Table of Contents

Clean and tidy R-script

A tidy script is easy to grasp quickly, and also to quickly spot mistakes (missing coma/parenthesis/mistype in the command). If you get good typing habits, it will not cost you more time when creating the script, but it will save you (potentially a lot of) time when reading or fixing the script in future. Some suggestions follow.

- **Use systematically the same style of syntax**. E.g. keep using space between function name and opening parenthesis, space after the comma separating arguments in a function, spaces around =, ==, ~ or `<-` operators etc. Or, do not, but always do not (but I would suggest you to keep using the spaces). Other example: if you assign variable, both = and `<-` are assign signs, so typing `a <- 10` and `a = 10` both results into creating object a and filling it by value `10`. Do not mix both `<-` and = in the same script, not to make it confusing (i.e. instead of `a <- 10; b = 15; d <- a + b` type `a <- 10; b <- 15; d <- a + b` (or, less optimally, `a = 10; b = 15; d = a + b`. If using `<-` sign, always use spaces around to avoid confusion (e.g. `a<-3` may seem to mean either “assign 3 to a”, or “a is less than -3” - the first is right, but to avoid this, use `a <- 3`).
- **Use meaningful and short names** for created R objects. If using names combined of several words, consider visually separating the words by dot (`my.data`), underscore (`my_data`) or lowercase-uppercase combination (`MyData`). But, don't mix styles (i.e. do not name one variables as `my.data.1` and the other as `my_data_2`, since for sure you will at least once type `my.data.2` in the following script). If you generate many variables, consider using names with the same number of letters, so as the script aligns into blocks. If you have several chunks of the code (i.e. the sequence of commands doing some job) and then you apply it on several different datasets (without changing anything else), consider naming the variables uniquely in the way that their name can be changed by find/replace function in RStudio.
- **Do not create many objects** if you don't necessarily need them. An (extreme) example can be: `a <- 34; b <- 2; result <- a^b` (making just `result <- 34^2` would be enough). But sometimes it may be better to separate complicated sequence of commands into more separate lines instead of wrapping all of them inside each other (this may not be easy to read). Or consider using piping (operator `%>%` from the library `magrittr`).
- **Sometimes to be more explicit is better than to be too concise**. This is e.g. the example of naming the arguments in the functions - you don't need to type them if you keep the argument order (e.g. `seq (1, 10, 1)` is doing the same as `seq (from = 1, to = 10, by = 1)`, but the more explicit (with arg. names) option could be more understandable to those who are not familiar with this order (in the case above, I would perhaps write `seq (1, 10, by = 1)`, since alternative option is `seq (1, 10, length.out = 100)`, which produces sequence from 1 to 10 with 100 elements).
- **Keep things simple**. There are often simple ways how to (in R) make a complex thing. E.g. instead of `a <- c(1,2,3,4,5)`, use `a <- 1:5`. However, if this simplifying would mean that for a relatively simple thing you need to use some library which is not installed in base R, it may be easier to type long instead of installing a package because of one function.

Example of tidy script:

```
col.rain <- rainbow (n = 50)   # rainbow chooses colors from hue circle
col.terr <- terrain.colors (n = 20)  # terrain colors
col.topo <- topo.colors (n = 20)  # topo colors
```

```
barplot (1:50, col = col.rain)
barplot (1:20, col = col.terr)
barplot (1:20, col = col.topo)
```

... and example of untidy script:

```
col_rainbow<- rainbow (n=50 )
col.ter =terrain.colors (n= 20)
barplot( 1:50,col =col_rainbow )
col_topog <-  topo.colors ( n =20)
barplot (c(1,2,3,4,5, 6,7,8,9,10,11,12,13,14,15 ,16,17,18,19, 20) ,col=
col.ter)

barplot (seq(1,20 ),col =col_topog)
```

Some follow up readings if interested: https://www.tidyverse.org/articles/2017/12/workflow-vs-script/