60 R Language Tips

Things I Wish I Knew When I Started Out With R

Here are 60 R tips that we hope will be useful through your journey in R. This set of tips is a result of knowledge accumulated by experience through the years. Hope some of these are new to you and will enhance your R skills.

If you wish to keep up with more of these, follow our tweets on twitter at @R_Programming where you will find more of these tips and R related updates.

1. Never use `as.numeric()` to convert a factor variable to numeric, instead use `as.numeric(as.character(myFactorVar))`.

2. `options(show.error.messages=F)` turns printing error messages off.

3. Use `file.path()` to create file paths. It works independent of OS platform.

4. `mixedsort()` from gtools package sorts strings with embedded numbers so even the numbers are in correct order. This is not achieved by regular `sort()` function.

5. Use `ylim = range(myNumericData) + 10` as an argument in `plot()` function to set and adjust the Y axis limits in your plot

6. Use `las` parameter in your `plot()` to customise the orientation of axis labels. Accepted values are {0, 1, 2, 3} for {parallel to axis, horizontal, perpendicular to axis, vertical}

7. A compilation of advanced regression types in R: http://rstatistics.net/special-forms-of-regression/

8. Use `memory.limit (size=2500)`, where the size is in MB, to manage the maximum memory allocated for R on a Windows machine.

9. Use `alarm()` to produce a short beep sound at the end of your script to notify that the run has completed.

10. `eval(parse(text=paste ("a <- 10")))` will create a new variable 'a' and assign value 10 to it. It executes your strings as if they are R statements.

11. `sessionInfo()` gets the version information about current R session and attached or loaded packages.

12. Compute the number of changes in characters required to convert one word to another using `adist(word1, word2)`.

13. `options(max.print=1000000)` sets the max no. of lines printable in console. Adjust this if you want to see more lines.

14. Introducing practical and robust anomaly detection in a time series: https://blog.twitter.com/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series

15. Two R sessions running simultaneously is guaranteed to have unique IDs. Get the ID of current R session using `Sys.getpid()`

16. Remove the names attributes from an R object using the `unname()` function.

17. Check if two R objects are same with `identical(x,y)`. Use `all.equal()` to test if values are equal.

18. Extract twitter feed and user tweets from R console. http://rstatistics.net/extracting-tweets-with-r/

19. A quick and simplified introduction to Time series Analysis http://rstatistics.net/time-series-analysis/

20. Use `withTimeout()` function from R.utils package to interrupt functions if run time exceeds a preset time limit and move to next step.

21. Use `dist()` to compute the distance between rows of a matrix.

22. Use `diff()` to calculate lagged and iterated differences of a numeric vector.

23. Turn off printing scientific notation such `1e-5` in output, using `options(scipen=999)`

24. `bagEarth()` from earth package performs a bagged MARS (Multivariate Adaptive Regressive Spline)

25. `setClass('myClass')` will define a new user defined class called 'myClass". Use setAs() to further customisation.

26. `assign ("varName", 10)` is a convenient way to create numerous variables, as the var name can be passed as a programmable string.

27. `dim(matrix)` returns the number of rows and columns.

28. 2 Tips to write awesome R functions. https://www.youtube.com/watch?v=ahRHTXNjixU

29. `data.matrix()` converts a data frame to a numeric matrix. Factors will be converted to appropriate numeric values.

30. Use `invisible(..)` to suppress printing the output to console. Widely used from within functions.

31. `cat("\014")` clears the R Console in Windows.

32. `dir('folder path')` shows the files in 'folder path'. Works much like the same way as in windows cmd prompt.

33. Make missing values in a factor variable as another category in one-line using: `levels(Var) <- c (levels (Var), "UNKNOWN")`

34. Initialise all required packages in one line: `lapply(x, require, character.only=T)`, where `x` is char of all required package names

35. `rev(x)` reverses the elements of `x`

36. Use `complete.cases()` to get the rows which are complete (with no missing values)

37. `avNNet()` from nnet pkg to implement Model Averaged Neural Network

38. `file.remove('filepath')` removes the file from directory. Use this wisely to delete multiple files esp in repetitive tasks.

39. Use `ada()` in ada pkg to implement Boosted classification trees.

40. Use `unclass()` on objects like 'lm' to break it down to a 'list'. Makes it easier to access un-printed elements this way.

41. Sort a data frame based on 2 columns together: `df[order(df$col1, df $col2), ]`

42. Convert One 'N-level factor var' to N 'binary-predictor-vars' with `model.matrix(~as.factor(Data)+0)`.

43. Use `seasadj()` to de-seasonalize a time series. http://goo.gl/Oio7s2

44. Use `<<-` instead of `<-` operator to assign value to a variable that exists outside the function from which it is called.

45. Set the memory size R uses using `memory.limit(size=desired-size)` in windows platform. On other platforms, use `mem.limits()`

46. Use `file.copy(from=fromFile, to = toFile, overwrite=TRUE )` to copy files with R , works even between connected servers.

47. Use `debugonce()` to run through debug step only once, instead of `debug()` which requires `undebug()` to come out of it.

48. Convert a R Factor Variable To A Collection of Multiple 1/0 Binary Vars: `bins <- model.matrix( ~ 0 + varName, data)`. Highly useful in regression modelling.

49. `discretize()` from arules pkg is a convenient function to convert continuous variables to categorical. It has convenient split criteria options.

50. `NROW()` is similar to `nrow()` function, but even works on a vector, treating it as `1`-column matrix. You can safely use in place of `length()` function.

51. `commandArgs()` returns the cmd line arguments passed with R script run from command. http://bit.ly/1yARCWj

52. Use `attr(myFunc, "AttrName") <- myVal`, within the function, it remembers the "AttrName" var in next call.

53. Use `object.size()` to estimate the memory a given R object consumes in bytes.

54. Use `ls.str()` (over `ls()`) to see structural details of objects when working on large R projects.

55. `dir(path='dir_path')` lists all files and directories in location specified by 'dir_path'.

56. `library(help = libname)` displays the documentation with all functions and datasets from 'libname' library.

57. `get("objectNameString")` fetches the object with name "objectNameString". Use the 'envir' argument if your object is within a specific environment.

58. Run R scripts directly from command prompt with `"C:\your-R-path\R.exe" CMD BATCH --vanilla --slave "c:\project-path\my_script.R"`

59. Use `cor.test(x,y)` to find the correlation between x & y, and test the statistical significance.

60. Turn your analyses into interactive web applications with Shiny. Get the shiny cheatsheet here: http://bit.ly/1pFWGJW

More on our Full R Course on Youtube

If you are complete beginner, wanting to learn R, subscribe to the full R language course on our youtube channel at (http://www.youtube.com/user/TheLearnR). This course is designed to be much more beginner friendly than other courses especially because it was created assuming you have no prior knowledge of the language. It is a guarantee, if you just follow along and practice, you will get good with R in no time. Good luck.