DATAGURU 炼数成金


# 数据分析、展现与R语言 第5周

2012.10.18

# 法律声明

【声明】本视频和幻灯片为炼数成金网络课程的教学资料，所有资料只能在课程内使用，不得在课程以外范围散播，违者将可能被追究法律和经济责任。

**课程详情访问炼数成金培训网站**

**http://edu.dataguru.cn**

# 传统表格

- 二维结构
- 数字与文字为主
- 缺乏润色

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 | | | |
| | 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 | 2100 | | | 2100 | 1610 | | 0.17 | 1610.17 |
| 2月份 | 1800 | | | 1800 | 2430 | | 10.3 | 2430.3 |
| 3月份 | 4274 | | | 4274 | 3474 | | 10 | 3483 |
| 4月份 | 12284 | | | 12284 | 11121 | 60 | 2807 | 13984 |
| 5月份 | 1321 | 132 | | 3462 | 1784 | 86 | 84 | 1974 |
| 6月份 | 2801 | | | 2801 | 1217 | 11 | 101 | 1333 |
| 7月份 | 1911 | | | 1911 | 4927 | 43 | 1101 | 6081 |
| 8月份 | 1388 | | | 1388 | 1881 | | 2021 | 2084 |
| 9月份 | 1950 | | | 1950 | 1419 | 7 | 914 | 1271 |
| 10月份 | 1210 | | | 1210 | 1911 | | 93 | 2004 |
| 11月份 | 1870 | | 700 | 4170 | 447 | | 134 | 781 |
| 12月份 | 4101 | | 2110 | 6211 | 1713 | | 2174 | 11099 |
| 合计 | 27674 | 132 | 2810 | 40678 | 23635 | 248 | 11682.17 | 41418.17 |

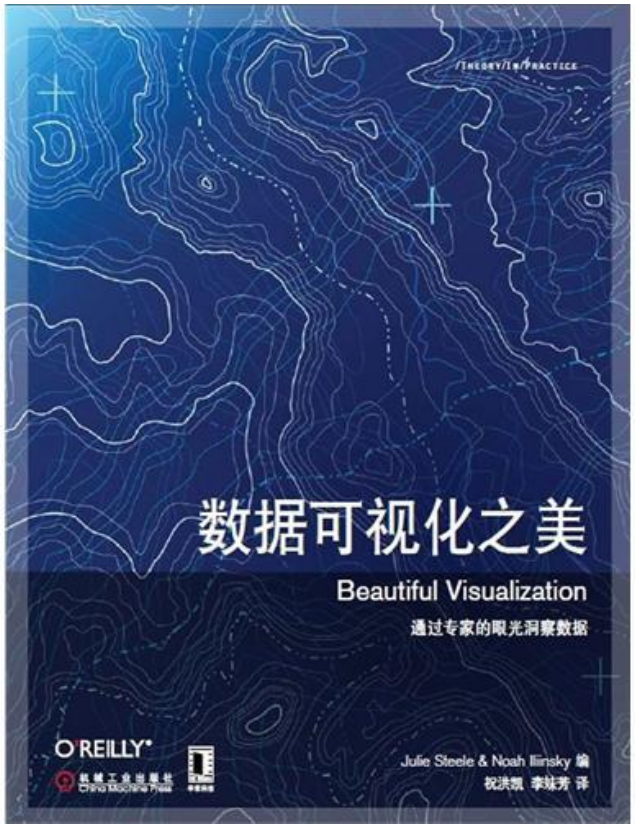
| 月份 | 支出 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 预算情况 | | | | 实际情况 | | | |
| | 经营活动 | 投资活动 | 筹资活动 | 合计 | 经营活动 | 投资活动 | 筹资活动 | 合计 |
| 1月份 | 1474 | 2082 | 19 | 7408 | 4961 | 1171 | 24 | 6177 |
| 2月份 | 1819 | 1244 | 19 | 3182 | 1887 | 108 | 24 | 2049 |
| 3月份 | 4174 | 1496 | 10 | 6077 | 4123 | 6088 | 10 | 10447 |
| 4月份 | 3184 | 1114 | 19 | 7120 | 4994 | 1210 | 11 | 6189 |
| 5月份 | 1841 | 2431 | 440 | 8713 | 4781 | 781 | 422 | 6809 |
| 6月份 | 4112 | 2904 | 87 | 7123 | 4867 | 1981 | 23 | 6851 |
| 7月份 | 4081 | 2191 | 121 | 7581 | 1218 | 2187 | 122 | 7127 |
| 8月份 | 1221 | 2830 | 2120 | 8121 | 1121 | 3472 | 2120 | 8721 |
| 9月份 | 1911 | 2581 | 91 | 6811 | 1859 | 1469 | 81 | 4214 |
| 10月份 | 4081 | 1199 | 49 | 6124 | 1124 | 1281 | 19 | 2124 |
| 11月份 | 1871 | 6824 | 240 | 12049 | 810 | 2864 | 149 | 6211 |
| 12月份 | 7421 | 2121 | 88 | 11210 | 7861 | 1828 | 84 | 8988 |
| 合计 | 38774 | 32753 | 3519 | 93396 | 48827 | 31714 | 3819 | 77160 |

# 现代信息图

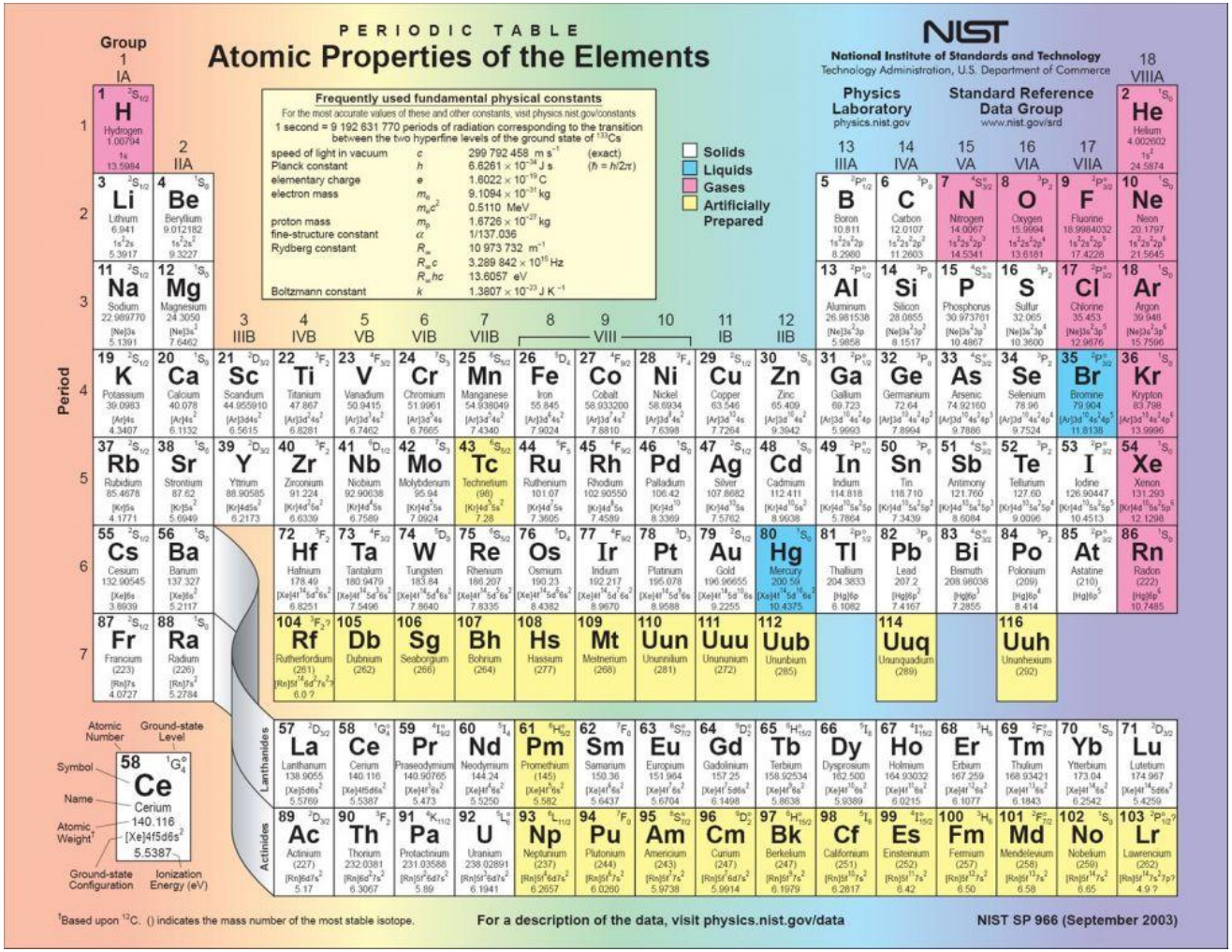
- 以人眼敏感的视觉元素为主
- 信息高度密集

# 何为美？

- 新颖
- 充实
- 高效
- 美感



# 学习经典：元素周期表

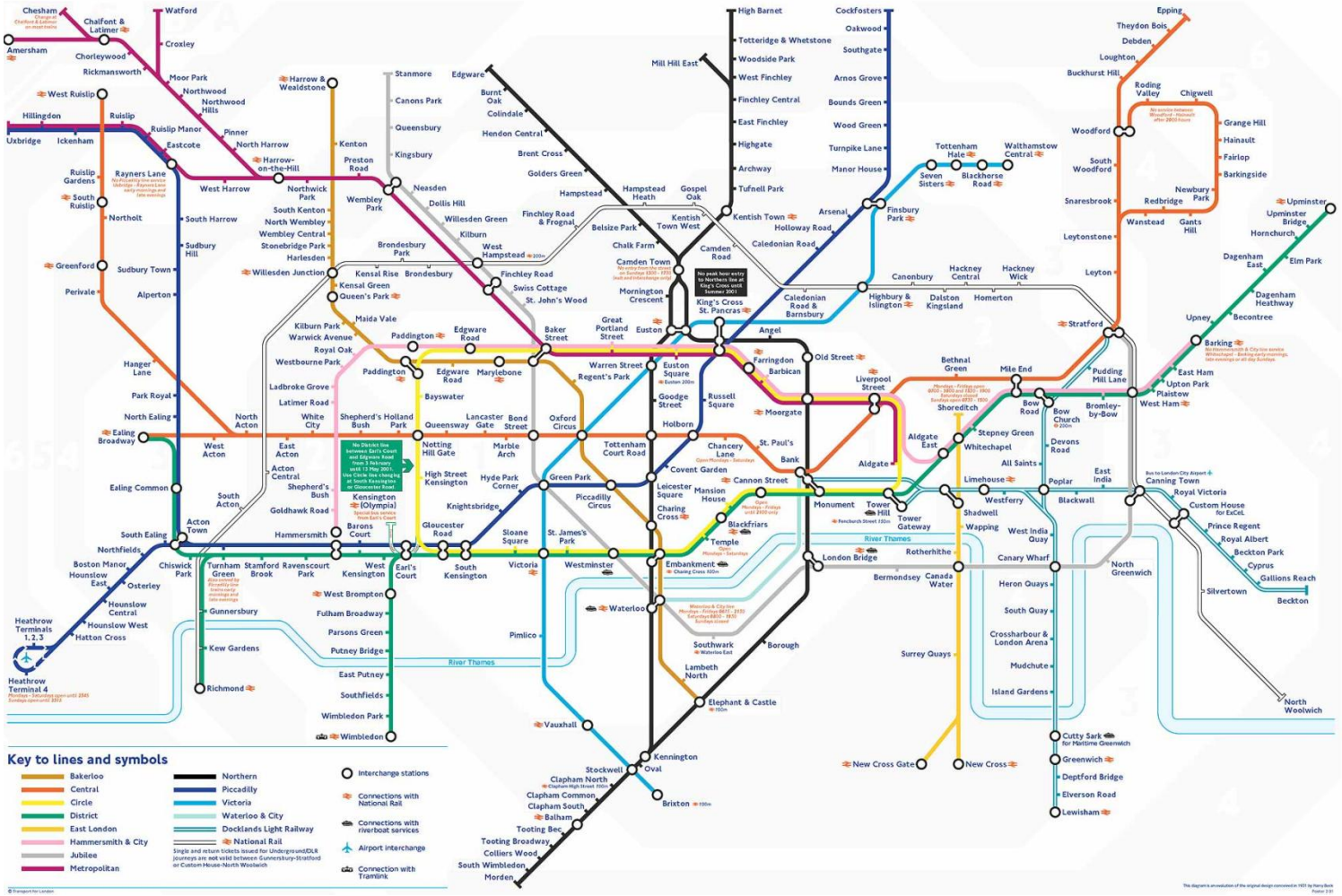



2012.10.18

# 学习经典：元素周期表

- 元素周期表的天才之处：通过元素的编排组织揭示了元素之间的相互关系以及周期变化的物理属性
- 蕴含巨大信息量，几乎就是半部化学
- 复杂数据可视化的早期杰作

# 学习经典：伦敦地铁图




2012.10.18

# 学习经典：伦敦地铁图

- Harry Beck的杰作，被收藏在伦敦交通博物馆
- 作者习惯于画电路草图，因此把绘制电路图的习惯带到地铁图中，例如45度和90度的直线段布局
- 把信息从具体的精确地理位置解放出来，突出了人们和系统里其它位置的逻辑关系。突出显示了最相关的信息，删除了很多不相关的信息
- 公认的杰作，有大量仿制品

# 学习经典：美国竞选结果图

- http://elections.nytimes.com/2008/president/whos-ahead/key-states/map.html

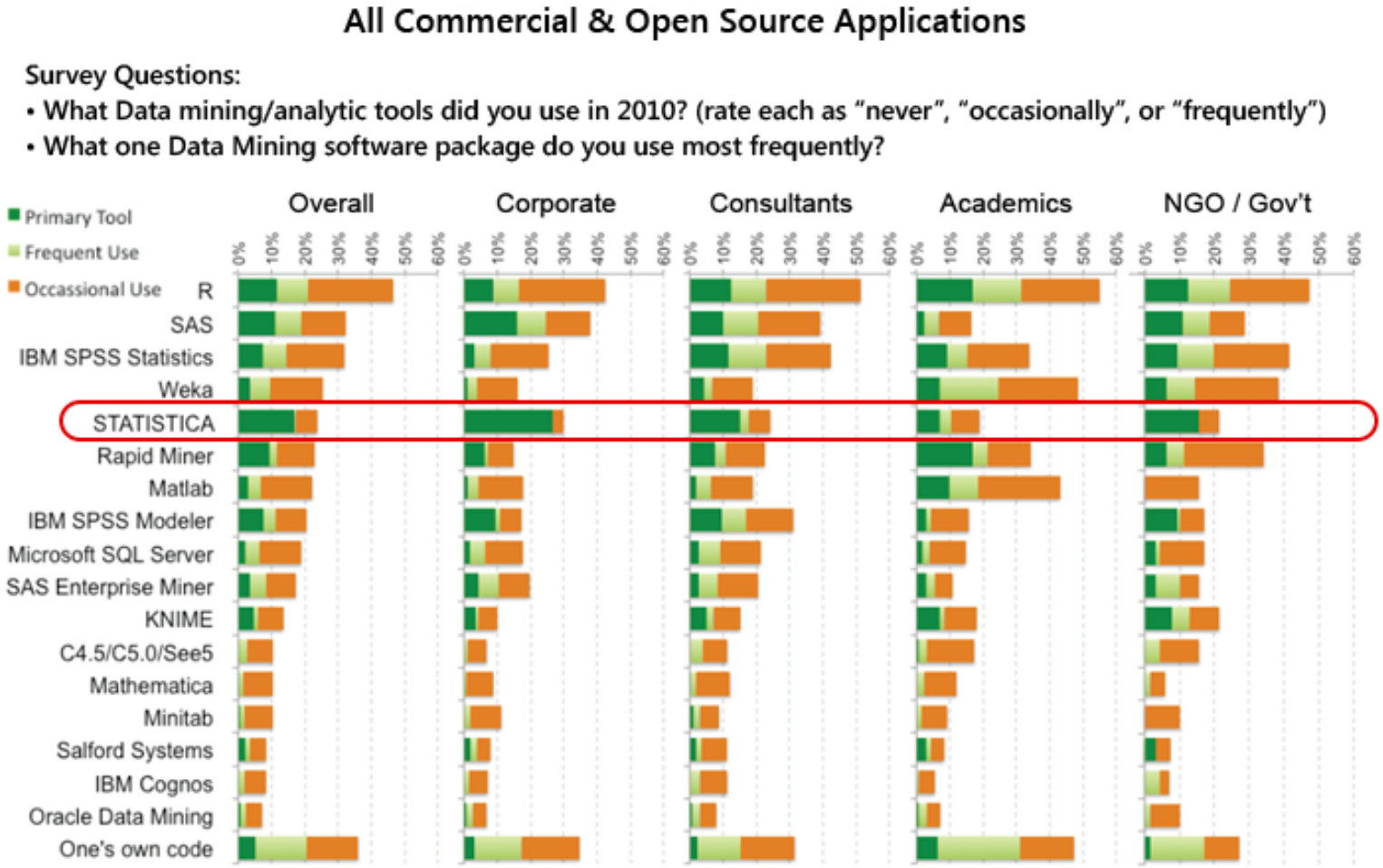
# 创建有效可视化的步骤

- 制定问题：所要讲述的故事的主题是什么？故事的主要情节是什么？
- 收集数据：原始数据，数据分析，结果的解析、组织、分组
- 应用一种可视化的形式：基本几何要素包括尺寸、彩色、位置、连线组成网络

# 问题+可视化数据+场景=故事

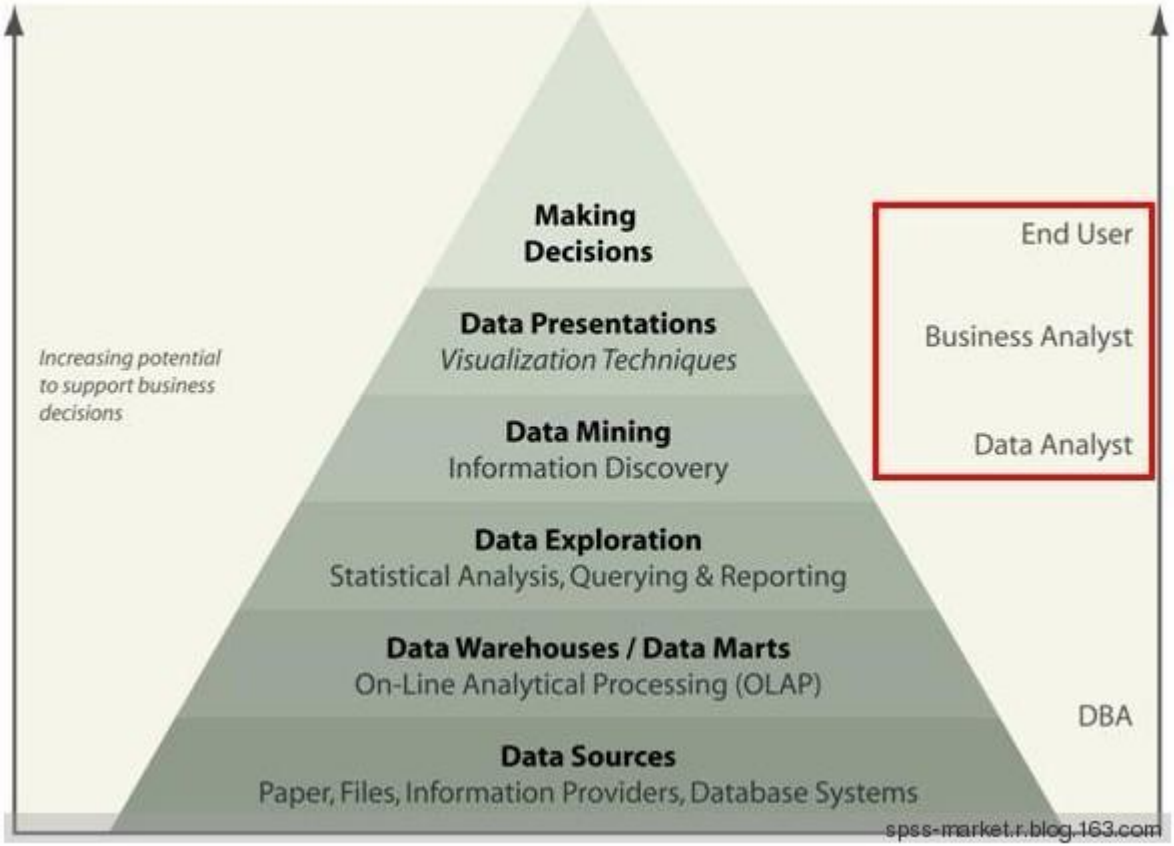
- Facebook的故事

# R、SAS的那些事儿




2012.10.18

# 数据体系金字塔


Making Decisions
Data Presentations
Visualization Techniques
Data Mining
Information Discovery
Data Exploration
Statistical Analysis, Querying & Reporting
Data Warehouses / Data Marts
On-Line Analytical Processing (OLAP)
Data Sources
Paper, Files, Information Providers, Database Systems
Increasing potential to support business decisions
End User
Business Analyst
Data Analyst
DBA
spss-market.r.blog.163.com

# R

- R的源起

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。

# R

- R is free

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入，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。

R官方网站地址：http://www.r-project.org

# R

- 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。

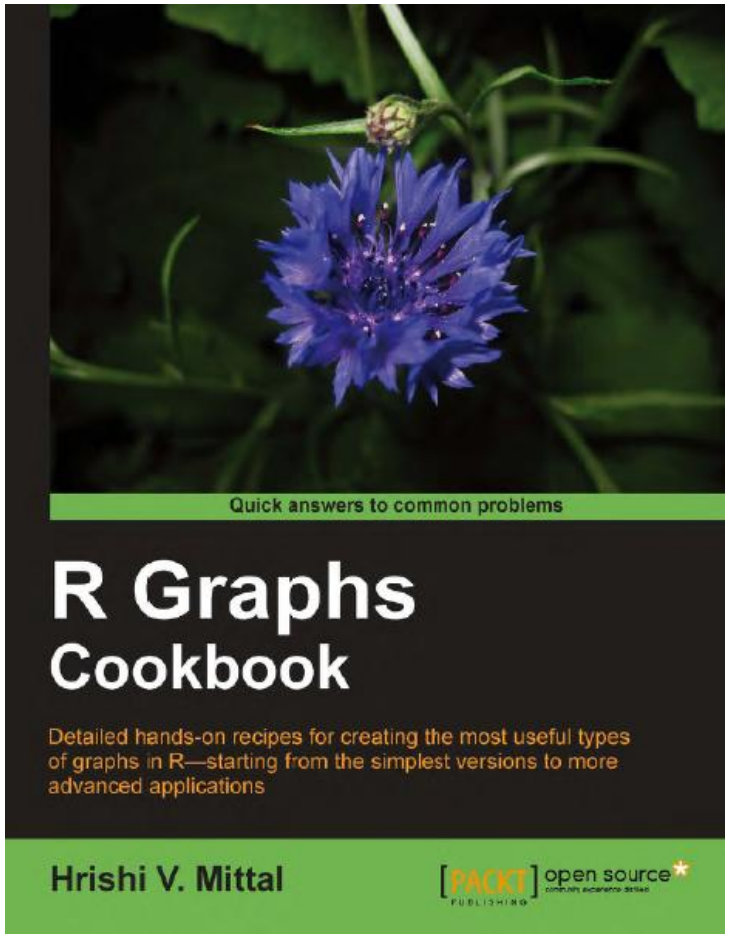
8．R语言是自由软件，可以放心大胆地使用，但其功能却不比任何其它同类软件差。

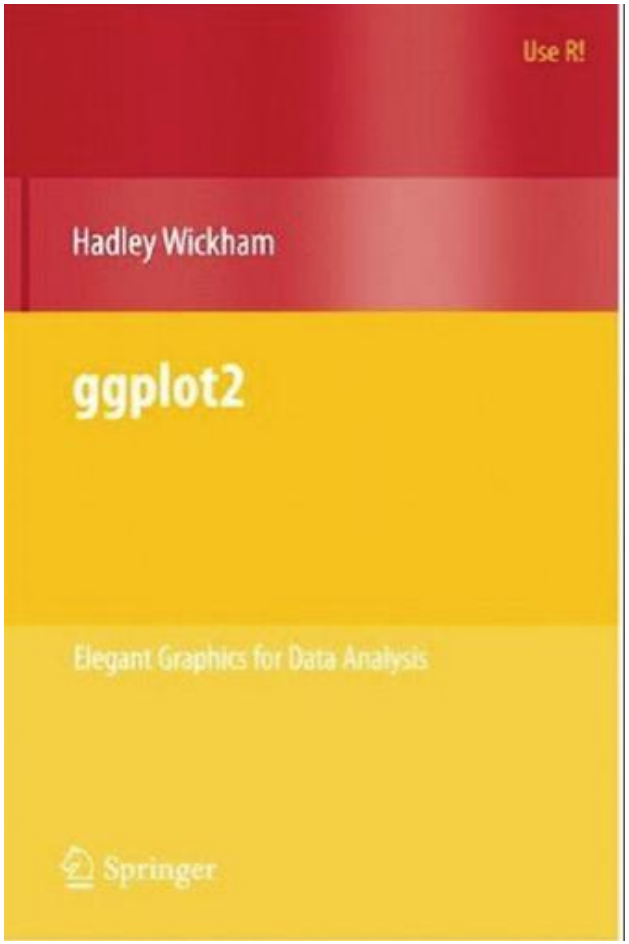
9．R语言具有丰富的网上资源

# R语言的图形能力

- 大量的画图函数
- 丰富的扩展包，部分的功能超越商业软件
- 没有昂贵的许可证费用
- 网上散布有大量的样例资源，很多数据分析或数据可视化类的著作都采用R作为画图软件
- R并非万能，很多信息图还需要专业人员用专业软件继续加工

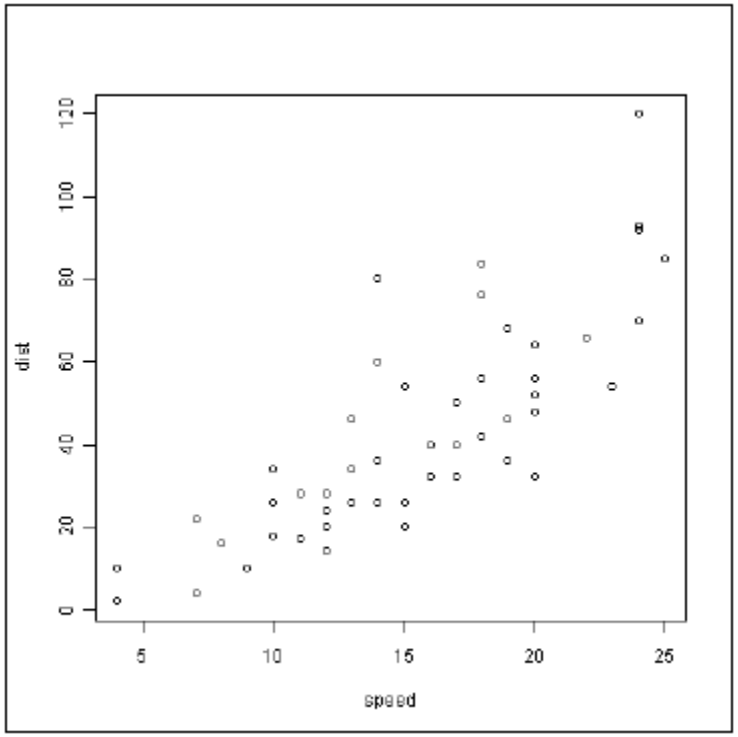
# 课程参考书


Quick answers to common problems
R Graphs
Cookbook
Detailed hands-on recipes for creating the most useful types of graphs in R—starting from the simplest versions to more advanced applications
Hrishi V. Mittal
[PACKT] open source


Use R!
Hadley Wickham
ggplot2
Elegant Graphics for Data Analysis
Springer

# 散点图

- 在坐标系里直观地显示样本数据的分布情况
- 一般可以描画2-3变量（维）样本数据，更高维度的可以使用辅助的标识方法，例如面积、颜色、文字等



# 样板数据：汽车数据集

- help(cars)

cars {datasets} R Documentation

Speed and Stopping Distances of Cars

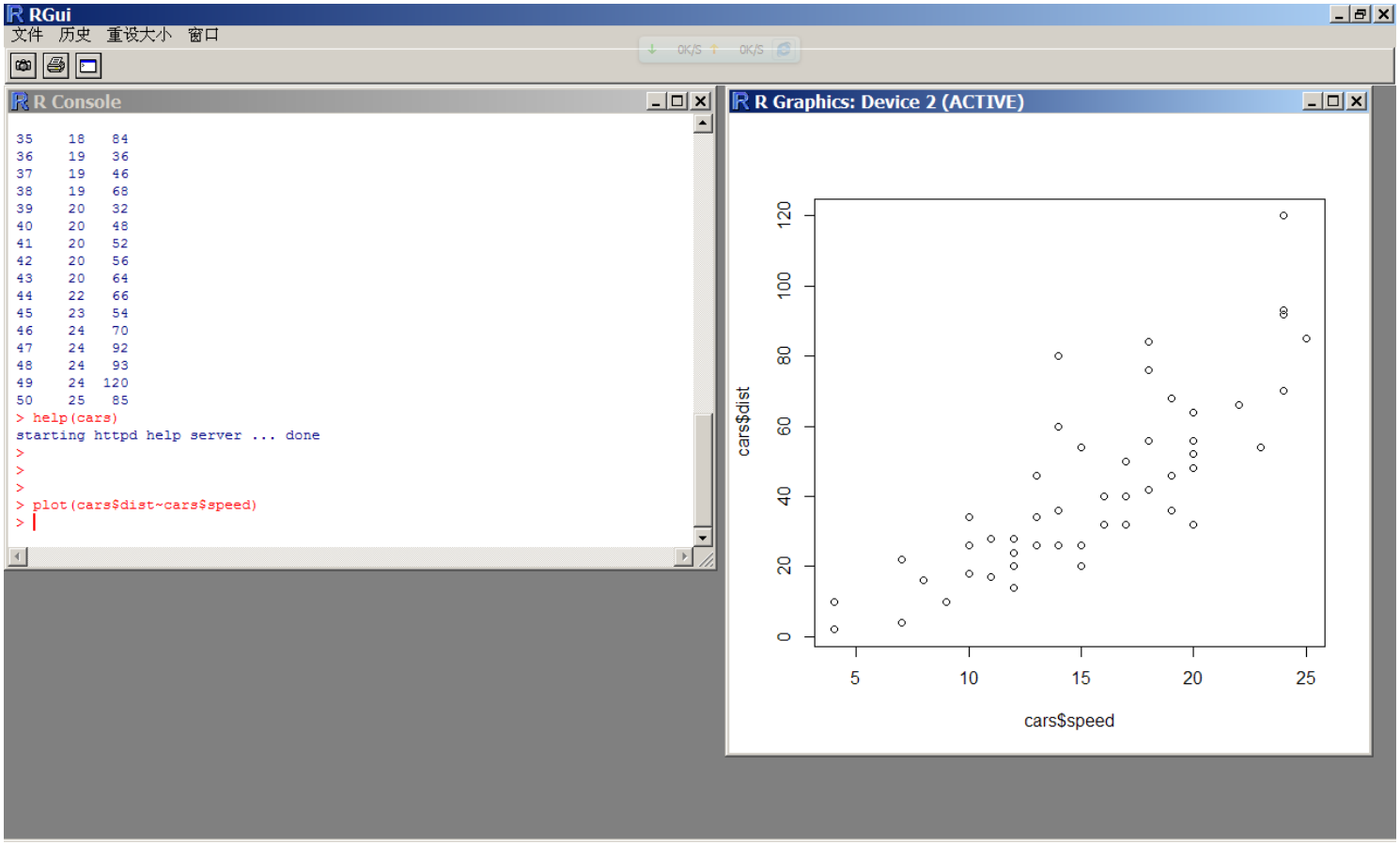
Description

The data give the speed of cars and the distances taken to stop. Note that the data were recorded in the 1920s.

```
> cars
   speed dist
1      4    2
2      4   10
3      7    4
4      7   22
5      8   16
6      9   10
7     10   18
8     10   26
9     10   34
10    11   17
```

# plot()函数

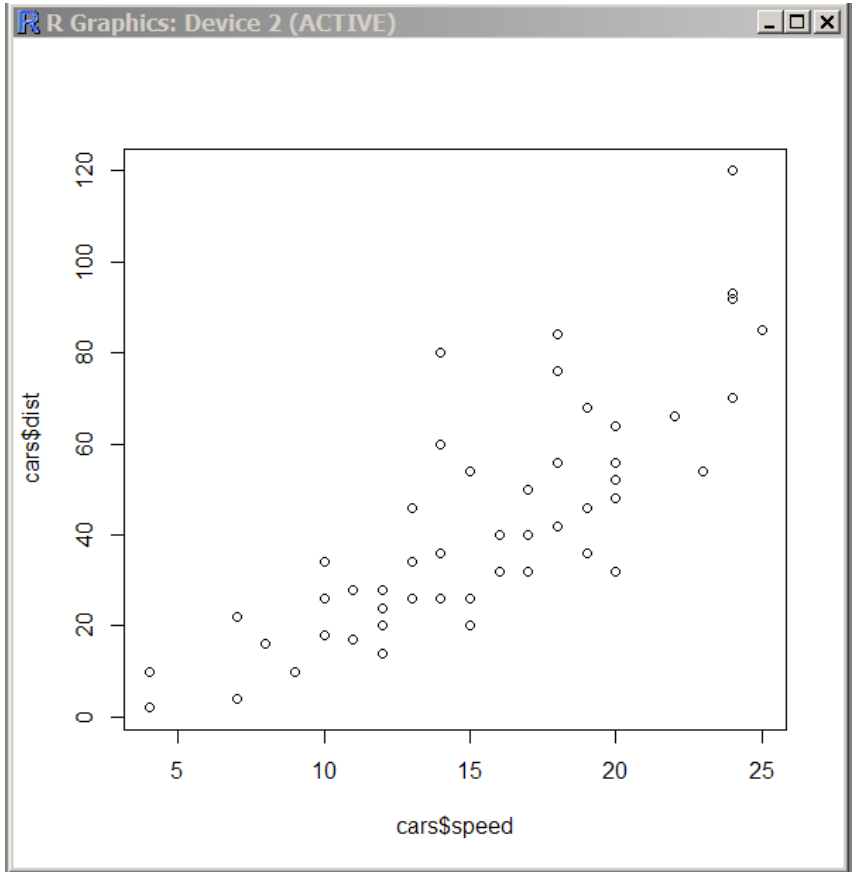
plot(cars$dist~cars$speed)

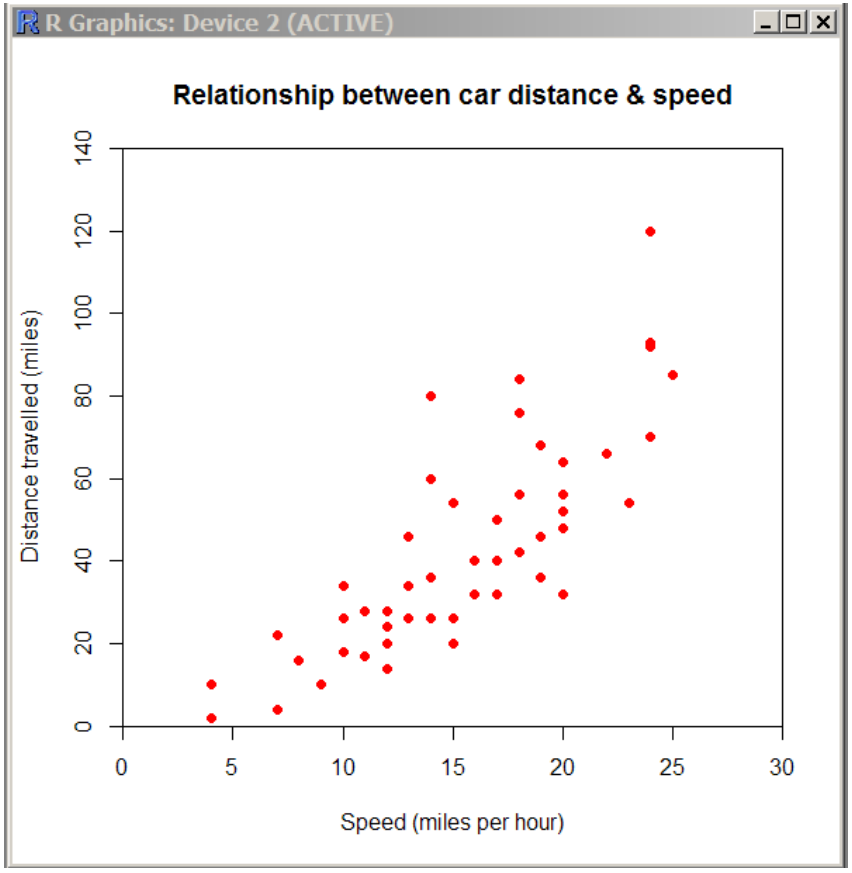


# plot()函数

```
plot(cars$dist~cars$speed, # y~x
main="Relationship between car distance & speed", # 画标题
xlab="Speed (miles per hour)", #X坐标轴标题
ylab="Distance travelled (miles)", #Y坐标轴标题
xlim=c(0,30), #设置X轴范围为从0到30
ylim=c(0,140), #设置Y轴范围为从0到140
xaxs="i", #设置X轴风格internal
yaxs="i", #设置Y轴风格internal
col="red", #设置“散点”的颜色为红色
pch=19) #设置散点的形状为实心圆点
```

# plot()函数


R Graphics: Device 2 (ACTIVE)
cars$dist
cars$speed
0
20
40
60
80
100
120
5
10
15
20
25


R Graphics: Device 2 (ACTIVE)
Relationship between car distance & speed
Distance travelled (miles)
Speed (miles per hour)
0
20
40
60
80
100
120
140
0
5
10
15
20
25
30

# internal风格

xaxs

The style of axis interval calculation to be used for the x-axis. Possible values are "r", "i", "e", "s", "d". The styles are generally controlled by the range of data or xlim, if given.
Style "r" (regular) first extends the data range by 4 percent at each end and then finds an axis with pretty labels that fits within the extended range.
Style "i" (internal) just finds an axis with pretty labels that fits within the original data range.
Style "s" (standard) finds an axis with pretty labels within which the original data range fits.
Style "e" (extended) is like style "s", except that it is also ensures that there is room for plotting symbols within the bounding box.
Style "d" (direct) specifies that the current axis should be used on subsequent plots.
(*Only "r" and "i" styles have been implemented in R.*)

# 下载教材代码和数据集

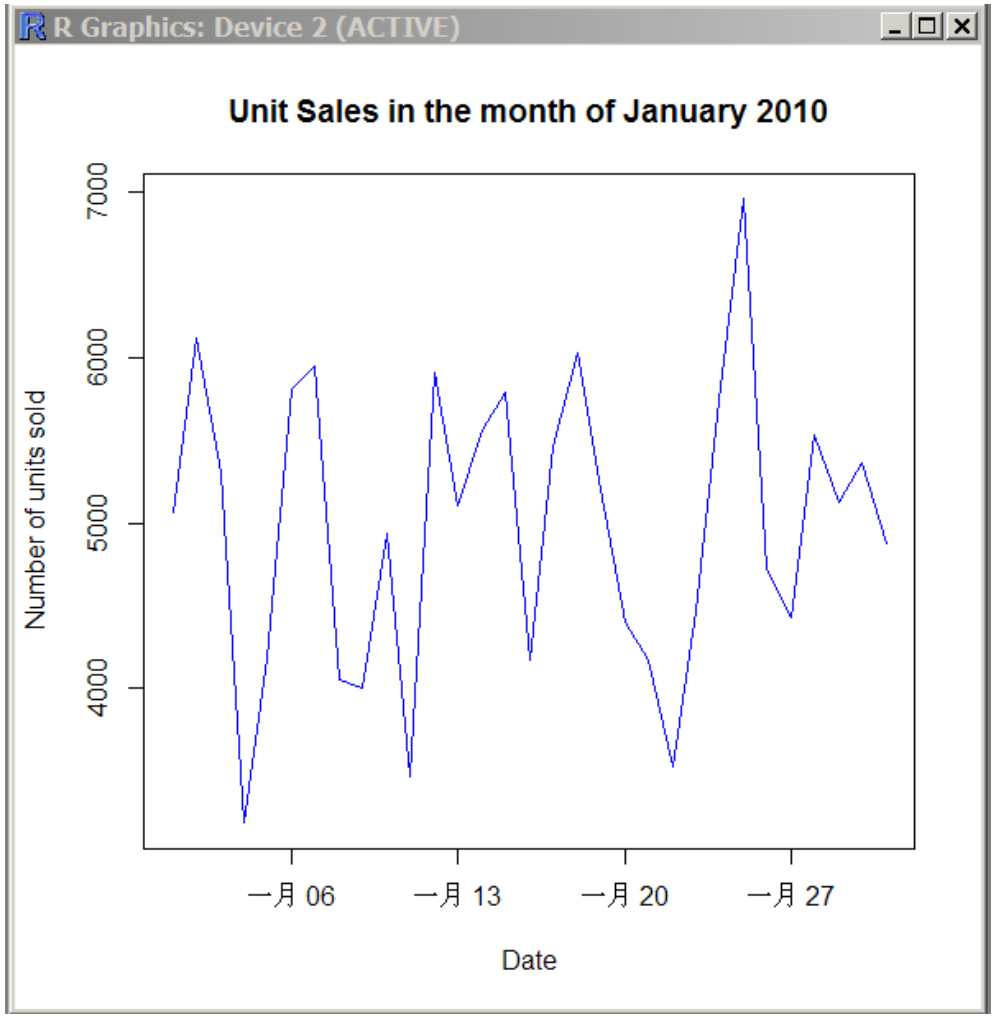
- 访问https://www.packtpub.com/r-graph-cookbook/book
- 注册账号，进入support项即可下载
- 已经分享到课程平台
- 使用时先将工作目录改至数据所在位置

# 画线图

```
sales<-read.csv("dailysales.csv", header=TRUE)


plot(sales$units~as.Date(sales$date,"%d/%m/%y"),
type="l", #指定散点图类型为“l”表示画线图
main="Unit Sales in the month of January 2010",
xlab="Date",
ylab="Number of units sold",col="blue")
```

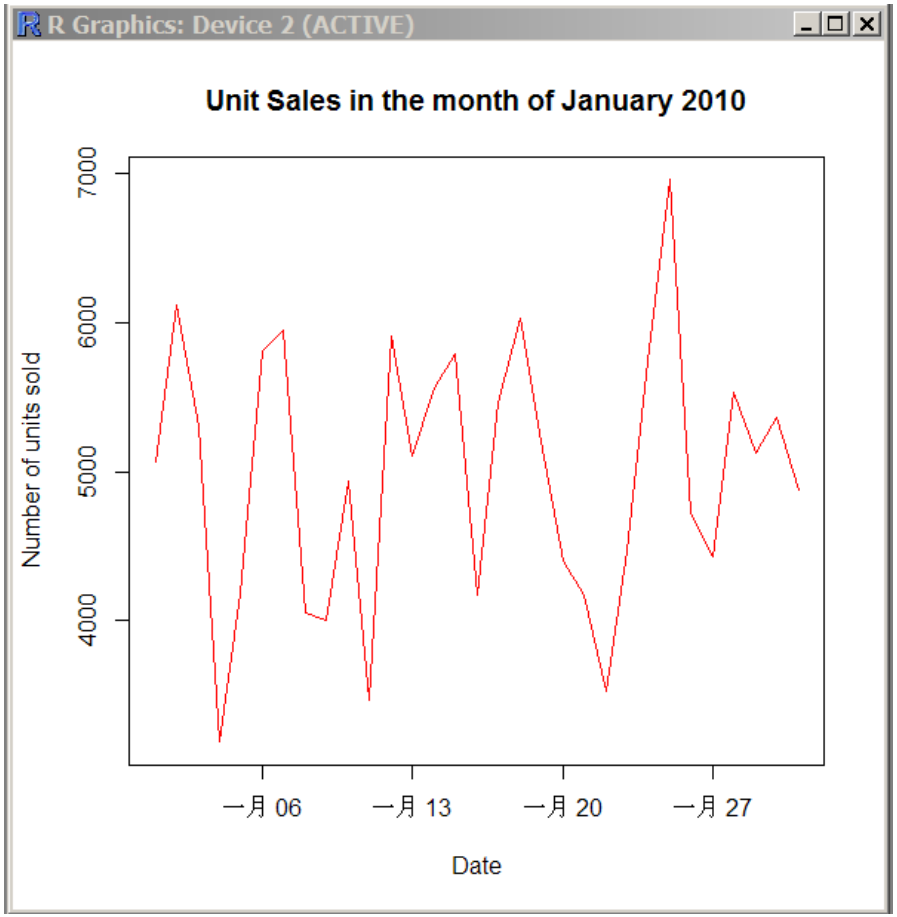
# 画线图




2012.10.18

# lines()函数

lines(sales$units~as.Date(sales$date,"%d/%m/%y"),col="red")




2012.10.18

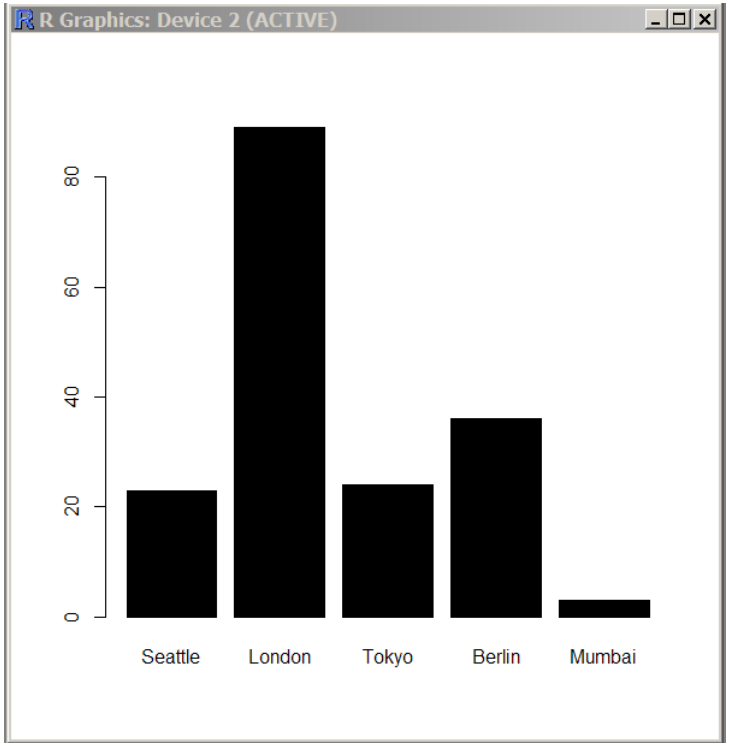
# 高水平作图函数与低水平作图函数

- 高水平作图函数：可以独立绘图，例如plot()
- 低水平作图函数：必须先运行高水平作图函数绘图，然后再加画在已有的图上面

# 柱形图与barplot()函数

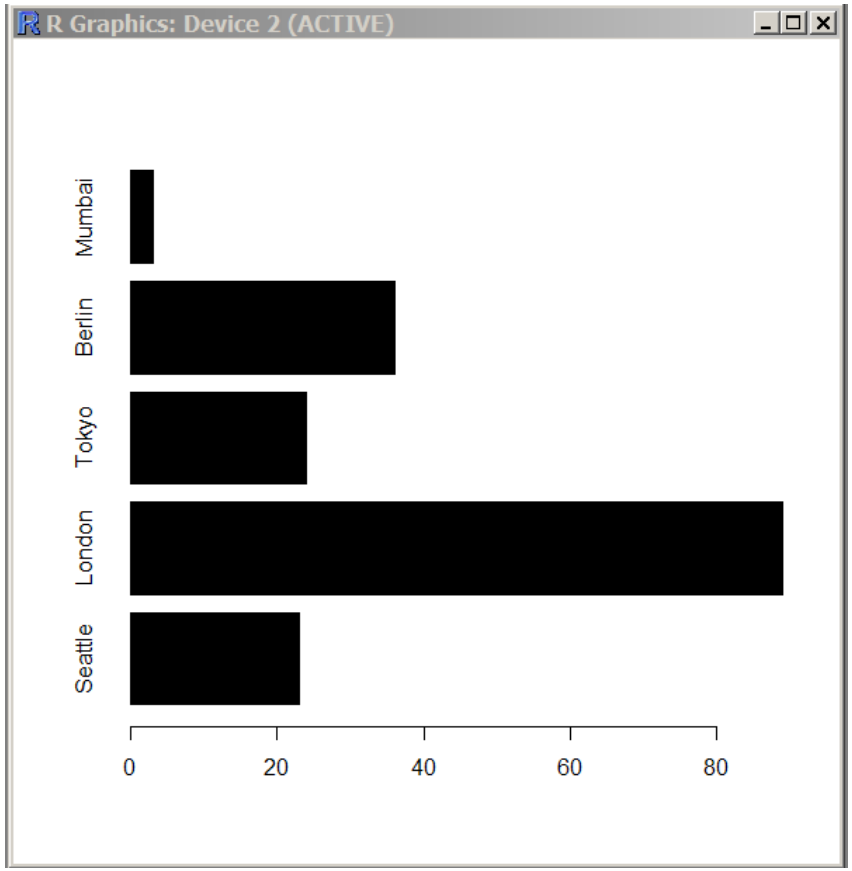
sales<-read.csv("citysales.csv",header=TRUE)

barplot(sales$ProductA,names.arg= sales$City,col="black")



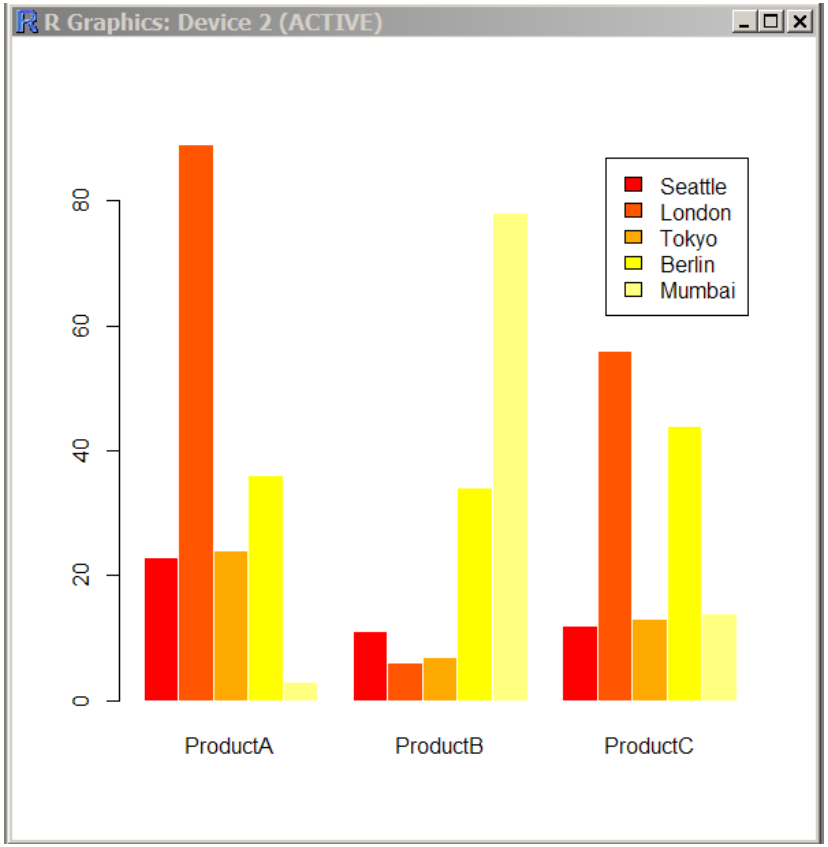
# 水平柱形图

```
barplot(sales$ProductA,names.arg=sales$City,horiz=TRUE,col="black")
```



# 彩色柱形图

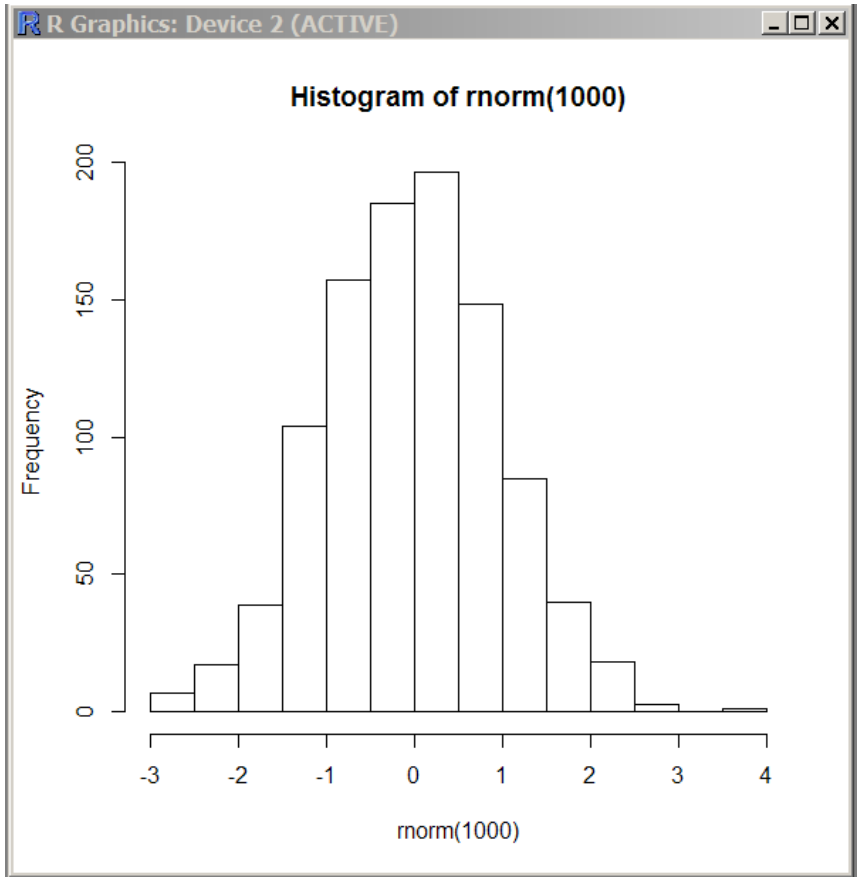
```
barplot(as.matrix(sales[,2:4]), beside=TRUE,
legend=sales$City,
col=heat.colors(5),
border="white")
```



# 直方图

直方图和柱形图有什么区别？

hist(rnorm(1000))



# 岛屿数据集

```
> islands
     Africa Antarctica        Asia   Australia Axel Heiberg      Baffin 
      11506        5500       16988        2968          16         184 
     Banks      Borneo     Britain     Celebes       Celon        Cuba 
         23         280          84          73          25          43 
     Devon    Ellesmere      Europe   Greenland      Hainan   Hispaniola 
        21          82        3745         840          13          30 
  Hokkaido     Honshu     Iceland     Ireland        Java     Kyushu 
        30          89          40          33          49          14 
     Luzon  Madagascar    Melville    Mindanao    Moluccas  New Britain 
        42         227          16          36          29          15 
New Guinea New Zealand (N) New Zealand (S) Newfoundland North America 
       306          44          58          43        9390 
Novaya Zemlya Prince of Wales     Sakhalin South America  Southampton 
        32          13          29        6795          16 
  Spitsbergen      Sumatra      Taiwan    Tasmania Tierra del Fuego 
        15         183          14          26          19 
       Timor   Vancouver     Victoria 
        13          12          82 
> 
```

islands {datasets}

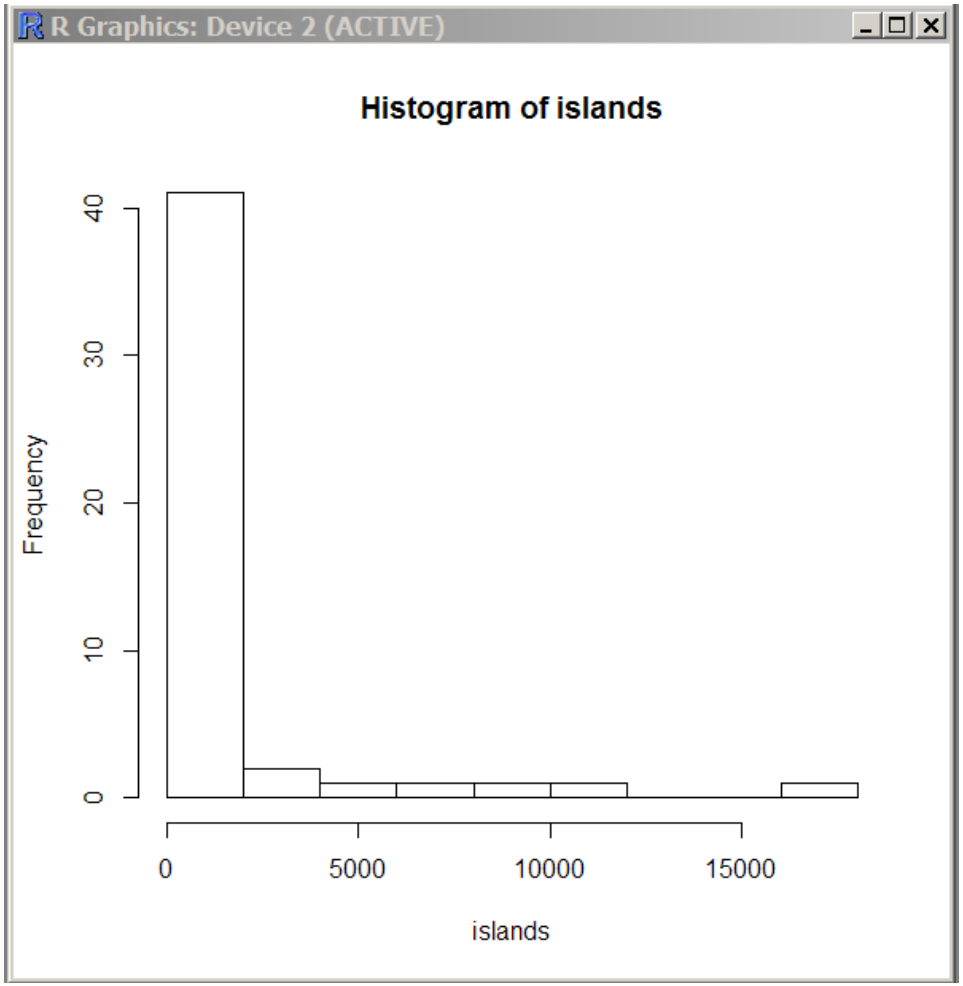
Areas of the World's Major Landmasses

## Description

The areas in thousands of square miles of the landmasses which exceed 10,000 square miles.

# 岛屿直方图

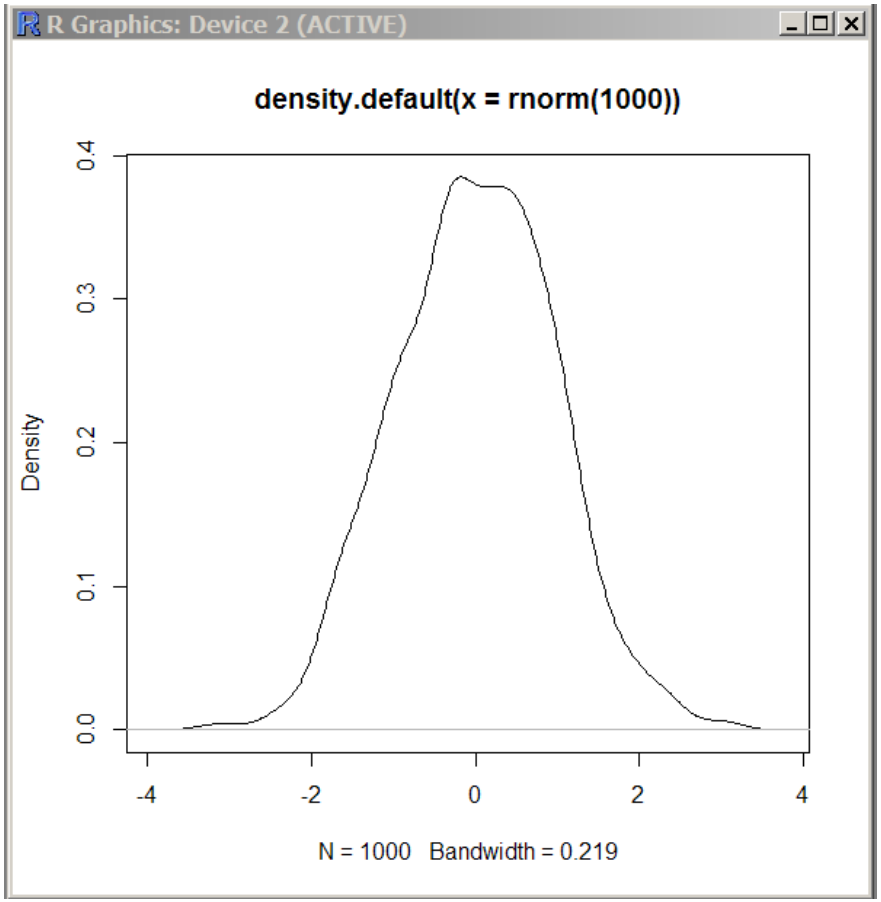
hist(islands)




2012.10.18

# 密度图

plot(density(rnorm(1000)))



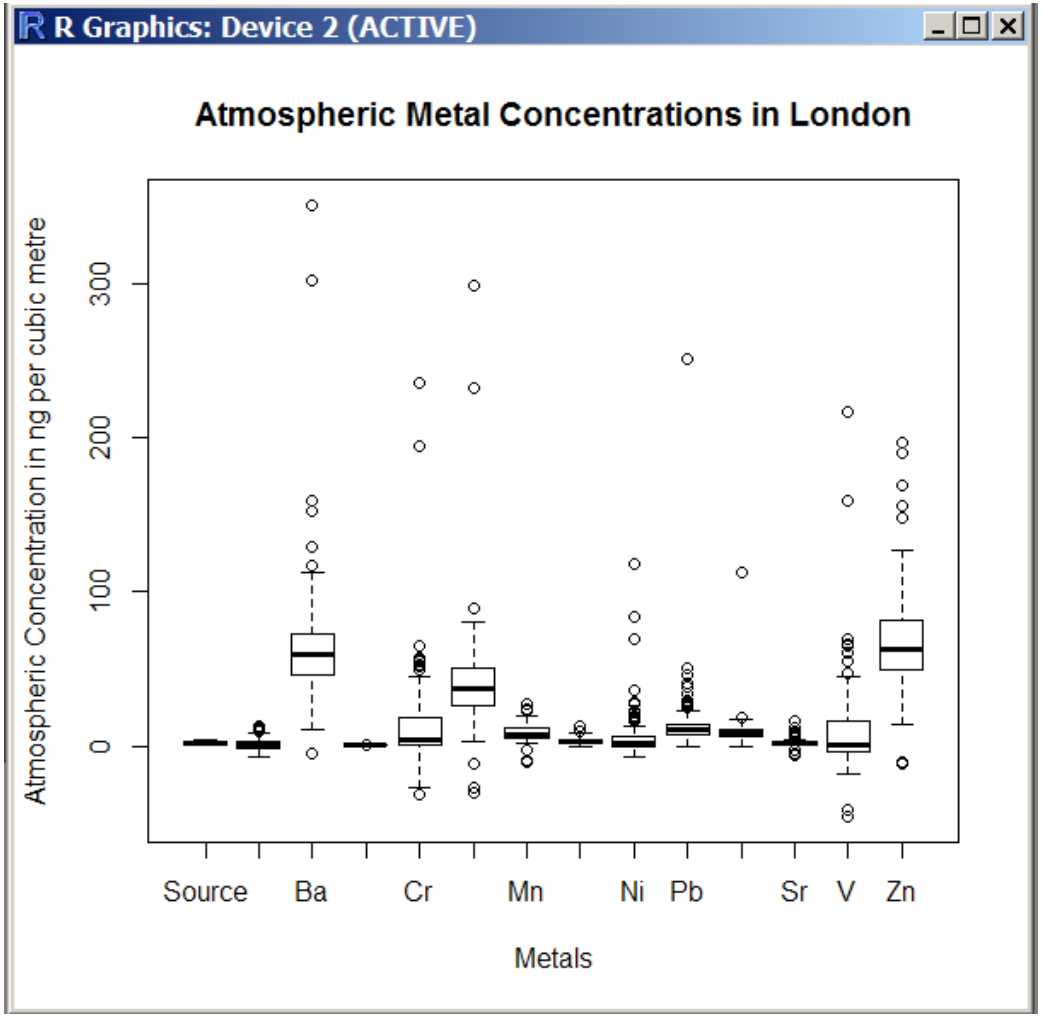
# 概率密度

- 什么是概率密度函数
- Density()函数

# 箱型图

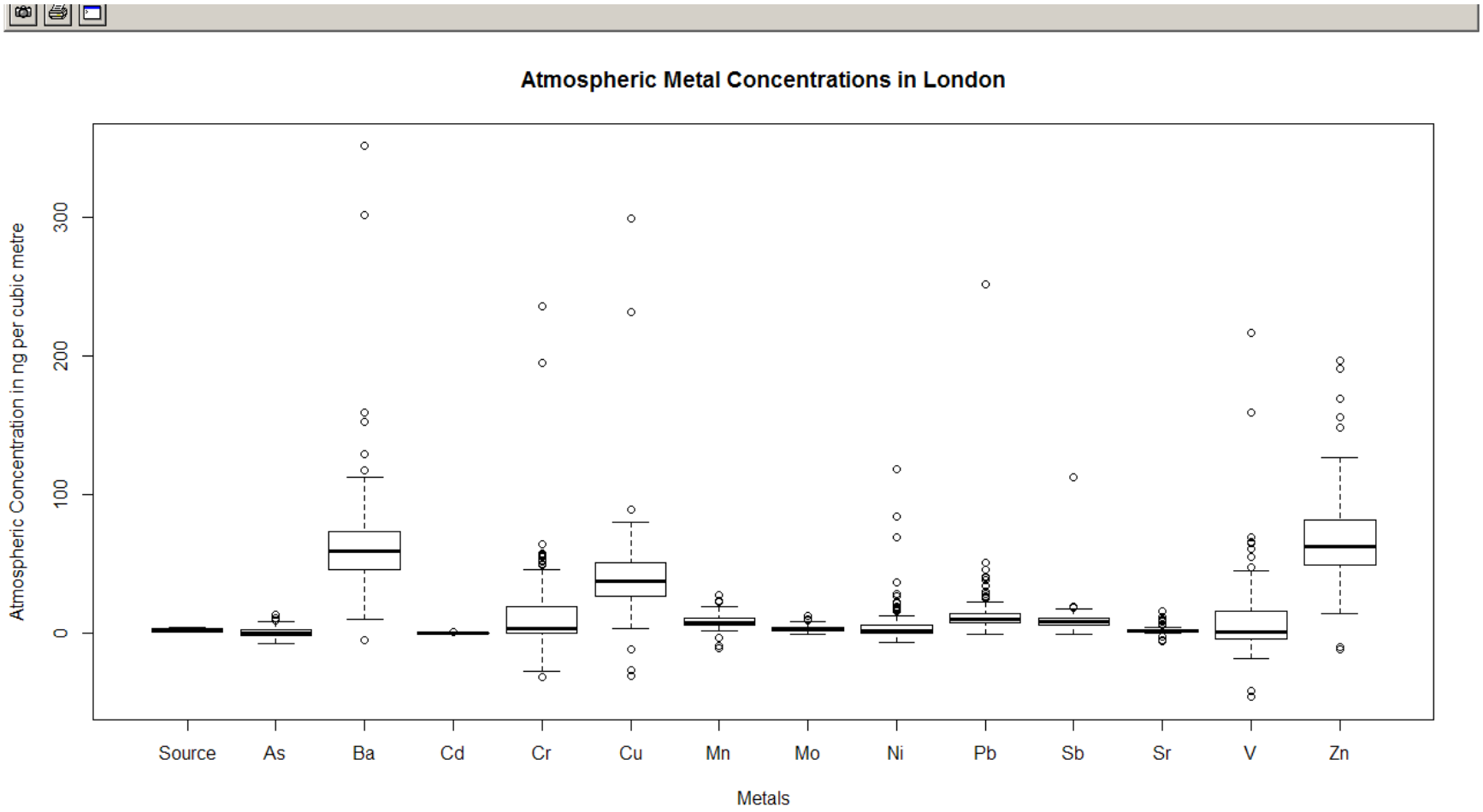
```
metals<-read.csv("metals.csv",header=TRUE)
boxplot(metals,
xlab="Metals",
ylab="Atmospheric Concentration in ng per cubic metre",
main="Atmospheric Metal Concentrations in London")
```

# 箱型图




2012.10.18

# 箱型图


Atmospheric Metal Concentrations in London
Atmospheric Concentration in ng per cubic metre
0
100
200
300
Source
As
Ba
Cd
Cr
Cu
Mn
Mo
Ni
Pb
Sb
Sr
V
Zn
Metals

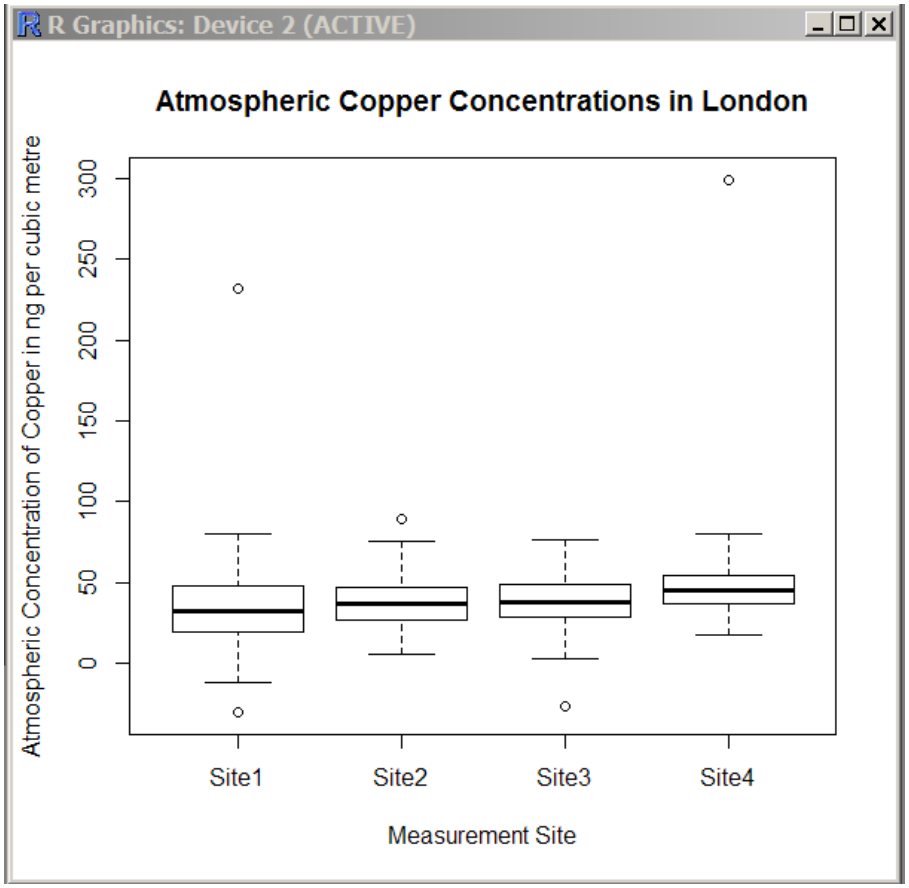
# 箱型图

- Metals数据集

```
> metals
    Source           As         Ba           Cd           Cr         Cu         Mn         Mo           Ni
1    Site1  0.701728639  76.178490  0.081646023 -3.470121296  19.699889   3.082433  0.7452889 -1.862277092
2    Site2 -1.919269099  41.009008 -0.003449090  30.685596390  10.642627   4.512707  0.9385502  7.227271387
3    Site1  0.739368011  71.914383  0.654752517  51.511445890  33.791547  14.815785  3.4669948 17.957165270
4    Site2 -2.596413594  41.875745  0.082761457  22.843039360   5.353111   4.546102  0.5725344  5.640279732
5    Site2  0.938645348  79.950683  0.105580364  42.084568950  19.889561   6.280715  2.2800103 16.368316280
6    Site2  2.713442060  83.472013  0.717620867  27.581914180  26.965605  27.520700  2.9449435  9.068604558
7    Site1 -3.034884138  64.770994  0.056873438  21.603194550  13.851614   7.967017  1.4370751  5.086855124
8    Site3  2.319939251  57.655505  0.434388285  49.617762240  21.993663  14.127151  2.7323799 17.243794580
9    Site3 -0.709771695  38.482657  0.286556361  28.722924380  19.720640   7.634066  2.2495872  9.890787708
10   Site1 -0.623042378  49.156437  0.091879950   8.318703181  23.240948   5.801627  1.1701979  0.367944809
11   Site3          NA         NA           NA           NA         NA         NA         NA           NA
12   Site1  1.243607892 112.862585  0.282025526   5.149518971  17.349278  16.769269  1.5637107  0.796301571
13   Site3 -1.326180752 100.479583  0.141942783  16.313233130  13.679218  12.401545  1.0556735  4.761978587
14   Site1 -2.241817707 129.210636  0.071583978  18.397330280  16.893993   7.215997  1.2574726  4.047844249
15   Site3  0.455225201  91.219655  0.172739837   2.397593274   8.766632   9.992623  0.6136857  0.548320637
16   Site1          NA         NA           NA           NA         NA         NA         NA           NA
17   Site3 -1.984350877  49.224376  0.111075531  26.109957660  16.440350   8.423137  1.4218853  7.962349927
18   Site1 -0.974415802  73.213467  0.066013499  23.454307860  12.729233   5.845597  1.1978293  6.728842480
19   Site3 -2.519845713 102.662629  0.230340797   5.546926802  36.900327   8.901423  2.3857929 -0.369218342
20   Site3 -0.673901264 159.570294  0.165666992  -0.225998726  30.713409   9.085403  1.9677706  2.038494867
21   Site1          NA         NA           NA           NA         NA         NA         NA           NA
22   Site3 -2.599664851  50.795689  0.163076706  28.459345530  12.005376   6.382453  1.2346256  9.686064688
23   Site1 -3.631637028  97.759112  0.039768241   0.868637187  29.087656   6.790746  2.1180497 -2.189379655
24   Site3 -4.033622187  69.034232  0.118983455   3.115317256  31.402296   4.679465  1.5355406  0.653094375
    Sb
1  7.28
2  4.80
3 50.66
4  4.72
5  8.97
6 38.65
7  7.57
8 11.03
9 10.12
10 5.00
11   NA
12 13.36
13  5.96
14  5.74
15  4.17
16   NA
17  5.76
18  4.34
19  9.56
20  6.93
21   NA
22  5.11
23  7.40
24  4.25
```

# 详解箱线图

```
copper<-
    read.csv("copper_site.csv",header
    =TRUE)
boxplot(copper$Cu~copper$Source,
xlab="Measurement Site",
ylab="Atmospheric Concentration of
    Copper in ng per cubic metre",
main="Atmospheric Copper
    Concentrations in London")
```



# mtcras数据集

DATAGURU 炼数成金

## Description

The data was extracted from the 1974 *Motor Trend* US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973-74 models).

## Usage

```
mtcars
```

## Format

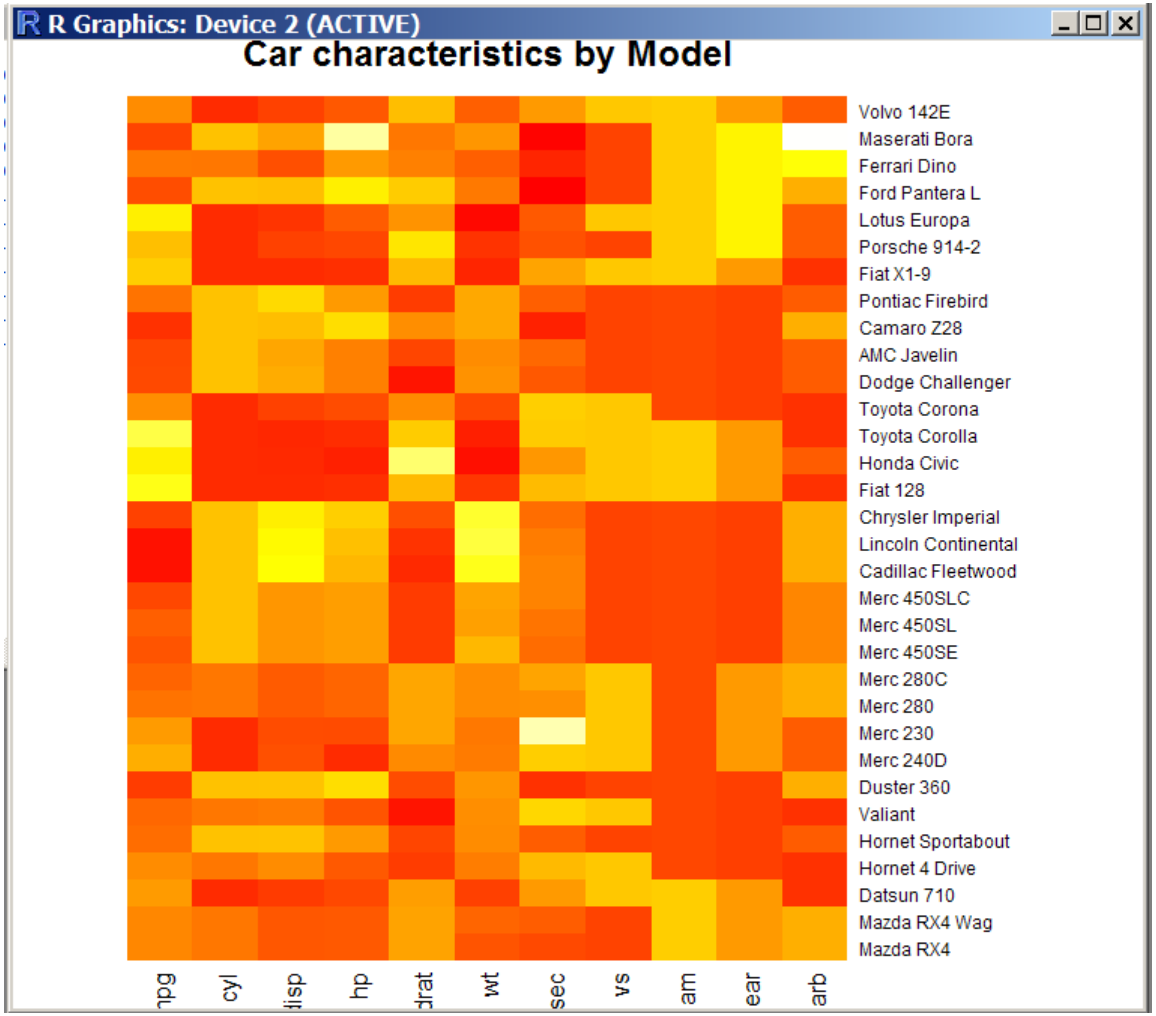
A data frame with 32 observations on 11 variables.

[, 1] mpg  Miles/(US) gallon
[, 2] cyl  Number of cylinders
[, 3] disp Displacement (cu.in.)
[, 4] hp   Gross horsepower
[, 5] drat Rear axle ratio
[, 6] wt   Weight (lb/1000)
[, 7] qsec 1/4 mile time
[, 8] vs   V/S
[, 9] am   Transmission (0 = automatic, 1 = manual)
[,10] gear Number of forward gears
[,11] carb Number of carburetors

# 热力图

```
heatmap(as.matrix(mtcars),
Rowv=NA,
Colv=NA,
col = heat.colors(256),
scale="column",
margins=c(2,8),
main = "Car characteristics by Model")
```

# 热力图

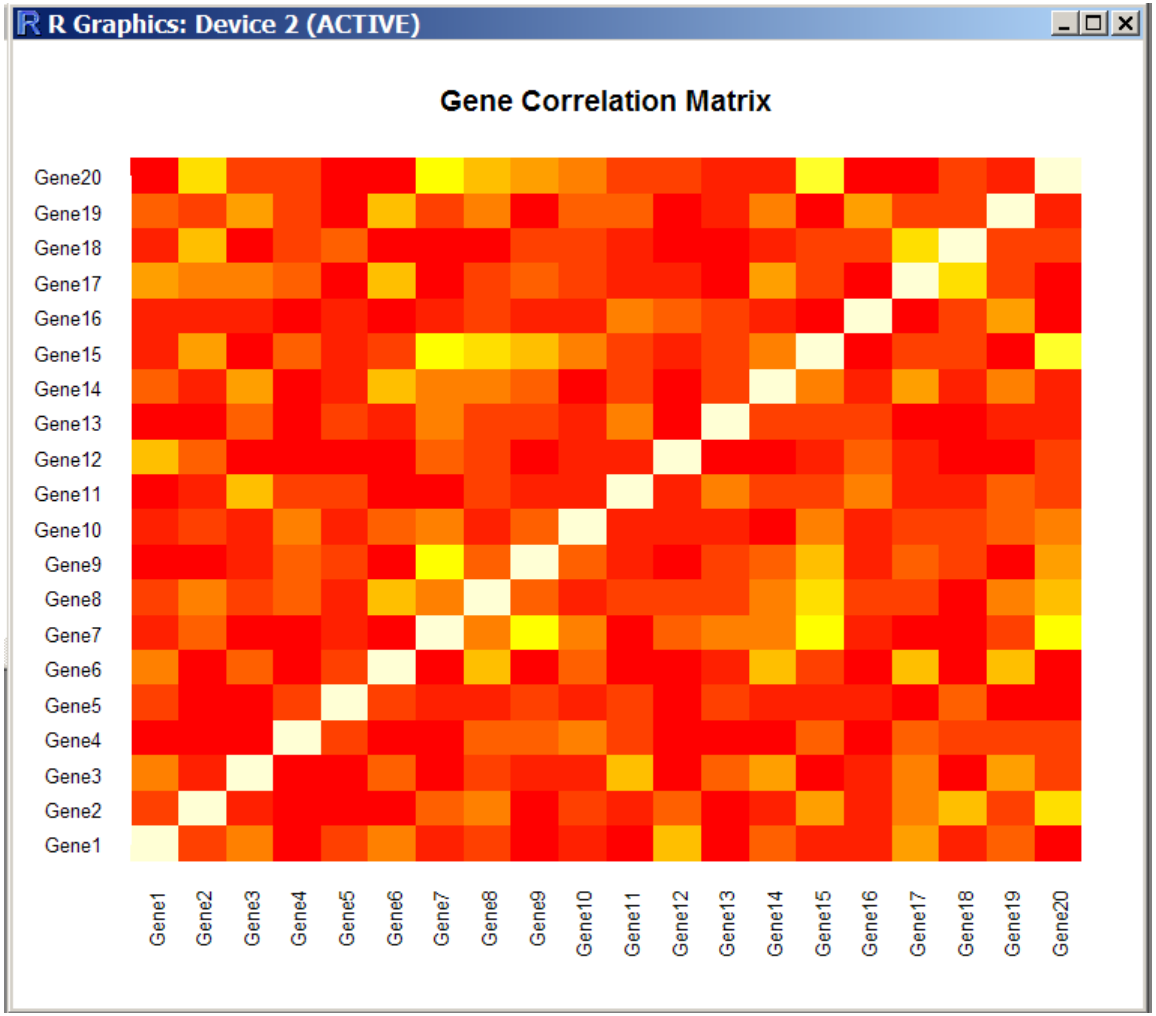



2012.10.18

# 基因热力图

```
genes<-read.csv("genes.csv",header=T)
rownames(genes)<-colnames(genes)
image(x=1:ncol(genes),
y=1:nrow(genes),
z=t(as.matrix(genes)),
axes=FALSE,
xlab="",
ylab="" ,
main="Gene Correlation Matrix")
axis(1,at=1:ncol(genes),labels=colnames(genes),col="white",
las=2,cex.axis=0.8)
axis(2,at=1:nrow(genes),labels=rownames(genes),col="white",
las=1,cex.axis=0.8)
```

# 基因热力图



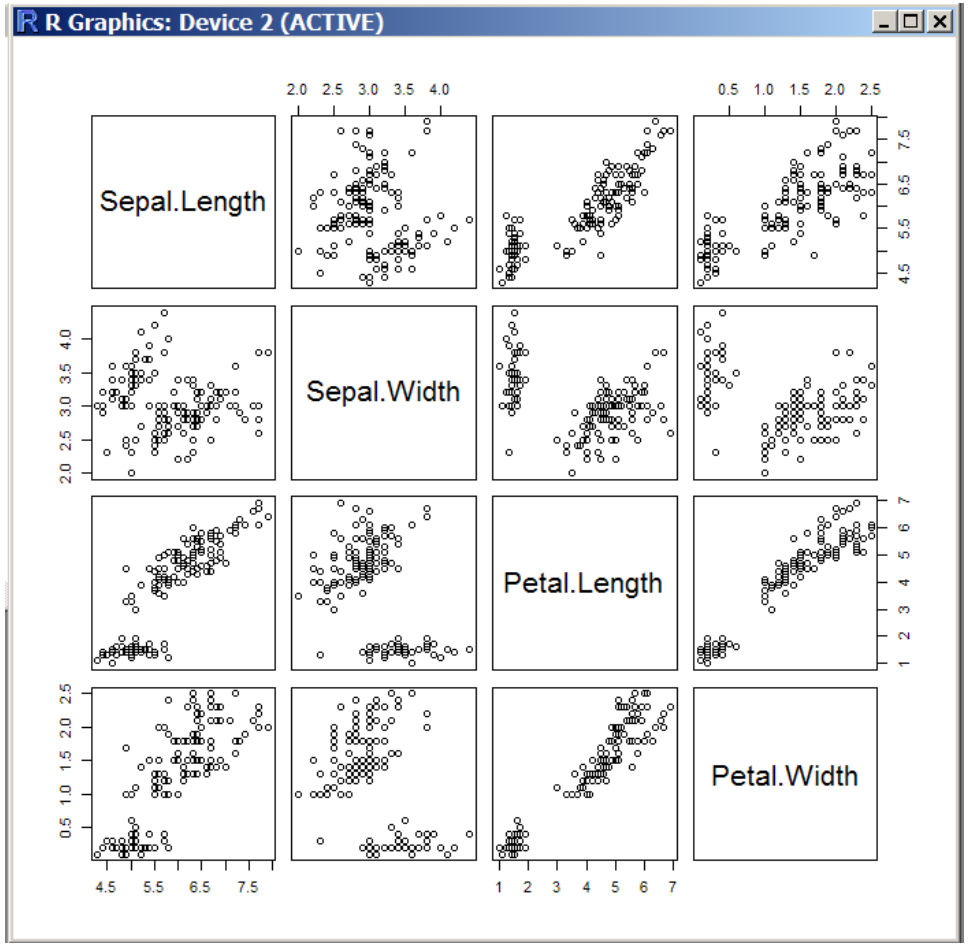

2012.10.18

# 鸢尾花数据集

```
> 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
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
14           4.3         3.0          1.1         0.1     setosa
15           5.8         4.0          1.2         0.2     setosa
16           5.7         4.4          1.5         0.4     setosa
17           5.4         3.9          1.3         0.4     setosa
18           5.1         3.5          1.4         0.3     setosa
19           5.7         3.8          1.7         0.3     setosa
20           5.1         3.8          1.5         0.3     setosa
21           5.4         3.4          1.7         0.2     setosa
```

# 散点图阵

pairs(iris[,1:4])

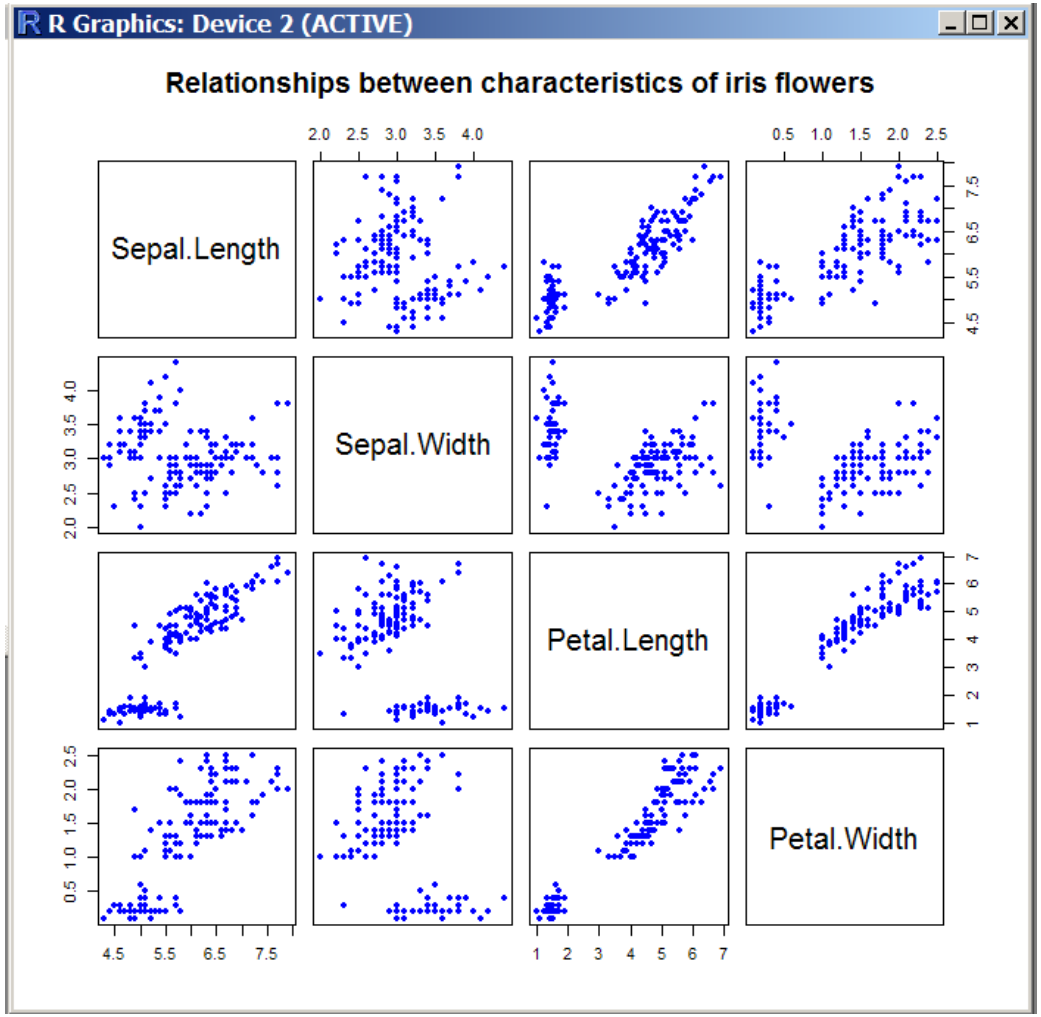



2012.10.18

# 散点图阵

```
plot(iris[,1:4],
main="Relationships between characteristics of iris flowers",
pch=19,
col="blue",
cex=0.9)
```

# 散点图阵

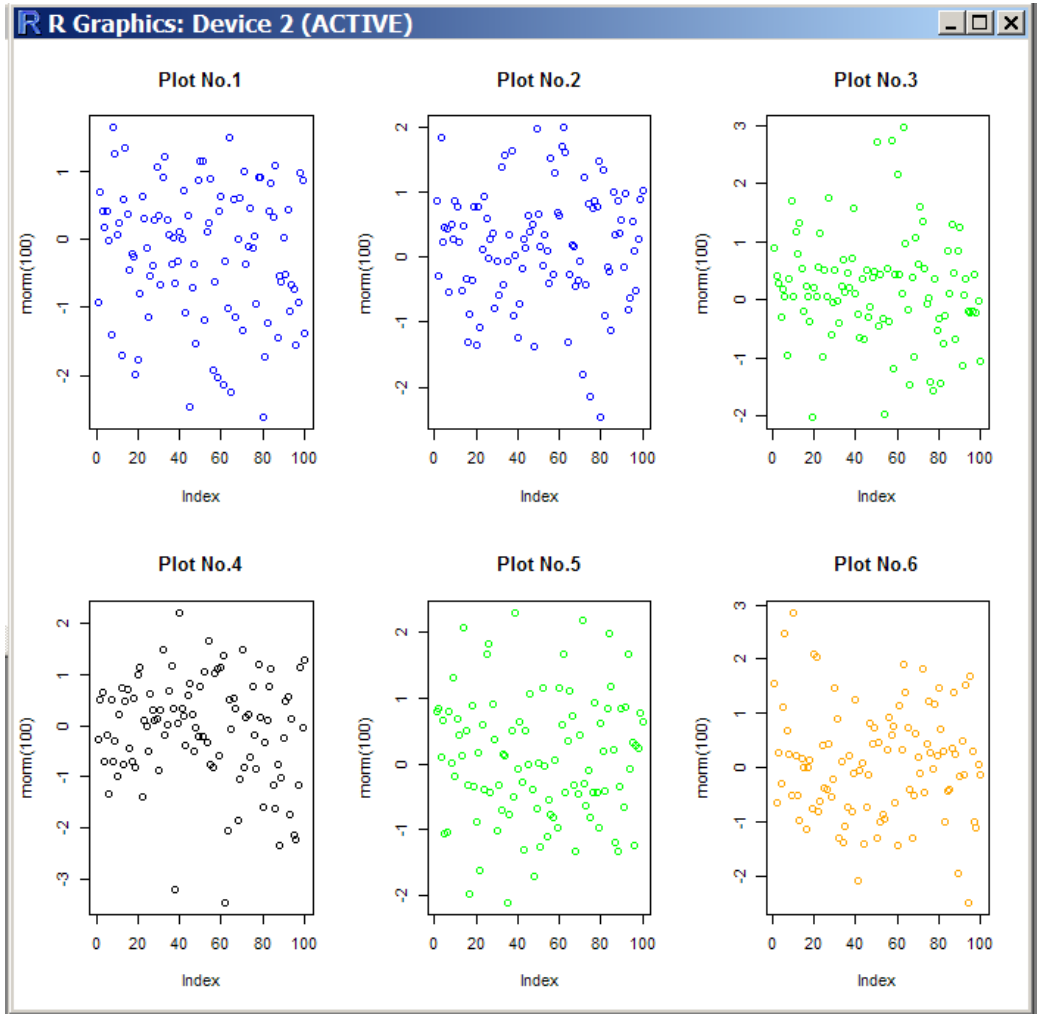



2012.10.18

# 在一张画板上画多个散点图

```
par(mfrow=c(2,3))
plot(rnorm(100),col="blue",main="Plot No.1")
plot(rnorm(100),col="blue",main="Plot No.2")
plot(rnorm(100),col="green",main="Plot No.3")
plot(rnorm(100),col="black",main="Plot No.4")
plot(rnorm(100),col="green",main="Plot No.5")
plot(rnorm(100),col="orange",main="Plot No.6")
```

# 在一张画板上画多个散点图

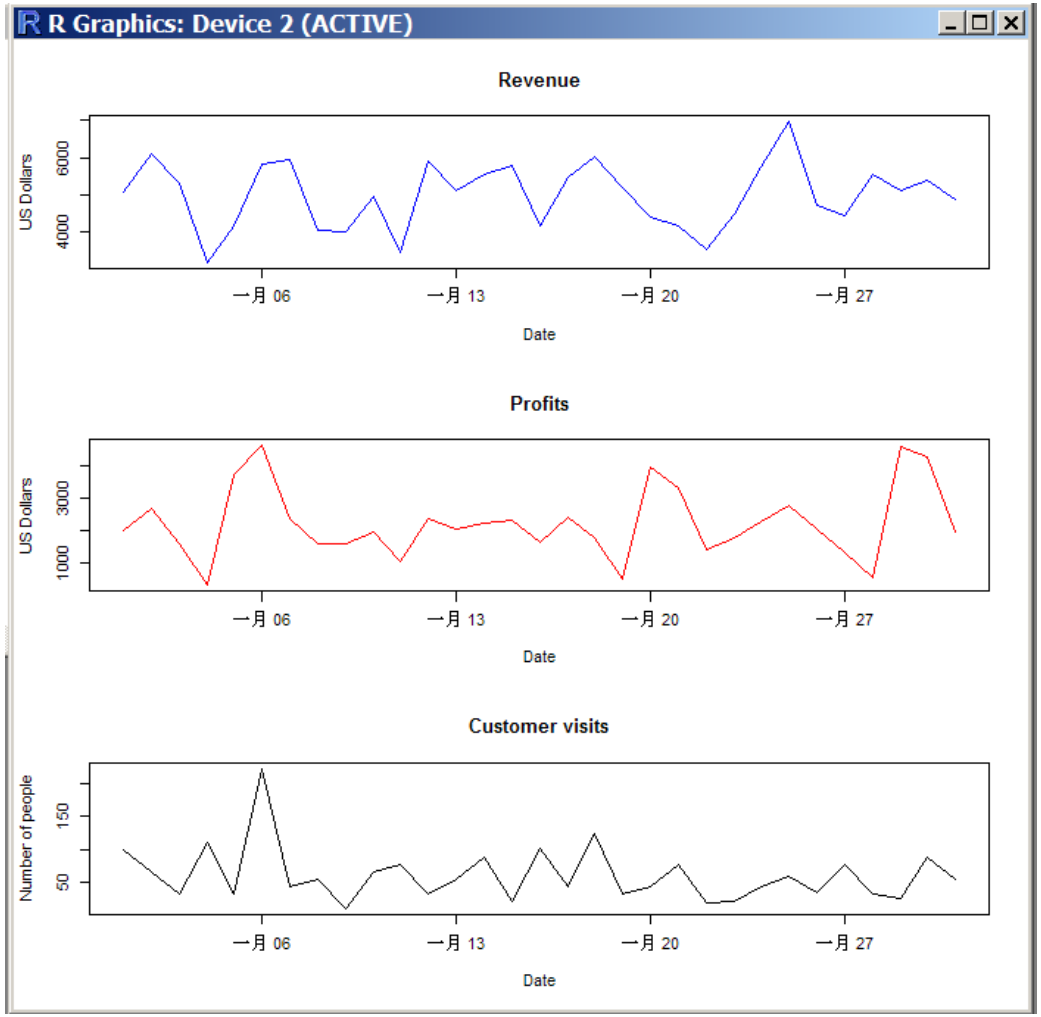



2012.10.18

# 市场数据

```
market<-read.csv("dailymarket.csv",header=TRUE)
par(mfrow=c(3,1))
plot(market$revenue~as.Date(market$date,"%d/%m/%y"),
type="l", #Specify type of plot as l for line
main="Revenue",
xlab="Date",
ylab="US Dollars",
col="blue")
plot(market$profits~as.Date(market$date,"%d/%m/%y"),
type="l", #Specify type of plot as l for line
main="Profits",
xlab="Date",
ylab="US Dollars",
col="red")
plot(market$customers~as.Date(market$date,"%d/%m/%y"),
type="l", #Specify type of plot as l for line
main="Customer visits",
xlab="Date",
ylab="Number of people",
col="black")
```

# 市场数据



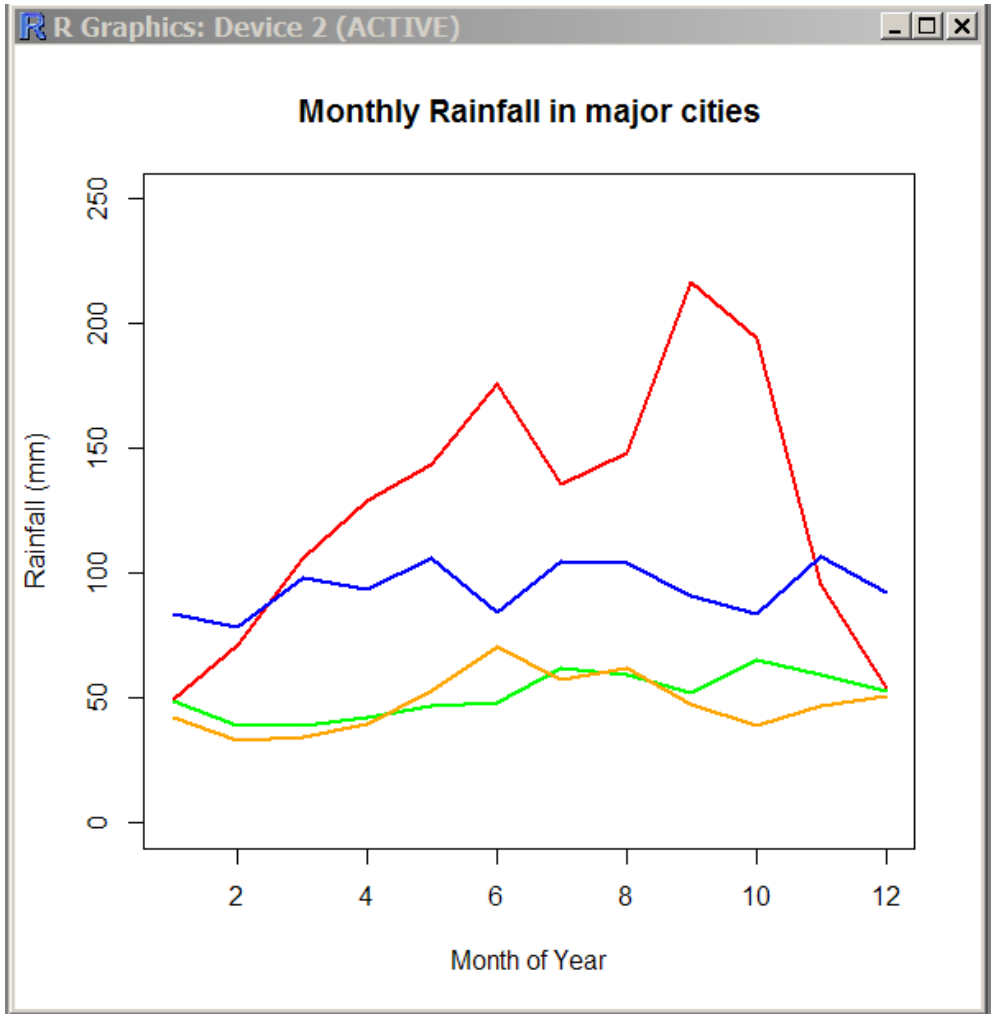

2012.10.18

# 增加图例说明

```
rain<-read.csv("cityrain.csv",header=TRUE)


plot(rain$Tokyo,type="l",col="red",
ylim=c(0,300),
main="Monthly Rainfall in major cities",
xlab="Month of Year",
ylab="Rainfall (mm)",
lwd=2)
lines(rain$NewYork,type="l",col="blue",lwd=2)
lines(rain$London,type="l",col="green",lwd=2)
lines(rain$Berlin,type="l",col="orange",lwd=2)
```

# 增加图例说明

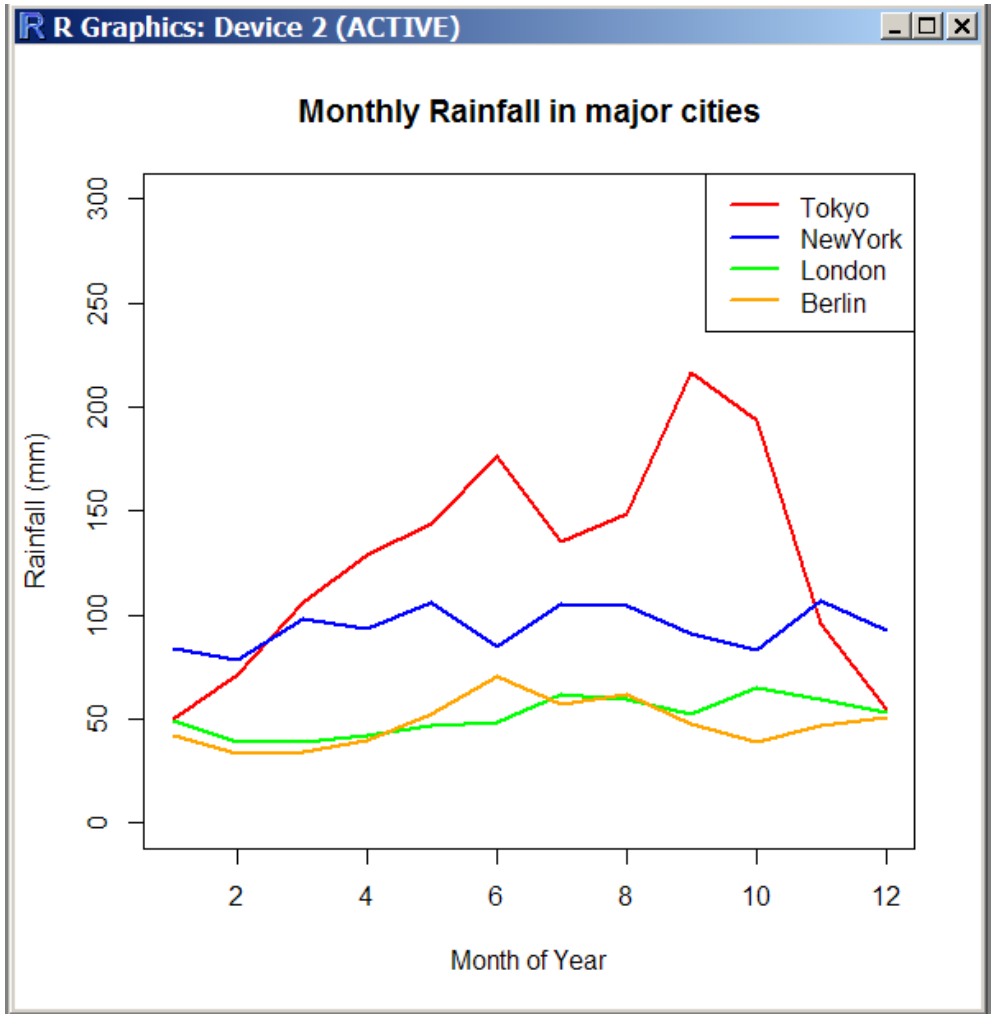



2012.10.18

# 增加图例说明

```
legend("topright",
legend=c("Tokyo","NewYork","London","Berlin"),
col=c("red","blue","green","orange"),
lty=1,lwd=2)
```

# 增加图例说明

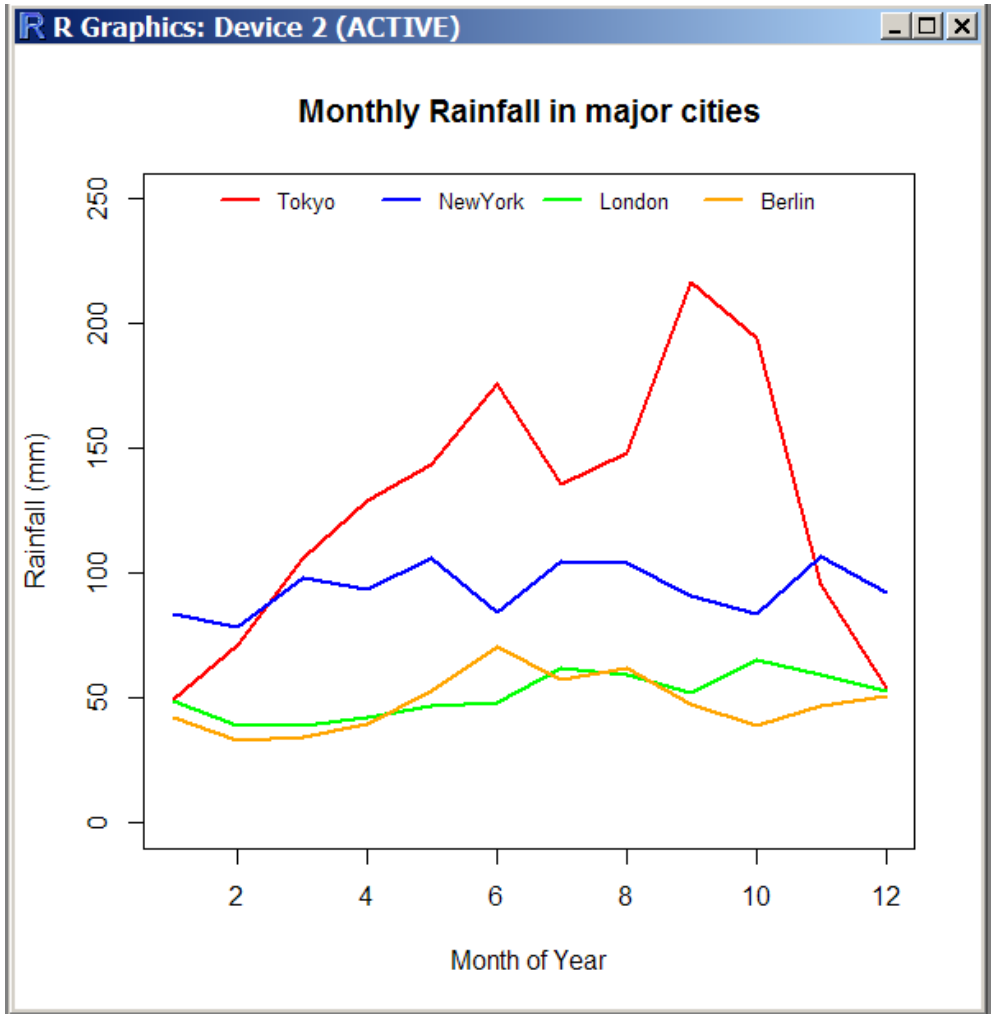



2012.10.18

# 增加图例说明——另一样式

```
legend("top",
legend=c("Tokyo","NewYork","London","Berlin"),
ncol=4,
cex=0.8,
bty="n",
col=c("red","blue","green","orange"),
lty=1,lwd=2)
```

# 另一样式




2012.10.18

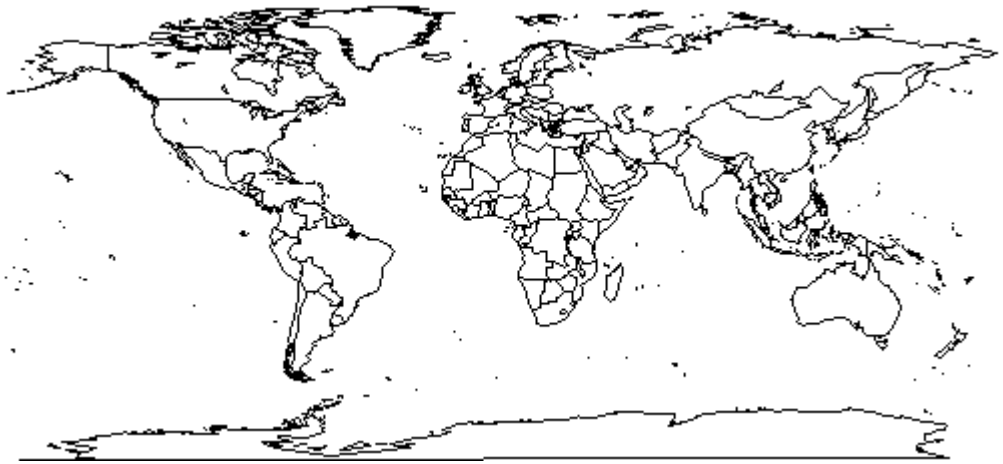
# 地图

- 安装地图包maps
- 用library函数加载

# 地图

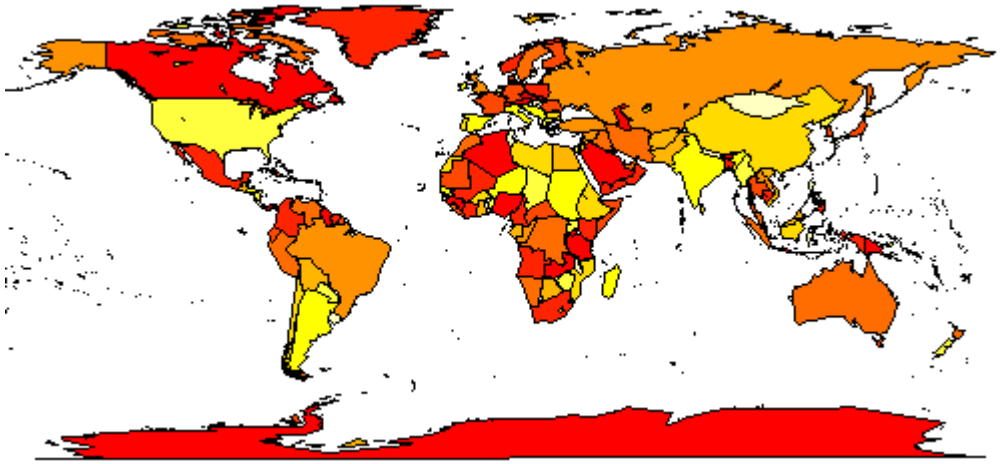
library(maps)

map()



# 地图

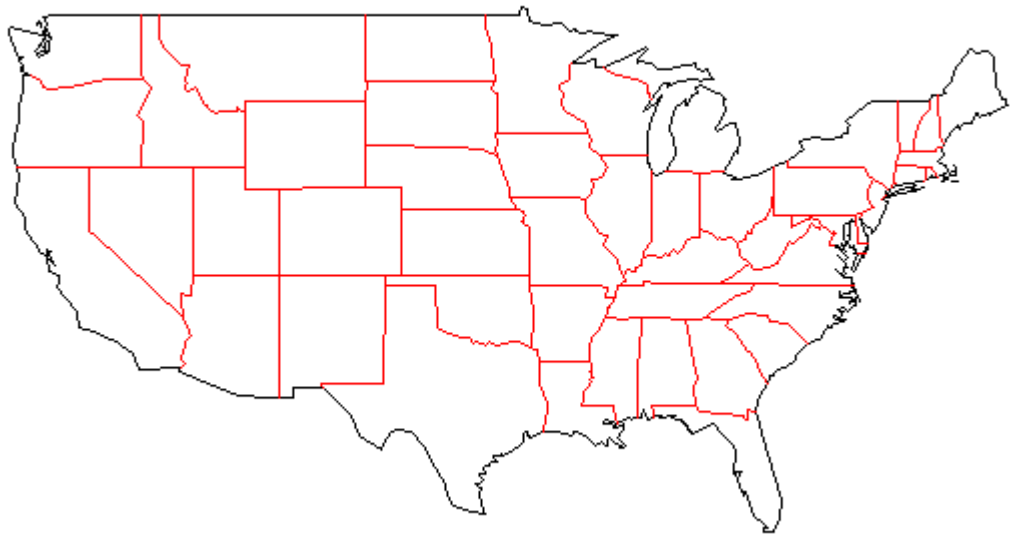
map('world', fill = TRUE,col=heat.colors(10))



# 美国地图

map("state", interior = FALSE)

map("state", boundary = FALSE, col="red", add = TRUE)

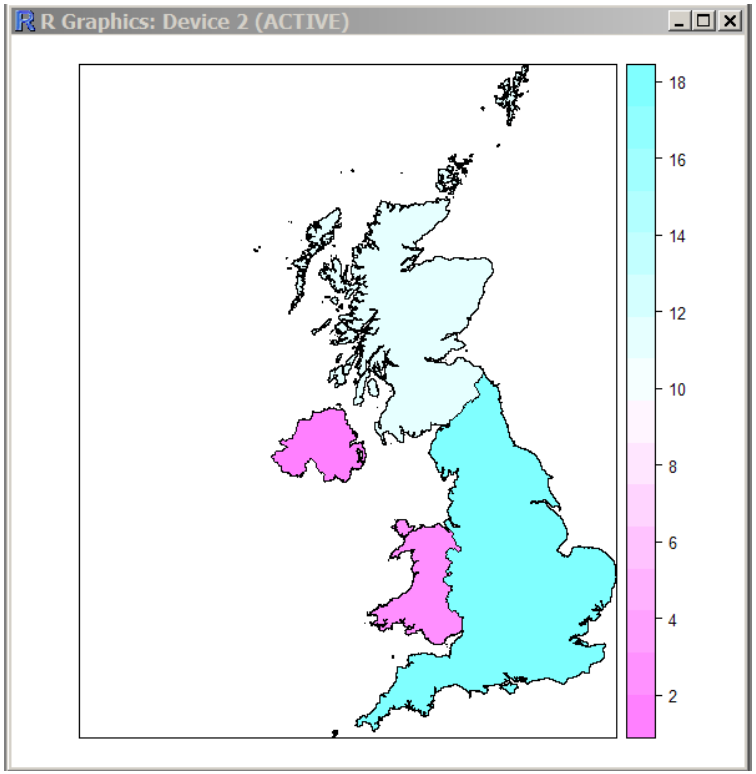


# GADM地理数据库

- 安装sp包
- 用library函数加载sp包

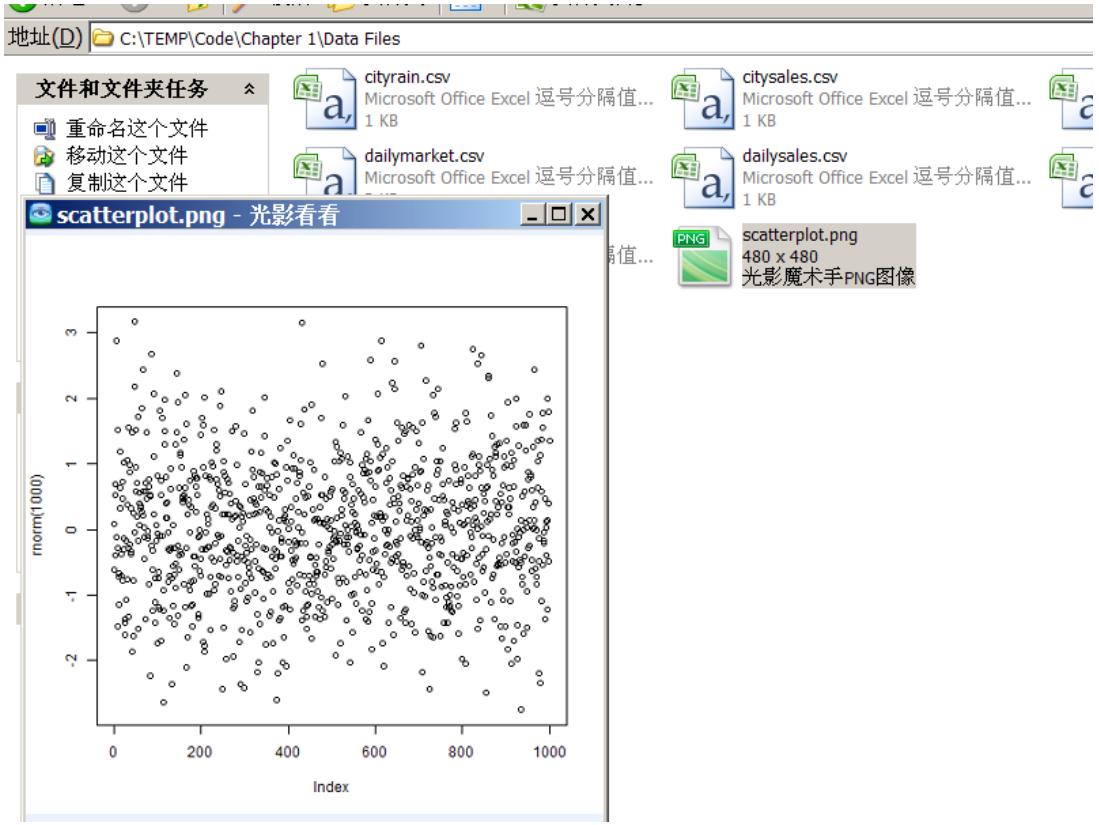
# 用sp包画英国地图

```
library(sp)
load(url("http://gadm.org/data/rda/GBR_adm1.RData"))
spplot(gadm,"Shape_Area")
```



# 输出为 图形文件

```
png("scatterplot.png")
plot(rnorm(1000))
dev.off()
```



# 输出选项

```
png("scatterplot.png",
height=600,
width=600)
```

```
png("scatterplot.png",
height=4,
width=4,
units="in")
```

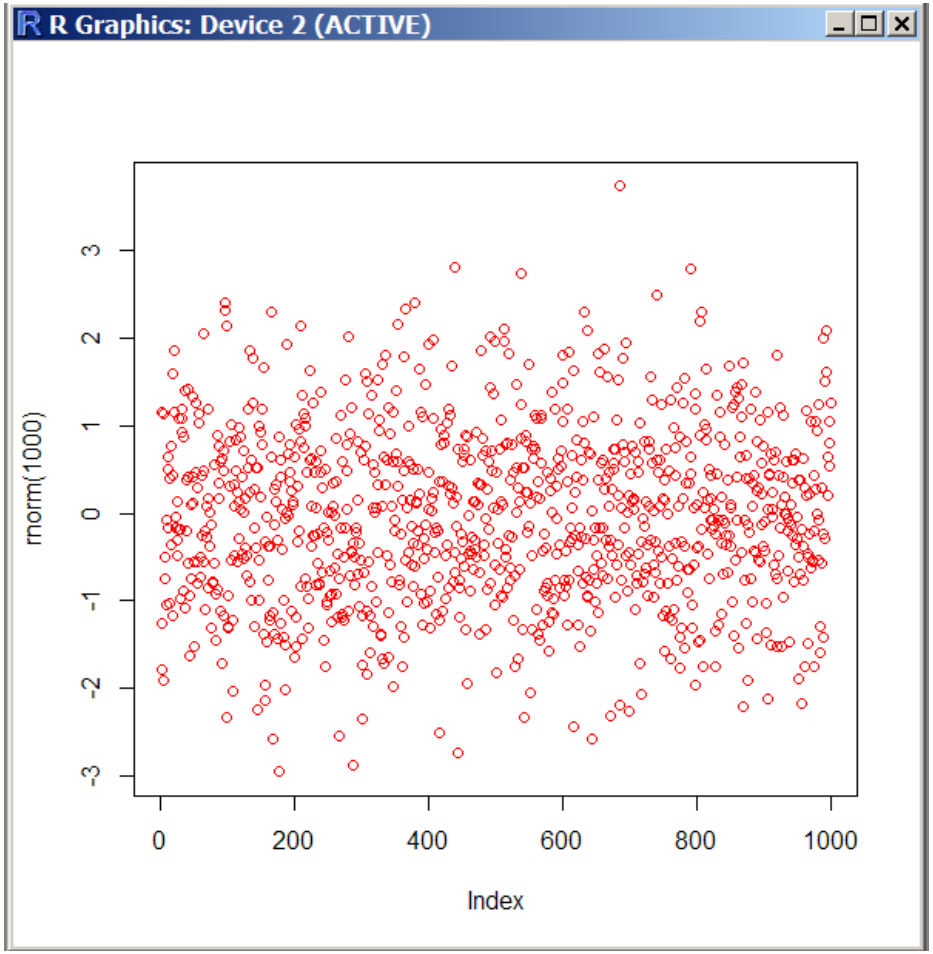
```
png("scatterplot.png",
res=600)
```

```
pdf("scatterplot.pdf")
```

# 设置图形要素的颜色（回顾）

设置点的颜色

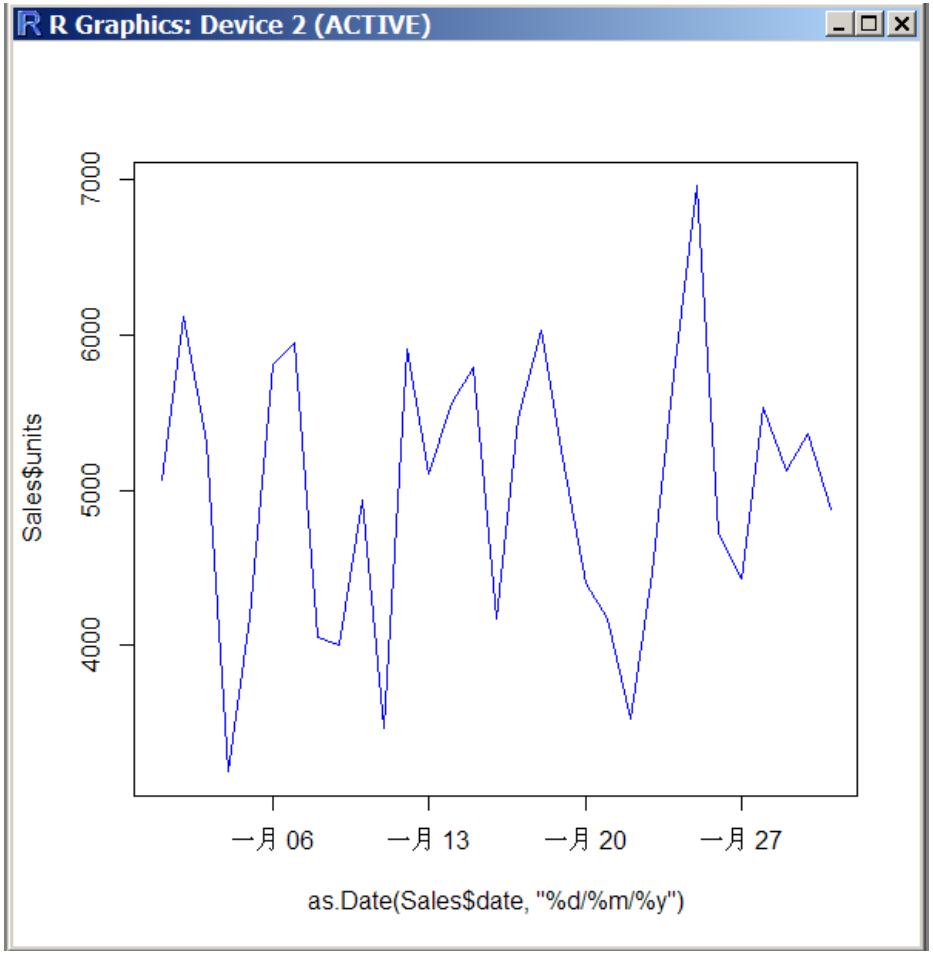
```
plot(rnorm(1000),col="red")
```



# 设置图形要素的颜色（回顾）

画线的颜色

```
sales <-
    read.csv("dailysales.csv",h
    eader=TRUE)
plot(sales$units~as.Date(sale
    s$date,"%d/%m/%y"),
type="l", #Specify type of
    plot as l for line
col="blue")
```



# 设置图形要素的颜色（回顾）

- Plot函数中，使用col=参数来决定要素的颜色
- 如果不指定plot type，颜色加在散点上，如果指定了plot type，例如line，则颜色加在线上
- 其它函数，例如barplot() 和 histogram() 也使用col=参数影响颜色

# 颜色的表达

- 可以使用颜色名
- 使用colors() 函数列出全部颜色名，约有660种

```
> colors()
  [1] "white"              "aliceblue"          "antiquewhite"
  [4] "antiquewhite1"      "antiquewhite2"      "antiquewhite3"
  [7] "antiquewhite4"      "aquamarine"         "aquamarine1"
 [10] "aquamarine2"        "aquamarine3"        "aquamarine4"
 [13] "azure"              "azure1"             "azure2"
 [16] "azure3"             "azure4"             "beige"
 [19] "bisque"             "bisque1"            "bisque2"
 [22] "bisque3"            "bisque4"            "black"
 [25] "blanchedalmond"     "blue"               "blue1"
 [28] "blue2"              "blue3"              "blue4"
 [31] "blueviolet"         "brown"              "brown1"
 [34] "brown2"             "brown3"             "brown4"
 [37] "burlywood"          "burlywood1"         "burlywood2"
 [40] "burlywood3"         "burlywood4"         "cadetblue"
 [43] "cadetblue1"         "cadetblue2"         "cadetblue3"
 [46] "cadetblue4"         "chartreuse"         "chartreuse1"
 [49] "chartreuse2"        "chartreuse3"        "chartreuse4"
 [52] "chocolate"          "chocolate1"         "chocolate2"
 [55] "chocolate3"         "chocolate4"         "coral"
 [58] "coral1"             "coral2"             "coral3"
 [61] "coral4"             "cornflowerblue"     "cornsilk"
 [64] "cornsilk1"          "cornsilk2"          "cornsilk3"
 [67] "cornsilk4"          "cyan"               "cyan1"
 [70] "cyan2"              "cyan3"              "cyan4"
 [73] "darkblue"           "darkcyan"           "darkgoldenrod"
 [76] "darkgoldenrod1"     "darkgoldenrod2"     "darkgoldenrod3"
 [79] "darkgoldenrod4"     "darkgray"           "darkgreen"
 [82] "darkgrey"           "darkkhaki"          "darkmagenta"
 [85] "darkolivegreen"     "darkolivegreen1"    "darkolivegreen2"
 [88] "darkolivegreen3"    "darkolivegreen4"    "darkorange"
```

# 使用数值表达颜色

- col=n
- n为当前调色板上的颜色值
- 缺省调色板下，1表示黑色，2表示红色，0表示背景色
- palette() 函数观看当前调色板

```
>
> palette()
[1] "black"   "red"     "green3"  "blue"    "cyan"    "magenta" "yellow"  "gray"
>
```

# 改变缺省调色板

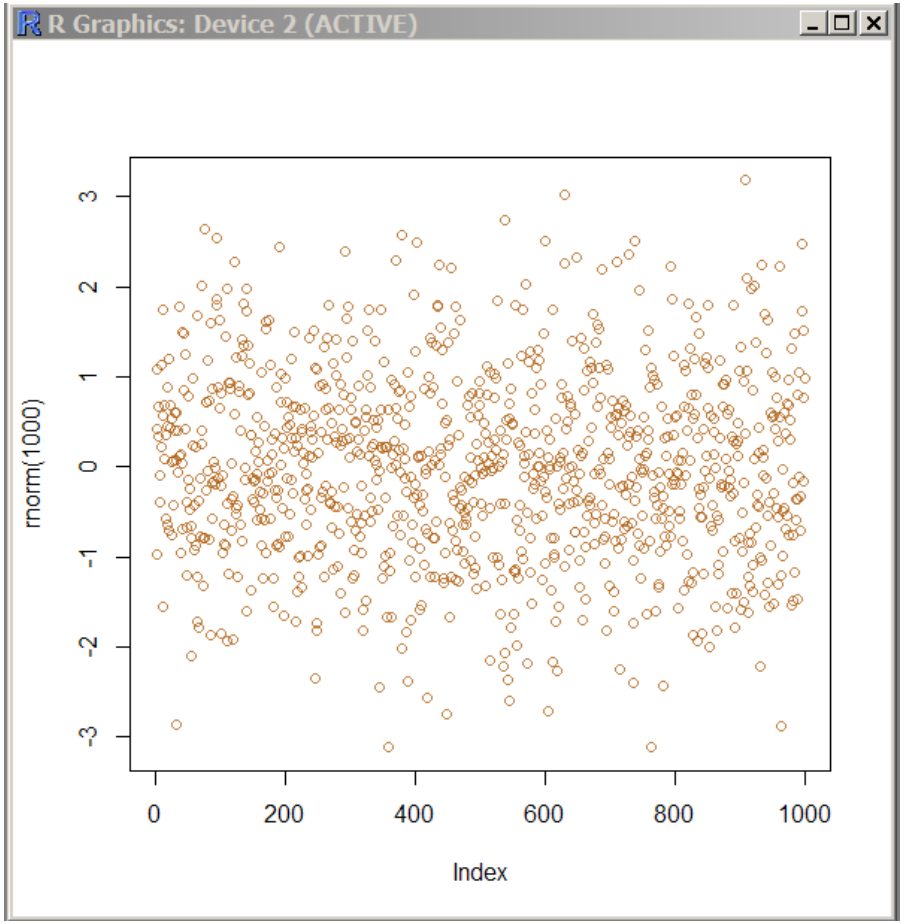
```
>
> palette(c("red","blue","green","orange"))
> palette()
[1] "red"    "blue"   "green"  "orange"
> palette("default")
> palette()
[1] "black"   "red"     "green3"  "blue"    "cyan"    "magenta" "yellow"  "gray"
>
```

# 十六进制表达的颜色

plot(rnorm(1000),col="#AC5500BB")

红色，绿色，蓝色，alpha（透明度）

```
> rgb(0.5,0.5,0.5,0.5)
[1] "#80808080"
> rgb(0.5,0.5,0.5,0.2)
[1] "#80808033"
> 
```



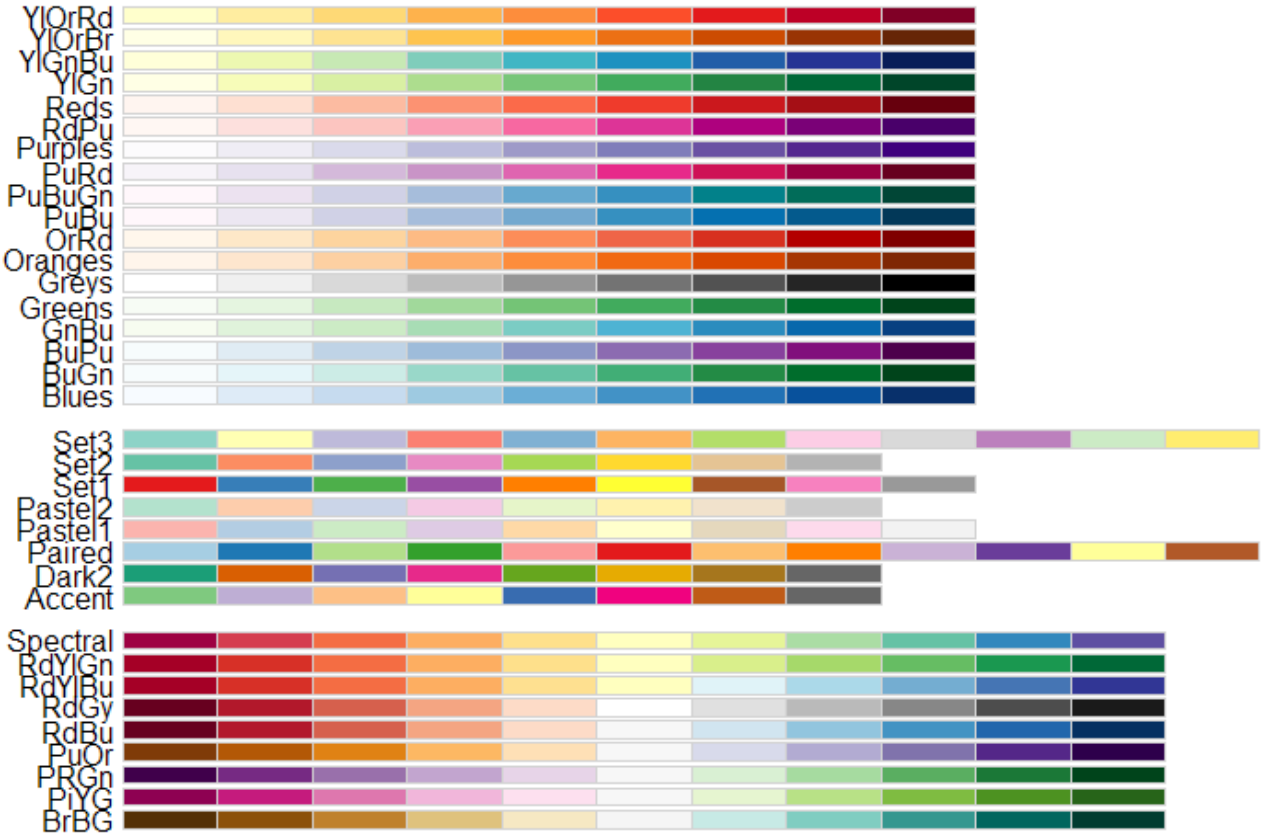
# heat.colors()

```
> heat.colors(5)
[1] "#FF0000FF" "#FF5500FF" "#FFAA00FF" "#FFFF00FF" "#FFFF80FF"
> heat.colors(15)
 [1] "#FF0000FF" "#FF1700FF" "#FF2E00FF" "#FF4600FF" "#FF5D00FF" "#FF7400FF"
 [7] "#FF8B00FF" "#FFA200FF" "#FFB900FF" "#FFD100FF" "#FFE800FF" "#FFFF00FF"
[13] "#FFFF2AFF" "#FFFF80FF" "#FFFFD5FF"
> heat.colors(25)
 [1] "#FF0000FF" "#FF0E00FF" "#FF1C00FF" "#FF2A00FF" "#FF3900FF" "#FF4700FF"
 [7] "#FF5500FF" "#FF6300FF" "#FF7100FF" "#FF8000FF" "#FF8E00FF" "#FF9C00FF"
[13] "#FFAA00FF" "#FFB800FF" "#FFC600FF" "#FFD500FF" "#FFE300FF" "#FFF100FF"
[19] "#FFFF00FF" "#FFFF15FF" "#FFFF40FF" "#FFFF6AFF" "#FFFF95FF" "#FFFFBFFF"
[25] "#FFFFEAFF"
>
```

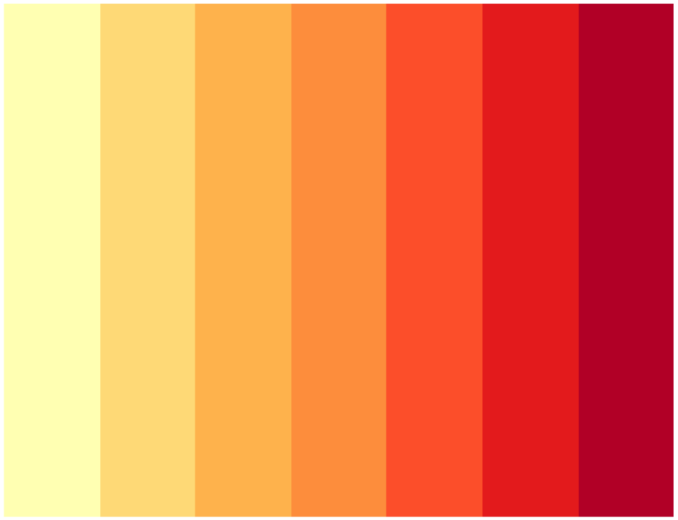
# 直观的调色板控制包RColorBrewer

library(RColorBrewer)

display.brewer.all()



```
> brewer.pal(7,"YlOrRd")
[1] "#FFFFB2" "#FED976" "#FEB24C" "#FD8D3C" "#FC4E2A" "#E31A1C" "#B10026"
> display.brewer.pal(7,"YlOrRd")
>
```



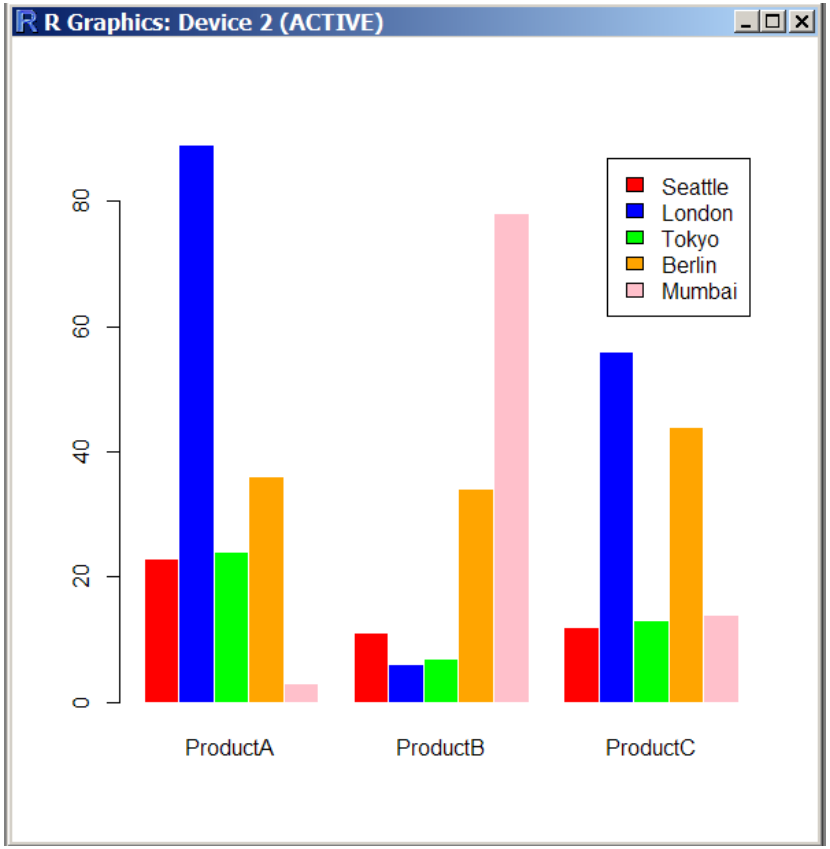
YlOrRd (sequential)

# 颜色向量：使用多颜色画图

```
sales<-
    read.csv("citysales.csv",header=TRUE)
barplot(as.matrix(sales[,2:4]), beside=T,
legend=sales$City,
col=c("red","blue","green","orange","pink"),
border="white")
```

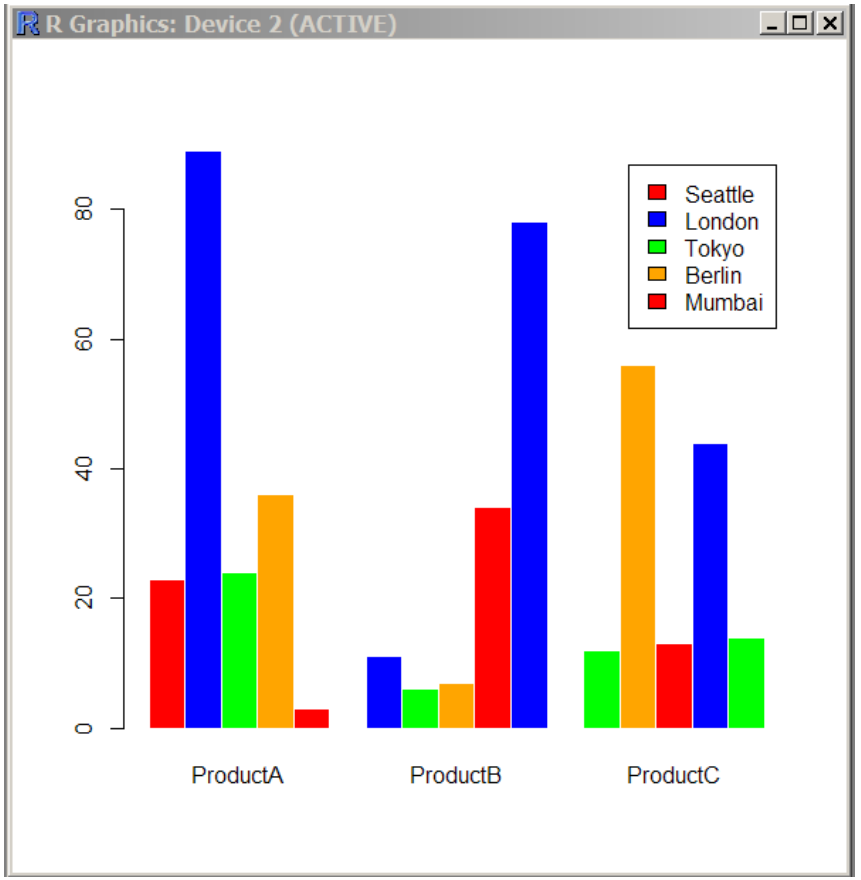
颜色数和样本数要相等

```
> sales
     City ProductA ProductB ProductC
1 Seattle        23        11        12
2  London        89         6        56
3   Tokyo        24         7        13
4  Berlin        36        34        44
5  Mumbai         3        78        14
>
```



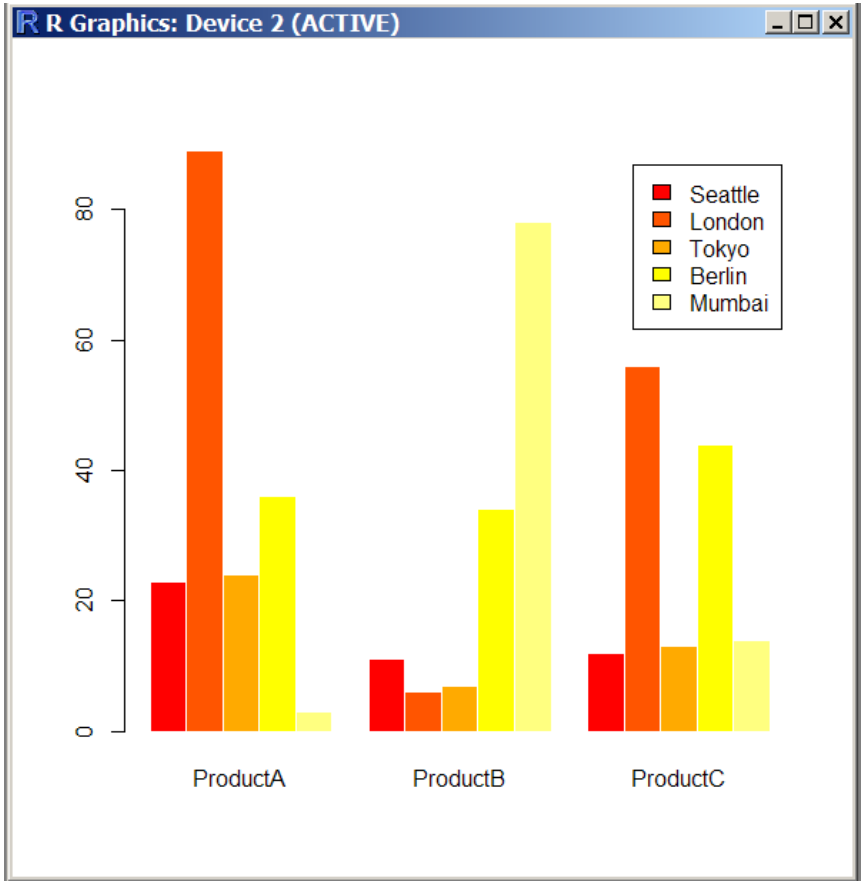
# 循环颜色

```
barplot(as.matrix(sales[,2:4]), beside=T,
legend=sales$City,
col=c("red","blue","green","orange"),
border="white")
```



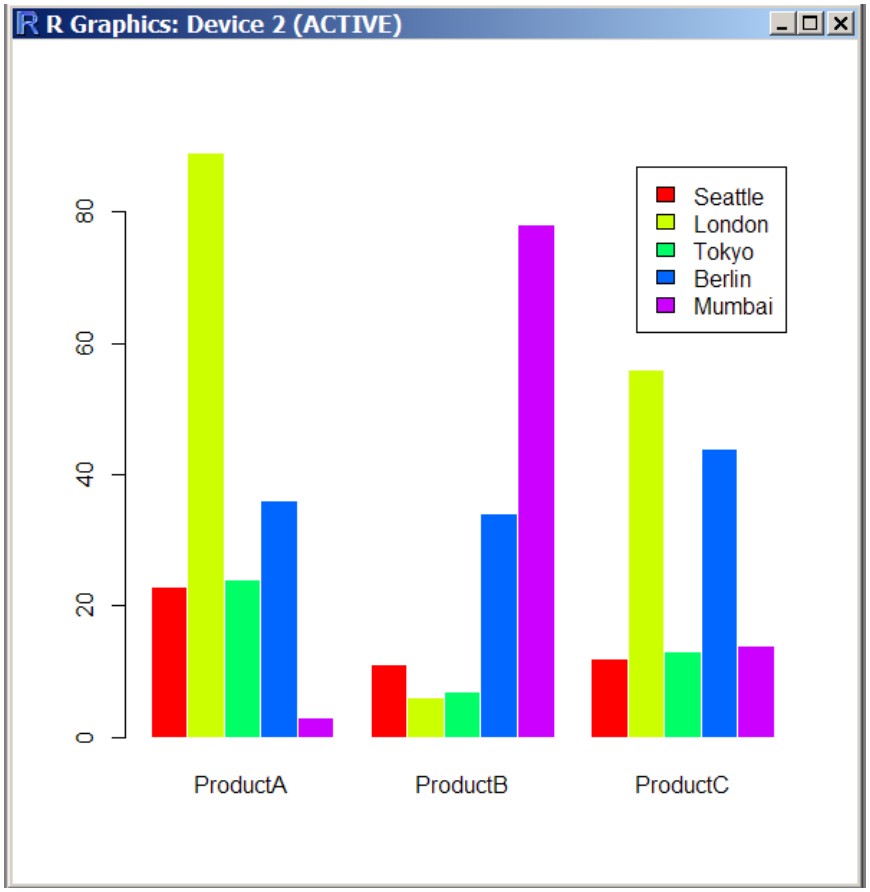
# 利用heat.colors()

```
barplot(as.matrix(sales[,2:4]), beside=T,
legend=sales$City,
col=heat.colors(length(sales$City)),
border="white")
```



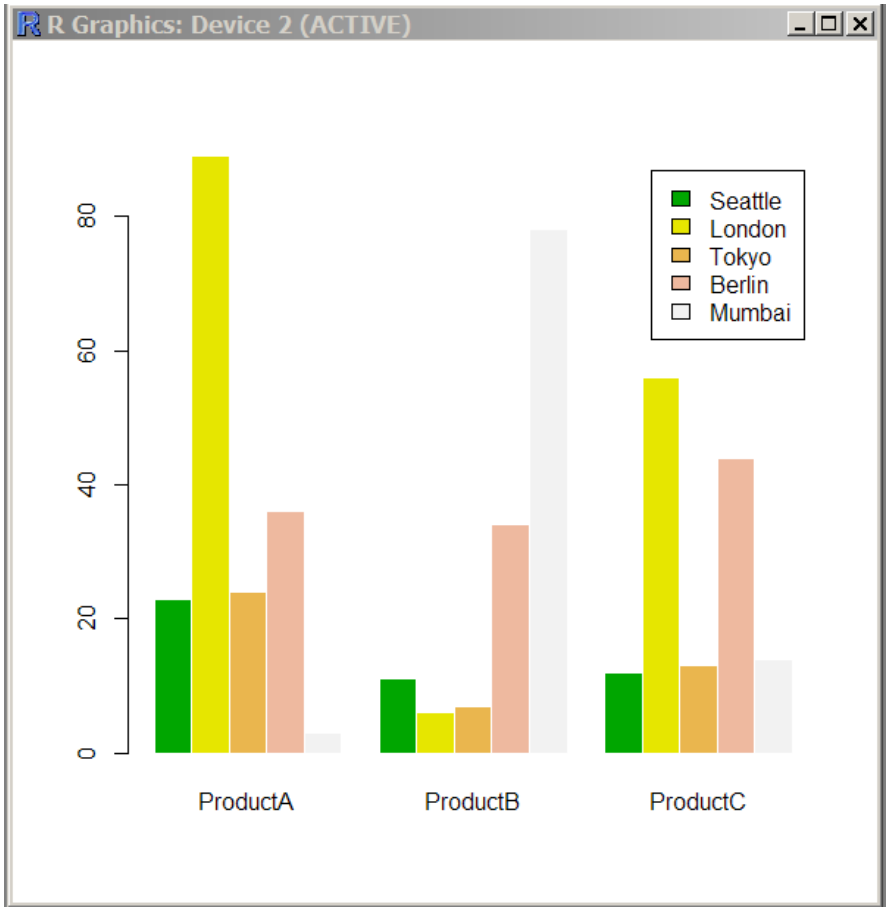
# 利用rainbow()

```
barplot(as.matrix(sales[,2:4]), beside=T,
legend=sales$City,
col=rainbow(length(sales$City)),
border="white")
```



# 利用terrain.colors()

```
barplot(as.matrix(sales[,2:4]), beside=T,
legend=sales$City,
col=terrain.colors(length(sales$City)),
border="white")
```



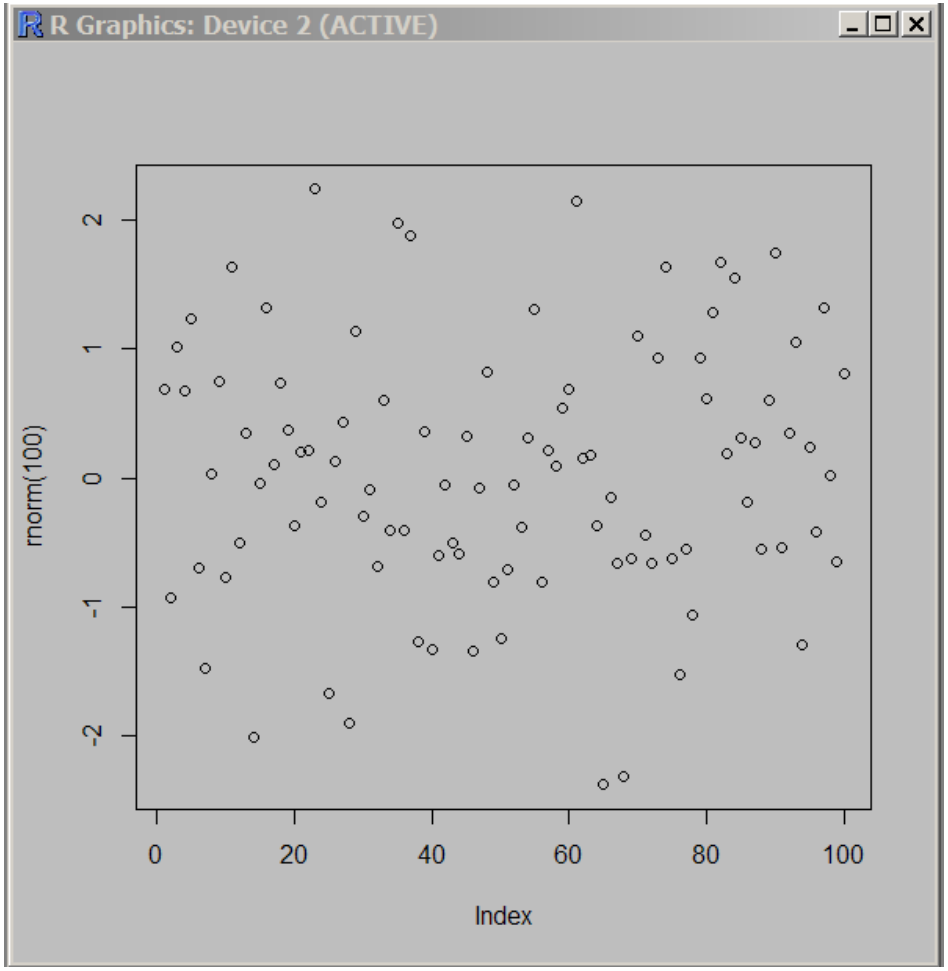
# 其它颜色函数

- cm.colors()
- topo.colors()

# 设置背景颜色

```
par(bg="gray")
plot(rnorm(100))
```

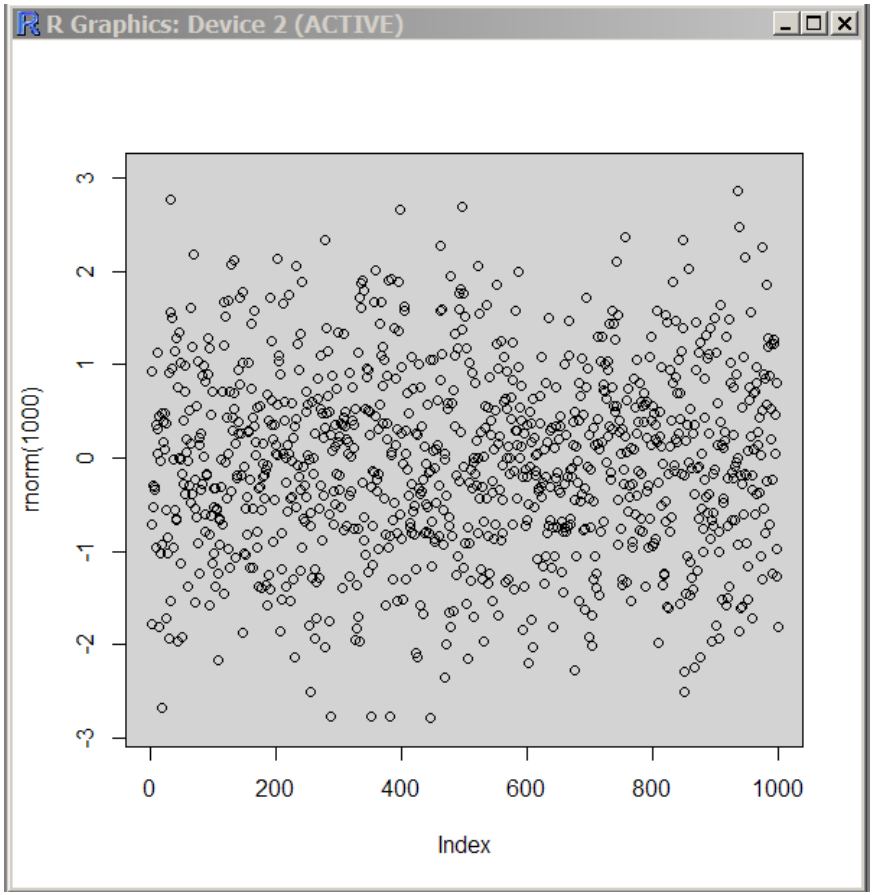
par()的作用直到画板被关闭为止



# 只设置坐标系内的背景颜色

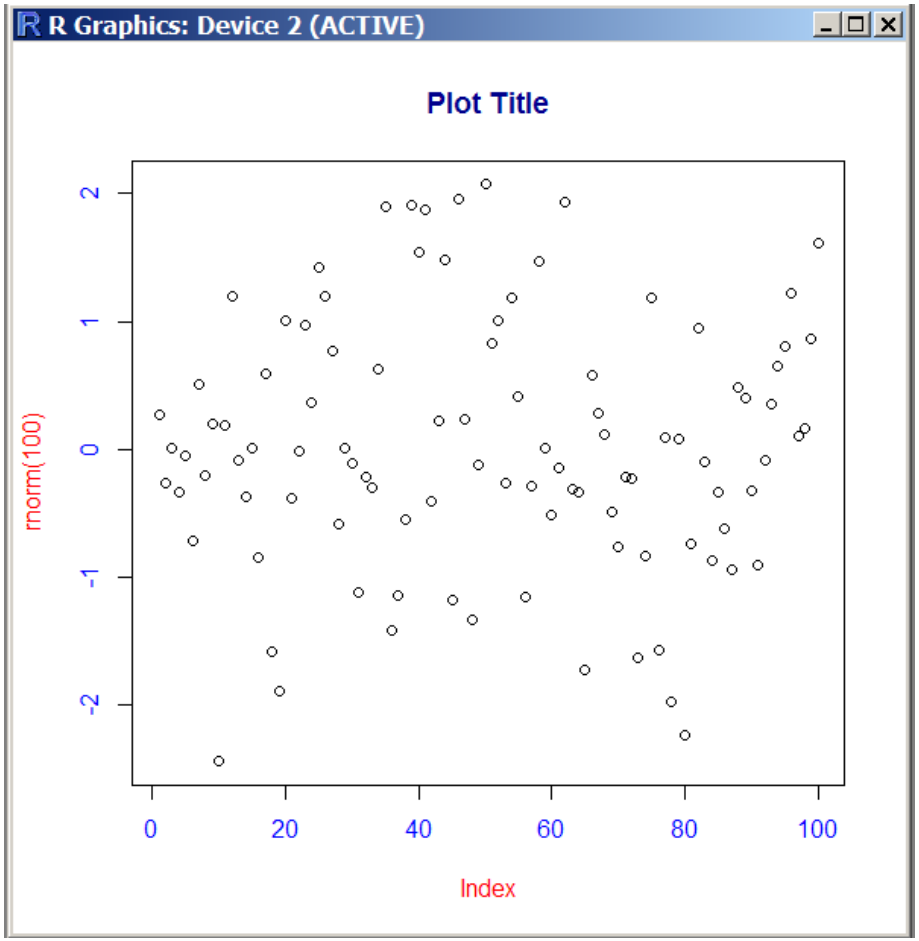
```
plot(rnorm(1000),type="n")
x<-par("usr")
rect(x[1],x[3],x[2],x[4],col="lightgra
     y ")
points(rnorm(1000))
```

```
> x
[1]  -38.960000 1039.960000   -3.092571    3.265140
>
```



# 设置标题、坐标轴标号等颜色

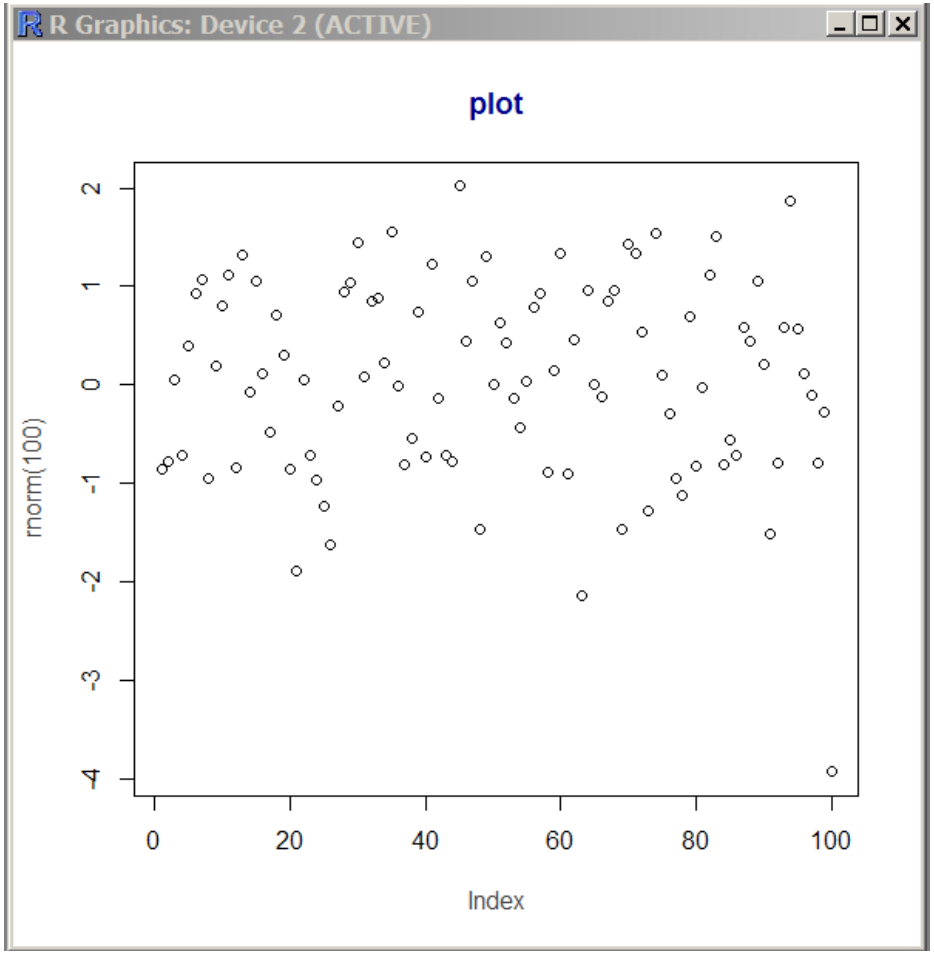
```
plot(rnorm(100),
main="Plot Title",
col.axis="blue",
col.lab="red",
col.main="darkblue")
```



# 使用par()设置

```
par(col.axis="black",
col.lab="#444444",
col.main="darkblue")
plot(rnorm(100),main="plot")
```

par()的作用直至下一条par()设置命令，或者重新开一个图形设备



# 使用title()函数

title("Sales Figures for 2010", col.main="blue")

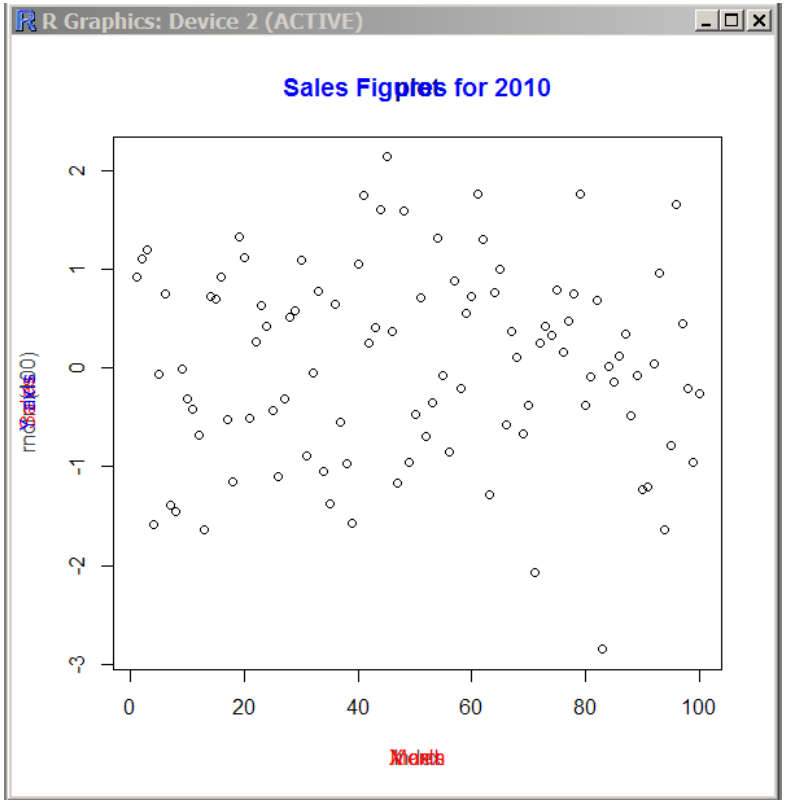
title(xlab="Month",ylab="Sales",col.lab="red")

title(xlab="X axis",col.lab="red")

title(ylab="Y axis",col.lab="blue")

可以覆盖缺省的标题设置

可以用于设置画图参数

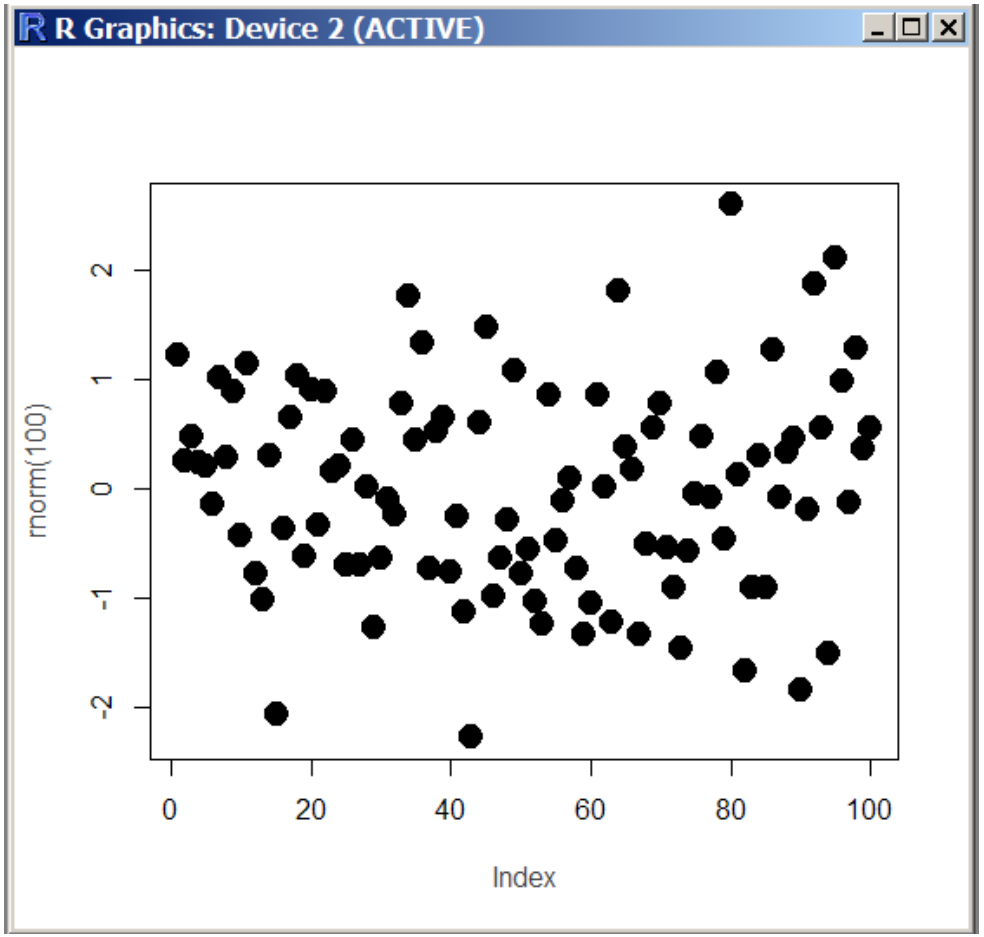


# 字体设置

- Win下能用的字体非常有限
- 用par(family=“serif”,font=2) 设置Times New Roman字体
- 0 缺省，1 粗体，2 斜体，3 粗斜体

# 设置散点的样式

```
rain<-read.csv("cityrain.csv")
plot(rnorm(100),pch=19,cex=2)
```

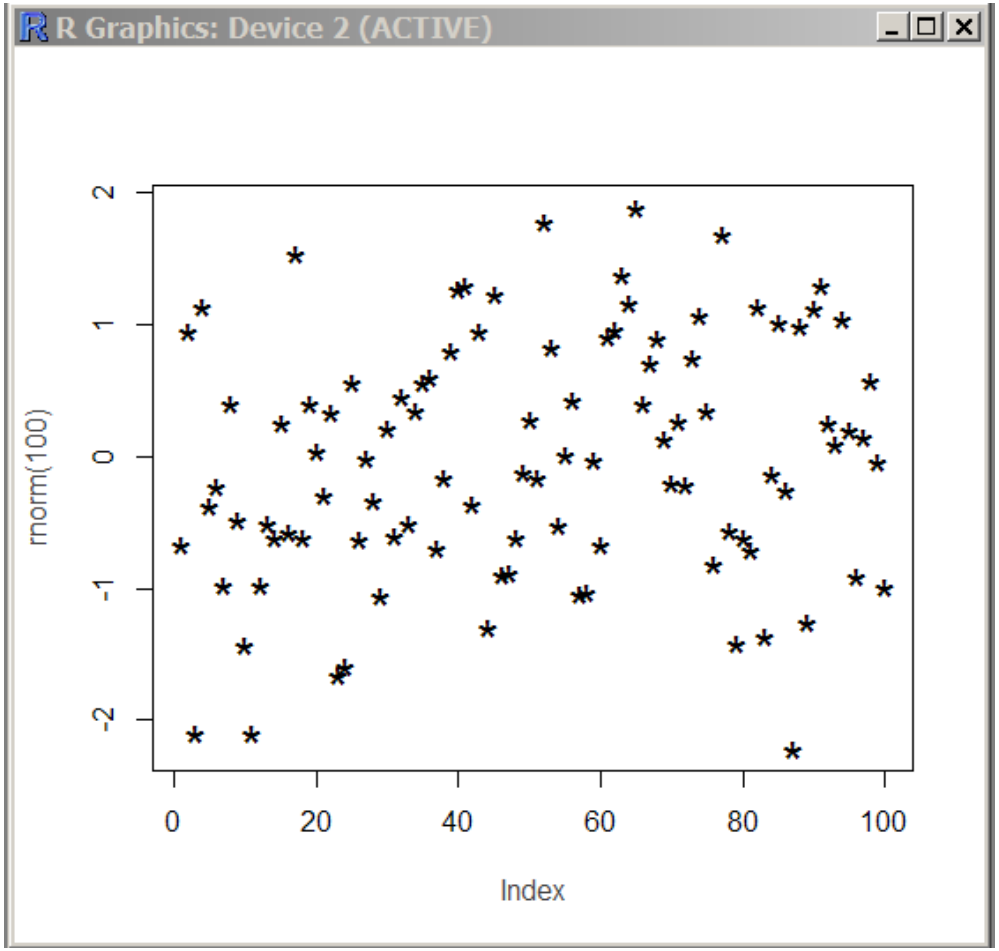


# pch参数

- 代表散点的符号

1 圆圈　2 三角形　3 加号 等等

```
rain<-read.csv("cityrain.csv")
plot(rnorm(100),pch="*",cex=2)
```

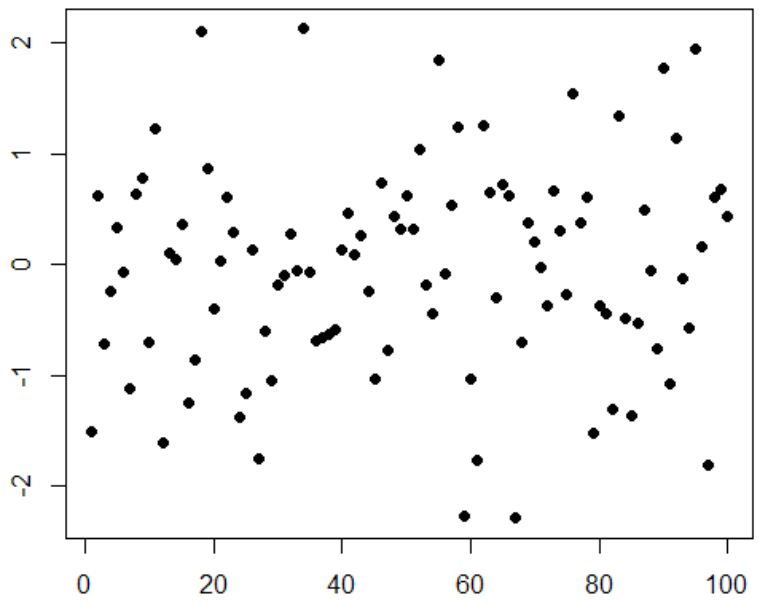


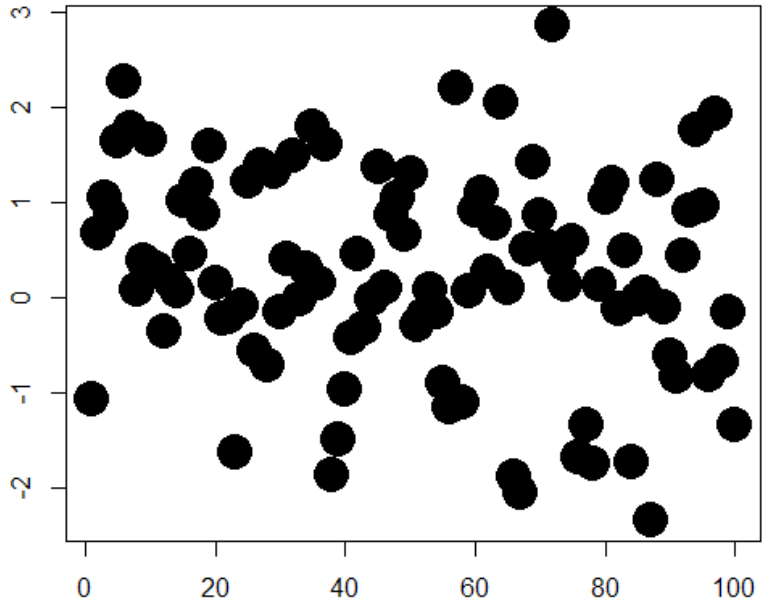

2012.10.18

# cex参数：控制散点的大小

plot(rnorm(100),pch=19,cex=1)

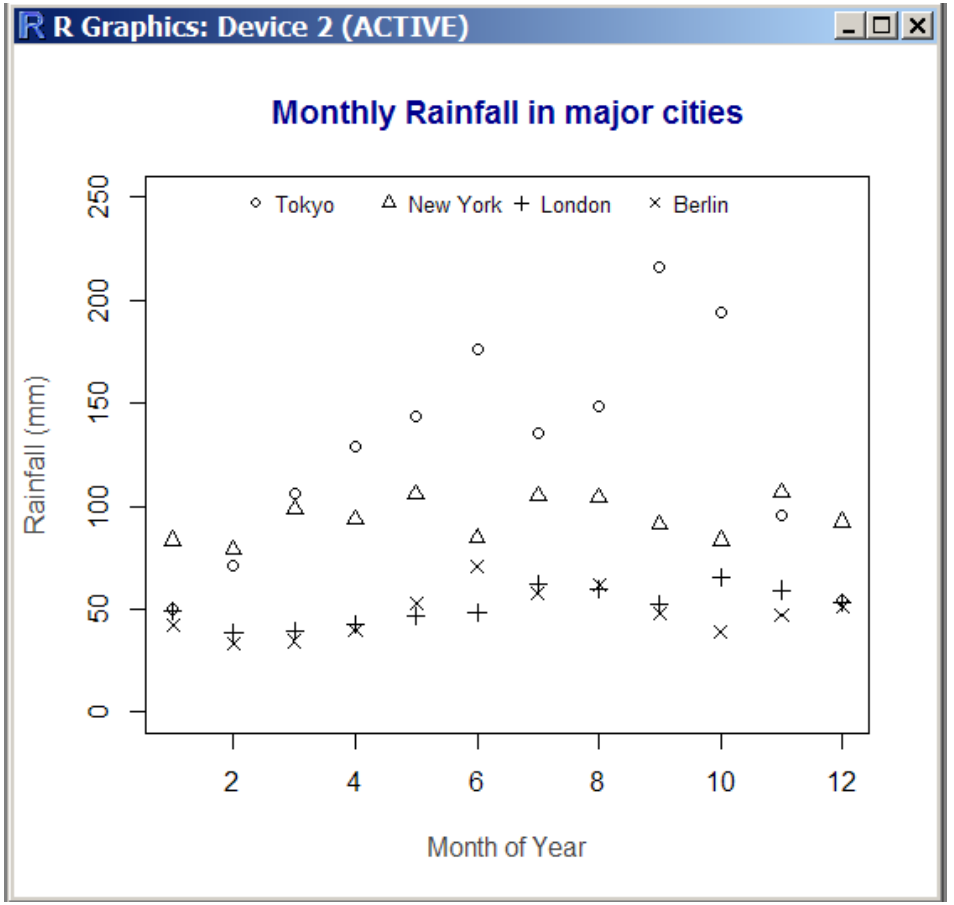
plot(rnorm(100),pch=19,cex=3)



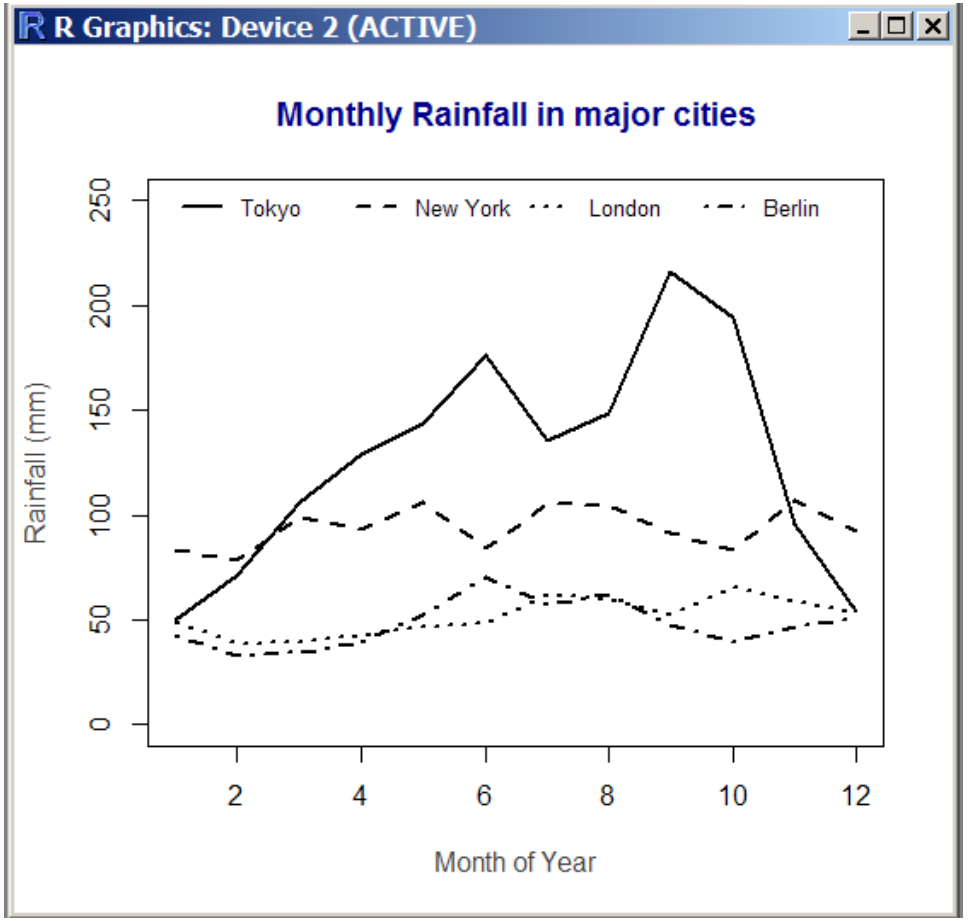
# 例子

```
plot(rain$Tokyo,
ylim=c(0,250),
main="Monthly Rainfall in major cities",
xlab="Month of Year",
ylab="Rainfall (mm)",
pch=1)
points(rain$NewYork,pch=2)
points(rain$London,pch=3)
points(rain$Berlin,pch=4)
legend("top",
legend=c("Tokyo","New York","London","Berlin"),
ncol=4,
cex=0.8,
bty="n",
pch=1:4)
```



# 决定线型与宽度

```
plot(rain$Tokyo,
ylim=c(0,250),
main="Monthly Rainfall in major cities",
xlab="Month of Year",
ylab="Rainfall (mm)",
type="l",
lty=1,
lwd=2)
lines(rain$NewYork,lty=2,lwd=2)
lines(rain$London,lty=3,lwd=2)
lines(rain$Berlin,lty=4,lwd=2)
legend("top",
legend=c("Tokyo","New York","London","Berlin"),
ncol=4,
cex=0.8,
bty="n",
lty=1:4,
lwd=2)
```

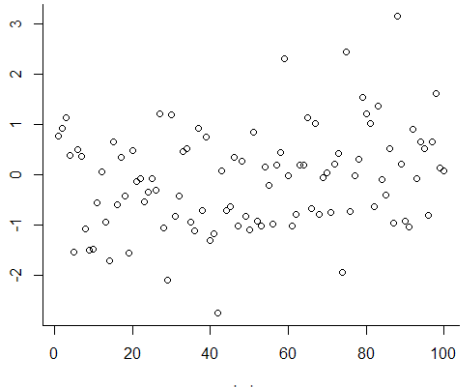


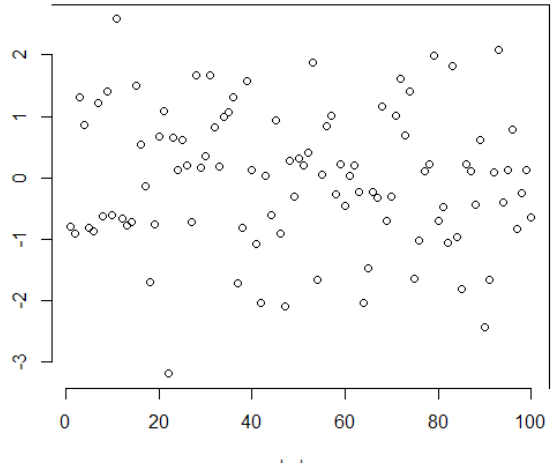
# lty参数：决定线型

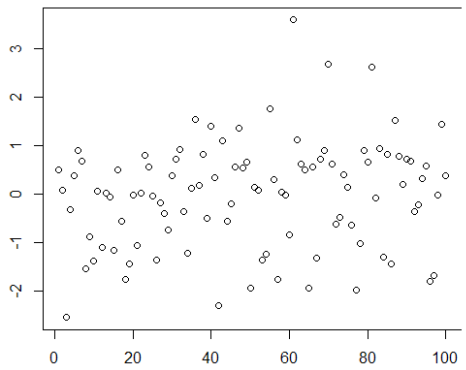
- 0: blank
- 1: solid (default)
- 2: dashed
- 3: dotted
- 4: dotdash
- 5: longdash
- 6: twodash

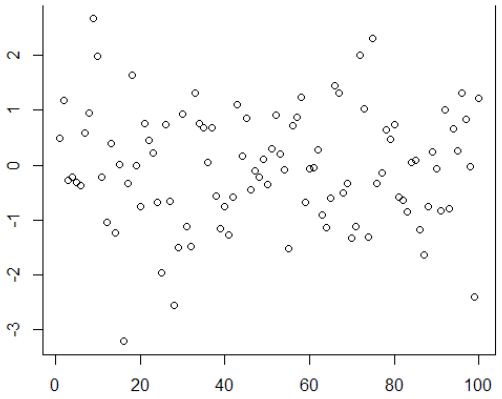
# 用bty参数控制坐标系的风格

```
par(bty="l")
plot(rnorm(100))
par(bty="7")
plot(rnorm(100))
par(bty="c")
plot(rnorm(100))
par(bty="u")
plot(rnorm(100))
```



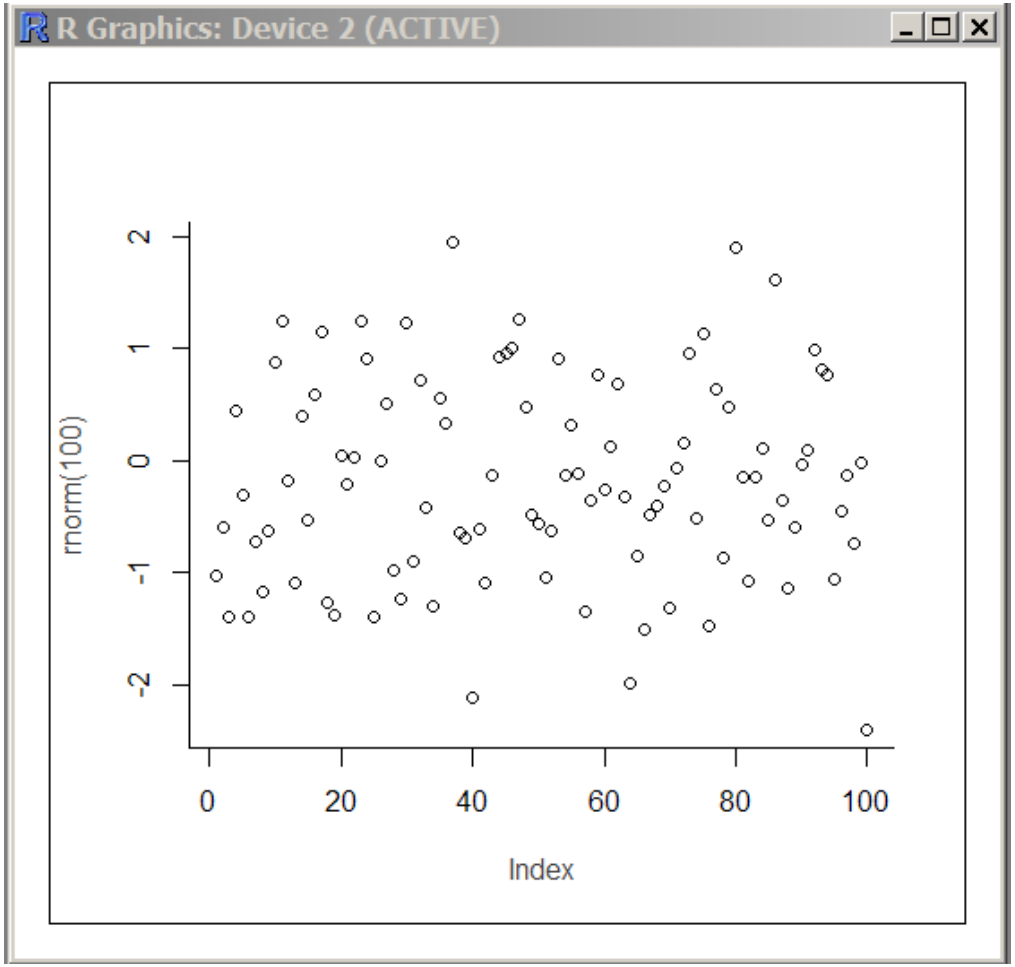



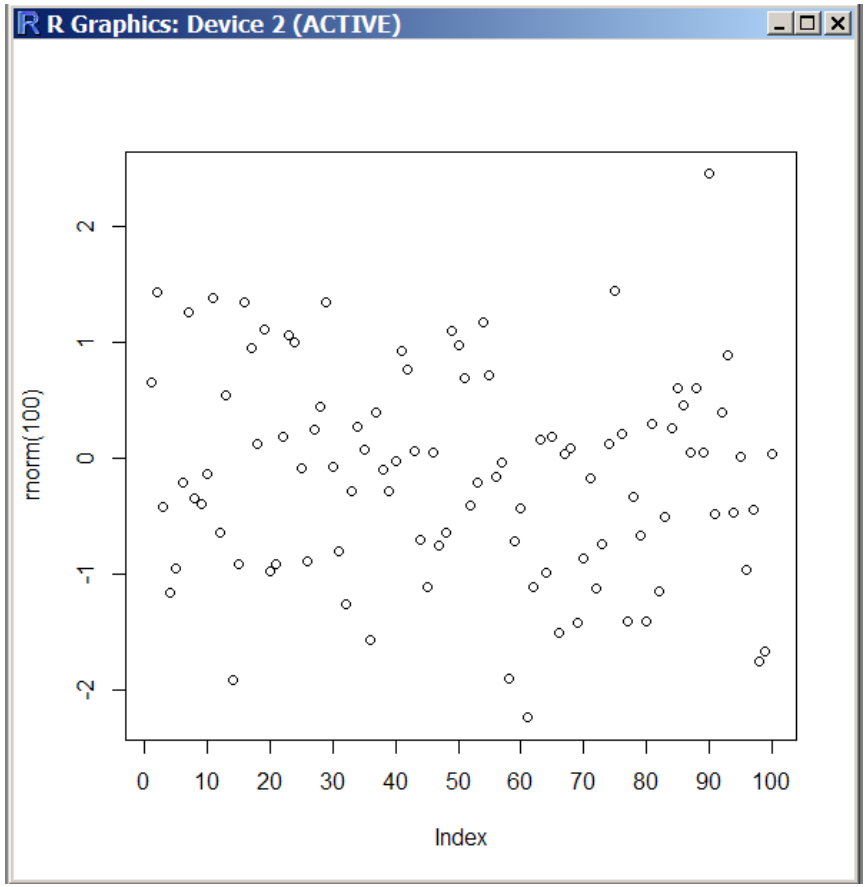
# box()函数

```
par(oma=c(1,1,1,1))
plot(rnorm(100),bty="l")
box(which="figure")
```



# 设置坐标轴刻度

plot(rnorm(100),xaxp=c(0,100,10))

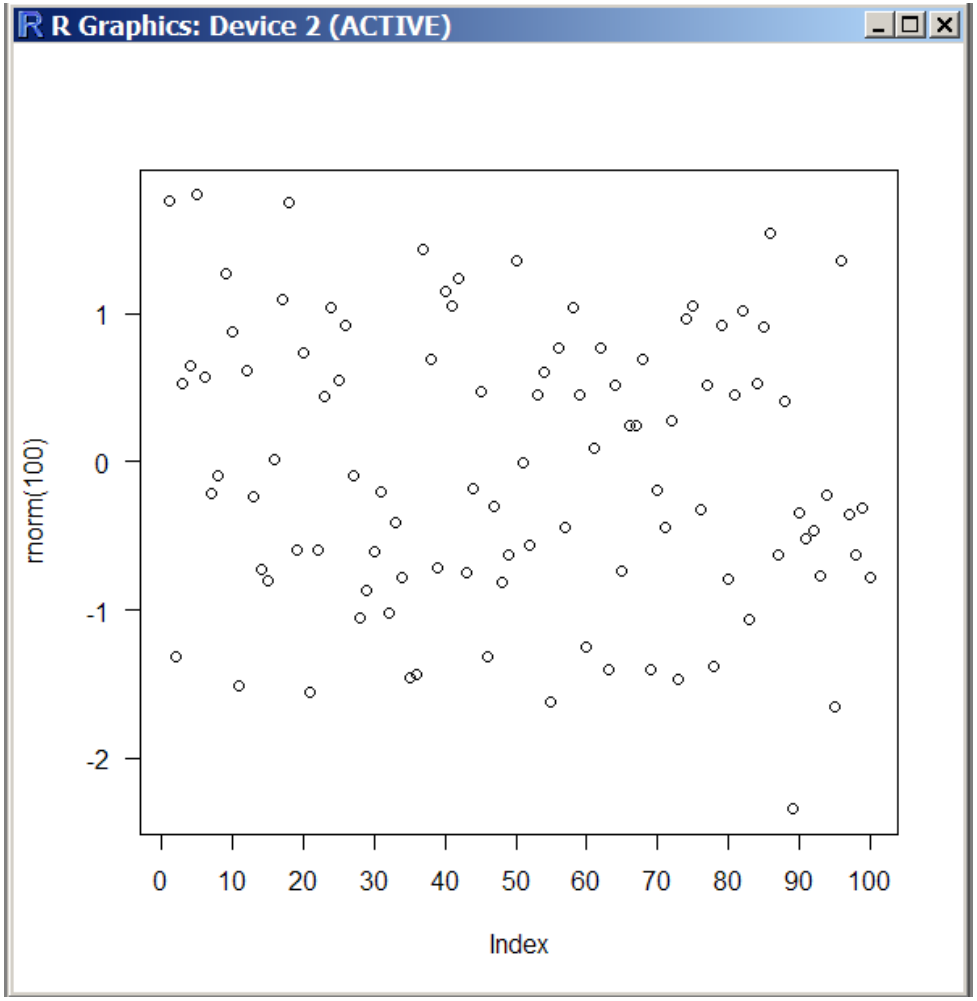


# xaxp与yaxp参数

- 缺省情况下比上限再增加大约4%，并且自动计算刻度
- 可以通过xaxs 来改变上述风格

# las参数：刻度数字的方向

```
par(las=3)
plot(rnorm(100),xaxp=c(0,100,10))
par(las=2)
plot(rnorm(100),xaxp=c(0,100,10))
par(las=1)
plot(rnorm(100),xaxp=c(0,100,10))
```

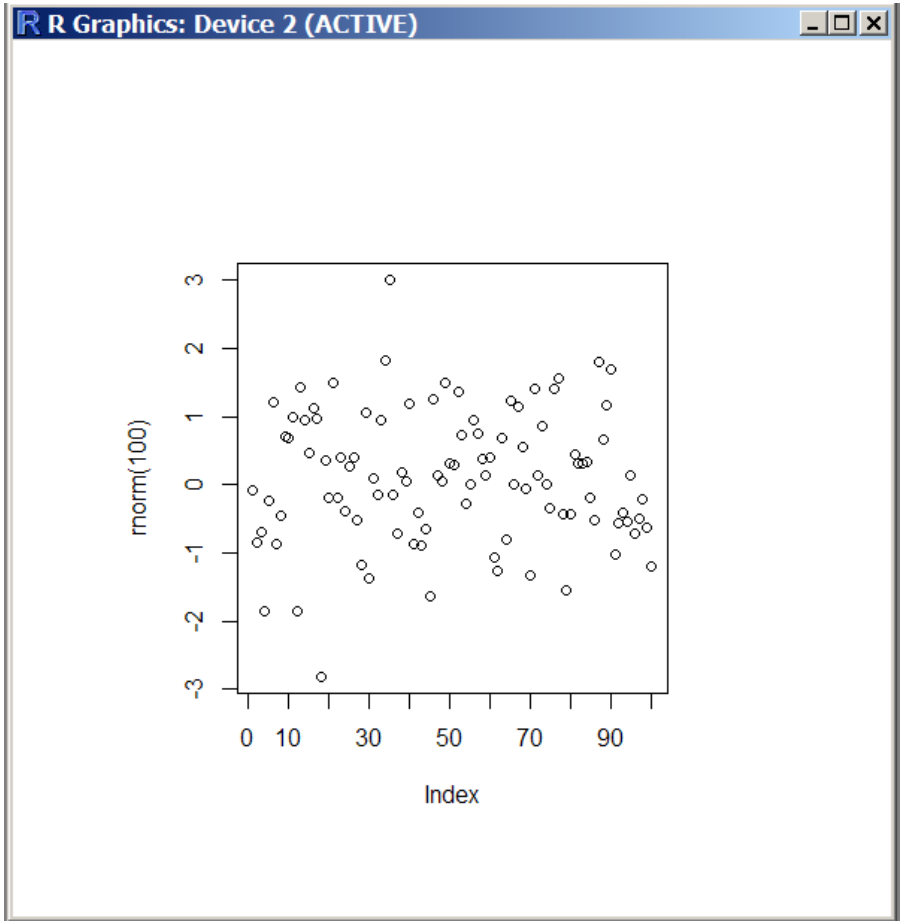



2012.10.18

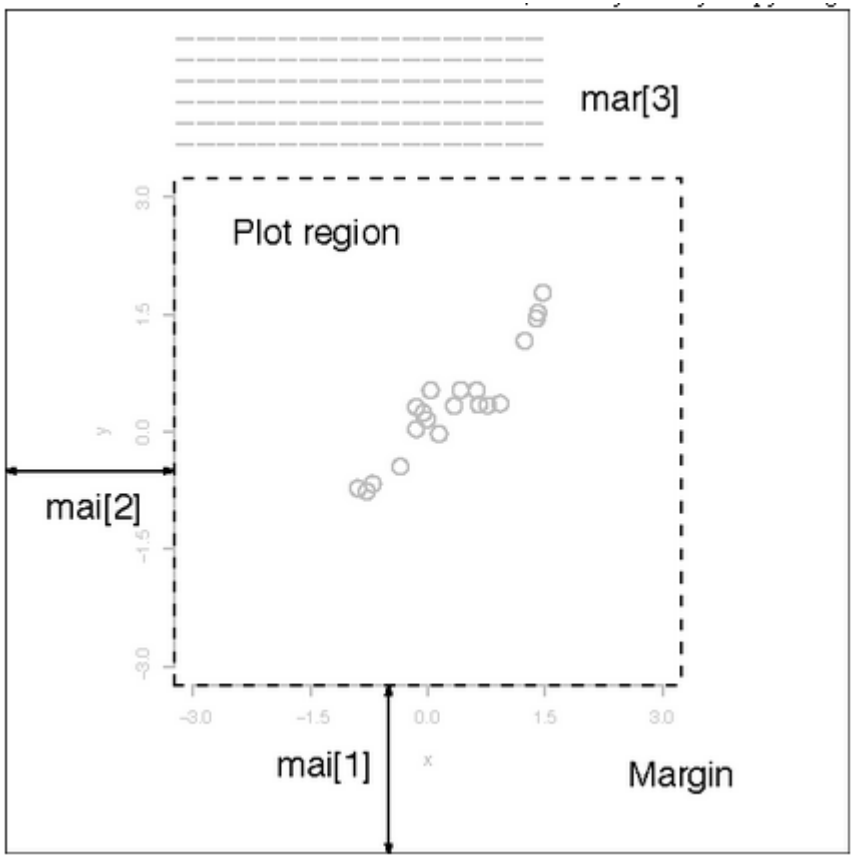
# 设置画图区域大小

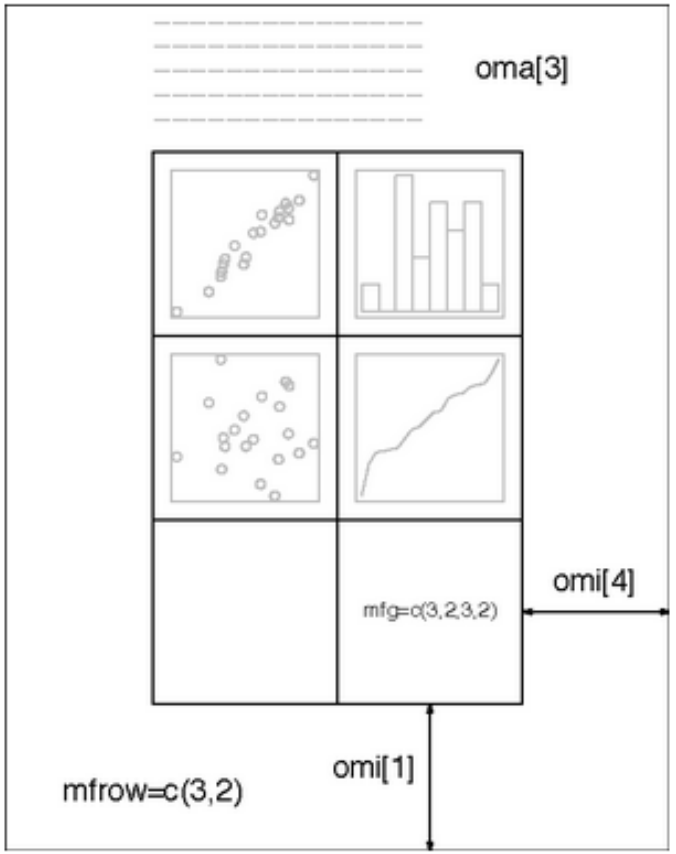
par(fin=c(5,5),

pin=c(3,3))



# mai和omi参数：控制边缘

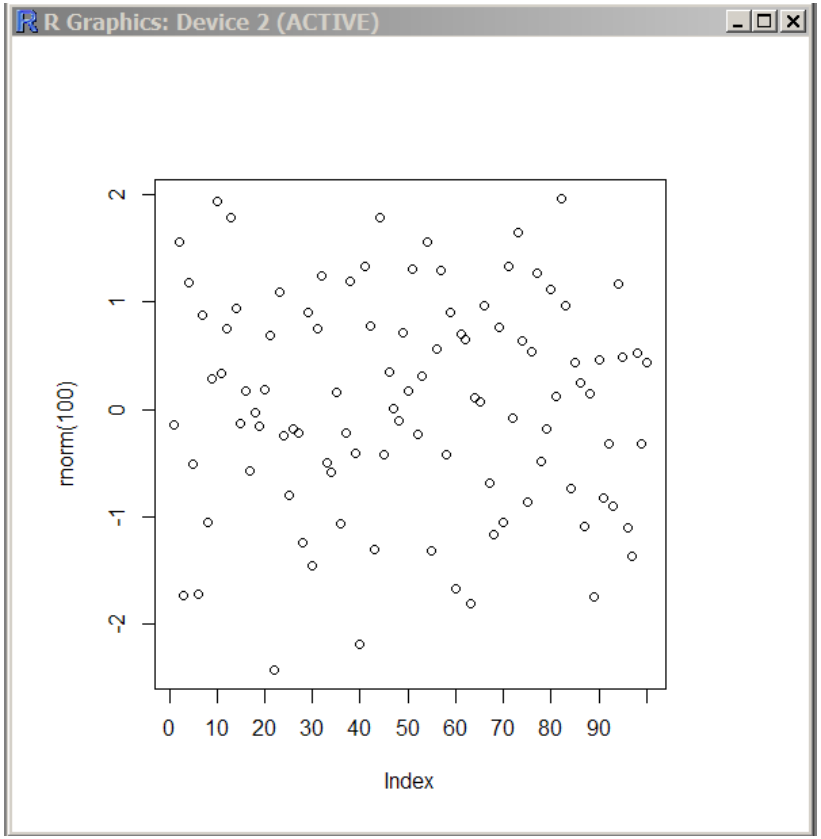


# 控制边缘

par(mai=c(1,1,1,1), omi=c(0.1,0.1,0.1,0.1))

plot(rnorm(100),xaxp=c(0,100,10))

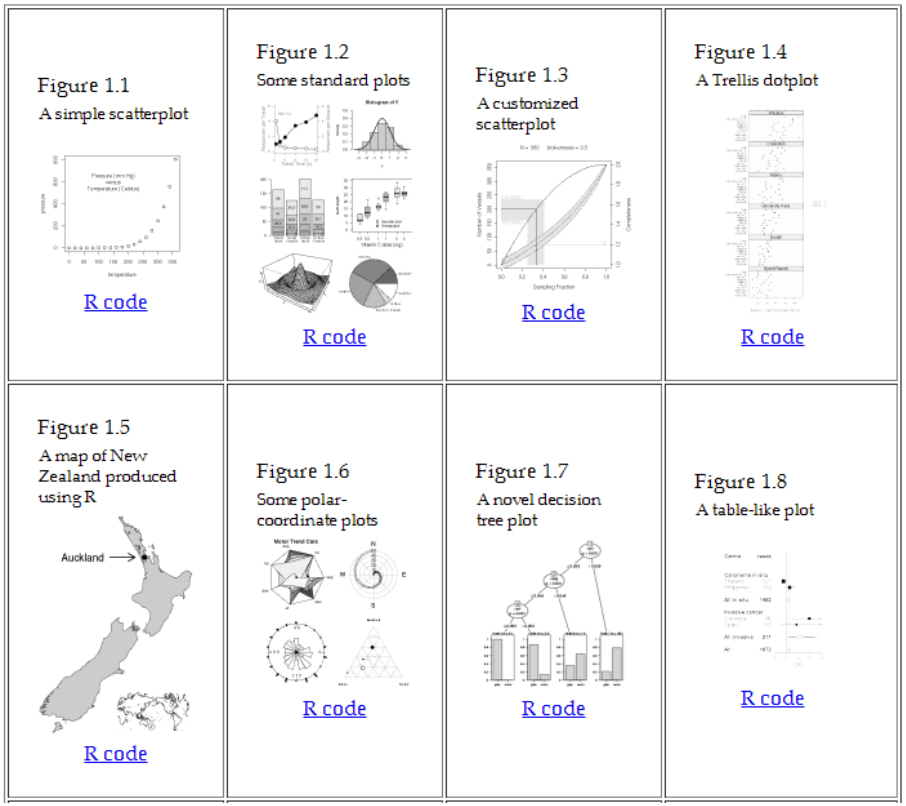



2012.10.18

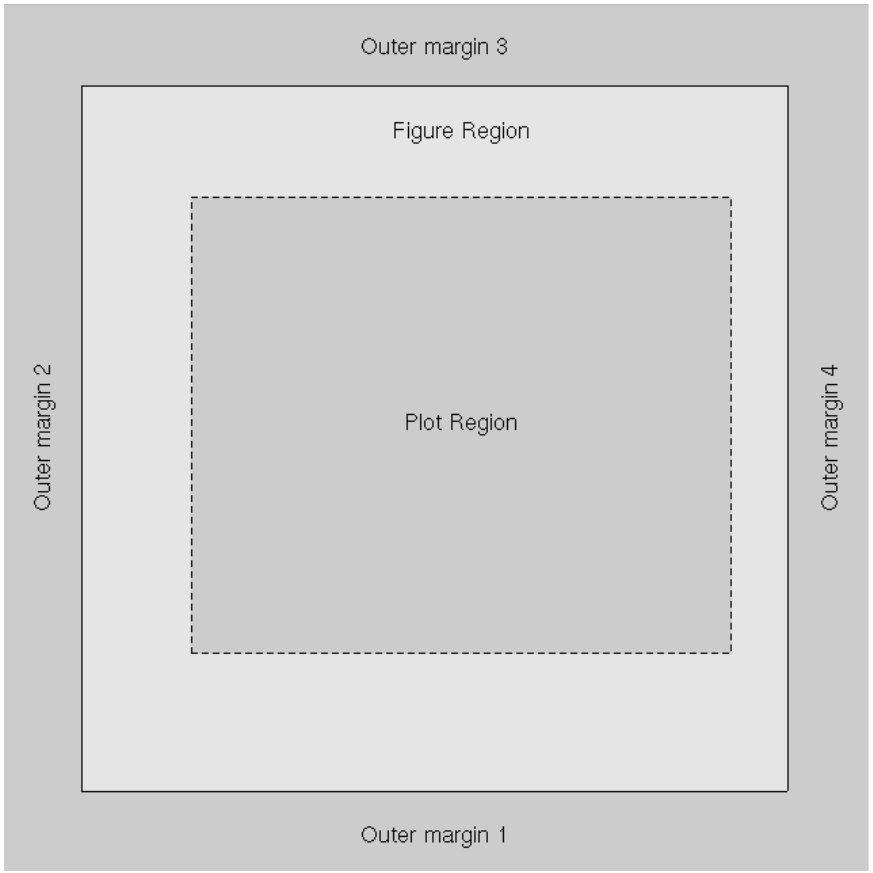
# 介绍一个R画图网站

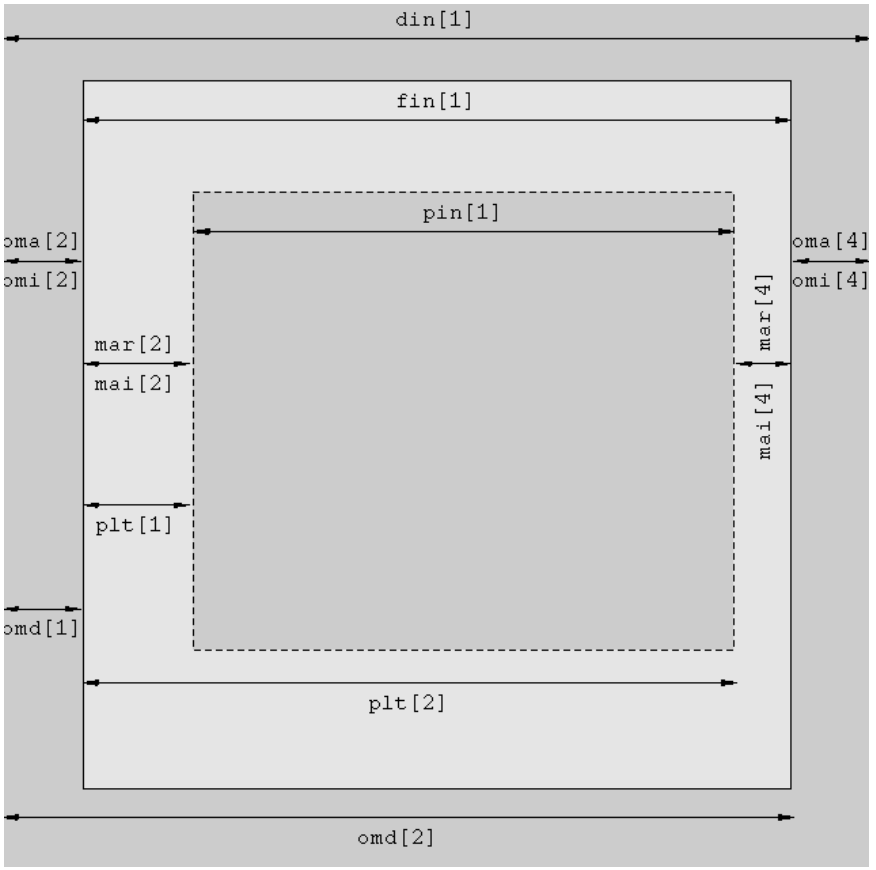
http://www.stat.auckland. ac.nz/~paul/RGraphics/rgraphics.html

第三章解释边缘



# 边缘


Outer margin 3
Figure Region
Plot Region
Outer margin 2
Outer margin 4
Outer margin 1


din[1]
fin[1]
pin[1]
oma[2]
omi[2]
oma[4]
omi[4]
mar[2]
mai[2]
mar[4]
mai[4]
plt[1]
omd[1]
plt[2]
omd[2]

# 维基百科可视化


2012.10.18

# 维基百科可视化

随机条目

帮助
- 帮助
- 社区专页
- 方针与指引
- 互助客栈
- 询问处
- 字词转换
- IRC即时聊天
- 联系我们
- 关于维基百科
- 资助维基百科

工具箱

选择下列任何一个版本的日期点击可以浏览。需要更多帮助请参看Help:页面历史和Help:编辑摘要。

页面信息 | 外部工具：修订历史统计 • 搜索编辑历史 • 监视者人数 • 本月页面浏览统计

选择不同版本：在两个不同版本的圆框分别单击一下，再按最底的“比较被选版本”键以作比较。

说明：（当前）= 与当前修订版本的差别，（先前）= 与前一个修订版本的差别，小 = 小修改，→ = 章节编辑，← = 自动编辑摘要。

（最近 | 最早） 查看（前50个 | 后50个）（20 | 50 | 100 | 250 | 500）

比较被选版本

- （当前 | 先前） 2012年10月29日（一）03:45 1.164.209.170（讨论） . . （91,188字节） (-21) . . *(→评价)* （撤销）
- （当前 | 先前） 2012年10月29日（一）03:08 61.70.83.145（讨论） . . （91,209字节） (-4) . . （撤销）
- （当前 | 先前） 2012年10月29日（一）03:07 61.70.83.145（讨论） . . （91,213字节） (+48) . . （撤销）
- （当前 | 先前） 2012年10月29日（一）03:04 61.70.83.145（讨论） . . （91,165字节） (+86) . . （撤销）
- （当前 | 先前） 2012年10月24日（三）06:24 EmausBot（讨论 | 贡献） 小 . . （91,079字节） (+37) . . *(r2.7.2+)(机器人添加：udm:Шекспир, Уильям)* （撤销）
- （当前 | 先前） 2012年9月27日（四）13:03 憨包（讨论 | 贡献） . . （91,042字节） (+18) . . *(→演出)* （撤销）
- （当前 | 先前） 2012年9月19日（三）05:43 183.213.57.218（讨论） . . （91,024字节） (0) . . *(→评价)* （撤销）
- （当前 | 先前） 2012年8月17日（五）20:56 ZéroBot（讨论 | 贡献） 小 . . （91,024字节） (+31) . . *(r2.7.1)(机器人添加：nds-nl:William Shakespeare)* （撤销）
- （当前 | 先前） 2012年8月15日（三）14:13 YFdyh-bot（讨论 | 贡献） 小 . . （90,993字节） (-1) . . *(机器人：mk:Вилијам Шекспир是一篇特色条目)* （撤销）
- （当前 | 先前） 2012年8月8日（三）09:49 Makecat-bot（讨论 | 贡献） 小 . . （90,994字节） (+115) . . *(CHECKWIKI)* （撤销）
- （当前 | 先前） 2012年8月3日（五）18:22 MerlIwBot（讨论 | 贡献） 小 . . （90,879字节） (+36) . . *(机器人添加：lez:Вилиам Шекспир)* （撤销）
- （当前 | 先前） 2012年8月1日（三）15:36 180.176.161.178（讨论） . . （90,843字节） (-28) . . *(→原作者)* （撤销）
- （当前 | 先前） 2012年7月22日（日）10:45 Mewaqua（讨论 | 贡献） . . （90,871字节） (-122) . . *(取消163.16.242.18的修訂22053240)* （撤销）
- （当前 | 先前） 2012年7月20日（五）02:20 李小绳（讨论 | 贡献） 小 . . （90,993字节） (+4) . . （撤销）
- （当前 | 先前） 2012年7月20日（五）02:17 163.16.242.18（讨论） . . （90,989字节） (+82) . . （撤销）

# 维基百科可视化

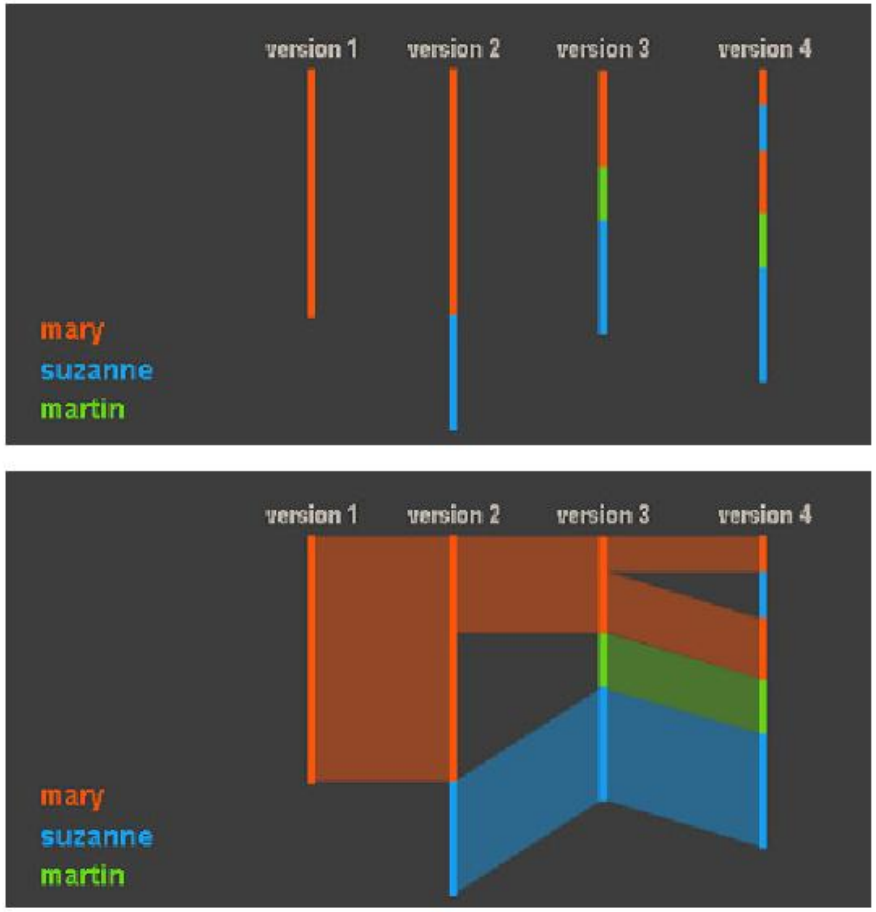



Figure 11-2. *Schematic of history flow's visualization mechanism*

2012.10.18

# 维基百科可视化

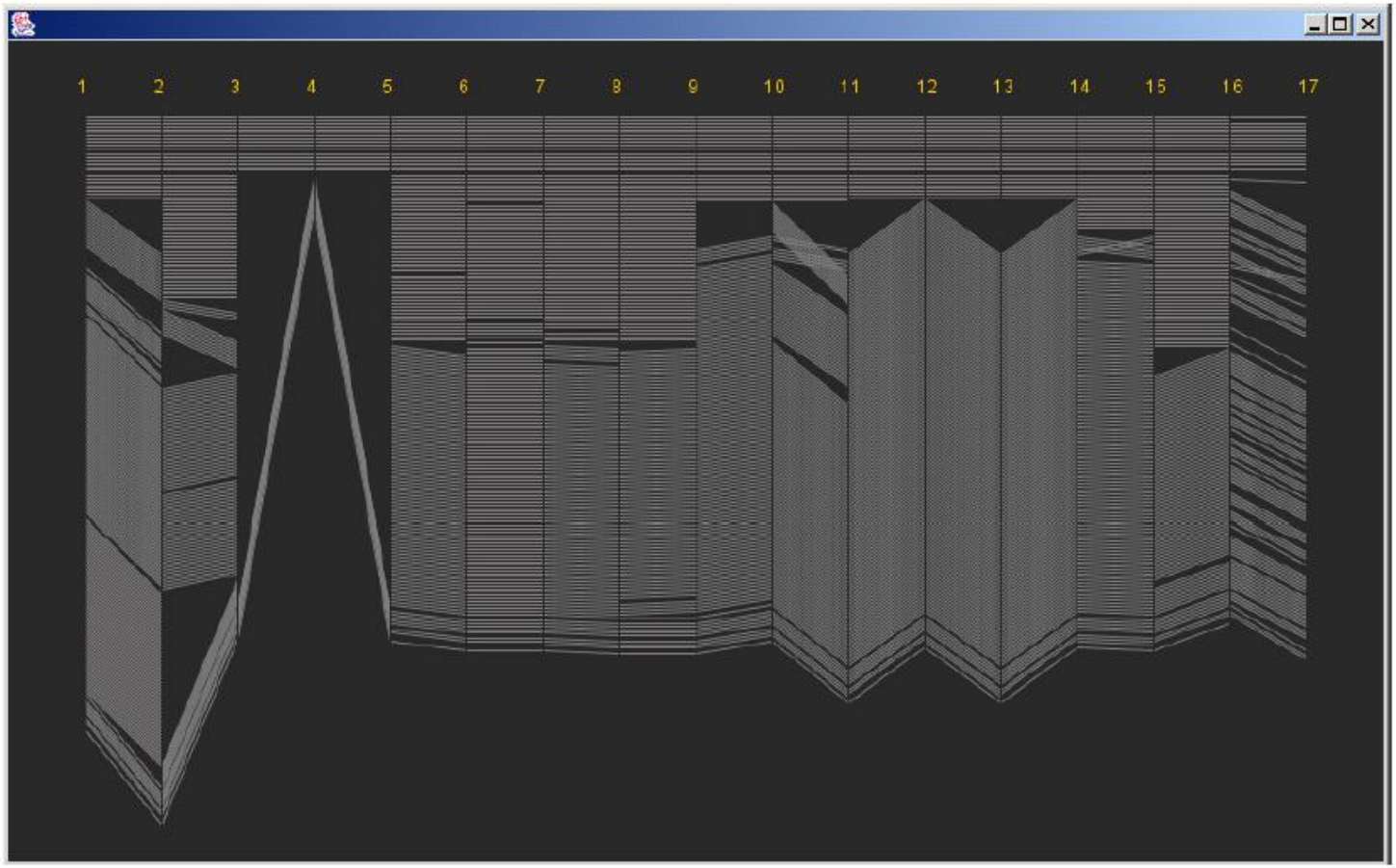



Figure 11-3. *An early version of history flow, with simple lines connecting pieces of text that survive intact over consecutive versions*

2012.10.18

# 维基百科可视化

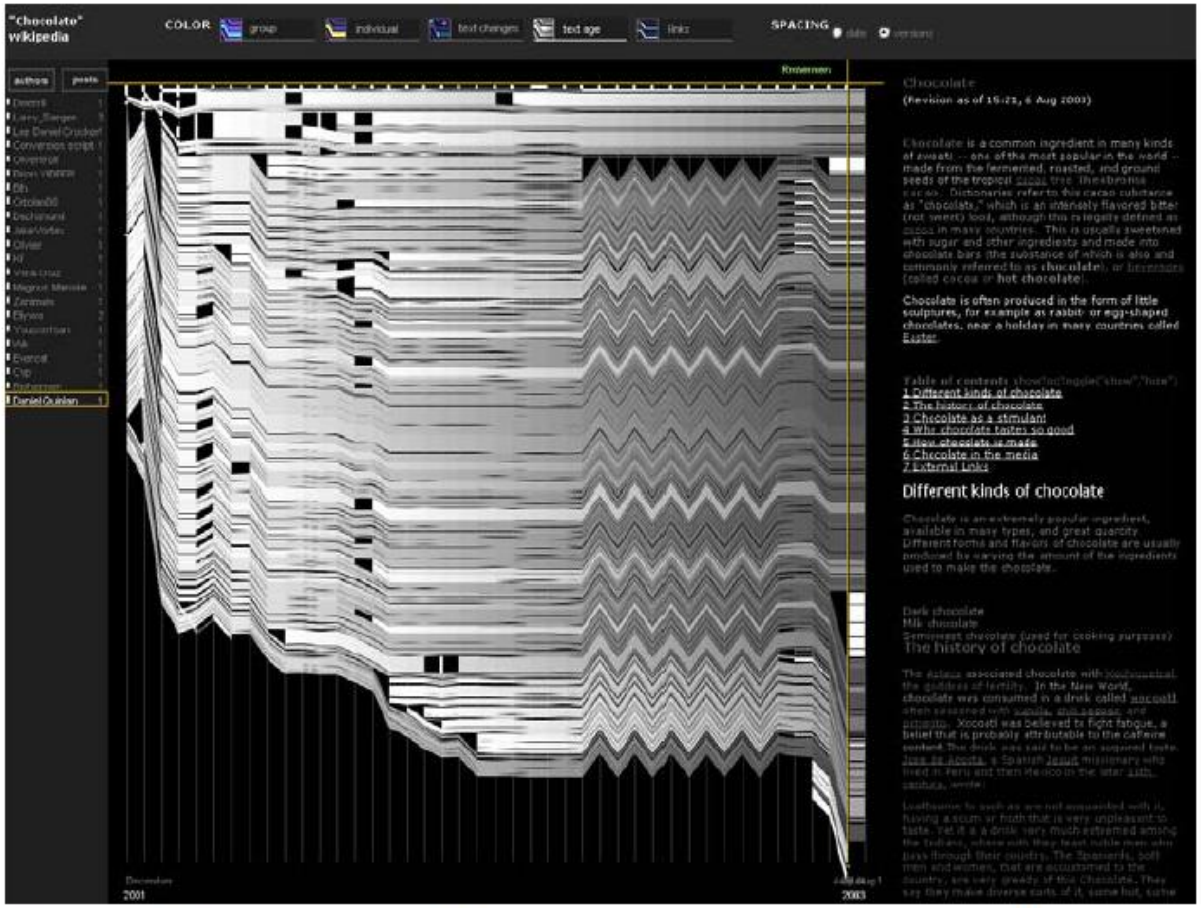



Figure 11-4. *History flow diagram showing text age on the "Chocolate" article in Wikipedia: darker patches represent older passages*

2012.10.18

# 维基百科可视化

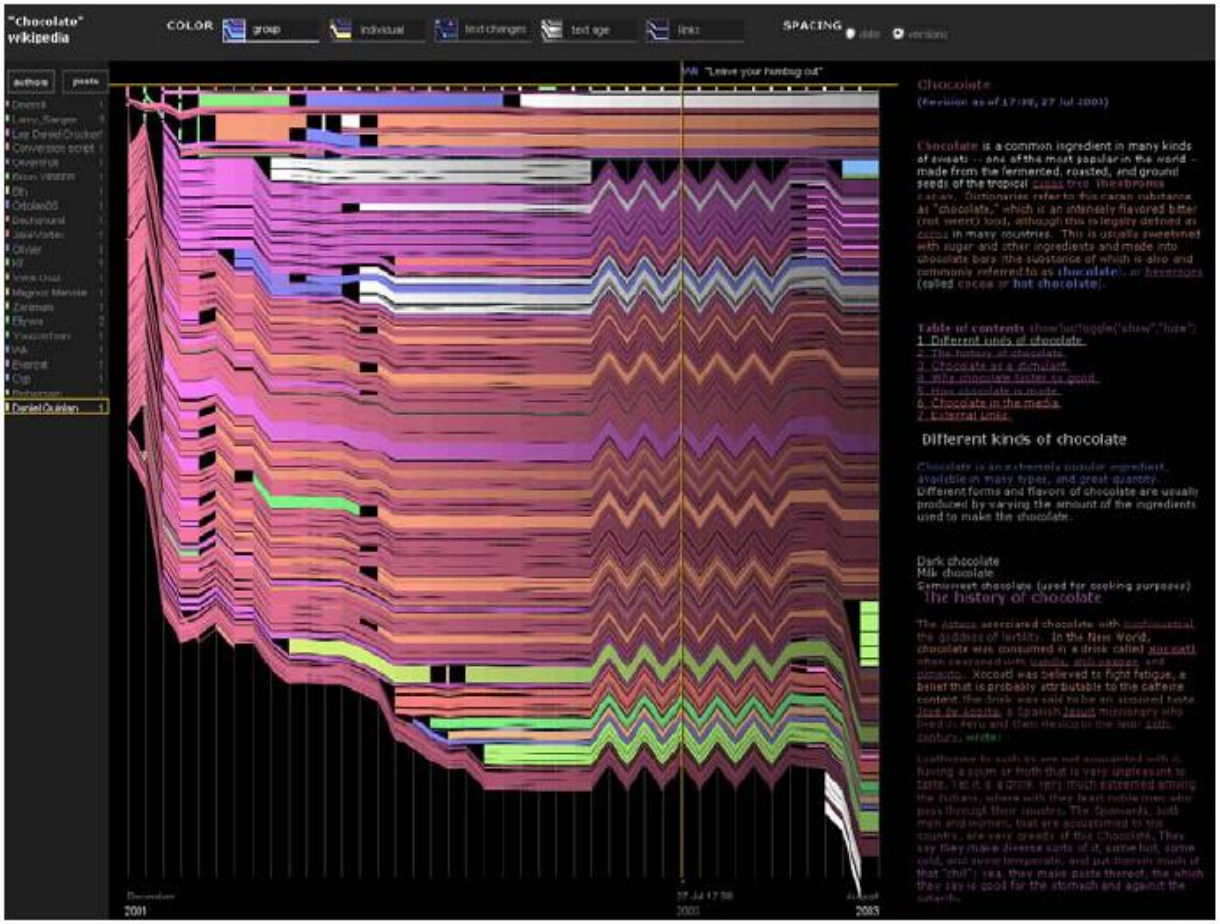



Figure 11-5. *History flow in color: each color represents text from a given author*

2012.10.18

# 维基百科可视化

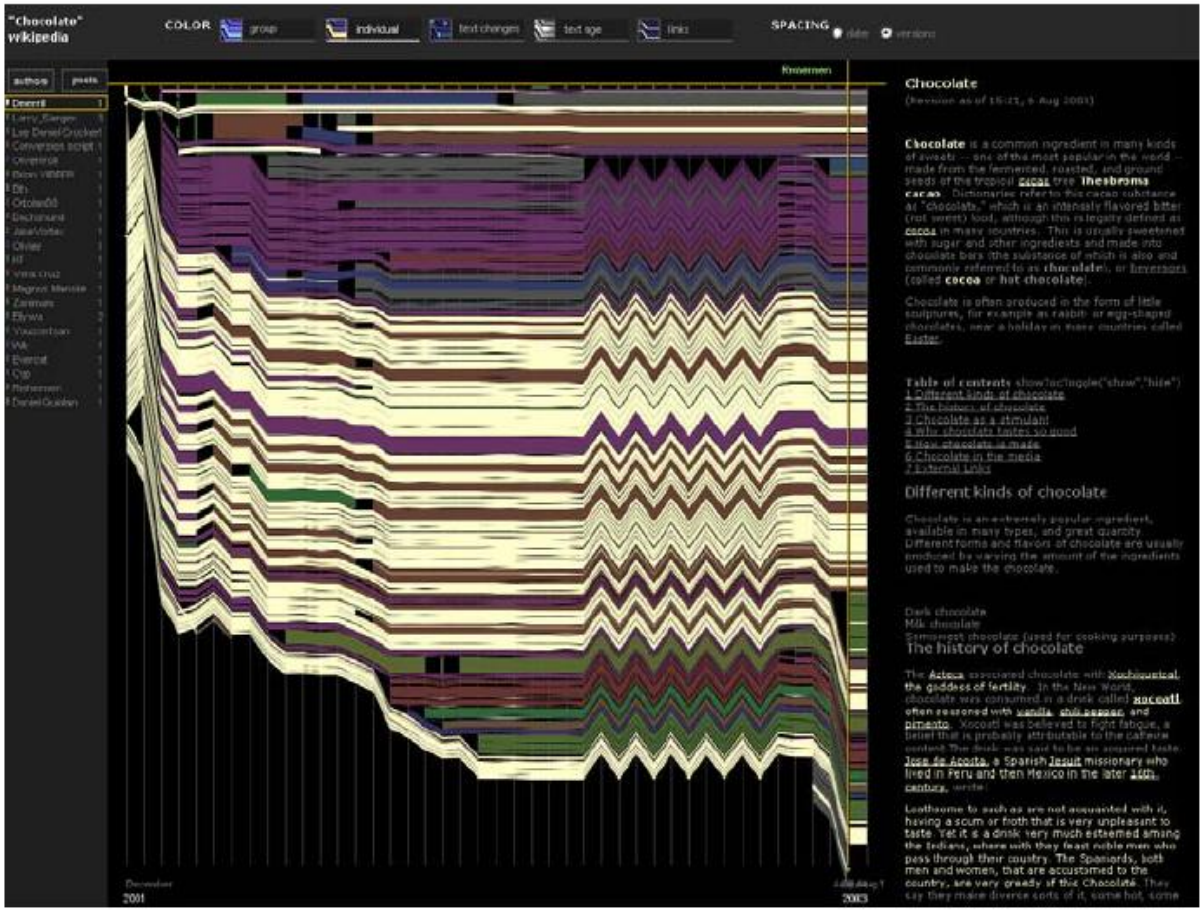



Figure 11-6. *Diagram showing, in cream, the contributions from a single author over time*

2012.10.18

# 维基百科可视化

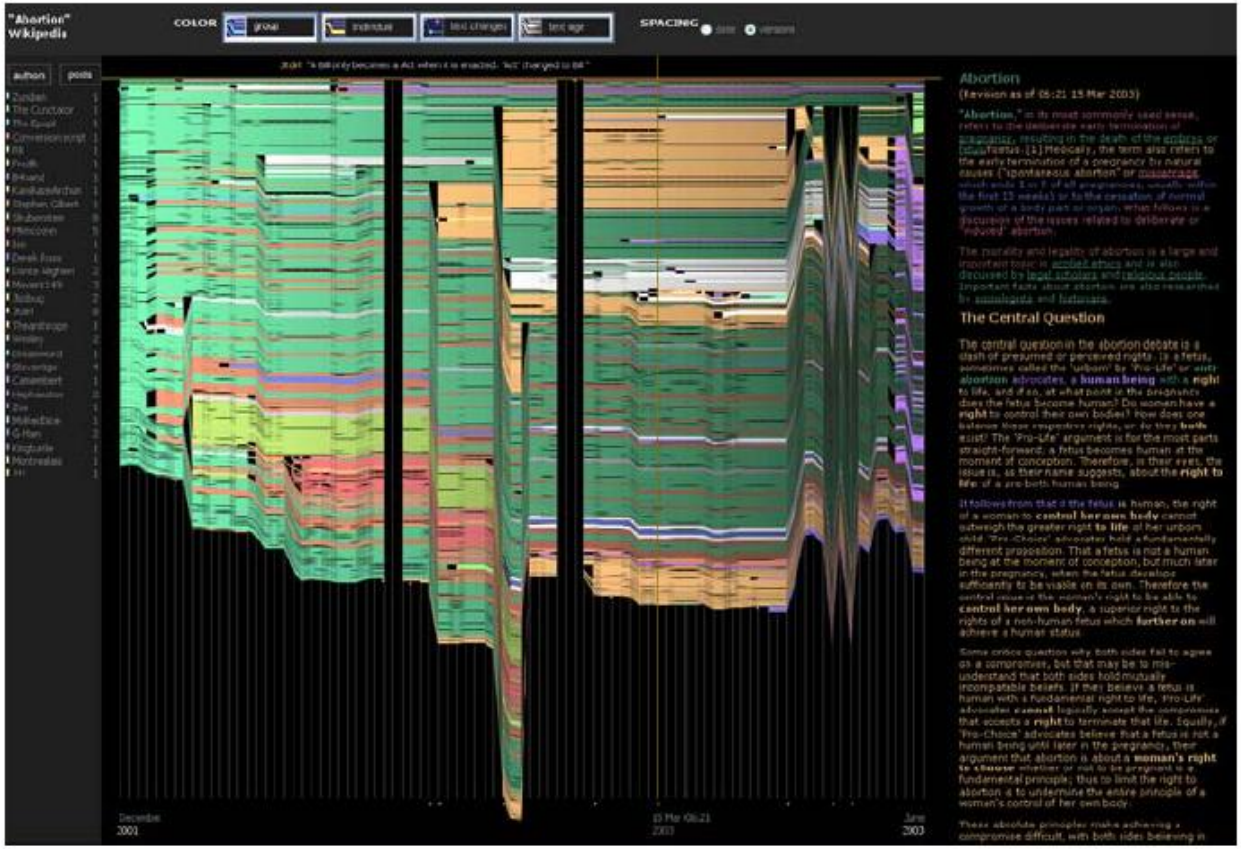



Figure 11-7. *Editing history for "Abortion" showing versions equally spaced—black gutters represent "mass deletions," an act of vandalism whereby a user deletes all content in a given article*

# 维基百科可视化

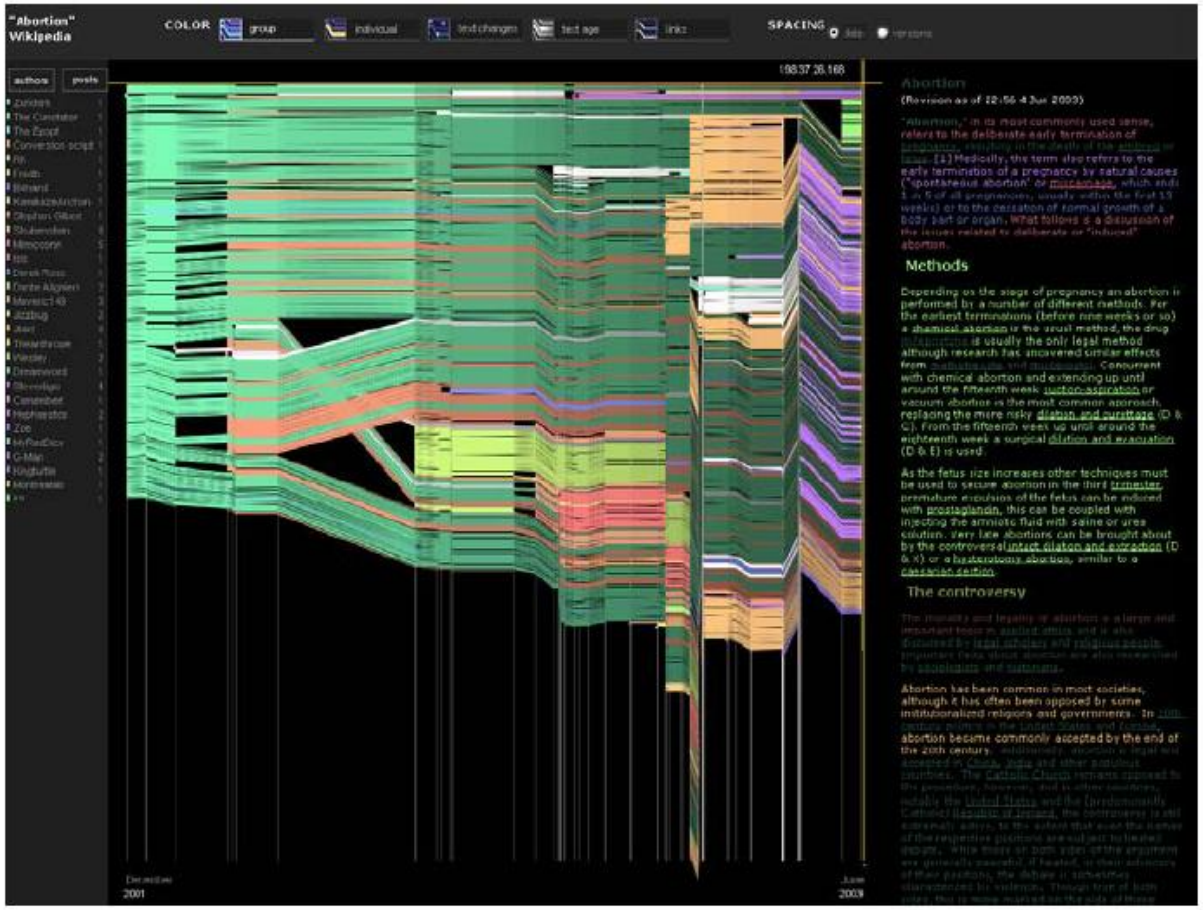



Figure 11-8. *Editing history for "Abortion" showing versions spaced by time*

2012.10.18

# 炼数成金逆向收费式网络课程

- Dataguru（炼数成金）是专业数据分析网站，提供教育，媒体，内容，社区，出版，数据分析业务等服务。我们的课程采用新兴的互联网教育形式，独创地发展了逆向收费式网络培训课程模式。既继承传统教育重学习氛围，重竞争压力的特点，同时又发挥互联网的威力打破时空限制，把天南地北志同道合的朋友组织在一起交流学习，使到原先孤立的学习个体组合成有组织的探索力量。并且把原先动辄成千上万的学习成本，直线下降至百元范围，造福大众。我们的目标是：低成本传播高价值知识，构架中国第一的网上知识流转阵地。
- 关于逆向收费式网络的详情，请看我们的培训网站 http://edu.dataguru.cn

# Thanks

## FAQ时间