R语言在生物医学领域的应用

崔　斌　黄金艳　主编

上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

前言 | *Preface*

这本书写给谁?

分子生物学发展到今天已有几十年的时间,从单个基因 real-time PCR 检测到几万个基因同时检测的芯片技术;从单个片段 sanger 法测序到百千万个 reads 的高通量测序技术,分子生物学的检测通量正在从单点分析过渡到大规模全面检测和分析。不仅仅在分子生物学领域,近十几年互联网和计算机技术的高速发展使得在生物医学日常生产和科研活动中产生的数据量也呈现极大增长。然而目前现状是具备专业知识的广大生物医学工作者虽然能够利用 Excel、SPSS 等软件进行一般的数据分析,但对于较大规模的数据和专业格式文件等往往束手无策。正是基于此,我们两位编者希望写一本书让非计算机专业的生物医学工作人员能够较快地学会信息分析和数据处理的方法与技巧,并可以自己动手完成相关数据工作。

为什么选择 R 语言?

这个问题对于专业从事计算机或者生物信息分析的人员来说可能就不成为问题,但是作为外行,如果之前对于计算机的主要功用就是上上网,使用 office 软件记录文字,绘制图表和制作幻灯片的话,那么摆在面前的第一个问题就是操作系统。Windows 操作系统是最普遍不过了,它最大的优点是可视化和简单的操作,但是更多以往的生物信息学软件基本上都是在 Linux 系统下运行,因此在开始学习生物信息分析之前,你可能首先需要学习 Linux 系统如何操作。讲到这里就可以开始说说 R 语言了,R 语言的第一个好处就是它本身可以安装在 Windows 操作系统、Lunix 系统以及 Mac OSX 系统上,这样就省去了学习 Linux 系统操作命令的诸多烦恼;第二点是它是一个免费的、开放源代码和有一定历史的语言。虽说免费的不一定就是好的,但是编者认为学术系统里面的免费在某种程度上决定了它的自由和公平;开放的源代码,这却是再好不过了,因为你可以轻松借鉴和学习他人的工作成果等一系列的便利;最后说说它的历史积淀,R 语言来源于 20 世纪 90 年代的一个基于 S 语言的 GNU 项目应用,最初的开发者是新西兰奥克兰大学

统计系的 Robert Gentleman 和 Ross Ihaka，这也是 R 语言命名的由来。目前 R 语言的维护和开发是由一个核心小组来完成，编者撰写本书时的版本是 3.1.1。最后编者想说的是一个基于统计和制图为目的开发的语言可以在这么长的发展中依然历久弥新，这本身就足以证明它的强大。

本书具体介绍了什么？

前面已经讲过编写此书目的是希望能够为非计算机专业的生物医学领域的工作者提供生物信息分析和数据处理的指导和经验性介绍，期望读者可以在较短的时间内了解 R 语言并使用 R 语言进行数据的分析工作。因此本书在章节的安排上从 R 语言软件的下载安装开始一步步带领读者了解和熟悉 R 语言软件的功能和如何使用，后面的具体应用章节是依据不同实验类型来加以介绍的，这样更有利于读者快速地找到所关心的兴趣点。在 R 语言使用方法介绍上，编者使用了类似实验中 protocol 方式分步讲解，同时配以图片来阐述说明如何导入数据，如何设置参数，如何选择命令等，其中结合了编者个人的实践经验，也穿插了一些方便的小软件和小技巧。但对于生物医学领域各个学科和专业的学术理论知识等，由于篇幅所限，本书并没有涉及，编者默认各位读者在阅读相关章节前已具备该领域的专业知识，本书所涉及的实验技术方法和原理以及经过 R 语言分析出来的结果如何解读等本书中并没有详细的说明和讲解。

最后，由于我们两位编者个人的视野和能力局限，本书中难免会有不足和疏漏之处，恳请读者给予批评和指正。

目录 | *Contents*

第 1 章　R 语言的安装和编辑器

本书所默认的读者群为对 R 语言在生物医学领域分析感兴趣的非计算机专业人士，因此本书所介绍的各种软件均运行在常用的 Windows 操作系统下。如果读者已下载并安装好 R 语言软件和编辑器可跳过本章，或直接阅读本章的 1.2 编辑器部分。

1.1　R 语言的下载和安装

首先介绍 R 语言的官方网站：http://www.r-project.org/（见图 1－1）。

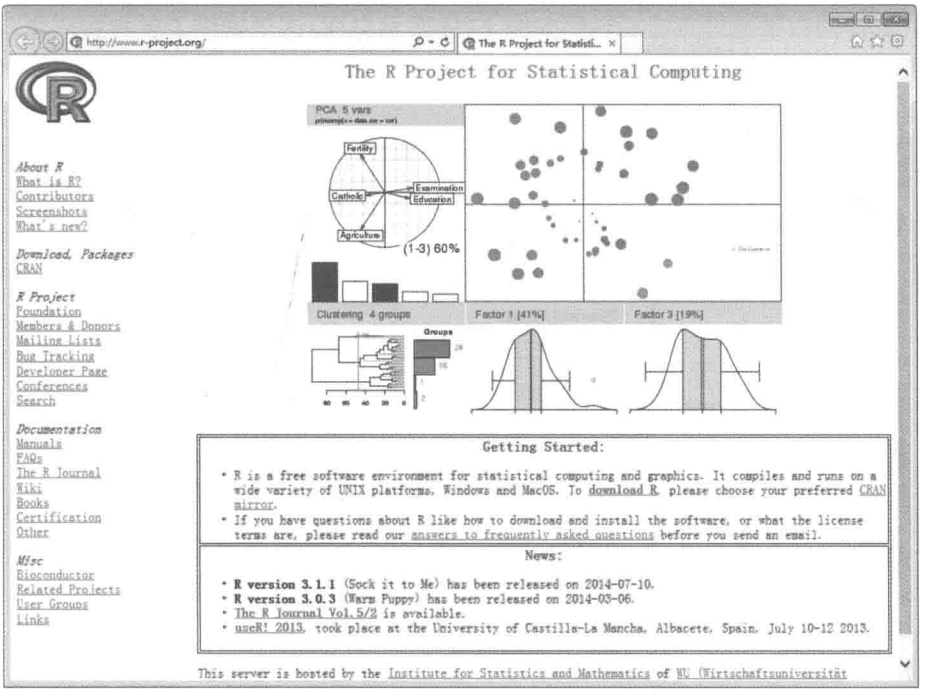

图 1－1　R 语言的官方网站首页

读者在登录 R 语言官方首页之后，鼠标点击下方蓝色的 download R 超链接，进入 CRAN(Comprehensive R Archive Network) Mirrors 页面(见图 1－2)，该

页面显示世界各地设立的R语言软件及其软件包的网络服务器地址列表，我们一般选择的国内某个镜像地址。

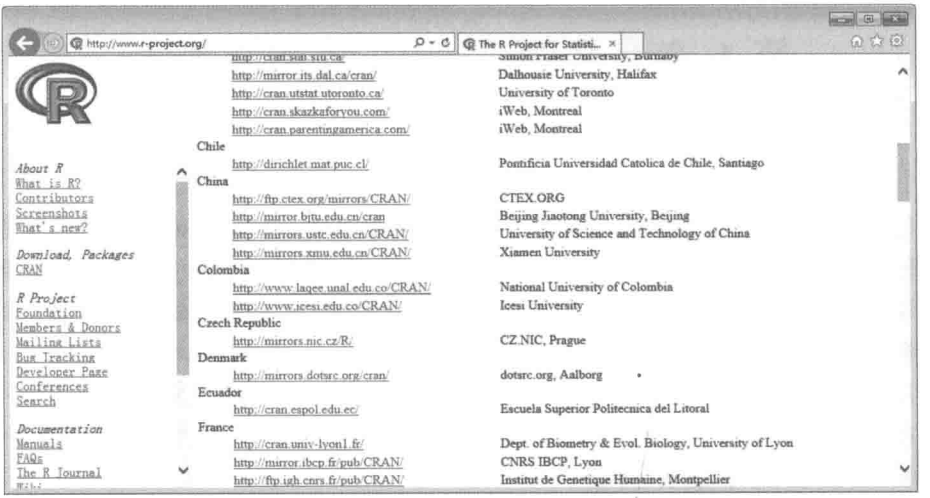

图1-2　R语言软件下载的各地镜像地址

选择国内某地址点击后转到如图1-3所示页面，这里提供了各种操作系统的安装软件包及版本更新信息，编者在开始撰写本书时R语言的版本为2014-07-10发布的R 3.1.1。

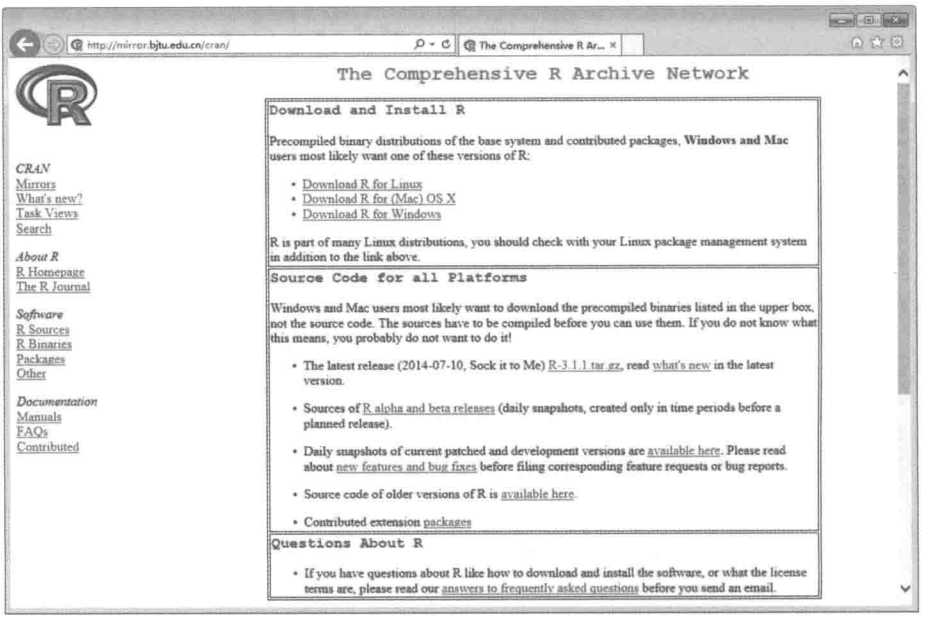

图1-3　R语言在不同操作系统的下载页面

鼠标单击 Download R for Windows(在 Windows 系统下的安装软件包),打开页面(见图 1-4),选择 base 基础安装软件包。

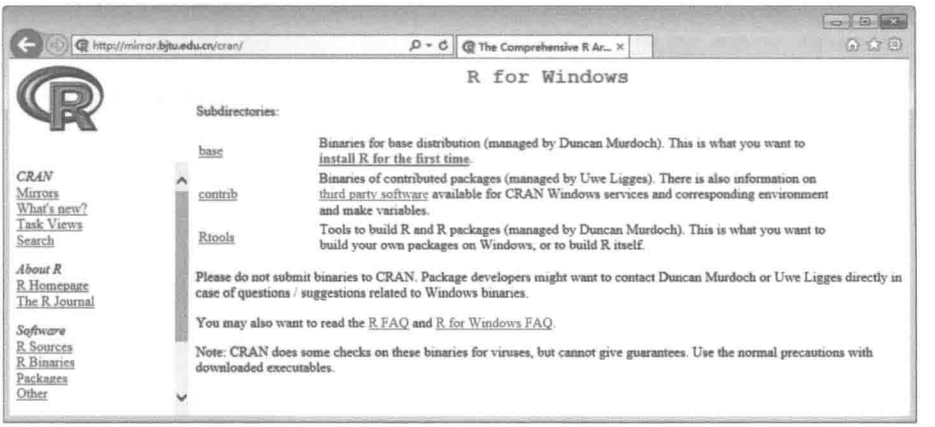

图 1-4　R 语言的 base、contrib 和 Rtool 软件包界面

如图 1-5 所示,下载 Download R 3.1.1 for Windows。

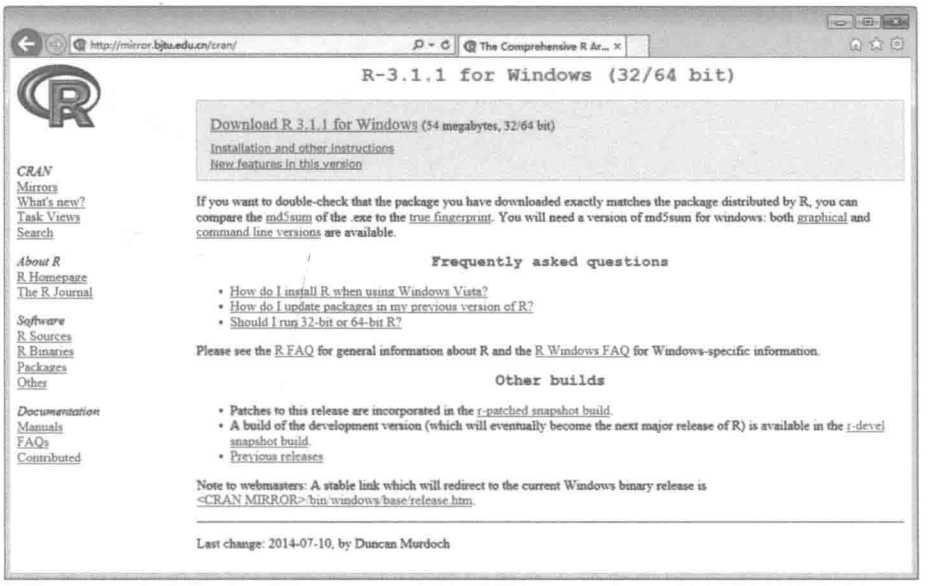

图 1-5　Windows 操作系统下版本 3.1.1 的 R 语言下载界面

双击下载的 R-3.1.1-win 安装程序进行 R 语言的安装,安装过程选择默认参数,连续点击“下一步”就完成了 R 语言的安装工作。此时,桌面上会出现 R 语言的快捷图标。若 64 位操作系统的计算机在默认参数安装的条件下,会生成两个 R 图标,一个对应 32 位 Windows 操作系统,快捷图标的名字为:R i386 3.1.1,另一个对应 64 位操作系统,快捷图标的名字为:R x64 3.1.1。选择桌面上的 R x64 3.1.1

图标双击，或者通过“开始 → 所有程序 → R → R x64 3.1.1”也可启动R语言程序界面（见图1-6）。

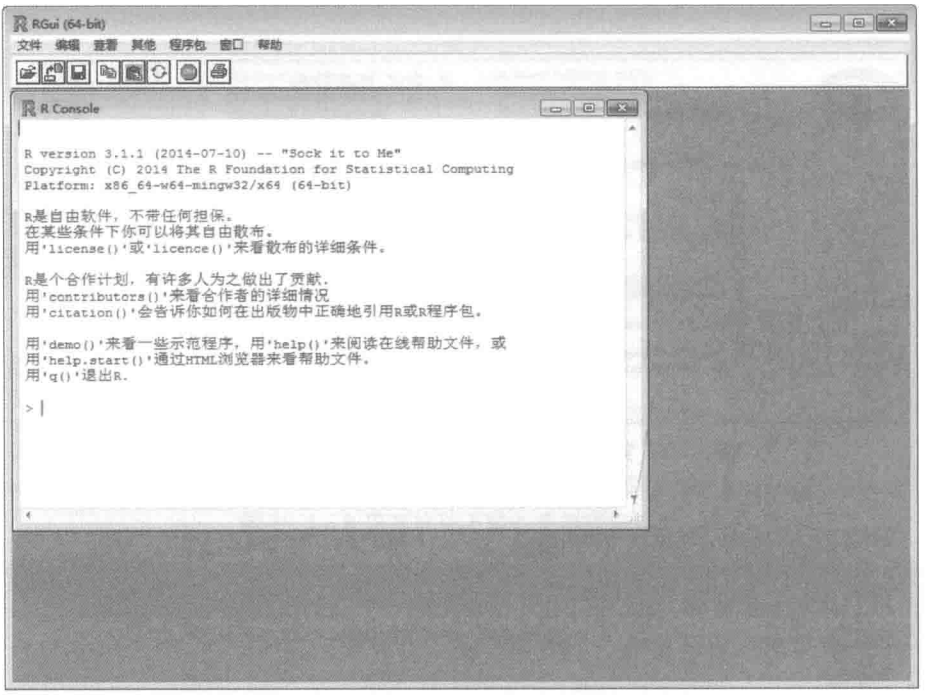

图1-6 R语言运行后的界面

到此为止就完成了R语言软件的下载和安装，如图1-6所示，从R官方网站上下载的R语言安装程序提供了一个基本的图形界面RGUI，在命令窗口中符号“>”之后的光标处键入R命令语句就可实现R的操作。

1.2 R语言的编辑器简介

上节中详细介绍了如何从官方网站下载和安装R语言，打开命令窗口后键入R命令语句，然后“回车”就可执行相应的命令。这样，对于简单的几行或者十几行的编程还可以方便实现，但是对于大量而复杂的脚本编程，特别是需要不断地修改调适的几十甚至几百行的R命令来说，这个基本的运行界面显然就有些不太方便。

基于以上原因和从更加方便的因素考虑，R语言诞生之后，一些能够集成编程、调试、运行和可视等功能在一起的集成开发环境（Integrated Development Environmen，IDE）编辑器就应运而生。目前，这些编辑器有：RStudio、Tinn-R、RWinEdt等，利用这些R语言编辑器可以更加方便地完成R语言的编程、保存、修改和运行等相关的工作。编者在本书中推荐使用的是RStudio编辑器，在下一

节里将详细介绍 RStudio 软件的安装等相关知识。

本书之后各章节中利用 R 语言运行的程序工作都是基于 Rstudio 编辑器完成的，对于其他的 R 语言编辑器，读者可根据个人喜好选择各自偏爱的 R 语言编辑器软件。但是无论选择哪一种 R 语言编辑器或者不用编辑器，对于同一个程序来说，R 语言运行的命令和结果基本都是一样的。

1.3　Rstudio 软件的安装

Rstudio 软件的官方网址为 http://www.rstudio.com/，登录后如图 1－7 所示。

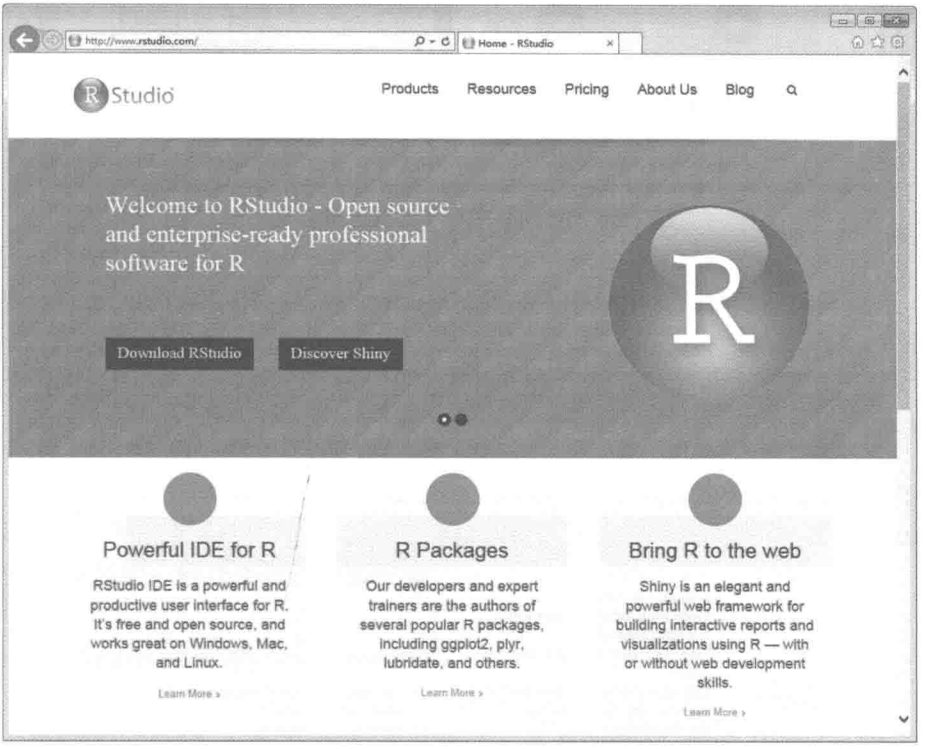

图 1－7　RStudio 软件的官方网站首页

鼠标点击“Powerful IDE for R”进入界面(见图 1－8)。

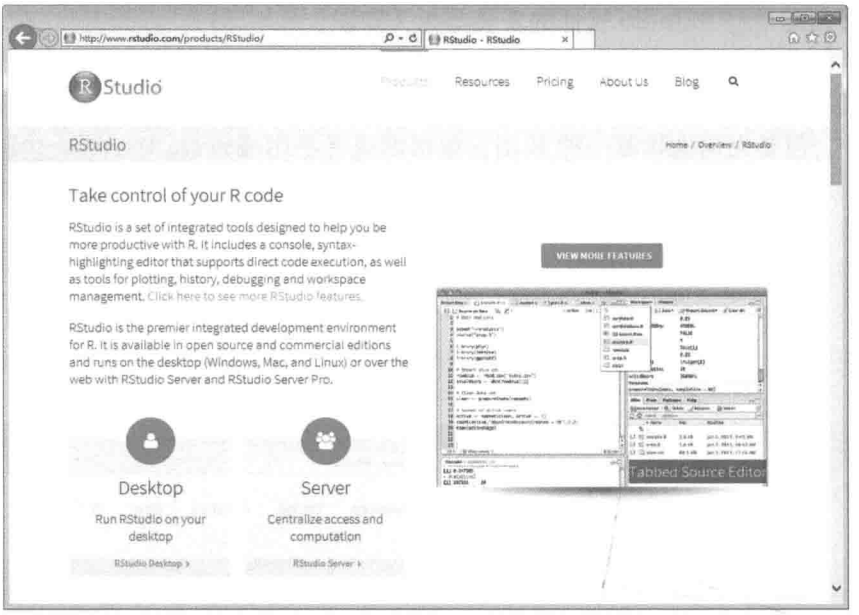

图 1-8　RStudio 软件下载流程界面之一

这里我们选择在个人电脑中安装的“Desktop”版本，向下移动本网页将会见到如图 1-9 所示的内容。

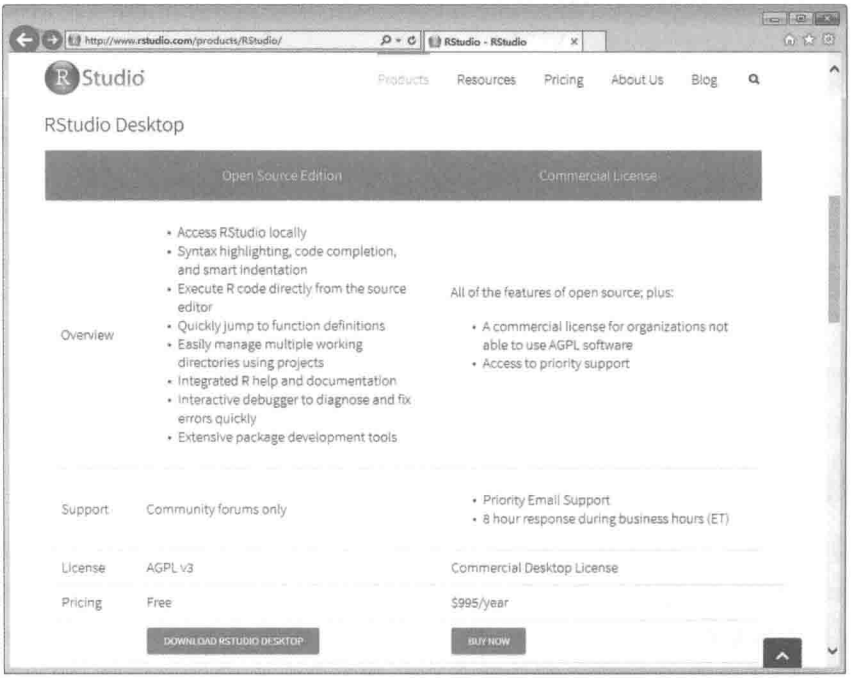

图 1-9　RStudio 软件下载流程界面之二

在如图 1 - 9 所示的界面上点击“Download RStudio Desktop”，进入 RStudio 软件的下载界面(见图 1 - 10)。

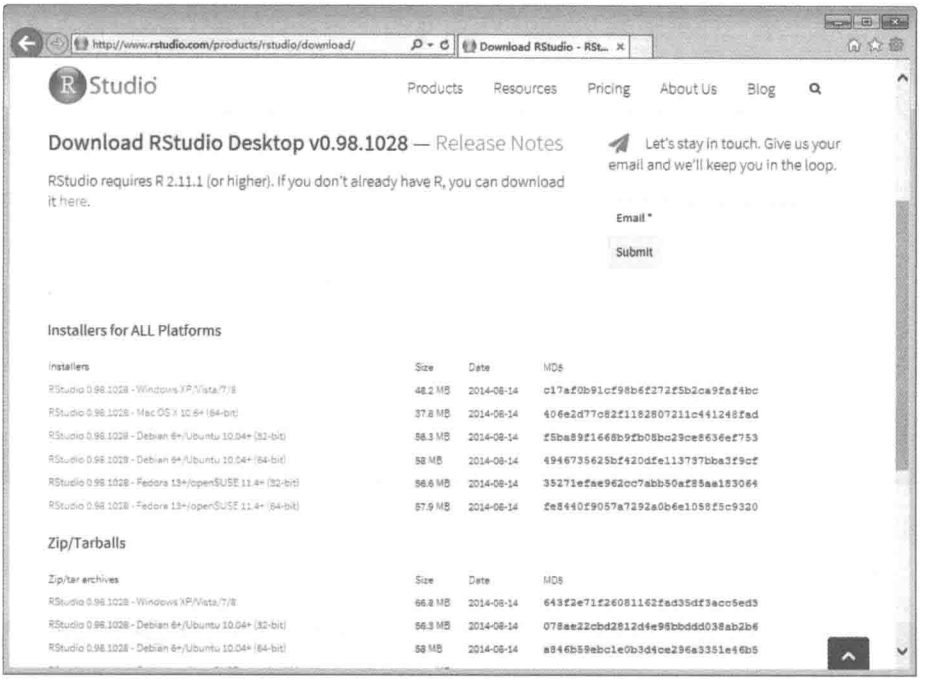

图 1 - 10　RStudio 软件下载流程界面之三

编者撰写本书时 Rsudio 软件版本为 RStudio 0.98.1028，读者点击“RStudio 0.98.1028 - Windows XP/Vista/7/8”下载 RStudio 软件，然后运行安装 RStudio 软件，安装过程一般选择默认参数安装。

1.4　RStudio 软件的使用

点击打开“开始 → 所有程序 → RStudio → RStudio”，启动 RStudio 软件，如图 1 - 11 所示。

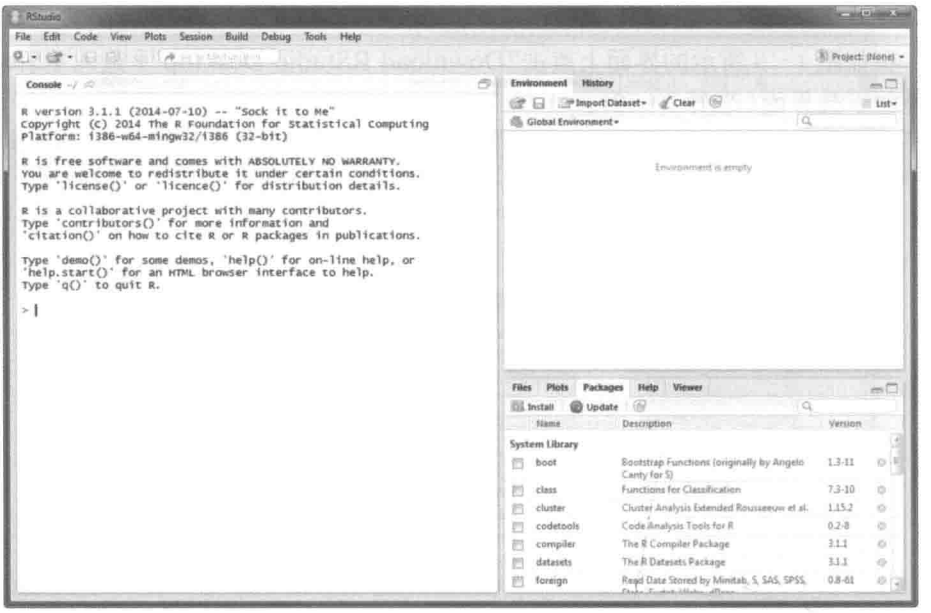

图 1-11　RStudio 软件初始界面

新建一个编辑页面，点击“File → New File → R Script”，如图 1-12 所示。

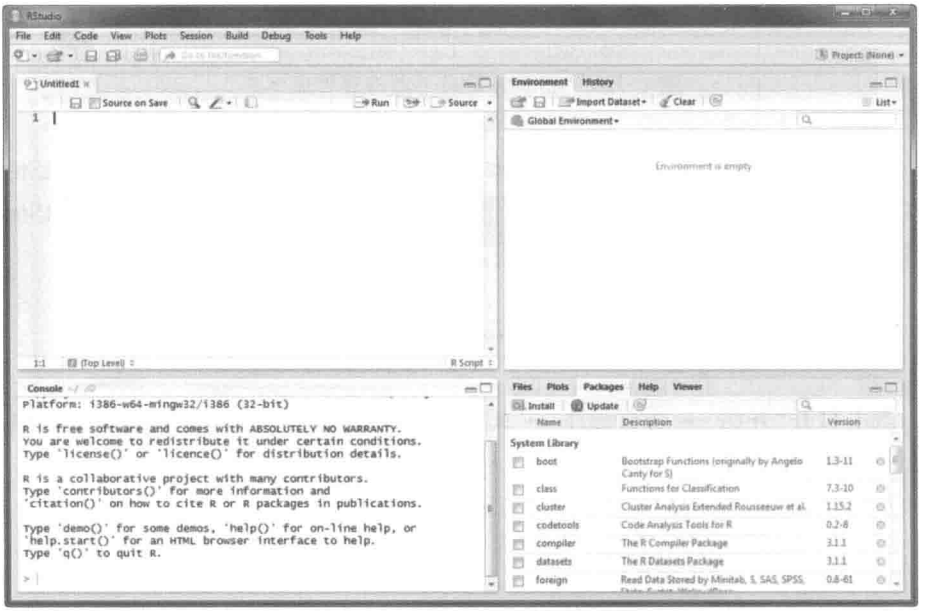

图 1-12　RStudio 软件脚本编辑界面

在 RStudio 软件的脚本编辑界面分成了 4 个子窗口，具体功能如下：

左上窗口：用于 Script 脚本的编写，编写好的脚本保存之后，在窗口左上方的

标签将显示以.R 为后缀的文件名。

左下窗口：R 语言编辑器运行的 Console 终端，在这里运行 R 语言编辑器的各种命令，这个窗口和之前介绍过的 R 语言图 1－6 界面的功能基本一致。

右上窗口：这个窗口包括两个部分，Enviroment 显示运行环境中的变量和产生的数据信息；History 记录执行过的命令。

右下窗口：这个窗口包括 Files（文件），显示当前路径下的文件信息；Plots（图形界面），R 语言程序绘制出的图片显示在这里；Packages（包）显示 Rstudio 软件当前已下载的程序包和包的相关信息；help（帮助）帮助文件的部分及在本地网页上显示的 viewer。

【使用小技巧】

（1）在执行命令编辑过程中，如果忘记某一函数的具体拼写，可以使用 Tab 键给出提示。例如，想要使用 names 函数，但忘记函数 name 后面是否需要加 s，只需键入“nam”后按 Tab 键，就会自动跳出当前环境下所有可运行以 nam 开头的函数等信息。

（2）在左上的脚本编辑窗口编辑完成某一命令行后，并把光标放置到该行，然后按“Ctrl＋回车”键就可在 console 窗口运行该命令行，若需要运行整个脚本，可同时按“Ctrl＋Shift＋回车”键。

（3）光标在 script（左上）和 console（左下）窗口的快速切换分别使用“Ctrl＋1”“Ctrl＋2”键。

（4）查看函数的帮助说明，使用“?”。例如，“?names”，运行后在右下窗口的 help 界面就会出现关于 names 函数的介绍说明。

(5) 点击“Tools → Global Options → Appearance”，可以更改文字的字体和大小等。

参考文献

[1] R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www. R-project.org/.

[2] RStudio Team (2015). RStudio: Integrated Development for R. RStudio, Inc., Boston, MA URL http://www.rstudio.com/.

第2章 R语言基础

本章重点介绍与R语言相关的函数等命令语句的基础知识。因本书的篇幅和结构安排，此章仅对R语言做一个基本介绍，编者建议初次接触R语言的读者在阅读本书的同时应翻阅学习更多介绍R语言的书籍和资料来拓展和加深对于R语言的理解。

2.1 R语言的对象

首先介绍R语言中的“对象”，这个词来自英文单词“objects”，“objects”在字典中也翻译成物体、目标，单从字面上就可以看出，R语言中的“对象”应该是分析的主体，也是数据交换、运算和储存的载体。所以无论是原始数据，还是运算分析中的临时数据及最终生成的结果都储存在“对象”中。因此，下面先介绍如何创建和删除对象。

通常情况下，我们使用<-或者=进行R语言对象的创建和赋值，举例如下：

```
> X<-3
> X
[1] 3
> Y=c("a")
> Y
[1] "a"
```

上面第一条命令使用“<-”符号把数值3赋值给了对象X[①]，如果之前程序中没有对象X，那么这也创建了对象X；利用“=”符号把字符a赋值给对象Y。

```
> ls()
[1] "X" "Y"
```

ls() 函数可以将当前运行环境中存在的对象列出来。

① 为了方便阅读，本书中对程序进行说明的各量均与程序保持一致而采用正体。——编注

```
> rm(X)
> ls()
[1] "Y"
```

rm()函数用来删除对象，如果想删除当前下的所有对象，可以使用命令 rm(list=ls())。

R 语言的对象按结构可分成不同的类型(class)，常见的有：向量(vextor)、因子(factor)、数据框(data.frame)和列表(list)。若按 R 语言中对象的内在特征分类，对象的样式(mode)可分为：数值(numeric)、字符(character)、逻辑(logical)等，使用 class 函数和 mode 函数可以查询对象属性。

2.2　向量(vector)

向量是最简单的一类对象，通常是存储一组数字，或者是一些字符串，下面结合实例详细说明。

2.2.1　向量的生成赋值

```
> X<-c(2,4,6,8,10,12)
> X
[1]  2  4  6  8  10  12
> mode(X)
[1] "numeric"
```

c()函数把 2,4,6,8,10,12 这 6 个数值赋值到向量 X 中，此时对象 X 为数值型向量。

```
> Y<-c("red","green","white","black","yellow")
> Y
[1] "red"   "green"   "white"   "black"   "yellow"
> mode(Y)
[1] "character"
```

red,green,white,black 和 yellow 5 个字符串被赋值到向量 Y 中，Y 成为字符型向量。

```
> seq(2,length = 4,by = 5)
[1]  2  7  12  17
```

seq() 函数定义生成一组有规律的数值，此例中生成 4 个数据，间距大小为 5。

```
> rep(1:3,each = 2,time = 3)
[1] 1 1 2 2 3 3 1 1 2 2 3 3 1 1 2 2 3 3
```

rep() 函数定义一组伴有重复的数据，读者需仔细体会其用法。

```
> runif(10,0,3)
[1] 0.8035489 0.1904190 0.4598794 1.8211824 2.7775007 0.3806452
    1.2560721 1.0142864 1.6222667 2.7075225
```

runif() 函数生成 10 个在 0 到 3 之间均匀分布的随机数字。

```
> rnorm(10,mean = 2,sd = 4)
[1] 4.8993378 1.1835671 0.1139385 0.4937648 6.8683881 7.6014563
    6.0237714  5.9316288 -4.0229830  4.4394594
```

rnorm() 函数生成 10 个 mean 为 2，SD 为 4 的符合正态分布的随机数字。

2.2.2 向量的提取

```
> X<-c(3,7,12,8,10,4,9,15,21,9)
> X[3]
[1] 12
```

利用 X[ ]提取特定位置的元素。

```
> X[3:6]
[1] 12  8 10  4
```

提取从位置排列从 3～6 之间的元素。

```
> X[X>12]
[1] 15 21
```

提取向量 X 中数值＞12 的元素。