

R包开发

温灿红

中国科学技术大学管理学院

R包开发简介

为什么要自己开发R包？

在R中，可分享代码的基本单位是包。软件包是可以把代码、数据、文档和测试整合在一起，很容易与人分享。

- **沟通**：在R包的公共发布网站CRAN有上万个包可用，这也是R为什么成功的原因之一。
- **跨平台/跨系统使用**：R包可使得代码具有可移植性，而不需要考虑平台或系统的问题。
- **和合作者分享代码**：解释文档等可帮助你与合作者高效分享代码。
- **可复现性**：通过版本的控制使得结果可复现。

软件要求（一）

- 首先要安装了R和RStudio，并安装了以下R包：

```
install.packages(c("devtools", "roxygen2", "knitr"))
```

- 需要一个C编译器和几个命令行工具
 - Windows：下载并安装Rtools
 - Mac：安装Xcode（可在App Store免费获取）或者Xcode命令行工具（<https://developer.apple.com/xcode/>）
 - Linux：R开发工具，如在Ubuntu上的r-base-dev包

软件要求（二）

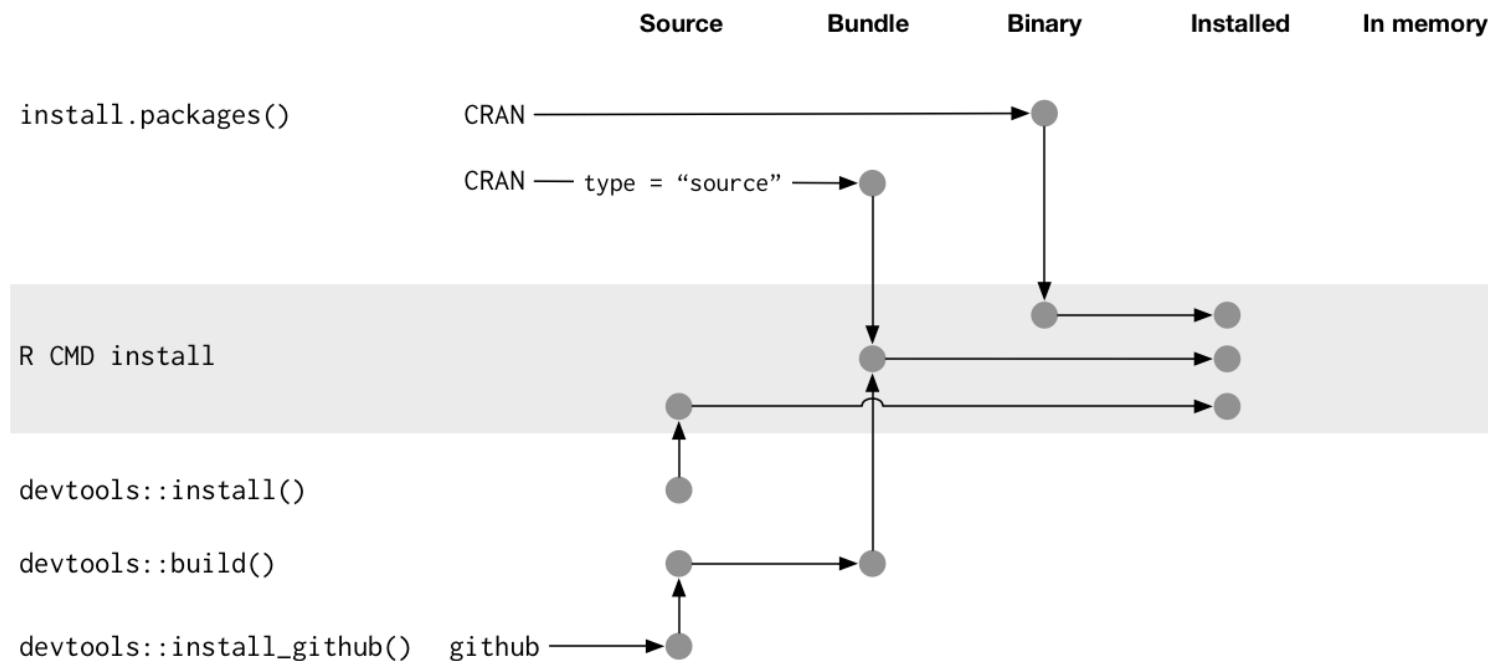
- 可以通过运行以下代码来检查是否所有的工具都安装妥当

```
library(devtools)  
has_devel()
```

R包的生命周期（一）

- 要掌握包的开发，特别是分发一个包给别人时，理解包生命周期是很有帮助的。
 1. 源码包 (source) 是处在开发版本的包，只是包含R/目录，DESCRIPTION等组件的一个目录。
 2. 压缩包 (Bundled) 是把源码包打包压缩成的，扩展名为.tar.gz，调用 `devtools::build()` 来创建。
 3. 二进制包 (Binary) 是可以不通过编译就可以安装的文件，依赖于平台，如Mac下的扩展名为.tgz，Windows下的扩展名为.zip。调用 `devtools::build(binary = TRUE)` 来创建。
 4. 已安装的包 (Installed) 是安装并解压到一个包库的二进制包
 5. 内存中的包 (In-memory)

R包的生命周期 (二)



什么是库？

- 库是一个包含已安装包的目录。
- 一般来说，每个人是找有两个库：
 - 一个用来放已经安装的包
 - 一个用来放R安装时自带的包（如base, stats等）
- 可通过`.libPaths()` 查看当前正在使用的库

```
.libPaths()
```

```
## [1] "C:/Users/wench/AppData/Local/R/win-library/4.2"
```

```
## [2] "C:/Program Files/R/R-4.2.3/library"
```

- 当使用`library(pkg)`加载包时，R会查找`.libPaths()` 的每个路径，看是否存在一个pkg目录。如果不存在，则返回一个错误信息。
- `library()`和`require()`之间的主要区别是：在包找不到的时候，`library()` 返回一个错误，而`require()`打印一个消息并返回FALSE。

从零创建R包

创建R包

- 下面我们从零开始学习如何创建一个R包。
- 假设我们现在需要创建一个R包，其目的是实现任意函数的蒙特卡洛积分，并进行过相应的绘图工作。我们需要以下两个函数：
 - `mc_int(x_range, fun, B, seed = 123)`: 主函数，用于实现函数 `fun` 的蒙特卡洛积分，其中 `x_range` 指定积分的上下限，`B` 是抽样的个数，`seed` 是随机种子数。
 - `plot.MCI(x, ...)`: 绘图函数，是一个关于 `mc_int()` 输出 `x` 的类函数，用于绘制函数和对应的积分面积。

创建R包：主函数

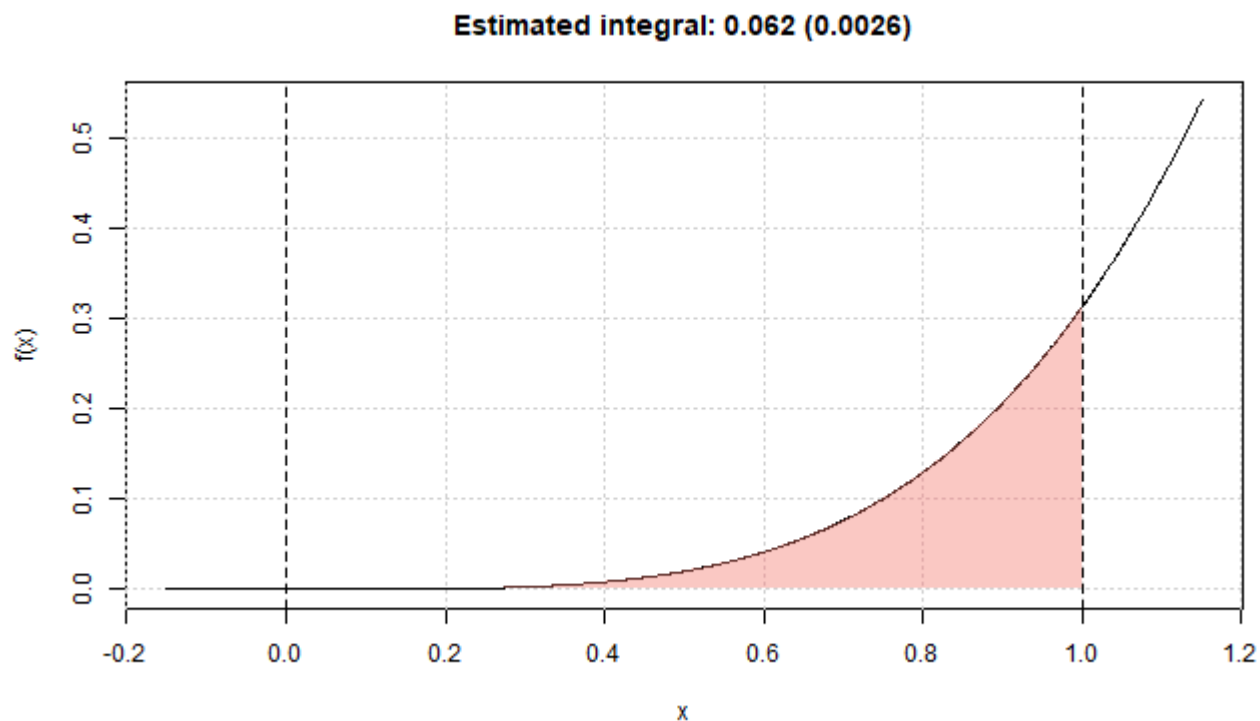
```
mc_int = function(x_range, fun, B, seed = 123){  
  # 检查形参x_range是否输入正确  
  if (length(x_range) != 2 || x_range[1] >= x_range[2]){  
    stop("x_range is incorrectly specified")  
  }  
  
  # 检查积分的函数名是否输入正确  
  if (class(fun) != "character"){  
    stop("fun is incorrectly specified and should be a character")  
  }  
  
  # 检查积分的函数是否存在并能执行  
  x = mean(x_range)  
  test_fun = try(eval(parse(text = fun)), silent = TRUE)  
  if (class(test_fun) == "try-error"){  
    stop("fun cannot be evaluated")  
  }  
  
  # 检查B是否输入正确  
  if (B < 1){  
    error("B is incorrectly specified")  
  }  
  
  set.seed(seed)
```

创建R包：绘图函数

```
plot.MCI = function(x, ...){  
  obj = x  
  x_range = obj$x_range  
  fun = obj$fun  
  
  Delta = diff(x_range)  
  x_range_graph = c(x_range[2] - 1.15*Delta, x_range[1] + 1.15*Delta)  
  x = seq(from = x_range_graph[1], to = x_range_graph[2], length.out  
  f_x = eval(parse(text = fun))  
  plot(NA, xlim = range(x), ylim = range(f_x), xlab = "x", ylab = "f  
  grid()  
  title(paste("Estimated integral: ", round(obj$I,4), " (", round(sq  
  lines(x, f_x)  
  x = seq(from = x_range[1], to = x_range[2], length.out = 10^3)  
  f_x = eval(parse(text = fun))  
  cols = hcl(h = seq(15, 375, length = 3), l = 65, c = 100, alpha = 0  
  polygon(c(x, rev(x)), c(rep(0, length(x)), rev(f_x)), border = NA,  
  abline(v = x_range[1], lty = 2)  
  abline(v = x_range[2], lty = 2)  
}
```

创建R包：示例函数

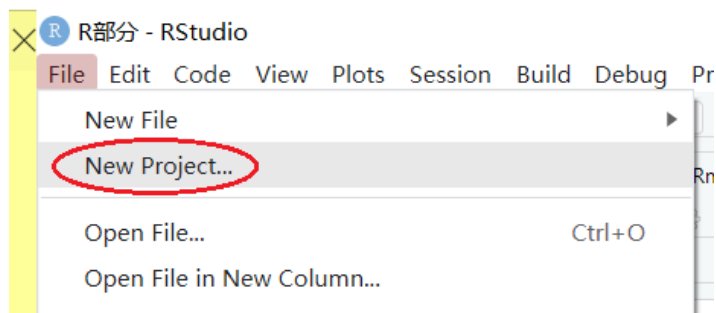
```
obj = mc_int(x_range = c(0,1), fun = "x^2*sin(x^2/pi)", B = 10^3)  
plot(obj)
```



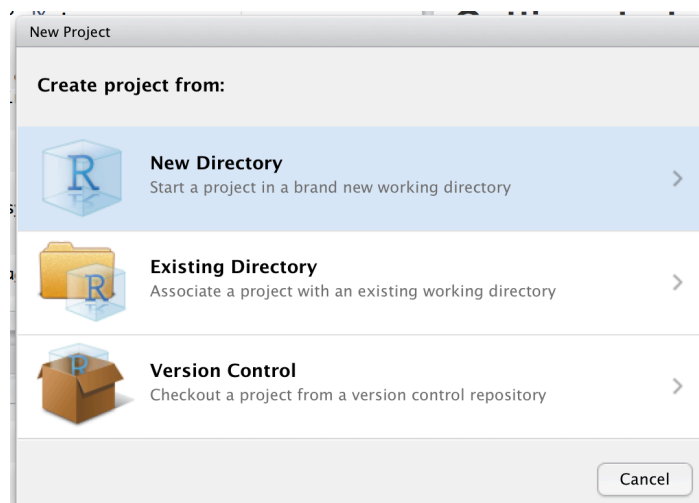
第一步：创建一个空的R包

第一步：创建一个空的R包（一）

1.单击 File | New Project

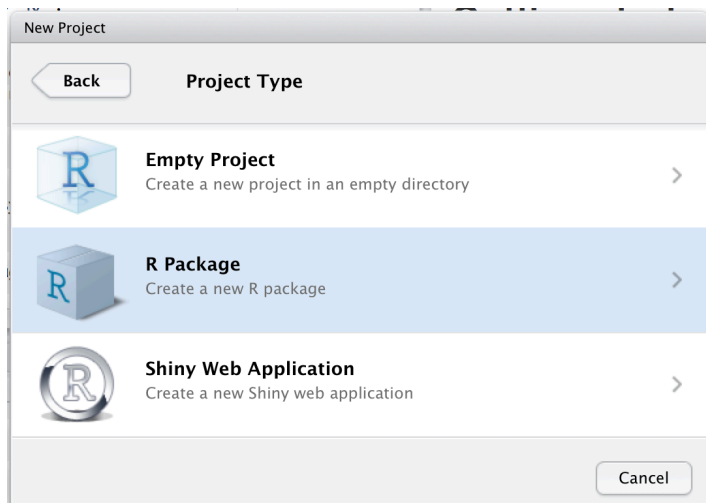


2.选择 New Directory

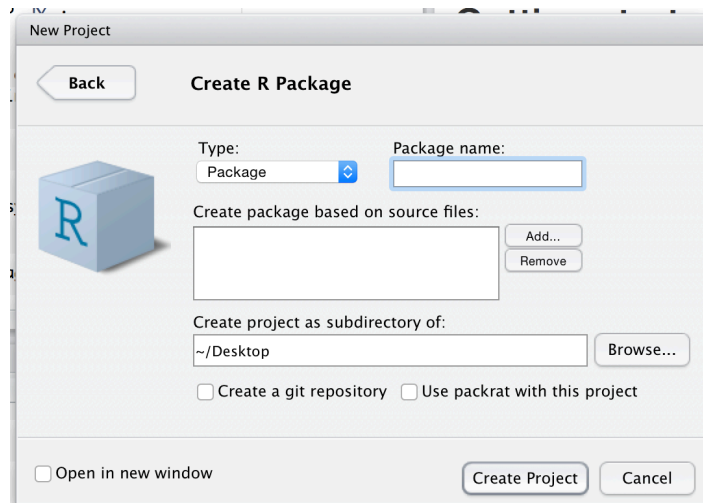


第一步：创建一个空的R包（二）

3.选择 R Package



4.给出包名，然后单击 Create Project



创建R包完成

- 按照上述步骤，我们就得到了一个最小的可用包，有以下几个组成部分：

1. R/目录

2. 描述文档 DESCRIPTION

3. 命名空间文件 NAMESPACE

4. man/目录

- 还有一个RStudio项目文件pkgname.Rproj，这将使得你开发包变得更加便利。
- 接下来，删除以下文档：
 - NAMESPACE：自动产生
 - man/hello.Rd：自动产生
 - R/hello.R：修改R代码，因此先行删除

R包命名规则

- 要求：名称只能包含字母，数字和点号(.)，必须以一个字母开始，且不能以点号结束。
- 命名的策略：
 - 选择一个容易被搜索到的独特的名字
 - 避免同时使用大写和小写字母，否则会让包名难以输入，更难以记住。
 - 找到一个和研究问题相关的词语，修改它使它变得特别：
 - `plyr` 是 `apply` 函数族的推广
 - `knitr` = `knit`(编织)+`r`
 - 使用缩略词
 - `Rcpp` = `R`+ `C++`(`C` plus plus)
 - `ggplot` = `g`ammer of `g`raphics `p`lots
 - 增加一个额外的`r`
 - `stringr` 提供字符串的工具

第二步：编辑描述文档

描述文档

- DESCRIPTION文件的作用是存储包中重要的元数据，一个最小描述文件应有：

Package: demo

Type: Package

Title: What the Package Does (Title Case)

Version: 0.1.0

Author: Who wrote it

Maintainer: The package maintainer yourself@somewhere.net

Description: More about what it does (maybe more than one line).

License: What license is it under?

Encoding: UTF-8

LazyData: true

RoxygenNote: 7.3.1

描述文档的呈现形式

BeSS: Best Subset Selection in Linear, Logistic and CoxPH Models

An implementation of best subset selection in generalized linear model and Cox proportional hazard model via the primal dual active set algorithm proposed by Wen, C., Zhang, A., Quan, S. and Wang, X. (2017) <[arXiv:1709.06254](https://arxiv.org/abs/1709.06254)>. The algorithm formulates coefficient parameters and residuals as primal and dual variables and utilizes efficient active set selection strategies based on the complementarity of the primal and dual variables.

Version: 1.0.6
Depends: R ($\geq 3.2.0$)
Imports: [Rcpp](#) ($\geq 0.12.8$), [Matrix](#) ($\geq 1.2-6$), [glmnet](#), [survival](#)
LinkingTo: [Rcpp](#), [RcppEigen](#)
Published: 2019-02-21
Author: Canhong Wen, Aijun Zhang, Shijie Quan, Xueqin Wang
Maintainer: Canhong Wen <wencanhong@gmail.com>
License: [GPL-3](#)
NeedsCompilation: yes
CRAN checks: [BeSS results](#)

Downloads:

Reference manual: [BeSS.pdf](#)
Package source: [BeSS 1.0.6.tar.gz](#)
Windows binaries: r-devel: [BeSS 1.0.6.zip](#), r-devel-gcc8: [BeSS 1.0.6.zip](#), r-release: [BeSS 1.0.6.zip](#), r-oldrel: [BeSS 1.0.6.zip](#)
OS X binaries: r-release: [BeSS 1.0.6.tgz](#), r-oldrel: [BeSS 1.0.6.tgz](#)
Old sources: [BeSS archive](#)

Linking:

描述文档最重要的域：依赖

- 可以通过Imports或Suggests来刻画新建的软件包所依赖的包，两者的区别在于依赖的程度。
 - Imports中的包是必须安装的，在别人安装你的包的时候会同时把Imports中的包安装。
 - Suggests是建议安装的，不是必须的。
- 定义方式如下：

```
Imports:  
  stats,  
  Matrix(>= 1.2-6)
```

```
Suggests:  
  stats,  
  Matrix
```

标题和描述

- 标题和描述域都是用来描述包是做什么的，区别仅在于长度。
 - 标题(Title)是包的一行描述，纯文本（无标记），并在大小写上遵循标题风格，不要超过65个字符。
 - 描述(Description)比标题更详细，可使用多个句子，但只限于一段。用四个空格缩进后续的行。

```
Title: Monte Carlo Integration
```

```
Description: An implementation of Monte Carlo integration within  
              a finite interval.
```

作者

- 简单形式:

```
Author: Canhong Wen, Who Who  
Maintainer: Canhong Wen <wench@ustc.edu.cn>
```

- 专业形式:

```
Authors@R: c(  
  person("Canhong", "Wen", email = "wench@ustc.edu.cn", role = c("a  
  person("Who", "Who", role = "aut")  
)
```

- 三个字母的代码来指定角色 role, 有四个重要的角色:
 - cre: 创建者或维护者。
 - aut: 作者, 对包有重大贡献的人。
 - ctb: 贡献者, 做出较小贡献的人, 如提供了一些补丁。
 - cph: 版权所有人。版权是作者以外的人, 通常是一个公司。

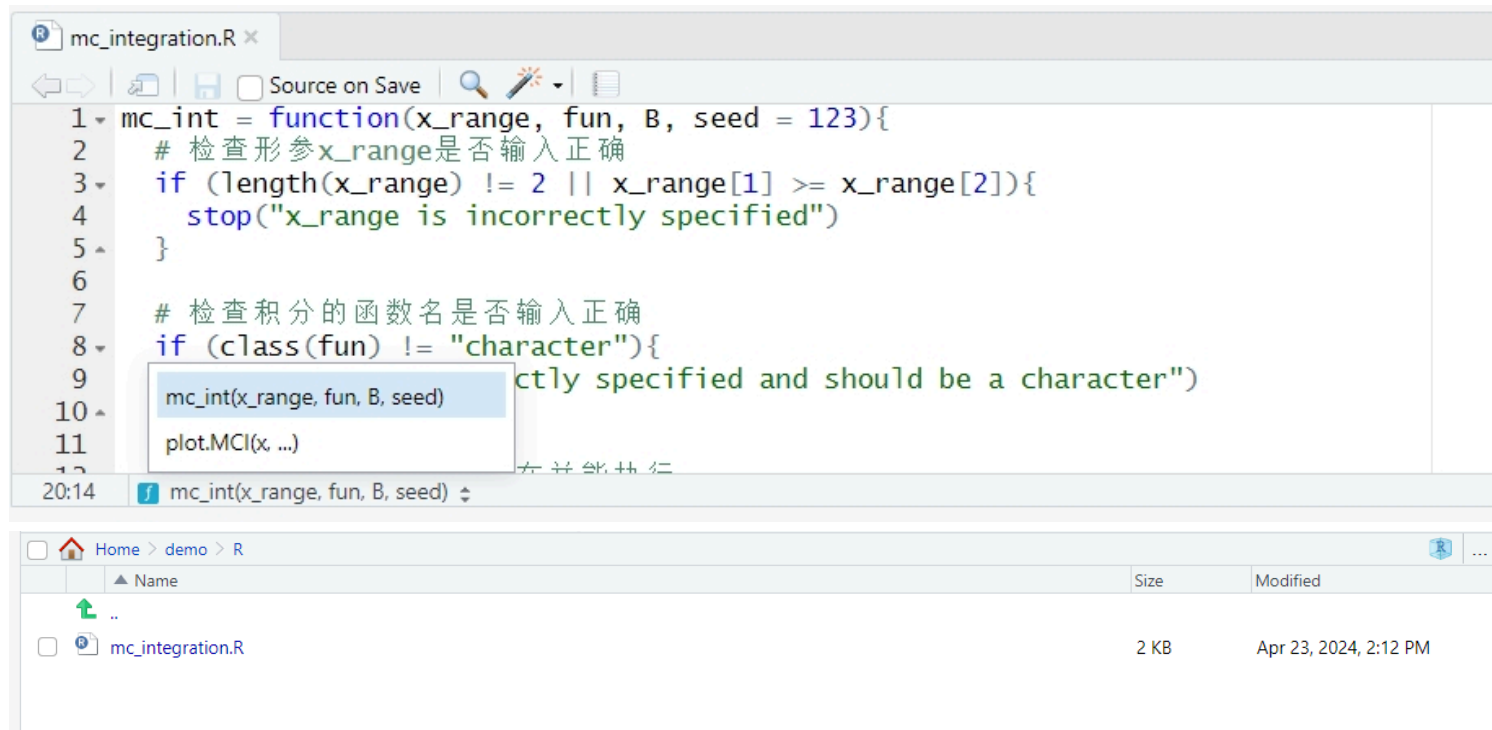
版本

- 发布的R软件包的版本包括三个数字，\<主版本号>.\<次版本号>.\<补丁版本号>，如 1.9.2。
- 开发中的R软件包有第四个数字：开发版本，从9000开始。如一般而言包的第一个版本是0.0.0.9000.
- 包的版本号随着后续版本的发布而增加。

第三步：将R代码保存到R/目录

R代码

- 之前定义了两个函数，将它们保存到.R脚本文件中，并将其保存在R/目录下。如下图所示：



The screenshot displays the RStudio environment. The top pane shows the script `mc_integration.R` with the following code:

```
1 mc_int = function(x_range, fun, B, seed = 123){  
2   # 检查形参x_range是否输入正确  
3   if (length(x_range) != 2 || x_range[1] >= x_range[2]){  
4     stop("x_range is incorrectly specified")  
5   }  
6  
7   # 检查积分的函数名是否输入正确  
8   if (class(fun) != "character"){  
9     stop("fun is incorrectly specified and should be a character")  
10  }  
11  plot.MCI(x_range, fun, B, seed)  
12 }
```

The bottom pane shows a file explorer view of the `demo` directory, listing the file `mc_integration.R` with a size of 2 KB and a modification date of April 23, 2024, at 2:12 PM.

.R脚本文件的命名规则

- R 代码通常保存成.R脚本文件。在起名的时候，应尽量使得文件的名称反映其功能，如：

```
# 好的命名  
fit_models.R  
utility_functions.R
```

```
# 不好的命名  
foo.r  
stuff.r
```

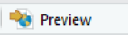
- 在命名文件中对象（变量和函数）时，不要把R自带的函数覆盖掉。一般而言，变量名应该为名词，函数名为动词。如：

```
# 好的命名  
day_one  
day_1
```

```
# 不好的命名  
first_day_of_the_month  
djm1  
T <- FALSE  
mean <- function(x) sum(x)
```

第四步：添加对象文档

对象文档

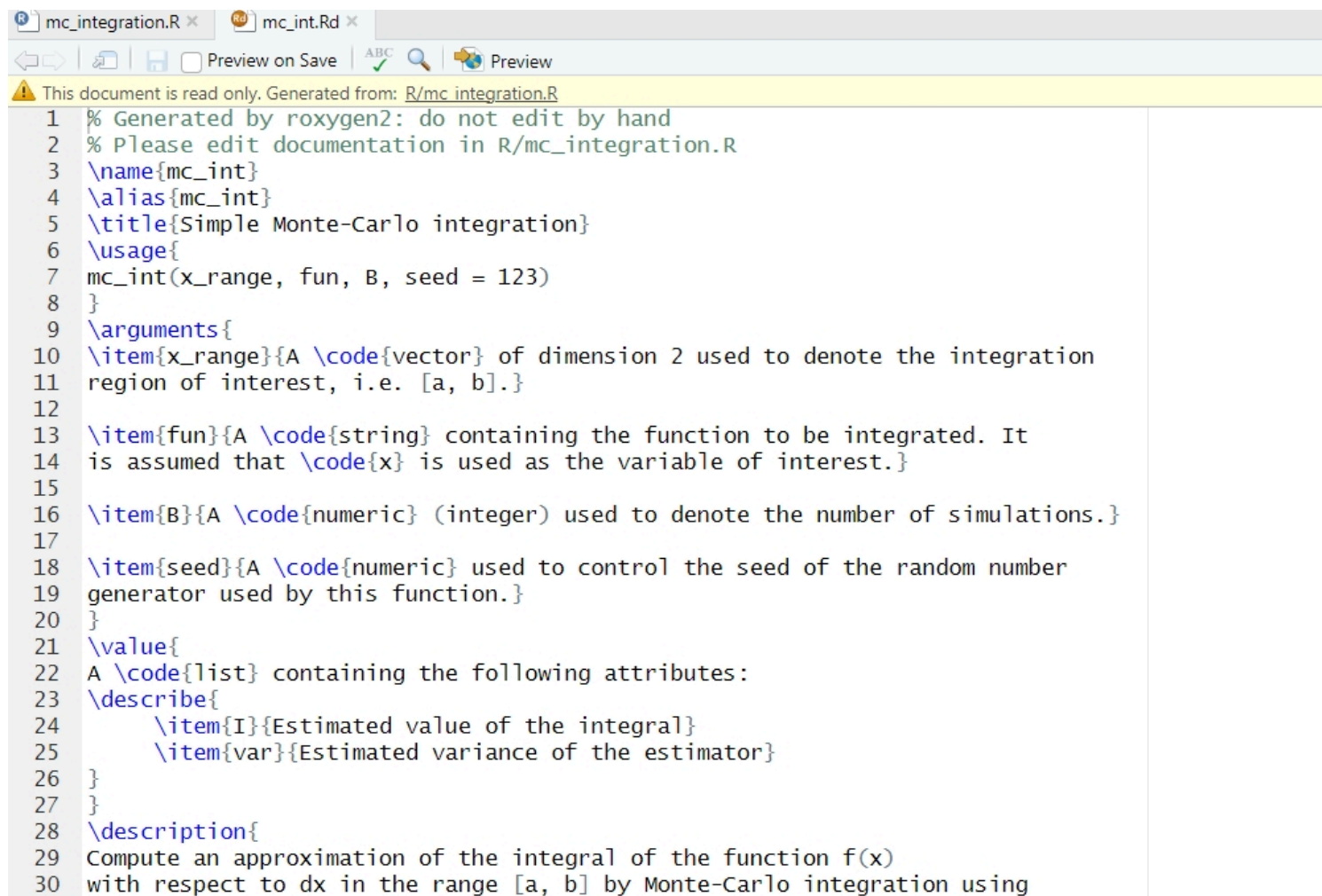
- 文档是一个**好包**最重要的组成部分。
- 文档通常保存在man/目录下，以 .Rd 作为后缀，语法大致基于LaTeX，最后会被编译成HTML、纯文本和PDF格式的帮助用户以供查看。
- 可手动编辑.Rd文档，但更推荐使用roxygen2包，因为可自动管理NAMESPACE和DESCRIPTION。
- 大致的工作流程如下：
 1. 在 .R 文档中添加roxygen注释；
 2. 运行 `devtools::document()` 或在RSudio中按Ctrl/Cmd + Shift + D 转化roxygen注释为.Rd 文档；
 3. 利用? 预览文档，或点击  Preview 来查看；
 4. 修改注释，重复上面的步骤，直到文档看起来是你想要的样子。

添加对象文档（一）

```
#' @title Simple Monte-Carlo integration
#
#' @description Compute an approximation of the integral of the function
#' with respect to dx in the range [a, b] by Monte-Carlo integration
#' using uniform sampling.
#' @param x_range A \code{vector} of dimension 2 used to denote the
#' region of interest, i.e. [a, b].
#' @param fun A \code{string} containing the function to be integrated.
#' is assumed that \code{x} is used as the variable of interest.
#' @param B A \code{numeric} (integer) used to denote the number of samples.
#' @param seed A \code{numeric} used to control the seed of the random
#' generator used by this function.
#' @return A \code{list} containing the following attributes:
#' \describe{
#'   \item{I}{Estimated value of the integral}
#'   \item{var}{Estimated variance of the estimator}
#' }
#' @author Canhong Wen
#' @importFrom stats runif
#' @export
#' @examples
#' mc_int(x_range = c(0,1), fun = "x^2", B = 10^5)
#' mc_int(x_range = c(0,1), fun = "x^2*sin(x^2/pi)", B = 10^5)
```

添加对象文档（二）

- 运行 `devtools::document()` 自动生成的 `mc_int.Rd` 文档



```
1 % Generated by roxygen2: do not edit by hand
2 % Please edit documentation in R/mc_integration.R
3 \name{mc_int}
4 \alias{mc_int}
5 \title{Simple Monte-Carlo integration}
6 \usage{
7   mc_int(x_range, fun, B, seed = 123)
8 }
9 \arguments{
10  \item{x_range}{A \code{vector} of dimension 2 used to denote the integration
11    region of interest, i.e. [a, b].}
12
13  \item{fun}{A \code{string} containing the function to be integrated. It
14    is assumed that \code{x} is used as the variable of interest.}
15
16  \item{B}{A \code{numeric} (integer) used to denote the number of simulations.}
17
18  \item{seed}{A \code{numeric} used to control the seed of the random number
19    generator used by this function.}
20 }
21 \value{
22  A \code{list} containing the following attributes:
23  \describe{
24    \item{I}{Estimated value of the integral}
25    \item{var}{Estimated variance of the estimator}
26  }
27 }
28 \description{
29  Compute an approximation of the integral of the function  $f(x)$ 
30  with respect to  $dx$  in the range  $[a, b]$  by Monte-Carlo integration using
```


添加对象文档（三）

- 运行?mc_int可查看帮助文档页面

mc_int.Rd ▾ Find in Topic

mc_int {demo} (preview) R Documentation

Simple Monte-Carlo integration

Description

Compute an approximation of the integral of the function $f(x)$ with respect to dx in the range $[a, b]$ by Monte-Carlo integration using uniform sampling.

Usage

```
mc_int(x_range, fun, B, seed = 123)
```

Arguments

x_range A vector of dimension 2 used to denote the integration region of interest, i.e. $[a, b]$.

fun A string containing the function to be integrated. It is assumed that x is used as the variable of interest.

B A numeric (integer) used to denote the number of simulations.

seed A numeric used to control the seed of the random number generator used by this function.

Value

A list containing the following attributes:

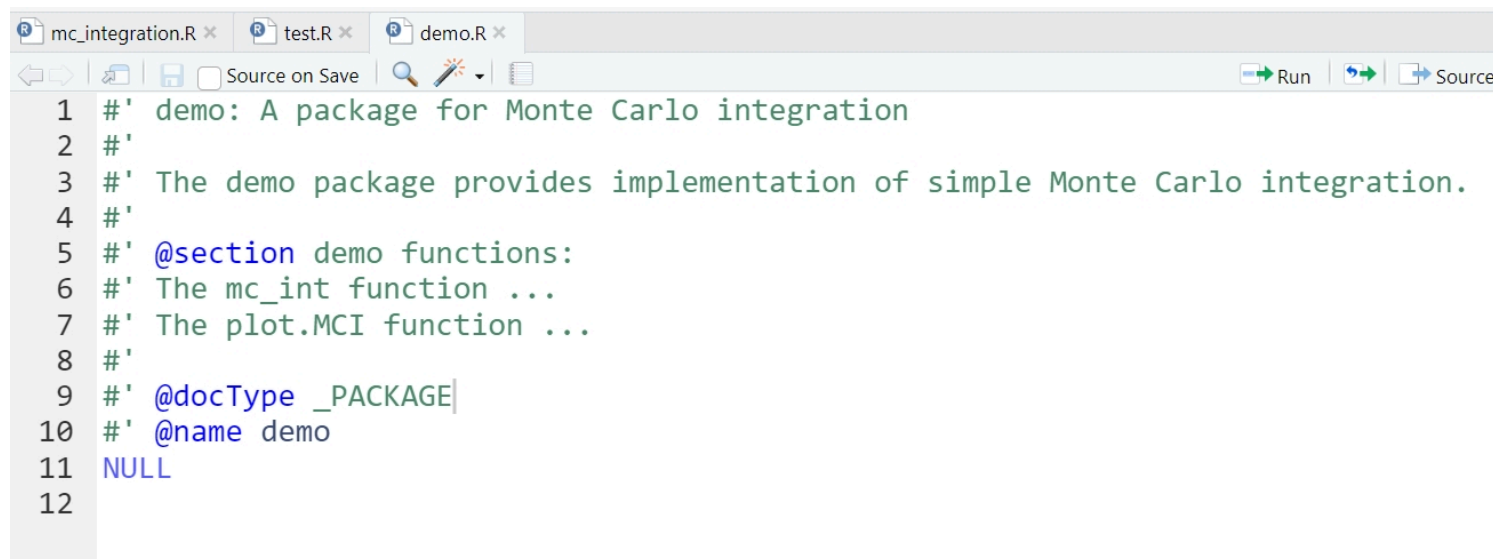
I	Estimated value of the integral
var	Estimated variance of the estimator

Roxygen 注释

- Roxygen 注释总是 '#' 开始，放在函数的前面。
 - 第一句是文档的标题，应该占一行（80个字符以内），句首字母大写，并以句号结束。
 - 第二段是描述，用以简要说明函数的功能。
 - 第三段以及随后的段落进行细节的描述，通常可以分解成标签。
- 大部分的函数文档有三个标签，包括
 - **@param** 描述函数的输入或参数，简要描述参数的类型，用途。所有的参数都必须需要提供文档。
 - **@examples** 提供了可执行的R代码，演示如何使用该函数。`\dontrun{}`可以让你在例子中包括无需运行的代码。
 - **@return** 描述函数的返回值。

为R包添加对象文档（一）

- 也可以为包撰写一个帮助页面，用以描述包中的最重要的组成部分。添加方式为NULL函数手动添加标签**@docType**和**@name**。



```
1 #' demo: A package for Monte Carlo integration
2 #'
3 #' The demo package provides implementation of simple Monte Carlo integration.
4 #'
5 #' @section demo functions:
6 #' The mc_int function ...
7 #' The plot.MCI function ...
8 #'
9 #' @docType _PACKAGE
10 #' @name demo
11 NULL
12
```

为R包添加对象文档（二）

- 也可以为包撰写一个帮助页面，用以描述包中的最重要的组成部分。添加方式为NULL函数手动添加标签**@docType**和**@name**。

R: demo: A package for Monte Carlo integration ▾ Find in Topic

demo {demo}

R Documentatio

demo: A package for Monte Carlo integration

Description

The demo package provides implementation of simple Monte Carlo integration.

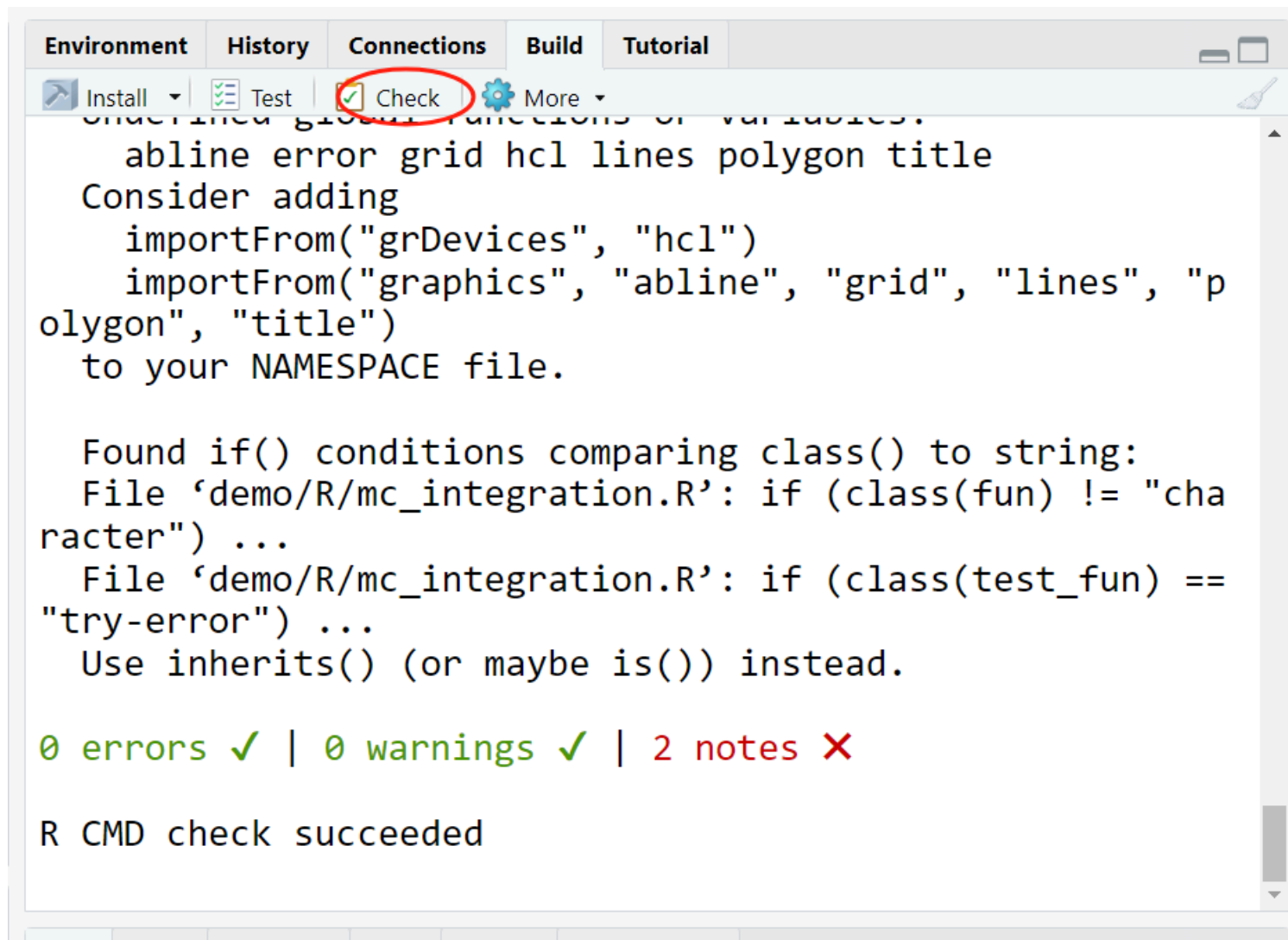
demo functions

The mc_int function ... The plot.MCI function ...

[Package *demo* version 0.1.0 [Index](#)]

第五步：测试R包

检查R包



The screenshot shows the RStudio interface with the 'Check' button in the top toolbar highlighted by a red circle. The main editor window displays the following content:

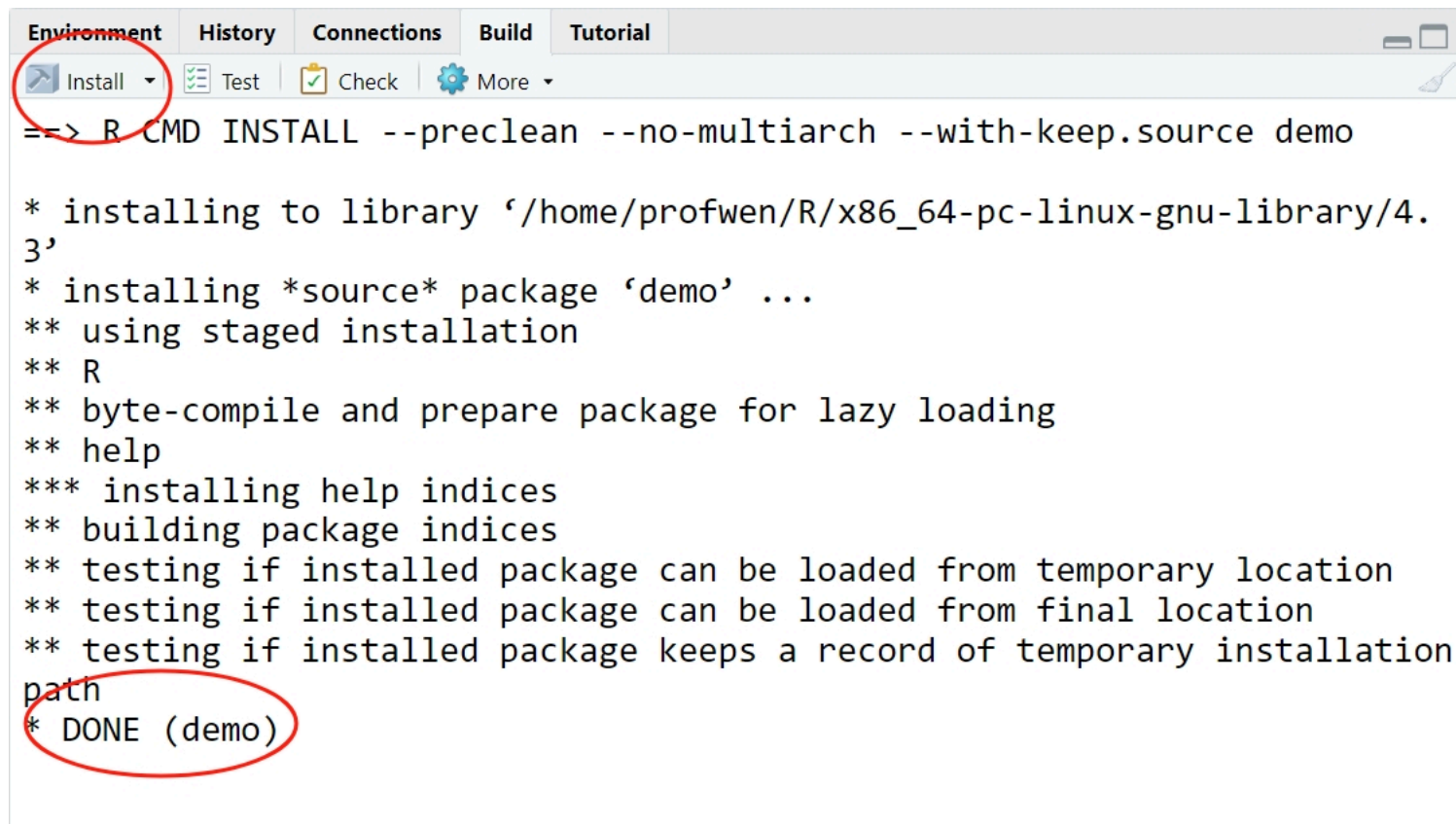
```
undefined global functions or variables:
  abline error grid hcl lines polygon title
Consider adding
  importFrom("grDevices", "hcl")
  importFrom("graphics", "abline", "grid", "lines", "p
olygon", "title")
to your NAMESPACE file.

Found if() conditions comparing class() to string:
File 'demo/R/mc_integration.R': if (class(fun) != "cha
racter") ...
File 'demo/R/mc_integration.R': if (class(test_fun) ==
"try-error") ...
Use inherits() (or maybe is()) instead.
```

0 errors ✓ | 0 warnings ✓ | 2 notes ✗

R CMD check succeeded

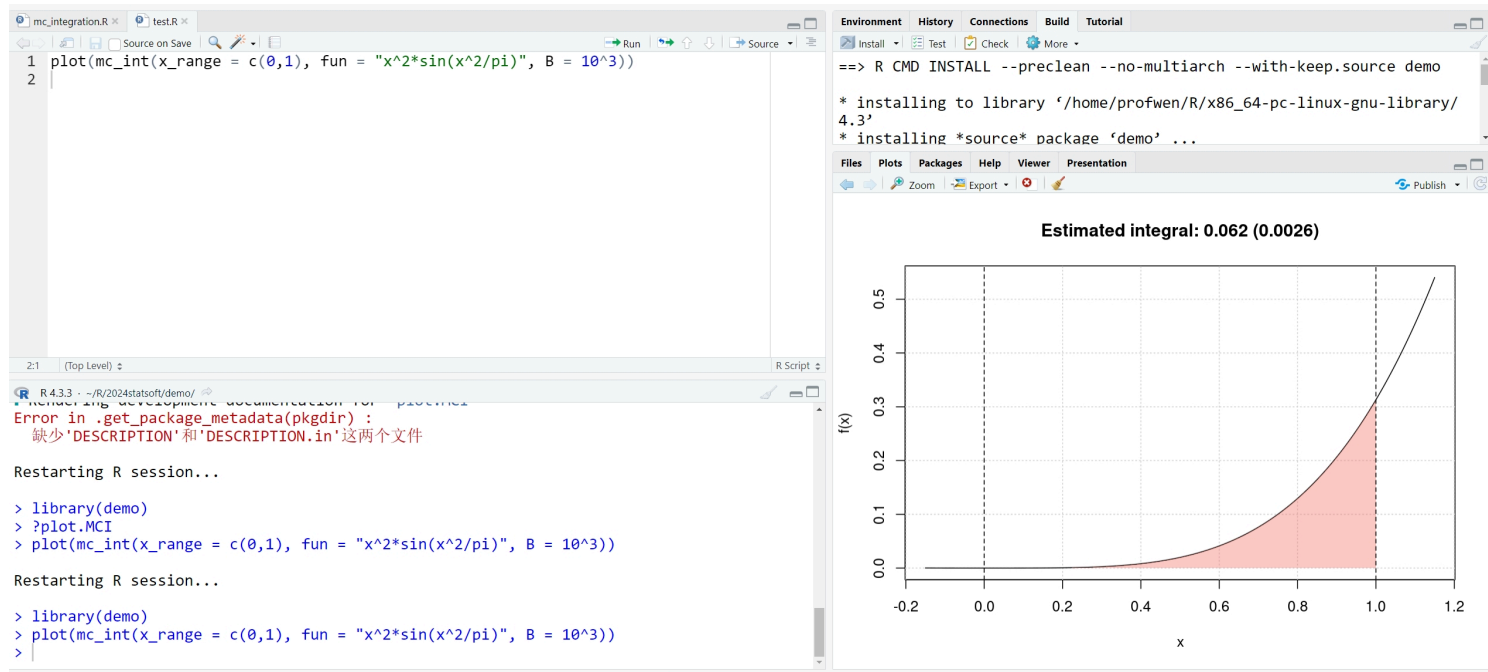
安装R包



```
Environment History Connections Build Tutorial
Install Test Check More
==> R CMD INSTALL --preclean --no-multiarch --with-keep.source demo

* installing to library '/home/profwen/R/x86_64-pc-linux-gnu-library/4.
3'
* installing *source* package 'demo' ...
** using staged installation
** R
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded from temporary location
** testing if installed package can be loaded from final location
** testing if installed package keeps a record of temporary installation
path
* DONE (demo)
```

测试R包是否正确



数据

数据

- 如果你的R包需要一些测试的数据，这时候需要新建文件夹data/。
- R通常会将数据保存在文件夹 data/中，并以 .RData 作为后缀，通过 `data(YourDataName)` 即可导入数据。
- 如果DESCRIPTION 文档中包含了 `LazyData: true`，则数据集只有等到你调用的时候才会读入内存。

数据的帮助文档

```
#' Flights data
#
# On-time data for all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013.
#
#' @source RITA, Bureau of transportation statistics,
#'   <https://www.transtats.bts.gov/DL\_SelectFields.asp?Table\_ID=236>
#' @format Data frame with columns
#' \describe{
#'   \item{year, month, day}{Date of departure.}
#'   \item{dep_time, arr_time}{Actual departure and arrival times (format HHMM or HMM)}
#'   ...
#' }
"flights"

#' @importFrom tibble tibble
NULL
|
```

- 在数据中有两个标签用来描述数据的：
 - **@format** 描述数据集。对于数据框，应包括一个描述每个变量的定义列表。
 - **@source** 提供数据来源的细节，通常是一个`\url{}`。
- 永远不要**@export** 一个数据集。

谢谢