

R 语言编程概述：5. 编写 R 程序包

张金龙

2018 年 6 月 23 日

R 程序包是什么？

按照 Writing R Extension 文档所编写的一系列 R 函数、数据及其文档等组成的文件夹及其所有文件，或该文件夹的 gz 或 zip 压缩文件。

程序包有几种状态：

- ① 源代码
- ② 压缩的源代码包，多为 gz 格式
- ③ Windows 包
- ④ MacOS 包
- ⑤ 安装到计算机后的版本

为什么要写程序包？

- ① 为了让分析工具更系统
- ② 为了让每一个需要的函数都有良好的文档，大部分情况下，如果不写好文档，过一段时间自己都完全不记得了。
- ③ 为了共享分析工具
- ④ 可重复研究
- ⑤ 挑战黑科技...

核心包和扩展包

安装 R 运行平台后，默认只安装了用于计算、统计与绘图的基本程序包。包括：

base, compiler, datasets, graphics, grDevices,
grid, methods, parallel, splines, stats,
stats4, tcltk, tools, utils, boot,
cluster, foreign, MASS, Matrix, mgcv,
nlme, nnet, rpart, survival, boot,
class, cluster, codetools, foreign, KernSmooth,
lattice, MASS, Matrix, mgcv, nlme,
nnet, rpart, spatial, survival 等

用 `installed.packages()` 函数可查看已安装了哪些程序包

R 程序包的主要网站

- CRAN: 最大的 R 包集成网络
- Bioconductor: 主要用于生物信息学分析和数据分享
- Rforge: 主要用于 R 包开发, 用 svn 进行版本控制, 自动编译和检查程序包
- github: 主要用于 R 包开发, 用 git 进行版本控制, 只提供源代码

截止到 2018 年 6 月 20 日, CRAN 上共有 12630 个 R 程序包

<https://cran.r-project.org/web/packages/index.html>

CRAN Task Views 和安装某一类的程序包

CRAN Task Views (截止到 2018 年 6 月 20 日, 共 36 个 CRAN Task Views)
<https://cran.r-project.org/web/views/>

```
library(ctv)
ctv::install.views("Environmetrics")
ctv::install.views("Phylogenetics")
ctv::update.views("Econometrics")
```

Bioconductor version 3.7 (Release)

▼ Software (1562)

- ▶ AssayDomain (618)
- ▶ BiologicalQuestion (614)
- ▶ Infrastructure (341)
- ▶ ResearchField (469)
- ▶ StatisticalMethod (527)
- ▶ Technology (991)
- ▶ WorkflowStep (833)
- ▶ AnnotationData (919)
- ▶ ExperimentData (342)
- ▶ Workflow (21)

图 1: Bioconductor

R 程序包的基本结构

- DESCRIPTION 文件
- NAMESPACE 文件
- R 文件夹
- man 文件夹
- vignettes 文件夹，用于保存 Rmd 文档或者 LaTeX 文档
- data 文件夹
- inst/extdata 文件夹

Writing R extensions

<https://cran.r-project.org/doc/manuals/r-release/R-exts.html>

DESCRIPTION 文件 1

主要用于说明程序包的用途，作者 (contributor)，以及维护人 (maitainer)，程序包的依赖关系，许可证等。

格式为 DCF，为 the Debian Control Format 的缩写。

```
1 Package: picante
2 Type: Package
3 Title: Integrating Phylogenies and Ecology
4 Version: 1.7
5 Date: 2018-05-01
6 Author: Steven W. Kembel <steve.kembel@gmail.com>, David D. Ackerly
P. Blomberg <s.blomberg1@uq.edu.au>, Will K. Cornwell <cornwell@zool
<pdc@berkeley.edu>, Matthew R. Helmus <mrhelmus@wisc.edu>, Helene Mo
<morlon.helene@gmail.com>, Campbell O. Webb <cwebb@oeb.harvard.edu>
7 Maintainer: Steven W. Kembel <steve.kembel@gmail.com>
8 Depends: ape, vegan, nlme, methods
9 Suggests: brglm, circular, corpcor, quantreg
```

图 2: Picante 程序包的 Description 文件

DESCRIPTION 文件 2

过去，DESCRIPTION 文件一般是用 package.skeleton 函数模版生成的，开发者只需要修改其中的参数即可。

现在也用 devtools::create_description() 或者 devtools::create() 生成。

```
9 Suggests: brglm, circular, corpcor, quantreg
10 Description: Functions for phylocom integration, community analyses, and
11 phylogenetic and trait diversity, phylogenetic signal, estimation of phylogenetic
12 taxa, null models for community and phylogeny randomizations, and utility
13 input/output and phylogeny plotting. A full description of package functionality
14 provided by Kembel et al. (2010) <doi:10.1093/bioinformatics/btq166>.
15 License: GPL-2
16 NeedsCompilation: yes
17 Packaged: 2018-05-14 15:17:00 UTC; steve
18 Repository: CRAN
19 Date/Publication: 2018-05-14 18:39:57 UTC
```

图 3: Picante 程序包的 Description 文件

Description 文件举例

Package: anRpackage

Type: Package

Title: What the package does (short line)

Version: 1.0

Date: 2018-06-23

Author: Who wrote it

Maintainer: Who to complain to <yourfault@somewhere.net>

Description: More about what it does

(maybe more than one line)

License: What license is it under?

License 许可证

任何一个软件，需要开发者或者版权所有人授予使用权，用户才能使用。对于开源的 R 软件来说更是如此。

目前，R 程序包常用的许可证为：GPL-2，GPL-3，LGPL-2，LGPL-2.1，
LGPL-3，AGPL-3，Artistic-2.0，BSD_2_clause，BSD_3_clause，
MIT.

一般选用 GPL-2 或 GPL-3，也可选择 MIT。

GPL 系列许可证意味着别人可以使用和修改你的源代码，但是他的程序必须仍然为开源，
MIT 许可证则没有这种限制。

程序包的依赖关系

若所编写程序包的函数中使用了其他程序包中的函数或数据，就构成了程序包的依赖，此时需要在 DESCRIPTION 文件中做出相应的说明，以便在安装程序包时自动安装上有依赖关系的程序包。

picante: Integrating Phylogenies and Ecology

| | |
|-------------------|--|
| Version: | 1.7 |
| Depends: | ape , vegan , nlme , methods |
| Suggests: | brglm , circular , corpcor , quantreg |
| Published: | 2018-05-14 |
| Author: | Steven W. Kembel, David D. Ackerly, Simon P. Blomberg, Matthew R. Helmus, Helene Morlon, Campbell O. Voigt |
| Maintainer: | Steven W. Kembel <steve.kembel at gmail.com> |
| License: | GPL-2 |
| NeedsCompilation: | yes |
| Citation: | picante citation info |
| Materials: | README |

图 4：程序包的依赖关系

程序包的依赖关系: Depends, Imports 和 Suggests

常见有三种情况:

- **Depends**: 程序包使用了其他程序包所定义的类 class, 或者程序包必须在 R 的某一个版本之后才能正常工作。此时, 需要设定所依赖的程序包为 Depends, 以便用 library() 导入本程序包时, 所依赖的程序包自动 attach, 保证所有函数和数据可直接调用。
- **Imports**: 如果程序包的函数只使用了其他程序包的部分函数, 用 library() 读取该程序包后, 只需要将函数或者数据 import, 而无需全部导入。
- **Suggests**: 程序包只在函数的 Example 或者 vignettes 中使用了所依赖函数的内容, 则放入 Suggests。

Suggests 的程序包使用 install.packages() 时, 默认不安装。

要安装 suggests 的程序包, 使用以下命令:

```
install.packages("apackage", dependencies = TRUE)
```

NAMESPACE 文件：函数名冲突与覆盖

不同程序包常会出现相同的函数名，在用 `library()` 函数导入程序包后，用函数名调用的究竟是哪个函数？

NAMESPACE 就是为了解决这个问题。

```
> library(tidyverse)
-- Attaching packages --
v ggplot2 2.2.1     v purrr   0.2.5
v tibble   1.4.2     v dplyr    0.7.5
v tidyr    0.8.1     v stringr  1.3.1
v readr    1.1.1     v forcats 0.3.0
-- Conflicts --
x dplyr::filter()  masks stats::filter()
x dplyr::lag()     masks stats::lag()
> library(picante)
Loading required package: ape
Loading required package: vegan
Loading required package: permute
Loading required package: lattice
This is vegan 2.5-2
Loading required package: nlme

Attaching package: 'nlme'

The following object is masked from 'package:dplyr':
  collapse

> library(MASS)
Attaching package: 'MASS'

The following object is masked from 'package:dplyr':
  select
```

图 5: 函数冲突

NAMESPACE 文件

在编写 R 函数时，一些函数我们并不一定希望用户看到，而是作为 Utility Functions。如果将这些函数全部放到主函数当中，主函数将显得十分“臃肿”。

NAMESPACE 中可以控制 export 出哪些函数，只为 export 出的函数写帮助文档，避免了函数和程序的“臃肿”。

```
> library(lme4)
Loading required package: Matrix

Attaching package: 'Matrix'

The following object is masked from 'package:tidyverse':
  expand

Attaching package: 'lme4'

The following object is masked from 'package:nlme':
  lmList
```

图 6: 函数冲突

NAMESPACE 文件

常用关键词为：

- `import`: 导入依赖的整个程序包
- `importFrom`: 导入所依赖程序包的某几个函数
- `export`: 导出本程序包中哪些函数，即让用户可见
- `S3method`: S3 对象
- `exportClasses`: S4 对象
- `exportMethods`: S4 方法

picante 的 NAMESPACE 文件

```
1 useDynLib(picante, .registration = TRUE)
2
3 # Export all names
4 exportPattern(".")
5
6 # Import all packages listed as Imports or Depends
7 import(
8   ape,
9   vegan,
10  nlme
11 )
12
13 importFrom("grDevices", "rainbow")
14 importFrom("graphics", "abline", "axis", "legend", "lines", "par",
15            "plot", "segments", "text", "title")
16 importFrom("methods", "is")
17 importFrom("stats", "aggregate", "as.dist", "coef", "cophenetic",
18            "cor", "cor.test", "cov", "dbinom", "dist", "glm",
19            "model.matrix", "na.omit", "optim", "pt", "quantile",
20            "reorder", "rlnorm", "rnorm", "runif", "sd",
21            "weighted.mean")
22 importFrom("utils", "read.table", "write.table")
23
```

图 7: picante 的 NAMESPACE 文件

R 文件夹

用来保存 R 函数的源代码，一般来说，有两种组织方式：

- ① 每个函数独立保存为 R 文件，文件以函数名命名。适用于函数较少的情况。一般通过 `package.skeleton()` 即可做到。
- ② 若干函数保存到一个 R 文件，文件以类别命名。适用于函数较多，以及使用了 S3, S4 等面向对象编程的情形。

man 文件夹

- man 文件夹用于保存每个函数的帮助文件，扩展名为.Rd。
- 若函数在 NAMESPACE 文件中已经导出，则必须为该函数提供使用说明。Rd 文件与 LaTeX 的语法接近。
- R 对象帮助文件的模版可以通过 prompt() 函数生成。
- 用文本编辑器直接编辑 Rd 文件简单直接，但是函数在更新以后，如果参数发生变化，更新 Rd 文件会比较麻烦为此 Rd 文件可以借助 roxygen2 程序包生成。

Rd 文件示例

```
1 \name{pd}
2 \alias{pd}
3
4 \title{ Calculate Faith's Phylogenetic Diversity }
5 \description{
6   Calculate the sum of the total phylogenetic branch length
7 }
8 \usage{
9 pd(samp, tree, include.root=TRUE)
10 }
11
12 \arguments{
13   \item{samp}{ Community data matrix }
14   \item{tree}{ A phylo tree object }
15   \item{include.root}{ Should the root node be included? }
16 }
17
18 \value{
19   Returns a dataframe of the PD and species richness
20 }
21
```

图 8: picante::pd 函数的帮助文件

vignettes 文件夹

vignettes 主要是让用户从总体上更好地理解程序包中的函数是如何组织起来进行一系列复杂分析的，特别是要让用户熟悉数据分析的流程与原理等。

- 保存程序包 vignettes 源文件。
- 源文件必须为 Rnw 文件或者 Rmd 文件。
- .Rnw 文件为 Sweave 文档，会通过 Latex 编译为 PDF。
- .Rmd 文件会通过 knitr 和 rmarkdown 包转换为 html 文件。Rmd 文件也就是 R Markdown 文档。

R Markdown vignettes 文件

Vignettes 文件与普通 Rmarkdown 文档不同之处在于 YAML 文件头的差异。有固定格式

```
1. ---
2 title: "Tree Manipulation"
3 author: "Guangchuang Yu and Tommy Tsan-Yuk Lam\\
4
5         School of Public Health, The University of Hong Kong"
6 date: "`r Sys.Date()`"
7 bibliography: ggtree.bib
8 biblio-style: apalike
9 output:
10    prettydoc::html_pretty:
11        toc: true
12        theme: cayman
13        highlight: github
14 pdf_document:
15     toc: true
16 vignette: >
17   \%VignetteIndexEntry{03 Tree Manipulation}
18   \%VignetteEngine{knitr::rmarkdown}
19   \%usepackage[utf8]{inputenc}
20 ---
```

图 9: ggtree Vignettes 的 YAML

data 文件夹

- 如果想让用户用 `data()` 访问数据，需要将 Rda 二进制文件放在 data 文件夹中。
- Rda 文件内部多为 `data.frame`, `list` 或者 `vector` 等数据类型，以 `data.frame` 较为常见。
- Rda 文件除通过 `package.skeleton()` 函数生成之外，也可通过 `save()` 生成。

inst/extdata 文件夹和 src 文件夹

- inst/extdata 文件夹一般用来保存 rda 格式以外的数据类型，如制表符间隔的 txt、csv、xls、xlsx 文件等。但是由于 CRAN policy 的限制，用户已经不能在示例中直接读取这些文件。
- src 文件夹用于保存.C, .Cpp, .f 等文件。在 Windows 中，放在 src 文件夹中的 C、C++、Fortran 函数会自动编译为 dll 文件，供 R 函数调用。

Windows 下创建 R 程序包的工具

- Rcmd.exe: 位置 C:\Program Files\R\R-3.5.0\bin\i386, 通过命令行调用
- Rtools: 包含 gcc 等一系列工具, 用于编译.C, .cpp, .f 等文件
- MikTeX: 用于生成 PDF 版本的 Manual, 或 vignettes 等, 在检查中也会用到
- devtools 程序包: 主要提供创建、安装、检查程序包的函数, 以及从 github 安装程序包的函数等
- Rstudio: 提供建立 R 程序包的集成环境

Windows 下 Rcmd 的操作

从源代码创建 Linux 包

```
Rcmd build picante
```

创建 Windows 下的包

```
Rcmd INSTALL --build picante
```

检查程序包是否有问题

```
Rcmd check picante
```

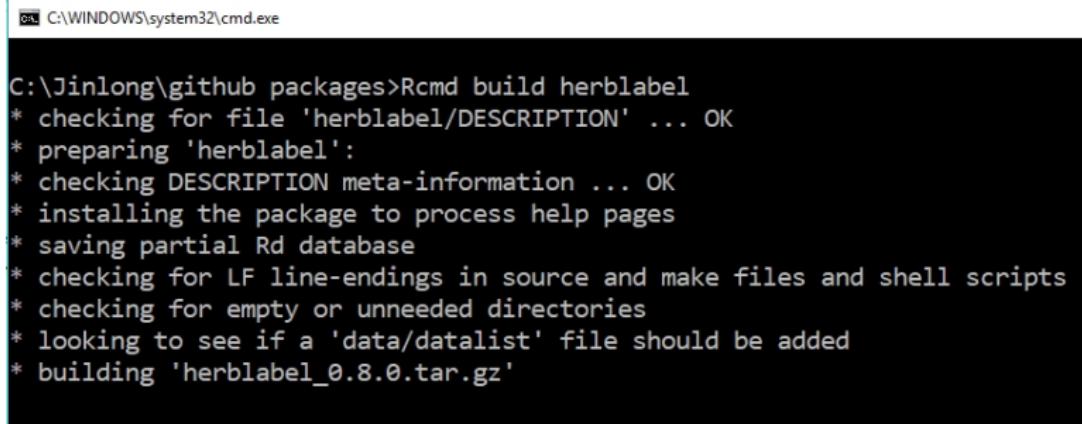
```
Rcmd check --as-cran picante
```

安装程序包

```
Rcmd INSTALL picante
```

一般将命令保存为.bat 文件，放在程序包源代码所在的文件夹下，之后双击.bat 文件即可运行。

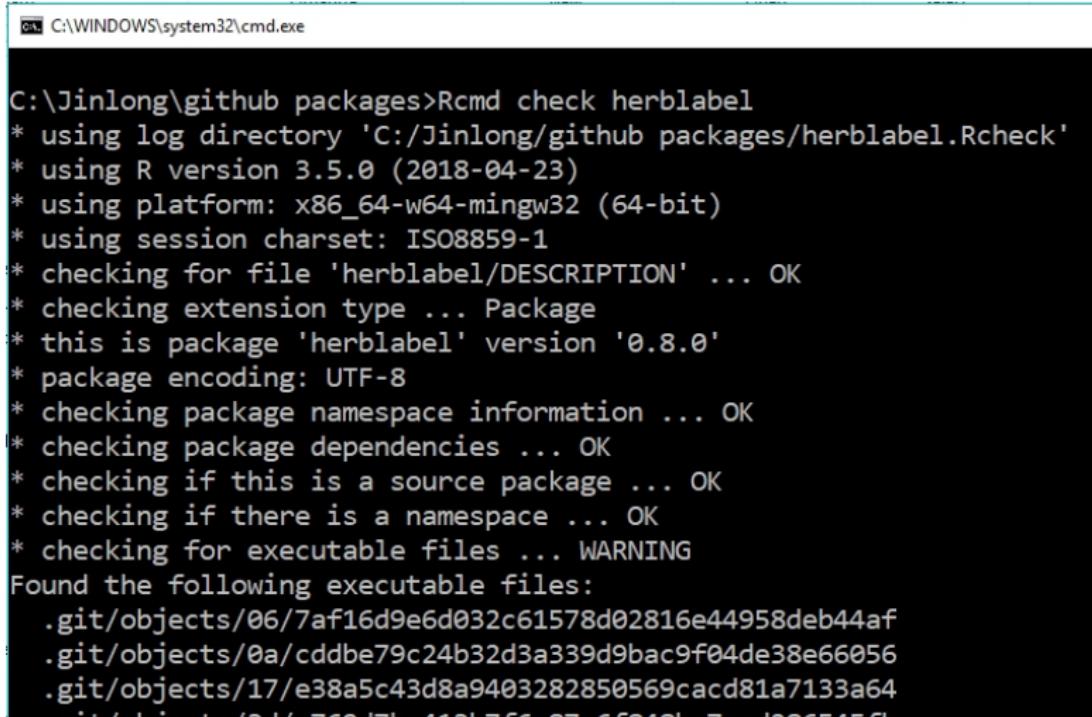
编译程序包



```
C:\Jinlong\github packages>Rcmd build herblabel
* checking for file 'herblabel/DESCRIPTION' ... OK
* preparing 'herblabel':
*   checking DESCRIPTION meta-information ... OK
*   installing the package to process help pages
*   saving partial Rd database
*   checking for LF line-endings in source and make files and shell scripts
*   checking for empty or unneeded directories
*   looking to see if a 'data/datalist' file should be added
*   building 'herblabel_0.8.0.tar.gz'
```

图 10: 通过 Rcmd 编译 herblabel 程序包

检查程序包



```
C:\Jinlong\github packages>Rcmd check herblabel
* using log directory 'C:/Jinlong/github packages/herblabel.Rcheck'
* using R version 3.5.0 (2018-04-23)
* using platform: x86_64-w64-mingw32 (64-bit)
* using session charset: ISO8859-1
* checking for file 'herblabel/DESCRIPTION' ... OK
* checking extension type ... Package
* this is package 'herblabel' version '0.8.0'
* package encoding: UTF-8
* checking package namespace information ... OK
* checking package dependencies ... OK
* checking if this is a source package ... OK
* checking if there is a namespace ... OK
* checking for executable files ... WARNING
Found the following executable files:
.git/objects/06/7af16d9e6d032c61578d02816e44958deb44af
.git/objects/0a/cddbe79c24b32d3a339d9bac9f04de38e66056
.git/objects/17/e38a5c43d8a9403282850569cacd81a7133a64
.git/objects/2d/760d7b413b756a87a65848ba7ad386545fb
```

图 11: 通过 Rcmd 检查 herblabel 程序包

过去的工作流程

- 用 `package.skeletons()` 创建程序包骨架
- 添加 R 函数到 R/ 文件夹下
- 修改 `NAMESPACE` 文件，指定导入和导出的函数
- 修改 `DESCRIPTION` 文件
- 手工修改 `Rd` 文件
- 用 `\LaTeX` `Sweave` 编写使用指南

主要参考: `Writing R Extensions, RShowDoc("R-exts")`

Wickham 推荐的工作流程

ggplot2 的作者 Wickham 推荐用 Rstudio 进行程序包开发 (<http://r-pkgs.had.co.nz/>)。

```
getwd()  
devtools::create("taxa")  
devtools::build()  
devtools::check()  
devtools::install()
```

roxygen2 使用要点

通过 `package.skeletons()` 函数生成的 Rd 文件与 R 函数在不同文件夹中，在 R 函数更新后，需要进一步手工更新，在函数很多时较为麻烦。

为此，Hadley Wickham 提出将 R 函数的文档信息保存在 R 脚本的注释中，使用 roxygen2 的关键词标注，以自动生成或者更新 Rd 文档。

这样做的好处在于，函数和帮助文档方便一起维护。

```
#'  
@details, @param, @return, @examples document the function  
@export 设定该函数是否为用户可见  
@import and @importFrom indicate (non-base) functions that are  
devtools::document() 生成Rd文件
```

测试 test: 检查程序的正确性与稳定性

- 对任何计算机程序来说，正确性是第一位的。没有经过严格测试的程序包，结果的可信度会大打折扣。
- 因为函数往往比较复杂，开发者要考虑在不同情形下，给出特定的结果，或者输出特定的错误提示。

为什么不用 R 帮助文件中的 Example 部分做检验？

因为：

- R example 中的代码必须是可正确运行的
- 用户体验：用户查看 example 一般是为了了解函数能做什么，而并不关心函数在不同情形下返回什么结果。

因此，相关的测试最好不要放在 Example 中。

testthat 或者 RUnit 程序包进行检验

进行测试的 R 程序包有 testthat 和 RUnit, 而 testthat 比较流行。

- Hadley Wickham. testthat: Get Started with Testing. *The R Journal*, vol. 3, no. 1, pp. 5–10, 2011
- Matthias Burger, Klaus Juenemann and Thomas Koenig (2018). RUnit: R Unit Test Framework. R package version 0.4.32.
<https://CRAN.R-project.org/package=RUnit>

testthat 程序包

进行测试的 R 脚本应放在源代码的 tests 文件夹下的 testthat 文件夹中..

- Write *unit tests* that validate the correctness of your functions. `usethis::use_testthat()`

```
$ tree
.
.
.
└── tests
    ├── testthat
    │   └── test_hi.R
    └── testthat.R
.
```

testthat 举例

多个类似的 test 放在一个.R 文件中，每个 R 文件均以 context 开头，指明要测试的内容。

后面用 expect_XXX 系列函数测试函数的返回值。

```
1 context('as.polytomy')
2
3 test_that('collapse tree to polytomy', {
4     file <- system.file("extdata/RAxML", "RAxML_bipart"
5     tree <- read.tree(file)
6     cutoff <- 70
7     tree2 <- as.polytomy(tree, 'node.label', function
8         expect_true(all(as.numeric(tree2$node.label) > 70,
9     })
10 })
```

图 12: ggtree 程序包的 test

testthat 的主要函数 1

| | |
|---|---|
| expect_condition | Expectation: does code produce output/message/warning/error? |
| expect_cpp_tests_pass | Test Compiled Code in a Package |
| expect_equal | Expectation: is the object equal to a value? |
| expect_equal_to_reference | Expectations: is the output or the value equal to a known good value? |
| expect_equivalent | Expectation: is the object equal to a value? |
| expect_error | Expectation: does code produce output/message/warning/error? |
| expect_false | Expectation: is the object true/false? |
| expect_gt | Expectation: is returned value less or greater than specified value? |
| expect_gte | Expectation: is returned value less or greater than specified value? |
| expect_identical | Expectation: is the object equal to a value? |
| expect_is | Expectation: does the object inherit from a S3 or S4 class, or a base type? |
| expect_known_hash | Expectations: is the output or the value equal to a known good value? |
| expect_known_output | Expectations: is the output or the value equal to a known good value? |
| expect_known_value | Expectations: is the output or the value equal to a known good value? |
| expect_length | Expectation: does a vector have the specified length? |
| expect_less_than | Expectation: is returned value less or greater than specified value? |
| expect_lt | Expectation: is returned value less or greater than specified value? |
| expect_lte | Expectation: is returned value less or greater than specified value? |

图 13: testthat 的主要函数

testthat 的主要函数 2

| | |
|---------------------------------|---|
| <code>expect_length</code> | Expectation: does a vector have the specified length? |
| <code>expect_less_than</code> | Expectation: is returned value less or greater than specified value? |
| <code>expect_lt</code> | Expectation: is returned value less or greater than specified value? |
| <code>expect_lte</code> | Expectation: is returned value less or greater than specified value? |
| <code>expect_match</code> | Expectation: does string match a regular expression? |
| <code>expect_message</code> | Expectation: does code produce output/message/warning/error? |
| <code>expect_more_than</code> | Expectation: is returned value less or greater than specified value? |
| <code>expect_named</code> | Expectation: does object have names? |
| <code>expect_null</code> | Expectation: does the object inherit from a S3 or S4 class, or a base type? |
| <code>expect_output</code> | Expectation: does code produce output/message/warning/error? |
| <code>expect_output_file</code> | Expectations: is the output or the value equal to a known good value? |
| <code>expect_reference</code> | Expectation: is the object equal to a value? |
| <code>expect_s3_class</code> | Expectation: does the object inherit from a S3 or S4 class, or a base type? |
| <code>expect_s4_class</code> | Expectation: does the object inherit from a S3 or S4 class, or a base type? |
| <code>expect_sequal</code> | Expectation: is the object equal to a value? |
| <code>expect_silent</code> | Expectation: does code produce output/message/warning/error? |
| <code>expect_true</code> | Expectation: is the object true/false? |
| <code>expect_type</code> | Expectation: does the object inherit from a S3 or S4 class, or a base type? |
| <code>expect_warning</code> | Expectation: does code produce output/message/warning/error? |

图 14: testthat 的主要函数

编写 R 程序包的建议

- ① 编写的函数要简单，每一个函数实现一个功能。
- ② 任何时候都要处理好异常，以提醒用户输入正确的数据类型以及取值范围。
- ③ 每个导出的函数都应有 example，example 中的数据要小，以免运行 example 的时间过长
- ④ 名称简洁，看到名称最好就能知道该程序包属于什么领域。
- ⑤ 要有完备的 vignettes 介绍分析流程，注意事项，算法等。

练习与答疑