数据分析与处理技术

商学院2016级 物流管理专业
徐宁

一. 数据类型-数值型

- 数值型：double 或 numeric

```
> 12
[1] 12
> -7
[1] -7
>
> pi
[1] 3.141593
```

```
> sin(0.5*pi)
[1] 1
```

科学计数法

```
> 2^50
[1] 1.1259e+15

> 2^-50
[1] 8.881784e-16
```

判断某变量中储存的数据是数值型

```
> a<-5
> a
[1] 5
> is.numeric(a)
[1] TRUE
```

某些特殊数值被赋予固定符号表示

例如：

```
> NA
[1] NA
> 1+NA
[1] NA
> 0/0
[1] NaN
> 6+Inf
[1] Inf
```

圆周率	pi	
空值	NA	Not available
空缺值	NaN	Not a number
无/删除	NULL	
无穷大	Inf	Infinite

数值运算

算术运算

+	-	*	/
^	%%		

算术运算的结果是数值类型

```
> 1+2
[1] 3
```

```
> 10%%3
[1] 1
> 10%%2
[1] 0
```

常用数学函数：

sin( )　cos( )　tan( )　exp( ) log( )

常见统计函数：

sum( )　mean( )　min( )　length( )

注意：R中的log其实是自然对数ln，而以10为底的对数函数是log10( )

算术运算符和函数运算的共同之处在于：它们都是函数运算。

向量与向量化运算

向量:建立向量需用组合符号c( )

```
> a<-c(1,2,5,7)
> a
[1] 1 2 5 7
```

无论用多少c( )镶套都会自动展开

```
> a<-c(1,2,5,c(2,4,9))
> a
[1] 1 2 5 2 4 9
```

观察下边三种等效的向量生成方法

```
> a<-c(1,2,3,4,5,6,7,8,9,10)
> a
 [1]  1  2  3  4  5  6  7  8  9 10
> a<-1:10
> a<-seq(1,10,by=1)
```

":"是非常常用的方法，并且两边不必加c( )

向量化运算：整体批量运算，即对向量里每一个元素进行运算并返回一个向量结果

算术运算情况 →

```
> a
[1] 1 2 5 2 4 9
> a*2
[1]  2  4 10  4  8 18
> a+2
[1]  3  4  7  4  6 11
```

函数运算情况 ↓

```
> sin(5)
[1] -0.9589243
> sin(c(1,2,3,5,7))
[1]  0.8414710  0.9092974  0.1411200 -0.9589243  0.6569866
```

常用数学函数

函数	功能描述
abs(x)	绝对值
sqrt(x)	开方，等效于x^0.5
ceiling(x)	不小于x的最小整数，ceiling(3.4)得到4
floor(x)	不大于x的最大整数
trunc(x)	截取x的整数部分
round(x,digits=n)	保留n位小数的四舍五入
signif(x,digits=n)	保留n位有效数字的四舍五入
sin(x) tan(x)	正弦 正切
asin(x) atan(x)	反正弦 反正切
log(x,base=n)	以n为底取对数
log(x) log10(x)	取自然对数，取以10为底的对数
exp(x)	指数函数

二. 数据类型-逻辑型

逻辑型：logical

逻辑型数据只有两个：TRUE和FALSE,简称T和F（注意必须全大写）

TRUE和FALSE对应了2进制中的1和0，关于它们的
运算规律在计算机基础以及高等数学中都涉及过。

条件判断的运算结果是逻辑型数据

```
> 1>2
[1] FALSE
> 1>0
[1] TRUE
```

TRUE与FALSE分别对应了1和0

```
> TRUE
[1] TRUE
> TRUE+FALSE
[1] 1
> TRUE+2
[1] 3
```

比较运算

- 比较运算返回的结果是逻辑数值“TRUE”或者“FASLE”，例如：

```
> 1>2
[1] FALSE
```

```
> 3>-1
[1] TRUE
```

```
> 2 %in% a
[1] TRUE
```

逻辑运算的向量化

```
> a
[1] 1 2 5 2 4 9
> a>3
[1] FALSE FALSE  TRUE FALSE  TRUE  TRUE
```

比较运算符

>	<	>=	<=
==	!=	%in%	

逻辑运算

逻辑型数据常常作为判断计算条件时的标志，经常会涉及条件的’与’、’或’、’非’等运算，即逻辑运算

```
> 2>3 & 2>0
[1] FALSE
```

逻辑符号	
符号	含义
&	与
|	或
!	非
&&	与运算
||	或运算

逻辑运算函数

```
> any(a>=9)
[1] TRUE
>
> all(a>5)
[1] FALSE
```

练习

1.计算从1连续加到125的结果。

2.利用随机函数生成100个期望为0方差为10的随机数，测试其中是否存在0

```
> x<-rnorm(100,mean=0,sd=10)
```

3.生成变量3<-1:6,思考下式运算结果：

```
> all(a>3)&any(a<3)
```

讨论：运算优先级

常用运算符优先级：
括号优先级最高，内层优先于外层；
乘方优先于乘除，乘除优先于加减，
负号优先于正号；加减优先于逻辑，
都算完后再赋值

+	-	*	/
^	%%		

>	<	>=	<=
==	!=	%in%	

&	|	!	

=	<-	:	( )

三. 数据类型-字符型

1 字符型数据需要扩在引号之内，
与数值型数据的创建方法相同

```
> a<-"Wang Qi"
> b<-c("Zhou Weijie","Zhang jingguang","吴毓雄")
```

2 字符数据的拼接需要使用一个
paste函数完成

```
> a='I am'
> b='a student'
> paste(a,b)
[1] "I am a student"
```

3 字符的显示使用cat( )函数

```
> cat(a,b)
I am a student
> cat(a,'\n',b)
I am
 a student
```

4 ‘\n’为转义符号，代表换行显示

```
> t<-"I am a  \n student"
> cat(t)
I am a
 student
```

转义符的规则来自于ASCII码，详
细内容见附件

四. 数据类型-因子型

因子型数据是一种专门用于分类的数据，相当于标签。例如，学生学籍档案中性别一栏只有'男'、'女'两种取值；某科成绩记录时可以记录为'优'、'良'、'中'、'差'。

因子型数据对应了这种类型。

标称型

```
> gender<-c("M","F","F","M","F")
> gender<-factor(gender)
> gender
[1] M F F M F Levels: F M
```

序数型

```
> status<-
c("Poor","Improved","Excellent","Poor")
> status<-factor(status,ordered=TRUE)
> status
[1] Poor Improved Excellent Poor Levels:
Excellent < Improved < Poor
```

```
> status<-factor(status,ordered=TRUE,
levels=c("Poor","Improved","Excellent")
)
> status
[1] Poor Improved Excellent Poor
Levels: Poor < Improved < Excellent
```

五. 数据类型-时间型

时间数据是建立在字符数据基础上的一种数据类型，我们创建时间数据时使用转类函数

```
> x<-as.Date('1997-07-01')
> x
[1] "1997-07-01"
> y<-as.Date('07/01/2001',format='%m/%d/%Y')
> y
[1] "2001-07-01"
```

遵守默认格式时可以省略format参数

例题：

计算你从入学到毕业的时间

```
> r1<-as.Date('2016-09-01')
> r2<-Sys.Date()
> r1
[1] "2016-09-01"
> r2
[1] "2018-09-09"
> r2-r1
Time difference of 738 days
```

调用系统时间

```
> Sys.time()
[1] "2017-09-12 16:12:00 CST“
> Sys.Date()
[1] "2017-09-12"
```

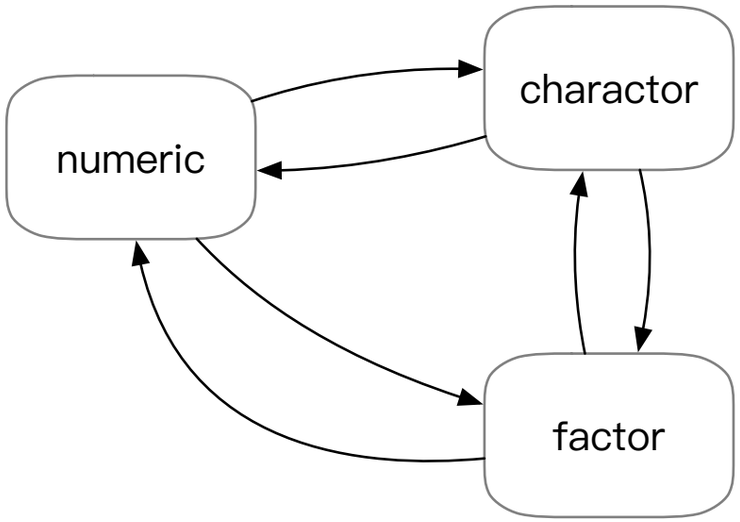
符号	含义
%d	数字日期
%a	缩写星期名
%A	非缩写星期
%m	数字月份
%b	缩写月份
%B	非缩写月份
%y	二位年份
%Y	四位年份

类型判断与类间强制转化

几种常用数据类型构成了R中数据类型基础，这已经足够解决绝大部分现实中的问题。

另外，R提供了一系列拓展数据类型的方法，但都是建立在基础类型之上。之后会逐步涉及创建新类型的方法。

```
> a
[1] 1 2 5 2 4 9
> a<-as.character(a)
> a
[1] "1" "2" "5" "2" "4" "9"
> a<-as.numeric(a)
> a
[1] 1 2 5 2 4 9
```




```
> t2<-factor(c('b','a','d'),ordered=T,levels=c('a','b','c','d','e'))
> t2
[1] b a d
Levels: a < b < c < d < e
> t2
[1] b a d
Levels: a < b < c < d < e
> t3<-as.character(t2)
> t3
[1] "b" "a" "d"
> t4<-as.numeric(t2)
> t4
[1] 2 1 4
```

```
> t5<-as.factor(t4)
> t5
[1] 2 1 4
Levels: 1 2 4
```

数据类型判断与转换命令

判断	转变
is.numeric()	as.numeric()
is.character()	as.character()
~~is.vector()~~	~~as.vector()~~
is.matrix()	as.matrix()
is.data.frame()	as.data.frame()
is.factor()	as.factor()
is.logical()	as.logical()
is.na()	as.Date()

数据集

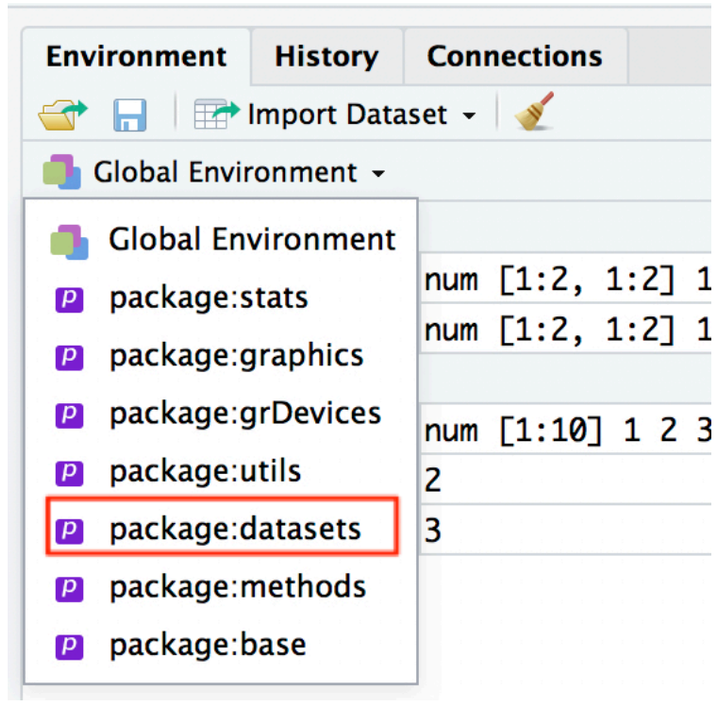
当数据符号集结成大量的集合时，便形成我们使用的主要对象：数据集。之后我们常常称之为数据的东西，实际指数据集。

- 认识几类常见数据集
  - 记录数据集
  - 图形数据集
  - 有序数据集

R的内置数据集

作为数据科学的专用工具，R中也内置了许多各种类型的数据集，这些数据集存在于datasets包环境中。全局环境中并不显示这些数据集，但我们可以直接访问使用。

```
> LifeCycleSavings
                sr pop15 pop75     dpi  ddpi
Australia    11.43 29.35  2.87 2329.68  2.87
Austria      12.07 23.32  4.41 1507.99  3.93
Belgium      13.17 23.80  4.43 2108.47  3.82
Bolivia       5.75 41.89  1.67  189.13  0.22
Brazil       12.88 42.19  0.83  728.47  4.56
```



内置数据集可以到附件1中查阅，今后课程中会大量使用这些数据集作为课堂案例