应用统计学系列教材 Texts in Applied Statistics

非参数统计
Non-parametric Statistics

王星 编著
Wang Xing

清华大学出版社 Springer

应用统计学系列教材 Texts in Applied Statistics

非参数统计

Non-parametric Statistics

王星 编著
Wang Xing

清华大学出版社 Springer
北京

内 容 简 介

本书是非参数统计教材，内容从经典非参数统计推断到现代前沿，包括R基础、基本概念、单一样本的推断问题、两独立样本数据的位置和尺度推断、多组数据位置推断、分类数据的关联分析、秩相关和分位数回归、非参数密度估计、一元非参数回归和数据挖掘与机器学习共计10章. 本书配有大量与社会、经济、金融、生物等专业相关的例题和习题，给出示范解答过程，方便自学.

本书可以作为高等院校统计、经济、金融、管理专业的本科生课程的教材，也可以作为其他相关专业研究生的教材和教学参考书，另外，对广大从事与统计相关工作的实际工作者也极具参考价值.

图书在版编目（CIP）数据

非参数统计/王星编著. —北京：清华大学出版社，2009.3

(应用统计学系列教材)

ISBN 978-7-302-19167-4

Ⅰ. 非… Ⅱ. 王… Ⅲ. 非参数统计-高等学校-教材 Ⅳ. O212.7

中国版本图书馆CIP数据核字（2008）第211337号

责任编辑：王海燕　赵从棉

责任校对：王淑云

责任印制：杨　艳

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座

http://www.tup.com.cn　　**邮　　编**：100084

社　总　机：010-62770175　　**邮　　购**：010-62786544

投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn

质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者：清华大学印刷厂

经　　销：全国新华书店

开　　本：170×230　**印　张**：19.75　**字　数**：428千字

附光盘1张

版　　次：2009年3月第1版　　**印　　次**：2009年3月第1次印刷

印　　数：1～3000

定　　价：37.00元

前　言

统计是一个面向问题解决的、系统收集数据和基于数据做出回答的过程, 其本质是通过在随机现象中寻找分布规律回答现实问题的科学过程. 实际问题的复杂性和人类认知的局限性, 造成反映实际问题的数据在问题表示的充分性、代表性和分布的单一性等方面, 与传统的统计应用要求不相匹配, 于是催生了对数据分布假定宽松的非参数统计的兴起与发展. 尤其是最近 20 年来, 随着信息技术和网络技术的快速发展, 基于大量数据计算探索数据分布特点的数据分析方法层出不穷, 成为非参数统计发展的新主题, 代表着统计学未来的方向. 非参数统计自然成为连接统计学、信息学和计算机科学等交叉研究的桥梁, 共同推动数据分析和信息利用整体地向前发展.

本书是一本专门讲授非参数统计理论和方法的教科书. 内容主要分为两个部分: 传统的非参数统计推断和现代非参数统计方法. 传统的非参数推断内容由单一样本、两样本及多样本非参数统计估计和假设检验、分类数据的关联分析方法、定量数据的相关和回归等内容构成; 现代非参数统计方法部分包含非参数密度估计、非参数回归和数据挖掘与机器学习技术等内容.

本书的主要特色是结合 R 软件讲解非参数统计方法的原理和应用, 我们的宗旨是塑造有独立专业思考能力, 对所学知识有比较地选择, 并能够使用恰当方法解决实际问题的统计专业人才. 据此, 我们在课程设计中, 专门设计了学生在接受知识的过程中对知识的运用和鉴别能力的训练. 本书大部分例题都给出 R 源程序解法示例, 各种理论条件的检验、讨论、分析和比较, 鼓励学生针对数据的特点, 独立编写数据分析程序. 为加强与 R 的结合, 书中图形大部分由 R 生成, 我们广泛收集了很多领域数据分析实例和应用编写成本书的例题和习题, 以扩展学生的应用领域, 提高学生解决实际问题的能力.

本书可作为统计、经济、管理、生物等宏、微观专业领域本科三、四年级以上学生以及相关研究人员学习非参数统计方法的教材, 也可以用作统计研究或从事数据分析的方法的参考书. 本书的先修课程只需具备初等统计学基础. 对统计基础略感陌生的读者, 可以阅读第 2 章相关内容作为补充. 本书的内容可以安排在一学期 54 课时内完成, 建议安排 10 课时左右用于学生上机实践. 本书备有丰富的习题, 兼有理论推导、方法应用和上机实践题目.

本书写作过程中, 得到众多老师的支持与鼓励. 感谢吴喜之先生多年来在非参数统计前沿和方法论上的引领和指导, 感谢袁卫、金勇进、易丹辉、张波、赵明德、

谢邦昌和郁彬等教授在学科发展动态上的启迪与建议, 感谢赵彦云、高敏雪等教授的支持与鼓励, 感谢朱建旭同学参与了第 10 章的部分编写, 协助整理了部分文献、图表和习题, 感谢孙兆楠、赵博元、詹瑾、李扬、王旭、王爱玲、伍燕然以及研究生讨论班的各位学生对部分内容进行的相关讨论, 感谢责任编辑王海燕和赵从棉, 正是凭借着她们对本书出版计划的坚定而耐心的支持, 才有本书的问世.

感谢我的恩师、朋友、学生和家人与我相伴的岁月!

王　星

中国人民大学统计学院

E-mail:wangxingscy@gmail.com

目　录

第1章　R 基础

R 是一种专业统计分析软件, 最早于 1995 年由 Auckland 大学统计系的 Robert Gentleman 和 Ross Ihaka 等研制开发, 1997 年开始免费公开发布 1.0 版本. 在短短的 10 年时间里, R 发展迅速, 现已发展到 R2.7 系列版本. 据不完全统计, 在欧美等发达国家的著名高等学府, R 不仅是专业学习统计的流行教学软件, 而且已成为从事统计研究的学生和统计研究人员必备的统计计算工具.

R 的主要特点归纳如下:

(1) R 是自由免费的专业统计分析软件, 拥有强大的面向对象的开发环境, 可以在 UNIX, Windows 和 MACINTOSH 等多种操作系统中运行.

(2) 使用可编程语言是 R 作为专业软件的基本特点. 众所周知, 目前流行的许多商业统计分析软件主要是通过单击菜单完成计算和分析组合任务, 用户不得不在预定义好的统计过程中选择可能接近的模块进行数据分析, 被迫接受预设的程式化输出, 许多应有的对数据的观察、体验和分析判断受到很大限制. 而 R 却克服了这些弱点.

(3) R 的语言与 S 语言非常相似, 虽实现方法不同, 但兼容性很强. 作为面向对象的语言, R 集数据的定义、插入、修改和函数计算等功能于一体, 语言风格统一, 可以独立完成数据分析生命周期的全部活动. 作为标准的统计语言, R 几乎集中了所有程序编辑语言的优秀特点. 用户可以在 R 中自由地定义各种函数, 设计实验, 采集数据, 分析得出结论. 在这个过程中, 用户不仅可能延伸 R 的基本功能, 而且还可能自创一些特殊问题的统计过程. R 是一种解释性语言, 语法与英文的正常语法和其他程序设计语言的语法表述相似, 容易学习, 编写的程序简练, 费时较短.

(4) R 提供了非常丰富的 2D 和 3D 图形库, 是数据可视化的先驱, 能够生成从简单到复杂的各种图形, 甚至可以生成动画, 满足不同信息展示的需要. 用户可以修改其中每个细节, 调整图形的属性满足报表报送要求. R 的兼容性比较好, 其图形不仅可以与 Microsoft Office 等办公软件兼容, 而且可以以.pdf, .ps, .eps 等格式保存输出, 于是就可能非常方便地输出到 Latex 等正式出版的文章编辑器中, 生成高品质的科技文章.

(5) R 更新迅速, 很多由最新的统计算法和前沿统计方法生成的程序都可能轻易地从 R 镜像 (CRAN) 下载到本地, 它是目前发展最快, 拥有方法最新、最多和最全的统计软件.

总而言之, R 从根本上摒弃了套用模型的傻瓜式数据分析模式, 而是将数据分

析的主动权和选择权交给使用者本身. 数据分析人员可以根据问题的背景和数据的特点, 更好地思考从数据出发如何选择和组合不同的方法, 并将每一层输出反馈到对问题和数据处理的新思考上. R 为专业分析提供了分析的弹性、灵活性和可扩展性, 是利用数据回答问题的最佳平台.

诚然, R 也存在不足, 与同类的 MATLAB 相比, 其最大的缺点是对超大量数据的运算速度过慢, 当然这是很多统计分析软件共同存在的问题. 原因是 R 往往需要将全部数据加载到临时存储库中进行运算 (这种情况在 R 2.0 以后的版本有逐步的改善). 尽管如此, R 的免费开放源代码, 使得它在与昂贵的商业分析软件的竞争中成为一枝独秀, 越来越多的数据分析人员已经开始尝试和接纳 R. 用 R 尝试最新的统计模型, 用 R 揭开数据的秘密, 用 R 实现数据的价值, 用 R 发展更好的统计算法. R 突破了数据分析的商业门禁, 将全球数据分析爱好者自然地集结在一起, 实现平等的经验分享与思想交流.

基于以上诸多优点, R 所承载的最新方法无障碍地迅速扩展到医疗、金融、经济、商业等各领域, 成为统计的时代符号.

1.1　R 基本概念和操作

1.1.1　R 环境

双击桌面上的 R 图标, 启动 R 软件, 就会呈现 R 窗口和 R 命令窗口“>”符号, 表示 R 等待使用者在这里输入指令, 如图 1.1 所示.

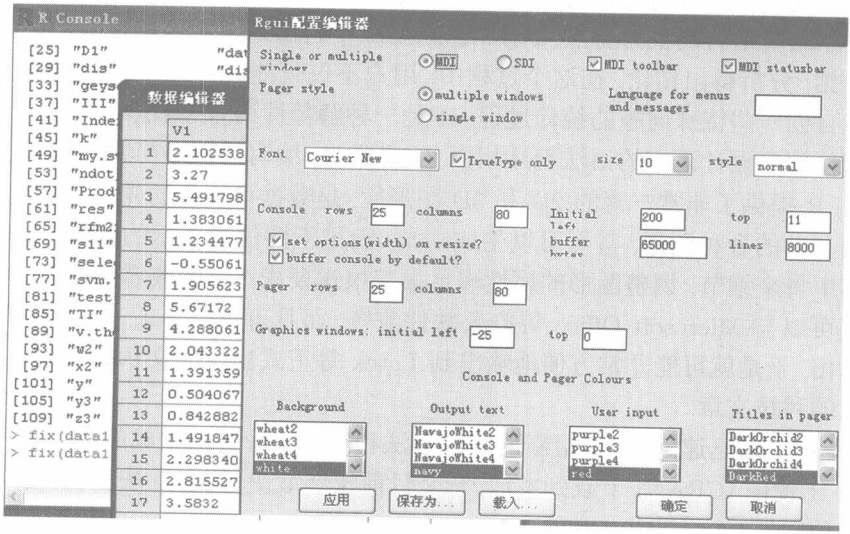

图 1.1　R 2.7.1 主界面

当输入指令后, 按 Enter 键就可以执行指令, 如:

```
> 2+3
> 5
```

1.1.2 常量

R 中的常量基本分为四种类型: 逻辑型、数值型、字符型和因子型. TRUE 和 FALSE 是逻辑型常量, 25.6, π 是数值型常量, 某人的身份证号码“11010...”以及地名如 “Beijing” 是字符型常量. 因子型包括分类数据和顺序数据, 分类数据如: 每个人的性别、学生的学号等. 性别可以表示为 1(男), 0(女), 1 或 0 仅仅表示不同的类别; 考试成绩分为 5(优), 4(良), 3(中), 2(及格) 和 1(不及格)5 个等级, 对这类数据不能进行加减乘除运算. 无论是字符类型还是因子类型的数据常常以数字的形态出现, 但不能将它们理解成普通的整数. 在许多分析中, 需要将字符类型的数据转换成因子类型, 以方便计算机识别. 下面是生成因子的命令:

```
> x<-c("Beijing","Shanghai","Beijing","Beijing","Shanghai")
> y<-factor(x)
> y
[1]Beijing Shanghai Beijing Beijing Shanghai
Levels: Beijing Shanghai
```

也可以写为:

```
> y<-factor(c(1,0,1,1,0))
> y
[1] 1 0 1 1 0
Levels: 1 0
```

这里 Levels 为因子水平, 表示有哪些因子. c() 为连接函数, 把单个标量连成向量, 下面将详细介绍. 有了变量名, 首先可以将 y 与 0 进行比较:

```
> y==0
 FALSE  TRUE FALSE FALSE  TRUE
```

此时, R 执行数 0 与 y 的每个值比较. 对象中的数据允许出现缺失, 缺失值用大写字母 NA 表示. 函数 is.na(x) 返回 x 是否存在缺失值.

1.1.3 算术运算

算术运算是 R 中的基本运算, R 默认的运算提示符是 “>”, 在 “>” 后可以进行运算. 下面先举几个例子.

(1) 计算 7×3, 可执行如下命令:

```
> 7*3
> 21
```

(2) 计算 $(7+2) \times 3$, 可执行如下命令:

```
> (7+2)*3
> 27
```

也可以调用 R 内置函数, 如:

(3) 计算 $\log_2\left(\frac{12}{3}\right)$, 可执行如下命令:

```
> log(12/3,2)
> 2
```

需要注意的是, 求对数与底的设置有关, 底称做函数的参数. R 中的函数都有不同的参数, 省略时为默认值. 对数函数的默认底数是常数 e, 其他的常用初等函数如: 三角函数 sin(),cos(),tan(); 反三角函数 acos(),asin(),atan(); 二值反三角函数 atan2(,); 指数函数 exp(); 对数函数 log(N,a),log(N,a) 表示 $\ln_a N$; 组合函数 choose(,); 求 n 的阶乘 gamma(n-1). 它们都在 R 的基础包里 (Base Package), 用?library(base) 可以查看, 用 ? 函数名或 help(函数名) 可以查阅函数的功能用法和参数设置.

1.1.4　赋值

给变量赋值用 "=" 或 "<−" 两个字符串, 比如将 3 赋给变量 x, 用变量 x 通过函数生成变量 y, 使用命令:

```
> x<-3
> y=1+x
> y=4
```

需要注意的是, R 中变量名函数名区分大小写, 这与 SAS 软件不同. 在 SAS 中关键函数字和 SAS 变量名可以不区分大小写.

1.2　向量的生成和基本操作

统计学的研究对象是群体, 所以将许多个体的观测值作为一个整体进行操作和研究在数据分析中相当普遍. 例如, 观察统计一个班 50 名学生的身高, 显然如果能把这些数据储存在一个对象中, 统一处理会很方便. 将多个单一数据排列在一起, 便产生了结构这个概念. 在数据结构中, 最简单的是向量, 此时观测是一维的; 如果观测是多维的, 则还可以用矩阵、数组、数据框和列表等储存更为复杂的数据结构. 本节以向量为例, 介绍数据结构中常用的操作和函数, 其他复杂的结构在 1.3 节介绍.

1.2.1　向量的生成

R 中有 3 个非常有用的命令可以生成向量.

1. c

c 是英文单词 concatenate 的缩写, 是连接命令, 它可以将单个的元素, 或分段的数列连接成一个更长的数列, 用户只需将组成向量的每个元素列出, 并用 c 组合起来即可. 基本运算如下:

```
> a<-c(15,27,89)
> a
[1]  15  27  89
> b<-c("cat","dog","fish")
> b
[1]  "cat" "dog" "fish"
```

2. seq

seq 是生成等差数列的命令, 其语法结构如下所示:

```
seq(from,to,by,length,...)
```

其中, from 表示序列起始的数据点, to 表示序列的终点, by 表示每次递增的步长. 默认状态表示步长为 1, length 表示序列长度. 如:

```
> x=seq(1,10)
[1]  1  2  3  4  5  6  7  8  9 10
> y=seq(100,0,-20)
[1] 100  80  60  40  20   0
```

seq(1,10) 还可用更简单的方式表示, 比如:

```
> 1:10                              #seq(1,10)
```

如果我们知道序列终点可能的值, 但不知道确切的终点, 可以通过 length 控制得到序列:

```
> seq(0,1,0.05,length=10)
```

上面这个序列起始于 0, 每次递增 0.05, 生成一个有 10 个数的数值型向量. 顺便提一下, length 命令可以表示向量中元素的个数, 称为向量的长度, 如

```
> length(y)
[1]  6
```

3. rep

rep 是生成循环序列的命令, 它的语法结构如下所示:

```
rep(x,times)
```

其中 x 表示序列所循环的数或向量, times 表示循环重复的次数.

例 1.1

(1) 生成由 5 个 2 组成的向量;

(2) 将 "1", "a" 依次重复 3 遍;

(3) 生成依次由 10 个 1, 20 个 3 和 5 个 2 组成的向量.

```
> rep(2,5)
2 2 2 2 2
> rep(c(1,"a"),3)
"1" "a" "1" "a" "1" "a"
> rep(c(1,3,2),c(10,20,5))
 [1] 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[29] 3 3 2 2 2 2 2
```

与 seq 类似, 可以使用 length 命令控制序列的长度, 比如:

```
> rep(c(1,4,6),length=5)
1 4 6 1 4
```

1.2.2　向量的基本操作

定义向量之后, 下面介绍如何对向量进行操作. 这些操作主要包括查找数据、插入数据、更新数据、删除数据、向量与向量的合并、拆分向量以及排序等. 值得一提的是, 这里介绍的大部分操作符对其他数据结构也适用, 也就是说, 对复杂的数据结构, 只要对我们介绍的命令略做修改就可以使用, 语法是相似的, 这种统一性的特点给初学者熟悉 R 带来极大方便.

1. 向量 a 中第 i 位置的元素表示

向量 a 中第 i 位置的元素 a[i] 表示为 a[i], 如:

```
> a=2:6
> a[1]
[1]    2
> a[length(a)]
[1]    6
```

如果输入的位置超出向量的长度, 则 R 输出 NA. NA 表示数据缺失, 如下所示:

```
> a[6]
[1]  NA
```

提取向量 a 的第 $i_1, i_2, \cdots, i_k$ 位置上元素的语法为 a[c(i_1, i_2, $\cdots$, i_k)], 如:

```
> subset1.a<-a[c(1,3,6)]
[1] 2 4 NA
> subset2.a<-a[c(1:3)]
[1] 2 3 4
```

2. 在向量中插入新的数据

在向量 a 第 i 位置后插入新数据 z 的方法是:

```
c(a[1:i-1],z,a[i:length(a)])
```

1 : n 表示从 1 到 n 间隔为 1 的数列. 下面在向量 a 的第三个位置插入数值 9:

```
> anew<-c(a[1:2],9,a[3:5])
> anew
[1]2 3 9 4 5 6
```

3. 向量与向量的合并

将 a 和 b 两个向量合并为一个新向量的方法是:

```
> b<-c(35,40,58)
> ab<-c(a,b)
[1] 2 3 4 5 6 35 46 58
```

然而, 值得注意的是, 如果将非数值型向量和数值型向量合并, 结果是所有数据类型被统一到 R 所默认的基本类型 —— 字符型. 如:

```
> z<-c(a,"good")
> z
[1] "2"    "3"    "4"    "5"    "6"   "good"
```

我们注意到, 所有数据都统一为字符型, 此时如果对 a 进行数值运算会发生错误, 如:

```
> z*3
```

错误于 z * 3 : 二进列运算符中有非数值变元

4. 在向量中删除数据

a[-i] 表示删除向量 a 的第 i 个元素, 如:

```
> delete.a<-a[-1]
> delete.a
[1] 3 4 5 6
```

如果要删除一串数, 可以定义一个位置变量 delete.1, 再做删除, 比如:

```
> delete.l<-c(1,3)
> again.delete.a<-delete.a[-delete.l]
[1] 4 6
```

5. 更新向量中的数据

将 a 向量中第 5 个位置改为 22 的程序如下:

```
> a[5]<-22
> a
[1] 2 3 4 5 22
```

6. 把向量逆序排列

```
> b=1:5
> rev(b)
[1] 5 4 3 2 1
```

7. 对向量排序

对向量 b 排序: