Springer

R语言应用系列

R语言初学者指南

A Beginner's Guide to R

〔英〕阿兰·F·祖尔
〔英〕埃琳娜·N·耶诺 著
〔荷〕埃里克·H·W·G·密斯特
周丙常 王亮 译

只用看前三章 !!!!!!!!!!!!!!!

只用看前三章 !!!!!!!!!!!!!!!!!!!!!!!!!!!!

Alain F. Zuur
Elena N.Ieno
Erik H.W.G. Meesters

强烈建议：边看、边敲代码

本书数据集下载 www.highstat.com

西安交通大学出版社
XI'AN JIAOTONG UNIVERSITY PRESS

ℛ语言应用系列

A Beginner's Guide to R

R语言初学者指南

〔英〕阿兰·F·祖尔
Alain F. Zuur

〔英〕埃琳娜·N·耶诺
Elena N. Ieno

〔荷〕埃里克·H·W·G·密斯特
Erik H. W. G. Meesters

周丙常　王亮　译

西安交通大学出版社
Xi'an Jiaotong University Press

陕西省版权局著作权合同登记号　图字 25－2010－115 号

图书在版编目(CIP)数据

R 语言初学者指南/(英)祖尔(Zuur,A. F.),(英)耶诺(Ieno,E. N.),(荷)密斯特(Meesters, E. H. W. G.)著;周丙常,王亮译. —西安:西安交通大学出版社,2011.9
书名原文:A Beginner's Guide to R
ISBN 978－7－5605－3942－3

Ⅰ.①R… Ⅱ.①祖… ②耶… ③密… ④周… ⑤王… Ⅲ.①程序语言-程序设计 Ⅳ.①TP312

中国版本图书馆 CIP 数据核字(2011)第 087243 号

书　　名	R 语言初学者指南
著　　者	(英)阿兰·F·祖尔,(英)埃琳娜·N·耶诺,(荷)埃里克·H·W·G·密斯特
译　　者	周丙常　王　亮
策划编辑	赵丽平　李　颖
责任编辑	李　颖
出版发行	西安交通大学出版社 (西安市兴庆南路 10 号　邮政编码 710049)
网　　址	http://ligong.xjtupress.com
电　　话	(029)82668357　82667874(发行中心) (029)82668315　82669096(总编办)
传　　真	(029)82668280
印　　刷	西安交通大学印刷厂
开　　本	787mm×1 092mm　1/16　**印张** 14.5
印　　数	0001～3000　**字数** 234 千字
版次印次	2011 年 9 月第 1 版　2011 年 9 月第 1 次印刷
书　　号	ISBN 978－7－5605－3942－3/TP・550
定　　价	36.00 元

读者购书、书店添货、如发现印装质量问题,请与本社发行中心联系、调换。
订购热线:(029)82665248　(029)82665249
投稿热线:(029)82665380
读者信箱:banquan1809@126.com

前　言

完全不懂 R 的初学者

这本书为谁而写？

自 2000 年以来，我们已经为超过 5000 名生命科学家讲授了统计学。这听起来似乎很多，事实确实如此，而且人数仍在迅速增加（虽然一些课程只有 6 名学生），有些课程有 200 名本科生。我们的大多数课程在欧洲讲授，但是我们也在南美洲、中美洲、中东和新西兰讲授课程。当然，在大学和研究机构教学意味着我们的学生几乎来自世界的任何地方。听课的学员包括本科生，但是大部分是理学硕士、研究生、博士后，或者资深科学家，以及一些顾问和非研究人员。

这样的经历让我们广泛地了解了典型的生命科学家的统计知识。“典型的”这个词可能会引起误解，因为那些参加统计课程的科学家可能不熟悉这样的主题或者已经忘记。通常，与我们一起工作的人，正处在他们教育或者事业的某一阶段，并且已经结束了一个统计课程，其覆盖的主题诸如均值、方差、t-检验、卡方检验以及假设检验，也许还包括半小时的线性回归的学习。

有许多介绍统计与 R 的书籍，但本书不处理统计问题，因为以我们的经验，同时讲授统计和 R 意味着两个陡峭的学习曲线：一个是统计方法，一个是 R 代码。这显然不在很多学生的承受范围之内。本书是为学习 R 基本介绍的人准备的。显然，“基本的”是含糊的；对一个人来说是基本的，可能对另外一个人是高级的。

R 包含“你需要知道你在做什么”这么一个高要求内容，并且它的应用需要大量的逻辑思维。作为统计学家，很容易呆在象牙塔里，希望生命科学家敲我们的门并请求学习我们的语言。这本书尽量使用简单的语言。如果短语“完全的初学者”冒犯了你，我们道歉，但是它回答了这个问题：这本书是为谁而写？

这本书的所有作者都是 Windows 用户，他们对 Linux 和 Mac OS 的经验有限，但 R 在具有这些系统的计算机上也是可行的，并且我们提供的所有 R 代码在这些系统上均能正确运行。然而，可能在保存图形上有一点区别。非 Windows 用户也需要寻找一个替代 Tinn-R 的文本编辑器（第 1 章讨论你在哪里可以找到这些信息）。

本书使用的数据集

本书主要使用的是生命科学的数据。然而，无论你研究的领域和数据是什么，所给的程序都是适用的。所有领域的科学家都需要载入数据、处理数据、生成图形，并且最后进行分析，每一个案例的 R 命令都非常相似。一本 200 页的书不能提供一个范围很大的多样化的数据集类型。并且以我们的经验，大相径庭的例子会使读者混淆。最理想的方法可能是使用单独的一个数据集示范所有的方法，但是这可能会使很多人感到不易接受。因此，我们使用生态学数据集（例如，涉及植物、海底生物、鱼类、鸟类）以及流行病学数据集。

本书使用的所有数据集可以通过网站 www.highstat.com 下载。

Newburgh　　阿兰·F·祖尔
Newburgh　　埃琳娜·N·耶诺
Den Burg　　埃里克·H·W·G·密斯特

目　录

第1章

引言

我们首先讨论如何获取和安装 R，并给出启动 R 时的使用和一般信息 P.1
的概述。1.6 节我们讨论编写代码的文本编辑器的使用，并给出了推荐使用的一般工作模式。1.7 节的重点是使用帮助文件和新闻组获得帮助。安装 R 和载入包在 1.8 节叙述，历史回顾和文献介绍放在 1.10 节。在 1.11 节，我们提供了一些阅读本书的一般性建议以及教师如何使用本书，在最后一节，我们总结了本章介绍的 R 函数。

1.1 什么是 R?

这虽然是一个简单的问题，但是并不太容易回答。广义地定义，R 是允许用户编辑算法并使用其它可编程工具的一种计算机语言。这种含糊的描述适用于许多计算机语言，解释 R 能做什么或许更有益。在我们的 R 课程中，我们告诉学生，“R 可以做你想象的任何事情”，这应该没有言过其实。借助 R 你可以编写函数，进行计算，应用很多可获得的统计技术，生成简单或者复杂的图形，甚至编写你自己的库函数。许多研究院、公司和大学已经使用 R，有一个很大的用户组支持 R。在过去 5 年里，许多包括参考 R 和应用 R 函数进行计算的图书相继出版。重要的一点是 R 是免费使用的。

那么为什么不是每个人都在使用它？这是一个容易回答的问题。R 有一个陡峭的学习曲线！它的使用需要编程，并且尽管各种图形用户界面存在，但是没有一个全面到足以完全避免编程。然而一旦你掌握了 R 的基本步骤，你将不再喜欢使用其它相似的软件包。

界面的结果和使用帮助菜单。移动光标到最后一点，在符号>（即光标显示的地方）后输入 2+2：

```
> 2 + 2
```

并单击回车键。命令里的空格是被忽略的。你也可以输入2+2，或者2 +2。我们用简单的 R 命令，是为了强调你必须在命令窗口中输入一些命令才能在 R 中得到输出结果。2+2 将得到：

```
[1] 4
```

在下一章中讨论[1]的含义，但是很明显 R 可以计算 2 与 2 的和。这个简单的例子显示了 R 是如何工作的；输入一些命令，单击回车键，R 将运行你的命令。技巧是输入正确的命令。错误很容易产生。例如，假设你想计算以 10 为底的 2 的对数。你可能输入：

```
> log(2)
```

并且得到：

```
[1] 0.6931472
```

但是 0.693 不是正确答案。这是自然对数。你应该用：

```
> log10(2)
```

它将给出一个正确的答案：

```
[1] 0.30103
```

尽管 log 和 log10 的命令能够并且应该记住，但是后面我们给出一个不可能记住代码的例子。输入错误也可能出现问题。输入 2 + 2w 会给出如下信息

```
> 2 + 2w
Error: syntax error in "2+2w"
```

P.7 R 当然不知道 w 键和 2 键紧邻（至少英文键盘如此），我们意外地同时击中了两个键。

输入代码的过程完全不同于使用图形用户界面，在图形用户界面里只需要从下拉菜单里选择变量单击或双击一个选项并/或者按下“运行”或

“完成”按钮。输入代码的优点是它会使你考虑输入什么、含义是什么，并且代码有更强的灵活性。主要的缺点是你需要知道输入什么。

R 有出色的图形工具。但同样你不能从方便的菜单里选择选项，而需要输入准确的代码或者从以前的项目复制代码。例如，如果想发现如何改变刻度线方向，可能需要搜索网络新闻组或者寻找在线手册。

1.4 脚本代码

1.4.1 编程的艺术

在本阶段，是否了解下面的代码并不重要，建议读者不必尝试输入代码。我们把它放在这里只是想说明，只要有一些努力，你就能用 R 生成非常漂亮的图形。

```
>setwd("C:/RBook/")
>ISIT<-read.table("ISIT.txt",header=TRUE)
>library(lattice)
>xyplot(Sources~SampleDepth|factor(Station),data=ISIT,
 xlab="Sample Depth",ylab="Sources",
 strip=function(bg='white', ...)
 strip.default(bg='white', ...),
 panel = function(x, y) {
 panel.grid(h=-1, v= 2)
 I1<-order(x)
 llines(x[I1], y[I1],col=1)})
```

从第三行(从 xyplot 开始)到最后，所有的代码组成一个单独的命令，因此我们只使用了一个“>”符号。在本节的后面，我们将提高该脚本代码的可读性。生成的图形在图 1.6 中给出，它给出了 19 个站点中深海浮游发光生物与深度的密度图。该数据是皇家发现号探险船在 2001 与 2002 年的一系列的四次巡航时，于爱尔兰西部的大西洋东北温带地区收集的(Gillibrand 等，2006)。生成图形花费了相当大的精力，但回报是，这个单个图形给出了所有的信息，并帮助确定应采用哪一种统计方法进行下一步的数据分析(Zuur 等，2009)。

1.4.2 录入脚本代码

除非你对计算代码有特殊的记忆能力，否则整块的 R 代码，例如用来生成图 1.6 的那些代码，几乎是不可能记住的。因此最重要的是把代码写

得尽可能简单和一般化并且细心地录入。仔细地录入代码可以使你在短短的几分钟内对别的数据集重新生成图形(或者别的分析),然而如果没有记录,你可能会脱离你自己的代码并且需要对一个完整的项目重新编程。作为一个例子,我们重新生成上节使用的代码,但是现在加一些注释。在符号“#”后的文本被 R 忽略。尽管我们还没有讨论 R 语法,但是代码开始给我们一些感觉。我们再次建议本阶段你不必尝试输入代码。

P.8

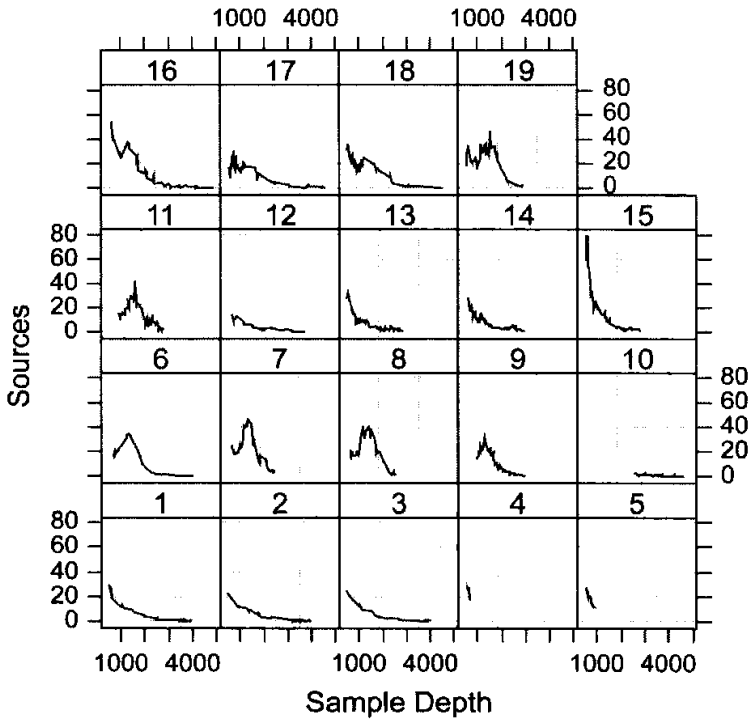


图 1.6 19 个站点中深海浮游发光生物与深度(单位:米)。数据来自 Zuur 等人(2009)的著作。允许 x 轴与 y 轴的坐标取不同的范围是相对容易的。数据由英国阿伯丁大学海洋实验室 Monty Priede 提供

```
>setwd("C:/RBook/")
>ISIT<-read.table("ISIT.txt",header=TRUE)
#Start the actual plotting
#Plot Sources as a function of SampleDepth, and use a
#panel for each station.
#Use the colour black (col=1), and specify x and y
#labels (xlab and ylab). Use white background in the
#boxes that contain the labels for station
```

P.9

```
>xyplot(Sources~SampleDepth|factor(Station),
  data = ISIT,xlab="Sample Depth",ylab="Sources",
  strip=function(bg='white', ...)
  strip.default(bg='white', ...),
  panel = function(x,y) {
      #Add grid lines
      #Avoid spaghetti plots
      #plot the data as lines (in the colour black)
        panel.grid(h=-1,v= 2)
        I1<-order(x)
        llines(x[I1],y[I1],col=1)})
```

尽管仍然难以理解代码是做什么的，但是我们至少可以观察它的一些结构。你可能已经注意到我们用空格说明哪些代码块在一起。这是常用的编程风格，并且对理解你的代码是很重要的。如果你不能理解你以前编写的代码，不要希望别人能理解！另外可以通过在命令、变量、逗号等附近添加空格来提高 R 的可读性。比较上面和下面的代码，你自己判断一下哪个看起来更容易理解。我们更喜欢下面的代码（再次说明，不必尝试输入代码）。

```
> setwd("C:/RBook/")
> ISIT <- read.table("ISIT.txt", header = TRUE)
> library(lattice) #Load the lattice package

#Start the actual plotting
#Plot Sources as a function of SampleDepth, and use a
#panel for each station.
#Use the colour black (col=1), and specify x and y
#labels (xlab and ylab). Use white background in the
#boxes that contain the labels for station
> xyplot(Sources ~ SampleDepth | factor(Station),
         data = ISIT,
         xlab = "Sample Depth", ylab = "Sources",
         strip = function(bg = 'white', ...)
         strip.default(bg = 'white', ...),
         panel = function(x, y) {
          #Add grid lines
          #Avoid spaghetti plots
          #plot the data as lines (in the colour black)
         panel.grid(h = -1, v = 2)
         I1 <- order(x)
         llines(x[I1], y[I1], col = 1)})
```

P.13 R 对格式是敏感的，编程需要用到大括号{}，小括号()，和中括号[]。开始的括号“{”与结束的括号“}”的正确配对以及在一项任务里正确使用括号的位置是很重要的。R 初学者的一些错误常与忘记括号或者使用错误的括号类型有关。Tinn-R 和 RWinEdt 使用颜色来显示匹配的括号，这

1.7 帮助文件和新闻组

当用 R 工作时，几乎对每一件任务你将有多重选择，并且，因为这里没有单独的资源去描述所有可能发生的事情，因此知道去哪里寻找帮助是非常重要的。假设你希望学习如何生成盒形图。在 R 中你最好的朋友是问号。输入：

```
> ?boxplot
```

并单击回车键。或者你也可以用：

```
> help(boxplot)
```

打开帮助窗口，将会出现标题为描述、用法、参数、详细、值、参考、另见和范例的文档。这些帮助文件不是“傻瓜指南”并且看起来有些吓人。我们推荐阅读描述部分，快速浏览用法部分（惊叹难以理解的选项），然后观

察范例了解 R 中盒形图的思想。复制一些例子代码并粘贴到 R 中。

下述的几行代码来自于帮助文件的范例。

```
> boxplot(count ~ spray, data = InsectSprays,
      col = "lightgray")
```

生成的盒形图如图 1.10 所示。语法 *count~spray*，确保昆虫喷雾剂的每个水平生成一个盒形图。昆虫喷雾剂数据的信息可以通过下述输入获得：

```
> ?InsectSprays
```

P. 14

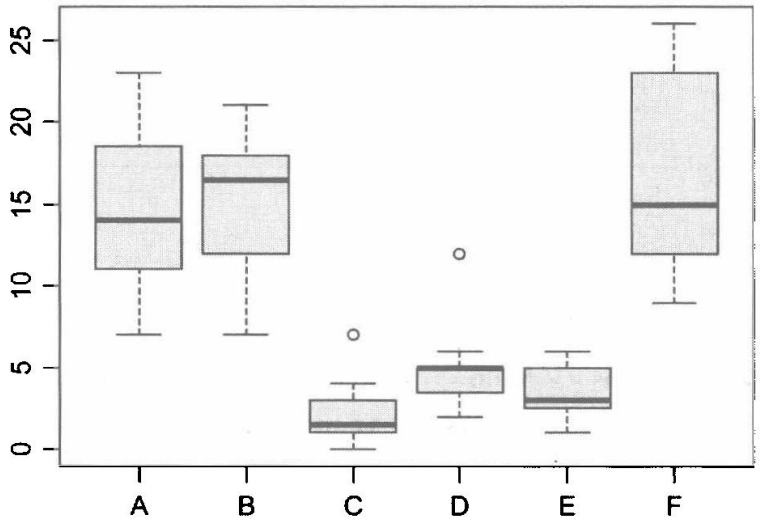


图 1.10 从盒形图帮助文件里复制并粘贴代码到 R 得到的盒形图。查看该图的数据来源，输入：? InsectSprays

重要的是复制整块的代码而不是仅包含该代码的部分片段。对于长块的代码，一般很难确定开始和结束点，有时需要猜测命令的特定结束位置在哪里。例如，如果你只是复制并粘贴了文本

```
> boxplot(count ~ spray, data = InsectSprays,
```

你将会看到一个“+”号(图 1.11)，提示 R 等待更多的代码。粘贴剩余的代码或者点击 Esc 键取消上述操作并继续复制粘贴完整的命令。

几乎所有的帮助文件都与 *boxplot* 函数的帮助文件具有相似的结构。

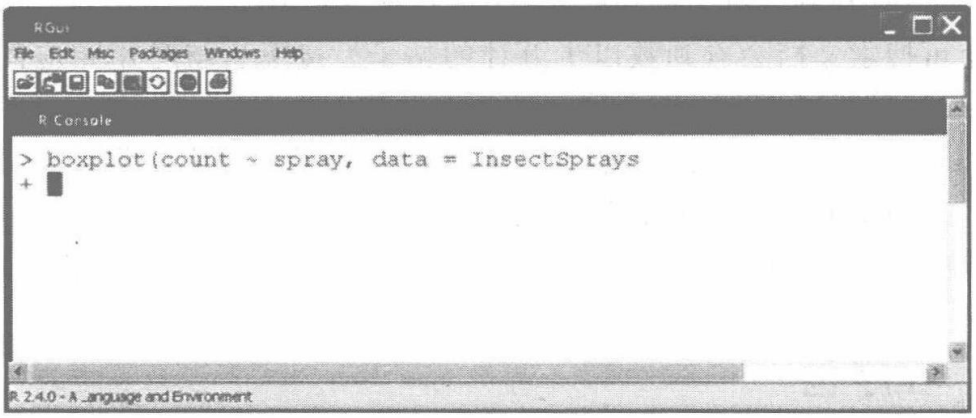


图 1.11 当不完整的命令输入时，R 会等待更多的代码。可以加入剩余的代码或者单击“Esc”键放弃绘制盒形图命令

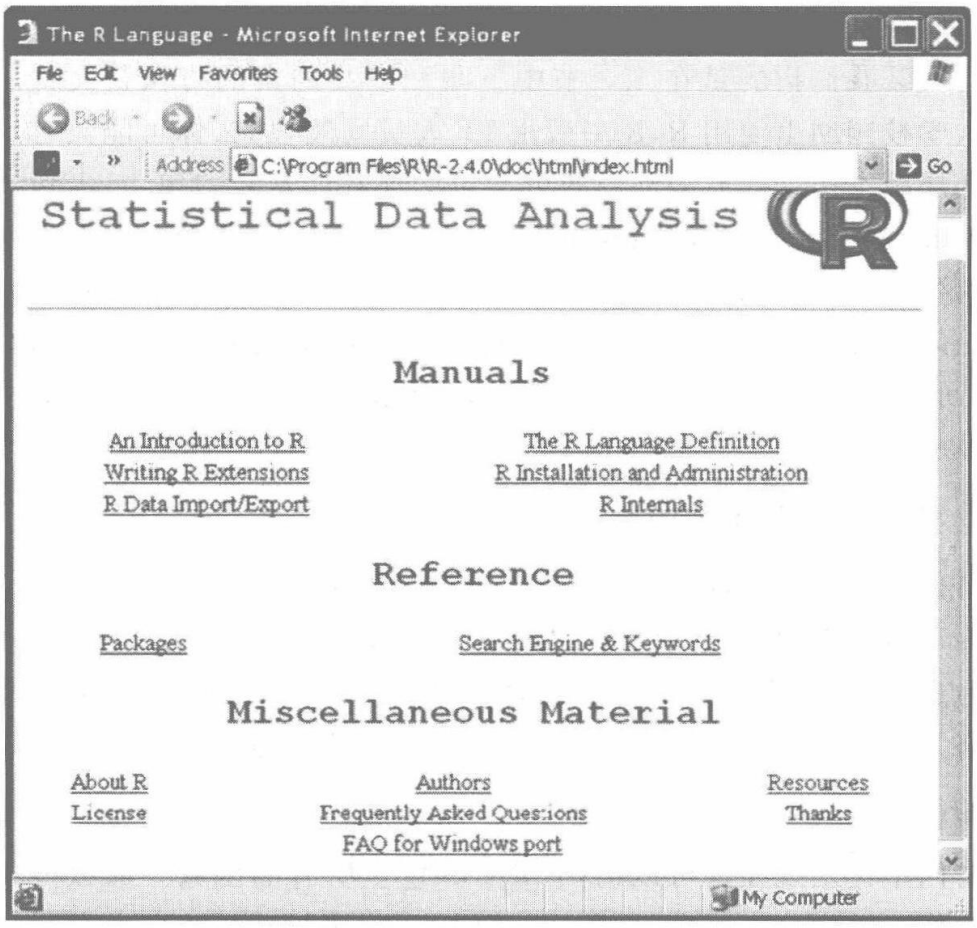


图 1.12 通过 R 的帮助菜单点击**帮助->Html 帮助(Help->Html help)**得到的窗口。搜索引擎和关键词允许搜索函数、命令和关键词。你需要关闭所有的弹出窗口拦截器

号下面的额外一行。

1.9.1 退出 R 和设置工作目录

另外一个有用的命令是： P.21

```
> q()
```

该命令退出 R。在退出之前，会询问是否保存工作空间。如果你决定保存，我们强烈建议你不要把它保存在默认目录下。如果这样做，当 R 重新启动时会自动载入所有的结果。为了避免 R 询问是否保存数据，使用：

```
> q(save = "no")
```

R 不做保存而直接退出。为了改变默认工作目录可以使用：

```
> setwd(file = "C:\\AnyDirectory\\")
```

这个命令只有在目录 AnyDirectory 存在时才起作用。选择一个合理的名称(我们的不是)。请注意在 Windows 操作系统里必须使用两个反斜杠。另一种可供选择的方法是使用：

```
> setwd(file = "C:/AnyDirectory/")
```

在目录结构中一般使用简单的名称。应该避免目录名称里包含字符比如 * ,&,o, $,£,“等。R 也不接受有变音符的字母符号比如 ä,í,á,ö,è,é 等。

我们推荐把 R 代码保存在文本编辑器中而不要保存在你的工作空间中。下次使用时，打开编辑好的存档文件，复制代码并把它粘贴到 R 里。你的结果和图形将会重新出现。保存你的工作空间只会使更多文件充斥于你的硬盘，或许一星期以后你将不记得如何得到所有的变量、矩阵等。从 R 代码里找回这些信息相对比较容易。唯一的例外是如果你的计算需要花费很长的时间去完成。如果是这种情况，建议把工作空间保存到你的工作目录里的某个地方。为了保存工作空间，点击**文件->保存工作空间(File -< Save Workspace)**，为了载入一个已经存在工作的空间，使用**文件->载入工作空间(File -< Load Workspace)**。

如果你想对不同的数据集开始一个新的分析，移除所有的变量可能是有用的。一种方法是退出 R 并重新启动。另一种方法是点击**其它->删除所有对象(Misc -< Remove all objects)**。可以通过下面的命令完成

```
> rm(list = ls(all = TRUE))
```

在**编辑(Edit)**菜单下还有一些其它有用的选项。例如，你可以点击**全选(Select all)**以复制所有的命令并输出到微软 Word 里。

对于引用程序包,例如格子包,你应该输入:

```
> citation("lattice")
```

它会给出如何引用该包的全部详细信息。在本书里,我们使用不同的包;我们提及和引用它们如下:foreign(R 核心成员等,2008),lattice(Sarkar,2008),MASS(Venables 和 Ripley,2002),nlme(Pinheiro 等,2008),plotrix(Lemon 等,2008),RODBC(Lapsley,2002;Ripley,2008),和 vegan(Oksanen 等,2008)。参考 R 本身是:R 开发核心小组(2008)。请注意根据使用 R 版本的不同,一些引用可能会有区别。 P.27

1.13 我们学习了哪些 R 函数?

每章我们用一节总结该章里介绍的 R 函数。本章我们只学习了一些命令。这里我们不重复发光格子函数和企鹅绘图函数,因为我们只是用它来做示例。表 1.1 列出了我们本章所讨论的函数。

表 1.1 第 1 章所介绍的 R 函数

函 数	功 能	示 例
?	访问帮助文件	? boxplot
#	添加注释	# Add your comments here
boxplot	生成盒形图	boxplot(y)boxplot(y~factor(x))
log	自然对数	log(2)
log10	以 10 为底的对数	log10(2)
library	载入包	library(MASS)
setwd	设置工作目录	setwd("C:/AnyDirectory")
q	关闭 R	q()
citation	提供对 R 的引用	citation()

第2章

R中的数据输入

P.29 在接下来的这一章，我们会讲述如何把数据录入R，并把数据系统地转化为标量（单值）、向量、矩阵、数据框或者列表。同时还将阐述如何从Excel、ascii文件、数据库和其它统计程序中载入数据。

2.1 R中的第1步

2.1.1 小型数据库中的数据录入

我们从把数量足够小的数据录入到R中的工作开始。使用这样一个数据库（康涅狄格大学Chris Elphick未公布的数据），它来自大概1100只沙鸥的7种身体测量数据（如头和翅膀的大小、踝骨的长度、体重等等）。为了方便起见，我们仅使用其中8只鸟的4种形态参数（见表2.1）。

表2.1 8只鸟的形态参数。符号NA代表缺失值，被测量的参数有翅膀的长度（翼弦的长度），腿的尺寸（踝骨尺寸），头的尺寸（从鸟嘴到后脑）和体重

翼弦	踝骨	头	体重
59	22.3	31.2	9.5
55	19.7	30.4	13.8
53.5	20.8	30.6	14.8
55	20.3	30.3	15.2
52.5	20.8	30.3	15.5

续表 2.1

翼弦	踝骨	头	体重
57.5	21.5	30.8	15.6
53	20.6	32.5	15.6
55	21.5	NA	15.7

将数据录入 R 最简单的一个方法就是以标量(仅含一个值的变量)的形式将数据一一输入,但此方法比较繁琐。比如,将翅膀长度的前五个观察值录入 R,需输入:

```
> a <- 59
> b <- 55
> c <- 53.5
> d <- 55
> e <- 52.5
```

P.30

程序中,符号"<-"可以用"="代替。并且,这些命令可以直接从文本编辑器中复制到 R 中,不需要其它改变。此时,若要查看 R 的计算结果,可以输入"a",然后敲回车键。

```
> a
[1] 59
```

正如我们所输入的,"a"的值为 59。这种方法的问题是大量的数据会很快用完所有的字母符号,并且,以 a、b、c 等字母作为变量名对刻画变量各表示什么意义是没有多大帮助的。因此,我们可以使用如下的变量名:

```
> Wing1 <- 59
> Wing2 <- 55
> Wing3 <- 53.5
> Wing4 <- 55
> Wing5 <- 52.5
```

若要输入剩余的数据,则需要更多的变量名。在改进变量名的处理方法前,我们先来讨论对于这些变量的操作。一旦定义了一个变量并且对其赋值后,我们就可以用它来进行计算,例如,下面这些都是有效的命令:

```
> sqrt(Wing1)
> 2 * Wing1
> Wing1 + Wing2
> Wing1 + Wing2 + Wing3 + Wing4 + Wing5
> (Wing1 + Wing2 + Wing3 + Wing4 + Wing5) / 5
```

此时,虽然 R 进行了计算,但它并没有存储结果,所以,最好定义新的变量:

```
> SQ.wing1 <- sqrt(Wing1)
> Mul.W1 <- 2 * Wing1
> Sum.12 <- Wing1 + Wing2
> SUM12345 <- Wing1 + Wing2 + Wing3 + Wing4 + Wing5
> Av <- (Wing1 + Wing2 + Wing3 + Wing4 + Wing5) / 5
```

当然，以上这些变量名仅仅是示范，你可以使用任何名字来代替，不过需要注意，“.”也是变量名的一部分。我们建议使用能帮助记忆具体代表什么的变量名，例如，SQ.wing1 表示第一只鸟翅膀长度的平方根。有时，在选择变量名的时候需要一定的想象力。但是注意，一些符号是不允许出现在变量名中的，例如“£，$，%，^，*，+，-，( )，[ ]，#，!，?，<，>”等，因为这些符号中的大部分都是运算符，比如乘法、乘幂等等。

P.31

根据如上所述，如果定义了

```
> SQ.wing1 <- sqrt(Wing1)
```

若要显示 SQ.wing1 的值，只需输入：

```
> SQ.wing1
[1] 7.681146
```

或者你可以把需要执行的命令放在圆括号内，R 将会即刻计算出结果：

```
> (SQ.wing1 <- sqrt(Wing1))
[1] 7.681146
```

2.1.2 应用c 函数连接数据

如上所述，对于 4 种形态参数的 8 个观测值，我们需要 32 个变量名。而 R 允许在一个变量中存储多个值，这个任务由 c( )函数来完成，这里 c 代表连接(Concatenate)。它可以这样使用：

```
> Wingcrd <- c(59, 55, 53.5, 55, 52.5, 57.5, 53, 55)
```

这里你可以在逗号的任意一边加上空格以增加代码的可读性，当然，也可以在“+”或“<-”任意一边加空格。总的来说，只要能使代码便于识别，这些做法都是允许的。

需要注意的是这里 c( )函数使用的是圆括号( )，不是方括号[ ]或者花括号{ }，它们具有其它的作用。

当然，如前面所讲，你也可以把这些命令从其它文本粘贴到 R 中。如果需要查看结果，只需要输入 Wingcrd，再回车就可以了：

```
> Wingcrd
[1] 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0
```

这时，c( )函数生成了一个长度是 8 的向量。如果需要查看 Wingcrd 的第一个值，可以输入 Wingcrd[1]，然后回车：

```
> Wingcrd [1]
[1] 59
```

P.32

结果是 59，如果需要查看 Wingcrd 的前五个值，可以输入：

```
> Wingcrd [1 : 5]
[1] 59.0 55.0 53.5 55.0 52.5
```

如果需要查看除了第二个值之外的其它值，可以输入：

```
> Wingcrd [-2]
[1] 59.0 53.5 55.0 52.5 57.5 53.0 55.0
```

可以看到，负号删除了这个值。R 有很多内置的函数，最基本的有例如 sum，mean，max，min，median，var 和 sd 等等，可以通过如下的方法来使用它们，

```
> sum(Wingcrd)
[1] 440.5
```

显然，我们可以将这个和存在一个新的变量中，

```
> S.win <- sum(Wingcrd)
> S.win
[1] 440.5
```

再次强调，这里“.”也是变量名的一部分。现在，我们来将表 2.1 中剩下的 3 个形态参数输入 R。这一步比较麻烦，可以选择先将它们输入到一个文本编辑器，再粘贴到 R 中的办法。

```
> Tarsus <- c(22.3, 19.7, 20.8, 20.3, 20.8, 21.5, 20.6,
              21.5)
> Head <- c(31.2, 30.4, 30.6, 30.3, 30.3, 30.8, 32.5,
            NA)
> Wt <- c(9.5, 13.8, 14.8, 15.2, 15.5, 15.6, 15.6,
          15.7)
```

这里我们付出了额外的空间，即每一个命令都占用了两行，只要你在第一行结束的时候使用反斜线或者是逗号，R 都能识别这是一条命令。

一般来说，R 中的变量名最好使用大写字母开头，这样可以避免将它和一些内部函数名混淆，因为大部分内部函数都不是以大写字母开头的，例如，“head”是 R 中的一个内部函数(见？head)，所以这里我们需要使用变量名 Head。如果还不放心的话，可以键入？Head，如果出现帮助文件的话，你就需要另换一个变量名了。

P.33　需要注意的是这里有一只鸟的头的尺寸是没有测量的，我们用 NA 来表示，这时如果调用内部函数，NA 的出现可能导致计算结果的错误，例如：

```
> sum(Head)
[1] NA
```

当然，调用其它如 mean，min，max 等函数会得到同样错误的结果。为了弄清楚为什么得到这样的错误结果，可以键入？sum，在 sum 的帮助文件中我们得到了如下的相关内容。

```
...
sum(..., na.rm = FALSE)
...
If na.rm is FALSE, an NA value in any of the arguments
will cause a value of NA to be returned, otherwise NA
values are ignored.
...
```

显然可以看到，在向量中如果有一个缺失值的话，默认选项“na. rm = FALSE”将会导致 R 函数 sum 返回 NA(rm 表示移除(remove))，为了避免这种情况，我们可以使用“na. rm = TRUE”，

```
> sum(Head, na.rm = TRUE)
[1] 216.1
```

此时，剩下 7 个值的和就被返回了，同样，可以用此方法来处理 mean，min，max，median 等函数。在大部分电脑上，你可以使用 na. rm = T 来代替 na. rm = TRUE。然而，由于我们面对教室内装有相同操作系统运行相同 R 版本的电脑时，有一些电脑也对 na. rm = T 指令提示错误，所以，还是建议使用 na. rm = TRUE 指令。　建议写完整，写成TRUE，而不是简写T

在 R 中使用内部函数前最好查看一下相应的帮助文件，以确保你知道这个函数如何处理缺失值，因为要把所有函数处理缺失值的方法都记住是不太可能的，有些函数使用 na. rm，有些使用 na. action，而有些使用其它的句式。

至此，我们已经完成了 4 个变量的输入，并且使用了一些简单的函数，例如 mean，min，max 等。接着，我们将考虑如何连接这 4 个变量中的数据：(1)c，cbind 和 rbind 函数；(2)matrix 和 vector 函数；(3)数据框；(4)列表。

练习使用 c 和 sum 函数完成 2. 4 节的习题 1。

2.1.3　使用c，cbind 和rbind 结合变量

P.34　我们已经有了 4 列数据，每列含有 8 只鸟的观察值，这 4 列数据分别以

变量 Wingcrd,Tarsus,Head 和 Wt 来标记。c 函数可以用来连结这些数据，同时连结这些数据中的 8 个值，具体操作如下：

```
> BirdData <- c(Wingcrd, Tarsus, Head, Wt)
```

我们使用了变量名 BirdData，而没有使用 data，因为 data 是 R 中的一个内部函数（见?data），这样做可以避免覆盖原函数 data 的值。键入 BirdData，并回车，可以看到这条命令的结果：

```
> BirdData
 [1] 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0 22.3
[10] 19.7 20.8 20.3 20.8 21.5 20.6 21.5 31.2 30.4
[19] 30.6 30.3 30.3 30.8 32.5   NA   9.5 13.8 14.8
[28] 15.2 15.5 15.6 15.6 15.7
```

BirdData 是一个长度为 32(4×8)的单个向量，符号[1]、[10]、[19]和[28]表示新的一行的第一个元素的索引编号，根据电脑显示器的大小不同这些编号在不同的电脑上可能有所不同，你不需理会这些数字。

此时，R 将包含缺失值的 32 个观察值生成了一个单个向量，并没有区分这些值都属于哪一个变量（前 8 个值属于变量 Wingcrd，第二组 8 个值属于变量 Tarsus 等等）。为了实现这一点，我们可以生成一个长度是 32 的向量，命名为 Id（表示“identity”），给它赋如下这些值。

```
> Id <- c(1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2,
      2, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4)
> Id
 [1] 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
[24] 3 4 4 4 4 4 4 4 4
```

相对于 BirdData，R 可以使 Id 向量在一行中显示更多的数字，只出现了[1]和[24]两个索引编码，这些索引编码在此时是完全不相关的。Id 向量的作用是指出具有相似 Id 值的观察值属于同一种形态变量。然而，当针对大数据库的时候，生成这样的一个向量是很费时间的，幸运的是，R 具有简化这个过程的函数，我们需要的函数是重复地输入值 1～4，每个 8 次：

```
> Id <- rep(c(1, 2, 3, 4), each = 8)
> Id
 [1] 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
[24] 3 4 4 4 4 4 4 4 4
```

P.35

这个函数产生了上述相同的结果，符号 rep 代表重复（repeat），它的使用还可以简化为：

```
> Id <- rep(1 : 4, each = 8)
> Id
 [1] 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
[24] 3 4 4 4 4 4 4 4 4
```

这和前面的结果都是相同的，我们可以通过键入如下的指令来查看1：4命令的作用：

```
> 1 : 4
```

将出现

```
[1] 1 2 3 4
```

显然，符号：并不是除号的意思(在其它程序包中其具有相同的意义)。此外，你还可以使用 seq 函数来实现这个目的，例如，命令

```
> a <- seq(from = 1, to = 4, by = 1)
> a
```

同样可以产生如下序列：

```
[1] 1 2 3 4
```

因此，对于前面所述的鸟类的观察值，我们同样可以使用如下的命令：

```
> a <- seq(from = 1, to = 4, by = 1)
> rep(a, each = 8)
 [1] 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
[24] 3 4 4 4 4 4 4 4 4
```

rep 函数使得“a”中的每个数字重复了 8 次。此时，你可能认为我们使用了过多的方法，从而使事情变得无谓的复杂起来。但是，在 R 中一些函数的使用需要提供类似于表 2.1 那样的数据(例如，对于主成分分析或多维尺度分析的多元分析函数)，而另一些函数的使用需要提供一个单独向量和识别这组观察值的一个变量(例如上述的 Id)，这样的函数主要包括 t-检验，单因子方差分析，线性回归，以及一些作图工具，如 lattice 包中的 xyplot(见第 8 章)。所以，熟练地使用 rep 函数将可以节省很多时间。

P.36 至此，我们仅仅实现了数字的连结，假如我们想生成一个长度为 32 的向量“Id”，这个向量包含了单词“Wingcrd”8 次，“Tarsus”8 次，等等。我们可以先产生一个名为 VarNames 的新变量，这个变量包含了前面所提到的 4 个形态参数：

```
> VarNames <- c("Wingcrd", "Tarsus", "Head", "Wt")
> VarNames
[1] "Wingcrd" "Tarsus" "Head" "Wt"
```

注意这些都只是名称，而不是含有数值的变量。然后，我们再利用 rep 函数

来生成所需要的向量：

```
> Id2 <- rep(VarNames, each = 8)
> Id2
 [1] "Wingcrd" "Wingcrd" "Wingcrd" "Wingcrd"
 [5] "Wingcrd" "Wingcrd" "Wingcrd" "Wingcrd"
 [9] "Tarsus"  "Tarsus"  "Tarsus"  "Tarsus"
[13] "Tarsus"  "Tarsus"  "Tarsus"  "Tarsus"
[17] "Head"    "Head"    "Head"    "Head"
[21] "Head"    "Head"    "Head"    "Head"
[25] "Wt"      "Wt"      "Wt"      "Wt"
[29] "Wt"      "Wt"      "Wt"      "Wt"
```

这里 Id2 是一个被赋予了具有固定顺序的名字的字符串，它和 Id 的区别仅仅是所包含内容的名称不同而已。注意这里不能丢掉"each = "，否则，你将得到：

```
> rep(VarNames, 8)
 [1] "Wingcrd" "Tarsus" "Head" "Wt"
 [5] "Wingcrd" "Tarsus" "Head" "Wt"
 [9] "Wingcrd" "Tarsus" "Head" "Wt"
[13] "Wingcrd" "Tarsus" "Head" "Wt"
[17] "Wingcrd" "Tarsus" "Head" "Wt"
[21] "Wingcrd" "Tarsus" "Head" "Wt"
[25] "Wingcrd" "Tarsus" "Head" "Wt"
[29] "Wingcrd" "Tarsus" "Head" "Wt"
```

它是将整个包含 4 个变量名的向量 VarNames 重复循环了 8 次，并不是这里我们所要的结果。

c 函数是我们结合数据或者变量的一种选择，另一种选择是 cbind 函数，它的作用是将所结合的变量以列的形式输出，例如，我们将 cbind 函数的输出存储在变量 Z 中，然后键入 Z 并回车，将看到以列的形式显示的数值：

P.37

```
> Z <- cbind(Wingcrd, Tarsus, Head, Wt)
> Z
     Wingcrd   Tarsus  Head    Wt
[1,]    59.0     22.3  31.2   9.5
[2,]    55.0     19.7  30.4  13.8
[3,]    53.5     20.8  30.6  14.8
[4,]    55.0     20.3  30.3  15.2
[5,]    52.5     20.8  30.3  15.5
[6,]    57.5     21.5  30.8  15.6
[7,]    53.0     20.6  32.5  15.6
[8,]    55.0     21.5    NA  15.7
```

当我们有某些特殊要求的时候，这样输出数据将是很必要的，例如，需要做主成分分析时。假设需要访问Z的第一列，则可以使用命令Z[,1]：

```
> Z[, 1]
[1] 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0
```

同样，也可以使用：

```
> Z[1 : 8, 1]
[1] 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0
```

结果是一样的。如需访问第二行，则可以输入：

```
> Z[2, ]
Wingcrd    Tarsus      Head      Wt
   55.0      19.7      30.4    13.8
```

也可以输入：

```
> Z[2, 1:4]
Wingcrd    Tarsus      Head      Wt
   55.0      19.7      30.4    13.8
```

如下所列的命令都是有效的。

```
> Z[1, 1]
> Z[, 2 : 3]
> X <- Z[4, 4]
> Y <- Z[, 4]
> W <- Z[, -3]
> D <- Z[, c(1, 3, 4)]
> E <- Z[, c(-1, -3)]
```

第一条命令访问的是第一只鸟的Wingcrd值；第二条命令给出的是第二列和第三列的全部数据；X表示的是第4只鸟的Wt值；Y列出了所有鸟的Wt值。负号在这里表示不包含所描述的列或者行，因此，W包含的是除了Head值之外的所有数据。我们还可以利用c函数来访问Z中无序的列或
P.38 者行，也就是说，D包含了Z中第一、第三和第四列的数据，E包含了除去第一列和第三列的数据。不过，必须保证所输入的下标值没有超过变量允许的范围，例如，Z[8,4]这种表示是有效的，但是Z[9,5]，Z[8,6]或者Z[10,20]都是没有定义的(我们只有8只鸟和4个形态参数变量)，如果你输入了这些命令，R将会显示如下的错误提示：

```
Error: subscript out of bounds
```

如果想知道Z的维数，可以使用：

```
> dim(Z)
[1] 8 4
```

此命令的输出是包含了两个元素的一个向量:分别表示 Z 的行数和列数。你还可以通过参考 nrow 和 ncol 的帮助文件来得到另外的可以实现此功能的办法。有时,将 dim 函数的输出存储起来会更加有用,可以使用

```
> n <- dim(Z)
> n
  [1] 8 4
```

或者,若仅需要存储 Z 的行数,可以使用

```
> nrow <- dim(Z)[1]
> nrow
  [1] 8
```

相比于 nrow,可能以 zrow 作为变量名会更合适。与 cbind 函数将变量以列的形式进行整理的功能类似,rbind 函数具有将数据以行进行结合的作用,我们可以这样使用它:

```
> Z2 <- rbind(Wingcrd, Tarsus, Head, Wt)
> Z2
        [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
Wingcrd 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0
Tarsus  22.3 19.7 20.8 20.3 20.8 21.5 20.6 21.5
Head    31.2 30.4 30.6 30.3 30.3 30.8 32.5   NA
Wt       9.5 13.8 14.8 15.2 15.5 15.6 15.6 15.7
```

这些数据和前面所讲的是相同的,只不过行表示形态变量而列表示鸟。

此外,edit 函数和 fix 函数都是可以对 Z 或者 Z2 进行操作的有用的工具,具体的使用方法可以参考帮助文件。

练习使用 c 和 cbind 函数完成 2.4 节的习题 2,这个习题使用的是一个流行病学数据库。

2.1.4 使用vector 函数结合数据*

P.39

为了避免引入过多的信息,我们在前面的学习中没有涉及到 vector 函数,初学者可以跳过这一部分内容。vector 函数的作用与 c 函数类似,它可以用来代替 c 函数。假如我们要在 R 中生成一个长度为 8,包含了所有 8 只鸟的 Wingcrd 数据的一个向量,我们可以像如下这么做。

```
> W <- vector(length = 8)
> W[1] <- 59
> W[2] <- 55
> W[3] <- 53.5
> W[4] <- 55
> W[5] <- 52.5
> W[6] <- 57.5
> W[7] <- 53
> W[8] <- 55
```

如果在第一条命令之后直接键入W,你将会得到一个FALSE值的向量,所以,必须在所有元素的值被输入之后再键入W:

```
> W
[1] 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0
```

这个结果和使用c函数所得到的结果是一样的,而vector函数的优点是我们可以事先定义向量的长度,这一点有时是很有用的,例如做循环运算的时候。但是,一般情况下还是使用c函数结合数据比较简单。

与c函数的输出类似,我们可以使用W[1],W[1:4],W[2:6],W[-2],W[c(1,3,5)]之类的指令来访问W中的一些特定的元素,同样,W[9]这样的指令将会输出NA,因为第9个元素是没有定义的。

练习使用vector函数完成2.4节的习题3,这个习题使用的是一个流行病学数据库。

2.1.5 使用矩阵结合数据*

初学者可以跳过这一部分内容。

为了代替向量显示4个变量Wingcrd,Tarsus,Head和Wt,每个长度为8,我们可以生成一个8×4的矩阵包括上述数据,该矩阵的生成可以通过如下命令:

P.40

```
> Dmat <- matrix(nrow = 8, ncol = 4)
> Dmat
     [,1] [,2] [,3] [,4]
[1,]   NA   NA   NA   NA
[2,]   NA   NA   NA   NA
[3,]   NA   NA   NA   NA
[4,]   NA   NA   NA   NA
[5,]   NA   NA   NA   NA
[6,]   NA   NA   NA   NA
[7,]   NA   NA   NA   NA
[8,]   NA   NA   NA   NA
```

起初，我们想将这个矩阵命名为 D，后来发现这个 D 在 Tinn-R 中是蓝色的字体，这表示它是一个已经存在的函数名。输入？D，可以看到它表示的是一个计算导数的函数，为了不覆盖它，我们使用记号“Dmat”，这里“mat”代表矩阵(matrix)。

注意到这里的 Dmat 是一个仅含有 NA 的 8×4 的矩阵，需要填入适当的数值，我们可以这样来进行操作：

```
> Dmat[, 1] <- c(59, 55, 53.5, 55, 52.5, 57.5, 53, 55)
> Dmat[, 2] <- c(22.3, 19.7, 20.8, 20.3, 20.8, 21.5,
                 20.6, 21.5)
> Dmat[, 3] <- c(31.2, 30.4, 30.6, 30.3, 30.3, 30.8,
                 32.5, NA)
> Dmat[, 4] <- c(9.5, 13.8, 14.8, 15.2, 15.5, 15.6,
                 15.6, 15.7)
```

这种情况下，Dmat 的值是以列的形式输入的，同样，我们也可以以行的形式输入。键入 Dmat，可以得到与使用 cbind 函数类似的结果，只是 Dmat 没有列标签。

```
> Dmat
     [,1] [,2] [,3] [,4]
[1,] 59.0 22.3 31.2  9.5
[2,] 55.0 19.7 30.4 13.8
[3,] 53.5 20.8 30.6 14.8
[4,] 55.0 20.3 30.3 15.2
[5,] 52.5 20.8 30.3 15.5
[6,] 57.5 21.5 30.8 15.6
[7,] 53.0 20.6 32.5 15.6
[8,] 55.0 21.5   NA 15.7
```

我们可以使用 colnames 函数来给 Dmat 的列加上名称： P.41

```
> colnames(Dmat) <- c("Wingcrd", "Tarsus", "Head","Wt")
> Dmat

     Wingcrd Tarsus Head   Wt
[1,]    59.0   22.3 31.2  9.5
[2,]    55.0   19.7 30.4 13.8
[3,]    53.5   20.8 30.6 14.8
[4,]    55.0   20.3 30.3 15.2
[5,]    52.5   20.8 30.3 15.5
[6,]    57.5   21.5 30.8 15.6
[7,]    53.0   20.6 32.5 15.6
[8,]    55.0   21.5   NA 15.7
```

显然，rownames 函数也是存在的，可以参考帮助文件来查看它的使用

方法。

简而言之，我们利用矩阵结合数据的步骤是，首先，定义了一个具有特定大小的矩阵，然后以列的形式对其元素进行了赋值。注意，在赋值之前必须先定义矩阵。当然，你也可以一个元素一个元素的进行赋值，例如，

```
> Dmat[1, 1] <- 59.0
> Dmat[1, 2] <- 22.3
```

等等，但是，这样做会很浪费时间。如果我们的数据已按照变量进行了分类，例如 Wingcrd，Tarsus，Head，Wt，我们可以不使用常规的方法进行矩阵的定义与赋值，下面的命令会更加好用：

```
> Dmat2 <- as.matrix(cbind(Wingcrd, Tarsus, Head, Wt))
```

Dmat2 和 Dmat 是完全相同的。这种使用不止一种方法来解决这个问题的思想是很必要的，因为有些函数需要使用矩阵作为输入，当使用数据框(见下一节)时就会提示错误，因此，如果能掌握这种思想，有些函数，例如 as.matrix，is.matrix(这些函数的参数如果是矩阵将会提示 TRUE，否则将会提示 FALSE)，as.data.frame，is.date.frame 将会随时给你提供很大的帮助。

对于矩阵 A 和 B 专门的操作符还有进行转置运算的 t(A)，进行矩阵乘法的 A %*% B，求逆矩阵的 solve(A)等。

练习处理 2.4 节习题 4 中的矩阵。

2.1.6 使用data.frame 函数结合数据

P.42 目前为止，我们已经使用了 c，cbind，rbind，vector 和 matrix 函数来结合数据，另外一个可供选择的就是数据框，我们可以使用数据框结合具有相同长度的变量，而数据框的每一行就包含有同一样本的不同观察值，这一点上它和 matrix 或者 cbind 函数是比较类似的。使用前面所讲的鸟的 4 种形态参数，可以这样生成一个数据框：

```
> Dfrm <- data.frame(WC = Wingcrd,
                     TS = Tarsus,
                     HD = Head,
                      W = Wt)
> Dfrm
   WC   TS   HD    W
1 59.0 22.3 31.2  9.5
2 55.0 19.7 30.4 13.8
```

```
3 53.5 20.8 30.6 14.8
4 55.0 20.3 30.3 15.2
5 52.5 20.8 30.3 15.5
6 57.5 21.5 30.8 15.6
7 53.0 20.6 32.5 15.6
8 55.0 21.5   NA 15.7
```

data.frame 函数在这里创建了一个名为 Dfrm 的对象，而 Dfrm 里存储了鸟的四种形态参数的值，这是这个函数最基本的用法。数据框的优点是可以在不影响原始数据的基础上改变数据，例如，我们可以在数据框 Dfrm 中结合原始(已重命名)的体重值和体重值的均方根：

```
> Dfrm <- data.frame(WC = Wingcrd,
                     TS = Tarsus,
                     HD = Head,
                      W = Wt
                    Wsq = sqrt(Wt))
```

在数据框中，我们也可以结合数值变量、字符串和因子，因子是一种名义(分类)变量，后面将会讨论它。

需要注意，在 c 函数中所生成的变量 Wt 和数据框 Dfrm 中的变量 W 是两个不同的实体，为了验证这一点，我们移除变量 Wt(这是在 c 函数中输入的变量)：

```
> rm(Wt)
```

如果此时再键入 Wt，R 将会提示错误：

```
> Wt
Error: object "Wt" not found
```

P.43

但是变量 W 却还存在于数据框 Dfrm 中：

```
> Dfrm$W
[1] 9.5 13.8 14.8 15.2 15.5 15.6 15.6 15.7
```

数据框较之 cbind 函数和 matrix 函数具有可以结合不同类型的数据的功能，所以，它的使用还是很必要的。我们通常这样使用数据框，首先，我们向 R 中输入数据，主要使用 2.2 节的方法，然后，对数据做一些改变(例如移除极端值，应用变换，增加分类变量等等)，再将数据存入数据框中以备后续分析的使用。

2.1.7 使用list 函数结合数据*

初学者可以跳过这一部分内容。目前为止，我们所学的结合数据的工具都是生成一个表格，表格的每一行代表一个样本单元(例如一只鸟)。假

设现在需要这样一个黑盒子,这个盒子中可以放入尽可能多的各种各样的变量:一些可能是相关的,一些可能具有相似的维数,一些可能是向量,一些是矩阵,一些可能是包含有变量名的字符串,这就是 list 函数可以完成的功能。它区别于我们以前所用的方法的最大不同点就是它的每一行不仅仅代表一个样本单元。举一个简单的例子,变量 x1,x2,x3 和 x4 都包含有一些数据:x1 是一个长为 3 的向量,x2 包含 4 个字符,x3 是一个一维变量,x4 是一个 2×2 的矩阵,而所有的这些变量都可以输入到一个 list 函数中:

```
> x1 <- c(1, 2, 3)
> x2 <- c("a", "b", "c", "d")
> x3 <- 3
> x4 <- matrix(nrow = 2, ncol = 2)
> x4[, 1] <- c(1, 2)
> x4[, 2] <- c( 3, 4)
> Y <- list(x1 = x1, x2 = x2, x3 = x3, x4 = x4)
```

此时再键入 Y,可以得到以下结果:

P.44

```
> Y

$x1
[1] 1 2 3

$x2
[1] "a" "b" "c" "d"

$x3
[1] 3

$x4
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

所有包含在 Y 中的信息都是可以通过键入相应的指令来访问的,例如,Y$x1,Y$x2 等等。我们之所以引入 list 函数的原因是因为几乎所有 R 中的函数(比如线性回归,广义线性模型,t-检验等等)的输出结果都是保存在列表中的。例如,如下代码应用线性回归模型实现将翅膀长度表示为体重的函数。

```
> M <- lm(WC ~ Wt, data = Dfrm)
```

我们不需要知道 lm 函数具体是如何执行的,或者 R 是怎样实现线性回归的(可以通过键入? lm 查看相应的帮助文件),我们所要强调的是 R 将

线性回归函数的所有结果都存储在了 M 中，如果键入

```
> names(M)
```

将得到如下这些奇特的输出结果：

```
 [1] "coefficients"  "residuals"      "effects"
 [4] "rank"          "fitted.values"  "assign"
 [7] "qr"            "df.residual"    "xlevels"
[10] "call"          "terms"          "model"
```

我们可以通过键入 M $ coefficients，M $ residuals 等命令来分别访问 coefficients，residuals 等数据。所以，可以说 M 是一个包含了不同类型数据的列表，和上面所提及的 Y 是类似的。比较好的一点是 R 提供了各种各样的函数来提取所需要的列表中的信息（比如估计值，p -值等等），具体可参考 lm 的帮助文件。

对于前面表 2.1 所给出的鸟的形态参数的数据，由于其每一行都表示同一只鸟的数据，所以将其存入一个列表中是没有多大的意义的。然而，当我们的任务是生成一个列表，表中需要将所有数据放在一个长向量中，还需要另一个向量来识别这些变量（例如 ID 的作用），需要一个 8×4 的矩阵来表示这些数据，并且还需要一个包含了 4 种形态参数名称的向量时，我们可以这样来处理它： P.45

```
> AllData <- list(BirdData = BirdData, Id = Id2, Z = Z,
                  VarNames = VarNames)
```

结果将是：

```
> AllData
$BirdData
 [1] 59.0 55.0 53.5 55.0 52.5 57.5 53.0 55.0 22.3
[10] 19.7 20.8 20.3 20.8 21.5 20.6 21.5 31.2 30.4
[19] 30.6 30.3 30.3 30.8 32.5   NA  9.5 13.8 14.8
[28] 15.2 15.5 15.6 15.6 15.7

$Id
 [1] "Wingcrd" "Wingcrd" "Wingcrd" "Wingcrd"
 [5] "Wingcrd" "Wingcrd" "Wingcrd" "Wingcrd"
 [9] "Tarsus"  "Tarsus"  "Tarsus"  "Tarsus"
[13] "Tarsus"  "Tarsus"  "Tarsus"  "Tarsus"
[17] "Head"    "Head"    "Head"    "Head"
[21] "Head"    "Head"    "Head"    "Head"
[25] "Wt"      "Wt"      "Wt"      "Wt"
[29] "Wt"      "Wt"      "Wt"      "Wt"
```

```
$Z
     Wingcrd  Tarsus  Head   Wt
[1,]    59.0    22.3  31.2  9.5
[2,]    55.0    19.7  30.4 13.8
[3,]    53.5    20.8  30.6 14.8
[4,]    55.0    20.3  30.3 15.2
[5,]    52.5    20.8  30.3 15.5
[6,]    57.5    21.5  30.8 15.6
[7,]    53.0    20.6  32.5 15.6
[8,]    55.0    21.5    NA 15.7

$VarNames
 [1]"Wingcrd""Tarsus" "Head" "Wt"
```

显然，以这种形式来存储数据并不是必须的，我们仅需要其中一种就足够了。但是，这种多样的存储形式为我们将来在这些数据上使用很多其它的函数提供了方便。然而，我们的程序设计方式还是仅在需要的时候对数据进行转化。

P.46 此时，在R中键入AllData，就可以看到我们在这一部分所涉及的大部分数据格式，能做到这一点是非常不错的。

在list函数中不能使用“<-”符号，只能使用“=”。图2.1给出了我们

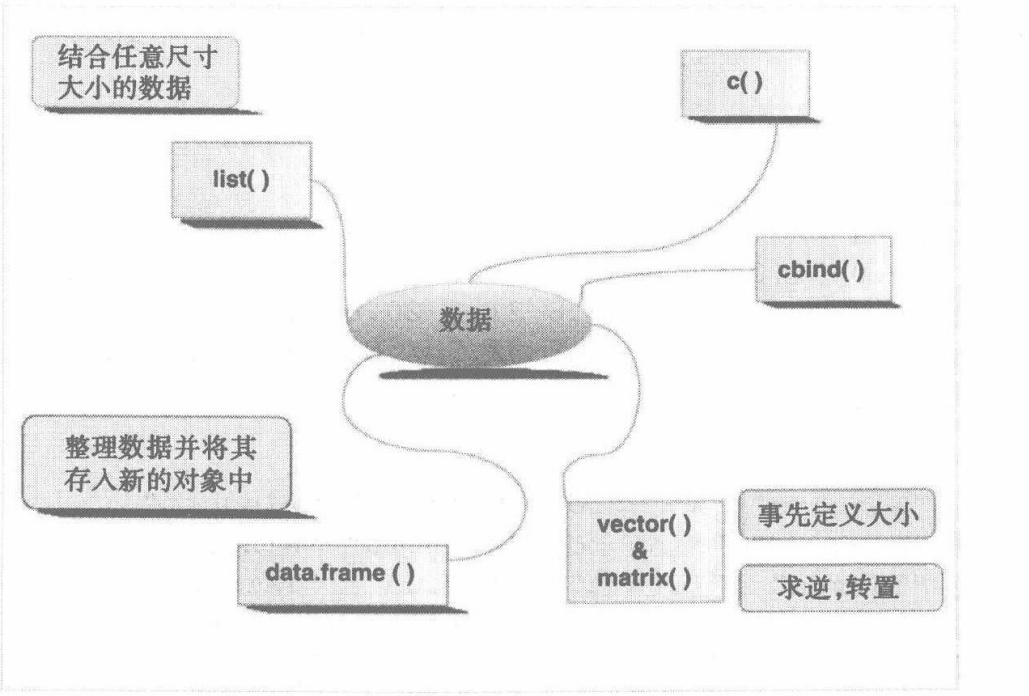


图2.1 各种数据存储方法的总结。当数据由cbind，matrix，或者data.frame存储时假设数据的每一行代表同一种观察值（比如一个样本）

迄今为止所学的数据存储方法的一个总结。

练习 2.4 节习题 5，此题考查使用 data.frame 和 list 命令处理流行病学数据库。

2.2 数据的载入

对于大型的数据库，像我们前面所讲的将其逐个键入 R 中是很不现实的。而学习一个新程序包中最困难的就是如何载入数据了，不过如果你掌握了这一步，你就可以很轻松的进行其它操作了。接下来的内容将讲述各种各样的载入数据的方法，我们将数据分为大型数据库和小型数据库以区别对待，并考虑它们是否存储于 Excel、ascii 文本文件、数据库程序，或者其它统计包中。

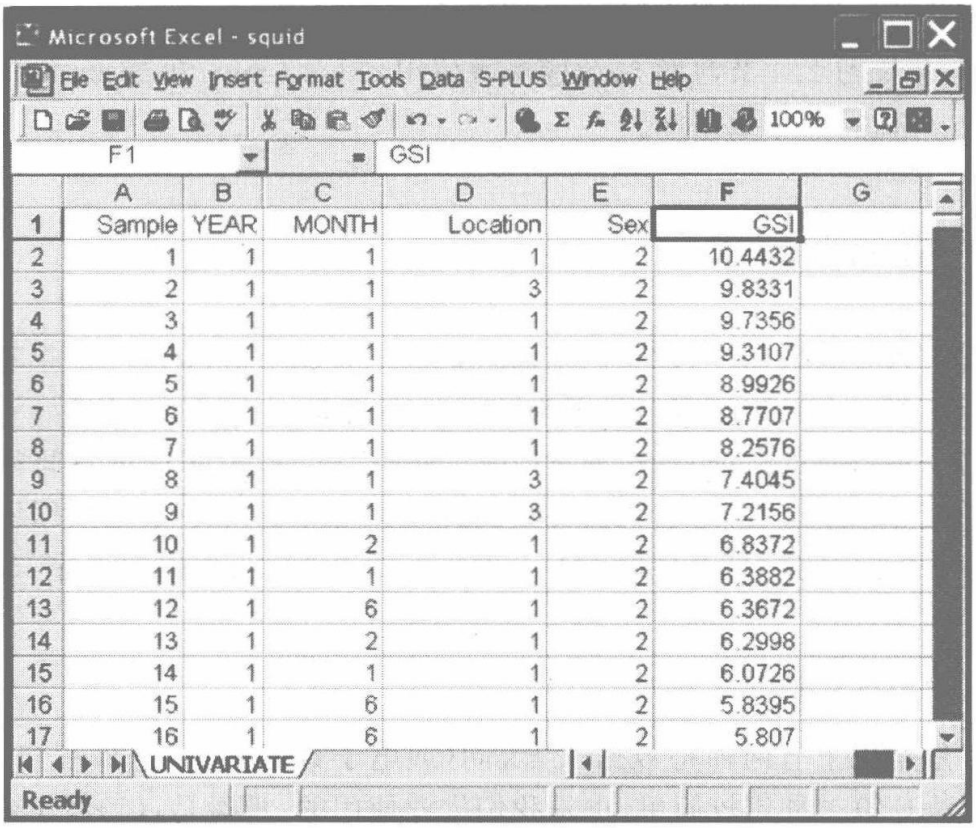
2.2.1 Excel 中的数据载入

一般情况下有两种将数据从 Excel（或电子数据表、数据库程序）载入 R 的方法。第一种比较简单，也是我们推荐使用的，步骤是：(1)将 Excel 中的数据准备好，(2)将其提取到制表符分隔的 ascii 文件中，(3)关闭 Excel，(4)使用 read.table 函数将数据载入到 R 中，接下来的内容将详细介绍这其中的每一步。第二种方法是一个专门的 R 程序包，RODBC，它可以访问 Excel 中选定的行和列。应该注意的是 Excel 并不是最适合于处理大型数据库的软件，因为它的列是有限制的。 P.47

2.2.1.1 Excel 中的数据准备

为了简单起见，我们建议将数据排列为样本-变量的形式，也就是说，列表示各种变量，行表示各种样本、观察值、案例、对象或者其它你称之为样本单元的东西。以 NA（大写）表示缺失值，一般最好以 Excel 中的第一列来识别样本单元，第一行作为变量名。与前面的叙述一致，最好避免使用包含如下一些符号的名称：£，$，%，^，&，*，(，)，－，#，?，'，.，<，>，/，|，\，[，]，{，和}，同样，也要避免使用包含空格的名称（字段或数值）。尽量使用简单的名称，不要太长，否则将会使图表中因为包含太长名字而不易识别。

图 2.2 中的 Excel 电子数据表是一组乌贼的性腺指数（GSI，即与身体总重量有关的性腺重量）的数据库（来自英国阿伯丁大学 Graham Pierce 的未发表的数据资料），各种数据均测量于苏格兰地区不同年月捕获的乌贼。


Microsoft Excel - squid

File Edit View Insert Format Tools Data S-PLUS Window Help

F1 = GSI

	A	B	C	D	E	F	G
1	Sample	YEAR	MONTH	Location	Sex	GSI	
2	1	1	1	1	2	10.4432	
3	2	1	1	3	2	9.8331	
4	3	1	1	1	2	9.7356	
5	4	1	1	1	2	9.3107	
6	5	1	1	1	2	8.9926	
7	6	1	1	1	2	8.7707	
8	7	1	1	1	2	8.2576	
9	8	1	1	3	2	7.4045	
10	9	1	1	3	2	7.2156	
11	10	1	2	1	2	6.8372	
12	11	1	1	1	2	6.3882	
13	12	1	6	1	2	6.3672	
14	13	1	2	1	2	6.2998	
15	14	1	1	1	2	6.0726	
16	15	1	6	1	2	5.8395	
17	16	1	6	1	2	5.807	

UNIVARIATE

Ready

图 2.2 拟被载入 R 中的数据库在 Excel 中的前期准备。行代表个体(每一行表示一只乌贼),列表示不同的变量。第一行和第一列是标签,均不含有空格,并且没有空的数据

2.2.1.2 数据提取到制表符分隔的 ascii 文件

在 Excel 中,依次进入**文件->另存为->保存类型(File -> Save As -> Save as Type)**,选择**文本文件(制表符分隔)**,将图 2.2 中关于乌贼的数据提取到一个制表符分隔的 ascii 文件中,存储目录为 C:\RBook,命名为 squid.txt。Excel 文件和 ascii 文件都可以从本书的主页中下载,如果你把它们下载到不同的目录下,则需要调整相应的路径。

在这一步中建议关闭 Excel 以方便其它程序访问新生成的文本文件。

警告:在某些情况下,如果你在电子数据表里输入了注释,Excel 有在 ascii 文件中加入额外一些全是 NA 列的趋势。在 R 中,这些列将会以 NA 出现,为了避免这种情况的发生,在提取数据前先删除这样的列。

2.2.1.3 read.table 函数的使用

P.48 当制表符分隔的 ascii 文件中没有空内容或者没有包含空格的名称时,我们就可以开始将数据载入 R 中。使用 read.table 函数,其基本的用法如下:

```
> Squid <- read.table(file = "C:\\RBook\\squid.txt",
                      header = TRUE)
```

这个命令实现了把数据从 *squid.txt* 文件中读取出来，以数据框的形式存储到 Squid 中。我们强烈建议使用简单、明确的变量名称。例如，我们不建议使用譬如 SquidNorthSeaMalesFemales 这样的名称，因为你很容易将它写错，导致 R 无法正确执行。函数 read.table 中的 header = TRUE 选项表示第一行包括了标签，如果你的文件中没有标签，可以将它改为 header = FALSE。还有另外一种识别这种文本文件地址的方法，它是：

```
> Squid <- read.table(file = "C:/RBook/squid.txt",
                      header = TRUE)
```

这两种命令的区别在于斜线的不同。如果在这一步出现了错误信息， P.49
可以首先检查文件名和目录路径是否正确。我们强烈建议使用简单的目录名，因为我们曾看到过很多人为了找出藏于 150～200 个字符长度的目录名中的一个错误而耽误半个多小时使 read.table 函数不能运行。在我们的示例中，目录名的结构是非常简单的，C:/RBook。而大多数情况下，目录的路径都要长一些，如果单纯依靠记忆来写这些路径是会经常出错的，这时，你可以右击文件 squid.txt(在 Windows 操作系统下)，选择属性(图 2.3)。在这里，可以直接把整个目录路径(包括文件名)复制粘贴到 R 的文本编辑器中。但是，不要忘记了多加一条斜线\。

如果你使用的名称含有“My Files”，一定要注意这里面的空格和大写 P.50
字母。另一个经常出现的错误就是小数点字符的使用，默认情况下，R 认为 ascii 文本文件中的数据使用点作为小数点，事实上，函数 read.table 的使用是这样的：

```
> Squid <- read.table(file = "C:/RBook/squid.txt",
                      header = TRUE, dec = ".")
```

如果你使用逗号作为小数点，应该将最后一个选项改为 dec = ",",并重新执行命令。

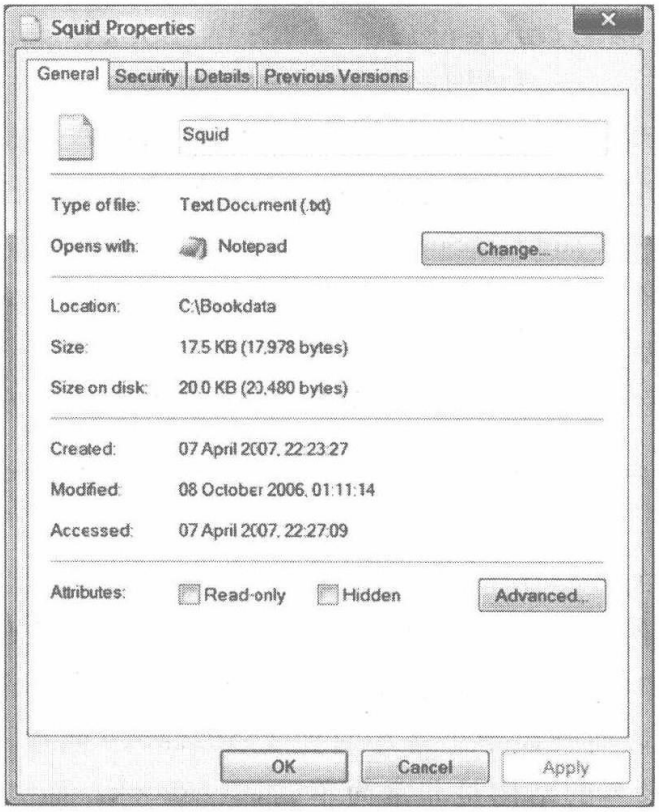


图 2.3 文件 *squid. txt* 的属性。文件名是 squid. txt，所在位置是 C:\Bookdata，你可以选择这个地址，并将其复制粘贴到 R 编辑器的 read. table 函数中，在 Windows 操作系统里需多加一条斜线\

先验证载入数据的格式再进行操作了。

如果变量名中含有空格，并且你按照上述的方式使用了 read. table 函数，你将会得到如下的结果（我们临时在 Excel 里将变量名 GSI 改为 G S I，以得到此错误结果）：

```
Error in scan(file,what,nmax,sep,dec,quote,skip,nlines,
na.strings,: line 1 did not have 8 elements
```

R 此时会对每一行元素的数量提出质疑，对此，一个很简单的处理办法是移除 Excel 中名称或数据字段中的空格，按上述步骤再执行一遍就可

```
read.table(file, header = FALSE, sep = "",
        quote = "\"' ", dec = ".", row.names, col.names,
        as.is = !stringsAsFactors,
        na.strings = "NA", colClasses = NA, nrows=-1,
        skip = 0, check.names = TRUE,
        fill = !blank.lines.skip,
        strip.white = FALSE, blank.lines.skip = TRUE,
        comment.char = "#", allowEscapes = FALSE,
        flush = FALSE,
        stringsAsFactors = default.stringsAsFactors())
```

P.51

这是一个有很多选项的函数。例如，如果在数据字段中含有空格，可以使用 strip.white = TRUE 选项，其它选项的解释可以在帮助文件语法部分找到。帮助文件还给出了以 csv 格式读取数据的信息。read.table 函数还包含了一个互联网上文本文件的 URL 连接。

如果你还需要从同一目录下读取更多的文件，利用 setwd 函数设置一下工作目录将会是更有效的方法。此时，你就可以省略掉 read.table 函数中的目录路径了，如下所示：

```
> setwd("C:\\RBook\\")
> Squid <- read.table(file = "squid.txt",
                      header = TRUE)
```

在这本书中，我们都是通过先使用 setwd 函数设置工作目录，再使用 read.table 函数来载入数据库的。这样做主要是因为并非这本书的所有读者都把数据文件存放在 C 盘中(一些电脑甚至没有 C 盘)。因此，这部分人需要做的仅仅是改一下 setwd 函数中的目录名就可以了。

这段不用看

除了 read.table 函数之外，你还可以通过 scan 函数来载入数据。它们的不同点是 read.table 函数把数据存储在数据框中，而 scan 函数把数据存储在矩阵中。在数据都是数值的情况下，scan 函数的运行速度更快(对于大数据库而言，一般指数百万个数据)。对于小数据库而言，就没有讨论运行速度的必要性了。关于 scan 函数更细节的内容，可以通过?scan 来参考它的帮助文件。

练习使用 read.table 函数和 scan 函数完成 2.4 节习题 6、习题 7，这些习题使用的是流行病学数据和深海研究数据。

2.2.2 从其它统计程序包中访问数据**

除了从 ascii 文件中访问数据之外，R 还可以从其它统计程序包中载入数据，例如，Minitab，S-PLUS，SAS，SPSS，Stata，Systat 等等。但是，最好

序包可以实现接口连接。

2.3 我们学习了哪些 R 函数?

表 2.2 列出了本章所介绍的 R 函数。

表 2.2 本章所介绍的 R 函数

函 数	功 能	示 例
sum	计算和	sum(x,na.rm = TRUE)
median	计算中位数	median(x,na.rm = TRUE)
max	计算最大值	max(x,na.rm = TRUE)
min	计算最小值	min(x,na.rm = TRUE)
c()	连接数据	c(1,2,3)
cbind	以列结合变量	cbind(x,y,z)
rbind	以行结合变量	rbind(x,y,z)
vector	以向量形式结合数据	vector(length = 10)
matrix	以矩阵形式结合数据	matrix(nrow = 5,ncol = 10)
data.frame	以数据框形式结合数据	data.frame(x = x,y = y,z = z)
list	以列表形式结合数据	list(x = x,y = y,z = z)
rep	循环数值或变量	rep(c(1,2,3),each = 10)
seq	生成一个有序的序列	seq(1,10)
dim	矩阵或者 cbind 输出的维数	dim(MyData)
colnames	矩阵或者 cbind 输出的列命名	colnames(MyData)
rownames	矩阵或者 cbind 输出的行命名	rownames(MyData)
setwd	设置工作目录	setwd("C:/Rbook/")
read.table	从 ascii 文件中读取数据	read.table(file = "test.txt", header = TRUE)
scan	从 ascii 文件中读取数据	scan(file = "test.txt")

2.4 习题

习题 1. c 和sum 函数的使用。

此题使用的是一个流行病学数据。Vicente 等(2006)通过观察生长在

西班牙一些地方的野猪和马鹿得到这些数据，数据库包含了两种生物的肺结核(tuberculosis，Tb)信息，寄生虫 *Elaphostrongylus cervi* 的信息，这种寄生虫只会感染马鹿。

在 Zuur 等人(2009)的著作中，Tb 被当作是一个连续变量的函数，动 P.55
物的长度由 LengthCT(CT 是 *cabeza-tronco* 的缩写，它是西班牙语，表示头体)表示。Tb 和 *E. cervi* 由 0 或者 1 的向量表示，分别代表未发现或发现了 Tb 和 *E. cervi* 的幼虫。下表中的前 7 行给出了鹿的数据。

农场	月份	年份	性别	LengthClass	LengthCT	Ecervi	Tb
MO	11	00	1	1	75	0	0
MO	07	00	2	1	85	0	0
MO	07	01	2	1	91.6	0	1
MO	NA	NA	2	1	95	NA	NA
LN	09	03	1	1	NA	0	0
SE	09	03	2	1	105.5	0	0
QM	11	02	2	1	106	0	0

使用 c 函数生成一个包含了 7 只动物长度值的一个变量，再生成一个包含 Tb 值的变量，包含 NA。并求 7 只动物的平均长度。

习题 2. 使用流行病学数据练习cbind 函数的应用。

继续习题 1 中关于鹿的问题。首先生成一个包含了农场和月份信息的变量，注意农场是字符串。然后使用 cbind 命令结合月份、长度和 Tb 值，并且将结果存储在变量 Boar 中，同时确保可以提取 Boar 中的行、列和每个元素。使用 dim，nrow，ncol 函数确定 Boar 中动物的数量和变量的数量。

习题 3. 使用流行病数据练习vector 函数的应用。

继续习题 1 中关于鹿的问题。类似于习题 2，使用 vector 函数结合 Tb 数据，使用不一样的变量名，例如 Tb2。

习题 4. 对矩阵的操作。

在 R 中生成下面的矩阵，并确定它的转置矩阵，逆矩阵，同时计算 D 和它的逆矩阵的乘积(结果将是单位矩阵)。

$$D=\begin{pmatrix}1 & 2 & 3\\ 4 & 2 & 1\\ 2 & 3 & 0\end{pmatrix}$$

习题 5. 使用流行病学数据练习data.frame 函数和list 函数的应用。

继续习题 1 至 3 中的问题。生成一个包含习题 1 表中所有数据的数据
P.56 框，并将长度数据值的均方根加到这个数据框中，再用 list 函数完成同样的工作，比较一下它们的不同点。

习题 6. 使用深海研究数据练习read.table 函数和scan 函数的应用。

文件 ISIT.xls 包含了深海生物发光的数据，图 1.6 就是由这些数据完成的，此图上面的段落是这些数据的描述。准备一个电子数据表(大概有4～5个问题需要解决)，并且把数据提取到 ascii 文件中，依次使用 read.table 函数和 scan 函数将这些数据载入到 R 中，使用两个不同的变量名存储数据，比较它们的不同点，使用 is.matrix 函数和 is.data.frame 函数回答这个问题。

习题 7. 使用流行病学数据练习read.table 函数或scan 函数的应用。

文件 *Deer.xls* 包含了习题 1 讨论的鹿的数据，但是也包含了其它动物的数据，把需要的数据从 Excel 提取到 ascii 文件中，并且将它载入 R。

第3章

访问变量和处理数据子集

上一章我们示范了从电子数据表或数据库将数据导入 R 中。我们也展示了如何输入小型数据集并把它们存储在一个数据框中。现在我们讨论访问数据子集。 P.57

3.1 访问数据框变量

假设前面章节中载入鱿鱼数据时没有出现错误，现在我们继续处理数据。

进行统计分析时，删除部分数据，选取若干子集，或加以分类都是很重要的。这些操作大部分可以在导入到 R 之前，使用 Excel 或者其它电子数据表(或数据库)程序完成，但是，由于种种原因，最好不要这样做。最终您可能需要每次选定部分数据重新载入。也可能一些数据文件太大以至于无法从电子数据表中载入。因此，一定程度上掌握在 R 里处理数据文件的知识是有益的。然而，对读者而言，这可能是 R 最难的一方面，但是一旦掌握了它是很有益的，因为这意味着所有的 Excel(或任何其它的电子数据表)中繁琐的数据处理都可以在 R 中完成。

我们利用上一章中载入的鱿鱼数据。如果你还没有这样做，使用下面的命令载入数据，并存储在数据框 squid 中。

```
> setwd("C:/RBook/")
> Squid <- read.table(file = "squid.txt",
                      header = TRUE)
```

read.table 函数生成了一个数据框，并且，因为 R 中大部分函数用数据框工作，我们更倾向于它，而不是 scan 函数。我们建议 read.table 命令后，立即使用 names 命令以查看我们正在处理的变量：

P.58

```
> names(Squid)
[1] "Sample" "Year" "Month" "Location" "Sex" "GSI"
```

我们经常注意到我们的课程参与者直接将代码输入到 R 控制台。正如第 1 章提到的，我们强烈建议将命令输入一个好的文本编辑器，比如基于 Windows 操作系统的 Tinn-R。（见第 1 章使用非 Windows 操作系统的编辑器资源。）为强调这一点，图 3.1 显示了目前为止我们的 R 代码快照。请注意我们复制并粘帖 names 命令的结果到 Tinn-R 文件。这使得我们能够迅速查看我们正在处理哪些变量并减少输入错误的机会。

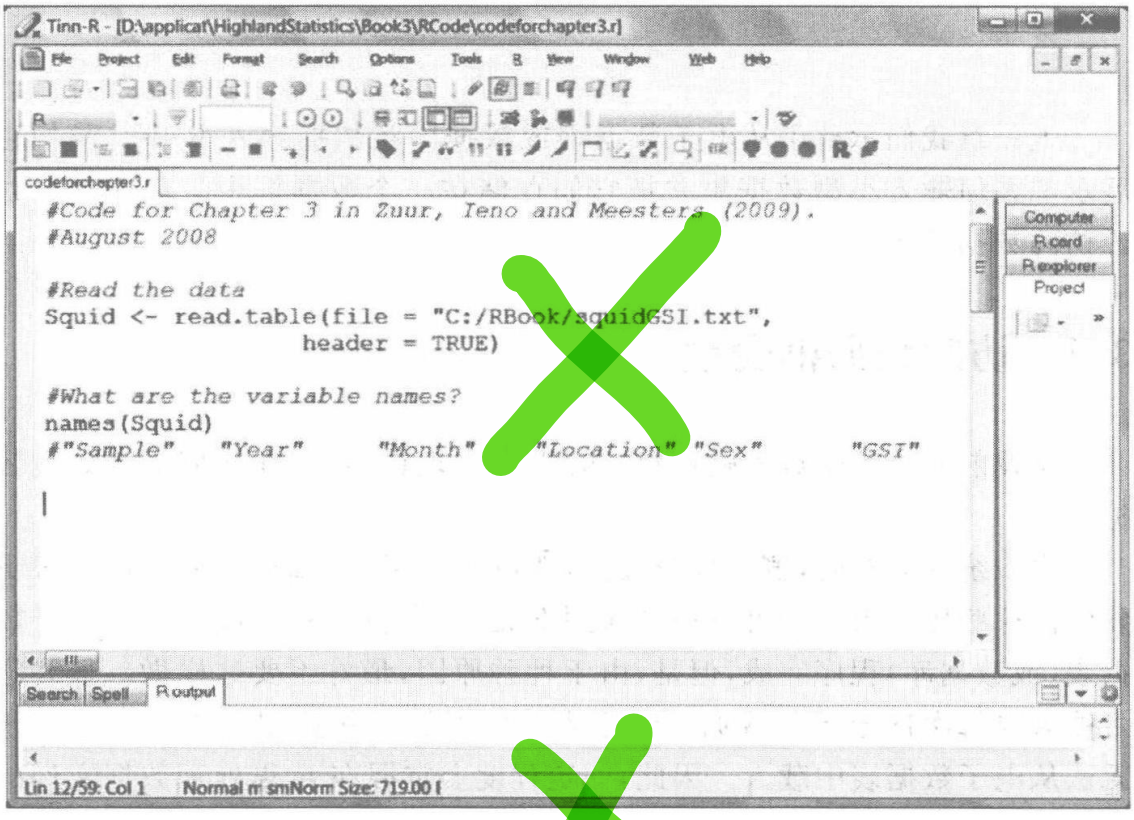


图 3.1　我们的 Tinn-R 文件快照。请注意“#”符号放在注释之前，代码是有很好的记录的，包括代码的编写日期。把所有变量名称复制并粘帖到文本文件可以让我们快速检查变量名称的拼写。你的文件结构尽可能透明，并且添加注释是很重要的。也要确保你有该文件和数据文件的备份

3.1.1　str 函数

P.59

str(结构)命令告诉我们数据框中每个变量的属性：

```
> str(Squid)
'data.frame': 2644 obs. of 6 variables:
 $ Sample    : int 1 2 3 4 5 6 7 8 9 10 ...
 $ Year      : int 1 1 1 1 1 1 1 1 1 1 ...
 $ Month     : int 1 1 1 1 1 1 1 1 1 2 ...
 $ Location  : int 1 3 1 1 1 1 1 3 3 1 ...
 $ Sex       : int 2 2 2 2 2 2 2 2 2 2 ...
 $ GSI       : num 10.44 9.83 9.74 9.31 8.99 ...
```

这组神秘的输出告诉我们变量样本、年份、月份、位置和性别是整数型，GSI 是数值型。假如你使用了错误的分隔符点：

3.1.2　函数中的数据参数

访问数据框中变量最有效的方法如下。确定R中的一个函数，例如，线性回归函数lm；根据变量GSI，Month，Year和Location指定模型；并告诉函数lm在数据框Squid中可以找到数据。尽管我们这本书中不进一步讨论线性回归，但给出代码如下。

```
> M1 <- lm(GSI ~ factor(Location) + factor(Year),
          data = Squid)
```

我们忽略了第一部分，它指定了实际的线性回归模型。上述语句的最后一部分(data =)告诉R，变量在数据框Squid中。这是一种简洁的方法，因为没有必要在数据框外定义变量；所有变量都很好地存储在数据框Squid中。这种方法的主要问题是，并不是所有函数都支持data选项。例如，

```
> mean(GSI, data = Squid)
```

会给出错误信息：

```
Error in mean (GSI, data = Squid) : object "GSI" not
found
```

因为函数mean不含data参数。有时帮助文件告诉你有data参数，在一些情况下可能有效，但是另外一些情形下可能无效。例如，下面的代码给出一个盒形图(这里没有显示)。

```
> boxplot(GSI ~ factor(Location), data = Squid)
```

但是这条命令给出错误信息：

```
> boxplot(GSI, data = Squid)
Error in boxplot(GSI, data = Squid) : object "GSI" not
found
```

P.61　总之，如果一个函数有data参数，就使用它；这是最简洁的编程方法。

3.1.3　$符号

那么，如果一个函数没有data参数，你可以做些什么呢？这里有两种方法可以访问变量。第一个方法是$符号：

```
> Squid$GSI
 [1] 10.4432  9.8331  9.7356  9.3107  8.9926
 [6]  8.7707  8.2576  7.4045  7.2156  6.8372
[11]  6.3882  6.3672  6.2998  6.0726  5.8395
```

<为了节省空间截至此处>

我们只复制和粘帖了包含 2644 个观察值的数据集的前几行作为输出。其它变量能够以相同的方式访问。输入数据框的名称，紧接着是 $ 符号以及变量名。原则上，你可以在 $ 符号和变量名之间加入空格：

```
> Squid$GSI
 [1] 10.4432 9.8331 9.7356 9.3107 8.9926
 [6] 8.7707 8.2576 7.4045 7.2156 6.8372
[11] 6.3882 6.3672 6.2998 6.0726 5.8395
```

<为了节省空间截至此处>

我们并不推荐这种方法(它看起来很奇怪)。

第二种方法是如果你要访问 GSI 数据，选择第 6 列：

```
> Squid[, 6]
 [1] 10.4432 9.8331 9.7356 9.3107 8.9926
 [6] 8.7707 8.2576 7.4045 7.2156 6.8372
[11] 6.3882 6.3672 6.2998 6.0726 5.8395
```

<为了节省空间截至此处>

它给出了完全相同的结果。无论是使用 Squid $ GSI 还是 Squid[,6]，现在你都可以计算均值：

```
> mean(Squid$GSI)
[1] 2.187034
```

我们倾向于使用 $ GSI 代码，在你输入 Squid[,6]一周后，可能会忘记 GSI 数据在第 6 列，符号 $ GSI 更清晰。 P.62

你也可以使用 Squid[,"GSI"]，它会增加混淆。请注意对于有些函数，

完成 3.7 节的习题 1。这是一个利用 read.table 函数并使用流行病学数据集访问变量的习题。

3.2 访问数据子集

本节，我们讨论如何访问并提取数据框 Squid 的成分。该方法可以应用到你自己输入数据创建的一个数据框，如第 2 章所示。

可能会出现这种情况，你只想处理，例如，雌性数据，某个位置的数据，或者某个位置的雌性数据。为了提取数据子集，我们需要知道性别是如何编码的。我们可以键入

```
> Squid$Sex
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[23] 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[45] 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 1 1 1 1 2 1 1
[67] 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1
```

<为了节省空间截至此处>

但是这显示了变量 Sex 的所有值。一个好的选择是使用 unique 命令显示这个变量里有多少个唯一值：

```
> unique(Squid$Sex)
[1] 2 1
```

这里 1 表示雄性，2 表示雌性。为了访问所有的雄性数据，使用 P.64

```
> Sel <- Squid$Sex == 1
> SquidM <- Squid[Sel, ]
> SquidM
    Sample Year Month Location Sex    GSI
24      24    1     5        1   1 5.2970
48      48    1     5        3   1 4.2968
58      58    1     6        1   1 3.5008
60      60    1     6        1   1 3.2487
61      61    1     6        1   1 3.2304
```

<为了节省空间截至此处>

第一行生成一个向量 Sel 与变量 Sex 具有相同的长度，如果 Sex 值为 1 则该变量的值是 TRUE，反之为 FALSE。这样的一个向量也称为布尔向量，可以用来选择行，因此我们命名为 Sel。在下一行，我们选择 Squid 中

Sel 等于 TRUE 的行，并把我们选择的数据存储在 SquidM 里。因为我们选择Squid 的行，我们需要使用方括号[ ]，并且，因为我们想要行，具有布尔值的向量 Sel 必须在逗号之前。也可以在一个命令里完成两行：

```
> SquidM <- Squid[Squid$Sex == 1, ]
> SquidM

    Sample Year Month Location Sex    GSI
24      24    1     5        1   1 5.2970
48      48    1     5        3   1 4.2968
58      58    1     6        1   1 3.5008
60      60    1     6        1   1 3.2487
61      61    1     6        1   1 3.2304
```

<为了节省空间截至此处>

雌性数据可以通过如下方式获得

```
> SquidF <- Squid[Squid$Sex == 2, ]
> SquidF
  Sample Year Month Location Sex     GSI
1      1    1     1        1   2 10.4432
2      2    1     1        3   2  9.8331
3      3    1     1        1   2  9.7356
4      4    1     1        1   2  9.3107
5      5    1     1        1   2  8.9926
```

P.65　　<为了节省空间截至此处>

基于第二个变量的值的条件下选择变量数据（或数据框）的过程称为条件选择。unique 命令应用到 Squid $ Location 上显示有 4 个位置编码为 1,2,3 和 4。为了提取位置 1,2 或 3 的数据，我们可以使用下面的语句它们都给出相同的结果（符号|表示布尔运算“或”，！ =表示“不等于”）。

```
> Squid123 <- Squid[Squid$Location == 1 |
                    Squid$Location == 2 | Squid$Location == 3, ]
> Squid123 <- Squid[Squid$Location != 4, ]
> Squid123 <- Squid[Squid$Location < 4, ]
> Squid123 <- Squid[Squid$Location <= 3, ]
> Squid123 <- Squid[Squid$Location >= 1 &
                    Squid$Location <= 3, ]
```

你可以选择它们中的任何一个。下面我们使用“&”，它是布尔“和”运算符。假定我们想从位置 1 提取雄性数据。这意味着数据既来自于雄性鱿鱼也来自于位置 1。下列代码提取满足这些条件的数据。

```
> SquidM.1 <- Squid[Squid$Sex == 1 &
                    Squid$Location == 1,]
     Sample Year Month Location Sex    GSI
24       24    1     5        1   1 5.2970
58       58    1     6        1   1 3.5008
60       60    1     6        1   1 3.2487
61       61    1     6        1   1 3.2304
63       63    1     6        1   1 3.1848
```

<为了节省空间截至此处>

来自位置 1 或 2 的雄性数据由下面给出

```
> SquidM.12 <- Squid[Squid$Sex == 1 &
       (Squid$Location == 1 | Squid$Location == 2), ]
```

不要使用下面的命令

```
> SquidM <- Squid[Squid$Sex == 1, ]
> SquidM1 <- SquidM[Squid$Location == 1, ]
> SquidM1
    Sample Year Month Location Sex    GSI
24      24    1     5        1   1 5.2970
58      58    1     6        1   1 3.5008

60      60    1     6        1   1 3.2487
61      61    1     6        1   1 3.2304
62      62    1     5        3   1 3.2263
...
NA.1113     NA   NA    NA        NA  NA   NA
NA.1114     NA   NA    NA        NA  NA   NA
NA.1115     NA   NA    NA        NA  NA   NA
NA.1116     NA   NA    NA        NA  NA   NA
```

P.66

第一行提取雄性数据并把它分配给 SquidM，因此它比 Squid(假定数据里有雌性鱿鱼)的维数小(较少的行)。下一行，布尔向量 Squid $ Location ==1 比 SquidM 的行的数量长，R 将对 SquidM 添加具有 NAs 值的多余的行。于是，我们得到一个数据框，SquidM1，它包含 NAs。问题是我们想要使用与 Squid 具有相同行数的布尔向量访问 SquidM 中的元素。

如果一个子集选择命令的输出显示了下面的信息，不要惊慌。

```
> Squid[Squid$Location == 1 & Squid$Year == 4 &
        Squid$Month == 1, ]
[1] Sample    Year      Month     Location  Sex
       GSI       fSex      fLocation
<0 rows> (or 0-length row.names)
```

这只是意味着这4年里没有测量值来自1月份的位置1。

3.2.1　数据排序

除了提取数据子集，有时重新排列数据也是有用的。对于鱿鱼数据，你可能想根据变量“月份”由低到高的值排列GSI数据，即使只是为了快速浏览。可以使用下面的代码。

```
> Ord1 <- order(Squid$Month)
> Squid[Ord1, ]
  Sample Year Month Location Sex     GSI
1      1    1     1        1   2 10.4432
2      2    1     1        3   2  9.8331
3      3    1     1        1   2  9.7356
4      4    1     1        1   2  9.3107
5      5    1     1        1   2  8.9926
```

P.67　　<为了节省空间截至此处>

因为我们是处理Squid的行，我们需要把Ord1放在逗号前。我们也可以只用一个变量完成这个练习，例如GSI。这种情况下，使用

```
> Squid$GSI[Ord1]
  [1] 10.4432  9.8331  9.7356  9.3107  8.9926  8.7707
  [7]  8.2576  7.4045  7.2156  6.3882  6.0726  5.7757
 [13]  1.2610  1.1997  0.8373  0.6716  0.5758  0.5518
 [19]  0.4921  0.4808  0.3828  0.3289  0.2758  0.2506
```

<为了节省空间截至此处>

完成3.7节的习题2。这是一个利用read.table函数并使用深海研究数据集访问数据框子集的习题。

3.3　使用相同的标识符组合两个数据集

到目前为止，我们已看到所有的数据点存储在相同的文件中的例子。然而，事实并非总是如此。本书的作者曾参与很多项目，在这些项目里包含相同动物的不同类型的测量数据。例如，其中的一个项目是由不同的研究所给出的大约1000条鱼的测量值；一个研究所计算形态度量值，另一个测量化学变量，另外还有一个计算寄生虫的数目。每个研究所生成包含工作组特定变量的电子数据表。关键的一点是，每个研究所的研究人员测量

每条鱼，因此所有的电子数据表包含一列确定鱼的类别。有些鱼在处理过程中丢失了或者不适合某种处理过程。因此，最终的结果是一系列的Excel电子数据表，每个包含上千个5～20组特定变量的观测值，但是对于每条鱼(案例)具有一个共同的标识符。

对于一个简单的数据集，见图3.2的电子数据表。设想鱿鱼数据以这种方式组织，两个具有相同标识符的不同的文件或工作表。现在的任务是合并两个数据集使得第一个数据集里第j个样品放在第二个数据集里第j个样品的旁边。为了说明目的，我们删除第二个电子数据表的第四行；正如假定某人忘记输入第四个观察值的年份、月份、位置和性别。R有一个有用的工具能合并文件，即merge函数。它由下面的代码运行。前两行用来读取两个单独的鱿鱼文件：

P.68

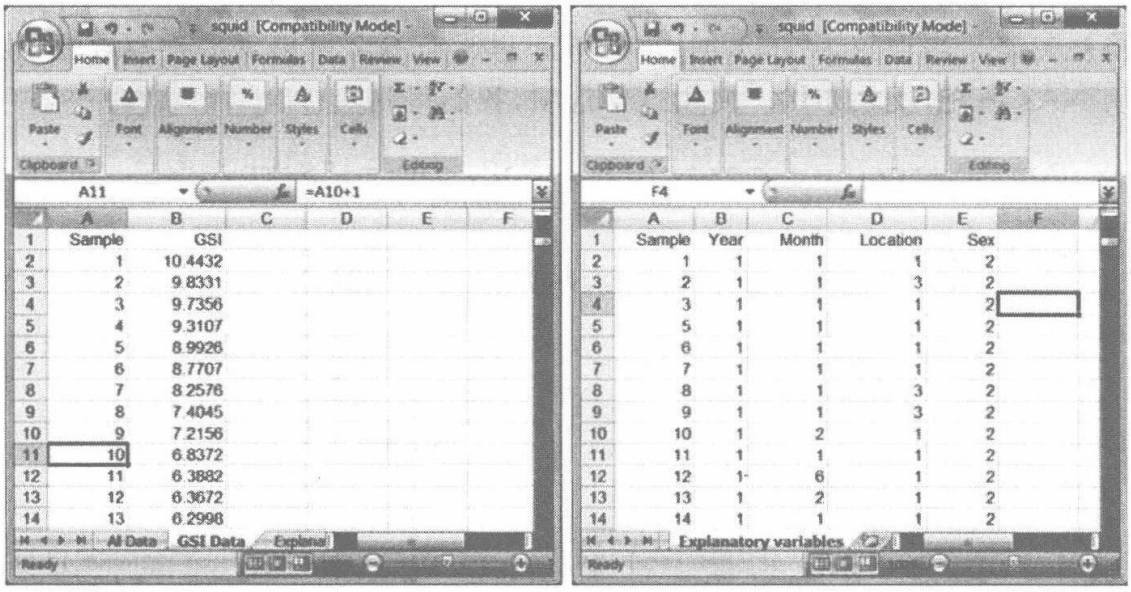


图3.2 带有样本数的GSI数据(左边)和带有样本数的其它变量(右边)。为了示范merge函数，我们删除了右边电子数据表的第四行

```
> setwd("C:/RBook/")
> Sq1 <- read.table(file = "squid1.txt",
                    header = TRUE)
> Sq2 <- read.table(file = "squid2.txt",
                    header = TRUE)
> SquidMerged <- merge(Sq1, Sq2, by = "Sample")
> SquidMerged
  Sample      GSI Year Month Location Sex
1      1 10.4432    1     1        1   2
2      2  9.8331    1     1        3   2
```

```
3      3   9.7356    1      1          1    2
4      5   8.9926    1      1          1    2
5      6   8.7707    1      1          1    2
6      7   8.2576    1      1          1    2
7      8   7.4045    1      1          3    2
8      9   7.2156    1      1          3    2
9     10   6.8372    1      2          1    2
10    11   6.3882    1      1          1    2
```

<为了节省空间截至此处>

merge命令采用两个数据框Sq1和Sq2作为参数并使用变量Sample作为相同的标识符合并两个数据集。merge函数的一个有用的选项是all。
P.69 缺省状态下它的设置是FALSE,它的含义是Sq1和Sq2的行如果有缺失值将被忽略。当设置为TRUE时,如果Sq1里没有Sq2里出现的样本数据,将用NAs填充,反之亦然。使用这个选项,我们得到

```
> SquidMerged <- merge(Sq1, Sq2, by = "Sample",
                       all = TRUE)
> SquidMerged
  Sample      GSI Year Month Location Sex
1      1 10.4432    1     1        1   2
2      2  9.8331    1     1        3   2
3      3  9.7356    1     1        1   2
4      4  9.3107   NA    NA       NA  NA
5      5  8.9926    1     1        1   2
6      6  8.7707    1     1        1   2
7      7  8.2576    1     1        1   2
8      8  7.4045    1     1        3   2
9      9  7.2156    1     1        3   2
10    10  6.8372    1     2        1   2
```

<为了节省空间截至此处>

请注意观察值(鱼/案例)4里的年份、月份、位置和性别是缺失值。为了避免混淆,回忆我们为了示范的目的只移除了第四行的观察值。更多的选项及例子在merge帮助文件里给出。

3.4 输出数据

除了read.table命令,R也有write.table命令。使用这个函数,你可以把数字信息输出到ascii文件。假设你提取了雄性鱿鱼数据,并且你想

把它输出到另外一个软件包，或把它传给一个同事。最简单的方法是输出雄性鱿鱼数据到 ascii 文件，然后把它输入到另外的软件包，或者通过电子邮件发送给你的同事。下列命令提取雄性数据（这种情况下你不必输入它），并把它输出到文件，MaleSquid.txt。

```
> SquidM <- Squid[Squid$Sex == 1, ]
> write.table(SquidM,
    file = "MaleSquid.txt",
    sep = " ", quote = FALSE, append = FALSE, na = "NA")
```

write.table 函数里第一个参数是你想要输出的变量，并且显然你也需要一个文件名。sep = " "保证数据用空格隔开，quote = FALSE 消除字符串（标题）的引号标志，na = "NA"允许你指定缺失值由什么来代替，append = FALSE 打开一个新的文件。如果你设置为 TRUE，它将把变量 SquidM 添加到一个已经存在的文件的尾部。 P.70

让我们说明一些这样的选项。当我们运行上面的代码，ascii 文件 MaleSquid.txt 的前六行如下。

```
Sample Year Month Location Sex GSI fLocation fSex
24 24 1 5 1 1 5.297  1 M
48 48 1 5 3 1 4.2968 3 M
58 58 1 6 1 1 3.5008 1 M
60 60 1 6 1 1 3.2487 1 M
61 61 1 6 1 1 3.2304 1 M
```

＜为了节省空间截至此处＞

3.5 重新编码分类变量

P.71 在 3.1 节，我们使用 str 函数给出了鱿鱼数据框的下列输出。

```
> str(Squid)
'data.frame': 2644 obs. of 6 variables:
 $ Sample   : int 1 2 3 4 5 6 7 8 9 10 ...
 $ Year     : int 1 1 1 1 1 1 1 1 1 1 ...
 $ Month    : int 1 1 1 1 1 1 1 1 1 2 ...
 $ Location: int 1 3 1 1 1 1 1 3 3 1 ...
 $ Sex      : int 2 2 2 2 2 2 2 2 2 2 ...
 $ GSI      : num 10.44 9.83 9.74 9.31 8.99 ...
```

变量 Location 编码为 1,2,3 或 4，Sex 为 1 或 2。这样的变量是分类或名义变量。在 Excel 里，我们可以把性别编码为雄性和雌性。把名义变量重新编码，在数据框里生成一个新的变量是很好的编程习惯。例如：

```
> Squid$fLocation <- factor(Squid$Location)
> Squid$fSex <- factor(Squid$Sex)
```

这两个命令生成数据框 Squid 里的两个新变量 fLocation 和 fSex。在变量名前使用 f 提醒我们它们是名义变量。在 R 里，我们也可以称它们为因子，因此用 f。输入

```
> Squid$fSex
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[18] 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2
[35] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2
 ...
[2602] 1 2 1 1 2 1 1 1 2 1 2 1 1 2 1 1 2
[2619] 1 2 2 1 1 1 1 1 1 1 1 1 2 1 1 1 2
[2636] 1 2 1 2 1 2 1 1 1
Levels: 1 2
```

请注意最后的额外一行。告诉我们 fSex 有两个水平，1 和 2。也可以重新标记这些水平为“雄性”和“雌性”，或者，可以更简洁地用 M 和 F：

```
> Squid$fSex <- factor(Squid$Sex, levels = c(1, 2),
  labels = c("M", "F"))
> Squid$fSex
 [1] F F F F F F F F F F F F F F F F
[18] F F F F F F M F F F F F F F F F
[35] F F F F F F F F F F F F F M F F F
 ...

[2602] M F M M F M M M F M F M M F M M F
[2619] M F F M M M M M M M M F M M M F
[2636] M F M F M F M M M
Levels: M F
```

P.72

每个 1 都被转换成一个“M”，每个 2 都被转换成一个“F”。现在你可以在函数比如 lm 或者 boxplot 里使用 fSex：

3.6 我们学习了哪些 R 函数?

表 3.1 列出了本章介绍的 R 函数。

表 3.1 本章介绍的 R 函数

函 数	功 能	示 例
write.table	把一个变量写入到 ascii 文件	write.table(Z,file = "test.txt")
order	确定数据的顺序	order(x)
merge	合并两个数据框	merge(x,y,by = "ID")
attach	使数据框里的变量可以利用	attach(MyData)
str	显示一个对象的内部结构	str(MyData)
factor	定义变量作为因子	factor(x)

3.7 习题

习题 1. 使用流行病学数据练习使用 read.table 函数并访问数据框里的变量。

文件 *BirdFlu.xls* 包含世界卫生组织(WHO)报告的一些国家已经证实的每年人类感染禽流感 A/(H5N1)的病例。数据来自于 WHO 网站(www.who.int/en/),复制是为了教育的目的。准备电子数据表并把这些数据载入到 R。如果你不是 Windows 用户,从文件 *BirdFlu.txt* 开始。请注意你需要调整列名称以及一些国家的名称。

在 R 里使用 names 和 str 命令观察这些数据。输出 2003 年禽流感病例数。2003 年和 2005 年禽流感病例总数是多少?哪个国家的病例最多?哪个国家禽流感死亡的人数最少?

使用第 2 章的方法,每个国家的禽流感病例总数是多少?每年的禽流感病例总数是多少?

习题 2. 使用深海研究数据练习使用 read.table 函数并访问数据框里的子集。

P.75 如果你还没有完成第 2 章的习题 6，做完它，并从 *ISIT.xls* 文件载入数据。

在 R 里，从站点 1 提取数据。这个站点有多少个观察值？站点 1 的样品深度的最小值、中位值、均值和最大值分别是多少？站点 2 的样品深度的最小值、中位值、均值和最大值分别是多少？站点 3 呢？

确定观察值相对较少的站点，生成一个忽略这些站点的新数据框。

提取来自 2002 年的数据。提取来自 4 月（所有年份）的数据。提取在深度超过 2000 米测量的数据（来自所有年份和月份）。根据深度值的增序显示这些数据。

显示在 4 月份并且深度超过 2000 米的测量数据。

习题 3. 使用深海研究数据练习使用 write.table 函数。

在上一个习题的最后一步，提取了在 4 月份并且深度超过 2000 米的测量数据。把这些数据输出到一个新的 ascii 文件。

习题 4. 使用深海研究数据练习使用 factor 函数并访问数据框里的子集。

站点 1 到 5 是 2001 年 4 月抽样，站点 6 到 11 是 2001 年 8 月抽样，站点 12 到 15 是 2002 年 3 月抽样，站点 16 到 19 是 2002 年 10 月抽样。在 R 里生成两个新的变量确定月份和年份。请注意这些是因子。把新的变量添加到数据框里。

参考文献

Barbraud C, Weimerskirch H (2006) Antarctic birds breed later in response to climate change. *Proceedings of the National Academy of Sciences of the USA* 103: 6048–6051.

Bivand RS, Pebesma EJ, Gómez-Rubio V (2008) *Applied Spatial Data Analysis with R*. Springer, New York.

Braun J, Murdoch DJ (2007) *A First Course in Statistical Programming with R*. Cambridge University Press, Cambridge.

Chambers JM, Hastie TJ (1992) *Statistical Models in S*. Wadsworth & Brooks/Cole Computer Science Series. Chapman and Hall, New York.

Claude J (2008) *Morphometrics with R*. Springer, New York.

Cleveland WS (1993) *Visualizing Data*, Hobart Press, Summit, NJ, 360 pp.

Crawley MJ (2002) *Statistical Computing. An Introduction to Data Analysis Using S-Plus*. Wiley, New York.

Crawley MJ (2005) *Statistics. An Introduction Using R*. Wiley, New York.

Crawley MJ (2007) *The R Book*. John Wiley & Sons, Ltd., Chichester.

Cruikshanks R, Laursiden R, Harrison A, Hartl MGH, Kelly-Quinn M, Giller PS, O'Halloran J (2006) *Evaluation of the use of the Sodium Dominance Index as a Potential Measure of Acid Sensitivity (2000-LS-3.2.1-M2) Synthesis Report*, Environmental Protection Agency, Dublin, 26 pp.

Dalgaard P (2002) *Introductory Statistics with R*. Springer, New York.

Everitt BS (2005) *An R and S-Plus Companion to Multivariate Analysis*. Springer, London.

Everitt B, Hothorn T (2006) *A Handbook of Statistical Analyses Using R*. Chapman & Hall/CRC, Boca Raton, FL.

Faraway JJ (2005) *Linear Models with R*. Chapman & Hall/CRC, FL, p 225.

Fox J (2002) *An R and S-Plus Companion to Applied Regression*. Sage Publications, Thousand Oaks, CA.

Gentleman R, Carey V, Huber W, Irizarry R, Dudoit S, editors (2005) *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Statistics for Biology and Health. Springer-Verlag, New York.

Gillibrand EJV, Bagley P, Jamieson A, Herring PJ, Partridge JC, Collins MA, Milne R, Priede IG (2006) Deep Sea Benthic Bioluminescence at Artificial Food falls, 1000 to 4800 m depth, in the Porcupine Seabight and Abyssal Plain, North East Atlantic Ocean. *Marine Biology* 149: doi: 10.1007/s00227-006-0407-0

Hastie T, Tibshirani R (1990) *Generalized Additive Models*. Chapman and Hall, London.

Hemmingsen W, Jansen PA, MacKenzie K (2005) Crabs, leeches and trypanosomes: An unholy trinity? Marine Pollution Bulletin 50(3): 336–339.

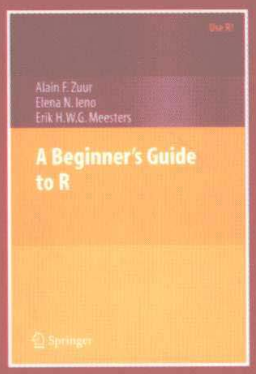
Hornik K (2008) The R FAQ, http://CRAN.R-project.org/doc/FAQ/

Jacoby WG (2006) The dot plot: A graphical display for labeled quantitative values. *The Political Methodologist* 14(1): 6–14.

Jolliffe IT (2002) *Principal Component Analysis*. Springer, New York.

Keele L (2008) *Semiparametric Regression for the Social Sciences*. Wiley, Chichester, UK.
Legendre P, Legendre L (1998) *Numerical Ecology* (2nd English edn). Elsevier, Amsterdam, The Netherlands, 853 pp.
Lemon J, Bolker B, Oom S, Klein E, Rowlingson B, Wickham H, Tyagi A, Eterradossi O, Grothendieck G, Toews M, Kane J, Cheetham M, Turner R, Witthoft C, Stander J, Petzoldt T (2008) Plotrix: Various plotting functions. R package version 2.5.
Loyn RH (1987) Effects of patch area and habitat on bird abundances, species numbers and tree health in fragmented Victorian forests. In: Saunders DA, Arnold GW, Burbidge AA, Hopkins AJM (eds) *Nature Conservation: The Role of Remnants of Native Vegetation*. Surrey Beatty & Sons, Chipping Norton, NSW, pp. 65–77.
Magurran, AE (2004) *Measuring Biological Diversity*. Blackwell Publishing, Oxford, UK.
Maindonald J, Braun J (2003) *Data Analysis and Graphics Using R* (2nd edn, 2007). Cambridge University Press, Cambridge.
Mendes S, Newton J, Reid R, Zuur A, Pierce G (2007) Teeth reveal sperm whale ontogenetic movements and trophic ecology through the profiling of stable isotopes of carbon and nitrogen. Oecologia 151: 605–615.
Murrell P (2006) *R Graphics*. Chapman & Hall/CRC, Boca Raton, FL.
Nason GP (2008) *Wavelet Methods in Statistics with R*. Springer, New York.
Oksanen J, Kindt R, Legendre P, O'Hara B, Simpson GL, Solymos P, Stevens MHH, Wagner H (2008) Vegan: Community Ecology Package. R package version 1.15-0. http://cran.r-project.org/, http://vegan.r-forge.r-project.org/
Originally Michael Lapsley and from Oct 2002, Ripley BD (2008) RODBC: ODBC Database Access. R package version 1.2-4.
Pinheiro J, Bates D, DebRoy S, Sarkar D and the R Core Team (2008) nlme: Linear and nonlinear mixed effects models. R package version 3.1-88.
R-Core Members, Saikat DebRoy, Roger Bivand and Others: See Copyrights File in the Sources (2008) Foreign: Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, dBase, ... R package version 0.8-25.
Lemon J, Bolker B, Oom S, Klein E, Rowlingson B, Wickham H, Tyagi A, Eterradossi O, Grothendieck G, Toews M, Kane J, Cheetham M, Turner R, Witthoft C, Stander J and Petzoldt T (2008). plotrix: Various plotting functions. R package version 2.5.
Pinheiro J, Bates D (2000) *Mixed Effects Models in S and S-Plus*. Springer-Verlag, New York, USA.
Quinn GP, Keough MJ (2002) *Experimental Design and Data Analysis for Biologists*. Cambridge University Press, Cambridge.
R Development Core Team (2008) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org
Reed JM, Elphick CS, Zuur AF, Ieno EN, Smith GM (2007) Time series analysis of Hawaiian waterbirds. In: Zuur AF, Ieno EN, Smith GM (eds) *Analysing Ecological Data GM*. Springer, New York.
Roulin A, Bersier LF (2007) Nestling barn owls beg more intensely in the presence of their mother than their father. Animal Behaviour 74: 1099–1106.
Sarkar D (2008) *Lattice: Lattice Graphics*. R package version 0.17-2
Shumway RH, Stoffer DS (2006) *Time Series Analysis and Its Applications with R Examples*. Springer, New York.
Sikkink PG, Zuur AF, Ieno EN, Smith GM (2007) Monitoring for change: Using generalised least squares, non-metric multidimensional scaling, and the Mantel test on western Montana grasslands. In: Zuur AF, Ieno EN, Smith GM (eds) *Analysing Ecological Data GM*. Springer, New York.
Spector P (2008) *Data Manipulation with R*. Springer, New York.
Venables WN, Ripley BD (2002) *Modern Applied Statistics with S* (4th edn). Springer, New York. ISBN 0-387-95457-0

Verzani J (2005) *Using R for Introductory Statistics*. CRC Press, Boca Raton.
Vicente J, Höfle U, Garrido JM, Fernández-de-Mera IG, Juste R, Barralb M, Gortazar C (2006) Wild boar and red deer display high prevalences of tuberculosis-like lesions in Spain. Veterinary Research 37: 107–119.
Wood SN (2006) *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC, NC.
Zar JH (1999) *Biostatistical Analysis* (4th edn). Prentice-Hall, Upper Saddle River, USA.
Zuur AF, Ieno EN, Smith GM (2007) *Analysing Ecological Data*. Springer, New York, 680p.
Zuur AF, Ieno EN, Walker NJ, Saveliev AA, Smith G (2009) *Mixed Effects Models and Extensions in Ecology with R*. Springer, New York.



最简单的入门书，不说统计只说R

作者基于他们对应用科学家讲授统计与R的丰富经验，为读者献上了《R语言初学者指南》这本书。为了避免同时讲授R与统计的困难，统计方法保持在最低限度。本书包括如何下载与安装R，载入和处理数据，基本绘图，函数简介，高级绘图以及初学者常见的错误。这本书包括了你开始学习R时想知道的所有内容。

“这本书的最大优点是它的目的只是讲解R……。它非常有效地组织R命令，包括许多教学指导。我会形容这本书是得心应手——它是那种你随时都想把它放在你的夹克口袋或者背包里的书，以备使用，像一把瑞士军刀。”

——Loveday Conquest，美国华盛顿大学

“虽然有几本书重点放在学习用R做统计……，本书作者主要关注学习R而几乎完全避免任何统计术语，从而填补了市场空白……。事实上作者有关于R的非常广泛的教学经验以使完全不懂R的初学者彻底理解它。”

——Mark Mainwaring，英国兰卡斯特大学

“正好是所需要的……。这是一个重大的、很好的工作。我喜欢生态学/生物学的例子；他们能提供很大帮助。”

——Andrew J. Tyne，美国内布拉斯加大学林肯分校

ISBN 978-7-5605-3942-3

9 787560 539423 >

定价：36.00元

策划编辑　赵丽平　李　颖
责任编辑　李　颖
封面设计　阎　亮

上架建议：统计软件/程序语言