One-Page Intro to R

Sara Emily Burke • Created 2012-09-18 • Last updated 2020-03-08

To download R, visit www.r-project.org, click "download R," and then choose any mirror site. Follow the directions for your operating system.

Executing R scripts

- Type commands into the command line to see immediate output.
- Type out lists of commands in a script file. From the R program, execute selected commands from a script using ctrl+R or command+enter.

If you need help

- For help with a specific function or command, type `?` followed by the name, e.g., `?aov`
- Use a search engine (e.g., Google). There are many free online tutorials for specific tasks in R.
- Try looking through this introductory manual: http://cran.r-project.org/doc/manuals/R-intro.html

Important symbols

`* + - / ^ ( )`	Basic arithmetic symbols work intuitively.
`x = 9` or `x <- 9`	variable assignment (these examples set `x` to the numeric value `9`)
`x="aa"` or `x='aa'`	Quotation marks enclose character data rather than numeric data.
`sum(2,3,5)`	Parentheses contain function input; commas separate multiple input values.
`x[2]`	Square brackets allow you to specify items within a vector.
`1:10`	A colon generates a vector of integers (in this example, from `1` to `10`).
`T TRUE F FALSE`	`TRUE` and `FALSE` are the Boolean values. `T` and `F` stand for them.
`< > <= >= != ==`	These symbols all compare two values, returning `True` or `False`.
`x == 9`	returns `True` if x is 9. Don't mix this up with the assignment operator!
`T|F T&F !F`	`|` means "or", `&` means "and", and `!` means "not"
`NA`	missing value
`# comment text`	Text that follows `#` on a line will be ignored by R.

Functions of vectors

`x = c(1,2,3)`	concatenates a set of values to form a vector, then calls that vector `x`
`length(x)`	returns the number of items in vector `x`
`table(x)`	returns a table of the unique values in vector `x`
`sum(x) mean(x)`	`var(x) sd(x) min(x) max(x) median(x)` basic statistics of vectors
`x * 2`	returns a vector containing each item in vector `x` multiplied by `2`
`log(x)`	returns the natural log of each item in `x`
`x == 9`	returns a vector of Boolean values indicating whether each item in `x` is equal to `9`
`sum(x == 9)`	The sum of a Boolean vector treats `True` as 1 and `False` as 0.
`is.na(x)`	returns a vector of Boolean values indicating whether each item in `x` is missing
`x[y==9]`	If `x` and `y` are vectors of equal length, this syntax returns all values of `x` for which the corresponding value of `y` is equal to `9`.
`which(y==9)`	returns the vector of indices for which `y` is equal to `9`

Organizational functions

`ls()`	returns a vector of all variables currently assigned a value
`rm(x)`	removes or deletes the variable `x`
`library()`	attaches a package. Useful packages include `car` and `psych`
`search()`	identifies data frames and packages that are attached

Working with data frames

`read.table()`	creates a data frame from a text file. Remember to specify `header=T`
`read.csv()`	creates a data frame from a comma separated text file
`names(data)`	shows all the column/variable names in the data frame called `data`
`str(data)`	shows basic properties of each variable in the data frame called `data`
`attach(data)`	attaches the data frame called `data` so variables can be easily referenced

Formula syntax

`y~x1`	predict the variable `y` using the variable `x1`
`y~x1+x2`	predict the variable `y` using the main effects of `x1` and `x2`
`y~x1*x2`	predict the variable `y` using the main effects of `x1` and `x2` and their interaction
`factor()`	transforms a variable into a factor so `lm()` or `aov()` will treat it as such

Common statistical tests

Use `?` at the command line to learn how to specify the inputs for these functions.

`lm()`	linear model – regression or ANOVA
`summary()`	Input your `lm()` object and this function will give you a regression summary.
`anova()`	Compare two regression models.
`aov()`	linear model – ANOVA only
`leveneTest()`	Input your `lm()` or `aov()` object and this function will perform Levene's test. This function requires the `car` package. Don't forget the capital T in Test.
`TukeyHSD()`	Input your `aov()` object and this function will perform a Tukey test.
`t.test()`	t test – input can be a single variable, two variables, or a formula
`var.test()`	F test to compare two variances
`chisq.test()`	Pearson chi-squared test – input can be one or two variables
`cor.test()`	correlation significance test

Scale analysis

`cbind()`	Sticks a series of vectors next to each other as columns of a matrix.
`rowMeans()`	Computes row means of a matrix – use this for averaging variables into a scale. Specify `na.rm=T` to compute means ignoring missing values.
`cronbach()`	Cronbach's alpha of matrix columns – requires the `multilevel` package.
`principal()`	principal component analysis of matrix columns – requires the `psych` package. Use the `GPArotation` package for extra rotations.

Plotting

Checking `?par` and `?plot` is very helpful when plotting.

`par()`	set graphics parameters before plotting – see `?par` for details
`plot()`	scatter plot by default – input can be two variables or a formula
`abline()`	draw a line on the current plot – if the input is a `lm` object, it draws the best fit line
`legend()`	add a legend to the existing plot – see `?legend` for details
`hist()`	histogram
`qqnorm()`	normal Q-Q plot
`barplot()`	other basic plot types – see me for custom versions with features I commonly use
`boxplot()`	