R语言介绍

目录

R语言来源

- R语言是从S统计绘图语言演变而来
- S语言是一种用来统计编程的语言，自1976年以来由贝尔实验室开发的数据分析交互式环境 ，后来基于S语言开发Splus的是一个商业软件。
- 1992年由新西兰奥克兰大学统计学系Ross Ihaka和Robertn Gentleman于1995年基于s语言的源代码，编写了一个能执行s语言的软件，这就是R软件，其命令统称为R语言。

R语言特点

（1） R免费开源，软件体积小，可根据需要安装扩展包，兼容各种常用操作系统，包括Windows、Mac OS X和Linux；

（2）专门为统计和数据分析开发的语言，有丰富的扩展包，是一个全面的统计研究平台；

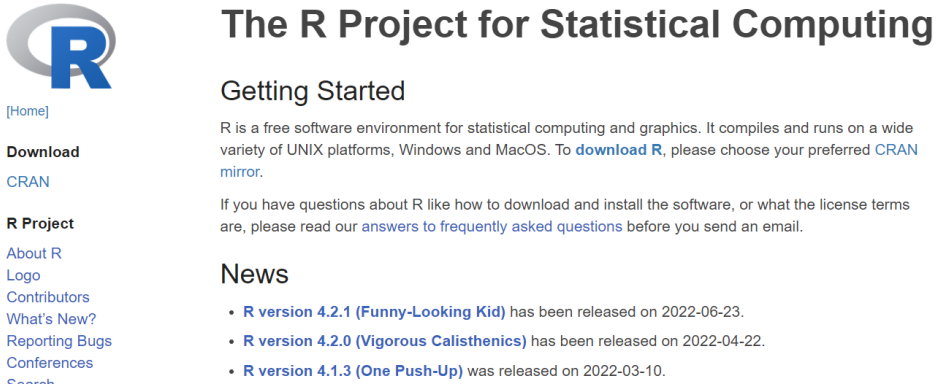
（3）拥有顶尖水准的制图功能；

（4）R可以轻松地从各种类型的数据源导入数据，它同样可以将数据输出并写入到这些系统中；

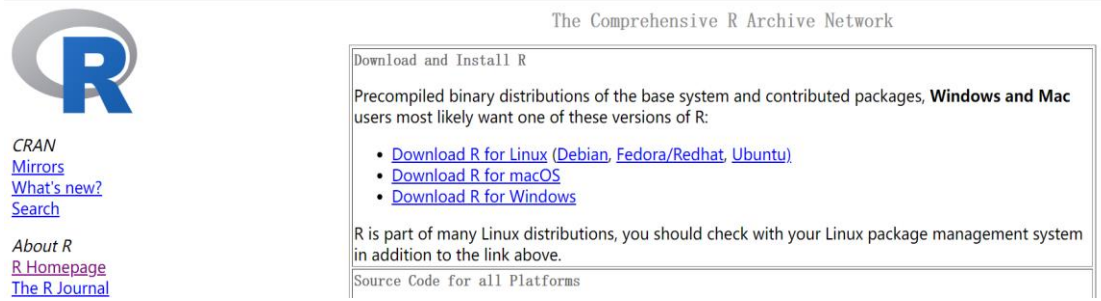
（5）面向对象，简单易学。

R下载与安装

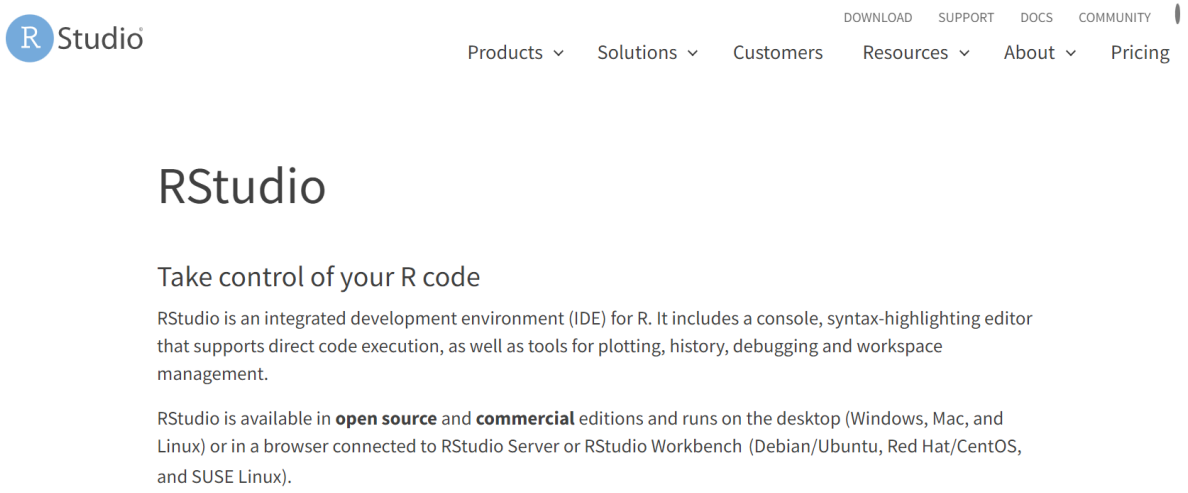
R下载链接
https://www.r-project.org/



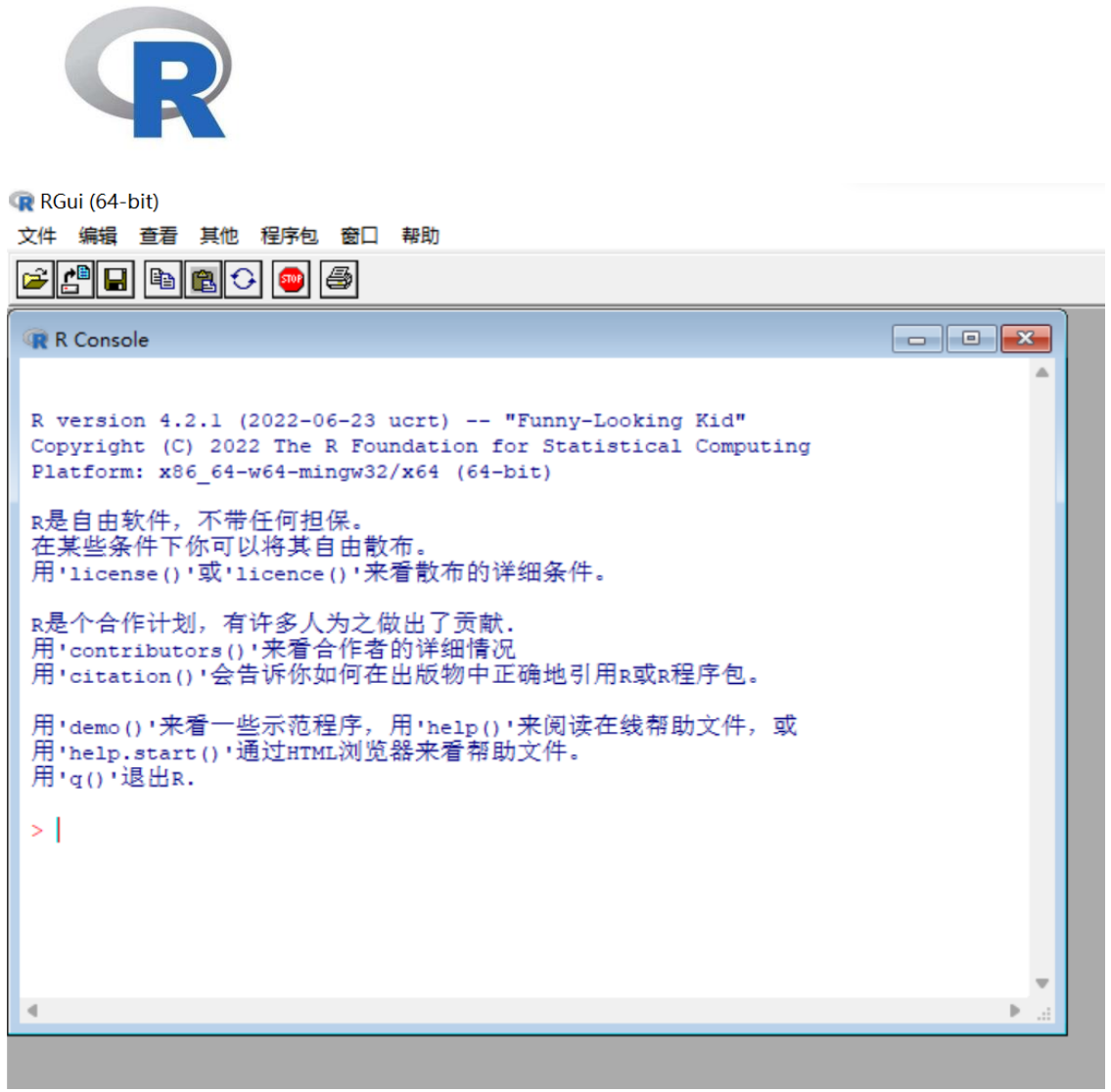
http://ftp.ussg.iu.edu/CRAN/

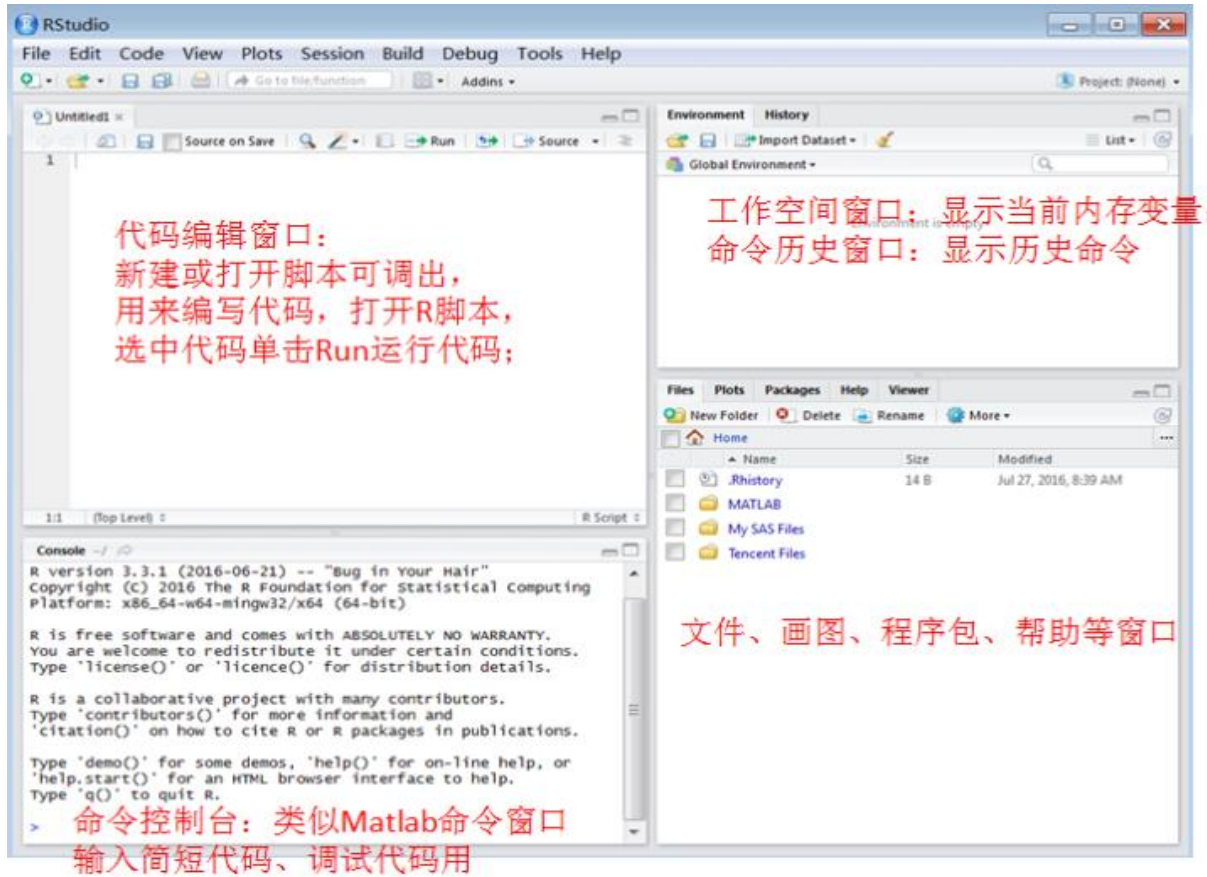


Rstudio 下载
https://www.rstudio.com/products/rstudio/



R下载与安装


RGui (64-bit)
文件 编辑 查看 其他 程序包 窗口 帮助
R Console
R version 4.2.1 (2022-06-23 ucrt) -- "Funny-Looking Kid"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R是自由软件，不带任何担保。
在某些条件下你可以将其自由散布。
用'license()'或'licence()'来看散布的详细条件。
R是个合作计划，有许多人为之做出了贡献.
用'contributors()'来看合作者的详细情况
用'citation()'会告诉你如何在出版物中正确地引用R或R程序包。
用'demo()'来看一些示范程序，用'help()'来阅读在线帮助文件，或
用'help.start()'通过HTML浏览器来看帮助文件。
用'q()'退出R.
>


RStudio
File Edit Code View Plots Session Build Debug Tools Help
代码编辑窗口：
新建或打开脚本可调出，
用来编写代码，打开R脚本，
选中代码单击Run运行代码；
工作空间窗口：显示当前内存变量；
命令历史窗口：显示历史命令
文件、画图、程序包、帮助等窗口
R version 3.3.1 (2016-06-21) -- "Bug in Your Hair"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
命令控制台：类似Matlab命令窗口
输入简短代码、调试代码用

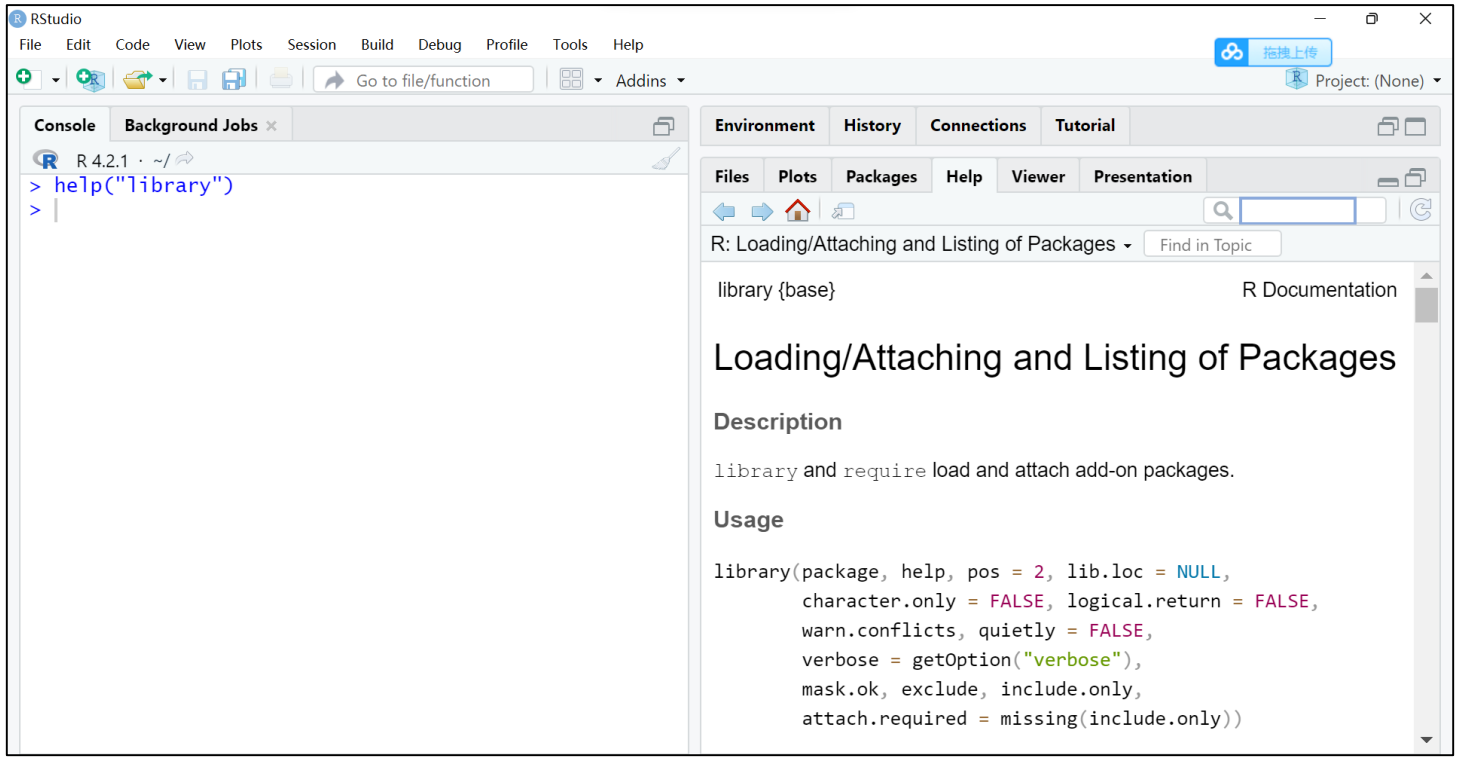
R获取帮助

>help.start( ) 打开帮助文档首页

>help("library") 或 >?library 查看函数foo的帮助（引号可以省略）

>example("foo") 函数foo的使用示例（引号可以省略）

>data() 列出当前已加载包中所含的所有可用示例数据集



R包（package）

包（package）是R函数、数据、预编译代码以一种定义完善的格式组成的集合。计算机上存储包的目录称为库（library）,R自带了一系列默认包，它们提供了种类繁多的默认函数和数据集。其他包可通过下载来进行安装。

>search( )	可以告诉你哪些包已加载并可使用。
>install.packages( )	将显示一个CRAN镜像站点的列表
>install.packages(“ggplot2”)	下载包ggplot2。
>update.packages( )	可以更新已经安装的包。
>library( )	可以显示库中有哪些包。
> .libPaths( )	能够显示库所在的位置
>library(ggplot2)	要在R会话中使用它，还需要使用此命令载入这个包
>help(package="package_name")	输出包的简短描述和包中的函数名称和数据集名称的列表

工作空间

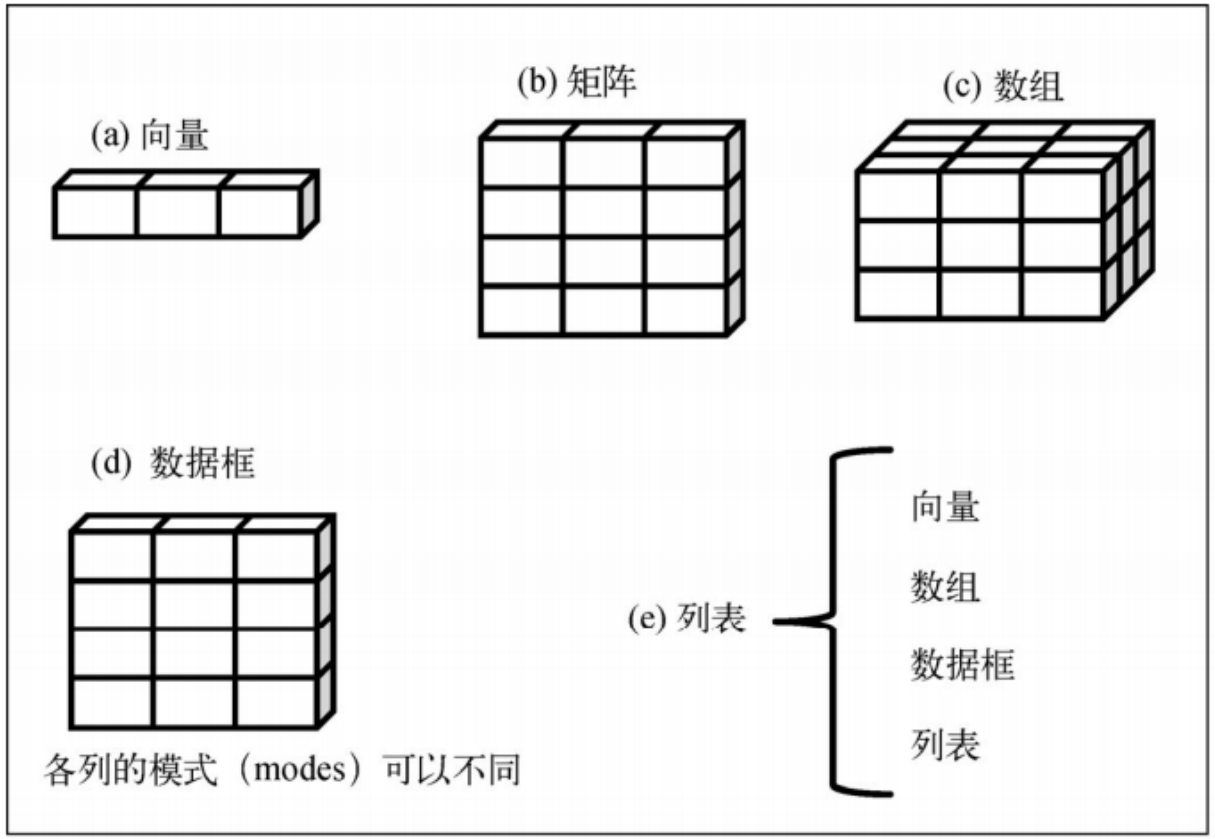
工作空间（workspace）是当前R的工作环境，它储存着所有用户定义的对象（向量、矩阵、函数、数据框、列表）。在一个R会话结束时，你可以将当前工作空间保存到一个镜像中，并在下次启动R时自动载入它。

表1 用于管理R工作空间的函数

函数	功能
getwd()	获取当前工作目录的位置
setwd（）	设置当前工作目录的位置
ls()	列出环境中的所有变量。
rm(x)	从环境中移除（删除）一个或多个对象
list（）	元素的集合，元素可以是不同类型的。
rm(list = ls())	从环境中删除所有变量。

R数据结构与数据集

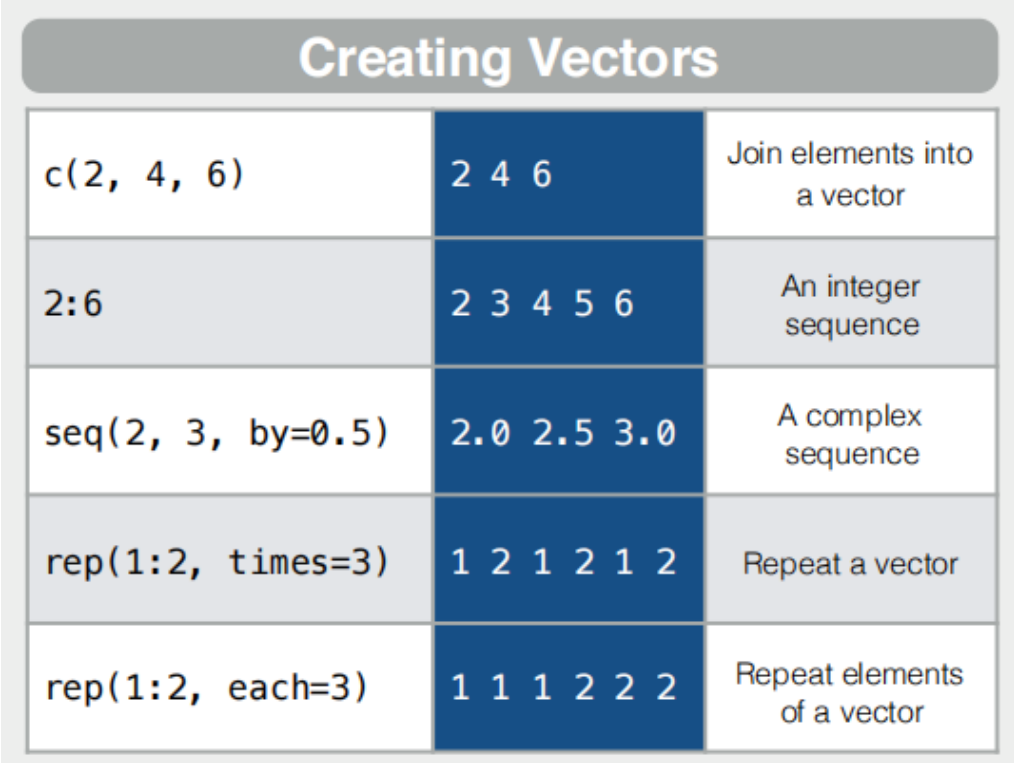
R拥有许多用于存储数据的对象类型，包括标量、向量、矩阵、数组、数据框和列表。它们在存储数据的类型、创建方式、结构复杂度，以及用于定位和访问其中个别元素的标记等方面均有所不同。



新建向量

a是数值型向量，b是字符型向量，而c是逻辑型向量。单个向量中的数据必须拥有相同的类型或模式（数值型、字符型或逻辑型）。同一向量中无法混杂不同模式的数据。

```
a <- c(1, 2, 5, 3, 6, -2, 4)
b <- c("one", "two", "three")
c <- c(TRUE, TRUE, TRUE, FALSE, TRUE, FALSE)
```



向量运算

By Position

`x[4]`	The fourth element.
`x[-4]`	All but the fourth.
`x[2:4]`	Elements two to four.
`x[-(2:4)]`	All elements except two to four.
`x[c(1, 5)]`	Elements one and five.

By Value

`x[x == 10]`	Elements which are equal to 10.
`x[x < 0]`	All elements less than zero.
`x[x %in% c(1, 2, 5)]`	Elements in the set 1, 2, 5.

Named Vectors

`x['apple']`	Element with name 'apple'.

```
> x<-seq(1,30,by=2)
> x
 [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29
> x[4]
[1] 7
> x[-4]
 [1]  1  3  5  9 11 13 15 17 19 21 23 25 27 29
> x[2:4]
[1] 3 5 7
> x[-(2:4)]
 [1]  1  9 11 13 15 17 19 21 23 25 27 29
> x[c(1,5)]
[1] 1 9
> x[x==11]
[1] 11
> x[x<20]
 [1]  1  3  5  7  9 11 13 15 17 19
> x[x %in% c(1,2,5)]
[1] 1 5
> x['apple']
[1] NA
```

向量运算

Vector Functions

sort(x)	**rev(x)**
Return x sorted.	Return x reversed.
table(x)	**unique(x)**
See counts of values.	See unique values.

sort（）：排序;

rev（）：倒序;

table（）：值出现的次数;

unique（）：删除重复的元素/行。

```
> a=1:20
> a
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
> rev(a)
 [1] 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1
> a=c(2,3,4,2,5,1,6,3,2,5,8,5,7,3)
> sort(a)
 [1] 1 2 2 2 3 3 3 4 5 5 5 6 7 8
> rev(sort(a))
 [1] 8 7 6 5 5 5 4 3 3 3 2 2 2 1
>
```

```
> a<-c(2,2,2,1,4,4,5)
> a
[1] 2 2 2 1 4 4 5
> table(a)
a
1 2 4 5
1 3 2 1
> unique(a)
[1] 2 1 4 5
```

创建矩阵

指定行和列的维数

```
mymatrix <- matrix(vector, nrow=number_of_rows, ncol=number_of_columns,
                   byrow=logical_value, dimnames=list(
                   char_vector_rownames, char_vector_colnames))
```

表明矩阵应当按行填（byrow=TRUE）还是按列填充（byrow=FALSE），默认情况下按列填充

❶ 创建一个5×4的矩阵

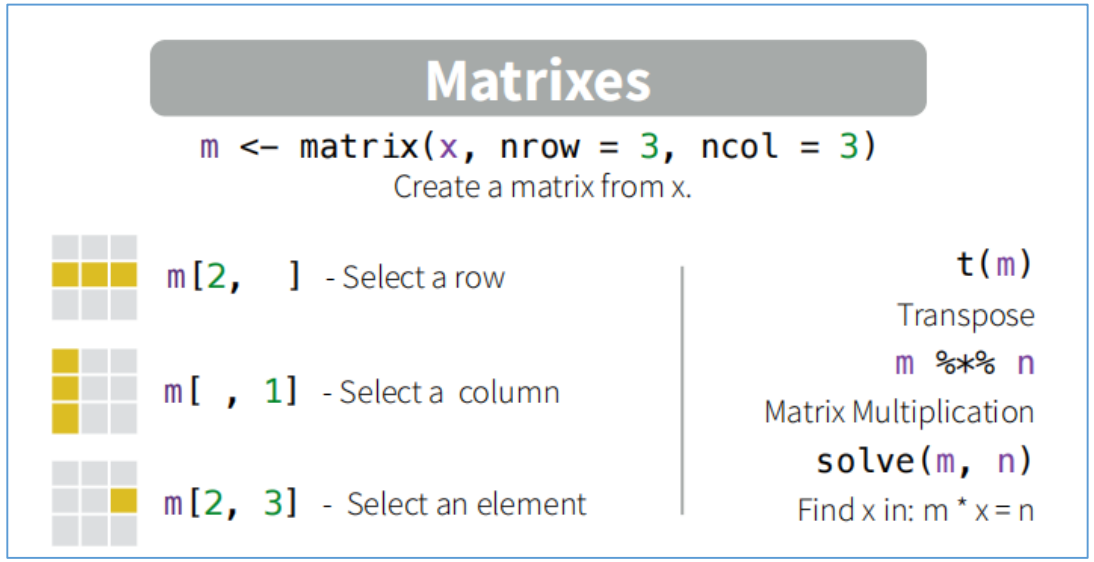
包含了可选的、以字符型向量表示的行名和列名

```
> y <- matrix(1:20, nrow=5, ncol=4)
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
> cells    <- c(1,26,24,68)
> rnames   <- c("R1", "R2")
> cnames   <- c("C1", "C2")
> mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=TRUE,
                     dimnames=list(rnames, cnames))
> mymatrix
   C1 C2
R1  1 26
R2 24 68
> mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=FALSE,
                     dimnames=list(rnames, cnames))
> mymatrix
  C1 C2
R1  1 24
R2 26 68
```

❷ 按行填充的2×2矩阵

❸ 按列填充的2×2矩阵

矩阵运算



函数t（）：矩阵转置

```
> a=matrix(1:12,nrow=3,ncol=4)
> a
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> t(a)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
```

矩阵加减

```
> a=b=matrix(1:12,nrow=3,ncol=4)
> a+b
     [,1] [,2] [,3] [,4]
[1,]    2    8   14   20
[2,]    4   10   16   22
[3,]    6   12   18   24
> a-b
     [,1] [,2] [,3] [,4]
[1,]    0    0    0    0
[2,]    0    0    0    0
[3,]    0    0    0    0
```

矩阵乘法

```
> a=matrix(1:12,nrow=3,ncol=4)
> b=matrix(1:12,nrow=4,ncol=3)
> a%*%b
     [,1] [,2] [,3]
[1,]   70  158  246
[2,]   80  184  288
[3,]   90  210  330
```

创建数组

```
> dim1 <- c("A1", "A2")
> dim2 <- c("B1", "B2", "B3")
> dim3 <- c("C1", "C2", "C3", "C4")
> z <- array(1:24, c(2, 3, 4), dimnames=list(dim1, dim2, dim3))
> z
```

```
, , C1

   B1 B2 B3
A1  1  3  5
A2  2  4  6

, , C2

   B1 B2 B3
A1  7  9 11
A2  8 10 12

, , C3

   B1 B2 B3
A1 13 15 17
A2 14 16 18

, , C4

   B1 B2 B3
A1 19 21 23
A2 20 22 24
```

创建数据框

```
df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))
```

A special case of a list where all elements are the same length.

x	y
1	a
2	b
3	c

List subsetting

`df$x` 

`df[[2]]` 

Understanding a data frame

`View(df)`	See the full data frame.
`head(df)`	See the first 6 rows.

Matrix subsetting

`df[ , 2]` 

`df[2, ]` 

`df[2, 2]` 

`nrow(df)`
Number of rows.

`ncol(df)`
Number of columns.

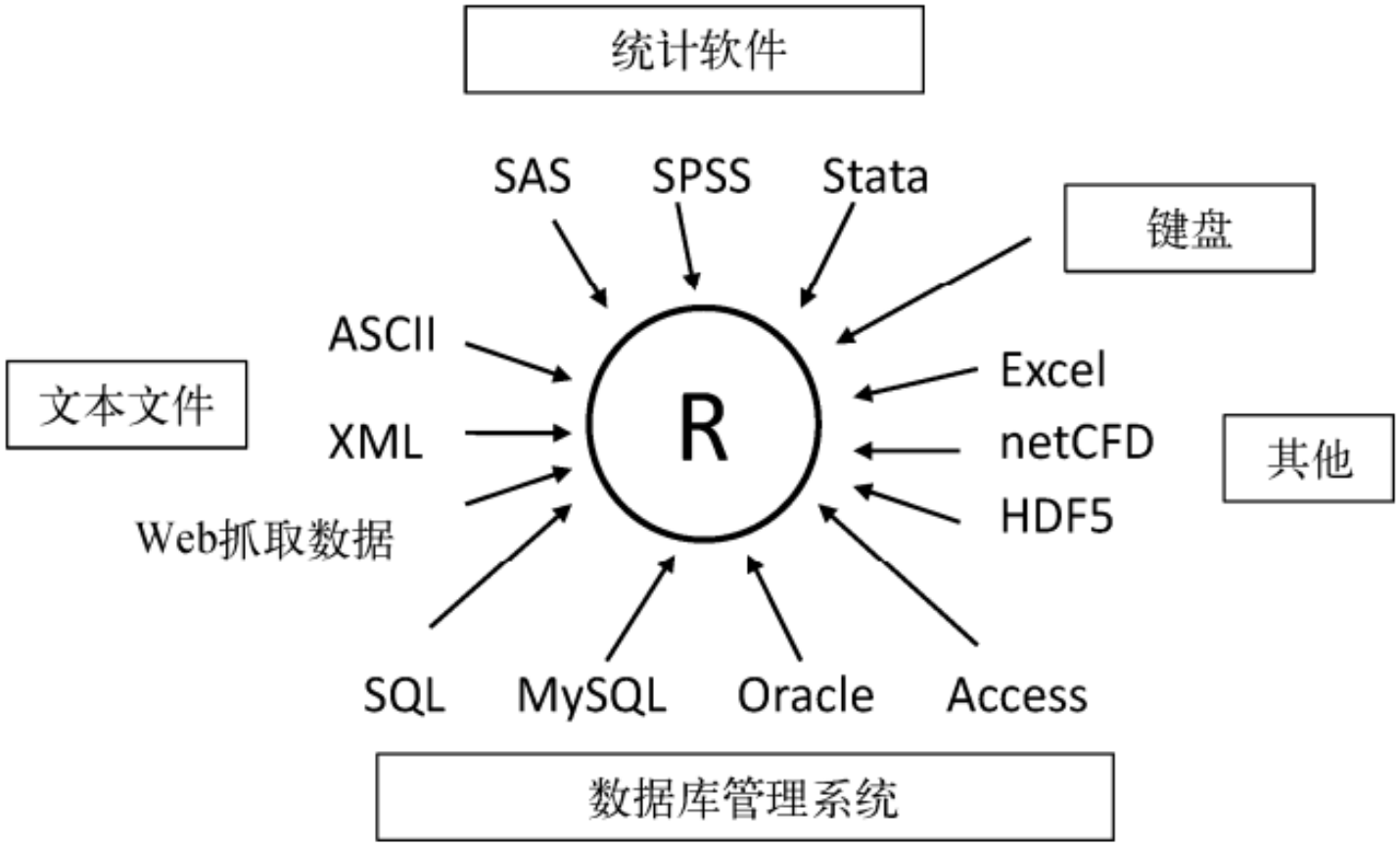
`dim(df)`
Number of columns and rows.

`cbind` - Bind columns.

`rbind` - Bind rows.

```
> df<-data.frame(x=1:3,y=c("a","b","c"))
> df$x
[1] 1 2 3
> df[[2]]
[1] "a" "b" "c"
> head(df)
  x y
1 1 a
2 2 b
3 3 c
> nrow(df)
[1] 3
> ncol(df)
[1] 2
> dim(df)
[1] 3 2
```

数据输入


统计软件
SAS
SPSS
Stata
键盘
ASCII
文本文件
XML
R
Excel
netCFD
其他
HDF5
Web抓取数据
SQL
MySQL
Oracle
Access
数据库管理系统

数据输入

（1）读取csv文件

- read.table() read.csv()

```
>df = read.table(file="test.txt", header=TRUE)
>df = read.csv(file="test.csv", header=TRUE)
```

- write.table() write.csv()

```
>write.table(df, file="test2.txt")
>write.csv(df, file="test2.csv")
>write.csv(df, file="D:/Data/test2.csv")
```

数据输入

（2）读取一个Excel文件的最好方式，就是在Excel中将其导出为一个逗号分隔文件（csv），并使用以上描述的方式将其导入R中。在Windows系统中，可安装 RODBC 包来访问Excel文件。电子表格的第一行应当包含变量/列的名称。

```
install.packages("RODBC")
library(RODBC)
channel <- odbcConnectExcel("myfile.xls")
mydataframe <- sqlFetch(channel, "mysheet")
odbcClose(channel)
```

处理对象函数

表2 处理数据对象的实用函数

函　数	用　途
length(*object*)	显示对象中元素/成分的数量
dim(*object*)	显示某个对象的维度
str(*object*)	显示某个对象的结构
class(*object*)	显示某个对象的类或类型
mode(*object*)	显示某个对象的模式
names(*object*)	显示某对象中各成分的名称
c(*object*, *object*,…)	将对象合并入一个向量
cbind(*object*, *object*, …)	按列合并对象
rbind(*object*, *object*, …)	按行合并对象
Object	输出某个对象
head(*object*)	列出某个对象的开始部分
tail(*object*)	列出某个对象的最后部分
ls()	显示当前的对象列表
rm(*object*, *object*, …)	删除一个或更多个对象。语句rm(list = ls())将删除当前工作环境中的几乎所有对象*
newobject <- edit(*object*)	编辑对象并另存为newobject
fix(*object*)	直接编辑对象

数据输出

生成制表符分隔的文本文件

```
>write.table(mydata, "c:/mydata.txt", sep="\t")
```

转换成Excel表格

```
>library(xlsReadWrite)
    >write.xls(mydata, "c:/mydata.xls")
```

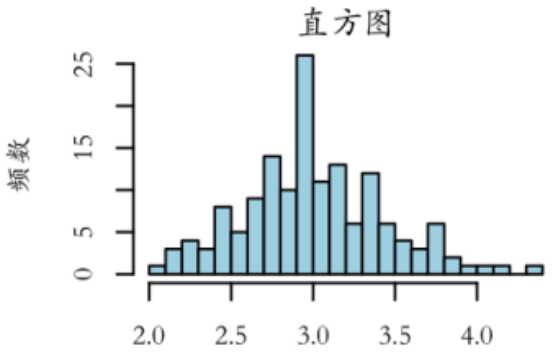
到SAS

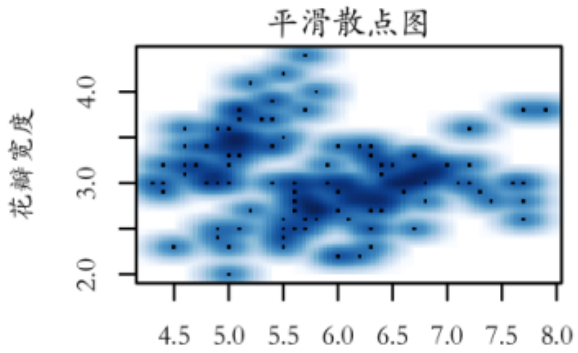
```
>library(foreign)
    >write.foreign(mydata,"c:/mydata.txt","c:/mydata.sas",   package="SAS")
```

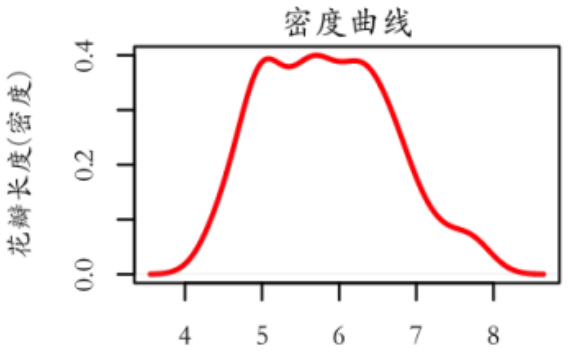
图的绘制

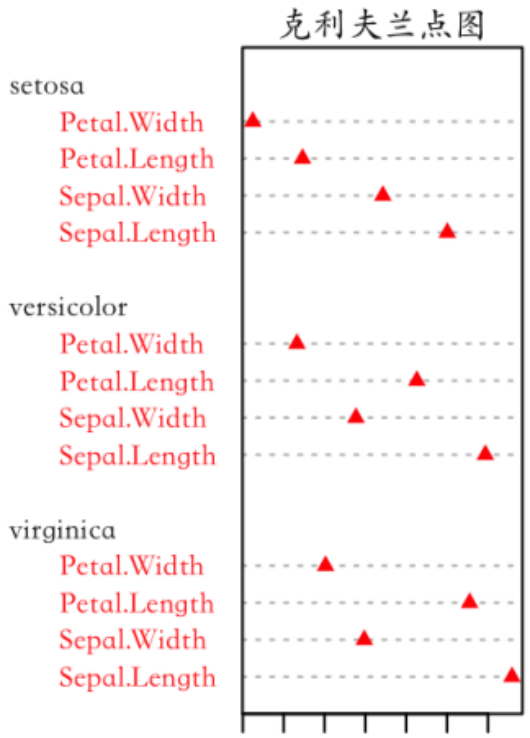
安装好R后，会自动加载一个数据可视化包graphics。

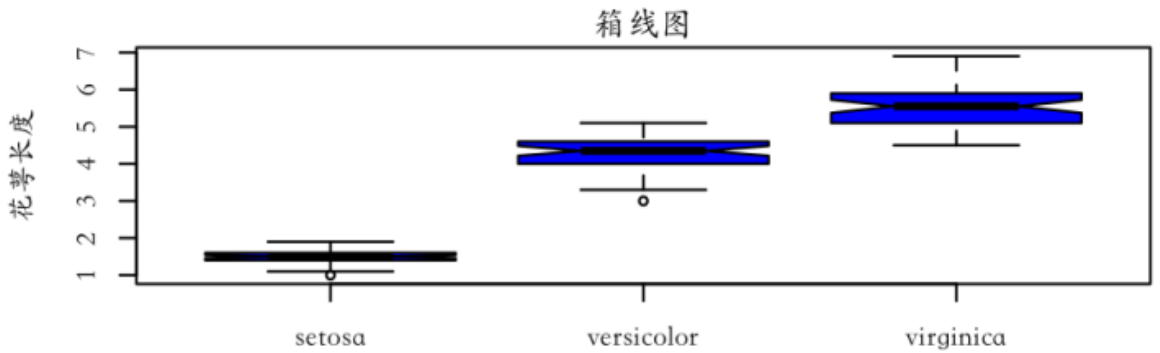
它含了R的基本绘图功能，可以绘制常用的直方图、线图、点图、饼图、密度曲线、三维透视图等

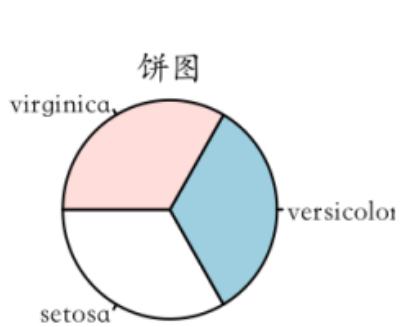


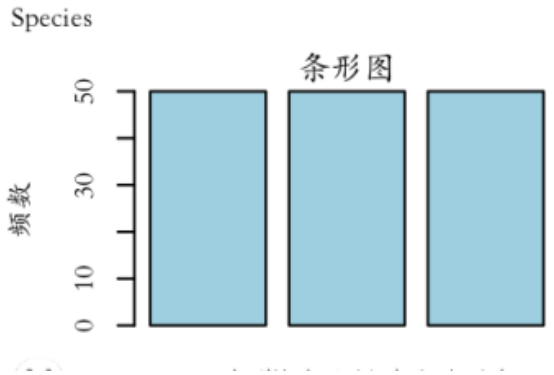












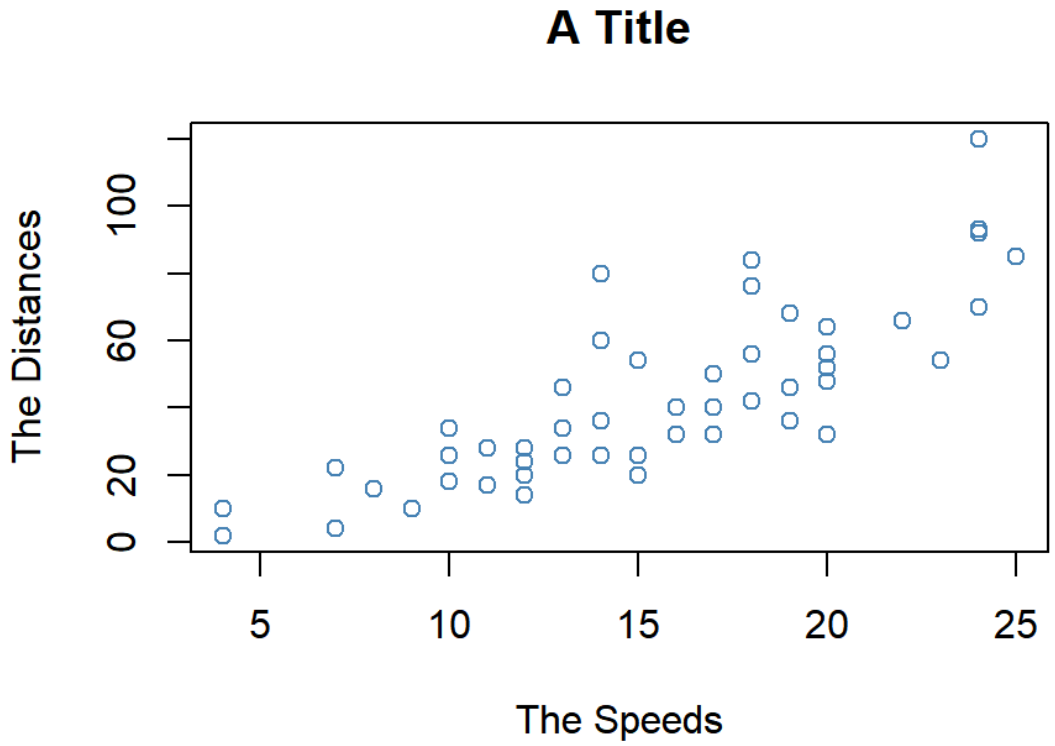
绘制基础

plot函数

plot(x=x轴数据,y=y轴数据,main="标题",sub="子标题"，type="线型",xlab="x轴名称"，ylab="y轴名称"，xlim = c(x轴范围，x轴范围),ylim = c(y轴范围,y轴范围))

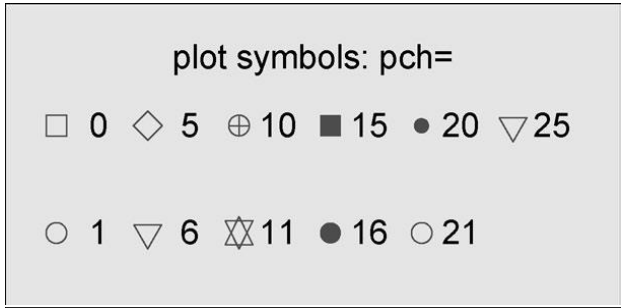
使用cars数据集

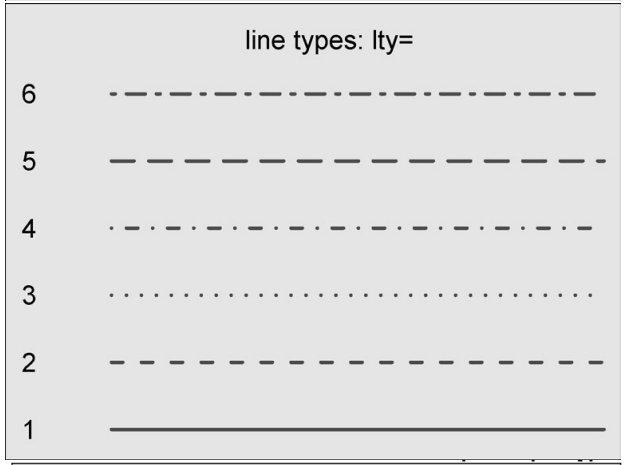
```
>head(cars)
>plot(cars$speed,cars$dist,main = "A Title",
xlab = "The Speeds", ylab = "The Distances",
col="steel blue")
```

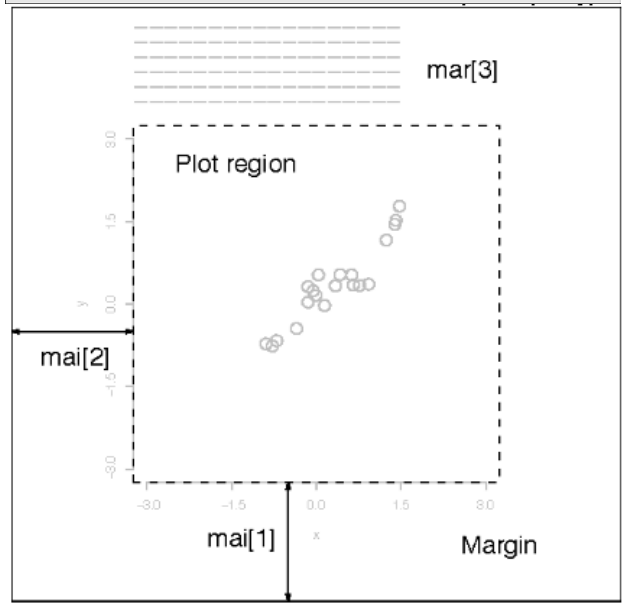


图形参数

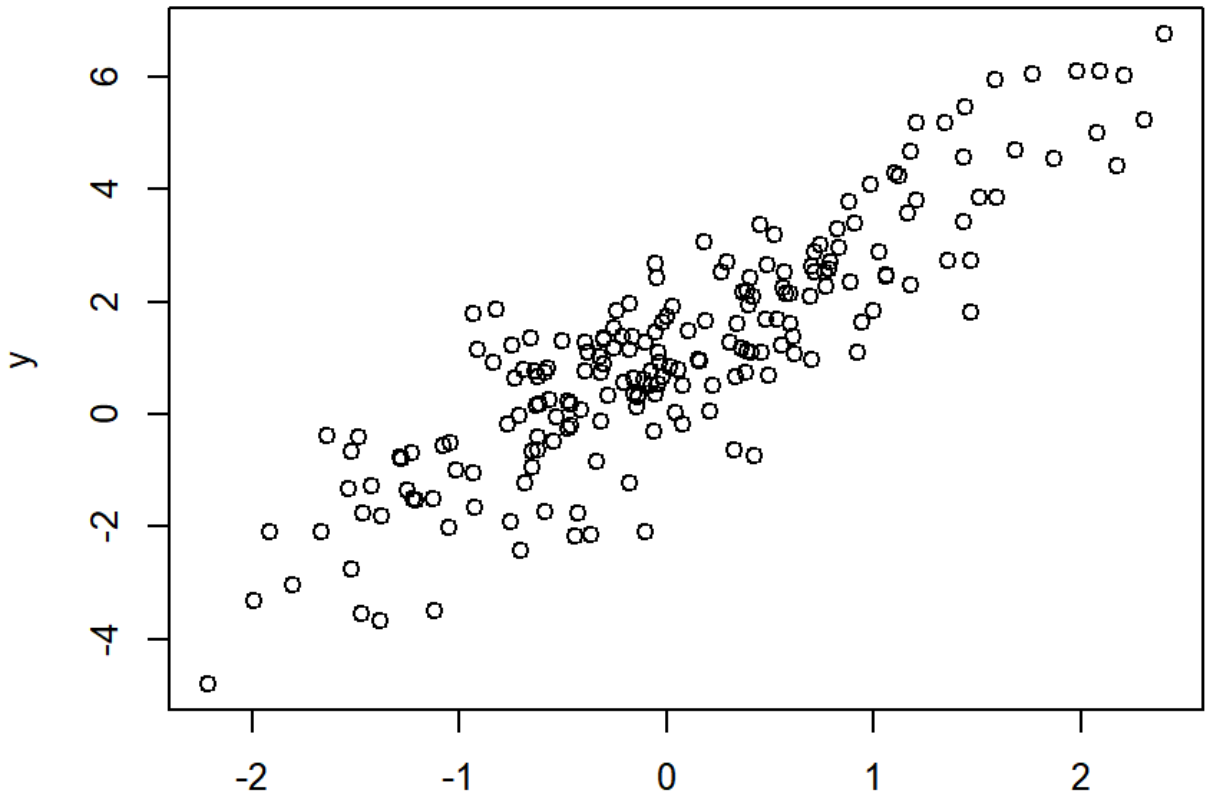
pch	指定绘制点时使用的符号
cex	是一个数值，表示绘图符号相对于默认大小的缩放倍数。
lty	指定线条类型
lwd	指定线条宽度。
col	默认的绘图颜色。
cex	表示相对于默认大小缩放倍数的数值。
font	整数。用于指定绘图使用的字体样式。
pin	以英寸表示的图形尺寸（宽和高）
mai	以数值向量表示的边界大小，
	顺序为“下、左、上、右”，单位为英寸





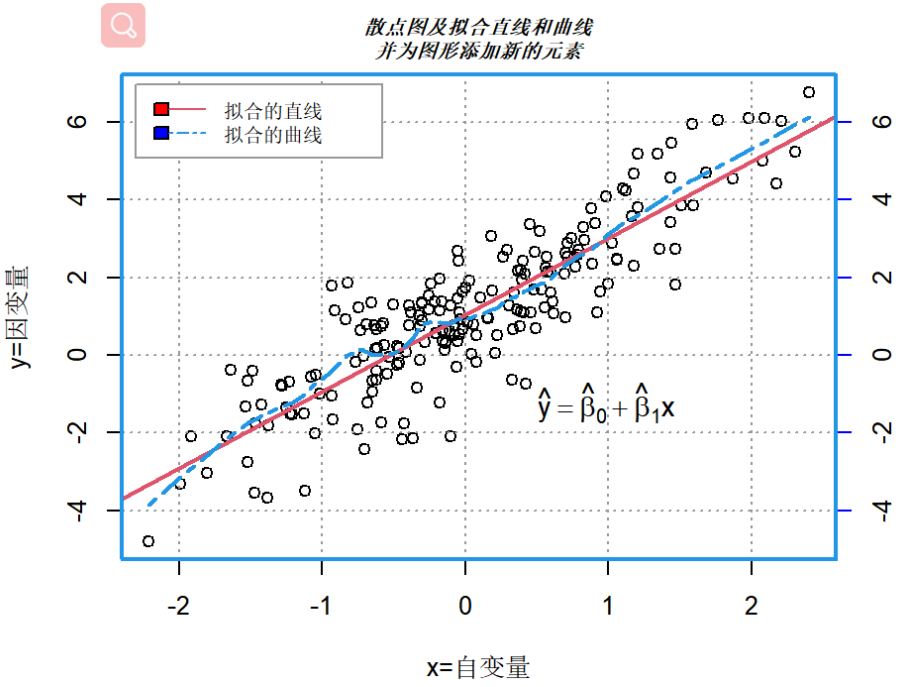


利用图形参数绘图




```
par(mfrow=c(1,1), mai=c(0.7, 0.7, 0.4, 0.4), cex=0.8)
set.seed(1)
x <- rnorm(200) #产生200个服从正态分布的随机数
y <- 1+2*x+rnorm(200)
d <- data.frame(x, y)
plot(x, y)   # 绘制散点图
```

利用图形参数绘图

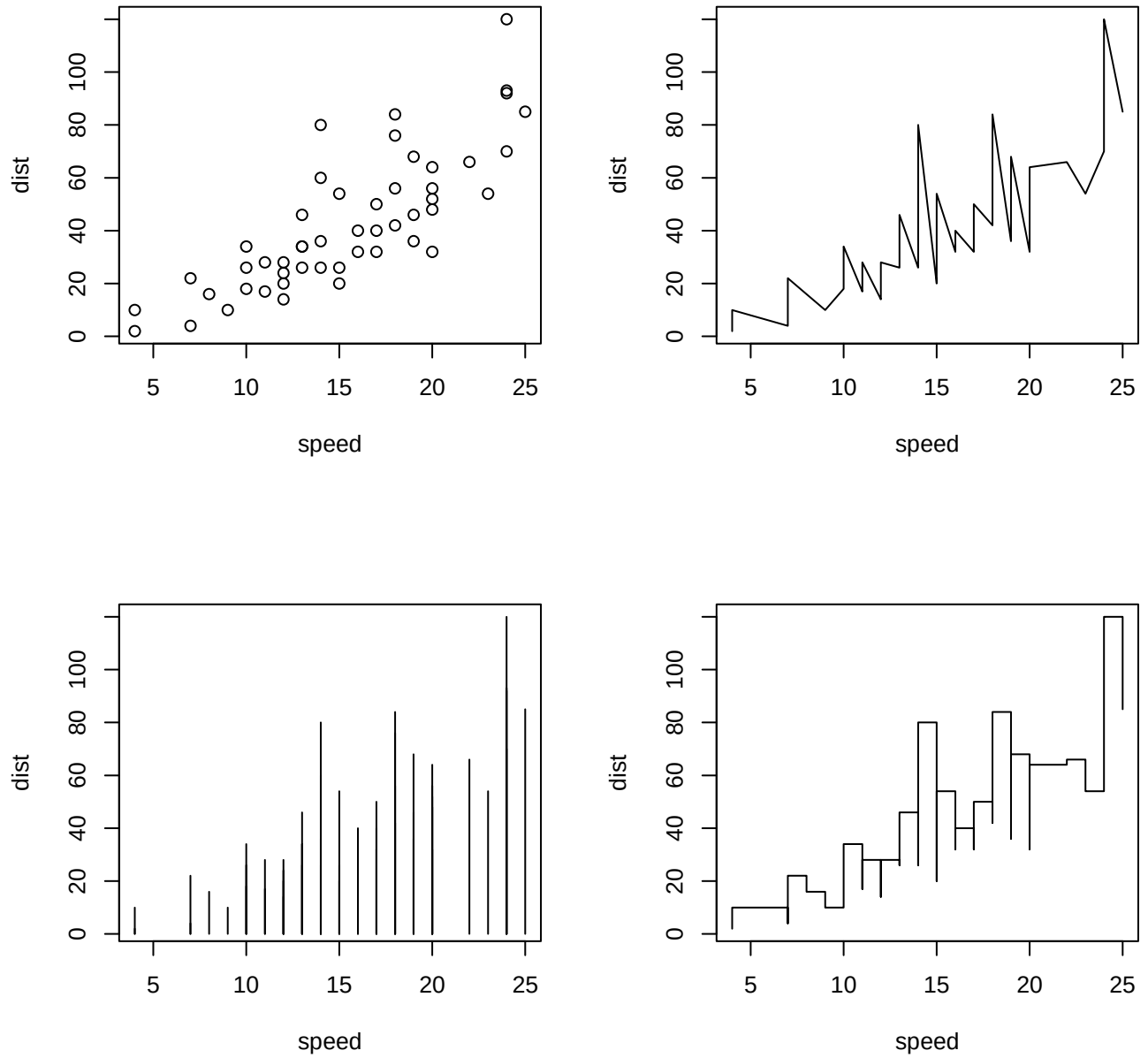



```
plot(x, y, xlab='x=自变量', ylab='y=因变量')  # 添加坐标轴标题
grid(col='grey60')                                  # 添加网格线
axis(side=4, col.ticks='blue', lty=1)               # 绘制坐标轴
abline(lm(y~x), lwd=2, col=2)                       # 添加回归直线
lines(lowess(y~x, f=1/6), col=4, lwd=2, lty=6) # 添加拟合曲线
mtext(expression(hat(y)==hat(beta)[0]+hat(beta)[1]*x), cex=0.9,
side=1,    line=-5.3, adj=0.72)                     # 添加注释表达式
legend('topleft', legend=c('拟合的直线', '拟合的曲线'), lty=c(1, 6),
      col=c(2, 4), cex=0.8, fill=c('red', 'blue'), box.col='grey60',
      ncol=1, inset=0.02)                           # 添加图例
title('散点图及拟合直线和曲线\n并为图形添加新的元素',
     cex.main=0.8, font.main=4)     # 添加标题并换行，使用斜体字
box(col=4, lwd=2)                                   # 添加边框
```

其他绘图函数

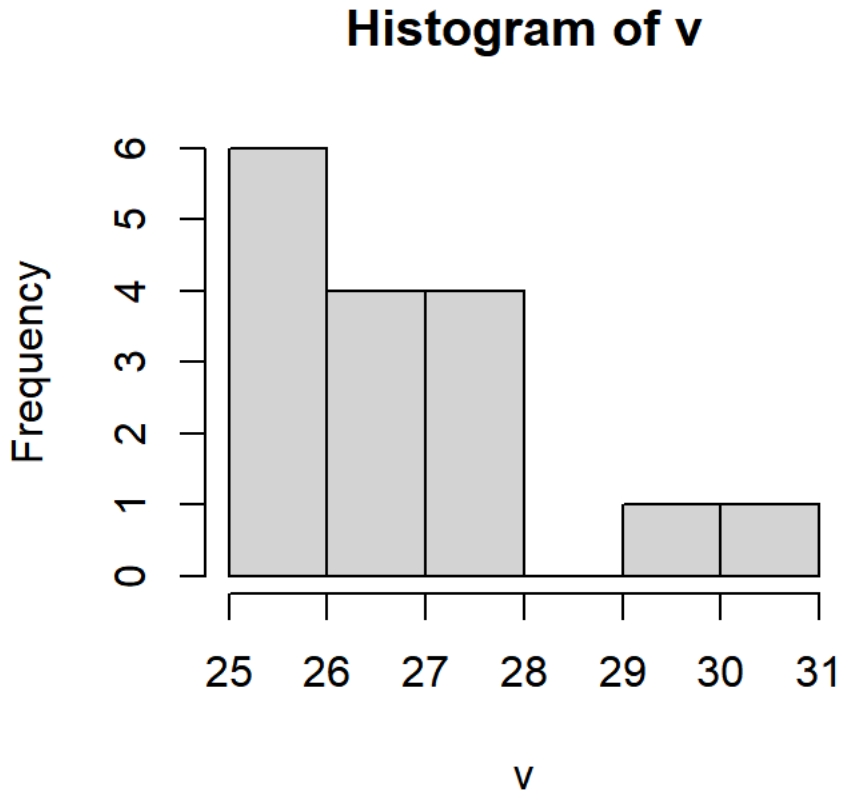
用函数par ()可以组合多幅图形

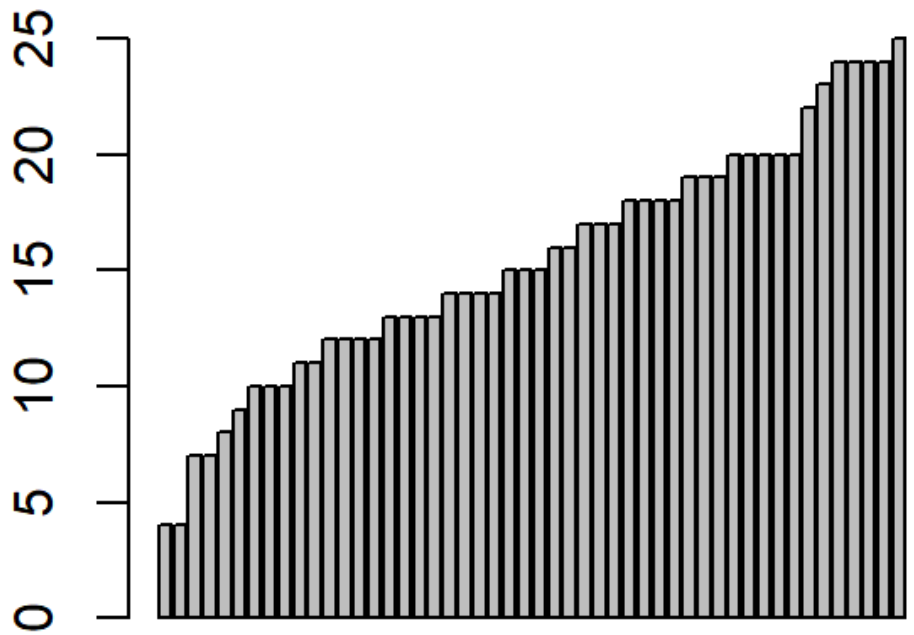
```
>head(cars)
>plot(cars)
>par(mfrow=c(2, 2))
>plot(cars, type="p")
>plot(cars, type="l")
>plot(cars, type="h")
>plot(cars, type="s")
```



直方图与条形图

```
> v <- c(27, 27, 27, 28, 27, 25, 25, 28, 26, 28, 26, 28, 31, 30, 26, 26)
>hist(v) #频率直方图
>barplot(v)#条形图
```





描述性统计量

表3 描述性统计量

函数	描述
mean( )	均值
weighted.mean( )	加权平均数
median( )	中位数
var( )	方差
sd( )	标准差
sum( )	求和
sort( )	排序
mix( )	最小值
max( )	最大值

```
> J<-c(7,8,7,9,5,7,9,10,7,11)
> Y<-c(7,8,9,10,11,5,7,9,8,6)
> mean(J)
[1] 8
> mean(Y)
[1] 8
> var(J)
[1] 3.111111
> var(Y)
[1] 3.333333
> sd(J)
[1] 1.763834
> sd(Y)
[1] 1.825742
```

summary( )函数

summary( )函数： 常用来展示详细结果，可以提供最小值、最大值、四分位数和数值型变量的均值，以及因子向量和逻辑型向量的频数统计等。

```
> J<-c(7,8,7,9,5,7,9,10,7,11)
> Y<-c(7,8,9,10,11,5,7,9,8,6)
> summary(J)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    5.0     7.0     7.5     8.0     9.0    11.0
> summary(Y)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      5       7       8       8       9      11
```

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

逻辑行向量频数统计