Statistics, Visualization and more using R

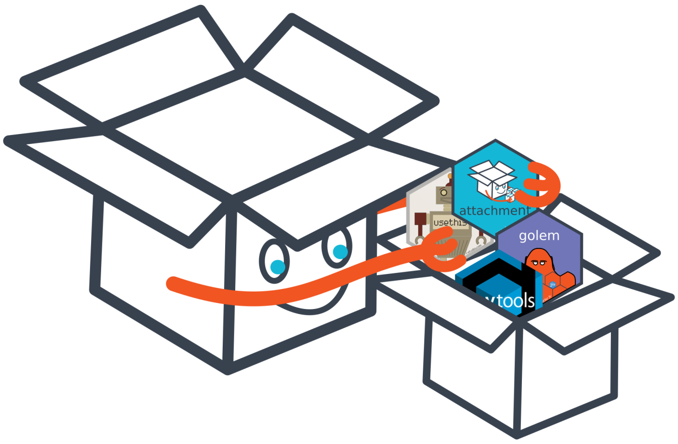
R Packages: A Beginner's Guide

Bradley Mackay, Clemens Ehlich

26. Juni 2020

Summary

One of the biggest advantages of the R programming language is the existence of an unbelievable number of extension packages, which make your daily work much easier. R packages are collections of functions and datasets and are mostly created by the community itself. As the title suggests, this short paper is about how to create your own little R package, because creating your own package has many advantages. Especially if you have to work on recurring tasks. For this we will first give a summary of important commands to work with packages. Then we will give an overview how packages are built and finally we will give a step by step guide how to create your first own package.



Inhaltsverzeichnis

1 Basics

1.1 What's a package

As already mentioned at the beginning R packages are collections of functions and data sets. They extend the basic functionalities of the Language itself or add new ones. The packages can be written in R code, but for example also in C++ (with the rcpp-package). But at the beginning you should concentrate on the Rcode variant.

1.2 Why Packages?

To use foreign packages, makes your workflow a lot easier and faster. Because for recurring tasks its not necessary to reinvent the wheel again and again. There are more than 19.000 packages on CRAN. For nearly every common Problem a package exist, for example to load or visualize data. Packages form also interfaces to other software and their file formats (foreign, caffe, RQGIS,...), databases (RODBC, RPostgreSQL,...), other programming languages (Rcpp, RPython,...) or webservices (Rfacebook, RGoogleAnalytics,...).

1.3 Where can i find packages?

To get an overview, which packages are already available its good to use the website *https://www.rdocumentation.org/*. It list nearly every Package which official exists. You can also look at *https://cran.r-project.org* the official R Website or BioConductor a website for Biology Content (*http://bioconductor.org/*). Of course there are also countless packages on GitHub (*https://github.com/*).

1.4 Some important R-Commands to work with packages

```r
# Installing Packages:
install.packages(”xyz”)
install.packages(c(”xyz”, ”123”))

# check and update Packages:
old.packages(”xyz”)
update.packages(”xyz”)

# remove Packages:
remove.packages(”xyz”)
```

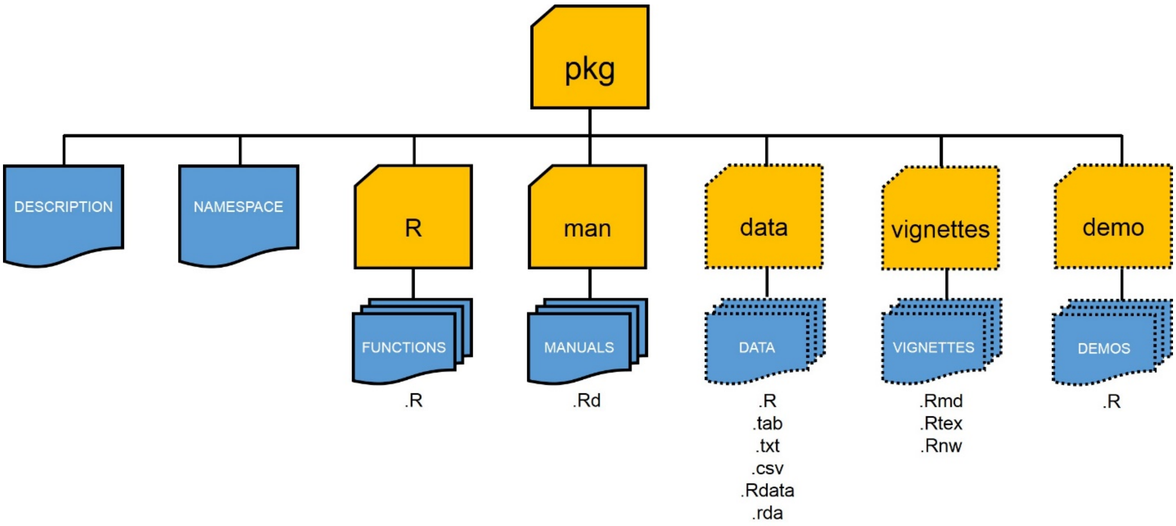
There is one big problem: each repository (CRAN, BioConductor,..) has its own way to install a package from them. To simplify this process you can use the package ”*devtools*”. But for this you might also need to install *Rtools*” for Windows, ”*Xcode*” for Mac or if you using Linux *r − devel*” for Linux. But then you can install packages directly from the repositorys with the following code:

```r
install_bioc() #from Bioconductor
install_cran() #from CRAN
install_github() #from GitHub
install_local() #from a local file
install_url() #from a URL
```

2 Making you own Pakage

2.1 What does a package consist of?

In this chapter we want to show how to create your own package. For this it is important to know how a package is built. If you create a new file to create an R-package, a folder structure is automatically created in the background. This folder structure is shown in the following figure. The DESCRIPTION-Files, the NAMESPACE, the FUNCTIONS and the MANUALS are mandatory. The rest, which is dotted in the graphic, is optional.



3 Related literature

Literatur

[text]