

四则运算

`^` 幂运算

`%%` 取余运算

`%/%` 整除运算

`round()` 四舍五入取整数

`floor()` 向下取整

`ceiling()` 向上取整

`round(3.14159, 2)` # 四舍五入到两位小数

[1] 3.14

逻辑运算应该注意的点

判断两个浮点型对象是否完全相同，不能直接采用==和identical()，而应该用all.equal()

字符型数据的处理

字符型数据的处理 (cont)

`strsplit()` 拆分两个及以上的字符型对象，默认一个

`substr(x, start, stop)` 还可以代替特定位置

`nchar(" R统计 软件", type = " by t es")` # 以字节为单位

[1] 9

`nchar("code monkey \t")`

[1] 12 (空格算，\t算一个)

`paste()` 中的 `sep` 控制用什么连接，要直接连接用''

`strsplit()` 中的 `split` 指示用什么字符拆分

日期时间类型数据

Date 一般用整数保存，数值为从1970-01-01经过的天数。

POSIXct 从1970年1月1日零时到该日期时间的时间间隔秒数

POSIXlt 一个包含年、月、日、星期、时、分、秒等成分的列表

```
diffftime( as.Date("2021-09-10"), as.Date("2020-09-10"),
units = 'days')
```

Time difference of 31.33333 days

```
diffftime( as.Date("2021-09-10"), as.Date("2020-09-10"),
units = 'days')
```

Time difference of 365 days

从字符串生成日期数据

- `as.Date()` 可将字符型数据转换成日期型数据
- 通过 `format` 指定输入的格式，默认的格式中分隔符为 - 或 /
- 如果不是标准格式，可通过以下方式指定

代码	意义
%d	日
%m	月 (数字格式)
%b	月 (英文简称)
%B	月 (英文全称)
%y	年 (两位)
%Y	年 (四位)

- 提取组成部分: `weekdays`, `months`, `days`, `quarters`

因子类变量

`factor(c(" 男", " 女"))`

因子类变量 (cont)

`factor(LE TTE RS[1:3], ordered = TRUE, " A")`

因子的 `levels()` (水平值) 属性是一个映射，把整数1,2,映射成这些水平值，因子在保存时会保存成整数1,2,等与水平值对应的编号。

R的数据类型

整型(int) : 如1L

数值型/双整型(numeric, double) : 如1,1.1

逻辑型(logical) : 只有两个值TRUE和FALSE, 缺失时为NA。

字符型(character) : 存储一小段文本，用双引号包住，其中单个元素称之为字符串(string)，如"Hello", "1"

复数类型(complex) : 如1+3i

日期时间类型(Date, POSIXct, POSIXlt) : 如 `Sys.time()`

因子类型(factor)

特殊符号 : NA(Not Available), NaN(Not a Number), Inf(infinite), NULL

typeof() 返回数据类型

is.foo() 判断是否属于某种类型foo，是返回TRUE，否返回FALSE

as.foo() 强制转换成foo类型

矩阵

`matrix()` 函数把矩阵元素以向量的形式输入，用 `nrow` 和 `ncol` 规定行数和列数，向量元素填入的缺省次序是按列填入，用 `byrow=TRUE` 选项可转换成按行填入。

`rbind()`, `cbind()`, `diag()`, `dim()`, `dimnames()`

对两个同形状的矩阵，`*` 表示两个矩阵对应元素相乘，`/` 表示两个矩阵对应元素相除

`%*%` 矩阵乘法，`t()` 转置，`det()` 行列式，`solve()` 逆

`solve(A, b)` 返回的是线性方程组 $Ax=b$ 的解

男 女

Levels:

男 女

`factor(LE TTE RS[1:3], ordered = TRUE)`

A B C

Levels:

A < B <

C

<code>nchar()</code>	计算字符串的长度
<code>toupper()</code> , <code>tolower()</code>	转换大小写
<code>substr(x, start, stop)</code>	从x中取出start到stop的子串
<code>substr ing(x, start)</code>	从x中取出start到末尾的子串
<code>gsub(p attern, replac ement, x)</code>	x中与p attern对应的字符-替换成replac ement
<code>paste()</code>	连接两个字符-型对象，默认-用空格连接



By 林漪 (kevin123)
cheatography.com/kevin123/

Not published yet.
 Last updated 5th June, 2022.
 Page 1 of 5.

Sponsored by **CrosswordCheats.com**
 Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

矩阵 (cont)

矩阵内积自己记一下

apply(A, i, FUN) 把矩阵 A 的每一列分别输入到函数 FUN 中, 得到对应于每一维度的结果, 其中 i = 1 表示对行进行运算, i = 2 表示对列进行运算。

矩阵的下标和子集与向量类似。

数据框

各列之间允许有不同的类型, 同一列中的元素保持相同类型。

数据框之中有增加行列, 命名, 访问, with- () 的用法, 稍后再来搞

列表

不同于之前, 列表(list)是用来保存不同类型的数据。

可通过 names() 来命名

```
names( dist) <- c("a ", " b", " c")
```

也可在一开始定义的时候就命名好

```
dist1 <- list(a = " exp one nti al", b = 7, c = FAL  
SE)
```

单个列表元素必须用两重方括号格式访问如dist[[1]]

使用单重方括号对列表取子集结果还是列表而不是列表元素

直接给列表不存在的元素名定义元素值就添加了新元素

把某个列表元素赋值为 NULL 就删掉这个元素

要把已经存在的元素修改为 NULL 值而不是删除此元素, 或者给列表增加一个取值为 NULL 的元素, 这时需要用单重的方括号取子集, 这样的子集会保持其列表类型, 给这样的子列表赋值为 list(NULL)。如d

```
ist[' was.es tim ated'] <- list(NULL)
```

as.list() 将其他转换成列表, unlist()把列表转换成基本向量

矩阵注意点

提取A的第一行结果为向量, 维数会有不同, drop

寻找矩阵中的最小元素, 并返回其位置

```
mat <- matrix (rn orm (40), 10, 4)
```

位置

```
which(mat == min(mat, na.rm= TRUE),
```

向量下标

正整数下标: 访问对应位置的元素和子集

负整数下标: 扣除相应的元素后的子集

下标超界返回NA

下标可以是与向量等长的逻辑表达式

元素名下标: 向量可以为每个元素命名, 命名后即可用元素名或者元素名向量作为向量的下标。

重复下标: R在使用整数或元素名作为向量

下标时, 允许使用重复下标。

向量

```
seq(from = 5, to = 25, by = 5)
```

```
seq(as.Da te( '20 21- 10-1'), by='days',  
h=2)
```

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

整理数据

行号不一样, 用 merge(dat1, dat2, by = x)

按照列名为x来合并, 只保留x元素相同的 TRUE

行, 即同时在两个数据框中的行

用merge(dat1, dat2, by=x 返回行号和列号

dat1中的x列和 dat2中的y列作为合并的标准。

如果想要保留 dat1 中的所有行, 则指定

all.x = TRUE。(?)

scale() 把每一列都标准化, 即每一列都减去该列的平均值, 然后除以该列的样本标准差。

scale(x, center=TRUE, scale=FALSE) 仅中心化而不标准化。

仅适用于数值型的变量

汇总数据

总体信息 summary(), table()

位置度量 5 mean(), median()

分散程度 (变异性) sd(), IQR(), mad()

度量 15

分位数 20 min(), max(),
25 quantile()

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

对于因子类型的变量, 可以通过table()查看其在每一类的频数分布。

可通过 na.rm = TRUE 将其中NA的数值去除来计算平均值、标准差、中位数等。

读入数据

```
d2 <- read.csv( " covid19.csv", header = TRUE, na.strings = " ",
= 10)
```

readLines()可以读文本文件

```
idx.na <- apply( is.na(d), 1
, any) d[idx.na, ]返回有缺失值的行
head(x, n) 选择数据框 x 的前 n 行
tail(x, n) 选择数据框 x 的倒数 n 行
```

C

By 林漪 (kevin123)
cheatography.com/kevin123/

Not published yet.
Last updated 5th June, 2022.
Page 2 of 5.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

分组汇总数据 (cont)

可以通过 `useNA = "always"` 或 `useNA = "ifany"` 来把 NA 计算在内
对两个分类变量进行交叉分组计算频数
`table(d[, "分型"], d[, "性别"])`

随机数

`sample(x, size, replace = FALSE, prob = NULL)`
`x` 用以存储有限集合的向量, `size` 指定抽样个数, `prob` = 指定以各种权重抽取, 默认是等概率。
`replace` = 指定是否为有放回抽样, `TRUE` 是有放回抽样, `FALSE` 是无放回抽样即样本数

`set.seed(seed, kind = NULL, normal.kind = NULL, sample.kind = NULL)`
`seed = k` 指定一个编号为 `k` 的种子, `kind` = 指定后续程序要使用的随机数发生器名称; `normal.kind` = 指定要使用的正态分布随机数发生器名称。

这是古典概型的例子。

随机排序 `sample(10) sample(letters)`

多项分布的随机抽样

`sample(1:3, size = 100, replace = TRUE, prob = c(.2, .3, .5))`

随机数函数

每一种分布都有自己的名字, 在其前面添加如下的字母分别代表不同的功能
`p` 分布函数 `q` 分位数 `d` 概率密度函数 `r` 随机数

随机数函数 (cont)

指数分布的概率密度函数

```
x <- seq(0, 8, .05)
plot(x, dexp(x), ty="l", main="题目",
      xlab="x", ylab="f(x)")
lines(x, dexp(x, rate=0.5), col="red")
lines(x, dexp(x, rate=0.2), col="blue")
legend("topright", legend = paste("lambda = ", c(1, 0.5, 0.2)), col=c("black", "red", "blue"), lty=1, inset = .02)
```

指数分布的随机数

```
x <- seq(0, 16, .05)
hist(rexp(1000, 0.5), freq = FALSE,
      xlab="x", main="题目")
lines(x, dexp(x, 0.5), col="red", lwd=2)
```

`runif(n)` 产生 `n` 个标准均匀分布随机数

`rnorm(n)` 产生 `n` 个标准正态分布随机数

