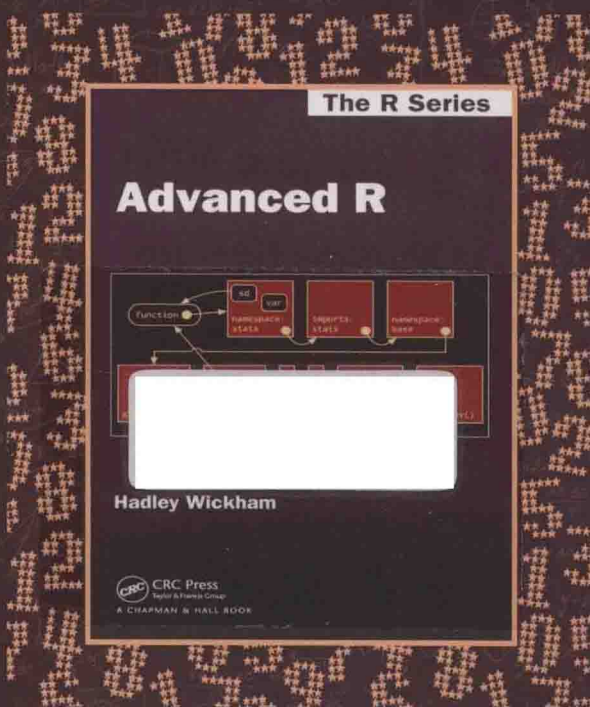


高级R语言编程指南

[美] 哈德利·威克汉姆 (Hadley Wickham) 著

李洪成 段力辉 何占军 译



ADVANCED R



机械工业出版社
China Machine Press

ADVANCED R

高级R语言编程指南

[美] 哈德利·威克汉姆 (Hadley Wickham) 著

李洪成 段力辉 何占军 译



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

高级 R 语言编程指南 / (美) 哈德利·威克汉姆 (Hadley Wickham) 著; 李洪成, 段力辉, 何占军译. —北京: 机械工业出版社, 2016.6

(数据科学与工程丛书)

书名原文: Advanced R

ISBN 978-7-111-54067-0

I. 高… II. ①哈… ②李… ③段… ④何… III. 程序语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2016) 第 130804 号

本书版权登记号: 图字: 01-2015-3694

Advanced R by Hadley Wickham (978-1-4665-8696-3).

Copyright © 2015 by Taylor & Francis Group, LLC.

Authorized translation from the English language edition published by CRC Press, part of Taylor & Francis Group LLC. All rights reserved.

China Machine Press is authorized to publish and distribute exclusively the Chinese (Simplified Characters) language edition. This edition is authorized for sale in the People's Republic of China only (excluding Hong Kong, Macao SAR and Taiwan). No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

Copies of this book sold without a Taylor & Francis sticker on the cover are unauthorized and illegal.

本书原版由 Taylor & Francis 出版集团旗下 CRC 出版公司出版, 并经授权翻译出版。版权所有, 侵权必究。

本书中文简体字翻译版授权由机械工业出版社独家出版并限在中国大陆地区销售。未经出版者书面许可, 不得以任何方式复制或抄袭本书的任何内容。

本书封面贴有 Taylor & Francis 公司防伪标签, 无标签者不得销售。

本书从 R 语言的基础知识入手, 深入而详细地介绍了 R 语言及其编程技术。本书共分 4 部分: R 语言基础、函数式编程、R 语言计算和 R 性能分析。第一部分详细介绍 R 的基础知识, 包括数据结构、子集选取、常用函数与数据结构、编程风格指南、函数、面向对象编程、环境、程序调试和防御性编程等。第二部分介绍函数式编程、泛函、函数运算符等。第三部分介绍非标准求值、表达式、领域特定语言 (HTML 和 LaTeX) 等。第四部分介绍 R 的性能、代码调优、内存管理、高性能计算和 C 语言编程接口等。

本书适合 R 的初学者和具有一定编程经验的 R 用户, 他们都能从书中找到对自己有用的内容。

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 盛思源

印 刷: 三河市宏图印务有限公司

开 本: 185mm×260mm 1/16

书 号: ISBN 978-7-111-54067-0

责任校对: 殷 虹

版 次: 2016 年 6 月第 1 版第 1 次印刷

印 张: 19.5

定 价: 79.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

购书热线: (010) 68326294 88379649 68995259

投稿热线: (010) 88379604

读者信箱: hzjsj@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

推荐序

根据调查，R 语言现在是高级分析、统计学、数据可视化和预测模型开发领域最流行的工具。在 [Statistics.com](https://www.statistics.com) 在线课程中，R 语言方面的课程都有大量的需求，这些课程包括本书作者 Hadley Wickham 开发的 ggplot2、数据挖掘、统计模型、地图、空间分析和贝叶斯统计等。R 语言流行的一个原因是，巨大的 R 社区贡献了数千个适用于各种应用的 R 添加包。

R 在统计学家和数据挖掘人员中的流行，奠定了它在编程语言中的独特地位。最初，用户应用 R 进行统计模型拟合、数据可视化或者其他的统计任务。对这些用户而言，R 是一个功能强大的、免费的、基于命令行的统计软件。至少，一开始他们可能不需要开发高效的、长度太大的复杂代码。这些用户随着应用 R 的范围变广、应用加深，把 R 作为一门编程语言来进行最佳实践的需求就变得迫切了。

本书是进一步学习 R 语言和进行标准编程实践一个很好的指南，它与应用 R 进行数据分析密切相关。该书系统地讲解了数据结构、子集选取、函数、泛函和面向对象编程。那些被 R 的速度慢而困扰的用户可能喜欢本书有关内存应用、代码优化和高性能计算部分的内容。

越多的 R 用户遵从本书开始几章列出的原则，在 R 社区中就会看到更多的高水平的 R 代码。



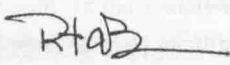
美国统计教育学院，[Statistics.com](https://www.statistics.com) 在线课程网站总裁和创始人 Peter Bruce

According to some surveys, R is now the most popular tool for advanced analytics, statistics, visualization and predictive model development. At [Statistics.com](https://www.statistics.com), courses in R are in high demand, including the ggplot2 course that Hadley Wickham developed, as well as courses in data mining, statistical modeling, mapping, spatial analysis, Bayesian statistics, and more. One reason for this is the huge global R community, which has contributed thousands of R packages for all kinds of purposes.

R's popularity among statisticians and data miners has given it a unique position among programming languages. Most of its users originally came to R with a goal of fitting a statistical

model, producing a visualization, or performing some other statistical task. For these users, R is essentially a powerful, free command-line statistical software program. They probably did not need to create code that was lengthy, complex, or especially fast, at least at first. With time, though, as the general use of R broadened and deepened, the need to treat R as a programming language, and observe “best practices,” grew.

This book provides a guide for these users who need to step back and learn the nitty-gritty of standard programming practices, as it they pertain to the use of R for data analysis. It provides a systematic treatment of data structures, notation, subsetting, functions, and functional and object-oriented programming. Those frustrated with R’s slowness will like the sections on memory use, code optimization and high performance computing. The more R-users who follow the principles outlined in the coming chapters, the better the overall level of code we will see in the R community.



Peter Bruce

President and Founder

The Institute for Statistics Education at Statistics.com

译者序

随着大数据的概念变得越来越流行，对数据的探索、分析和预测将成为大数据分析领域的基本技能之一。作为探索和分析数据的基本工具，数据分析软件包是必不可少的。R 软件作为功能强大且开源的数据分析工具，在数据分析领域获得了越来越多用户的青睐。目前，市场上出版了大量的与 R 语言有关的书籍，这些书籍基本可以分为两类：一类是通过 R 语言介绍某个主题或者课程；另一类是 R 软件或者 R 语言入门性质的介绍。目前尚没有一本深入而详细地介绍 R 语言与编程的书籍。本书恰好填补了这方面的空白。

本书从 R 语言的基础知识入手，深入介绍 R 函数式编程、R 语言的面向对象特性、程序调试、代码优化和性能调优。同时，本书也介绍了 R 语言如何与 HTML 和 LaTeX 语言相结合的技术，介绍了高性能计算以及 C 语言编程接口。

本书作者 Hadley Wickham 是 R 语言专家，他具有 10 年以上应用 R 语言的实践经验，他编写了许多高质量的 R 添加包，例如 ggplot2、plyr、reshape2 等，这些都是在 R 社区广泛使用的添加包。通过本书的学习，一方面可以更深入地了解 R 语言编程的核心知识，另一方面也可以在某种程度上了解这样一位众所周知的 R 语言专家所编写的 R 添加包。

本书共分 4 部分：R 语言基础、函数式编程、R 语言计算和 R 性能分析。第一部分详细介绍 R 的基础知识，包括数据结构、子集选取、常用函数与数据结构、编程风格指南、函数、面向对象编程、环境、程序调试和防御性编程；第二部分介绍函数式编程、泛函、函数运算符；第三部分介绍非标准求值、表达式、领域特定语言（HTML 和 LaTeX）；第四部分介绍 R 的性能、代码调优、内存管理、高性能计算和 C 语言编程接口。

本书是学习 R 语言难得的手册。不管是初学者还是具有一定编程经验的 R 用户，都可以从本书获益。初学者可以先从第一部分入手，然后根据需要逐步学习后面的部分。熟练的 R 用户可以从自己感兴趣的内容入手。

本书的翻译得到了国家自然科学基金（项目编号 71461005）和广西高校数据分析与计算重点实验室的资助。在本书的翻译过程中，得到了明永玲编辑的大力支持和帮助。本书责任编辑盛思源老师具有丰富的经验，为本书的出版付出了大量的劳动。这里对她们的支持和帮助表示衷心的感谢。本书的翻译工作主要由李洪成、段力辉共同完成，何占军和吴立明协助翻译了本书的部分内容。

由于时间和水平所限，难免会有不当之处，希望同行和读者多加指正。

目 录

推荐序	2.4.4 特殊列	19
译者序	2.4.5 练习	19
第1章 简介	2.5 答案	19
1.1 本书的目标读者	第3章 子集选取	21
1.2 通过本书你可以学到什么	3.1 数据类型	22
1.3 元技术	3.1.1 原子向量	22
1.4 推荐阅读	3.1.2 列表	23
1.5 获取帮助	3.1.3 矩阵和数组	23
1.6 致谢	3.1.4 数据框	24
1.7 约定	3.1.5 S3 对象	25
1.8 声明	3.1.6 S4 对象	25
	3.1.7 练习	25
	3.2 子集选取运算符	26
	3.2.1 简化与保留	26
	3.2.2 \$	27
	3.2.3 缺失/超出索引边界 (越界引用)	28
	3.2.4 练习	28
	3.3 子集选取与赋值	29
	3.4 应用	30
	3.4.1 查询表(字符子集选取)	30
	3.4.2 人工比对与合并(整数子集 选取)	30
	3.4.3 随机样本/自助法(整数子集 选取)	31
	3.4.4 排序(整数子集选取)	31
第一部分 基础知识		
第2章 数据结构		8
2.1 向量		9
2.1.1 原子向量		9
2.1.2 列表		11
2.1.3 练习		12
2.2 属性		12
2.2.1 因子		13
2.2.2 练习		15
2.3 矩阵和数组		15
2.4 数据框		17
2.4.1 数据框构建		17
2.4.2 类型判断与强制转换		18
2.4.3 合并数据框		18

3.4.5 展开重复记录 (整数子集 选取)	32	6.2.1 名字屏蔽	47
3.4.6 剔除数据框中某些列 (字符 子集选取)	33	6.2.2 函数与变量	48
3.4.7 根据条件选取行 (逻辑子集 选取)	33	6.2.3 重新开始	48
3.4.8 布尔代数与集合 (逻辑和 整数子集选取)	34	6.2.4 动态查找	49
3.4.9 练习	35	6.2.5 练习	50
3.5 答案	35	6.3 每个运算都是一次函数调用	50
第 4 章 常用函数与数据结构	36	6.4 函数参数	51
4.1 基础函数	36	6.4.1 函数调用	52
4.2 常见数据结构	37	6.4.2 使用参数列表来调用函数	53
4.3 统计函数	38	6.4.3 默认参数和缺失参数	53
4.4 使用 R	39	6.4.4 惰性求值	54
4.5 I/O 函数	39	6.4.5 ... 参数	56
第 5 章 R 编程风格指南	40	6.4.6 练习	57
5.1 符号和名字	40	6.5 特殊调用	57
5.1.1 文件名	40	6.5.1 中缀函数	57
5.1.2 对象名	40	6.5.2 替换函数	58
5.2 语法	41	6.5.3 练习	59
5.2.1 空格	41	6.6 返回值	59
5.2.2 大括号	42	6.6.1 退出时	61
5.2.3 行的长度	42	6.6.2 练习	62
5.2.4 缩进	42	6.7 答案	62
5.2.5 赋值	43	第 7 章 面向对象编程指南	64
5.3 结构	43	7.1 基础类型	65
第 6 章 函数	44	7.2 S3	66
6.1 函数组成部分	45	7.2.1 认识对象、泛型函数和方法	66
6.1.1 原函数	45	7.2.2 定义类和创建对象	67
6.1.2 练习	46	7.2.3 创建新方法和泛型函数	69
6.2 词法作用域	46	7.2.4 方法分派	69
		7.2.5 练习	71
		7.3 S4	71
		7.3.1 识别对象、泛型函数和方法	72
		7.3.2 定义类并创建对象	73
		7.3.3 创建新方法和泛型函数	74
		7.3.4 方法分派	74

7.3.5 练习	75
7.4 RC	75
7.4.1 定义类和创建对象	75
7.4.2 识别类和方法	77
7.4.3 方法分派	77
7.4.4 练习	77
7.5 选择一个系统	77
7.6 答案	78

第 8 章 环境 79

8.1 环境基础	79
8.2 环境递归	83
8.3 函数环境	85
8.3.1 封闭环境	85
8.3.2 绑定环境	86
8.3.3 执行环境	87
8.3.4 调用环境	88
8.3.5 练习	90
8.4 绑定名字和数值	90
8.5 显式环境	92
8.5.1 避免复制	93
8.5.2 软件包状态	93
8.5.3 模拟 hashmap	93
8.6 答案	94

第 9 章 调试、条件处理和防御性编程 95

9.1 调试技巧	96
9.2 调试工具	97
9.2.1 确定调用顺序	98
9.2.2 查看错误	99
9.2.3 查看任意代码	100
9.2.4 调用栈: <code>traceback()</code> 、 <code>where</code> 和 <code>recover()</code>	100
9.2.5 其他类型的故障	101

9.3 条件处理	102
9.3.1 使用 <code>try</code> 来忽略错误	102
9.3.2 使用 <code>tryCatch()</code> 处理 条件	103
9.3.3 <code>withCallingHandlers()</code>	105
9.3.4 自定义信号类	106
9.3.5 练习	107
9.4 防御性编程	107
9.5 答案	109

第二部分 函数式编程

第 10 章 函数式编程 112

10.1 动机	112
10.2 匿名函数	116
10.3 闭包	117
10.3.1 函数工厂	119
10.3.2 可变状态	119
10.3.3 练习	120
10.4 函数列表	120
10.4.1 将函数列表移到全局 环境中	122
10.4.2 练习	123
10.5 案例研究: 数值积分	124

第 11 章 泛函 127

11.1 第一个泛函: <code>lapply()</code>	128
11.1.1 循环模式	129
11.1.2 练习	130
11.2 for 循环泛函: <code>lapply()</code> 的相似 函数	131
11.2.1 向量输出: <code>sapply</code> 和 <code>vapply</code>	131
11.2.2 多重输入: <code>Map</code> (和 <code>mapply</code>)	133

11.2.3	滚动计算	134
11.2.4	并行化	135
11.2.5	练习	136
11.3	操作矩阵和数据框	137
11.3.1	矩阵和数组运算	137
11.3.2	组应用	138
11.3.3	plyr 添加包	139
11.3.4	练习	140
11.4	列表操作	140
11.4.1	Reduce()	140
11.4.2	判断泛函	141
11.4.3	练习	141
11.5	数学泛函	142
11.6	应该保留的循环	143
11.6.1	原位修改	143
11.6.2	递归关系	144
11.6.3	while 循环	144
11.7	创建一个函数系列	145

第 12 章 函数运算符 149

12.1	行为函数运算符	150
12.1.1	缓存	152
12.1.2	捕获函数调用	153
12.1.3	惰性	155
12.1.4	练习	155
12.2	输出函数运算符	156
12.2.1	简单修饰	156
12.2.2	改变函数的输出	157
12.2.3	练习	158
12.3	输入函数运算符	159
12.3.1	预填充函数参数：局部 函数应用	159
12.3.2	改变输入类型	159
12.3.3	练习	160
12.4	组合函数运算符	161

12.4.1	函数复合	161
12.4.2	逻辑判断和布尔代数	163
12.4.3	练习	163

第三部分 语言计算

第 13 章 非标准计算 166

13.1	表达式获取	167
13.2	在子集中进行非标准计算	168
13.3	作用域问题	171
13.4	从其他函数调用	173
13.5	替换	175
13.5.1	为替换提供应急方案	177
13.5.2	捕获未计算的表达式	177
13.5.3	练习	178
13.6	非标准计算的缺点	178

第 14 章 表达式 180

14.1	表达式的结构	180
14.2	名字	183
14.3	调用	184
14.3.1	修改调用	185
14.3.2	根据调用的元素来创建 调用	186
14.3.3	练习	186
14.4	捕获当前调用	187
14.5	成对列表	189
14.6	解析与逆解析	191
14.7	使用递归函数遍历抽象 语法树	192
14.7.1	寻找 F 和 T	193
14.7.2	寻找通过赋值创建的所有 变量	194
14.7.3	修改调用树	197
14.7.4	练习	198

第 15 章 领域特定语言	200	第 17 章 代码优化	225
15.1 HTML	200	17.1 性能测试	226
15.1.1 目标	201	17.2 改进性能	229
15.1.2 转义	202	17.3 组织代码	229
15.1.3 基本标签函数	203	17.4 有人已经解决了这个问题吗	230
15.1.4 标签函数	204	17.5 尽可能少做	231
15.1.5 处理所有标签	205	17.6 向量化	236
15.1.6 练习	206	17.7 避免复制	237
15.2 LaTeX	206	17.8 字节码编译	238
15.2.1 LaTeX 数学	206	17.9 案例研究: t 检验	238
15.2.2 目标	207	17.10 并行化	240
15.2.3 <code>to_math</code>	207	17.11 其他技术	241
15.2.4 已知符号	207	第 18 章 内存	243
15.2.5 未知符号	208	18.1 对象大小	243
15.2.6 已知函数	209	18.2 内存使用与垃圾回收	246
15.2.7 未知函数	210	18.3 使用 <code>lineprof</code> 对内存进行性能分析	248
15.2.8 练习	211	18.4 原地修改	250
		18.4.1 循环	252
		18.4.2 练习	253
		第 19 章 使用 Rcpp 编写高性能函数	254
		19.1 开始使用 C++	255
		19.1.1 没有输入, 标量输出	256
		19.1.2 标量输入, 标量输出	256
		19.1.3 向量输入, 标量输出	257
		19.1.4 向量输入, 向量输出	258
		19.1.5 矩阵输入, 向量输出	258
		19.1.6 使用 <code>sourceCpp</code>	259
		19.1.7 练习	260
		19.2 属性和其他类	261
		19.2.1 列表和数据框	262
第 16 章 性能	214		
16.1 R 为什么速度慢	214		
16.2 微测试	215		
16.3 语言性能	216		
16.3.1 极端动态性	216		
16.3.2 可变环境下的名字搜索	218		
16.3.3 惰性求值开销	219		
16.3.4 练习	219		
16.4 实现的性能	220		
16.4.1 从数据框提取单一值	220		
16.4.2 <code>ifelse()</code> 、 <code>pmin()</code> 和 <code>pmax()</code>	220		
16.4.3 练习	222		
16.5 其他的 R 实现	222		

第四部分 性能

19.2.2	函数	262	19.6.2	R 向量化与 C++ 向量化	274
19.2.3	其他类型	263	19.7	在添加包中应用 Rcpp	275
19.3	缺失值	263	19.8	更多学习资源	276
19.3.1	标量	263	19.9	致谢	277
19.3.2	字符串	265			
19.3.3	布尔型	265			
19.3.4	向量	265			
19.3.5	练习	266			
19.4	Rcpp 语法糖	266	第 20 章	R 的 C 接口	278
19.4.1	算术和逻辑运算符	266	20.1	从 R 中调用 C 函数	279
19.4.2	逻辑总结函数	267	20.2	C 数据结构	280
19.4.3	向量视图	267	20.3	创建和修改向量	281
19.4.4	其他有用的函数	267	20.3.1	创建向量和垃圾回收	281
19.5	STL	268	20.3.2	缺失值和非有限值	282
19.5.1	使用迭代器	268	20.3.3	访问向量数据	283
19.5.2	算法	269	20.3.4	字符向量和列表	284
19.5.3	数据结构	270	20.3.5	修改输入	284
19.5.4	向量	270	20.3.6	强制转换标量	285
19.5.5	集合	271	20.3.7	长向量	285
19.5.6	图	272	20.4	成对列表	286
19.5.7	练习	272	20.5	输入验证	287
19.6	案例研究	272	20.6	寻找一个函数的 C 源代码	289
19.6.1	Gibbs 采样器	273	索引		292

第1章 简介

本书可以帮助你避免犯我曾经犯过的错误，带你走出我曾经无法跳出的死胡同，并会教你使用一些有用的工具、技术以及习惯，从而帮助你解决编程中遇到的多种问题。在这个过程中，我希望让你知道虽然 R 语言有很多令人沮丧的奇怪之处，但是 R 从本质上来说是一款十分适合数据分析和统计的优雅而美丽的语言。

如果你刚刚接触 R，你可能会问这门奇怪的语言有什么值得学习的呢。对于我来说，R 语言最好的特性有下面几条：

- ❑ R 是免费、开源、跨平台的。所以如果使用 R 进行数据分析，每个人都可以很容易地复制它（重现分析过程）。
- ❑ 大量可以用于统计建模、机器学习、可视化、数据导入与操作的添加包（package）可以供 R 使用。可能别人都已经做好了你正在尝试构建的任何模型或者图形。即使不能完全照搬，你也会从他们的工作中学习到很多经验从而对你的工作起到帮助作用。
- ❑ 最前沿的工具。统计学和机器学习领域的研究者在发表他们的研究论文时，经常会同时发布一个相应的 R 添加包。这就意味着你可以马上获得最新的统计技术并可以迅速实施。
- ❑ 对数据分析根深蒂固的支持。它包括很多特性，比如缺失值、数据框和子集选取。
- ❑ R 语言的爱好者组成了一个非常棒的社区。你可以非常容易地在 R- 帮助邮件列表（<https://stat.ethz.ch/mailman/listinfo/r-help>），stackoverflow（<http://stackoverflow.com/questions/tagged/r>），或者在其他一些针对特殊项目的邮件列表如 R-SIG-mixed-models（<https://stat.ethz.ch/mailman/listinfo/r-sig-mixed-models>）或 ggplot2（<https://groups.google.com/forum/#!forum/ggplot2>）中得到专家的帮助。你也可以通过 Twitter（<https://twitter.com/search?q=%23rstats>）、LinkedIn（<http://www.linkedin.com/groups/R-Project-Statistical-Computing-77616>），或者用户组（<http://blog.revolutionanalytics.com/local-r-groups.html>）与其他 R 学员进行联系。
- ❑ 交流结果的强大工具。R 添加包可以非常容易地将结果输出成 html 或 pdf 报告（<http://yihui.name/knitr/>），或者创建一个交互式网站（<http://www.rstudio.com/shiny>）。
- ❑ 强大的函数式编程基础。函数式编程的思想非常适合应对很多数据分析的挑战。R 提供了一个强大而又灵活的工具包，让你可以写出简洁的描述性代码。

- ❑ 一个为交互式数据分析和统计编程量身定做的 IDE（交互式开发环境）（<http://www.rstudio.com/ide/>）。
- ❑ 强大的元编程（metaprogramming）工具。R 不仅仅是一门编程语言，还是一个数据分析的交互环境。它的元编程功能可以让你写出超级简洁明了的函数并为设计领域特定语言（domain-specific language）提供了出色的环境。
- ❑ 可以与高性能编程语言（如 C、Fortran 和 C++）连接。

当然 R 也不是完美的。R 最大的挑战就是大多数用户都不是程序员。这就意味着：

- ❑ 你看到的很多 R 代码都是在急于解决某个紧迫问题的情况下编写的。因此这些代码并不是非常简洁、高效或者易于理解。大多数用户不会修改他们的代码来克服这些缺点。
- ❑ 与其他编程语言相比，R 社区更注重结果而非过程。软件工程最佳实践的知识不够完整。例如，使用源代码控制或自动化测试的 R 程序员还不够多。
- ❑ 元编程是一把双刃剑。有太多的 R 函数通过使用一些技巧来减少代码的输入量，由此造成的结果就是使代码变得很难理解，有些还会以意想不到的方式失败。
- ❑ 不同作者贡献的各种 R 添加包之间经常会出现矛盾，甚至与 R 的基础包发生冲突。每次使用 R 时，你都要面对它 20 多年的进化史。由于需要记住很多特例，所以 R 语言的学习会比较困难。
- ❑ R 并不是速度很快的编程语言，尤其是写得很差的 R 代码运行起来会非常慢。R 还非常耗费内存。

从个人角度来看，我觉得这些挑战也为有经验的程序员创造了一个极大的机会，让他们可以对 R 和 R 社区产生深远而又有益的影响。R 用户确实应该写出高质量的代码，尤其是在进行可重复研究时，但是他们现在还不具有这样的能力。我希望本书不仅可以帮助更多的 R 用户成为 R 程序员，还非常鼓励正在使用其他语言的程序员对 R 语言做出贡献。

1.1 本书的目标读者

本书是针对两个互补的群体：

- ❑ 想深入学习 R 并学习解决各种问题的新策略的中级 R 程序员。
- ❑ 正在学习 R，并想知道 R 为什么这样工作的其他语言的程序员。

要从本书获得最大收益，你需要编写大量的 R 语言或者其他语言的代码。你可能不了解函数的所有细节，尽管你目前可能为如何高效地应用它们而努力，但是你应该熟悉 R 中的函数是如何工作的，你应该熟悉 apply 系列函数（如 `apply()` 和 `lapply()`）。

1.2 通过本书你可以学到什么

我认为本书描述的这些技能是一个高级 R 程序员应该掌握的：能够写出可以在各种环境中使用的高质量代码。

读完本书之后，你将：

- ❑ 熟悉 R 的基础。你将理解复杂的数据类型和对它们进行运算的最佳方法。对函数如何工作有更深入的理解，还将认识并使用 R 的 4 个对象系统。
- ❑ 理解什么是函数式编程，以及为什么函数式编程是数据分析的有用工具。你将能快

速地学习如何使用现有工具，以及如何在需要的时候创建自己的函数。

- 欣赏元编程这把双刃剑。你将能在遵守原则的前提下使用非标准运算（non-standard evaluation）来创建函数，从而减少代码的输入量并创建优雅的代码来表达重要的运算。你将理解元编程的危险并知道为什么使用它时要小心。
- 对于 R 中什么样的操作会很慢且耗费内存能够产生很好的直觉。你将知道如何使用分析来找到阻碍性能提高的瓶颈，还会学到足够多的 C++ 知识，让你可以将很慢的 R 函数转换成非常快的 C++ 程序。
- 自如地阅读并理解大多数 R 代码。你将认识一些 R 的常用语（即使你自己不使用它们），并能够对其他人的代码做出评判。

1.3 元技术

有两种元技术对改善 R 程序员的技术非常有帮助：阅读源代码和采用科学的思维方式。

阅读源代码之所以重要是因为它可以帮助你写出更好的代码。开始培养这种技能的一种好方法就是查看你经常使用的函数和添加包的源代码。你会发现其中有很多值得效仿的地方，这样你就会知道什么可以使你的代码更好。你也会看到一些你不喜欢的东西，或许因为它的优点不够明显或者它让你感到不舒服。这样的代码也不是没有意义，它可以让你对好代码和坏代码有更具体的认识。

4

学习 R 时具有科学的思维方式是极有帮助的。如果你不知道有些事情是如何运行的，那就提出一个假设，设计一些实验，做实验并记录结果。这种训练是非常有用的，即使你不能解决这个问题并需要别人的帮助，但在寻求帮助时你可以告诉他们你已经尝试过哪些方法。另外，当你得到新的答案后，你也已经在思想上做好了准备来更新你的世界观。当我向别人描述问题时（构建一个可再现示例（<http://stackoverflow.com/questions/5963269>）的艺术），我都会指出自己的解决方案。

1.4 推荐阅读

R 仍然是一门相对比较年轻的语言，可以帮助你理解它的资源还在不断地完善之中。在理解 R 的整个过程中，我发现使用其他编程语言的资源是非常有帮助的。R 同时具有函数式编程和面向对象（OO）编程的特点。学会如何使用 R 来实现这些概念可以帮助你利用起你自己关于其他编程语言的知识，并让你发现（R）哪里还需要改进。

我发现 Harold Abelson 和 Gerald Jay Sussman 的《The Structure and Interpretation of Computer Programs》（<http://mitpress.mit.edu/sicp/full-text/book/book.html>）（SICP）一书对于理解 R 的对象系统为什么那样运行是非常有帮助的。这是一本简明但又深奥的书。读完本书之后，我第一次感觉到我可以设计出自己的面向对象系统。在本书中我第一次认识了 R 面向对象系统中常用的泛型函数（generic function）风格。它帮助我理解了 R 的优缺点。SICP 同样讲述了很多函数式编程的知识，以及如何构建简单的函数，当把这些简单函数结合在一起时它又会变得非常强大。

要理解 R 相对于其他编程语言的优缺点，我发现 Peter van Roy 和 Sef Haridi 的《Concepts, Techniques and Models of Computer Programming》（<http://amzn.com/0262220695?tag=devtools-20>）非常有用。它让我知道了为什么 R 的 copy-on-modify 语义使得对代码的理解变

得如此简单, 虽然现在实现起来还不是非常有效, 但这个问题是可以解决的。

- 5 如果你想成为一个更好的程序员, 没有哪本书比 Andrew Hunt 和 David Thomas 的《The Pragmatic Programmer》(<http://amzn.com/020161622X?tag=devtools-20>) 更好了。这本书并不针对特定的语言, 它对如何使你成为更好的程序员提供了很多建议。

1.5 获取帮助

目前当你遇到困难并且不知道如何解决它时, 你有两个渠道可以获得帮助: stackoverflow (<http://stackoverflow.com>) 和 R-help 邮件列表。在这两个地方你能得到有益的帮助, 但是每个社区都有自己的文化和期望。通常情况下你最好先花一段时间潜水, 在了解了该社区的文化和期望之后再提出你的第一个问题。

一些建议:

- ❑ 当你遇到问题时首先要确保你使用的是 R 的最新版本和添加包, 很可能这些问题已经在最新版本中被修复了。
- ❑ 花一段时间构建一个可再现示例 (<http://stackoverflow.com/questions/5963269>)。这通常是一个非常有用的过程, 在再现问题的过程中你经常会找到问题的原因。
- ❑ 在求助前首先搜索一下相关问题。如果有人已经问过同样的问题并得到了回答, 使用现有答案会更快速方便。

1.6 致谢

- 6 我要感谢 R-help 以及 stackoverflow (<http://stackoverflow.com/questions/tagged/r>) 上不知疲倦的贡献者。有太多人需要感谢了, 尤其要感谢 Luke Tierney、John Chambers、Dirk Eddelbuettel、JJ Allaire 和 Brian Ripley 花费了大量的时间和精力为我纠正无数的错误。

本书是以开源方式编写的 (<https://github.com/hadley/adv-r>), Twitter (<https://twitter.com/hadleywickham>) 上的朋友对本书的章节给出了很多建议。这是社区的智慧结晶: 很多人阅读了书稿, 修改了错别字, 对修改提出了建议, 对本书的内容做出了贡献。没有这些人的帮助, 本书不可能像现在这样给力, 非常感谢他们的帮助。尤其要感谢 Peter Li, 他从头到尾阅读了本书并修订了很多错误。其他非常给力的贡献者有: Aaron Schumacher、@crtahlin、Lingbing Feng、@juancentro 和 @johnbaums。

以字母顺序感谢所有的贡献者: Aaron Schumacher、Aaron Wolen、@aaronwolen、@absolutelyNoWarranty、Adam Hunt、@agrabovsky、@ajdm、@alexbbrown、@alko989、@allegretto、@AmeliaMN、@andrewla、Andy Teucher、Anthony Damico、Anton Antonov、@aranlunzer、@arilamstein、@avilella、@baptiste、@blindjesse、@blmoore、@bnjmn、Brandon Hurr、@BrianDiggs、@Bryce、C. Jason Liang、@Carson、@cdrv、Ching Boon、@chiphogg、Christopher Brown、@christophergandrud、Clay Ford、@cornelius1729、@cplouffe、Craig Citro、@crossfitAL、@crowding、CrtAhlin、@crtahlin、@escheid、@csgillespie、@cusanovich、@cwarden、@cwickham、Daniel Lee、@darrkj、@Dasonk、David Hajage、David LeBauer、@dchudz、dennis feeihan、@dfeehan、Dirk Eddelbuettel、@dkahle、@dlebauer、@dlschweizer、@dmontaner、@dougmitarotonda、@dpatschke、@duncandonutz、@EdFineOKL、@EDiLD、@eipi10、@elegrand、@EmilRehnberg、

Eric C. Anderson、@etb、@fabian-s、Facundo Muñoz、@flammy0530、@fpopin、Frank @Farach、@freezby、@fyears、Garrett Grolemond、@garrettgman、@gavinsimpson、@gggtest、Gökçen Eraslan、Gregg Whitworth、@gregorp、@gsee、@gsk3、@gthb、@hassaad85、@i、Iain Dillingham、@ijlyttle、Ilan Man、@immanuelcostigan、@initdch、Jason Asher、Jason Knight、@jasondavies、@jastingo、@jcborras、Jeff Allen、@jeharmse、@jentjr、@JestonBlu、@JimInNashville、@jinlong25、JJ Allaire、Jochen Van de Velde、Johann Hibschan、John Blischak、John Verzani、@johnbaums、@johnjosephhorton、Joris Muller、Joseph Casillas、@juancentro、@kdauria、@kenahoo、@kent37、Kevin Markham、Kevin Ushey、@kforner、Kirill Müller、Kun Ren、Laurent Gatto、@Lawrence-Liu、@ldfmrails、@lgatto、@liangcj、Lingbing Feng、@lynaghk、Maarten Kruijver、Mamoun Benghezal、@mannyishere、Matt Pettis、@mattbaggott、Matthew Grogan、@mattmalin、Michael Kane、@michaelbach、@mjsduncan、@Mullefa、@myqlarson、Nacho Caballero、Nick Carchedi、@nstjhp、@ogennadi、Oliver Keyes、@otepoti、Parker Abercrombie、@patperu、Patrick Miller、@pdb61、@pengyu、Peter F Schulam、Peter Lindbrook、Peter Meilstrup、@philchalmers、@picasa、@piccolbo、@pierrerroudier、@pooryorick、R. Mark Sharp、Ramnath Vaidyanathan、@ramnathv、@Rappster、Ricardo Pietrobon、Richard Cotton、@richardreeve、@rmflight、@rmsharp、Robert M Flight、@RobertZK、@robiRagan、Romain François、@rrunner、@rubenfcasal、@sailingwave、@sarunasmerkliopas、@sbgraves237、Scott Ritchie、@scottko、@scottl、Sean Anderson、Sean Carmody、Sean Wilkinson、@sebastian-c、Sebastien Vigneau、@shabbychef、Shannon Rush、Simon O'Hanlon、Simon Potter、@SplashDance、@ste-fan、Stefan Widgren、@stephens999、Steven Pav、@strongh、@stuttgartur、@surmann、@swnydick、@taekyunk、Tal Galili、@talgalili、@tdenes、@Thomas、@thomasherbig、@thomaszumbrunn、Tim Cole、@tjmahr、Tom Buckley、Tom Crockett、@ttriche、@twjacobs、@tyhenkaline、@tylerritchie、@ulrichatz、@varun729、@victorkryukov、@vijayarve、@vzemlys、@wchi144、@wibeasley、@WilCrofter、William Doane、Winston Chang、@wmc3、@wordnerd、Yoni Ben-Meshulam、@zackham、@zerokarmaleft、Zhongpeng Lin。

7

1.7 约定

在本书中 $f()$ 代表函数， g 代表变量和函数参数， $h/$ 代表路径。

大的代码块包含输入和输出。输出已经被注释掉了，所以如果你有本书的电子版，例如，<http://adv-r.had.co.nz>，你就可以很简单将例子复制并粘贴到 R 中。为了与正常的注释相区别，将注释输出的注释符设定为 $\#>$ 。

1.8 声明

本书是在 Rstudio (<http://www.rstudio.com/ide/>) 中使用 Rmarkdown (<http://rmarkdown.rstudio.com/>) 编写的。使用 knitr (<http://yihui.name/knitr>) 和 pandoc (<http://johnmacfarlane.net/pandoc/>) 将原始的 Rmarkdown 转换成 html 和 pdf。本书的网站 (<http://adv-r.had>。