbook_PACKTPUB.pdf; p.181)
# Finalizing graphs for publications and presentations (Chapter 10 of R Graph Cookbook_PACKTPUB.pdf; p.223)
# SEE ALSO: R Graph Essentials_PACKTPUB.pdf

```

Cookbook for R - Graphs: <http://www.cookbook-r.com/Graphs>

### R for Statistical Analysis:

- Begin with:
  - Instant R - An Introduction to R for Statistical Analysis:** <http://www.instantr.com> (& the [book](#): Using R for Statistics\_2014.pdf)
- Statistical Analysis with R- Beginner's Guide\_PACKT 2010 (and <http://rtutorialseries.blogspot.com>)
- Instant R Starter\_PACKT 2013.pdf
- Quick-R: Statistics (<https://www.statmethods.net/stats/index.html>) (& the book: R in Action\_Manning)
- Cookbook for R - Statistical analysis: [http://www.cookbook-r.com/Statistical\\_analysis](http://www.cookbook-r.com/Statistical_analysis)
- r4stats: statistics: <http://r4stats.com/examples/statistics>
- R Tutorials @ R Tutorial: <http://www.r-tutor.com> (Statistics: <http://www.r-tutor.com/elementary-statistics>)
- R Tutorials @ R-Statistics: <http://r-statistics.co/R-Tutorial.html>
- Stats and R: <https://www.statsandr.com>
- Using R for Introductory Statistics\_Verzani\_2005 ([link](#))
- Learning statistics with R (online book): <https://learningstatisticswithr.com/book> (PDF: <https://learningstatisticswithr.com/lsr-0.6.pdf>)
- Introduction to Data Science\_Jeffrey Stanton\_2013.PDF ([link](#))
- Learn to Use R\_v3\_Computer World.pdf
- Statistics with R- Computing and Graphics\_Kjell Konis.pdf (includes a practice run) ([link](#))
- simpleR – Using R for Introductory Statistics.PDF (including a sample R session / Teaching with R; [link](#))
- Introduction to R (StatEducation): <http://statseducation.com/Introduction-to-R>
- VIB Course on [Basic Statistics in R](#) with downloadable training material including [slides](#)
- Using R for statistical analyses (Gardener's Own): <http://www.gardenersown.co.uk/Education/Lectures/R/basics.htm>
- [A Handbook of Statistical Analyses Using R \(online book\) by Everitt & Hothorn.pdf](#)
- Matthias Kohl: Introduction to Statistical Data Analysis with R, Bookboon (2015): <https://bookboon.com/en/introduction-to-statistical-data-analysis-with-r-ebook>
- Analyzing Data in R- From A to Z : <http://www.deeplytrivial.com/p/the-to-z-of-r.html>
- Tutorials on Advanced Stats and Machine Learning with R: <http://r-statistics.co>

**EDx Course- Introduction to Applied Biostatistics: Statistics for Medical Research: Learn data analysis for medical research with practical hands-on examples using R Commander**  
<https://www.edx.org/course/introduction-applied-biostatistics-osakaux-med101x-0>

**Get Started: Make Your Own Plot and More!**  
[Check out these examples of basic statistics](#)

**R packages for undergraduate statistics education @ Citizen-Statistician:**  
<http://citizen-statistician.org/2015/08/09/r-packages-for-undergraduate-stat-ed>

**OpenIntro Statistics Book (for UG education with R examples):**  
<https://www.openintro.org/stat/textbook.php> (see [package "openintro"](#))



### Statistics with R (online book):

[http://zoonek2.free.fr/UNIX/48\\_R/all.html](http://zoonek2.free.fr/UNIX/48_R/all.html)

### Introduction to Statistical Thinking (With R, Without Calculus) by Yakir (2011):

<http://pluto.huji.ac.il/~msby/StatThink/index.html>

### An R Companion for the Handbook of Biological Statistics:

<http://rcompanion.org/rcompanion>

RCompanion Handbook (Statistics): <https://rcompanion.org/handbook/>

### Bitesize: Using R for Statistical Tests

<https://bitesizebio.com/21422/r-you-ready-using-r-for-statistical-tests>

### An Introduction to Statistical and Data Sciences via R (Modern Dive e-Book):

<http://www.moderndive.com/index.html>

(see also: <http://blog.revolutionanalytics.com/2017/02/moderndive.html>)

### Statistic on aiR:

<http://statistic-on-air.blogspot.com>

### Linear Regression from Scratch in R:

<https://datascienceplus.com/linear-regression-from-scratch-in-r>

### Survival analysis with R:

<https://rviews.rstudio.com/2017/09/25/survival-analysis-with-r>

### Practical Data Science for Stats:

<http://blog.revolutionanalytics.com/2017/09/practical-data-science-for-stats.html> >>>

<https://peerj.com/collections/50-practicaldatascistats>

### The Practical R- Primer on Statistical Programming in R:

<https://thepracticalr.wordpress.com>

### Online R Courses:

Udemy: <https://www.udemy.com/courses/academics/math-and-science>

<https://www.udemy.com/data-analysis-with-r>

Udemy Basic R Course (incl ggplot & Free): <https://www.udemy.com/machlearn1>

EDx, Coursera, Stanford Lagunita, FutureLearn and other MOOC providers also offer various R courses

DataCamp and DataQuest are other sources of online R courses and tutorials

Statistics.com: <http://www.statistics.com/course-catalog> (R-bloggers link)

### Stata to R:

#### A Quick and Easy Way To Turn Your Stata Knowledge Into R Knowledge:

<http://dlab.berkeley.edu/blog/quick-and-easy-way-turn-your-stata-knowledge-r-knowledge> Error! Hyperlink reference not valid.

### Statistics 506: Computational methods and tools in statistics

<http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506> Error! Hyperlink reference not valid.

Stata intro: [http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506/stata\\_intro](http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506/stata_intro) Error! Hyperlink reference not valid.

R intro: [http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506/r\\_intro](http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506/r_intro) Error! Hyperlink reference not valid.

R tips and common errors: [http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506/r\\_common\\_errors](http://dept.stat.lsa.umich.edu/~kshedden/Courses/Stat506/r_common_errors)

### R Statistics & Programming:

> R for Beginners: Basic R code for common statistical procedures - part I : <https://dmwiiq.net/2016/12/18/r-for-beginners-basic-r-code-for-common-statistical-procedures-part-i>

> R for Beginners: Basic R code for common statistical procedures - part II: <https://dmwiig.net/2016/12/27/r-for-beginners-some-simple-r-code-to-do-common-statistical-procedures-part-two>

**Getting Started in R~Stata- Notes on Exploring Data**  
<http://dss.princeton.edu/training/RStata.pdf>

\* \* \* \* \*

## Some Applications:

**Merge the two tables based on common ID field (in R):**

**FROM:** [http://manuals.bioinformatics.ucr.edu/home/R\\_BioCondManual](http://manuals.bioinformatics.ucr.edu/home/R_BioCondManual)

```
# search " merge(frame1, frame2, by.x = "frame1col_name", by.y = "frame2col_name",
all = T) "
# master <- merge(master, target, by.x="id", by.y="id", all.x=TRUE)
# x is the first dataframe ("master") and y is the dataframe being merged
("target")
# To obtain only the common rows, change 'all = TRUE' to 'all = FALSE'
# Example (merging two SNP genotype files to the master 1KG_HLA file):
frame1 <- read.csv(file.choose()) # select the master file (1KG_HLA.csv)
frame2 <- read.csv(file.choose()) # select SNP1 genotype file
frame3 <- read.csv(file.choose()) # select SNP2 genotype file
master <- merge(frame1, frame2, by.x = "id", by.y = "id", all = TRUE)
master1kg <- merge(master, frame3, by.x = "id", by.y = "id", all = TRUE)
head (master1kg, 1) # to confirm that the merging is successful
write.csv(master1kg, "master.csv") # exports the merged file as a CSV file to
working directory
```

**Making a WordCloud in R:**

> R-Bloggers: <https://www.r-bloggers.com/word-cloud-in-r>  
> Script (with annotations): <http://www.stat.cmu.edu/~hseltman/files/makeWordCloud.R> (needs editing)  
> <http://www.sthda.com/english/wiki/text-mining-and-word-cloud-fundamentals-in-r-5-simple-steps-you-should-know>  
> <http://www.sthda.com/english/wiki/word-cloud-generator-in-r-one-killer-function-to-do-everything-you-need>  
> R Graph Gallery: <https://www.r-graph-gallery.com/wordcloud>  
> <https://www.analyticsvidhya.com/blog/2014/05/build-word-cloud-text-mining-tools>  
> Changing fonts: <https://stackoverflow.com/questions/43460652/change-font-in-wordcloud-package-r>  
> package("wordcloud"): <https://github.com/manjushajoshi/R-code/blob/master/DiggingdatawithR-v2/DiggingdatawithR-v2.R> (see also: <https://thenewstack.io/data-visualization-basics-r-programming-language/>), wordcloud function (options): <https://www.rdocumentation.org/packages/wordcloud/versions/2.6/topics/wordcloud>

**10 Favorite R Packages and the Cool Things You Can Do with Them (Bitesize):**

<http://bitesizebio.com/23003/my-10-favorite-r-packages-and-the-cool-things-you-can-do-with-them>

**The Undergraduate Guide to R:** <https://sites.google.com/site/undergraduateguidetor/manual-files>

**Fun Data for Teaching R:** <https://bartomeuslab.com/2016/01/21/fun-data-for-teaching-r>

**Two minutes video tutorials:** <http://www.twotutorials.com>

**Top R language resources to improve your data skills:**

<https://www.computerworld.com/article/2497464/top-r-language-resources-to-improve-your-data-skills.html>

\* \* \* \* \*

## Useful R Functions: A random selection

### ## If you are going to download lots of packages, start with:

```
chooseCRANmirror()          # OR select "Set CRAN Mirror..." in the Packages menu
# and select the closest mirror to you for all downloads in the current session
```

### ## Object:

# In R, every data structure is an object which belongs to a class. To find out the class of an object (x):

```
data.class(x)
```

#### # Examples:

```
data.class(iris)
> [1] "data.frame"
data.class(iris$Sepal.Length)
> [1] "numeric"
data.class(iris$Species)
> [1] "factor"
```

### ## Creating a data frame:

```
df <- data.frame()          # creates an empty data frame
fix(df)                     # allows to fill it up
```

### ## Editing a dataframe:

```
data(iris)
edit(iris) # Opens a spreadsheet for editing
```

#### #OR:

```
fix(iris)
```

# The difference between edit() and fix() is pretty trivial:

## edit() lets you edit an object and returns the new version

## fix() lets you edit an object and modifies the original

### ## Converting a horizontal dataframe to vertical (long) format (converting multiple columns in a dataset to a single column):

# Dataframe `iris` is in long format. In vertical format, there would be columns for each of the three species and their data would be in those columns rather than having a species variable and having all the data for all species in single columns.

Package: `reshape2`

Function: `melt()`

See: <https://www.statmethods.net/management/reshape.html>

<https://www.r-bloggers.com/melt>

<https://seananderson.ca/2013/10/19/reshape>

### ## Copying an Excel file to R via the clipboard:

# Copy the spreadsheet on clipboard, and import it to R using the following function:

```
x <- read.delim("clipboard", header = TRUE) # if no header, use FALSE
```

### ## Inserting multiple plots in one panel (not in ggplot):

```
par(mfrow = c(3,1))
boxplot(iris$Sepal.Length); boxplot(iris$Sepal.Width); boxplot(iris$Petal.Width)
```

#### #OR

```
par(mfrow = c(2,2)) # par = parameters, and mfrow = multiple figures row-wise
boxplot(iris$Sepal.Length); boxplot(iris$Sepal.Width);
boxplot(iris$Petal.Length); boxplot(iris$Petal.Width)
```

### ## Having two plots simultaneously on two different graphics windows:

# FROM: <https://stackoverflow.com/questions/20135744/how-to-open-new-plot-window-during-plot>

```
library("DataExplorer")
data(iris)
plot_histogram(iris)
windows() # opens a new graphics window while keeping the first one open
plot_density(iris)
```

#OR in one line:

```
plot_histogram(iris); windows(); plot_density(iris)
```

### ## Making R to pause between plots:

# Each plot created in R overwrites the previous one. You can set R to pause between plots so you can view each one before it's overwritten.

# Use the global graphics option called "ask" is set to TRUE, R will pause before each new plot:

```
par(ask = TRUE)
```

# When you are tired of R pausing between plots, set it to FALSE:

```
par(ask = FALSE)
```

# An example is provided in the demo\_ggplot2.R script.

### ## Creating a variable and printing it (1: the usual way):

```
x <- (3 * 4)
```

```
x
```

```
# [1] 12
```

### ## Creating a variable and printing it (2: the quick way):

```
(x <- (3 * 4)) # wrapped by brackets
```

```
# [1] 12 # printing is automatic
```

### ## file.choose() to select a file from anywhere on the computer:

```
x <- read.csv(file.choose(), header=TRUE) # if no header, use FALSE
```

```
dim(x) # to check the dimensions of the imported CSV file
```

### ## Demos

# Type 'demo()' for some demos {like: >demo(glm.vr)}, 'help()' for on-line help, 'help.start()' for an HTML browser interface to help, help.search() for online search {like: help.search("glm")}. Type 'q()' to quit R.

### ## How to Cite:

```
citation() # For R
```

```
citation("DescTools") # For a specific package
```

## Check: R Functions of the Day: <https://rfunction.com> for more tricks

## Generating Factor Levels with gl() function. SEE: [https://www.tutorialspoint.com/r/r\\_factors.htm](https://www.tutorialspoint.com/r/r_factors.htm)

### ## Dealing with dataframes:

```
data() # To find out which built-in dataframes come with base R
```

```
data(women) # Calls the built-in dataframe "women"
```

```
write.csv(women, "test.csv") # Exports the dataframe "women" as a CSV file (test.csv) to the default working directory
```

```
csv <- read.csv("test.csv") # Imports the dataframe "women" as a CSV file (test.csv) from the default working directory
```

```
csv <- read.csv(file.choose()) # If unsure of the file location, you can browse your computer
```

```
csv <- read.csv("c:\\temp\\women.csv") # You can also type the full location (in this case, the file is in c:/temp folder)
```

## ## Working directory:

```
getwd()      # To find out the working directory where exported files will be saved and files to be
              imported will be looked for by default
setwd()      # To set the working director like setwd("C:\TMP") for the duration of the session
```

## ## List of objects / Removal of objects:

```
ls()         # list current objects
rm(object)   # delete an object
rm(list=ls()) # delete all objects
```

## ## Clearing the console or specific variables:

```
rm(list=ls(all=TRUE)) # To clear R console (everything generated in the current session will
                      # be removed)
rm(variablename)     # To clear a specific variable
(Just to clear the screen without removing anything g from the memory, press CTRL + L)
```

## Generating numerical vectors:

```
seq(from=1, to=3, by=0.1) # To have a series of numbers starting from 1 to 3 and increasing by
                           0.1. This can also be written as: seq(1, 3, 0.1)
runif(4)                  # To generate four random numbers within the default range of 0 to 1
runif(10, min=1, max=100) # To generate 10 random numbers within the range 1 to 100
                           # For "runif",
                           # see: http://www.cookbook-r.com/Numbers/Generating\_random\_numbers
rnorm(n, mean = 0, sd = 1) # To generate "n" number of random numbers fitting to normal
                           distribution with default mean=0 and sd=1
rnorm(25, 30, 5)         # To generate a sample of size 25 from a normal distribution with mean
                           30 and standard deviation 5
a <- rbinom(100, 10, 0.5) # 100 people tossing a fair coin (third argument = 0.5) 100 times
mean(a)                  # should be close to 10*0.5 = 5
c(1,2,3,4)               # to create a vector of these numbers
x <- sample(1:10, 50, replace = TRUE)
                           # creates a vector of 50 elements drawn from 1 to 10 with replacement
x
  [1] 2 1 6 9 1 3 5 2 10 2 7 8 8 4 9 2 2 6 9 10 5 6 2 5 7
 [26] 9 1 6 10 1 10 5 1 7 5 2 7 9 2 6 5 1 4 9 2 4 2 4 8 3

y <- rep(1, 100)         # creates a vector by repeating the element '1' one hundred times
```

## Generating a numerical matrix:

```
M1 = matrix( c(2, 4, 3, 1, 5, 7), nrow=3, ncol=2) # to create a matrix of these
six numbers as three rows and two columns
or
M2 = matrix( nrow=2, ncol=2, data=c(1,3,5,9), byrow=T )

length(x) # to get the length of vector x
```

## # To create a simulated dataframe:

```
c1 <- rnorm(100, 1, 3)
c2 <- rnorm(100, 2, 5)
c3 <- rnorm(100, 3, 6)
a <- data.frame(c1, c2, c3)
dim(a)
>100 3
```



### # To choose a subset of a dataframe based on a value in a column, try this:

```
data(iris)
petal0.4 <- iris[iris$Petal.Width == 0.4, ]
petal0.4
# it will print the data only for Petal.Width = 0.4 (this subset is now a new object named petal0.4) .
```

### # OR:

```
data(iris)
iris_subset2 <- subset(iris, Species = "virginica")
iris_subset2
# it will print the subset of iris for Species = virginica only.
```

### # OR:

```
iris_subset3 <- iris[iris$Species == "virginica", ]
iris_subset3
# it will print the subset of iris for Species = virginica only.
```

### # We can extract specific rows of that column using square brackets, [] :

```
subset_Puromycin <- Puromycin[10:15, 2:3]
```

```
iris_subset <- subset(iris, select = c(Sepal.Length, Petal.Length))
head(iris_subset)
```

### # Alternative to head():

```
data(iris)
iris[95:105, ] # prints rows 95 to 105 (and all columns)
iris[100:105, 1:4] # prints rows 95 to 105 and only the first four columns
# equivalent to iris[100:105, -5]
```

### # To find out the dimensions, row and column numbers of a dataframe:

```
dim(dataframename)
nrow(dataframename)
ncol(dataframename)
```

### # attach() and detach()

```
# attach(dataframename) # The dataframe is now the default dataset for any manipulation and need
# not be defined by name in following procedures
# detach(dataframename) # Cancels the attach() and removes the dataframe from the search
# path
```

### ## Loading an RData file:

```
load("filename.RData")
```

### # To save a dataframe as RData file:

```
save(dataframename, file = "filename.RData")
```

### # apply() will apply the same command (like mean()) to all rows (second argument = 1) or columns (second argument = 2) [mean() function only works for numeric data (rows or columns)]

```
apply(iris[-5], 1, mean) # will give means of each row
apply(iris[-5], 2, mean) # will give means of each column
```

### A t-test demonstration:

```
a <- rnorm(25, 30, 5) # To generate a sample of size 25 from a normal distribution
# with mean 30 and standard deviation 5
b <- rnorm(25, 25, 7) # To generate a sample of size 25 from a normal distribution
# with mean 25 and standard deviation 7
t.test(a,b) # To run the t-test
```

```
# Change the sample size (first argument) in a and b to show the statistical power concept
# Change the SD (third argument) to show the effect of intra-group variability
# Change the mean (second argument) to show the effect of difference in the mean
```

### # Boxplot by categories in a table:

```
data <- Puromycin
head(Puromycin)
boxplot(data$conc ~ data$state)      # state variable should be a factor
# OR:
by(data$conc, data$state, boxplot)  # this will generate two boxplots for each level of the factor
                                     variable "state) and only the last one will be visible
par(mfrow = c(1,2))                 # changes the settings for two plots to appears in two columns
by(data$conc, data$state, boxplot)  # the two boxplots will be side-by-side

boxplot((ChickWeight[ChickWeight$Time == 21, 1]) ~
(ChickWeight[ChickWeight$Diet]))
boxplot(ChickWeight$weight ~ ChickWeight$Diet, xlab = "Weight", xlab = "Diet")
```

```
OPTIONS for boxplot: pch = 24, col = "red", main = "MY PLOT", col.axis="blue", col.lab="red",
col.main="darkblue", boxwex = 0.2
```

### # Adding a line to the boxplot:

```
data(iris)
boxplot(iris[-5])
abline(h = 2.5, col = "blue")
```

### # Correlation:

```
data(women)
data <- women
cor(data$height, data$weight)        # Provides Pearson's correlation coefficient (r) by default
cor(data$height, data$weight, method = "pearson")
cor(data$height, data$weight, method = "kendall")    # Provides Kendall's tau value when
                                                       specified
cor(data$height, data$weight, method = "spearman")  # Provides Spearman's rho value
                                                       when specified
cor.test(data$height, data$weight)    # Provides Person's correlation coefficient (r), its 95% CI
                                       and the P value
cor(data)                             # Generates a matrix of all pairwise correlations in the
                                       dataset/dataframe for Pearson test by default
cor(data, method = "spearman")        # Generates a matrix of all pairwise correlations in the
                                       dataset for Spearman test
pairs(data)                           # Generates plots of pairwise correlations in the whole
                                       dataset
```

### # Correlation scatter plot by a factor (in different colours):

```
library("ggplot2")
data(iris)
qplot(Petal.Width, Petal.Length, color = Species, data = iris)
```

### # Correlation matrix with heatmap:

```
library(DataExplorer)
plot_correlation(iris)
```

### # Correlation matrix with colored ellipses ([http://www.cookbook-r.com/Graphs/Correlation\\_matrix](http://www.cookbook-r.com/Graphs/Correlation_matrix))

```
install.packages("ellipse")
library("ellipse")
```

```

data(iris)
ctable <- cor(iris[-5])    # if desired, specify the decimal numbers by round(ctable, 2)
plotcorr(ctable)          # no colors
# Adding color:
colorfun <- colorRamp(c("#CC0000", "white", "#3366CC"), space="Lab") # Defining the
                                                                    color palette
plotcorr(ctab, col=rgb(colorfun((ctab+1)/2), maxColorValue=255), mar = c(0.1,
0.1, 0.1, 0.1))          # Same graph with color and reduced margins

```

### # J-shaped curve = $2*(x**2) - 3*x + 4$

```

X = seq(1:1000)
y = 2*((seq(1:1000))**2) - 3*(seq(1:1000)) + 4
plot(x,y)

```

### # Simulating a dataframe:

```

set.seed(955)
vvar <- 1:20 + rnorm(20, sd=3)
wvar <- 1:20 + rnorm(20, sd=5)
xvar <- 20:1 + rnorm(20, sd=3)
yvar <- (1:20)/2 + rnorm(20, sd=10)
zvar <- rnorm(20, sd=6)
data <- data.frame(vvar, wvar, xvar, yvar, zvar)
head(data)
#>      vvar      wvar      xvar      yvar      zvar
#> 1 -4.252354  5.1219288 16.02193 -15.156368 -4.086904
#> 2  1.702318 -1.3234340 15.83817 -24.063902  3.468423
#> 3  4.323054 -2.1570874 19.85517  2.306770 -3.044931
#> 4  1.780628  0.7880138 17.65079  2.564663  1.449081
#> 5 11.537348 -1.3075994 10.93386  9.600835  2.761963
#> 6  6.672130  2.0135190 15.24350 -3.465695  5.749642

```

## **Genomics & Bioinformatics analysis:**

eBook: *Computational Genomics with R*: <https://compgenomr.github.io/book>

[A Little Book of R for Bioinformatics](#).pdf

**Bioinformatics for Beginners - Bioconductor (Omixon):** <https://www.omixon.com/bioinformatics-for-beginners-bioconductor>

**Bioconductor Cheat Sheet:**

<https://github.com/mikelove/bioc-refcard>

**R & Bioconductor Manual:**

[http://manuals.bioinformatics.ucr.edu/home/R\\_BioCondManual](http://manuals.bioinformatics.ucr.edu/home/R_BioCondManual)

**Bioconductor Installation:**

<http://127.0.0.1:27887/library/BiocInstaller/html/biocLite.html>

**Bioconductor workflows:** <http://www.bioconductor.org/help/workflows>

Variant Annotations: <http://www.bioconductor.org/help/workflows/variants>

**Introduction to Bioconductor and R Refresher:**

<http://bioinformatics-core-shared-training.github.io/cruk-bioinf-sschool/Day1/bioc-intro.pdf>

**Introduction to R: Exploring the genes of the human genome:**

[https://jdblischak.github.io/2013-09-19-chicago/lessons/uchicago-r/intro\\_exploring\\_human\\_genome.html](https://jdblischak.github.io/2013-09-19-chicago/lessons/uchicago-r/intro_exploring_human_genome.html)

**Introduction to R: Evolution and Genomics:**

<http://evomics.org/learning/programming/introduction-to-r>

**Teaching RNAseq at Undergraduate Institutions: A tutorial and R package from the Genome**

**Consortium for Active Teaching @ CourseSource:** <https://www.coursesource.org/courses/teaching-rnaseq-at-undergraduate-institutions-a-tutorial-and-r-package-from-the-genome-0> (may be obsolete, but see all courses @ <https://qubeshub.org/community/groups/coursesource/publications>)

**Big Data to the Bench- Transcriptome Analysis for Undergraduates (CBE Life Sci Educ 2019) @**

<https://www.lifescied.org/doi/10.1187/cbe.18-08-0161>

<https://qubeshub.org/community/groups/coursesource/publications?id=2538&v=1>

<https://qubeshub.org/publications/2538/serve/1/9499?el=1&download=1>

<https://bitbucket.org/petersmp/publishedtutorials/src/master>





<https://github.com/gladstone-institutes/Bioinformatics-Workshops/wiki/Introduction-to-Pathway-Analysis-Using-R>

Using [genomation](#) to analyze methylation profiles from Roadmap epigenomics and ENCODE:  
<http://zvfak.blogspot.com/2015/03/using-genomation-to-analyze-methylation.html>

Annotating sets of genomic intervals with genomic annotations such as chromHMM:  
[http://zvfak.blogspot.com/2016/08/annotating-sets-of-genomic-intervals\\_12.html](http://zvfak.blogspot.com/2016/08/annotating-sets-of-genomic-intervals_12.html) (using [genomation package](#))

Summary, annotation and visualization of genomic intervals with [genomation package](#):  
<http://zvfak.blogspot.com/2014/02/summary-annotation-and-visualization-of.html>

Finding Candidate Binding Sites for Known Transcription Factors:  
<http://www.bioconductor.org/help/workflows/generegulation>

Changing genomic coordinate systems:  
<http://www.bioconductor.org/help/workflows/liftOver>

Getting Genetics Done:  
<https://www.gettinggeneticsdone.com>

7 Interactive Bioinformatics Plots made in Python and R:  
<https://moderndata.plot.ly/bioinformatics-plots-made-in-python-and-r>

15 practical tips for a bioinformatician:  
<http://www.r-bloggers.com/my-15-practical-tips-for-a-bioinformatician>

### Phylogenetics in R:

Quick and Dirty Tree Building in R (Mol Ecol): <https://www.molecularecologist.com/2016/02/quick-and-dirty-tree-building-in-r>

Phylogenetic Tree Reconstruction with R (ReconLearn): <https://www.reconlearn.org/post/practical-phylogenetics.html>

### Population Genetics in R:

eBook: *Applied Population Genetics*: [https://dyerlab.github.io/applied\\_population\\_genetics/index.html](https://dyerlab.github.io/applied_population_genetics/index.html)

Population Genetics and Genomics in R: [https://grunwaldlab.github.io/Population\\_Genetics\\_in\\_R](https://grunwaldlab.github.io/Population_Genetics_in_R)

See also: Best Practices for Population Genetic Analyses (Pythopathology 2017) @  
<https://apsjournals.apsnet.org/doi/10.1094/PHYTO-12-16-0425-RVW>

Population Genetics in R Web Interface: <https://popgen.nescent.org>

REF: Developing educational resources for population genetics in R: an open and collaborative approach:  
<https://onlinelibrary.wiley.com/doi/full/10.1111/1755-0998.12558>

Calculating Basic Population Genetic Statistics from SNP Data:

<https://popgen.nescent.org/StartSNP.html>

Detection of the signal of selection from genome scan - Population based analysis:

<https://popgen.nescent.org/2016-01-26-SNP-selection.html> (CHECK all tutorials under the Users tab  
@ <https://popgen.nescent.org>)

Types of Clustering Methods- Overview and Quick Start R Code ([Data Novia](#)):

<https://www.datanovia.com/en/blog/types-of-clustering-methods-overview-and-quick-start-r-code>

Creating a phylogeny from the 1000 Genome Database:

<https://benbowlab.github.io/Phylogeny.html>

**learnPopGenet:** <https://www.rdocumentation.org/packages/learnPopGen/versions/1.0.4>

**REF:** <https://onlinelibrary.wiley.com/doi/epdf/10.1002/ece3.5412>

Web interface: <http://www.phytools.org/PopGen>

GitHub (codes): <https://github.com/liamrevell/learnPopGen/tree/master/man>

**PCA analysis:**

<https://intobioinformatics.wordpress.com/2019/05/22/easy-quick-pca-analysis-in-r>

<https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python>

**Genome-wide association studies in R:**

<https://poissonisfish.wordpress.com/2017/10/09/genome-wide-association-studies-in-r>

**qqman: an R package for creating Q-Q and manhattan plots from GWAS results:**

<http://www.r-bloggers.com/qqman-an-r-package-for-creating-q-q-and-manhattan-plots-from-gwas-results>

**Pathway Analysis for High-Throughput Genomics Studies Using R:**

<http://www.gettinggeneticsdone.com/2012/03/pathway-analysis-for-high-throughput.html>

**Annotating sets of genomic intervals with genomic annotations such as chromHMM on R:**

[http://zvfak.blogspot.co.uk/2016/08/annotating-sets-of-genomic-intervals\\_12.html](http://zvfak.blogspot.co.uk/2016/08/annotating-sets-of-genomic-intervals_12.html)

**SuRFing the genomics wave: an R package for prioritising SNPs by functionality:**

<https://genomemedicine.biomedcentral.com/articles/10.1186/s13073-014-0079-1>

**R tutorial for methylation clock analysis:**

<https://genomebiology.biomedcentral.com/articles/10.1186/gb-2013-14-10-r115>

[13059\\_2013\\_3156\\_MOESM20\\_ESM.docx](#) Additional file 20: **R software tutorial**. This file contains an R software tutorial that describes how to estimate DNAm age for data set 55. Further, it shows how to relate two measures of age acceleration to autism disease status. The R tutorial requires Additional files [21](#), [22](#), [23](#), [24](#), [25](#), [26](#) and [27](#) as input. (DOCX 57 KB)

[13059\\_2013\\_3156\\_MOESM21\\_ESM.csv](#) Additional file 21: **Probe annotation file for the Illumina 27K array**. This comma-delimited text file (.csv file) is needed for the R software tutorial. (CSV 1 MB)

[13059\\_2013\\_3156\\_MOESM22\\_ESM.csv](#) Additional file 22: **Additional probe annotation file for the R tutorial**. This comma-delimited text file (.csv file) is needed for the R software tutorial. (CSV 1 MB)

[13059\\_2013\\_3156\\_MOESM23\\_ESM.csv](#) Additional file 23: **Coefficient values of the age predictor**. This comma-delimited text file (.csv file) is needed for the R software tutorial. This file is very similar to Additional file [3](#) but rows appear in a different order. (CSV 131 KB)

[13059\\_2013\\_3156\\_MOESM24\\_ESM.txt](#) Additional file 24: **R code for normalizing the DNA methylation data**. This text file is needed for the R software tutorial. It contains R code for normalizing the DNA methylation data and adapts R functions described in [\[89\]](#). (TXT 20 KB)

[13059\\_2013\\_3156\\_MOESM25\\_ESM.txt](#) Additional file 25: **This text file is needed for the R software tutorial**. It contains R code implementing analysis steps. (TXT 6 KB)

[13059\\_2013\\_3156\\_MOESM26\\_ESM.csv](#) Additional file 26: **Methylation data from data set 55**. This comma-delimited text file (.csv file) contains the DNA methylation data needed for the R software tutorial. (CSV 5 MB)

[13059\\_2013\\_3156\\_MOESM27\\_ESM.csv](#) Additional file 27: **This comma-delimited text file (.csv file) contains the sample annotation data needed for the R software tutorial**. (CSV 2 KB)

**R script for correlation analysis between genotypes and phenotypes in 1KG datasets:**

Identification of minor histocompatibility antigens based on the 1000 Genomes Project Cell Therapy & Immunotherapy 2014.PDF: 1000 Genomes Project correlation analysis

The correlation between mHag phenotypes of individuals included in the 1000 Genomes Project and the more than 40 million variations of these individuals was analyzed using the free software environment R, v.2.15.2. Genotype files from the 1000 Genomes Project were downloaded from their ftp-server (<ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp>) and modified to allow analysis on a regular PC. The R-script runs a Fisher's Exact test for the mHag phenotypes



You'll need to click through the link to start the demo in a new window, and you'll need to be using Chrome and allow access to your microphone. (If you can't get it to work on your system, here's a [video demo of Yihui using the app](#).) But with that done, you can use voice commands like "color green" to change the point color, the title, or even add a regression line.

Behind the scenes, this is a simple [Shiny app](#), implemented in the [R language](#). (You can find the [R code for the app here](#).) The voice recognition is handled by the [annyang](#) Javascript library, which in turn triggers events captured in R with the Shiny.onInputChange function. It's not perfect (try as I might I couldn't get it to put "customized" instead of "customize" in the title), but it's a very promising prototype of voice-controlled apps in R.

Yihui Xie: [A Shiny app based on annyang that responds to voice input](#)

## Online eBooks on R:

**Big Book of R** @ <https://www.bigbookofr.com>

**R Community Public Library of eBooks** @ <https://rviews.rstudio.com/2021/11/04/bookdown-org>

**Learn R : 12 Free Books and Online Resources** @ <http://ucanalytics.com/blogs/learn-r-12-books-and-online-resources>

**An Introduction to R for Research** @ <https://bookdown.org/rwnahhas/IntroToR>

**A Sufficient Introduction to R** @ <https://dereksonderegger.github.io/570L>

**GeeksForGeeks R Tutorial** @ <https://www.geeksforgeeks.org/r-tutorial>

**Getting started in R and RStudio** @ [http://milton-the-cat.rocks/learnr/r/r\\_getting\\_started](http://milton-the-cat.rocks/learnr/r/r_getting_started)

(See also: <https://www.r-bloggers.com/2020/08/tutorial-getting-started-with-r-and-rstudio>)

**R Without Statistics** @ <https://book.rwithoutstatistics.com>

**R Graphics Cookbook** @ <https://r-graphics.org>

**Data Visualization** @ <https://socviz.co>

**Fundamentals of Data Visualization** @ <https://clauswilke.com/dataviz>

**R for Data Analysis and Visualization** @ <https://jonpage.github.io/r-course>

**Data Visualisation Using R, for Researchers Who Don't Use R** @ <https://psyteachr.github.io/introdataviz>

**An Introduction to ggplot2** @ <https://bookdown.org/ozancanozdemir/introduction-to-ggplot2>

**Becoming Visual-R Fundamentals** @ <http://becomingvisual.com/rfundamentals/index.html>

**Hands-On Programming with R** @ <https://rstudio-education.github.io/hopr>

**R Packages** @ <https://r-pkgs.org/index.html>

**Introduction to Data Science** @ <https://rafalab.github.io/dsbook>

**R+ Data Science** @ <https://yards.albert-rapp.de/index.html>

**Statistical Thinking for the 21st Century** @ @ <https://statstinking21.github.io/statstinking21-core-site>

(See also R Companion: <https://statstinking21.github.io/statstinking21-R-site>)

**Introduction to Statistical Learning in R** @ <https://fcorowe.github.io/sl>

**Introduction to Statistical Thinking** @ <https://pluto.huji.ac.il/~msby/StatThink>

**Learning Statistics with R** @ <https://learningstatisticswithr.com/book>

(Download as PDF @ <https://learningstatisticswithr.com/lr-0.6.pdf> ; see also: <https://learningstatisticswithr.com>)

**Stats Book (Medical Statistics)** @ <http://pcool.dyndns.org:8080/statsbook>

**Essential Statistics with R** (Statistics on NHANES data) @ <https://bioconnector.github.io/workshops/r-stats.html>

**R Practices for Learning Statistics** @ [https://bookdown.org/logan\\_kelly/r\\_practice](https://bookdown.org/logan_kelly/r_practice)

**Lessons in Statistical Thinking** @ <https://dtkaplan.github.io/Lessons-in-statistical-thinking>

**Statistical Inference via Data Science** @ <https://moderndiver.com>

**Introduction to R for Data Science: A LISA 2020 Guidebook** @ <https://bookdown.org/jdholster1/idsr>

**Reproducible Medical Research with R** @ [https://bookdown.org/pdr\\_higgins/rmrwr](https://bookdown.org/pdr_higgins/rmrwr)

**Introduction to R for Health Data Science** @ [https://bookdown.org/m\\_p\\_sperin/intro\\_to\\_r\\_2020](https://bookdown.org/m_p_sperin/intro_to_r_2020)

**The Epidemiologist R Handbook** @ <https://epirhandbook.com>

**R-Tips eBook** @ [https://bookdown.org/lyzhang10/lzhang\\_r\\_tips\\_book](https://bookdown.org/lyzhang10/lzhang_r_tips_book)

**Bioconductor 2018 Workshop Compilation** @ <https://bioconductor.github.io/BiocWorkshops>

**Computational Genomics with R** @ <https://compgenomr.github.io/book>

**Doing Meta-analysis in R** @ [https://bookdown.org/MathiasHarrer/Doing\\_Meta\\_Analysis\\_in\\_R](https://bookdown.org/MathiasHarrer/Doing_Meta_Analysis_in_R)

(See also: A nontechnical primer for conducting a meta-analysis to synthesize correlational data @ [https://github.com/dsquintana/corr\\_meta](https://github.com/dsquintana/corr_meta))

**Applied Population Genetics** @ [https://dyerlab.github.io/applied\\_population\\_genetics/index.html](https://dyerlab.github.io/applied_population_genetics/index.html)

**Biomedical Knowledge Mining using GOSemSim and clusterProfiler** @ <https://yulab-smu.top/biomedical-knowledge-mining-book>

**Rhandsontable** @ <https://jrowen.github.io/rhandsontable/#Introduction>

(See also CRAN: <https://cran.r-project.org/web/packages/rhandsontable/index.html> / R Manual: <https://cran.r-project.org/web/packages/rhandsontable/rhandsontable.pdf> / Vignette: [https://cran.r-project.org/web/packages/rhandsontable/vignettes/intro\\_rhandsontable.html](https://cran.r-project.org/web/packages/rhandsontable/vignettes/intro_rhandsontable.html))

**Computer-age Calculus with R** @ <https://dtkaplan.github.io/RforCalculus>

**MOSAIC Calculus** @ <https://www.mosaic-web.org/MOSAIC-Calculus>

**Experimental Methods in Agriculture** @ <https://www.statforbiology.com/statbookeng>

**How to Build a Shiny Application from Scratch** @

[https://bookdown.org/hadrien/how\\_to\\_build\\_a\\_shiny\\_app\\_from\\_scratch](https://bookdown.org/hadrien/how_to_build_a_shiny_app_from_scratch)

**Mastering Shiny** @ <https://mastering-shiny.org>

**R Markdown** @ The Definitive Guide @ <https://bookdown.org/yihui/rmarkdown>

**R Markdown Cookbook** @ <https://bookdown.org/yihui/rmarkdown-cookbook>

**RMarkdown for Scientists** @ <https://rmd4sci.njitierney.com>

**R Workflow (R Workflow for Reproducible Biomedical Research Using Quarto)** @ <http://hbiostat.org/rflow>

**Data Science at the Command Line- Obtain, Scrub, Explore, and Model Data with Unix Power Tools** @ <https://jeroenjanssens.com/dsatcl>