概率统计及其R语言实现

王志良　程慧慧　娄妍　黄春艳　李鹏　译著

黄河水利出版社

概率统计及其R语言实现

王志良　程慧慧　娄　妍　黄春艳　李　鹏　译著

黄河水利出版社
·郑州·

内 容 提 要

本书主要介绍概率统计的基本知识，及其如何用 R 语言实现。内容包括 R 语言、概率、离散型分布、连续型概率分布、多元分布、抽样分布、参数估计和假设检验。在每一章节中，先介绍相关的概率统计的基本知识，然后给出对应的 R 语言。

本书相对于一般的概率统计方面的著作，更注重于与 R 语言相结合，强调其可操作性和实用性。

图书在版编目(CIP)数据

概率统计及其 R 语言实现/王志良等译著. —郑州：黄河水利出版社，2016.5

ISBN 978-7-5509-1429-2

Ⅰ.①概… Ⅱ.①王… Ⅲ.①概率统计-高等学校-教材②程序语言-程序设计-高等学校-教材 Ⅳ.①O211②TP312

中国版本图书馆 CIP 数据核字(2016)第 108824 号

组稿编辑：李洪良 电话：0371-66026352 E-mail：hongliang0013@163.com

出 版 社：黄河水利出版社

地址：河南省郑州市顺河路黄委会综合楼 14 层 邮政编码：450003

发行单位：黄河水利出版社

发行部电话：0371-66026940、66020550、66028024、66022620(传真)

E-mail：hhslcbs@126.com

承印单位：河南新华印刷集团有限公司

开本：787 mm×1 092 mm 1/16

印张：11

字数：254 千字 印数：1—1 000

版次：2016 年 5 月第 1 版 印次：2016 年 5 月第 1 次印刷

定价：35.00 元

前 言

本书是作者在扬斯敦州立大学上学时一本名为《概率论与数理统计》课本的扩展，大部分内容是基于读研究生时的笔记，主要内容是概率论与数理统计。

本书分为三个基本部分：第一部分包括基本的描述性统计；第二部分是概率论；最后一部分包括点估计和区间估计，以及假设检验，并介绍了应用统计的相关知识。

对于概率和数理统计的学习，读者可以参看豪格和坦尼斯的《概率和统计推断》，卡塞拉和伯杰的《统计推断》，莱曼的《点估计理论》和《检验统计假设》。关于 R 的书籍，推荐 Dalgaard 的《统计与 R》和 R Verzani 的《R 在统计中的应用》。特别是这两本免费的书：迈克尔拉文的《介绍统计思想》、Grinstead 和 Snell 的《概率介绍》，也同样值得推荐，也最终说服作者免费开放 IPSUR。

请记住这本书的标题是“概率统计及其 R 语言实现”，而不是“在 R 中概率论与数理统计的介绍”。只要安装、导入和读取就可以使用 IPSUR

```
> install. package("IPSUR")
> library(IPSUR)
> read(IPSUR)
```

这本书能够成功，正是因为博林格林州立大学的教授们坚实的数学和统计学基础，他们是 Drs. Gábor Székely, Craig Zirbel, Arjun K. Gupta, Hanfeng Chen, Truc Nguyen, James Albert，同时也要感谢 Drs. Neal Carothers 和 Kit Chan。

还要感谢扬斯敦州立大学的同事们对我的支持，特别要感谢 Dr. G. Andy Chang，同样感谢 Richard Heiberger。最后，感谢我的妻子。

目 录

第 1 章　概率与数理统计介绍

这一章节已经有很多人都总结得很好了。正因为如此，不再多说，建议刚开始学习的人可以从 Wikipedia 开始，它确实是不错的学习资源。

在我的讲稿中，常常介绍关于 Fisher、Galton、Gauss、Laplace、Quetelet 和 Chevalier de Mere 的事情。

1.1　概　率

概率起源于 1 000 多年前，但是当时没有得到数学家的关注，直到 1654 年，Chevalier de Mere 提出如果一场赌局不得不因故中断的时候如何对两个人进行公平的赌资分配的问题时，才得到广泛关注。

1.2　统　计

统计关心的是数据以及对它们的收集、分析和理解。在本书中主要区分两种统计：描述性统计和推理性统计。

描述性统计关心数据的综合处理，就是利用多种方法对数据集进行描述，通常需要描述方法或手段对数据进行计算，比如百分比、求和、平均数等。

推理性统计做得更多。根据数据集做出推理，从最原始的数据集得出更多的结论。

我想提一提两种统计学派：频率学派和贝叶斯学派。这两种学派的区别在于如何对待潜在概率。频率学派更认可在 20 世纪早期的 Fisher、Neyman 和 Pearson 的基础工作，并且这个学派一直因为简单的计算功能占据统治地位，而现在贝叶斯学派得到了更多的、持续增长的关注。

本书致力于频率学派的观点，并且在 4.8 节和 7.3 节中有一些实例说明，在后续的版本中将加入贝叶斯学派观点的讨论。

第2章　R简介

2.1　下载和安装R

R的获取与用户的硬件和用户使用的操作系统有关,R项目组已经给出了安装说明和精确的使用说明书,告诉用户如何使用R,并且该如何得到关于R更多的信息。下面给出初学者的入门说明书。

2.1.1　安装R

根据用户使用的操作系统,选择以下链接下载最新的R版本:

Microsoft Windows:http://cran.r-project.org/bin/windows/base/

MacOS:http://cran.r-project.org/bin/amcosx

Linux:http://cran.r-project.org/bin/linux

对于Microsoft Windows用户,点击R-x.y.z.exe开始安装。当出现"Customized startup options"选项,点击"Yes"。在下一步界面窗口,确保选择SDI(single document interface)选项,这个选项对于使用rgl包画三维图像非常有用。

使用USB驱动安装R(在Windows系统下)。

使用这种选项,可以在没有管理员授权的情况下方便地使用R。使用过程如下:

(1)使用前面介绍的Windows操作系统下的安装软件,正常安装。当询问安装路径时,选择USB驱动选项中top-level选项替代默认的C盘驱动器。

(2)当询问是否修改Windows注册表时,不做选择,即不对注册表做任何修改。

(3)安装后,将R-x.y.z这个文件夹的名字修改为R(如果可能,在第一步修改即可)。

(4)选择USB驱动器的top-level选项,下载下面的快捷方式,并且将其与R文件夹放在一起,不要将它放在文件夹内部。

http://ipsur.r-forge.r-project.org/book/download/R.exe

使用快捷方式运行R。

2.1.2　安装和加载额外包

R有基础包(安装R时自带的包),还有贡献包(这个必须下载安装)。比如:现在使用的R版本中自带的默认的R基础包安装后启动如下:

```
>getOption("dfaultPackage")
[1] "datasets"  "utils"  "grDevices" "graphics" "stats" "methods"
```

R中一些被称为"R核心包"的基础包是由一些志愿者编写的。除了基础包,还有很

多志愿者在致力于贡献包的编写,这些志愿者来自于世界各地。这些贡献包保存在被称为“R综合典藏网”(简称“CRAN”)的镜像里,这个镜像位于世界各地。给出任何一个活跃的网络连接,每个人都可以从CRAN免费地下载和安装这些贡献包,甚至可以查看源代码。

如果要安装名为foo的包,打开R并且键入install.package(“foo”)。如果想要安装foo和安装foo所依赖的所有其他包,只需键入install.package(“foo”,depends=TRUE)。

总命令install.package()(在大多数操作系统)打开了一个界面窗口,这个窗口包含了一系列可用包,最简单的就是选择一个或多个可用包进行安装。

不管在系统中已经安装了多少包,在使用每一个包之前都必须先使用library函数将它们先加载到库函数中去。例如,foreign包包含了从别的软件如spss、SAS和其他相关软件将数据导入到R中的功能,但是这些功能只有在使用了library(foreign)这个函数命令后才可以使用。

在命令提示符下输入library()(详细描述如下)查看所有可用的软件包。

更多关于R的安装、额外包的安装以及关于R的完整精确的信息,可以参见R安装和用户使用手册:http://cran.r-project.org/manuals.html。

2.2 连接R

一次一行:这是最基本的方法,是初学者的首选方法。

一次多行:对于更长的被称为脚本的程序需要在命令提示符下一次性写入很多代码。而且,对于更长的程序来说,仅仅修改其中一部分特定的脚本就可以很方便地再次在R中运行。一个被称为脚本编辑器的程序是专门用来帮助编写这些代码过程的。它包含各种各样非常有用的功能,诸如R符号标亮,代码自动完成,分隔符匹配,并且提供实时动态帮助。甚至,它还有像Microsoft Word那样的文本编辑功能的文本编辑程序的功能,而且,大多数的脚本编辑器是可以完全自定义的,用户可以自定义外观界面的显示,选择什么样的颜色,什么时候显示它们以及如何显示它们。

R编辑器(视窗):在Microsoft窗口界面,RGui又被称作R编辑器的内置脚本编辑器。在控制窗口,选择File→New Script,打开一个脚本窗口,便可以在窗口中编写代码。当完成代码时,用户可以标亮所有满足要求的命令,然后按下Ctrl + R键,该命令可以在R中自动运行并显示输出结果。想要保存该脚本,请单击File→Save as...按钮。在R编辑器中,点击File→Open Script可以重新打开刚才的脚本。

R文本编辑器:这个选项和Latex中WinEdt编辑器是一致的,而且具有附加功能,诸如代码标亮显示,远程资源和许多其他的事情。但是,你首先需要下载并安装一个共享版的程序WinEdt,这只是试用版本,一段时间后会要求填写注册信息。如果你已经拥有WinEdt或近期想购买它,RWinEdt是一个非常不错的选择。

Tinn-R/Sciviews-K:是完全免费的,包含上述介绍的所有选项,甚至更多。用户在安装后可以立即使用它,使用很简单。但是,Tinn-R只适用于Microsoft Windows操作系统。如果你使用的是MacOS或Linux操作系统,可以用Sci-Views-Komodo编辑器代替。

Emacs/ESS:Emacs是一个多功能的文本编辑器。它可以做比如修改、搜索、编辑和操作的许多事情。而且如果Emacs做不了这些,那么你可以编写一个扩展程序使Emacs做到。扩展后的名字称为ESS,它支持Emacs Speaks Statistics。使用ESS,用户可以和R进行对话,做所有别的脚本编辑器所做的所有工作,甚至更多。关于更多安装细节、文档、参考卡,以及更多介绍请参考http://ess.r-project.org。

公平警告:如果你想使用Emacs,但你已经习惯使用Microsoft Windows和MacOS操作系统,那么你将用一生重新学习你所知道的电脑知识(或者既然Emacs是完全可定义的,你可以重新配置运行Emacs,使得它更符合你自己的使用习惯,得到你想要的Emacs的行为方式)。

JGR(读作"Jaguar"):这是基于Java的RGui的且含有警示功能的附加功能,所以它可以在多个操作系统上进行工作。它有类似R编辑器的脚本编辑器,但又具有附加的诸如符号标亮的更多功能。如果你不使用Microsoft Windows(或者即使你使用),你一定要看看这个。

Kate,Bluefish等,还有很多其他可用的文本编辑器,其中有一些是免费的,而且各有各的优点。因此,只介绍一些经常使用的和比较感兴趣的跟大家分享一下。你们也可以试着了解下,看看它们究竟能做什么。

图形用户界面(GUI):是指用户通过选择,点击菜单上的某些内容来操作R的界面。同样,它有很多的选项,只选我用过和熟悉的。其他一些更受欢迎的脚本编辑器可以从R项目网站下载:http://www.sciviews.org/_rgui/。在屏幕的左边(R项目的下面)有几个选项可用。

R Commander:提供了一种点击式的方式进行基本的统计工作。它被称为"Commander",因为每次从菜单中进行一项,在输出窗口中列出对应相关任务的代码。用户可以将这个代码复制粘贴到一个文本文件中,然后在没有R Commander的支持下重新运行代码。它非常适合入门级别的人使用。

Poor Man's GUI:是Rcmdr的另一种形式,基于GTk而不基于Tcl/Tk。作者(我)使用它已经有一段时间了,并且非常喜欢使用。在这里可以查看更多信息:http://wiener.math.sci.cuny.edu/pmg/。

Rattle:是用来管理或分析大数据的数据挖掘工具包。值得一提的是,它提供了很多其他功能。更多信息参见参考文献[1]。

Deducer:是相对较新的,在我所了解的范围内有较大改进。但我还没有在教学中用过它。

2.3　R的基本操作和概念

R开发人员写了一篇名为"R简介"的介绍性文档。其中介绍了关于R基本操作的示例会话。建议所有新用户阅读该文档,但必须牢记,文档里提出的概念可能对一个初学者而言是陌生的。

下面是R的最基本操作。几乎每一本关于初学者学习R的书中都从下面这些内容

入手。看看在最初阶段R所有能做的基本操作。

2.3.1 数值

```
> 2+3  #加法
[1] 5
> 4*5/6  #乘法和除法
[1] 3.333333
> 7^8  #7的8次方
[1] 5764801
```

注意注释字符#,在符号#后的任何事物都被R忽略掉了。20/6是一个循环小数,但是上面的例子只给出了7位数字。可以采用下面的方法修改数字显示的位数。

```
> options(digits=16)
> 10/3  #可以看到更多的数字
[1] 3.333333333333333
> sqrt(2)  #开方
[1] 1.414213562373095
> exp(1)  #欧拉常数e
[1] 2.718281828459045
> pi
[1] 3.141592653589793
> options(digits=7)  #重新返回默认状态
```

注意可以允许将数字设置到22,但是不建议将数字设置超过16(更多的数字是不显示的)。注意上述的sqrt函数是求平方根函数,exp函数是e的次方。

2.3.2 赋值、对象名字和数据类型

可以很方便地对后面要用到的变量赋值。对一个变量赋值的正确方法是采用"<-"算子。"="也可以做同样的工作,只是R的创建者使用"="表示别的特殊的函数参数符号。在这本书中采用他们的建议使用"<-"符号进行赋值,一旦对一个变量进行赋值后,就可以很方便地使用这个变量名字显示它的值了。

```
> x<-7*41/pi  #看不到计算的结果
> x  #看结果
[1] 91.35494
```

对一个变量赋值是可以使用字符、数字、点".",或者下划线"_"字符。不能使用数学算子,并且数字后面不能跟点。有效的命名例子:x,x1,y.value或y_hat。(更准确地讲,项目名字可使用的字符依赖于每个人的系统和设置,参看R介绍了解更多关于这方面的讨论)

对象可以是多种类型、模式和种类。在这种情况下,没有必要去探讨所有的错综复杂的数据类型,但是有一些数据类型是需要你熟悉的。

整型:数值0,+1,+2,…,在R中可以精确表示。

双精度型:实数(合理的,不合理的);这些数字不是很精确的表示(保存整数或分母为2的整数次幂的分数,参见[2]。

字符型:包含在"或'符号内的元素。

逻辑型:包含TRUE、FALSE和NA(保留字),NA代表不可使用,比如缺失数据。

可以利用typeof函数确定一个对象的类型。除了上面所描述的数据类型,还有复数类型。

```
> sqrt(-1)  #没有定义
[1] NaN
> sqrt(-1+0i)  #定义了
[1] 0+1i
> sqrt(as.complex(-1))  #做同样的事
[1] 0+1i
> (0+1i)^2  #应该是-1
[1] -1+0i
> typeof((0+1i)^2)
[1] "complex"
```

注意你可以输入(1i)^2得到同样的结果。NaN代表没有数字,内在的来讲是双精度型。

2.3.3　向量

上面已经介绍了长度为1的向量的操作。现在输入多行向量。

2.3.3.1　输入数据向量

1. c函数

如果你想将74,31,95,61,76,34,23,54,96输入到R,你可以用c函数创建一个数据向量(c是concatenate联系在一起的缩写)

```
> x<-c(74,31,95,61,76,34,23,54,96)
> x
[1] 74 31 95 61 76 34 23 54 96
```

向量里的元素经常被R强制性的转换成最基本的类型,因此如果你输入c(1,"2"),结果将会是c("1","2")。

2. scan函数

这个方法对于存储在别的某个地方的数据是非常有用的。比如,可以在提示符下键入x<-scan(),R将会显示1:等待输入第一个数据元素。输入一个数据元素后按下回车键,将会显示2:,如此反复。注意输入一个空行结束输入。这种方法对于存在一个文档里或是电子表格里的一列数据是非常方便的。你可以在出现提示符时进行复制粘贴,R会立即将它们作为向量存储起来。

3. 重复数据

正常规则模式：seq 函数会产生所有类型的数据序列。它含有参数 from，to，by 和 length. out。下面给出几个实例说明它们如何工作：

```
> seq(from = 1, to = 5)
[1] 1 2 3 4 5
> seq(from = 2, by = -0.1, length. out = 4)
[1] 2.0 1.9 1.8 1.7
```

注意，我们可以使用冒号更快地得到上述的第一个例子：

```
> 1:5
[1] 1 2 3 4 5
```

向量 LETTERS 得到 26 个大写英文字母，letters 得到 26 个小写英文字母。

2.3.3.2 索引数据向量

有时候不希望得到整个数据向量，只想得到它的一部分。可以采用[ ]算子快速得到其中的部分数据。观察（x 是前面定义过的）以下程序：

```
> x[1]
[1] 74
> x[2:4]
[1] 31 95 61
> x[c(1,3,4,8)]
[1] 74 95 61 54
> x[-c(1,3,4,8)]
[1] 31 76 34 23 96
```

注意使用负号表示不需要的数据元素。

```
> LETTERS[1:5]
[1] "A" "B" "C" "D" "E"
> letters[-(6:24)]
[1] "a" "b" "c" "d" "e" "y" "z"
```

2.3.4 函数和表达式

函数输入相应参数并且返回一个输出对象。函数可以做很多事情。下面给出一些实例：

```
> x < -1:5
> sum(x)
[1] 15
> length(x)
[1] 5
> min(x)
[1] 1
```

```
> mean(x)  #均值的例子
[1] 3
> sd(x)  #标准差
[1] 1.581139
```

使用R不久之后,如果用户想知道一个特定的函数是如何工作的,因为R是开源的,每一个人都可以免费去查看这个函数,研究这个函数究竟如何工作。更详细的介绍可以参看由Uwe Ligges编写的文章“进入源代码”。

简要介绍如下:

(1)输入不带圆括号和任何参数的函数名。如果幸运的话你可以查看整个函数的源代码。例如:假设想看intersect这个函数是如何工作的:

```
> intersect
function (x,y)
{
    y <- as.vector(y)
    unique(y[match(as.vector(x),y,0L)])
}
<environment: namespace:base>
```

(2)如果输入UseMethod(“something”)这个函数,那么需要选择输入对象的类型并且参看和这个对象匹配的方法。比如,输入rev:

```
> rev
function (x)
UseMethod("rev")
<environment: namespace:base>
```

这个输出告诉我们有很多和rev函数相关的方法。想看看有哪些,可以输入:

```
> methods(rev)
[1] rev.default      rev.dendrogram*
Non-visible functions are asterisked
```

现在发现有两种不同的rev(x)函数,其中有一种选择依赖x的值,还有一种是dendrogram对象和别的默认的方法。很简单的方式是键入名字看看每一种方法是如何工作的。比如,默认的方法描述如下:

```
> rev.default
function (x)
if (length(x)) x[length(x):1L] else x
<environment: namespace:base>
```

(3)有一些函数名字在命名空间被隐含了(参看R的简介,见参考文献[2]),并且在第一次尝试的时候是不可见的。比如,如果想看看wilcox.test的代码,得到如下代码:

```
> wilcox.test
function (x,...)
```

```
UseMethod("wilcox.test")
<bytecode: 0x08bc4b20>
<environment: namespace:stats>
> methods(wilcox.test)
[1] wilcox.test.default* wilcox.test.formula*
Non-visible functions are asterisked
```

如果想了解 wilcox.test.default,可能会得到“没有找到”的错误,因为它被隐藏在 stats 包的名字空间中。在这种情况下要在函数名字的前面加上空间的名字的前缀,并且用 3 个冒号连接;这个命令 stats:::wilcox.test.default 将会显示源代码,在这里就不再赘述了。

(4)如果想查看 Internal(something) 或 Primitive(“something”),那么下载 R 的源代码是很有必要的,并且可以查看内部代码。更多的讨论参看 Ligges[3]。exp 的例子如下:

```
> exp
function(x)  .Primitive("exp")
```

注意,大多数的 Internal 函数采用其他计算机语言描写,初学者可能不明白,至少是刚开始接触的时候不是很理解。

2.4 获得帮助

当你使用 R 时,你会很快需要帮助。幸运的是,R 有海量的帮助资源并且你可能会很快熟悉。这只需要在 RGui 界面上点击 Help 按钮,就可以得到如下选项:

(1)Console(操作):给出有用的快捷键,比如 Ctrl + L 清除 R 操作屏。

(2)FAQ on R(R 中常见问题解答):R 基本操作的常见问题。

(3)FAQ on R for Windows(Windows 系统下 R 中常见问题解答):针对 Windows 操作系统下 R 操作中的常见问题。

(4)Manuals(手册):R 系统的所有特征的技术手册。包括安装、完整语言定义和附加包。

(5)R function(text)…(R 函数(文档)):如果你清楚知道函数的名字,使用它就可以了解关于这个函数的更多信息。比如,mean 或 plot 函数。在窗口命令行中输入 mean 和输入 help(“mean”)的效果是一样的,更为简单的方法就是直接输入? Mean。但是需要注意的是,你所感兴趣的函数必须是已经利用library函数安装过的。

(6)HTML Help(HTML 帮助):使用点击式的链接方式浏览帮助使用手册。它同样可以采用搜索引擎和查找关键词来寻找相应的帮助页面标题,同样可以利用点击式的方式得到查询结果。这对于初学者来说可能是最好的帮助方法,可以在命令行输入 help.start()开始。

(7)Search help…(查找帮助):如果不知道感兴趣的函数的确切名字,或者如果还没有下载包含这个函数的包,可以使用这种方法。比如,可以进入 plo 并且文档窗口将会使用正则表达式返回和 plo 相匹配的所有帮助信息;在命令行输入 help.search(“plo”)可以

得到同样的结果。这个优点在于不需要知道函数的确切名字;不足之处在于不能采用点击式的方式得到这种结果,但你可以在命令行中使用?? Plo 调用 HTML Help 搜索引擎来代替。

(8) Search. r-project. org(查找 R-project. org):可以在帮助列表中搜索关键词并且通过邮件的方式进入 R-project,它可以帮助我们查找别的用户是否也遇到过同样的问题。

(9) Apropos(恰当的)…:使用它可以得到更多复杂的、部分匹配的函数。参看?? apropos。

在函数的帮助页面底部列出了一些"实例",如果在命令行复制粘贴它们就可以自动运行。比如,可以使用 example(mean)查看一些实例,看看 mean 函数是如何工作的。

有一些和 R 相关的邮件列表,并且有许多人回答和 R 相关的问题。参看 http://www. r-project. org/mail. html 可以得到非常有用的想法。特别关注页面底部所列出的和 R 相关的特别感兴趣小组(SIGs)。

请牢记 R 是一个免费软件,这就意味着 R 是由一些志愿者编写的,经常更新邮件列表的志愿者不需要客户支付费用。因此,如果你想使用邮件列表得到免费的建议,你必须遵守基本的规则,或者你可能根本就没有得到答复,甚至更糟糕的是,你得到了一些你并不需要的回复。

下面是一些建议:

(1)参看常见问题指导(http://cran. r-project. org/faqs. html)。注意不同的操作系统有不同的问题指导。你可以阅读它们,甚至如果没有问题也可以关注 R 的一些特性。

(2)查找档案。如果你的问题并不是常见问题,那也有可能你问的问题和邮件列表中出现的问题很相像。如果你想了解 foo 这个话题,那么你可以使用 RSiteSearch("foo")来寻找邮件列表档案(和在线帮助)来查找它。

(3)进行 google 引擎搜索和 RSeek. org 搜索。

如果你的问题不是常见问题,并且在 R-help 中也没有遇到过,因此也不可能产生 Google(或别的)搜索,那么你可以考虑写 R-help。

下面给出一些额外建议:

(1)在邮寄前参阅邮件向导(http://www. r-project. org/posting-guide. html),这会免去你很多的麻烦。

(2)去掉命令提示符(>)。阅读你邮件的人会直接从你的邮件中复制粘贴到 R 会话中。如果你不给别人添加麻烦,那么会得到更多回报。

(3)问题往往涉及特定的数据集,而最好的处理数据的方法是用 dump 命令。例如,如果你的问题涉及一个存储名为 X 的数据,你可以在命令提示中键入:dump("X"," "),并将结果复制粘贴到你的电子邮件中,然后读者就可以很容易地从你的电子邮件中复制粘贴到 R 中,直接运行得到 X 的值。

(4)有时候问题的答案与使用的操作系统、附件包和 R 的具体版本有关。sessionInfo( )命令收集所有这些复制粘贴到电子邮件的信息。