Introduction to Machine Learning and Data Mining

Introduction to R

Agenda

- R语言概述
- R语言数据类型
- R语言数据管理
- R语言绘图
- R语言高级数据管理

R语言概述

R是一种程序设计语言，主要用于统计计算和绘图。它是一种开源的数据分析解决方案，由一个庞大且活跃的全球性研究型社区维护。

R有很多优秀的特性：

- 免费
- 全面的统计研究平台，提供各种各样的数据分析技术。
- 顶尖水平的制图功能。
- 可进行交互式数据分析和探索。
- …

R语言概述

R 可以在http://cran.r-project.org上下载。

R常用的开发工具：Rstudio

setwd("d:\\testR")
setwd("d:/testR")

工作空间：当前R的工作环境

函　数	功　能
getwd()	显示当前的工作目录
setwd("*mydirectory*")	修改当前的工作目录为*mydirectory*
ls()	列出当前工作空间中的对象
rm(*objectlist*)	移除（删除）一个或多个对象
help(options)	显示可用选项的说明
options()	显示或设置当前选项
history(#)	显示最近使用过的#个命令（默认值为25）
savehistory("*myfile*")	保存命令历史到文件*myfile*中（默认值为.Rhistory）
loadhistory("*myfile*")	载入一个命令历史文件（默认值为.Rhistory）
save.image("*myfile*")	保存工作空间到文件*myfile*中（默认值为.RData）
save(*objectlist*, file="*myfile*")	保存指定对象到一个文件中
load("*myfile*")	读取一个工作空间到当前会话中（默认值为.RData）
q()	退出R。将会询问你是否保存工作空间

R语言概述

R 的帮助

函　数	功　能
help.start()	打开帮助文档首页
help("*foo*")或?*foo*	查看函数*foo*的帮助（引号可以省略）
help.search("*foo*")或??*foo*	以*foo*为关键词搜索本地帮助文档
example("*foo*")	函数*foo*的使用示例（引号可以省略）
RSiteSearch("*foo*")	以*foo*为关键词搜索在线文档和邮件列表存档
apropos("*foo*", mode="function")	列出名称中含有*foo*的所有可用函数
data()	列出当前已加载包中所含的所有可用示例数据集
vignette()	列出当前已安装包中所有可用的vignette文档
vignette("*foo*")	为主题*foo*显示指定的vignette文档

R语言概述

R 提供了大量开箱即用的功能，可以通过可选模块的下载和安装来实现。

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。

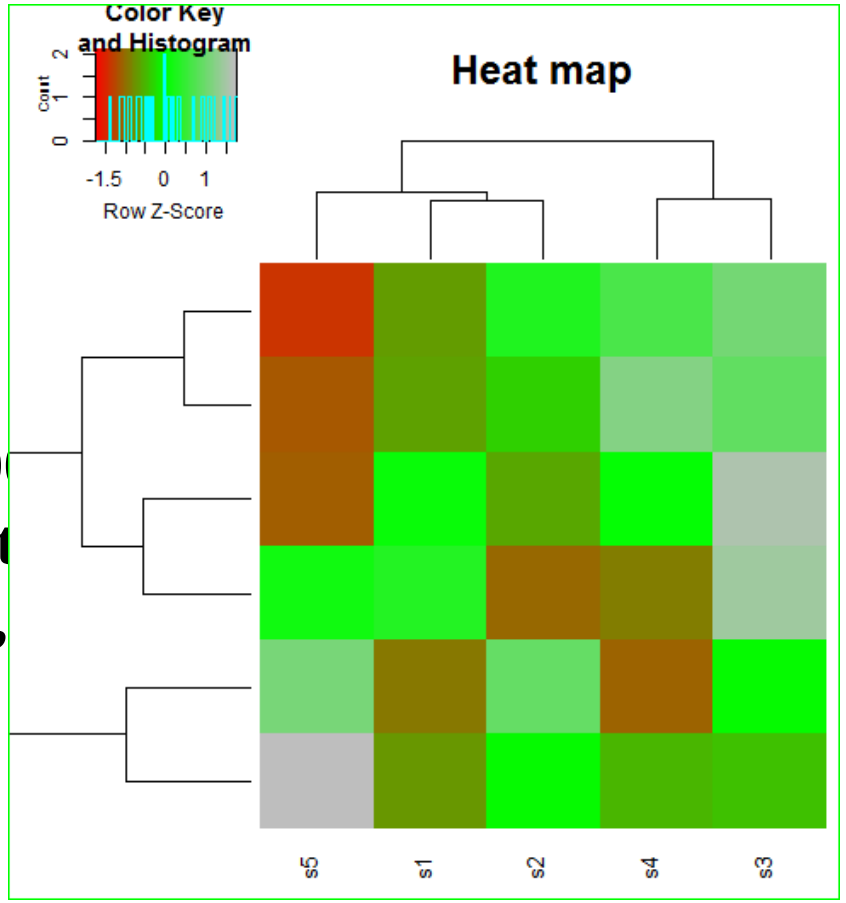
- 大量的包可以从http://cran.r-project.org/web/packages下载。
- 存储包的目录称为库library。
- 函数.libPaths()能够显示库所在的位置，函数library()则可以显示库中有哪些包。
- R自带了一系列默认包(base, datasets, utils, grDevices, graphics, stats, methods)
- 其他包则需要下载安装然后载入才能使用，例如：install.packages(“gclus”), library(“gclus”)

R语言概述

Hello例子

```
install.packages("gplots")
library(gplots)
test <- read.table("test.txt")
data <- as.matrix(test)
heatmap.2(data,col=colorpanel(10
gh="gray"),dendrogram="both",t
symkey=F, scale="row",cexCol=1,
```

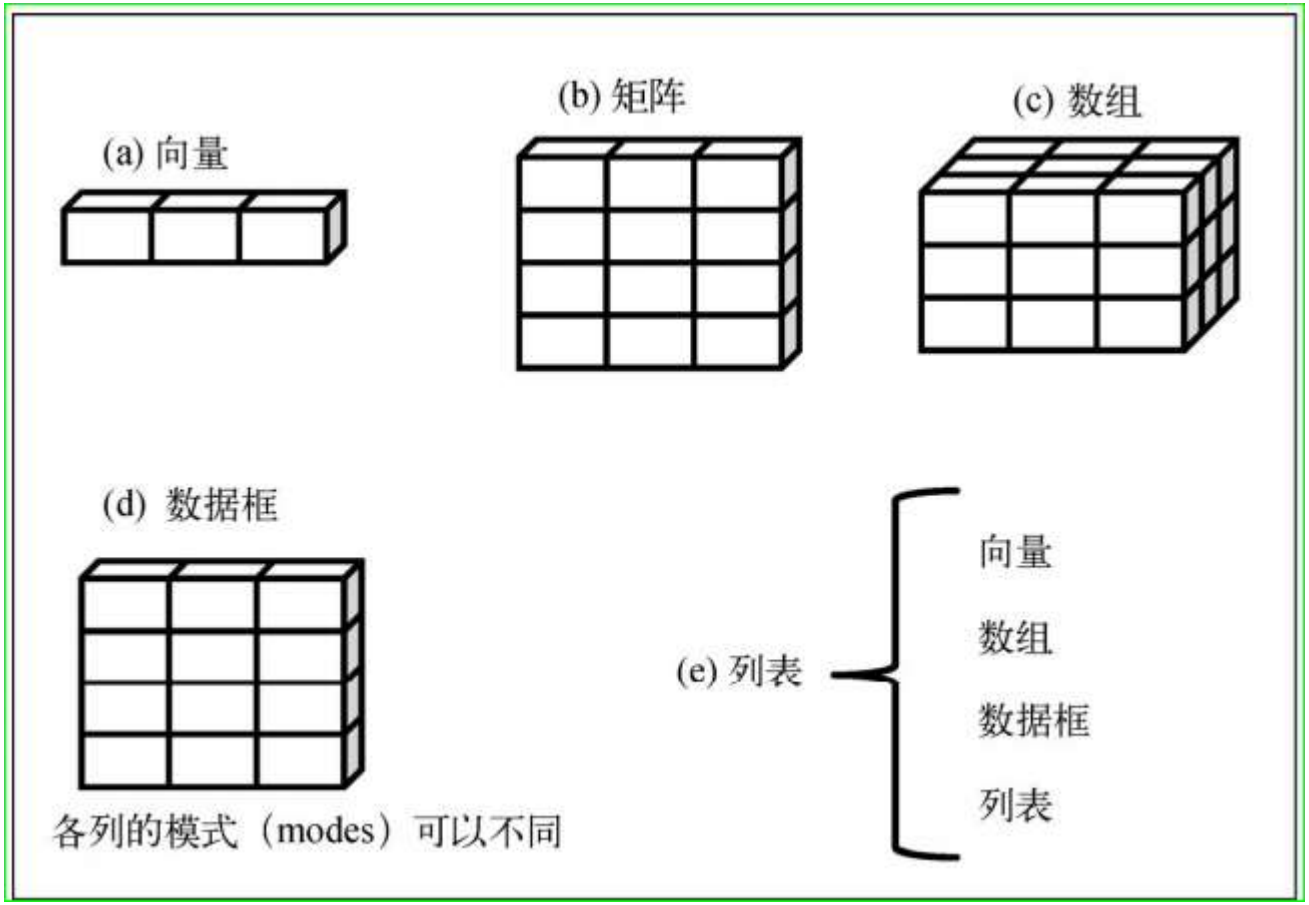
s1	s2	s3	s4	s5	
A	10	20	14	13	46
B	20	30	70	80	4
C	13	64	37	9	68
D	44	23	90	43	8
E	23	54	78	65	1
F	44	19	64	22	41



source(“test.R")
sink(“myoutput”)

R语言数据类型

R中的常用数据类型



R语言数据类型

向量：是用于存储数值型、字符型或逻辑型数据的一维数组。执行组合功能的函数c()可用来创建向量。

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

```
a[c(2, 4)]
```

```
a[2:6]
```

R语言数据类型

矩阵是一个二维数组，只是每个元素都拥有相同的模式（数值型、字符型或逻辑型）。可通过函数matrix

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

```
  > y <- matrix(1:20, nrow=5, ncol=4)
> y
     [,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
```

❶ 创建一个5×4的矩阵

❷ 按行填充的2×2矩阵

❸ 按列填充的2×2矩阵

R语言数据类型

数组（array）与矩阵类似，但是维度可以大于2。数组可通过array函数创建，

```
myarray <- array(vector, dimensions, dimnames)
```

```
> 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))
```

R语言数据类型

由于不同的列可以包含不同模式（数值型、字符型等）的数据，数据框的概念较矩阵来说更为一般。数据框将是R中最常处理的数据结构。数据框可通过函数data.frame()创建。

```
mydata <- data.frame(col1, col2, col3,…)
```

```
> patientID <- c(1, 2, 3, 4)
> age <- c(25, 34, 28, 52)
> diabetes <- c("Type1", "Type2", "Type1", "Type1")
> status <- c("Poor", "Improved", "Excellent", "Poor")
> patientdata <- data.frame(patientID, age, diabetes, status)
> patientdata
  patientID age diabetes    status
1         1  25    Type1      Poor
2         2  34    Type2  Improved
3         3  28    Type1 Excellent
4         4  52    Type1      Poor
```

R语言数据类型

类别（名义型）变量和有序类别（有序型）变量在R中称为因子（factor）。函数factor()以一个整数向量的形式存储类别值，整数的取值范围是[1... k]（其中k 是名义型变量中唯一值的个数），同时一个由字符串（原始值）组成的内部向量将映射到这些整数上。

```
diabetes <- c("Type1", "Type2", "Type1", "Type1")
```

语句**diabetes <- factor(diabetes)**将此向量存储为**(1, 2, 1, 1)**，并在内部将其关联为**1=Type1**和**2=Type2**（具体赋值根据字母顺序而定）。

```
status <- c("Poor", "Improved", "Excellent", "Poor")
```

语句**status <- factor(status, ordered=TRUE)**会将向量编码为**(3, 2, 1, 3)**，并在内部将这
些值关联为**1=Excellent**、**2=Improved**以及**3=Poor**。

R语言数据类型

因子

```
> patientID <- c(1, 2, 3, 4)
> age <- c(25, 34, 28, 52)
> diabetes <- c("Type1", "Type2", "Type1", "Type1")
> status <- c("Poor", "Improved", "Excellent", "Poor")
> diabetes <- factor(diabetes)
> status <- factor(status, order=TRUE)
> patientdata <- data.frame(patientID, age, diabetes, status)
> str(patientdata)
'data.frame':   4 obs. of  4 variables:
 $ patientID: num  1 2 3 4
 $ age      : num  25 34 28 52
 $ diabetes : Factor w/ 2 levels "Type1","Type2": 1 2 1 1
 $ status   : Ord.factor w/ 3 levels "Excellent"<"Improved"<..: 3 2 1 3
> summary(patientdata)
   patientID          age          diabetes       status
 Min.   :1.00   Min.   :25.00   Type1:3   Excellent:1
 1st Qu.:1.75   1st Qu.:27.25   Type2:1   Improved :1
 Median :2.50   Median :31.00             Poor     :2
 Mean   :2.50   Mean   :34.75
 3rd Qu.:3.25   3rd Qu.:38.50
 Max.   :4.00   Max.   :52.00
```

❶ 以向量形式输入数据

❷ 显示对象的结构

❸ 显示对象的统计概要

R语言数据类型

列表（list）是R的数据类型中最为复杂的一种。一般来说，列表就是一些对象（或成分，component）的有序集合。列表允许你整合若干（可能无关的）对象到单个对象名下。例如，某个列表中可能是若干向量、矩阵、数据框，甚至其他列表的组合。可以使用函数list()创建列表。

```
mylist <- list(object1, object2, …)
```

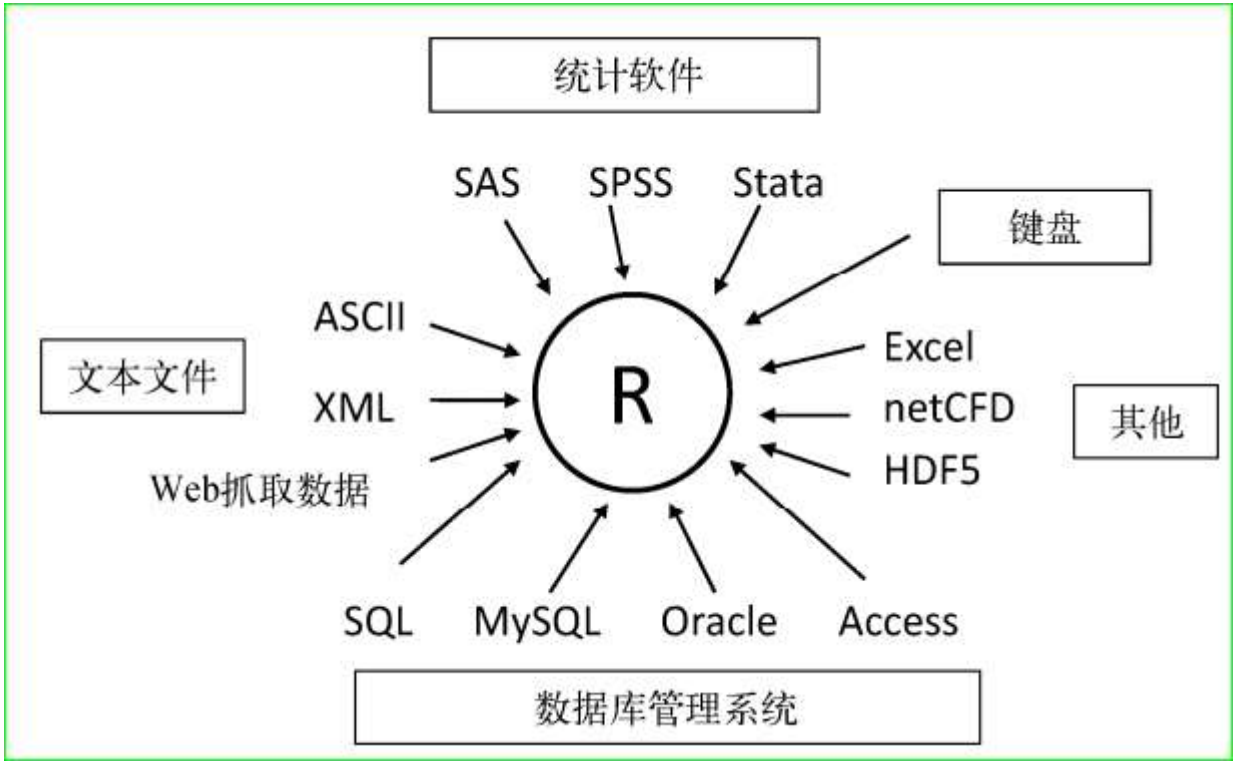
R语言数据类型

列表

```
> g <- "My First List"
> h <- c(25, 26, 18, 39)
> j <- matrix(1:10, nrow=5)
> k <- c("one", "two", "three")
> mylist <- list(title=g, ages=h, j, k)        ⟵ 创建列表
> mylist                                       ⟵ 输出整个列表
$title
[1] "My First List"

$ages
[1] 25 26 18 39

[[3]]
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10

[[4]]
[1] "one"   "two"   "three"

> mylist[[2]]                                  ⟵ 输出第二个成分
[1] 25 26 18 39
> mylist[["ages"]]
[1] 25 26 18 39
```

R语言数据类型

数据输入



R语言数据类型

输入

```
mydataframe <- read.table(file, header=logical_value,
  sep="delimiter", row.names="name")
```

R语言数据类型

一些实用函数

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

R语言数据类型

一些实用函数

函　数	用　途
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

R语言数据管理

数据

这名经理在做出人事决策之前会询问我的意见。

1	2	3	4	5
非常不同意	不同意	既不同意也不反对	同意	非常同意

经理人	日　期	国　籍	性　别	年　龄	q1	q2	q3	q4	q5
1	10/24/08	US	M	32	5	4	5	5	5
2	10/28/08	US	F	45	3	5	2	5	5
3	10/01/08	UK	F	25	3	5	5	5	2
4	10/12/08	UK	M	39	3	3	4		
5	05/01/09	UK	F	99	2	2	1	2	1

R语言数据管理

```
manager <- c(1, 2, 3, 4, 5)
date <- c("10/24/08", "10/28/08", "10/1/08", "10/12/08", "5/1/09")
country <- c("US", "US", "UK", "UK", "UK")
gender <- c("M", "F", "F", "M", "F")
age <- c(32, 45, 25, 39, 99)
q1 <- c(5, 3, 3, 3, 2)
q2 <- c(4, 5, 5, 3, 2)
q3 <- c(5, 2, 5, 4, 1)
q4 <- c(5, 5, 5, NA, 2)
q5 <- c(5, 5, 2, NA, 1)
leadership <- data.frame(manager, date, country, gender, age,
                         q1, q2, q3, q4, q5, stringsAsFactors=FALSE)
```

R语言数据管理

创建新变量

```
mydata<-data.frame(x1 = c(2, 2, 6, 4),
                   x2 = c(3, 4, 2, 8))

mydata$sumx  <-  mydata$x1 + mydata$x2
mydata$meanx <- (mydata$x1 + mydata$x2)/2

attach(mydata)
mydata$sumx  <-  x1 + x2
mydata$meanx <- (x1 + x2)/2
detach(mydata)

mydata <- transform(mydata,
                    sumx  =  x1 + x2,
                    meanx = (x1 + x2)/2)
```

R语言数据管理

变量的重编码

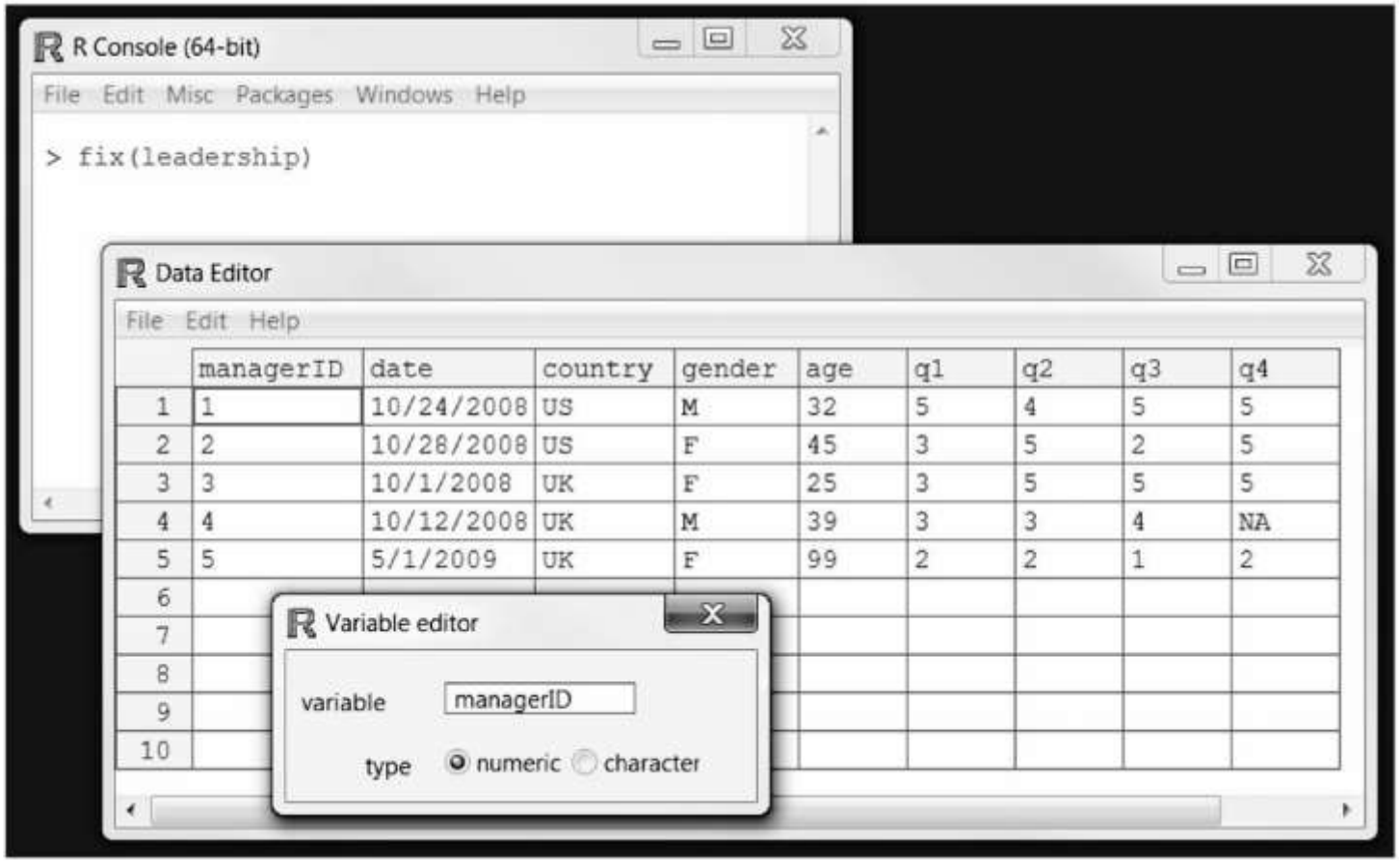
```
leadership$age[leadership$age  == 99]      <- NA

leadership$agecat[leadership$age  > 75]   <- "Elder"
leadership$agecat[leadership$age >= 55 &
                  leadership$age <= 75]   <- "Middle Aged"
leadership$agecat[leadership$age  < 55]   <- "Young"
```

```
leadership <- within(leadership,{
                     agecat <- NA
                     agecat[age > 75]                 <- "Elder"
                     agecat[age >= 55 & age <= 75] <- "Middle Aged"
                     agecat[age < 55]                 <- "Young" })
```

R语言数据管理

变量重命名

```
fix(leadership)
```



R语言数据管理

变量重命名

```
rename(dataframe, c(oldname="newname", oldname="newname",…))
```

```
library(reshape)
leadership <- rename(leadership,
                     c(manager="managerID", date="testDate")
)
```

R语言数据管理

变量重命名

```
names(leadership)[2] <- "testDate"
```

```
> names(leadership)
 [1] "manager" "date"    "country" "gender"  "age"     "q1"      "q2"
 [8] "q3"      "q4"      "q5"
> names(leadership)[2] <- "testDate"
> leadership
  manager testDate country gender age q1 q2 q3 q4 q5
1       1 10/24/08      US      M  32  5  4  5  5  5
2       2 10/28/08      US      F  45  3  5  2  5  5
3       3  10/1/08      UK      F  25  3  5  5  5  2
4       4 10/12/08      UK      M  39  3  3  4 NA NA
5       5   5/1/09      UK      F  99  2  2  1  2  1
```

```
names(leadership)[6:10] <- c("item1", "item2", "item3", "item4", "item5")
```

R语言数据管理

缺失值

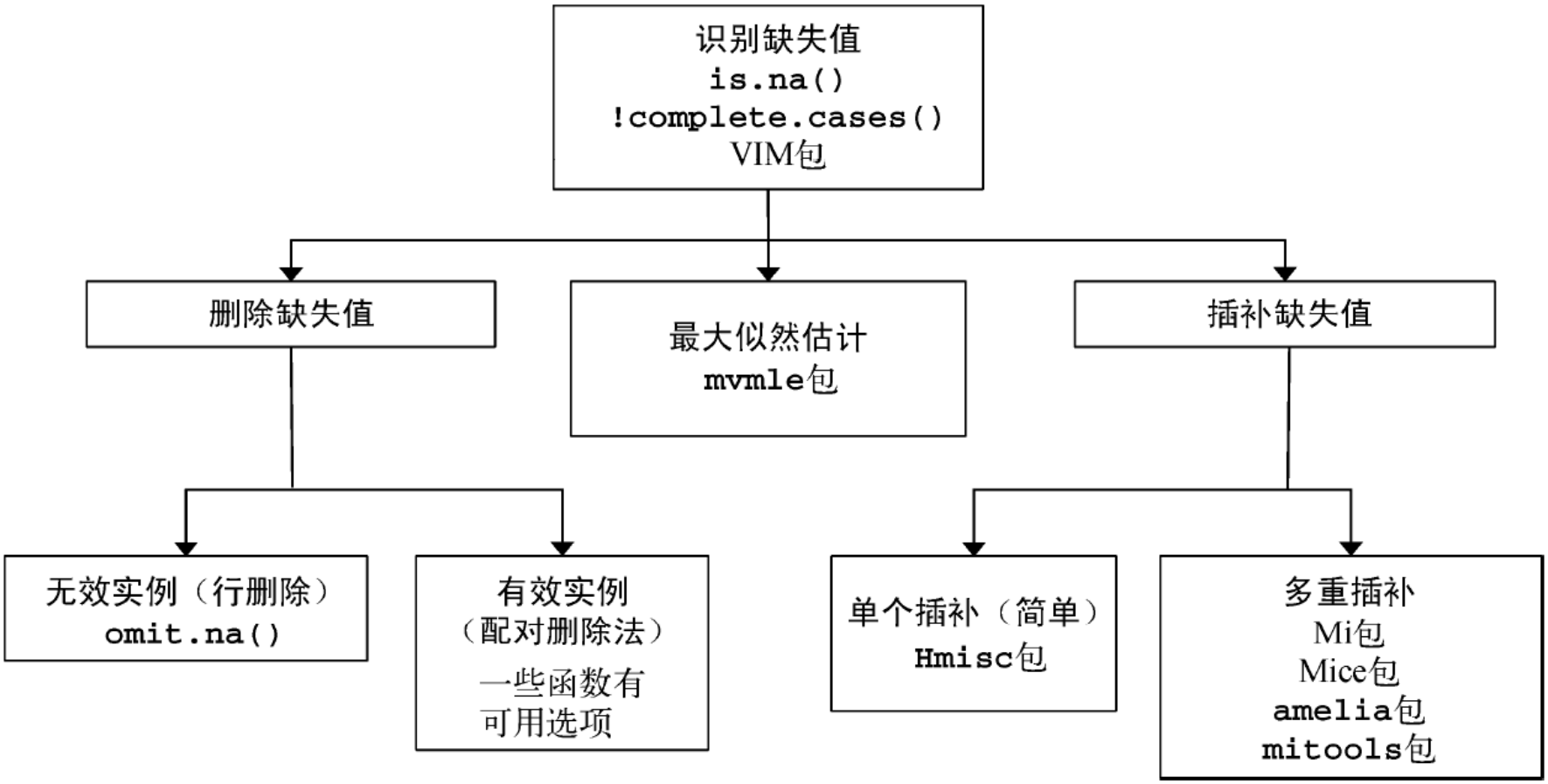
```
y <- c(1, 2, 3, NA)

is.na(y)

leadership$age[leadership$age == 99] <- NA

x <- c(1, 2, NA, 3)
y <- x[1] + x[2] + x[3] + x[4]
z <- sum(x)

x <- c(1, 2, NA, 3)
y <- sum(x, na.rm=TRUE)
```

R语言数据管理

缺失值

```
> leadership
  manager     date country gender age q1 q2 q3 q4 q5
1       1 10/24/08      US      M  32  5  4  5  5  5
2       2 10/28/08      US      F  40  3  5  2  5  5
3       3 10/01/08      UK      F  25  3  5  5  5  2
4       4 10/12/08      UK      M  39  3  3  4 NA NA
5       5 05/01/09      UK      F  99  2  2  1  2  1

> newdata <- na.omit(leadership)
> newdata
  manager     date country gender age q1 q2 q3 q4 q5
1       1 10/24/08      US      M  32  5  4  5  5  5
2       2 10/28/08      US      F  40  3  5  2  5  5
3       3 10/01/08      UK      F  25  3  5  5  5  2
5       5 05/01/09      UK      F  99  2  2  1  2  1
```

缺失值


识别缺失值
is.na()
!complete.cases()
VIM包
删除缺失值
最大似然估计
mvmle包
插补缺失值
无效实例（行删除）
omit.na()
有效实例
（配对删除法）
一些函数有
可用选项
单个插补（简单）
Hmisc包
多重插补
Mi包
Mice包
amelia包
mitools包

R语言数据管理

缺失值

```
# 加载数据集
data(sleep, package="VIM")

# 列出没有缺失值的行
sleep[complete.cases(sleep),]

# 列出有一个或多个缺失值的行
sleep[!complete.cases(sleep),]
```

```
> sum(is.na(sleep$Dream))
[1] 12
> mean(is.na(sleep$Dream))
[1] 0.19
> mean(!complete.cases(sleep))
[1] 0.32
```

R语言数据管理

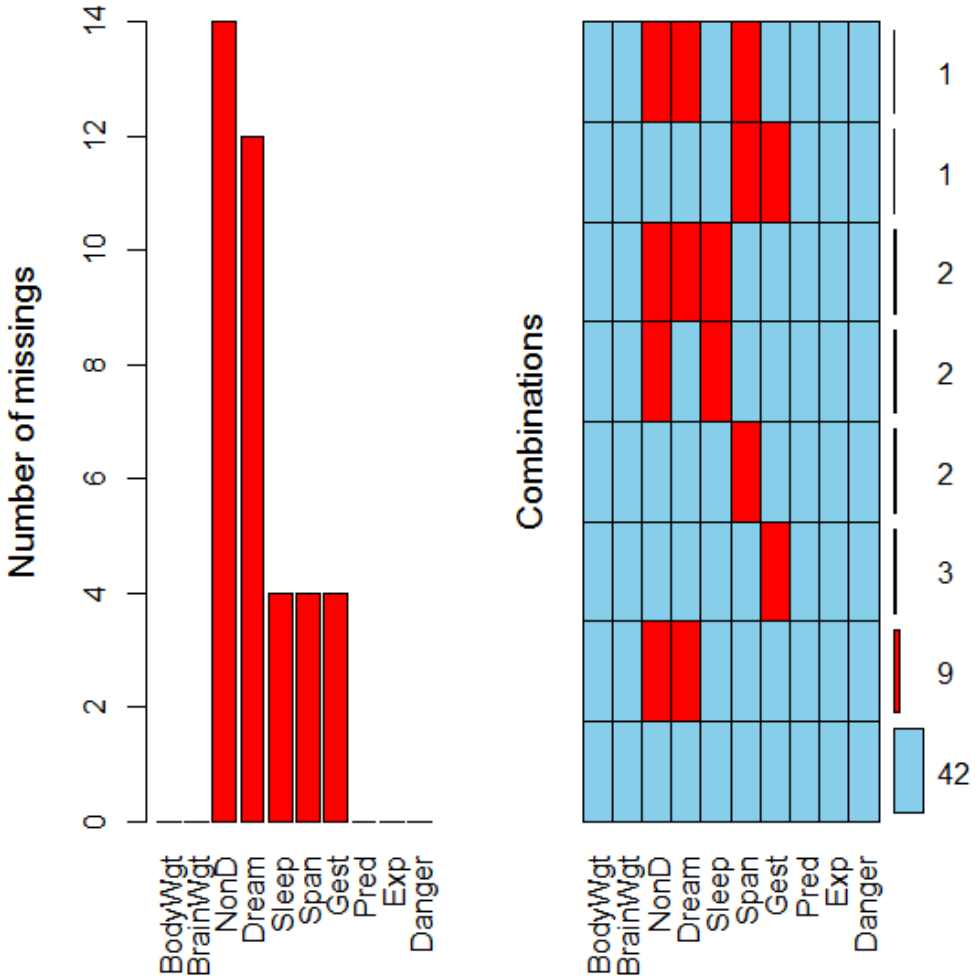
缺失值

```
> library(mice)
> data(sleep, package="VIM")
> md.pattern(sleep)
   BodyWgt BrainWgt Pred Exp Danger Sleep Span Gest Dream NonD
42       1        1    1   1      1     1    1    1     1    0
 2       1        1    1   1      1     1    0    1     1    1
 3       1        1    1   1      1     1    1    0     1    1
 9       1        1    1   1      1     1    1    1     0    0
 2       1        1    1   1      1     0    1    1     1    0
 1       1        1    1   1      1     1    0    0     1    1
 2       1        1    1   1      1     0    1    1     0    0
 1       1        1    1   1      1     1    0    1     0    0
         0        0    0   0      0     4    4    4    12   14
```

R语言数据管理

缺失值

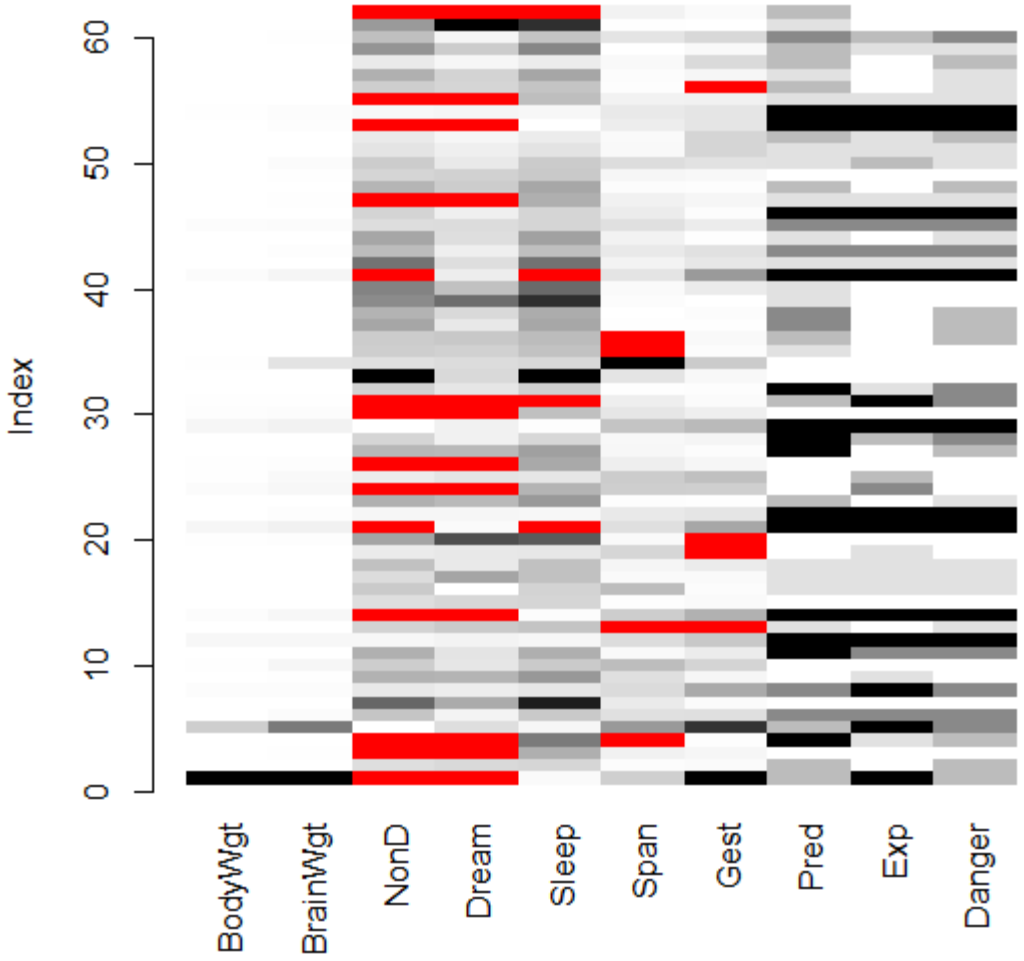
```
library("VIM")
aggr(sleep, prop=FALSE, numbers=TRUE)
```



R语言数据管理

缺失值

```
matrixplot(sleep)
```



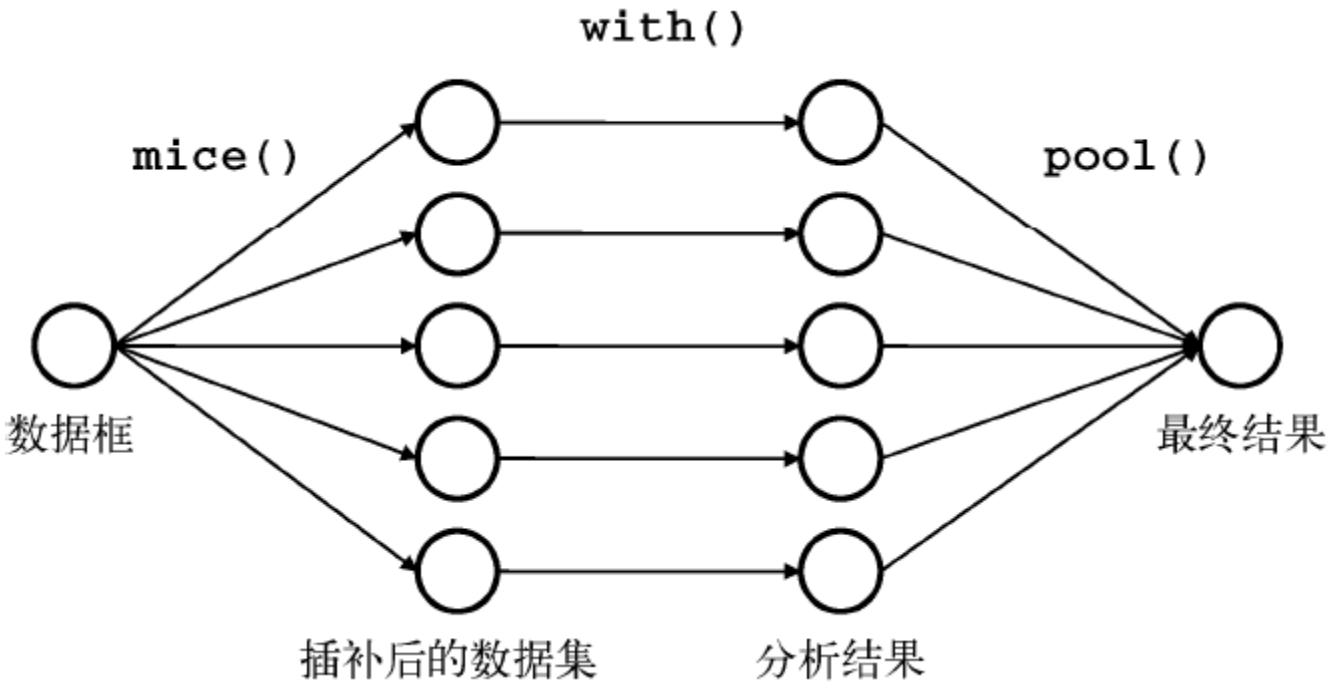
R语言数据管理

缺失值

```
newdata <- mydata[complete.cases(mydata),]

newdata <- na.omit(mydata)
```

```
> options(digits=1)
> cor(na.omit(sleep))
          BodyWgt BrainWgt NonD Dream Sleep  Span  Gest  Pred  Exp Danger
BodyWgt       1.0      0.9 -0.4 -0.07  -0.3   0.5   0.7   0.1  0.4    0.3
BrainWgt      1.0      1.0 -0.4 -0.07  -0.3   0.6   0.7 -0.02  0.3    0.1
NonD         -0.4     -0.4  1.0  0.52   1.0 -0.4  -0.6  -0.4 -0.6   -0.5
Dream        -0.1     -0.1  0.5  1.00   0.7 -0.3  -0.4  -0.4 -0.5   -0.6
Sleep        -0.3     -0.3  1.0  0.72   1.0 -0.4  -0.6  -0.4 -0.6   -0.6
Span          0.5      0.6 -0.4 -0.27  -0.4   1.0   0.6  -0.2  0.3    0.0
Gest          0.7      0.7 -0.6 -0.41  -0.6   0.6   1.0   0.1  0.6    0.3
Pred          0.1     -0.0 -0.4 -0.40  -0.4  -0.2   0.1   1.0  0.6    0.9
Exp           0.4      0.3 -0.6 -0.50  -0.6   0.3   0.6   0.6  1.0    0.8
Danger        0.3      0.1 -0.5 -0.57  -0.6   0.0   0.3   0.9  0.8    1.0
```

缺失值



R语言数据管理

缺失值

```
library(mice)
imp <- mice(mydata, m)
fit <- with(imp, analysis)
pooled <- pool(fit)
summary(pooled)
```

```
> library(mice)
> data(sleep, package="VIM")
> imp <- mice(sleep, seed=1234)

 […output deleted to save space…]

> fit <- with(imp, lm(Dream ~ Span + Gest))
> pooled <- pool(fit)
> summary(pooled)
                 est      se      t   df Pr(>|t|)    lo 95
(Intercept)  2.58858 0.27552  9.395 52.1 8.34e-13  2.03576
Span        -0.00276 0.01295 -0.213 52.9 8.32e-01 -0.02874
Gest        -0.00421 0.00157 -2.671 55.6 9.91e-03 -0.00736
                hi 95 nmis    fmi
(Intercept)  3.14141   NA 0.0870
Span         0.02322    4 0.0806
Gest        -0.00105    4 0.0537
```

R语言数据管理

缺失值

软　件　包	描　　述
Hmisc	包含多种函数，支持简单插补、多重插补和典型变量插补
mvnmle	对多元正态分布数据中缺失值的最大似然估计
cat	对数线性模型中多元类别型变量的多重插补
arrayImpute、arrayMissPattern、SeqKnn	处理微阵列缺失数据的实用函数
longitudinalData	相关的函数列表，比如对时间序列缺失值进行插补的一系列函数
kmi	处理生存分析缺失值的Kaplan-Meier多重插补
mix	一般位置模型中混合类别型和连续型数据的多重插补
pan	多元面板数据或聚类数据的多重插补

R语言数据管理

数据排序

```
newdata <- leadership[order(leadership$age),]

attach(leadership)
newdata <- leadership[order(gender, age),]
detach(leadership)


attach(leadership)
newdata <-leadership[order(gender, -age),]
detach(leadership)
```

R语言数据管理

数据集合并

```
total <- cbind(A, B)

total <- rbind(dataframeA, dataframeB)

total <- merge(dataframeA, dataframeB, by="ID")

total <- merge(dataframeA, dataframeB, by=c("ID","Country"))
```

R语言数据管理

数据子集

```
newdata <- leadership[, c(6:10)]

newdata <- leadership[c(-8,-9)]

newdata <- leadership[1:3,]

newdata <- leadership[which(leadership$gender=="M" &
                            leadership$age > 30),]

attach(leadership)
newdata <- leadership[which(gender=='M' & age > 30),]
detach(leadership)
```

R语言数据管理

数据子集

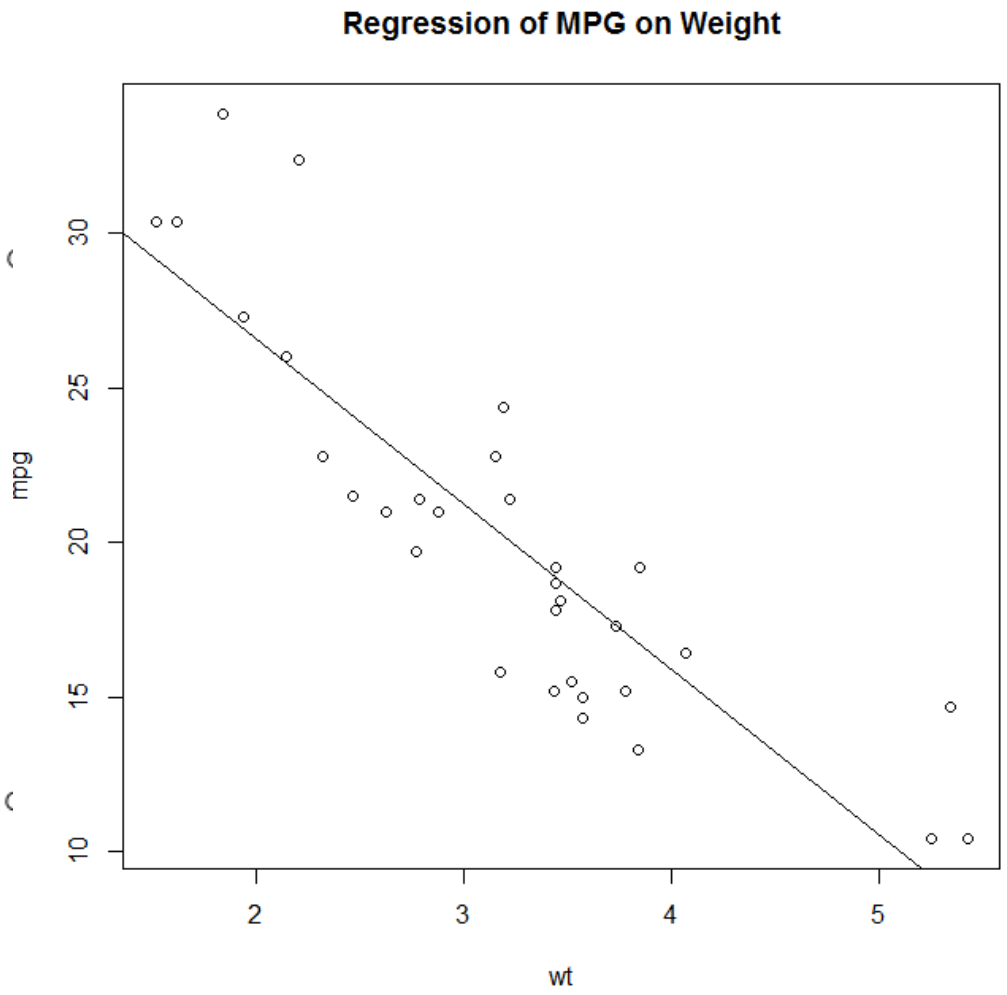
```
newdata <- subset(leadership, age >= 35 | age < 24,
                  select=c(q1, q2, q3, q4))

newdata <- subset(leadership, gender=="M" & age > 25,
                  select=gender:q4)


mysample <- leadership[sample(1:nrow(leadership), 3, replace=FALSE),]
```

R语言绘图

Hello例子

```
attach(mtcars)
plot(wt, mpg)
abline(lm(mpg~wt))
title("Regression of MPG (
detach(mtcars)
```

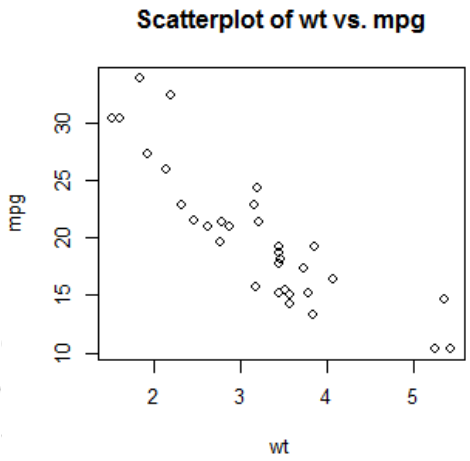
```
pdf("mygraph.pdf")
attach(mtcars)
plot(wt, mpg)
abline(lm(mpg~wt))
title("Regression of MPG (
detach(mtcars)
dev.off()
```

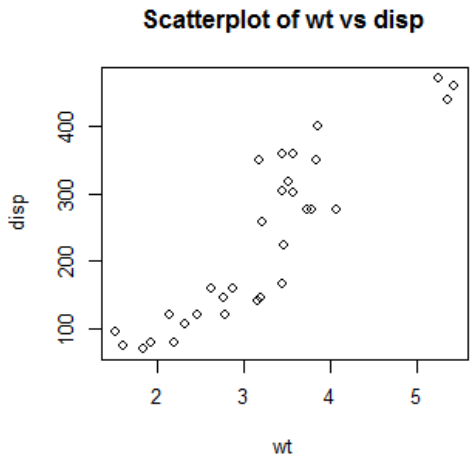


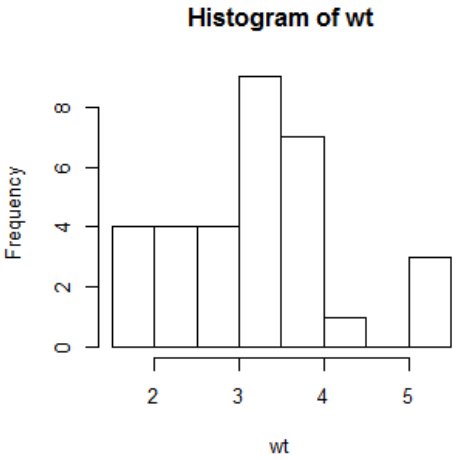
R语言绘图

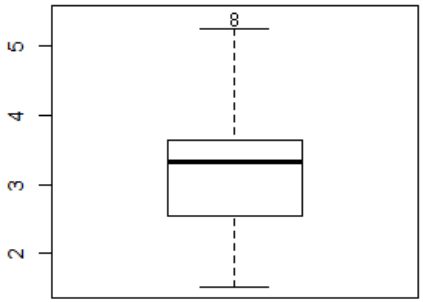
plot

```
attach(mtcars)
opar <- par(no.readonly=TRUE)
par(mfrow=c(2,2))
plot(wt,mpg, main="Scatterplo
plot(wt,disp, main="Scatterpl
hist(wt, main="Histogram of w
boxplot(wt, main="Boxplot of
par(opar)
detach(mtcars)
```





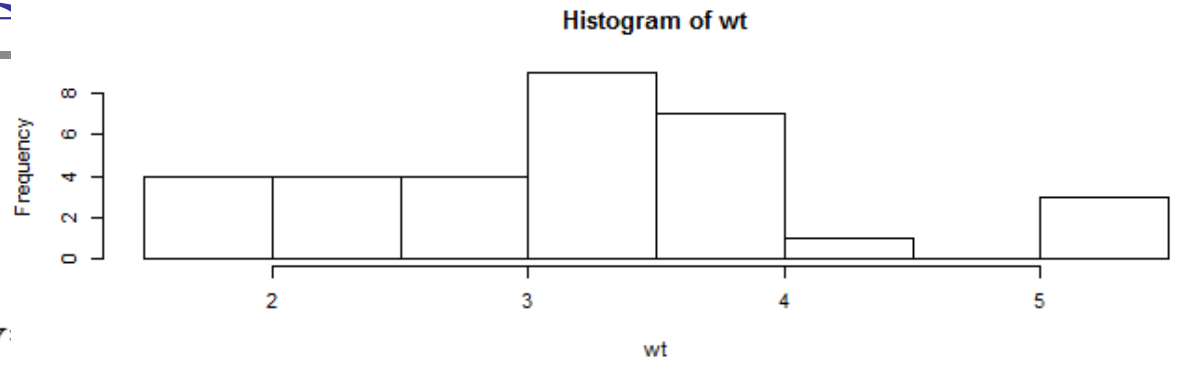


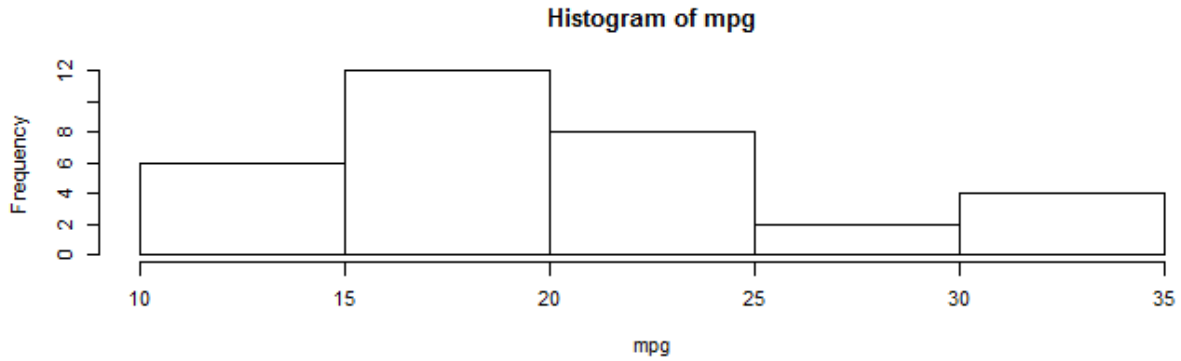


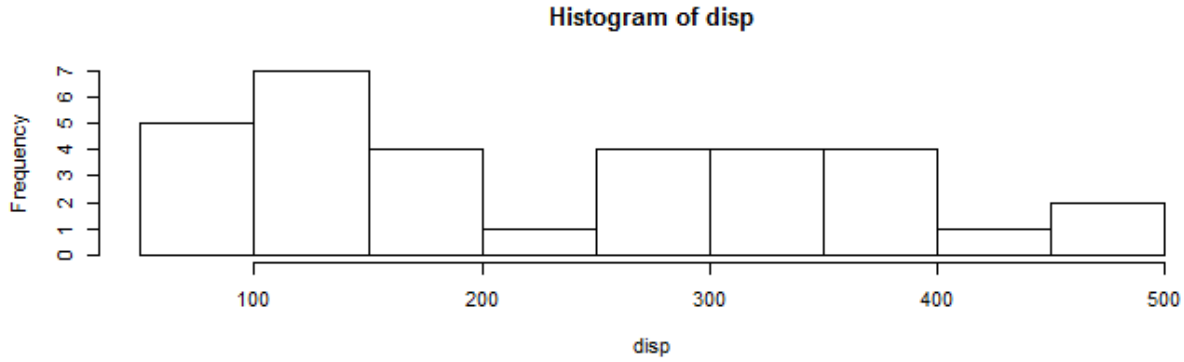
R语言绘图

plot

```
attach(mtcars)
opar <- par(no.readonly
par(mfrow=c(3,1))
hist(wt)
hist(mpg)
hist(disp)
par(opar)
detach(mtcars)
```



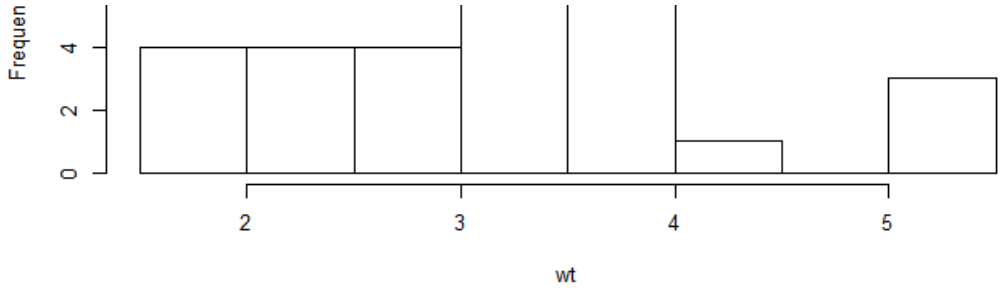


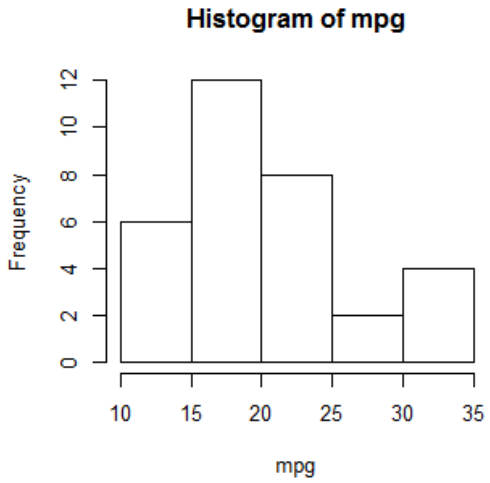


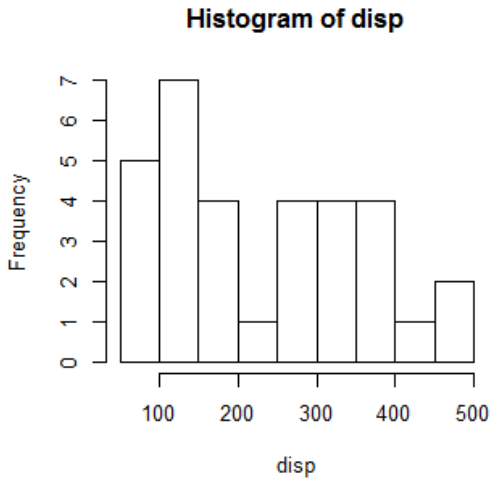
R语言绘图

plot

```
attach(mtcars)
layout(matrix(c(1,1,2,3), 2, 2, byrow = TRUE))
hist(wt)
hist(mpg)
hist(disp)
detach(mtcars)
```



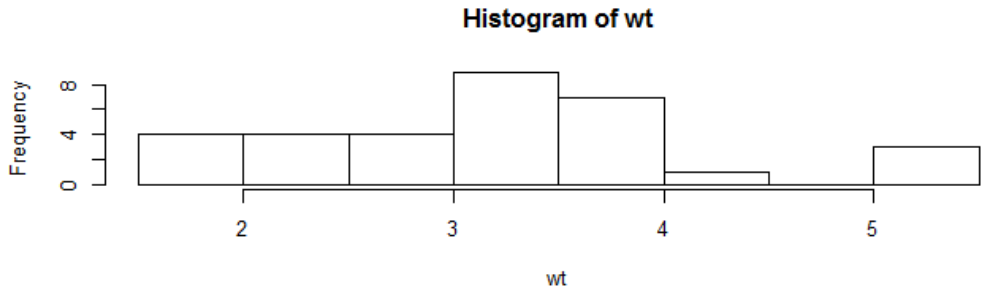


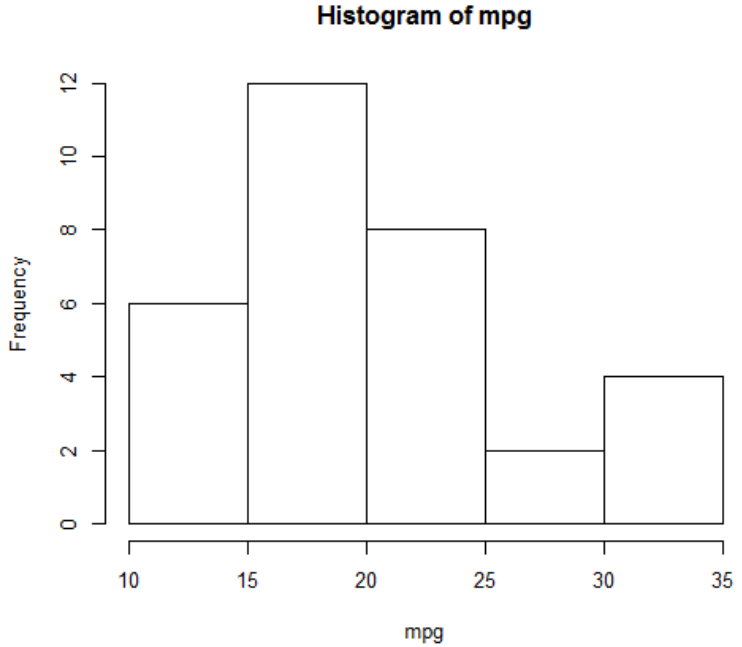


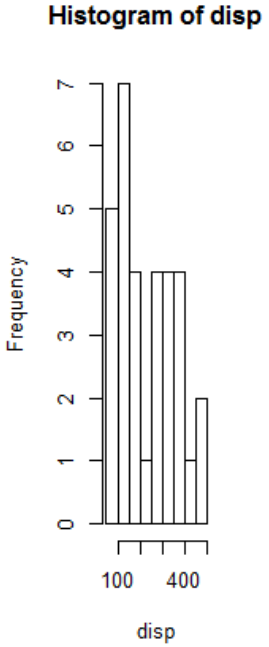
R语言绘图

plot

```
attach(mtcars)
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE),
       widths=c(3, 1), heights=c(1, 2))
hist(wt)
hist(mpg)
hist(disp)
detach(mtcars)
```







plot

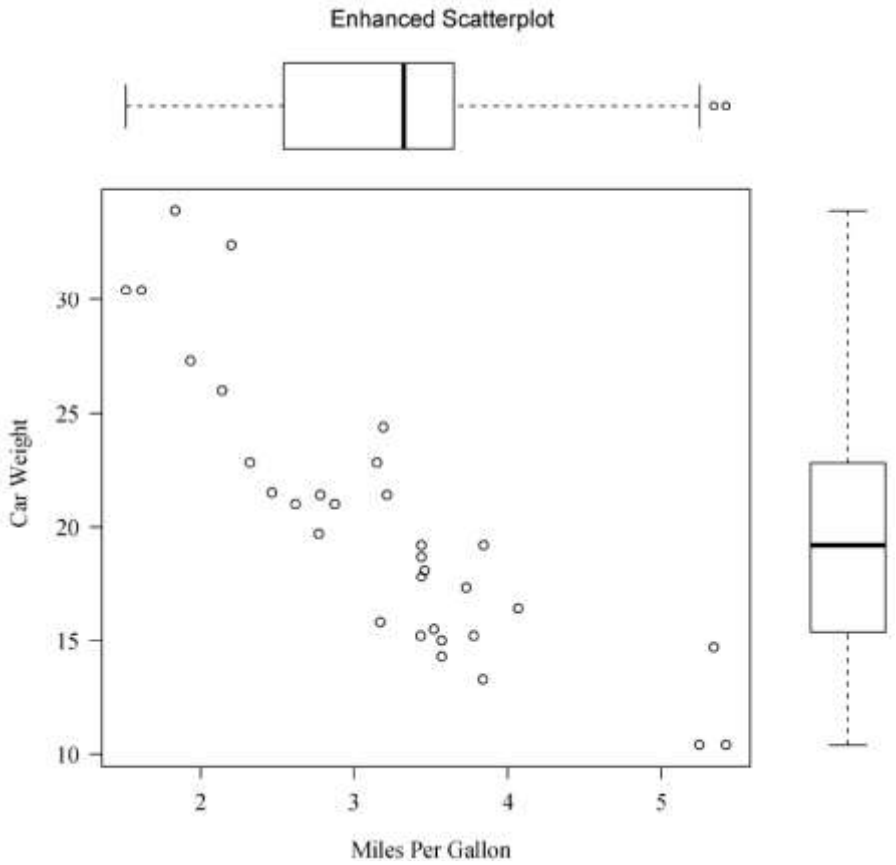
```
opar <- par(no.readonly=TRUE)
par(fig=c(0, 0.8, 0, 0.8))
plot(mtcars$wt, mtcars$mpg,
     xlab="Miles Per Gallon",
     ylab="Car Weight")
```

⟵ 设置散点图

```
par(fig=c(0, 0.8, 0.55, 1), new=TRUE)
boxplot(mtcars$wt, horizontal=TRUE, axes=FALSE)
par(fig=c(0.65, 1, 0, 0.8), new=TRUE)
boxplot(mtcars$mpg, axes=FALSE)

mtext("Enhanced Scatterplot", side=3, outer=TRUE, line=-3)
par(opar)
```

⟵ 在上方添加箱线图

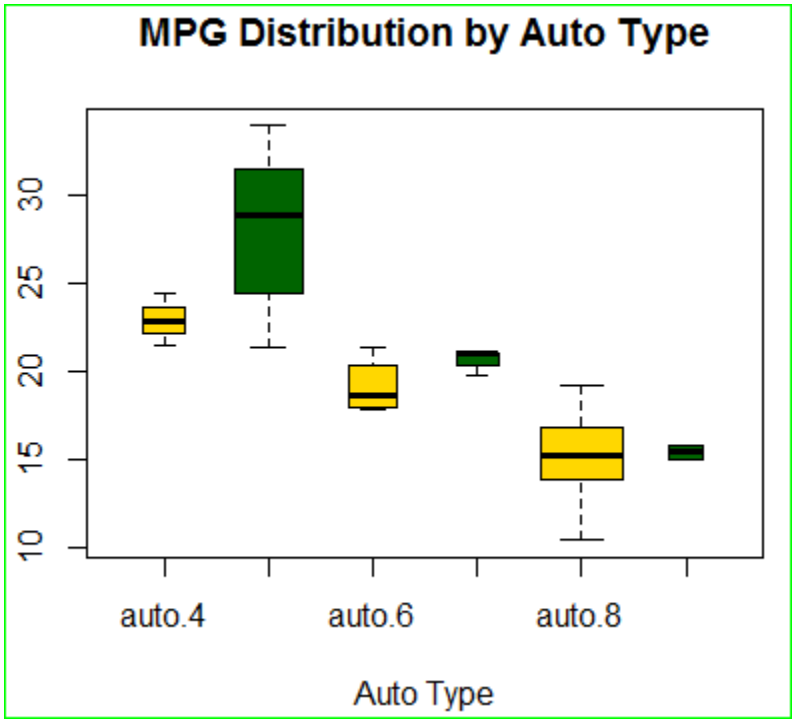
⟵ 在右侧添加箱线图



R语言绘

plot

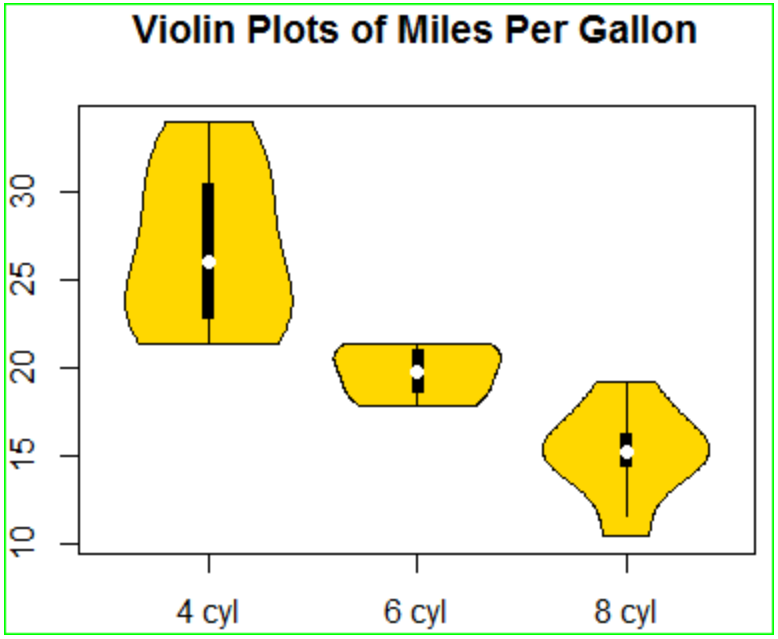
```
mtcars$cyl.f <- factor(mtcars$cyl,
                       levels=c(4,6,8),
                       labels=c("4","6","8"))

mtcars$am.f <- factor(mtcars$am,
                      levels=c(0,1),
                      labels=c("auto", "standard"))

boxplot(mpg ~ am.f *cyl.f,
        data=mtcars,
        varwidth-TRUE,
        col=c("gold","darkgreen"),
        main="MPG Distribution by Auto Type",
        xlab="Auto Type")
```



R语言绘图

gplot

```
library(vioplot)
x1 <- mtcars$mpg[mtcars$cyl==4]
x2 <- mtcars$mpg[mtcars$cyl==6]
x3 <- mtcars$mpg[mtcars$cyl==8]
vioplot(x1, x2, x3,
        names=c("4 cyl", "6 cyl", "8 cyl"),
        col="gold")
title("Violin Plots of Miles Per Gallon")
```



R语言绘图

ggplot2

- 核心理念是将绘图与数据分离，数据相关的绘图与数据无关的绘图分离
- 按**图层**作图
- 保有命令式作图的调整函数，使其更具灵活性
- 将常见的统计变换融入到了绘图中。
- 的绘图的特点：1. 有明确的起始（以ggplot函数开始）与终止（一句语句一幅图）；2. 图层之间的叠加是靠“+”号实现的，越后面其图层越高。
- 图的元素：最大的是plot（指整张图，包括background和title），其次是axis（包括stick，text，title和stick）、legend（包括backgroud、text、title）、facet这是第二层次，其中facet可以分为外部strip部分（包括backgroud和text）和内部panel部分（包括backgroud、boder和网格线grid，其中粗的叫grid.major，细的叫grid.minor）。

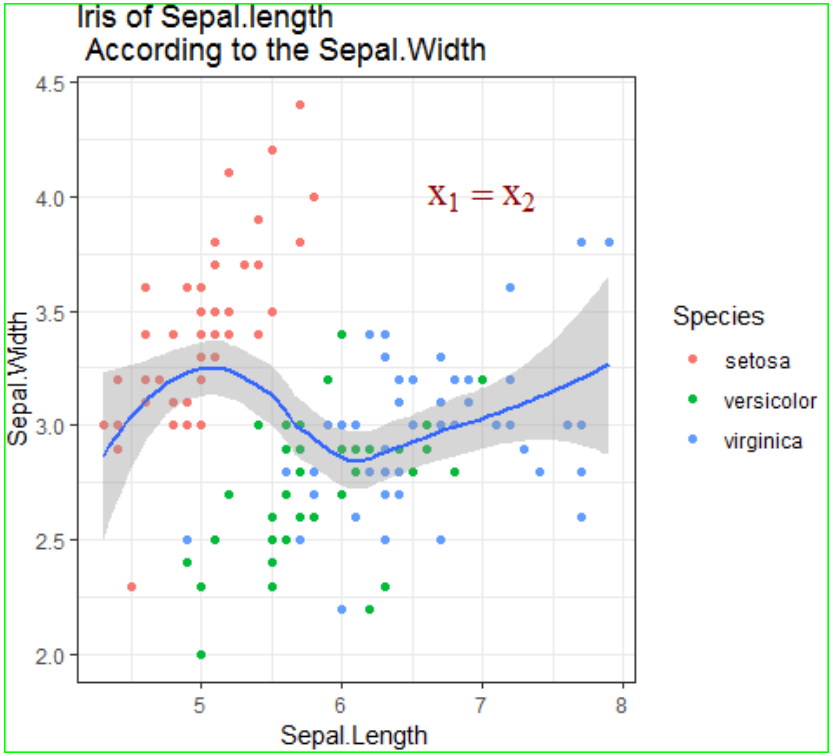
R语言绘图

ggplot2

```
ggplot(data = , aes(x = , y = )) +
   geom_XXX(...) + ... + stat_XXX(...) + ... +
   annotate(...) + ... + labs(...) +
   scale_XXX(...) + coord_XXX(...) + guides(...) +
theme(...) +
   facet_XXX(...)
```

R语言绘图

ggplot2




```
library(ggplot2)
attach(iris)
p <- ggplot(data=iris,aes(x =
Sepal.Length,y = Sepal.Width))
p + geom_point(aes(colour = Species)) +
stat_smooth() +
labs(title = "Iris of Sepal.length \n
According to the Sepal.Width") +
theme_classic() + theme_bw()
+annotate("text",x=7,y=4,parse =
T,label = "x[1]==x[2]",size=6,
family="serif",fontface="italic",
colour="darkred")
```

一个例子

将学生的各科考试成绩组合为单一的成绩衡量指标、基于相对名次（前20%，下20%，等等）给出从A到F的评分、根据学生姓氏和名字的首字母对花名册进行排序。

R语言高级数据管理

一个例子

```
> options(digits=2)

> Student <- c("John Davis", "Angela Williams", "Bullwinkle Moose",
               "David Jones", "Janice Markhammer", "Cheryl Cushing",
               "Reuven Ytzrhak", "Greg Knox", "Joel England",
               "Mary Rayburn")
> Math <- c(502, 600, 412, 358, 495, 512, 410, 625, 573, 522)
> Science <- c(95, 99, 80, 82, 75, 85, 80, 95, 89, 86)
> English <- c(25, 22, 18, 15, 20, 28, 15, 30, 27, 18)
> roster <- data.frame(Student, Math, Science, English,
                       stringsAsFactors=FALSE)
```

R语言高级数据管理

一个例子

```
> options(digits=2)
> roster
               Student Math Science English
1          John Davis  502      95      25
2    Angela Williams  600      99      22
3   Bullwinkle Moose  412      80      18
4         David Jones  358      82      15
5  Janice Markhammer  495      75      20
6     Cheryl Cushing  512      85      28
7     Reuven Ytzrhak  410      80      15
8           Greg Knox  625      95      30
9       Joel England  573      89      27
10      Mary Rayburn  522      86      18
```

R语言高级数据管理

一个例子

```
> z <- scale(roster[,2:4])
> z
         Math Science English
 [1,]  0.013   1.078   0.587
 [2,]  1.143   1.591   0.037
 [3,] -1.026  -0.847  -0.697
 [4,] -1.649  -0.590  -1.247
 [5,] -0.068  -1.489  -0.330
 [6,]  0.128  -0.205   1.137
 [7,] -1.049  -0.847  -1.247
 [8,]  1.432   1.078   1.504
 [9,]  0.832   0.308   0.954
[10,]  0.243  -0.077  -0.697
```

R语言高级数据管理

一个例子

```
> score <- apply(z, 1, mean)
> roster <- cbind(roster, score)
> roster
               Student   Math  Science English  score
1          John Davis   502       95       25    0.559
2     Angela Williams   600       99       22    0.924
3    Bullwinkle Moose   412       80       18   -0.857
4         David Jones   358       82       15   -1.162
5   Janice Markhammer   495       75       20   -0.629
6      Cheryl Cushing   512       85       28    0.353
7      Reuven Ytzrhak   410       80       15   -1.048
8           Greg Knox   625       95       30    1.338
9         Joel England   573       89       27    0.698
10        Mary Rayburn   522       86       18   -0.177
```

R语言高级数据管理

一个例子

```
> y <- quantile(roster$score, c(.8,.6,.4,.2))
> y
  80%   60%   40%   20%
 0.74  0.44 -0.36 -0.89
```

```
> roster$grade[score >= y[1]] <- "A"
> roster$grade[score < y[1] & score >= y[2]] <- "B"
> roster$grade[score < y[2] & score >= y[3]] <- "C"
> roster$grade[score < y[3] & score >= y[4]] <- "D"
> roster$grade[score < y[4]] <- "F"
> roster
               Student Math Science English  score grade
1          John Davis  502      95      25  0.559     B
2    Angela Williams  600      99      22  0.924     A
3   Bullwinkle Moose  412      80      18 -0.857     D
4        David Jones  358      82      15 -1.162     F
5  Janice Markhammer  495      75      20 -0.629     D
6     Cheryl Cushing  512      85      28  0.353     C
7     Reuven Ytzrhak  410      80      15 -1.048     F
8          Greg Knox  625      95      30  1.338     A
9        Joel England  573      89      27  0.698     B
10       Mary Rayburn  522      86      18 -0.177     C
```

R语言高级数据管理

一个例子

```
> name <- strsplit((roster$Student), " ")
> name

[[1]]
[1] "John"   "Davis"

[[2]]
[1] "Angela"    "Williams"

[[3]]
[1] "Bullwinkle" "Moose"

[[4]]
[1] "David" "Jones"

[[5]]
[1] "Janice"      "Markhammer"
```

R语言高级数据管理

控制流

```
for (i in 1:10)  print("Hello")
```

```
i <- 10
while (i > 0) {print("Hello"); i <- i - 1}
```

```
if (is.character(grade)) grade <- as.factor(grade)
if (!is.factor(grade)) grade <- as.factor(grade) else print("Grade already
    is a factor")
```

```
ifelse(score > 0.5, print("Passed"), print("Failed"))
outcome <- ifelse (score > 0.5, "Passed", "Failed")
```

```
feelings <- c("sad", "afraid")
for (i in feelings)
  print(
    switch(i,
      happy  = "I am glad you are happy",
      afraid = "There is nothing to fear",
      sad    = "Cheer up",
      angry  = "Calm down now"
    )
  )
```

R语言高级数据管理

自定义函数

```
myfunction <- function(arg1, arg2, ... ){
  statements
  return(object)
}
```

R语言高级数据管理

自定义函数

```
mystats <- function(x, parametric=TRUE, print=FALSE) {
  if (parametric) {
    center <- mean(x); spread <- sd(x)
  } else {
    center <- median(x); spread <- mad(x)
  }
  if (print & parametric) {
    cat("Mean=", center, "\n", "SD=", spread, "\n")
  } else if (print & !parametric) {
    cat("Median=", center, "\n", "MAD=", spread, "\n")
  }
  result <- list(center=center, spread=spread)
  return(result)
}
```

```
set.seed(1234)
x <- rnorm(500)
```

```
y <- mystats(x, parametric=FALSE, print=TRUE)
```

Thank You!