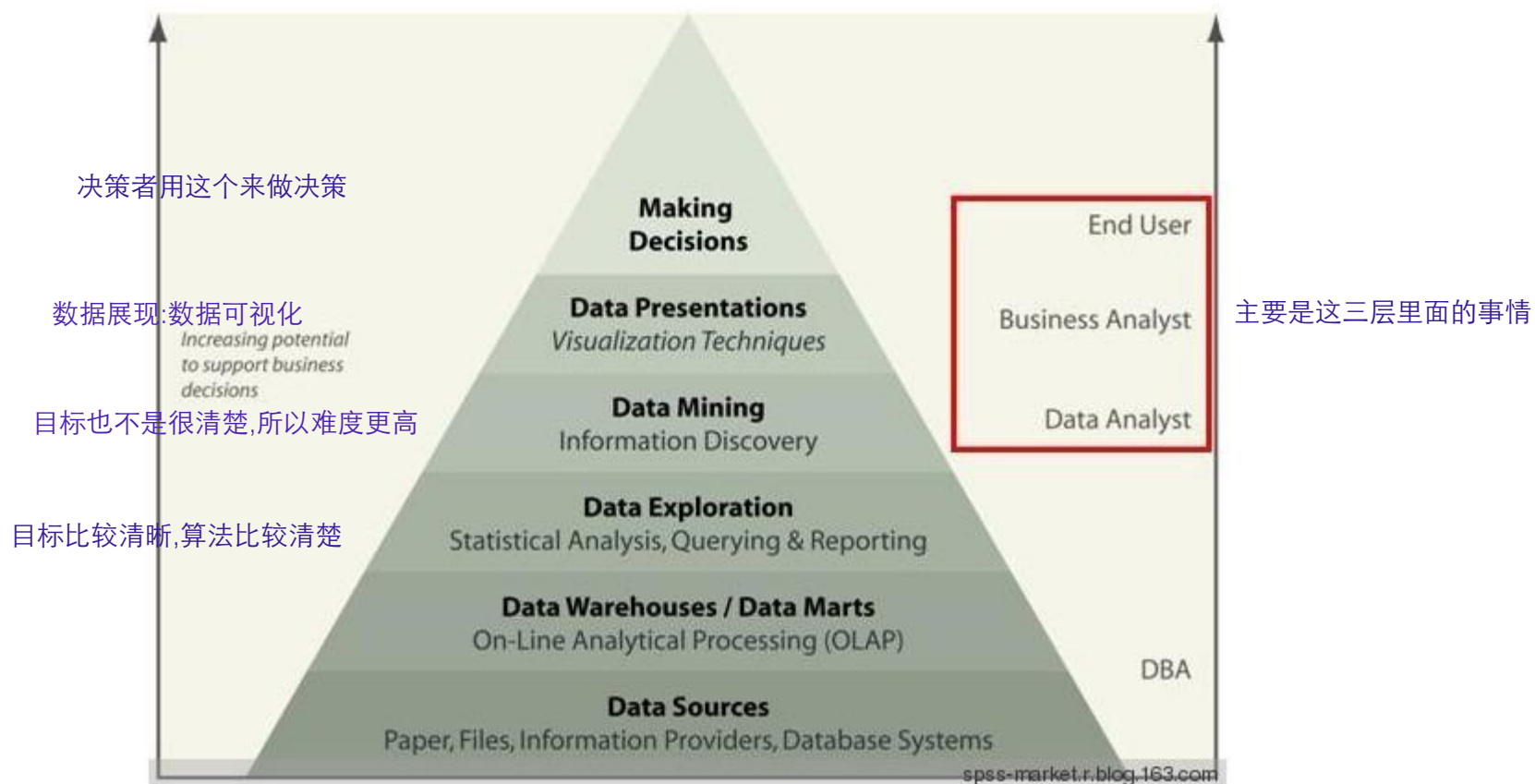




# 数据分析与R语言 第1周

2012.5.3



- 使用统计方法，有目的地对收集到的数据进行分析处理，并且解读分析结果

集中趋势指标	均值(mean)	<ul style="list-style-type: none"> <li>即平均数，<math>mean = 1/n * \sum(X1:Xn)</math>;</li> <li>均值能够利用所有已知信息，但是对异常值(极小或极大值)很敏感;</li> </ul>
	中位数(median)	<ul style="list-style-type: none"> <li>排序后居于中间位置的数值，有序尺度常用;</li> <li>不能充分利用已知的所有变量信息，但不受异常值的影响;</li> </ul>
	众数(mode)	<ul style="list-style-type: none"> <li>出现最频繁的数值，代表分布中的高峰;</li> <li>名义尺度(分组数据)常用</li> </ul>
变异性指标	极差(range)	<ul style="list-style-type: none"> <li>最大值与最小值之差，<math>range = max - min</math>;</li> <li>直接受到异常值影响;</li> </ul>
	方差(variance)	<ul style="list-style-type: none"> <li>离均差(观测值与均值之间的差)平方的均值;</li> <li><math>var = 1/(n-1) * \sum((Xi - mean)^2)</math>;</li> <li>数据分布越分散(远离均值)，方差越大;</li> </ul>
	标准差 (standard deviation)	<ul style="list-style-type: none"> <li>方差的平方根，<math>stdev = \sqrt{var}</math>;</li> <li>与数据本身有相同的量纲，常用;</li> </ul>
变异性指标	偏度(skewness)	<ul style="list-style-type: none"> <li>刻画数据在均值两侧偏差趋势的差异性</li> <li>对称分布: <math>skewness = 0</math>, <math>mean = median = mode</math>;</li> <li>右偏分布: <math>skewness &gt; 0</math>, <math>mean &gt; median &gt; mode</math>;</li> <li>左偏分布: <math>skewness &lt; 0</math>, <math>mean &lt; median &lt; mode</math>;</li> </ul>
	峰度(kurtosis)	<ul style="list-style-type: none"> <li>衡量分布曲线相对平滑或突起程度</li> <li><math>kurtosis = 3</math>，正态分布(Norm distribution);</li> <li><math>kurtosis &gt; 3</math>，分布曲线比正态分布突起;</li> <li><math>kurtosis &lt; 3</math>，分布曲线比正态分布平缓;</li> </ul>

<http://spss-market.r.blog.163.com/>

## ■ 常用算法

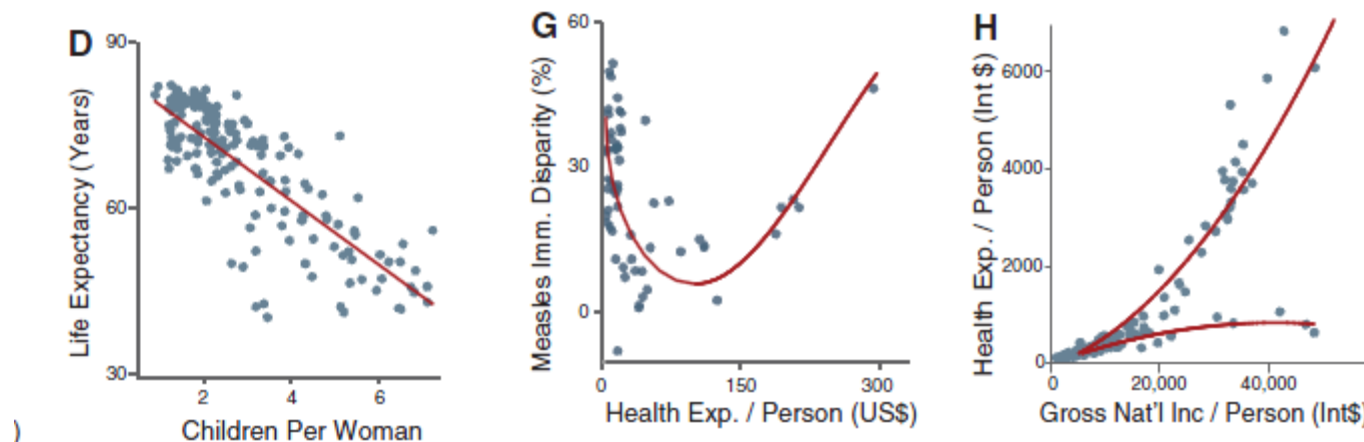


## ■ 数据分析工具



数据分析软件市场排名:  
第一位: R语言(开源)  
第二位: SAS(历史最悠久)  
第三位: SPSS

- 数据挖掘是以查找隐藏在数据中的信息为目标的技术，是应用算法从大型数据库中提取知识的过程，这些算法确定信息项之间的隐性关联，并且向用户显示这些关联
- 数据挖掘思想来源：假设检验，模式识别，人工智能，机器学习
- 常见数据挖掘任务：关联分析，聚类分析，孤立点分析等等
- 例：啤酒与尿布的故事
- 例：《Science》的文章《[科学家摸索出大型数据集内的趋势](#)》



2012.5.3



# 展现层：报表与图形


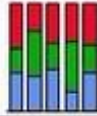

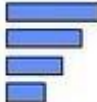





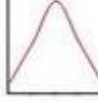
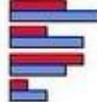
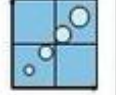


## ■ 老土的报表

人对数字的感觉不敏感,对图形比较敏感

2006年资金预算收支执行情况表																
																单位: 万元
月份	收 入								支 出							
	预算情况				实际情况				预算情况				实际情况			
	经营活动	投资活动	筹资活动	合 计	经营活动	投资活动	筹资活动	合 计	经营活动	投资活动	筹资活动	合 计	经营活动	投资活动	筹资活动	合 计
1月份	2700			2700	3610		0.17	3610.17	5476	2082	50	7608	4961	1175	35	6171
2月份	3800			3800	2420		10.2	2430.2	3809	1244	50	5103	2887	108	54	3049
3月份	4274			4274	5474		11	5485	4526	1496	50	6072	4529	6088	30	10647
4月份	12396			12396	11121	68	2097	13286	5586	1514	50	7150	4246	1230	33	5509
5月份	5311	152		5463	5784	98	94	5976	5841	2431	440	8712	4785	792	432	6009
6月份	3801			3801	1217	15	103	1335	4332	2904	87	7323	4067	1903	33	6003
7月份	5951			5951	4427	65	3593	8085	4085	2591	331	7007	5218	2187	332	7737
8月份	5388			5388	1883		2021	3904	3375	3830	2120	9325	3133	3472	2120	8725
9月份	2830			2830	2459	2	914	3375	3955	2905	93	6953	2800	1469	85	4354
10月份	3250			3250	2855		49	2904	4285	2209	40	6534	3526	1591	39	5156
11月份	3870		700	4570	647		134	781	5873	6036	540	12449	810	3861	540	5211
12月份	4105		2150	6255	7723		2576	10299	7631	3551	88	11270	7065	1838	86	8989
合 计	57676	152	2850	60678	53620	248	11602.37	65470.37	58774	32793	3939	95506	48027	25714	3819	77560

2012.5.3

## ■ 常见的报表

要表达的数据和信息	建议采用图形					
	饼图	垂直柱	水平柱	线图	水泡	其他
整体的一部分						
不同数据的比较						
时间序列						
频率						
两组数据的相关性						
和多重数据、标准相比较						

spss-market.r.blog.163.com

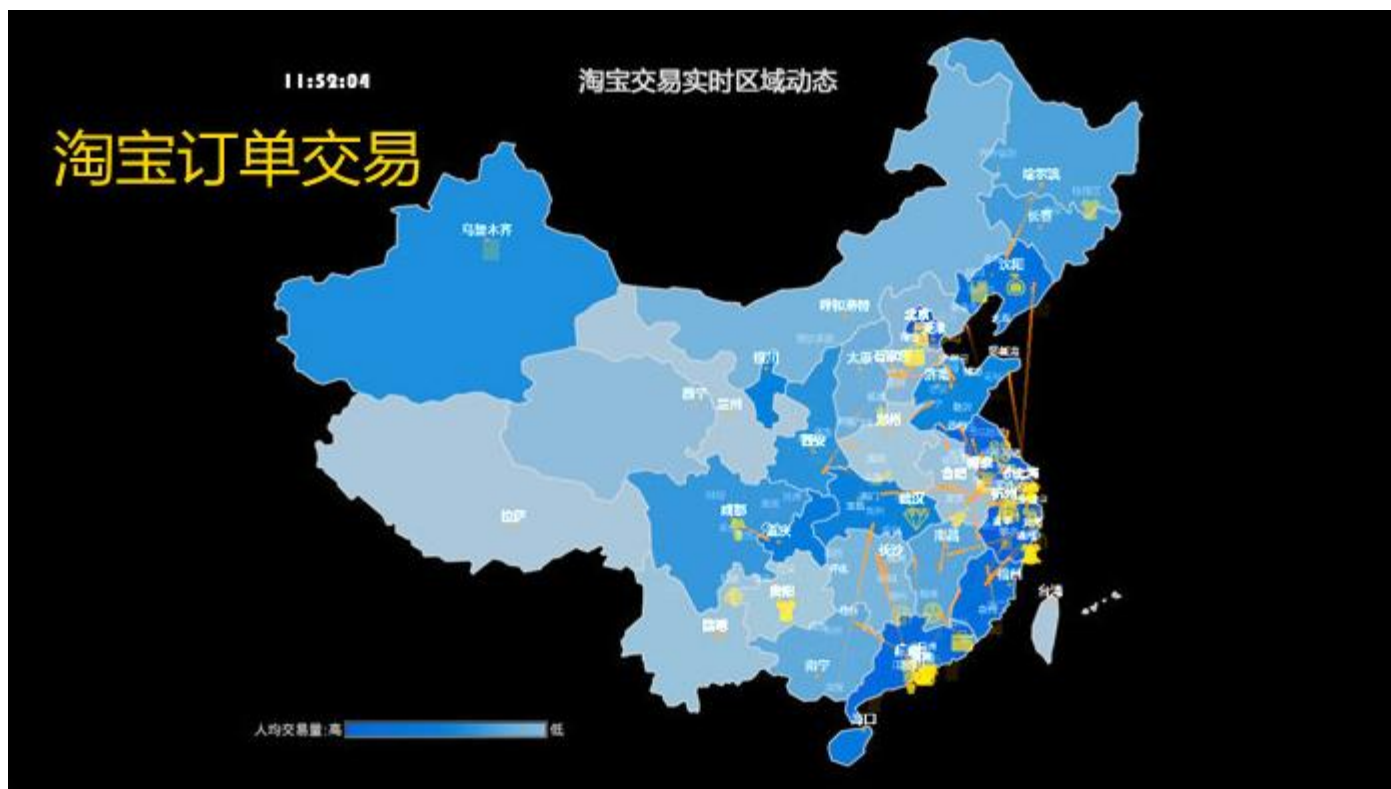


## ■ 仪表盘



## ■ 一些有趣的图表

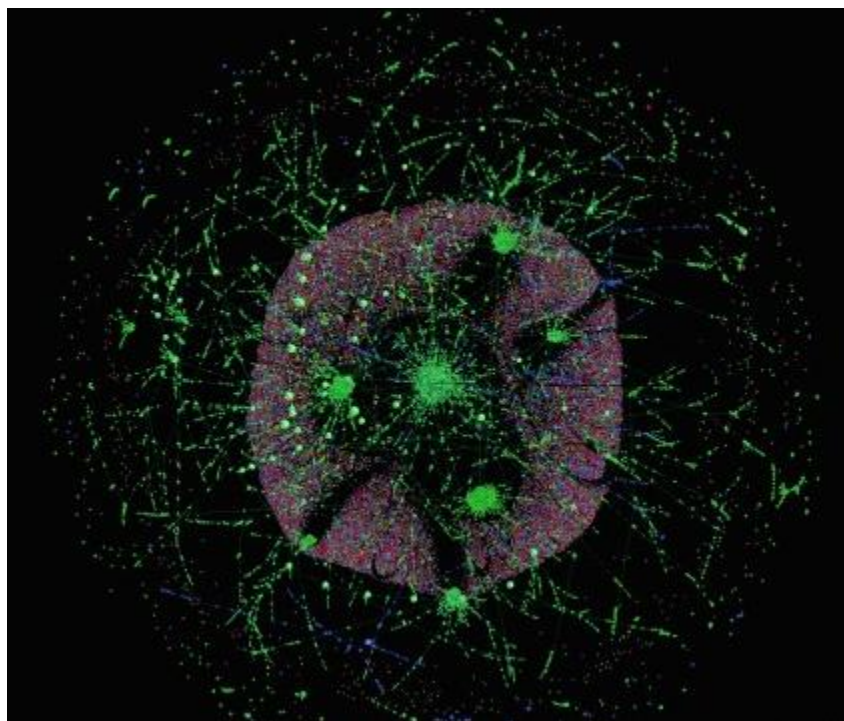
在地图上展现数据的形式  
R里面有专门的地图包



2012.5.3

## ■ 某条微博的扩散路径

R画的社交网络图:信息扩散的情况  
亮点表示人

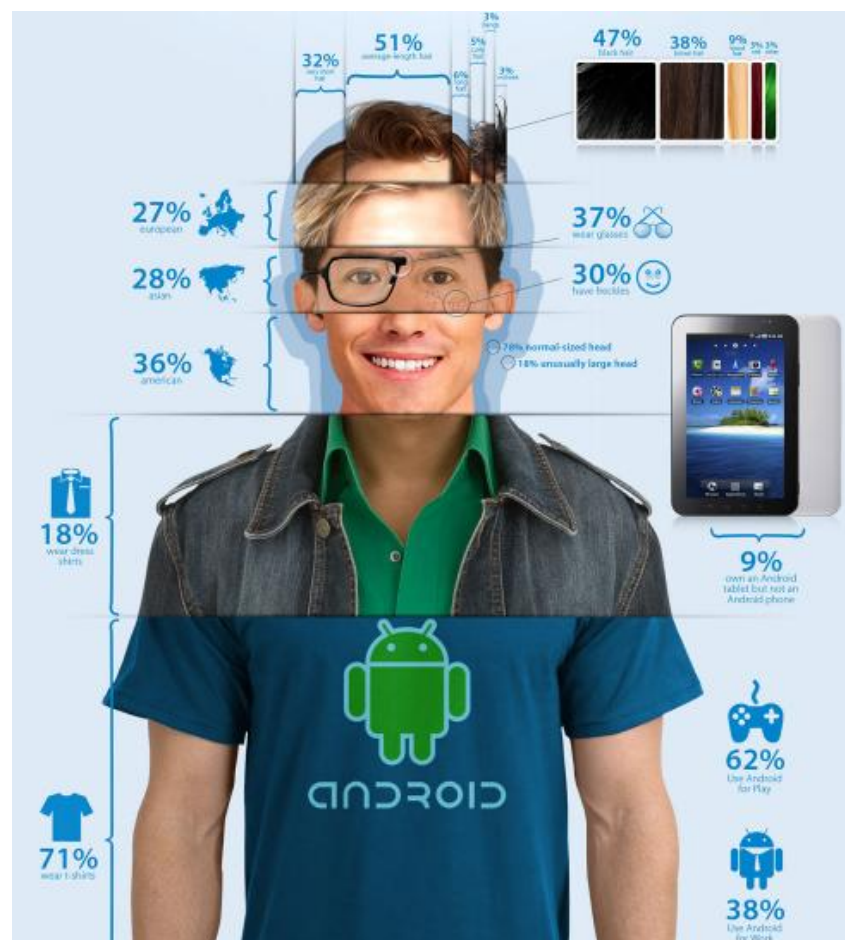


## ■ Mr Android 玩安卓的人的特点的各方面的比例

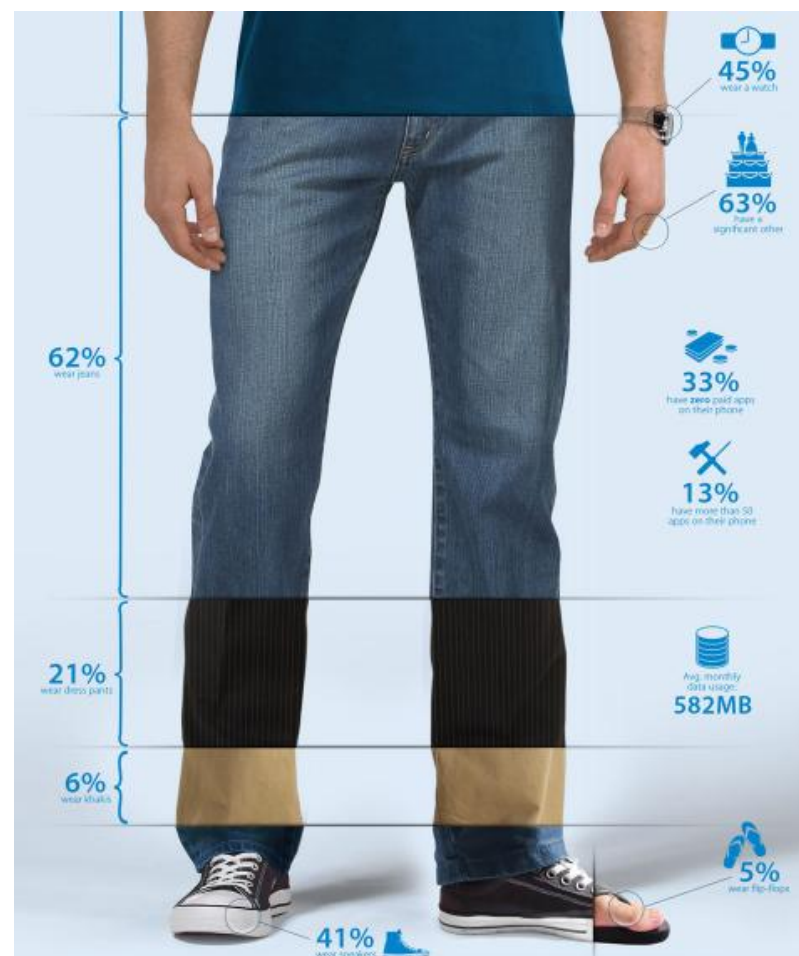
根据信息图显示，Android先生的头发有47%的可能是黑色的，戴眼镜的几率为37%，有36%的可能是北美人，30%的可能脸上长雀斑。71%的时间会穿T恤，下身穿牛仔裤的时间占了62%。工作只占了38%，玩游戏却占了62%，平均每个月会用掉582MB的数据流量。

老板可能会拒绝玩安卓的人

此类图成为信息图,信息图画得好对统计结果的展现有很大帮助



## ■ Mr Android



2012.5.3



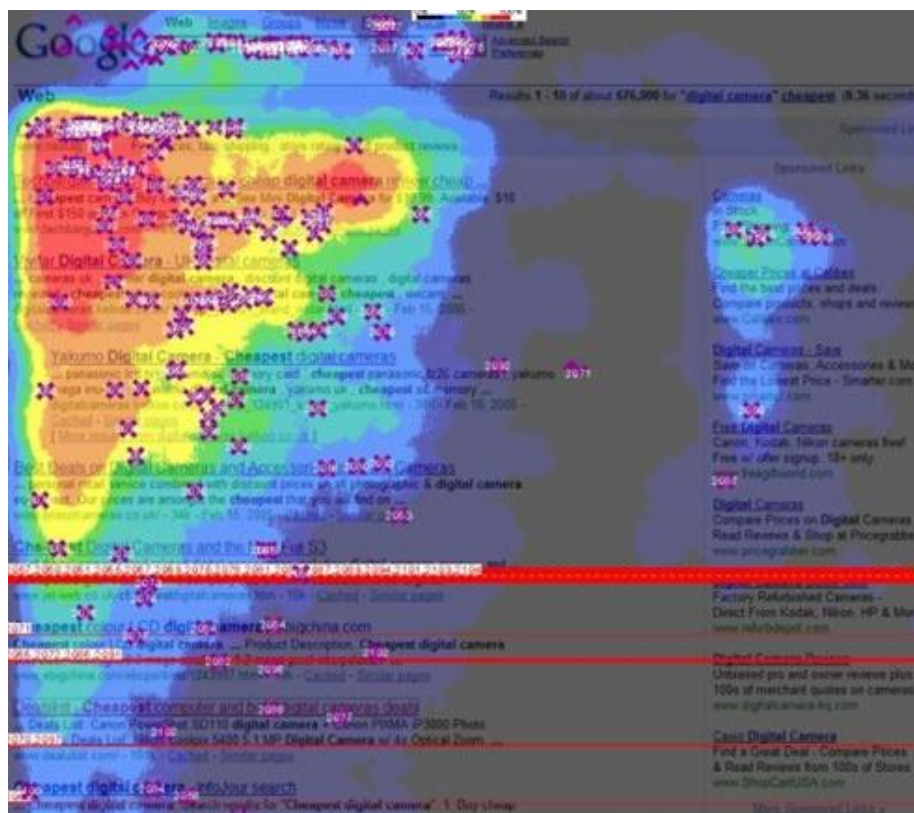
## ■ 网站点击“热力图”

F的形状

先横看一行,然后继续往下看

接下去还会横扫,但是越来越少

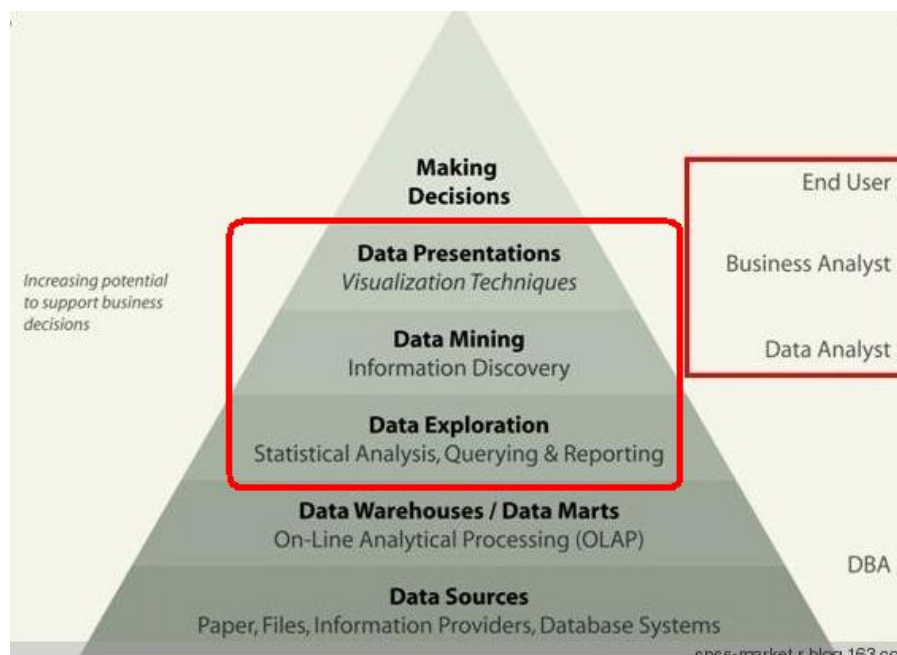
如果想要吸引注意,建议放在左上角的三角区



2012.5.3



- Business Intelligence , 简写为BI
- BI=数据仓库（存储层）+数据分析和数据挖掘（分析层）+报表（展现层）
- 我们课程的位置



## ■ R的源起

S很专业,很好用,但是很昂贵

R是S语言的一种实现。S语言是由 AT&T贝尔实验室开发的一种用来进行数据探索、统计分析、作图的解释型语言。最初S语言的实现版本主要是S-PLUS。S-PLUS是一个商业 软件，它基于S语言，并由 MathSoft公司的统计科学部进一步完善。后来Auckland大学的 Robert Gentleman 和 Ross Ihaka 及其他志愿人员开发了一个R系统。R的使用与S-PLUS有很多类似之处，两个软件有一定的兼容性。

生物学是统计学很大的动力

语法上几乎是一模一样,说明书都可以直接互相交换

## ■ R is free

GNU许可证

R是用于统计分析、绘图的语言和操作环境。R是属于GNU系统的一个自由、免费、源代码开放的软件，它是一个用于统计计算和统计制图的优秀工具。

R是一套完整的数据处理、计算和制图软件系统。其功能包括：数据存储和处理系统；数组运算工具（其向量、矩阵运算方面功能尤其强大）；完整连贯的统计分析工具；优秀的统计制图功能；简便而强大的编程语言：可操纵数据的输入和输出，可实现分支、循环，用户可自定义功能。

R是一个免费的自由软件，它有UNIX、LINUX、MacOS和WINDOWS版本，都是可以免费下载和使用的，在那儿可以下载到R的安装程序、各种外挂程序和文档。在R的安装程序中只包含了8个基础模块，其他外在模块可以通过CRAN获得。

目前共有5000多个包

谁都可以开发，发给R官方，R官方觉得通过了就可以放到官网上去

R官方网站地址：<http://www.r-project.org>

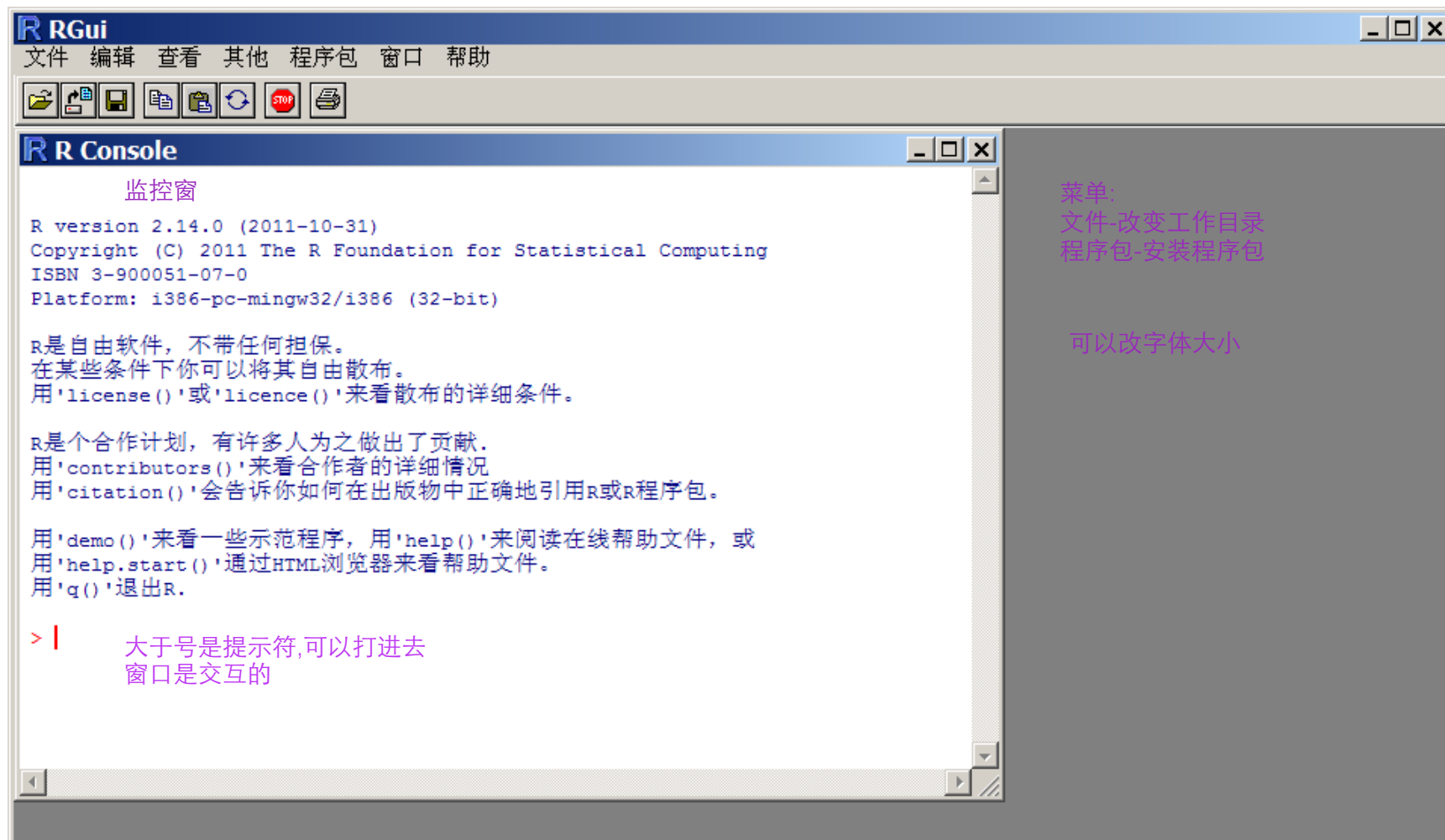
## ■ R的特点

1. 有效的数据处理和保存机制。
2. 拥有一整套数组和矩阵的操作运算符。
3. 一系列连贯而又完整的数据分析中间工具。
4. 图形统计可以对数据直接进行分析和显示，可用于多种图形设备。
5. 一种相当完善、简洁和高效的程序设计语言。它包括条件语句、循环语句、用户自定义的递归函数以及输入输出接口。
6. R语言是彻底面向对象的统计编程语言。
7. R语言和其它编程语言、数据库之间有很好的接口。
8. R语言是自由软件，可以放心大胆地使用，但其功能却不比任何其它同类软件差。
9. R语言具有丰富的网上资源

- 商业版本的R

Revolution R ( 官网 : <http://www.revolutionanalytics.com/> )

很多大型厂商也在开始推出自己的R或兼容R的产品 , 例如Oracle、IBM、Sybase





## 创建向量和矩阵

### ■ 函数c( ), length( ), mode( ), rbind( ), cbind( )

c是用来创建向量,数列

用c把这个向量赋给x1这个向量

```
> x1=c(2,4,6,8,0)
> x2=c(1,3,5,7,9)
> length(x1)
[1] 5
> mode(x1)
[1] "numeric"
> |
```

```
> x1 返回x1的值
[1] 2 4 6 8 0
> x1[3]
[1] 6
> |
```

```
> a1=c(1:100)
> length(a1)
[1] 100
> |
```

按照行把两个向量合成一个矩阵

```
> rbind(x1,x2)
      [,1] [,2] [,3] [,4] [,5]
x1      2    4    6    8    0
x2      1    3    5    7    9
> m1=rbind(x1,x2)
> m1
      [,1] [,2] [,3] [,4] [,5]
x1      2    4    6    8    0
x2      1    3    5    7    9
> |
```

按照列把两个向量合成一个矩阵

```
> cbind(x1,x2)
      x1 x2
[1,]  2  1
[2,]  4  3
[3,]  6  5
[4,]  8  7
[5,]  0  9
> |
```

# 求平均值，和，连乘，最值，方差，标准差

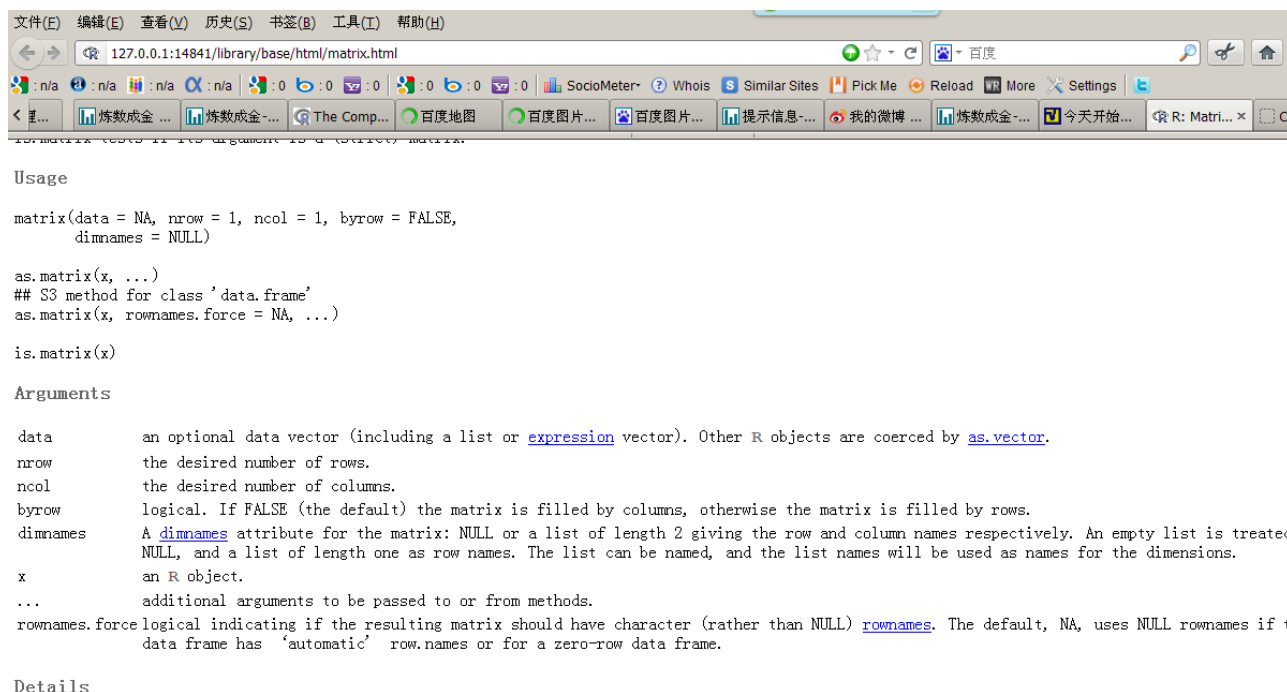
- 函数mean(), sum(), min(), max(), var(), sd(), prod()

```
> x=c(1:100)
> mean(x)
[1] 50.5
> sum(x)
[1] 5050
> max(x)
[1] 100
> min(x)
[1] 1
> var(X)          R是区分大小写的,变量名里面大小写都是敏感的
错误于is.data.frame(x) : 找不到对象'x'
> var(x)
[1] 841.6667
> prod(x)         连乘
[1] 9.332622e+157
> sd(x)
[1] 29.01149
```

## ■ 函数help( ) 非常完善的帮助体系

```
> help(matrix)
starting httpd help server ... done
> |
```

在R里面,等号=也可以写为小于减<-  
都可以用来赋值



```
>
> 1:10
[1] 1 2 3 4 5 6 7 8 9 10
> 1:10-1
[1] 0 1 2 3 4 5 6 7 8 9
> 1:10*2
[1] 2 4 6 8 10 12 14 16 18 20
> 2:60*2+1
[1] 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
[19] 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75
[37] 77 79 81 83 85 87 89 91 93 95 97 99 101 103 105 107 109 111
[55] 113 115 117 119 121
> |
```

```
> a=2:60*2+1
> a
[1] 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
[19] 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75
[37] 77 79 81 83 85 87 89 91 93 95 97 99 101 103 105 107 109 111
[55] 113 115 117 119 121
> a[5]
[1] 13
> a[-5]
[1] 5 7 9 11 15 17 19 21 23 25 27 29 31 33 35 37 39 41
[19] 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77
[37] 79 81 83 85 87 89 91 93 95 97 99 101 103 105 107 109 111 113
[55] 115 117 119 121
```

```
> a[1:5]           显示第一个到第五个数据
[1]  5  7  9 11 13
> a[-(1:5)]        第一个到第五个数据不显示,其他都显示出来
[1] 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
[19] 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85
[37] 87 89 91 93 95 97 99 101 103 105 107 109 111 113 115 117 119 121
> a[1,2,3]         显示a里面的下标第1,2,3,注意一定要有一个c,否则会出错
错误于a[1, 2, 3] : 量度数目不对
> a[c(2,4,7)]
[1]  7 11 17
> a[3:8]
[1]  9 11 13 15 17 19

> a[a<20]
[1]  5  7  9 11 13 15 17 19
> a[a>30 & a<50]
[1] 31 33 35 37 39 41 43 45 47 49
> a[a[3]]
[1] 21
```

## ■ Seq( )函数

```
> seq(5,20)      从5增长到20,步长为1
[1] 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
> seq(5,121,by=2)  参数的写法,都是by=          以2为步长增长,从5增长到121
[1] 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
[19] 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75
[37] 77 79 81 83 85 87 89 91 93 95 97 99 101 103 105 107 109 111
[55] 113 115 117 119 121
> seq(5,121,by=2,length=10)  参数太多,不能又有公差又有数字,互相矛盾
错误于seq.default(5, 121, by = 2, length = 10) : 太多参数
> seq(5,121,length=10)      从5增长到121,总共10项,你自己来算公差是多少,反正是等差数列
[1] 5.00000 17.88889 30.77778 43.66667 56.55556 69.44444 82.33333
[8] 95.22222 108.11111 121.00000
```



- 产生字母序列 letters 是固定向量名,里面是26个字母

```
> letters[1:30]
[1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r"
[19] "s" "t" "u" "v" "w" "x" "y" "z" NA  NA  NA  NA
> |
```

## ■ Which()函数

注意,这里W大写是不对的,应该是小写

```
> a=c(2,3,4,2,5,1,6,3,2,5,8,5,7,3)
> which.max(a)    返回的是 a里面哪个元素最大, 但是返回的是第几个元素,是下标,而不是这个元素
[1] 11
> which.min(a)
[1] 6
> a[which.max(a)]
[1] 8
> which(a==2)
[1] 1 4 9
> a[which(a==2)]
[1] 2 2 2
> which(a>5)
[1] 7 11 13
> a[which(a>5)]
[1] 6 8 7
```

## ■ rev()函数, sort()函数

```
> 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)  逆转顺序
[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)  排序,从小到大
[1] 1 2 2 2 3 3 3 4 5 5 5 6 7 8
> rev(sort(a))  再反过来
[1] 8 7 6 5 5 5 4 3 3 3 2 2 2 1
> |
```

## ■ 函数matrix()

```
> a1=c(1:12) 生成矩阵  
先排满列再排下一列  
> matrix(a1,nrow=3,ncol=4)
```

三行四列

	[,1]	[,2]	[,3]	[,4]
[1,]	1	4	7	10
[2,]	2	5	8	11
[3,]	3	6	9	12

```
> matrix(a1,nrow=4,ncol=3)
```

四行三列

	[,1]	[,2]	[,3]
[1,]	1	5	9
[2,]	2	6	10
[3,]	3	7	11
[4,]	4	8	12

```
> |
```

```
> matrix(a1,nrow=4,ncol=3,byrow=T)
```

使它  
先排满行再排下一行

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6
[3,]	7	8	9
[4,]	10	11	12

```
> |
```

## ■ 函数t( ), 矩阵加减

```
> 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)  矩阵的转置
```

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6
[3,]	7	8	9
[4,]	10	11	12

```
> |
```

```
> a=b=matrix(1:12,nrow=3,ncol=4)
```

```
> a+b  矩阵加法
```

	[,1]	[,2]	[,3]	[,4]
[1,]	2	8	14	20
[2,]	4	10	16	22
[3,]	6	12	18	24

```
> a-b  矩阵减法
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0	0	0	0
[2,]	0	0	0	0
[3,]	0	0	0	0

```
> |
```

## ■ 矩阵相乘，函数diag( )

```
> 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
> |
```

矩阵相乘  
这个好像有点像点乘

```
> a=matrix(1:16,nrow=4,ncol=4)
> a
      [,1] [,2] [,3] [,4]
[1,]     1     5     9    13
[2,]     2     6    10    14
[3,]     3     7    11    15
[4,]     4     8    12    16
> diag(a)  取对角线
[1]  1  6 11 16
> diag(diag(a))  好像又把对角线扩充成矩阵
      [,1] [,2] [,3] [,4]
[1,]     1     0     0     0
[2,]     0     6     0     0
[3,]     0     0    11     0
[4,]     0     0     0    16
> diag(4)  生成对角线为4个1的方阵
      [,1] [,2] [,3] [,4]
[1,]     1     0     0     0
[2,]     0     1     0     0
[3,]     0     0     1     0
[4,]     0     0     0     1
> |
```



## ■ 矩阵求逆，函数rnorm( ), solve( )

生成16个r的正态分布随机数,然后按照四行四列的矩阵

```
> a=matrix(rnorm(16),4,4)
> a
      [,1]      [,2]      [,3]      [,4]
[1,] 0.60714591 0.9354156 0.6471921 1.7788818
[2,] 0.03972303 -0.4784529 0.1773237 0.1755301
[3,] -1.59620992 -0.4553338 2.1706594 1.3569393
[4,] -0.56335648 -1.2811563 1.7136756 -1.2032154
> solve(a) 忘了逆矩阵了,好像是乘起来等于1
      [,1]      [,2]      [,3]      [,4]
[1,] 0.58375607 0.6422022 -0.51939339 0.37098393
[2,] 0.23718393 -1.8782607 0.01174602 0.08990033
[3,] 0.42730886 -0.5205396 -0.03332176 0.51823304
[4,] 0.08272524 0.9578674 0.18321945 -0.36243660
> |
```

## ■ 函数solve(a,b)

```
> a=matrix(rnorm(16),4,4)
> a
      [,1]      [,2]      [,3]      [,4]
[1,] 0.09502486 -0.2002975 -0.9340249  1.067134
[2,] 0.91382126 -0.8181392  0.8628442 -2.094286
[3,] -1.32881330 -0.5173477 -0.9182241 -1.635026
[4,] 0.28637823  0.6505220 -0.0399500 -1.469619
> b=c(1:4)
> b
[1] 1 2 3 4
> solve(a,b)
      [,1]      [,2]      [,3]      [,4]
[1,] 1.8470707  0.9692533 -3.1994782 -1.8458519
. . .
```

a乘以多少等于b,求解

# 矩阵的特征值与特征向量

## ■ 函数eigen()

```
> a=diag(4)+1
```

 生成了对角线为4个1的方阵,然后都加1

```
> a
```

```
      [,1] [,2] [,3] [,4]
[1,]     2     1     1     1
[2,]     1     2     1     1
[3,]     1     1     2     1
[4,]     1     1     1     2
```

```
> a.e=eigen(a,symmetric=T)
```

```
> a.e
```

```
$values
```

```
[1] 5 1 1 1
```

```
> a.e$vectors%*%diag(a.e$values)%*%t(a.e$vectors)
```

```
      [,1] [,2] [,3] [,4]
[1,]     2     1     1     1
[2,]     1     2     1     1
[3,]     1     1     2     1
[4,]     1     1     1     2
```

```
$vectors
```

```
      [,1]      [,2]      [,3]      [,4]
[1,] -0.5  0.8660254  0.0000000e+00  0.0000000
[2,] -0.5 -0.2886751 -6.408849e-17  0.8164966
[3,] -0.5 -0.2886751 -7.071068e-01 -0.4082483
[4,] -0.5 -0.2886751  7.071068e-01 -0.4082483
```

# 数据的R语言表示——数据框

- 矩阵形式，但列可以不同数据类型

矩阵里面一定全都是数值  
但是数据框不一定,可以有一列是数值,另外一列是字符  
每一列是观测值

- 每列是一个变量，每行是一个观测值

每列是变量(或属性),每行是观测值(指某一个同学)

```
> x1=c(10,13,45,26,23,12,24,78,23,43,31,56)
```

```
> x2=c(20,65,32,32,27,87,60,13,42,51,77,35)
```

```
> x=data.frame(x1,x2) 关键是两个向量的长度要相等
```

```
> x 用data.frame处理出来了以后,赋值给x中,x就是一个数据框
```

	x1	x2
1	10	20
2	13	65
3	45	32
4	26	32
5	23	27
6	12	87
7	24	60
8	78	13
9	23	42
10	43	51
11	31	77
12	56	35

```
> |
```

```
> (x=data.frame('重量'=x1,'运费'=x2))
```

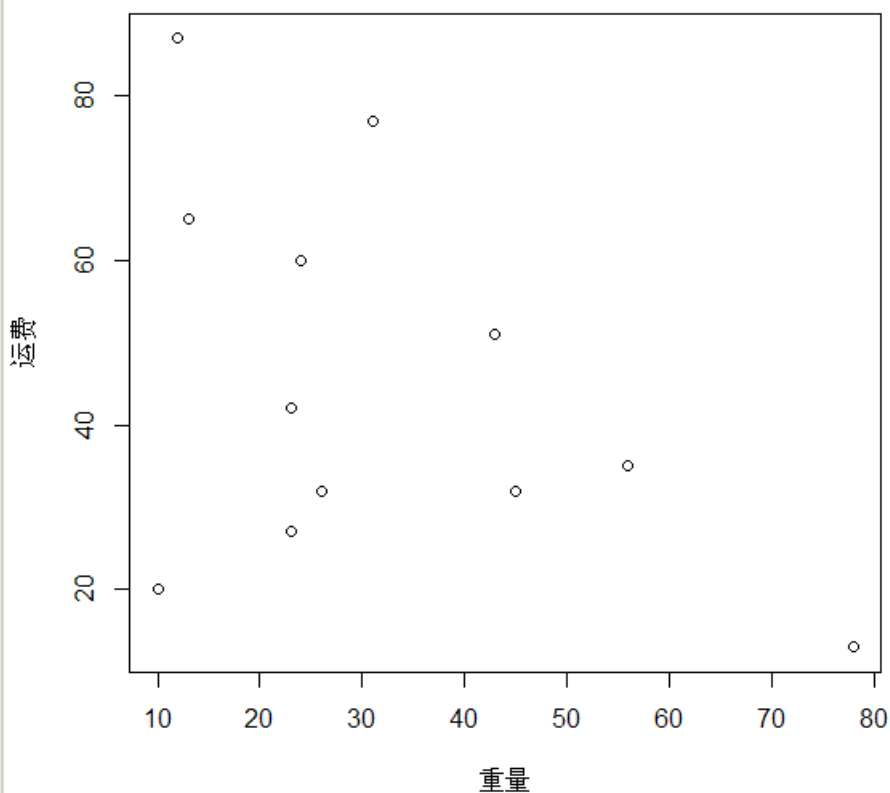
	重量	运费
--	----	----

1	10	20
2	13	65
3	45	32
4	26	32
5	23	27
6	12	87
7	24	60
8	78	13
9	23	42
10	43	51
11	31	77
12	56	35

```
> |
```

## ■ 函数plot()

```
> plot(x)  
> |
```



- 先设置工作目录，把文本文件放于该目录下

把文本文件读取到数据框里面

```
> (x=read.table("abc.txt"))  
      V1 V2  
1    175 67  
2    183 75  
3    165 56  
4    145 45  
5    178 67  
6    187 90  
7    156 43  
8    176 58  
9    173 60  
10   170 56
```

- 文本或excel的数据均可通过剪贴板操作

```
> y<-read.table("clipboard",header=F)
```

```
> y
```

```
      V1 V2
1    175 67
2    183 75
3    165 56
4    145 45
5    178 67
6    187 90
7    156 43
8    176 58
9    173 60
10   170 56
```

```
> |
```

```
> z<-read.table("clipboard",header=T)
```

```
> z
```

```
 商品 价格
1    A    2
2    B    3
3    C    5
4    D    5
```

```
> |
```



- 方法1：先把excel另存为空格分隔的prn文本格式再读

```
> w<-read.table("test.prn",header=T)
> w
  商品  价格
1    A    2
2    B    3
3    C    5
4    D    5
> |
```

## ■ 方法2：安装RODBC包，再通过ODBC读

```
> local({pkg <- select.list(sort(.packages(all.available = TRUE)), graphics=T$  
+ if(nchar(pkg)) library(pkg, character.only=TRUE))})
```

警告信息：

程辑包‘RODBC’是用R版本2.14.1 来建造的

```
> library(RODBC)
```

```
> z<-odbcConnectExcel("test.xls")
```

```
> (w<-sqlFetch(z, "Sheet1"))
```

	商品	价格
--	----	----

1	A	2
---	---	---

2	B	3
---	---	---

3	C	5
---	---	---

4	D	5
---	---	---

```
> |
```

## ■ for语句

```
> for (i in 1:59) {a[i]=i*2+3}
> a
[1] 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
[19] 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75
[37] 77 79 81 83 85 87 89 91 93 95 97 99 101 103 105 107 109 111
[55] 113 115 117 119 121
>
> for (i in 1:59) {a[i]=i*2+3;b[i]=i*5-4}
错误于b[i] = i * 5 - 4 : 找不到对象'b'
> b=0
> for (i in 1:59) {a[i]=i*2+3;b[i]=i*5-4}
> b
[1] 1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86
[19] 91 96 101 106 111 116 121 126 131 136 141 146 151 156 161 166 171 176
[37] 181 186 191 196 201 206 211 216 221 226 231 236 241 246 251 256 261 266
[55] 271 276 281 286 291
```

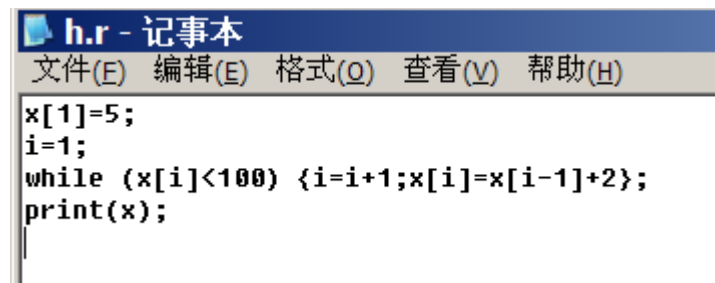
## ■ while语句

```
> a[1]=5
> i=1
> while (a[i]<121) {i=i+1;a[i]=a[i-1]+2}
> a
```

必须要满足这个条件才会进入循环体进行运行

[1]	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
[19]	41	43	45	47	49	51	53	55	57	59	61	63	65	67	69	71	73	75
[37]	77	79	81	83	85	87	89	91	93	95	97	99	101	103	105	107	109	111
[55]	113	115	117	119	121													

- `source()`函数 读取脚本,一般都是.r来运行
- `print()`函数 脚本里面如果要显示一个变量的话,不能只打一个x,应该用print函数



```
h.r - 记事本
文件(E) 编辑(E) 格式(O) 查看(V) 帮助(H)
x[1]=5;
i=1;
while (x[i]<100) {i=i+1;x[i]=x[i-1]+2};
print(x);
```

```
> source("D:\\h.r")
 [1] 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
[19] 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75
[37] 77 79 81 83 85 87 89 91 93 95 97 99 101
```

- 模拟产生统计专业同学的名单（学号区分），记录数学分析，线性代数，概率统计三科成绩，然后进行一些统计分析

```
> num=seq(10378001,10378100) 共100个元素,代表100个不同的学号
> num
[1] 10378001 10378002 10378003 10378004 10378005 10378006 10378007 10378008
[9] 10378009 10378010 10378011 10378012 10378013 10378014 10378015 10378016
[17] 10378017 10378018 10378019 10378020 10378021 10378022 10378023 10378024
[25] 10378025 10378026 10378027 10378028 10378029 10378030 10378031 10378032
[33] 10378033 10378034 10378035 10378036 10378037 10378038 10378039 10378040
[41] 10378041 10378042 10378043 10378044 10378045 10378046 10378047 10378048
[49] 10378049 10378050 10378051 10378052 10378053 10378054 10378055 10378056
[57] 10378057 10378058 10378059 10378060 10378061 10378062 10378063 10378064
[65] 10378065 10378066 10378067 10378068 10378069 10378070 10378071 10378072
[73] 10378073 10378074 10378075 10378076 10378077 10378078 10378079 10378080
[81] 10378081 10378082 10378083 10378084 10378085 10378086 10378087 10378088
[89] 10378089 10378090 10378091 10378092 10378093 10378094 10378095 10378096
[97] 10378097 10378098 10378099 10378100
```

# 分布函数

- 正态分布函数rnorm( )
- 泊松分布函数rpois( )
- 指数分布函数rexp( )
- Gamma分布函数rgamma( )
- 均匀分布函数runif( )
- 二项分布函数rbinom( )
- 几何分布函数rgeom( )



## ■ 用runif和rnorm

```
> x1=round(runif(100,min=80,max=100))
```

用runif产生100个均匀分布的随机数,下限是80,上限是100  
round可以使之四舍五入

```
> x1
```

[1]	95	97	88	82	95	85	81	81	91	99	84	95	89	92	89	93	96	87
[19]	90	81	94	94	88	91	90	90	97	92	91	97	96	93	80	93	86	89
[37]	81	87	86	85	89	92	84	91	92	86	91	85	96	96	83	99	80	97
[55]	88	98	85	97	94	99	82	89	96	85	80	88	93	97	97	91	100	89
[73]	98	86	97	88	88	95	99	83	96	85	95	88	88	91	90	85	84	86
[91]	94	87	99	93	89	87	95	89	84	81								

```
> |
```

```
> x2=round(rnorm(100,mean=80,sd=7))
```

```
> x2
```

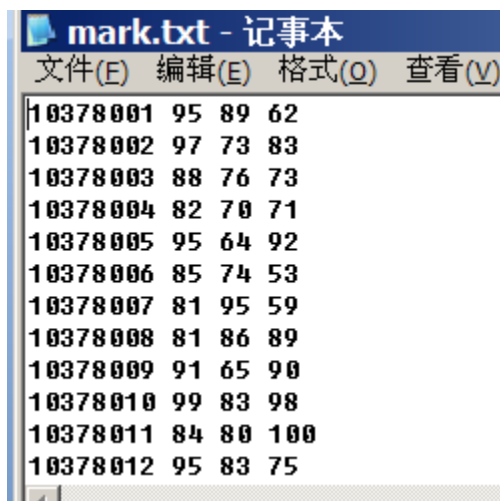
[1]	89	73	76	70	64	74	95	86	65	83	80	83	71	72	83	79	91	70	75	84	72	94	74	81
[25]	81	74	85	96	69	84	73	93	72	76	73	81	76	92	73	81	88	80	87	81	81	66	76	85
[49]	97	77	83	77	82	86	76	69	83	83	77	79	84	90	82	81	81	79	76	90	80	83	88	93
[73]	75	83	89	93	69	97	85	85	93	73	73	79	75	64	81	81	55	63	81	80	84	78	88	75
[97]	75	78	78	87																				

```
> |
```

```
> x3=round(rnorm(100,mean=83,sd=18))
> x3
 [1] 62 83 73 71 92 53 59 89 90 98 123 75 107 108 69 73 110 61
[19] 88 83 76 96 81 56 41 70 64 78 80 61 94 108 77 91 83 93
[37] 66 64 56 87 97 92 99 82 45 93 86 77 82 75 69 94 75 98
[55] 75 65 63 75 88 79 80 104 88 94 92 77 63 97 87 85 89 58
[73] 83 84 93 64 109 115 104 87 78 58 74 67 120 66 64 80 72 88
[91] 86 97 97 114 89 41 104 76 70 81
> x3[which(x3>100)]=100 把超过100分的人,变成100分
> x3 因为正态分布产生的数据没法指定上下限
 [1] 62 83 73 71 92 53 59 89 90 98 100 75 100 100 69 73 100 61
[19] 88 83 76 96 81 56 41 70 64 78 80 61 94 100 77 91 83 93
[37] 66 64 56 87 97 92 99 82 45 93 86 77 82 75 69 94 75 98
[55] 75 65 63 75 88 79 80 100 88 94 92 77 63 97 87 85 89 58
[73] 83 84 93 64 100 100 100 87 78 58 74 67 100 66 64 80 72 88
[91] 86 97 97 100 89 41 100 76 70 81
> |
```

# 合成数据框并保存到硬盘

- data.frame() 用data.frame这个函数把学号和三门成绩合成一个数据框
- write.table 用write.table这个函数把数据框写入文件里去



10378001	95	89	62
10378002	97	73	83
10378003	88	76	73
10378004	82	70	71
10378005	95	64	92
10378006	85	74	53
10378007	81	95	59
10378008	81	86	89
10378009	91	65	90
10378010	99	83	98
10378011	84	80	100
10378012	95	83	75

```
> x=data.frame(num,x1,x2,x3)  
> x
```

	num	x1	x2	x3
1	10378001	95	89	62
2	10378002	97	73	83
3	10378003	88	76	73
4	10378004	82	70	71
5	10378005	95	64	92
6	10378006	85	74	53
7	10378007	81	95	59
8	10378008	81	86	89
9	10378009	91	65	90
10	10378010	99	83	98
11	10378011	84	80	100
12	10378012	95	83	75
13	10378013	89	71	100

```
> write.table(x,file="d:\\mark.txt",col.names=F,row.names=F,sep=" ")  
> |
```

# 计算各科平均分

## ■ 函数mean(), colMeans(), apply()

算平均值

是指对列也求平均值

apply函数的意思:对x这个数据框,在行或者列的方向上进行相应的操作  
行是1,列是2 比如mean

```
> mean(x)
      num      x1      x2      x3
10378050.50 90.19 80.00 80.47
警告信息: 把学号也求了平均值
mean(<data.frame>) is deprecated.
Use colMeans() or sapply(*, mean) instead.
> colMeans(x)
      num      x1      x2      x3
10378050.50 90.19 80.00 80.47
> colMeans(x)[c("x1", "x2", "x3")]
      x1      x2      x3
90.19 80.00 80.47
> apply(x, 2, mean)
      num      x1      x2      x3
10378050.50 90.19 80.00 80.47
> |
```

## 求各科最高最低分

### ■ 函数max( ),min( ),apply( )

```
> apply(x, 2, max)
      num      x1      x2      x3
10378100    100     97    100
> apply(x, 2, min)
      num      x1      x2      x3
10378001     80     55     41
> |
```

## 求出每人总分

```
> apply(x[c("x1","x2","x3")],1,sum)
```

```
[1] 246 253 237 223 251 212 235 256 246 280 264 253 260 264 241 245 287 218  
[19] 253 248 242 284 243 228 212 234 246 266 240 242 263 286 229 260 242 263  
[37] 223 243 215 253 274 264 270 254 218 245 253 247 275 248 235 270 237 281  
[55] 239 232 231 255 259 257 246 279 266 260 253 244 232 284 264 259 277 240  
[73] 256 253 279 245 257 292 284 255 267 216 242 234 263 221 235 246 211 237  
[91] 261 264 280 271 266 203 270 243 232 249
```



# Thanks

## FAQ时间