

R应用入门

田锴 南京大学

2019.3.1

Part 1 —— R 简介

Part 2 —— R语言的命令行式数据操作

Part 3 —— 用R实现基础统计

Part 4 —— 图形基础与绘制

Part 1

R 简介

主要作用

- 1) 数据整理与分析
- 2) 数学统计与建模
- 3) 绘图

优点

1. 用户制定性强

- 控制所有的参数
- 定义自己的函数

2. 分析、作图的可重复性

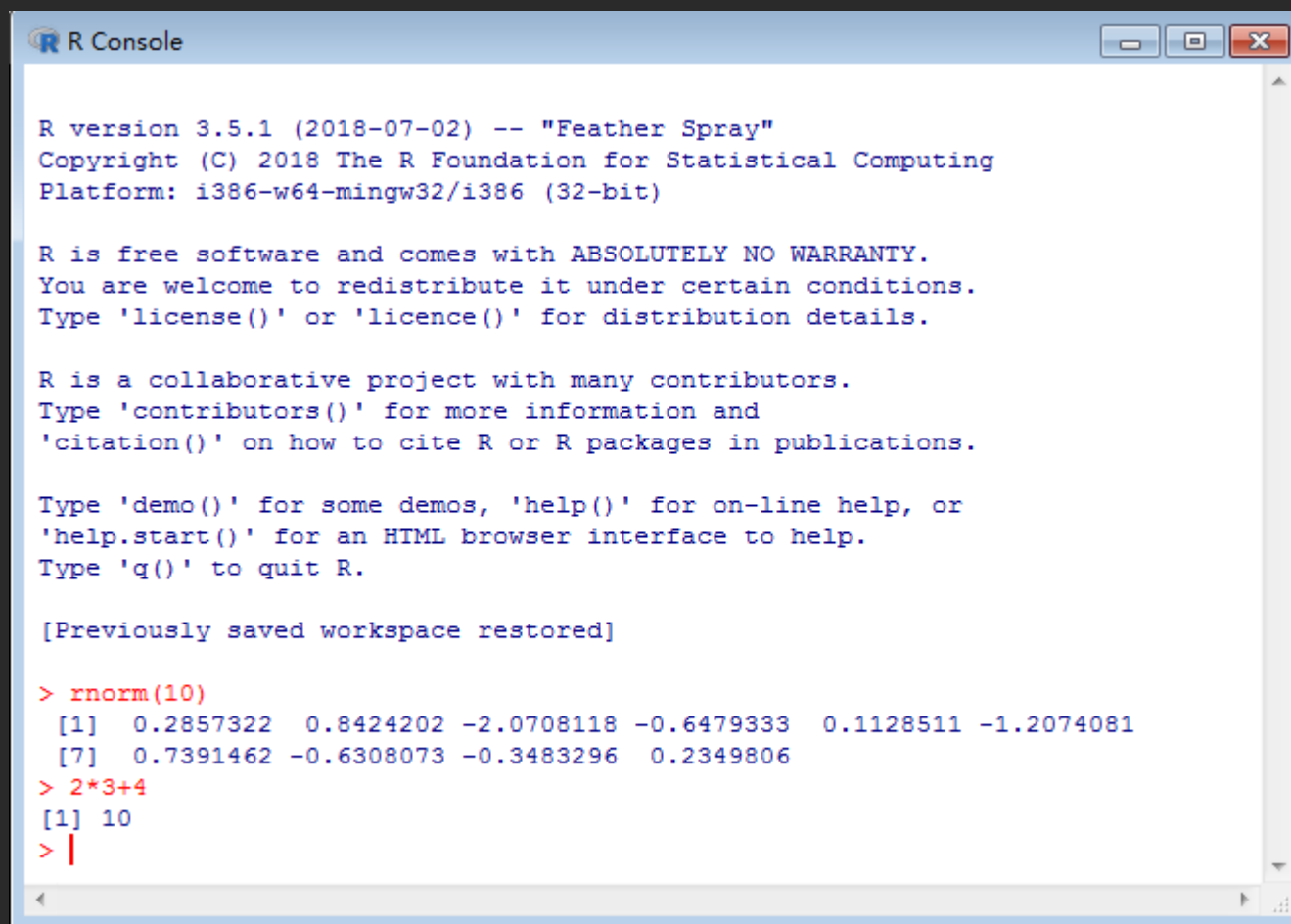
3. 新研究方法的即时性传播

- **R**是开源的，可实现代码共享

4. 省时、高效地数据管理及质量检验

5. 与各种软件的互动与兼容

- 控制台



```
R R Console

R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> rnorm(10)
[1] 0.2857322 0.8424202 -2.0708118 -0.6479333 0.1128511 -1.2074081
[7] 0.7391462 -0.6308073 -0.3483296 0.2349806
> 2*3+4
[1] 10
> |
```

- <https://cran.r-project.org/>



CRAN

[Mirrors](#)

[What's new?](#)

[Task Views](#)

[Search](#)

About R

[R Homepage](#)

[The R Journal](#)

Software

[R Sources](#)

[R Binaries](#)

[Packages](#)

[Other](#)

Documentation

[Manuals](#)

[FAQs](#)

[Contributed](#)

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2018-12-20, Eggshell Igloo) [R-3.5.2.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Questions About R

- 下载：选择镜像

China

<https://mirrors.tuna.tsinghua.edu.cn/CRAN/>

<http://mirrors.tuna.tsinghua.edu.cn/CRAN/>

<https://mirrors.ustc.edu.cn/CRAN/>

<http://mirrors.ustc.edu.cn/CRAN/>

<https://mirror-hk.koddos.net/CRAN/>

TUNA Team, Tsinghua University

TUNA Team, Tsinghua University

University of Science and Technology of China

University of Science and Technology of China

KoDDoS in Hong Kong

- 安装：最新版Base

R-3.5.2 for Windows (32/64 bit)

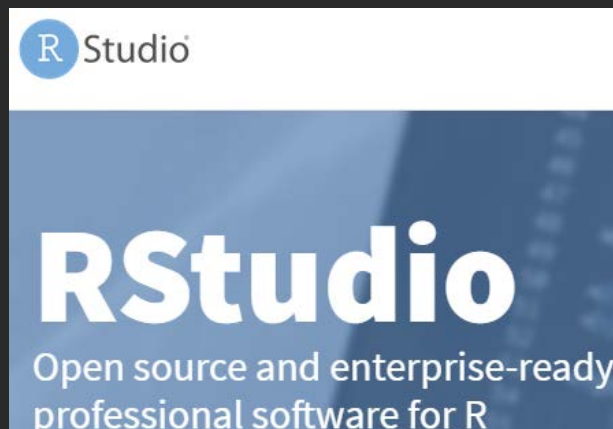
[Download R 3.5.2 for Windows](#) (79 megabytes, 32/64 bit)

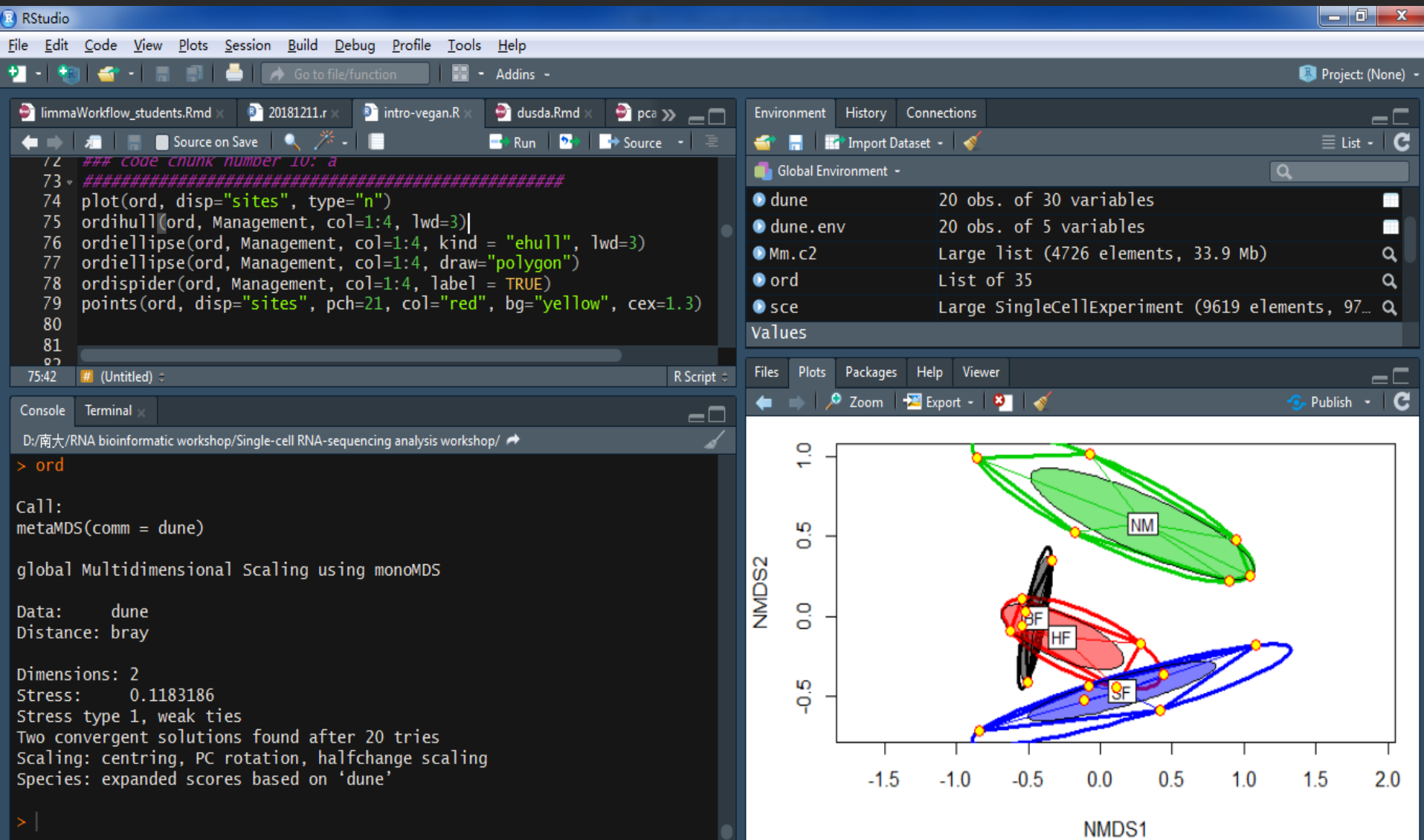
[Installation and other instructions](#)

[New features in this version](#)

- 编辑器等：Rstudio

<https://www.rstudio.com/>





```
Untitled - R Editor
library(vegan)
plo|
```

Console Terminal x

~/ ➔

Run 12 stress 0.1183186
... Procrustes: rmse 4.542622e-05 max resid 0.0001313951
... Similar to previous best
Run 13 stress 0.1808912
Run 14 stress 0.2035424
Run 15 stress 0.1192679
Run 16 stress 0.1183186
Procrustes: rmse 8.19422e-06 max resid 1.756244e-05
:
◆ plogis {stats}
:
R ◆ plot {graphics}
R ◆ plot.default {graphics}
R ◆ plot.design {graphics}
:
◆ plot.ecdf {stats}
:
R ◆ plot.function {graphics}
* ◆ plot.new {graphics}
> ◆ plot.spec.coherency {stats}
> plo|

Install ↻

	Name
<input type="checkbox"/>	utf8
<input type="checkbox"/>	utils
<input type="checkbox"/>	vcd
<input checked="" type="checkbox"/>	vegan
<input type="checkbox"/>	VGAM
<input type="checkbox"/>	vipor
<hr/>	
<input type="checkbox"/>	XML
<input type="checkbox"/>	xtable
<input type="checkbox"/>	XVector
<input type="checkbox"/>	yaml

plogis(q, location = 0, scale = 1, lower.tail = TRUE, log.p = FALSE)

Density, distribution function, quantile function and random generation for the logistic distribution with parameters `location` and `scale`.

Press F1 for additional help

Files/R/R-3.5.1/library")

- 基础软件包

stats	# 统计包
graphics	# 作图包
grDevices	# 图形设备包
utils	# 工具包
datasets	# 数据仓库
methods	# 方法包
base	# 基础包

```
> ls(name = "package:stats")
[1] "acf"           "acf2AR"         "add.scope"
[4] "add1"          "addmargins"     "aggregate"
[7] "aggregate.data.frame" "aggregate.ts"   "AIC"
[10] "alias"         "anova"          "ansari.test"
[13] "aov"           "approx"         "approxfun"
[16] "ar"            "ar.burg"        "ar.mle"
```

```
> length(ls(name = "package:stats"))
[1] 447
```

Contributed Packages

Available Packages

Currently, the CRAN package repository features 13714 available packages.

[Table of available packages, sorted by date of publication](#)

[Table of available packages, sorted by name](#)

各软件包及其函数是利用R快速解决问题的关键。

- 安装包

install.packages("vegan") # 安装vegan包

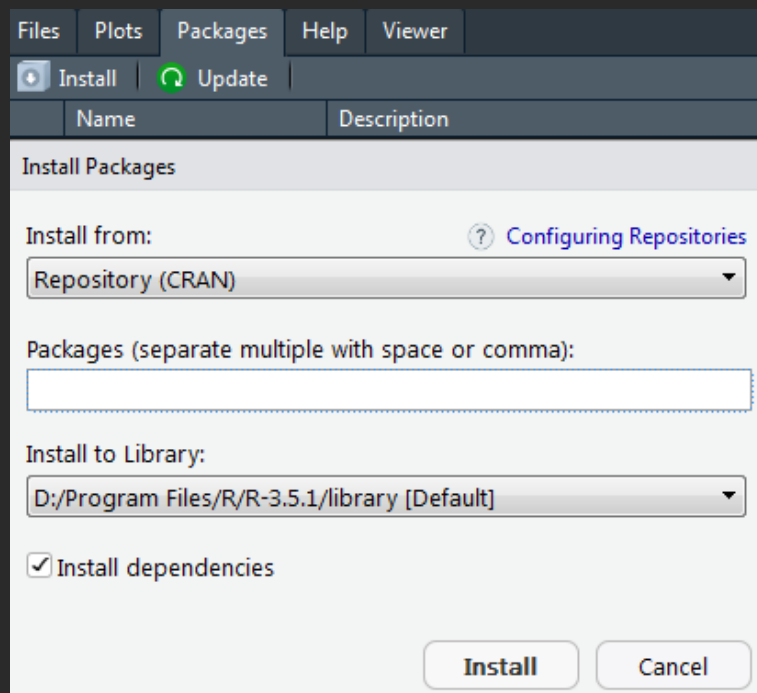
available.packages()

update.packages()

installed.packages()

remove.packages()

(提前选择下载的镜像地址)



- 调用包至当前工作环境

library("vegan") # 加载vegan包

工作空间管理

- 查看工作路径: `getwd()`
- 更改工作路径: `setwd("D:/")` # 非常重要
- 列出工作空间中的所有对象: `ls()`
- 保存工作空间中的所有对象: `save.image(file="xxx.Rdata")`
- 保存部分制定对象: `save(data1,data2,file="d1d2.Rdata")`
- 加载以保存的工作空间: `load(file)`
- 工作空间内的历史代码查看: `history(max.show=50)`
- 退出工作空间: `q()`

数据读写

读取表格形式的数据: `read.table()`、`read.csv()`

- `read.table()`

可读入txt、csv文件，读取的结果是一个数据框(data.frame)

```
read.table(file, header = FALSE, sep = "", quote = "\"'",  
  dec = ".", numerals = c("allow.loss", "warn.loss", "no.loss"),  
  row.names, col.names, as.is = !stringsAsFactors,  
  na.strings = "NA", colClasses = NA, nrows = -1,  
  skip = 0, check.names = TRUE, fill = !blank.lines.skip,  
  strip.white = FALSE, blank.lines.skip = TRUE,  
  comment.char = "#",  
  allowEscapes = FALSE, flush = FALSE,  
  stringsAsFactors = default.stringsAsFactors(),  
  fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)
```

设置`header = TRUE`，可以将第一行数据作为行名。

	col2	col2	4	2	c
row1	1	a	5	1	a
row2	3	b	6	3	b

- `read.csv()`

读取csv文件，读入后也是一个数据框(data.frame)

```
read.csv(file, header = TRUE, sep = ",", quote = "\"",  
         dec = ".", fill = TRUE, comment.char = "", ...)
```

- 写出数据框至文件

`write.table(data, "xxx.txt")`

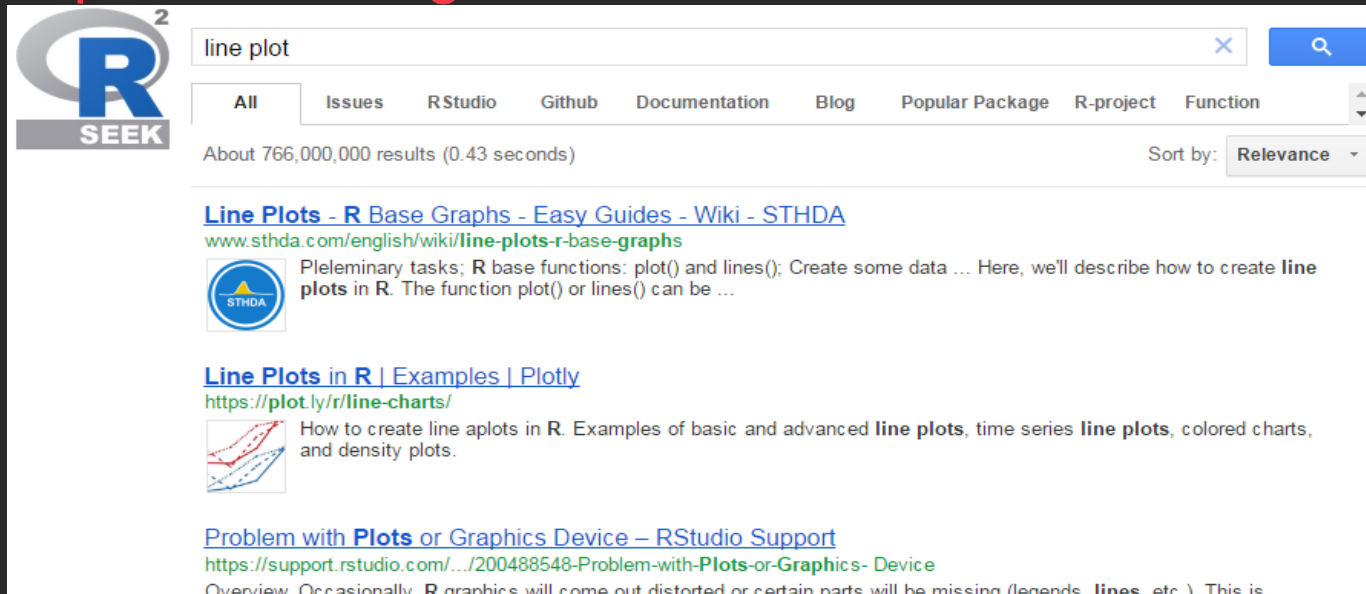
`write.csv(data, "xxx.csv")`

```
write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ",  
           eol = "\n", na = "NA", dec = ".", row.names = TRUE,  
           col.names = TRUE, qmethod = c("escape", "double"),  
           fileEncoding = "")
```

寻求帮助

- ?
- ??
- 输入某一个函数的名字
- `help (package="...")`
- `data (package="...")`
- <https://rseek.org/>

```
> getS3method("mean","default")
function (x, trim = 0, na.rm = FALSE, ...)
{
  if (!is.numeric(x) && !is.complex(x) && !is.logical(x)) {
    warning("argument is not numeric or logical: returning NA")
    return(NA_real_)
  }
  if (na.rm)
    x <- x[!is.na(x)]
  if (!is.numeric(trim) || length(trim) != 1L)
    stop("'trim' must be numeric of length one")
  n <- length(x)
  if (trim > 0 && n) {
    if (is.complex(x))
      stop("trimmed means are not defined for complex data")
    if (anyNA(x))
      return(NA_real_)
    if (trim >= 0.5)
      return(stats::median(x, na.rm = FALSE))
    lo <- floor(n * trim) + 1
    hi <- n + 1 - lo
    x <- sort.int(x, partial = unique(c(lo, hi)))[lo:hi]
  }
  .Internal(mean(x))
}
<bytecode: 0x02e5a7c8>
<environment: namespace:base>
```



The screenshot shows the Rseek search engine interface. The search bar contains the text "line plot". Below the search bar, there are tabs for "All", "Issues", "RStudio", "Github", "Documentation", "Blog", "Popular Package", "R-project", and "Function". The "All" tab is selected. The search results show "About 766,000,000 results (0.43 seconds)". The results are sorted by "Relevance". The first result is "Line Plots - R Base Graphs - Easy Guides - Wiki - STHDA" with the URL "www.sthda.com/english/wiki/line-plots-r-base-graphs". The second result is "Line Plots in R | Examples | Plotly" with the URL "https://plot.ly/r/line-charts/". The third result is "Problem with Plots or Graphics Device - RStudio Support" with the URL "https://support.rstudio.com/.../200488548-Problem-with-Plots-or-Graphics-Device".

R FAQ

Frequently Asked Questions on R

Version 2017-10-04

Kurt Hornik

Table of Contents

[1 Introduction](#)

- [1.1 Legalese](#)
- [1.2 Obtaining this document](#)
- [1.3 Citing this document](#)
- [1.4 Notation](#)
- [1.5 Feedback](#)

[2 R Basics](#)

- [2.1 What is R?](#)
- [2.2 What machines does R run on?](#)
- [2.3 What is the current version of R?](#)
- [2.4 How can R be obtained?](#)
- [2.5 How can R be installed?](#)
 - [2.5.1 How can R be installed \(Unix-like\)](#)
 - [2.5.2 How can R be installed \(Windows\)](#)
 - [2.5.3 How can R be installed \(Mac\)](#)
- [2.6 Are there Unix-like binaries for R?](#)
- [2.7 What documentation exists for R?](#)
- [2.8 Citing R](#)

R-intro.pdf - Adobe Acrobat Pro

文件(F) 编辑(E) 视图(V) 窗口(W) 帮助(H)

创建 保存 打印 复制 粘贴 撤销 重做 删除 查找 替换 注释 批注 签名 盖章 模板 颜色 字体 语言 格式 工具

1 / 105 117%

书签

- Preface
- Introduction and preliminaries
- Simple manipulations; numbers and vectors
- Objects, their modes and attributes
- Ordered and unordered factors
- Arrays and matrices
- Lists and data frames
- Reading data from files
- Probability distributions
- Grouping, loops and

An Introduction to R

Notes on R: A Programming Environment for Data Analysis and Graphics
Version 3.5.1 (2018-07-02)

help (package="vegan")

Community Ecology Package



Documentation for package 'vegan' version 2.5-3

- [DESCRIPTION file.](#)
- [User guides, package vignettes and other documentation.](#)
- [Package NEWS.](#)

Help Pages

[A](#)[B](#)[C](#)[D](#)[E](#)[F](#)[G](#)[H](#)[I](#)[K](#)[L](#)[M](#)[N](#)[O](#)[P](#)[Q](#)[R](#)[S](#)[T](#)[U](#)[V](#)[W](#)

[vegan-package](#)

Community Ecology Package: Ordination, Diversity and Dissimilarities

-- A --

[add1.cca](#)

Add or Drop Single Terms to a Constrained Ordination Model

[adipart](#)

Additive Diversity Partitioning and Hierarchical Null Model Testing

[adipart.default](#)

Additive Diversity Partitioning and Hierarchical Null Model Testing

add1.cca {vegan}

Add or Drop Single Terms to a Constrained Ordination Model

Description

Compute all single terms that can be added to or dropped from a constrained ordination model.

Usage

```
## S3 method for class 'cca'
add1(object, scope, test = c("none", "permutation"),
      permutations = how(nperm=199), ...)
## S3 method for class 'cca'
drop1(object, scope, test = c("none", "permutation"),
       permutations = how(nperm=199), ...)
```

Arguments

Part 2

R语言的命令行式数据操作

- 不清楚R的逻辑规则和命令行式操作，使用R就会很艰难。
- 开始使用R时，经常遇到问题、出错的地方，大都是这部分内容。

R 参考卡片



英文文档最初由 Tom Short tshort@eprisolutions.com 撰写, 在 www.Rpad.org 上可以得到最新文档.

中文版本文档(已获得翻译发布许可) 结构上同原版类似, 局部添加了若干命令.

后续修订以及维护由 刘思喆 负责, 如有批评或建议

请联系: sunbjt@hotmail.com, bjt@ruc.edu.cn

中文版本: 1.1 2007-1-23

帮助和基础

大部分R 函数都有在线文档.

help(topic) 关于topic的文档.

?topic 同上

help.search("topic") 搜索帮助系统

apropos("topic") 返回所有在搜索路径下满足正则表达式"topic"的所有对象名称

help.start() HTML形式的帮助

demo R 功能演示

example(f) 运行在线帮助中的例子

str(a) 显示R 对象的内在属性(*structure)或简要说明对象

summary(a) 给出a的概要, 通常是一个一般性统计概要; 且它对不同属性的a 有不同的操作方式.

ls() 显示搜索路径下的对象; 指定pat="pat"时, 按式样条件搜索

ls.str() str() 搜索路径下的每个变量

read.csv("filename", header=TRUE) 同上, 但默认设置为读取逗号分割文件

read.delim("filename", header=TRUE) 同上, 默认设置为读取tab 分割文件

read.fwf(file, widths, header=F, sep="\t", as.is=FALSE) 以fixed width formatted形式读取数据至数据框; widths 是整数向量, 用于设置调整宽度字段

save(file, ...) 以不分平台的二进制保存指定的对象

save.image(file) 保存所有的对象

dump("x", "...") 将对象x 保存在"..."里

cat(..., file="", sep=" ") 强制转化为字符后打印arguments; sep 为arguments间的分割字符

print(a, ...) 显示arguments; 更一般的, 它对于不同的对象可以有不同的表达方式.

format(x, ...) 格式化, 更好的显示R 对象

write.table(x, file="", row.names= T, col.names= T, sep="") 在把x转化为数据框后, 写到文件; 如果quote 为TRUE, 字符和因子列就会被(")所包围; sep 是字段分隔符; eol 为尾行分割符; na 为缺失值字符串; 使用col.names=NA 增加列标题以便于和表格输入一致

sink(file) 输出到文件file, 直到输入命令sink()

大部分 I/O 函数都有file 参量. 它经常用一个字符串来命名文件或连接. file="" 意味着标准输入或输出. 连接(Connections)可以包涵文件(file), 管道(pipes), 压缩文件(zippped files)或 R 变量.

在 windows 操作环境下, 文件共享使用可以通过写字板(clipboard)的方式. 读取 Excel 表, 可以将 Excel 中数据拷贝至写字板, 使用

x <- read.delim("clipboard") 方式读取数据. 如果要将数据写入到写

数据分割和选取

向量索引

x[n]

x[-n]

x[1:n]

x[-(1:n)]

x[c(1, 4, 2)]

x["name"]

x[x > 3]

x[x > 3 & x < 5]

x[x %in% c("a", "and", "the")]

列表索引

x[n]

x[[n]]

x[["name"]]

x\$name

矩阵索引

x[i, j]

x[i,]

x[, j]

x[, c(1, 3)]

x["name",]

x["name",] 名为"name"的行

x[["name"]] 列名为"name"的列

x\$name

变量变换

as.array(x), as.data.frame(x), as.numer:

第n个元素

除了第n个元素的x

前n个元素

第n+1 至最后的元素

指定元素

名为"name"的元素

所有大于3的元素

区间(3,5)的元素

给定组中的元素

列表显示元素n

列表的第n个元素

名为"name"的元素

同上.

下标为(i,j)的元素

第1行

第j列

第1和3列

名为"name"的行

数据框索引(矩阵索引加下述)

列名为"name"的列

同上.

R对大小写字母敏感!

◆ 对象(object)

创建对象

- 赋值符

“=” 或 “<-” (“>-”)

```
> x = c(1:5)
> y <- c(2:6)
> c(3:7) -> z
> x
[1] 1 2 3 4 5
> y
[1] 2 3 4 5 6
> z
[1] 3 4 5 6 7
> |
```

命名的注意事项:

1. 首字母不能为数字
2. 不能用for, if, break, ifelse, function等作为对象名；最好不用R里的函数名作为对象名

删除对象

- `rm()`

`rm (ls = list ())` # 删除所有对象

`rm (c(x, y, z))` # 删除之前创建的x, y, z

- 一些常用的R默认对象:

逻辑型

```
> TRUE
[1] TRUE
> T
[1] TRUE
> FALSE
[1] FALSE
> F
[1] FALSE
> |
```

数值型

```
> pi
[1] 3.141593
```

空值和缺失值

```
> NULL
NULL
> NA
[1] NA
```

字符型

```
> LETTERS
[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"
> letters
[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"
```

日期型

```
> month.name
[1] "January" "February" "March" "April" "May" "June"
[7] "July" "August" "September" "October" "November" "December"
> month.abb
[1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
```

- 针对一些特殊值，如NA，有些函数提供了如na.rm=T这样的参数，当值为TRUE 会删除带有NA 值的记录，或者提供na.action这样的参数，指定用于处理NA值的函数。

```
is.na # 是否NA? NA, NaN 都会被认为是NA  
complet.cases # 是否不包含NA 值?  
is.nan # 是否NaN?  
is.null # 是否NULL?  
na.fail # 当存在NA 值时，生成一个错误(error)。  
na.omit # 删除带有NA 值的记录。
```

- options()中存储着一些全局设置，其中na.action=na.omit，即默认对包含NA 的记录会做删除处理。也可改变这个设置。

◆ 基本数据类型

str()可查看数据类型

➤ 向量

函数: **c()**

连接多个数值(整数、实数、逻辑值、字符)或数值向量, 生成一个向量。

举例:

```
> int<-c(1:3)
> stg<-c("a","b","c")
> c(int,stg)
[1] "1" "2" "3" "a" "b" "c"
> |
```

```
> str(int)
int [1:3] 1 2 3
> str(stg)
chr [1:3] "a" "b" "c"
> str(c(int,stg))
chr [1:6] "1" "2" "3" "a" "b" "c"
> |
```


- 向量化计算的实现是r语言非常大的优势，省去了很多写循环的麻烦，可以很方便地处理一批数据。

如果两个向量进行算术运算或二元逻辑运算，那么它们对应位置的元素会分别进行计算。

如果它们长度不同，较短的向量会被循环使用至另一个向量的长度。如果较长向量的长度不是较短向量的整数倍，则会产生一个警告。

```
> paste(stg, int, sep="_")  
[1] "a_1" "b_2" "c_3"
```

```
> 1:10+100  
[1] 101 102 103 104 105 106 107 108 109 110  
> 1:10>5  
[1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
> v1<-1:10+c(100,200)  
> v1  
[1] 101 202 103 204 105 206 107 208 109 210
```

向量按类型又可分为数值、字符串、逻辑型、因子型向量等

数据类型转换

- `as.factor()`

```
> apropos("^as\\.")
[1] "as.array"           "as.array.default"
[3] "as.call"            "as.character"
[5] "as.character.condition" "as.character.Date"
[7] "as.character.default" "as.character.error"
[9] "as.character.factor"  "as.character.hexmode"
[11] "as.character.numeric_version" "as.character.octmode"
[13] "as.character.POSIXt"  "as.character.srcref"
[15] "as.complex"           "as.data.frame"
[17] "as.data.frame.array"  "as.data.frame.AsIs"
```

```
> group<-rep(c("CK","High"),5)
> class(group)
[1] "character"
> gp<-as.factor(group)
> class(gp)
[1] "factor"
> group
[1] "CK"    "High" "CK"    "High" "CK"    "High" "CK"    "High" "CK"    "High"
> gp
[1] CK    High CK    High CK    High CK    High CK    High
Levels: CK High
```

数值型→字符型、因子型、逻辑型

字符型→数值型、因子型

因子型变量

- 分类变量通常需要以因子(factor)的形式存储，以便可以正确地使用在模型中。因子对象的因子水平以字符形式存储，而数据本身则用整数形式存储，指向对应的因子水平。

```
> gp<-factor(1:10,labels=rep(c("CK","High"),5))
> gp
[1] CK    High CK    High CK    High CK    High CK    High
Levels: CK High
```

```
levels() # 可以查看及设置因子水平
nlevels() # 返回因子水平数
reorder() # 根据一个连续变量在该因子上的分类汇总结果排序
relevel() # 设置其中一个水平为参考水平(一般是第一个水平)
```

由数字组成的因子对象，使用其字面的数字进行计算时，直接使用as.numeric 是不合适的。

```
> fac1=factor(seq(10,50,by=10))
> fac1
[1] 10 20 30 40 50
Levels: 10 20 30 40 50
> as.numeric(fac1)
[1] 1 2 3 4 5
> as.numeric(levels(fac1))[fac1]
[1] 10 20 30 40 50
```

➤ 数组(array)和矩阵(matrix)

数组是向量的多维形式，矩阵是维数为2的数组。通过给向量赋予dim属性，就可将向量转换成数组或矩阵。

• array()

```
> myarr<-array(1:16,  
+             dim=c(4,2,2),  
+             dimnames=  
+               list(c("a","b","c","d"),  
+                   c("A","B"),  
+                   c("first","second"))  
+             )  
> |
```

```
> myarr  
, , first  
  
  A B  
a 1 5  
b 2 6  
c 3 7  
d 4 8  
  
, , second  
  
  A B  
a 9 13  
b 10 14  
c 11 15  
d 12 16
```

• matrix()

```
> mat<-matrix(1:6,  
+            nrow=2,  
+            byrow=T,  
+            dimnames=list(c("a","b"),LETTERS[1:3]))  
+
```

```
> mat  
  A B C  
a 1 2 3  
b 4 5 6
```

➤列表 (list)

最为灵活的数据结构。**list ()**可以将若干个不同结构(class), 不同类型(mode)的数据连接成一个整体。由于list最佳的扩展性。

很多函数值是以列表形式输出, 把要返回的对象组合成一个list。

```
> lst1<-list(first=int, second=stg, third=mat)
> lst1
$`first`
[1] 1 2 3

$second
[1] "a" "b" "c"

$third
  A B C
a 1 2 3
b 4 5 6
```

➤数据框(data.frame)

数据表型式。最为常用的数据类型，本质上是一种由等长向量组成的列表。数据框有行名和列名等属性。

```
daf1 = data.frame(  
  name = c("wang feng", "zhang yan", "liu jun",  
           "li zhenying", "zhang yu", "chen yu"),  
  age = c(28, 18, 23, 20, 29, 30),  
  height = c(171, 166, 169, 182, 183, 178)  
)
```

```
> daf1
```

	name	age	height
1	wang feng	28	171
2	zhang yan	18	166
3	liu jun	23	169
4	li zhenying	20	182
5	zhang yu	29	183
6	chen yu	30	178

- 数据结构就是以特定形式将一些元素(**element**)组合起来。对**向量(vector)**来说，其元素是单个数值；**矩阵(matrix)**和**数组(array)**本质上以向量的形式存储，其元素仍是单个数值；**数据框(data.frame)**是由若干个不同属性的向量元素组成的，**列表(list)**的要求则最为宽泛，任何结构和类型的数据都可以组合成一个列表。
- **原子型数据(atomic objects)**指向量、矩阵和数组，这类数据的所有元素属于同一类型。列表和数据框不是原子型数据，它们属于**递归对象(recursive objects)**。

- 生成随机数据

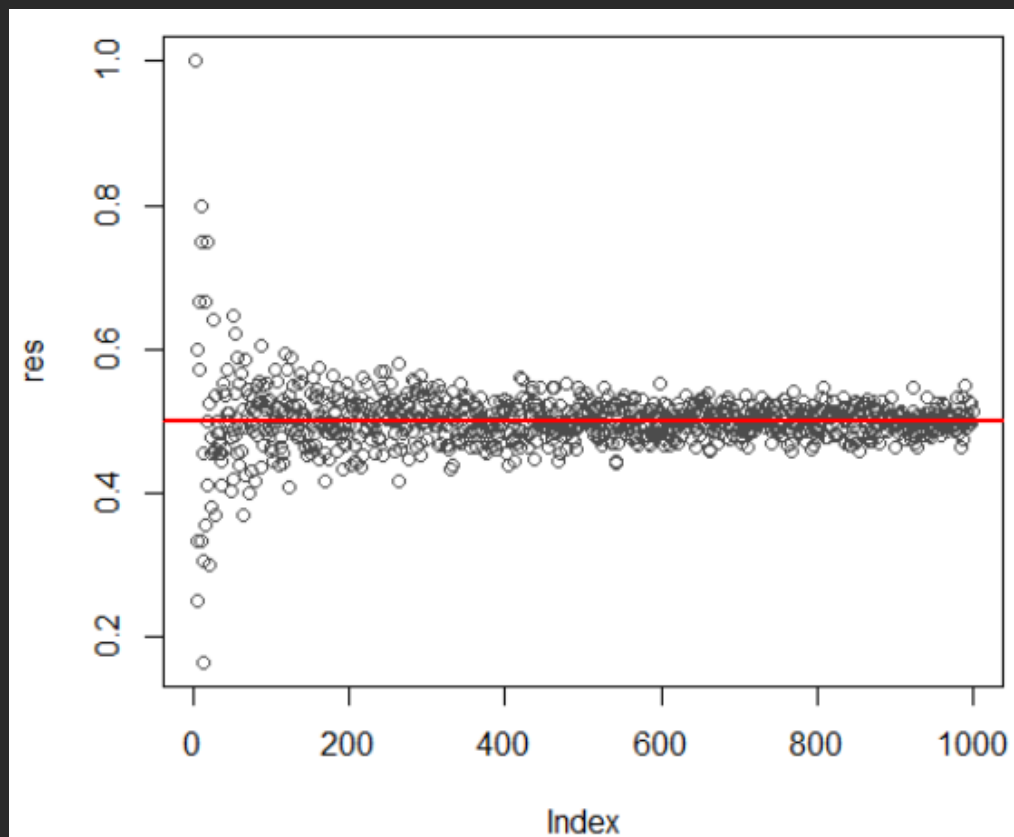
sample()

实验设计或统计分析时可能用到的随机序列

```
> set.seed(1)
> sample(1:9, size=9)
[1] 3 9 5 6 2 4 8 7 1
> sample(1:9, size=9)
[1] 1 2 8 5 7 4 6 3 9
> set.seed(1)
> sample(1:9, size=9)
[1] 3 9 5 6 2 4 8 7 1
> sample(1:9, size=6)
[1] 1 2 8 5 7 4
> set.seed(1)
> sample(1:9, size=6)
[1] 3 9 5 6 2 4
```

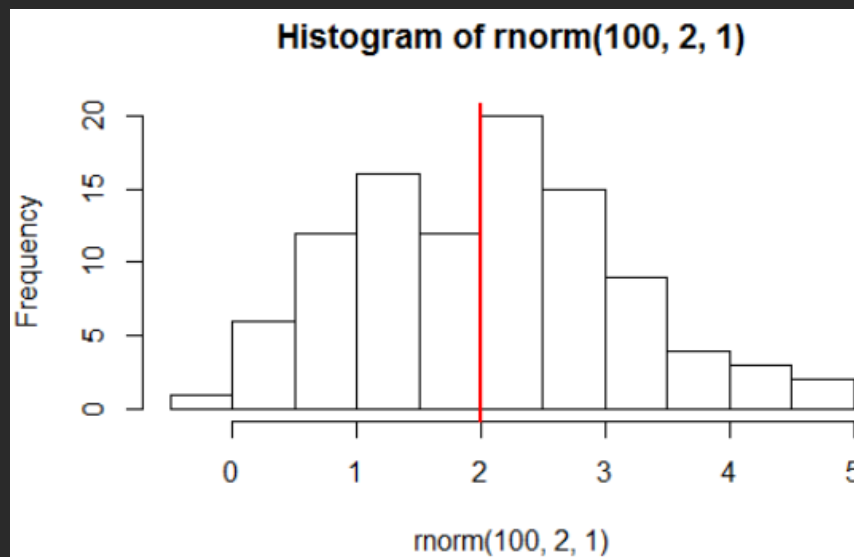

放回式抽样 (& for循环举例)

```
N=1000
res=NULL
for(i in 1:N){
  res[i]=sum(sample(c(0,1),
                    size=i,
                    replace=TRUE)
             )/i
}
```



```
rnorm      # 从正态分布中抽样  
rpois      # 从泊松分布中抽样  
rbinom     # 从二项分布中抽样  
rnbinom    # 从负二项分布中抽样  
rexp       # 从指数分布中抽样  
rt         # 从t-分布中抽样  
runif      # 从均匀分布中抽样
```

```
> head(rnorm(100,2,1), 20)  
[1] 2.8508669 1.4738039 2.5726813 2.2953568 2.6564098 1.5574977  
[7] 1.8841061 2.2484500 2.7230385 3.5543523 1.9836286 2.6632191  
[13] 2.3350653 2.8385062 1.2171247 2.7912601 -0.5682313 1.2073326  
[19] 0.8960981 3.1531415
```



◆ 数据的属性、索引和整理

向量和列表型数据(列表和数据框)具有**names** 属性, 存储对应元素的名称标签; 数组或矩阵有**dimnames**属性; 数据框除**names** 属性外还有**row.names** 属性。这些属性都可以用同名函数提取或设置。

colnames 和**rownames** 函数用于矩阵和数据框这类数据表形式的对象, 分别获取行名和列名。

- **class()**

```
> class(daf1)
[1] "data.frame"
```

- **str()**

head(); tail()

```
> str(daf1)
'data.frame':   6 obs. of  3 variables:
 $ name  : Factor w/ 6 levels "chen yu","li zhenying",...: 4 5 3 2 6 1
 $ age   : num  28 18 23 20 29 30
 $ height: num  171 166 169 182 183 178
```

- **length()**

```
> length(daf1)
[1] 3
```

```
> colnames(daf1)
[1] "name"  "age"   "height"
> rownames(daf1)
[1] "1" "2" "3" "4" "5" "6"
```

- 此外，R 对象还可以有很多种属性，例如 **comment** 函数可以用来获取及设置 "comment" 属性。**attributes** 函数用来列出 R 对象的所有属性。**attr()** 函数则可以获取及设置指定属性的值。用户也可以添加新的属性。

```
> comment(daf1)
NULL
```

```
> comment(daf1) <- "一个属性说明举例"
> str(daf1)
'data.frame':  6 obs. of  3 variables:
 $ name  : Factor w/ 6 levels "chen yu","li
 $ age   : num  28 18 23 20 29 30
 $ height: num  171 166 169 182 183 178
- attr(*, "comment")= chr "一个属性说明举例"
```

```
> attributes(daf1)
$`names`
[1] "name"  "age"   "height"

$class
[1] "data.frame"

$row.names
[1] "WF" "ZY" "LJ" "LZ" "ZY" "CY"

$comment
[1] "一个属性说明举例"
```

```
attr(daf1,"row.names") <- c("WF","ZY","LJ","LZ","ZY","CY")
```

- 对象的 **class** 属性是面向对象编程的基础，泛型函数就是根据对象的 **class** 属性为不同结构的对象提供不同的操作。

➤ 索引和调用

- 下标操作符

[]、 [[]]、 \$、 @ 等，用以获取数据的一部分。

[]:

常用于索引一个向量, 索引值可以是正值，负值，逻辑值或字符串。

矩阵和数组本质是以向量形式存储的，所以适用于向量的索引方式也同样适用于矩阵和数组。最常用的方式是使用逗号分开的一组下标，空值下标表示获取该维度上所有元素。

```

> letters[2]
[1] "b"
> letters[2:7]
[1] "b" "c" "d" "e" "f" "g"
> tag=seq(from=1, to=26, by=2)
> LETTERS[tag]
[1] "A" "C" "E" "G" "I" "K" "M" "O" "Q" "S" "U" "W" "Y"
> LETTERS[tag+1]
[1] "B" "D" "F" "H" "J" "L" "N" "P" "R" "T" "V" "X" "Z"

```

```

> letters[27]
[1] NA

```

```

> mat
  A B C
a 1 2 3
b 4 5 6

```

```

> mat[2,1]
[1] 4
> mat[,2]
a b
2 5
> mat[2,]
A B C
4 5 6

```

```

> mat[, -1]
  B C
a 2 3
b 5 6
> mat[-1,]
A B C
4 5 6

```

对数据框和列表而言，也可使用 `x[i]` 的形式索引，索引结果是对象的若干个元素，索引结果的**class**和被索引对象相同。对数据框来说，则是获取数据框的列。

- `[[]]`和`$`

获取数据框或列表的单个元素，索引结果的**class** 也就是元素的**class**。

```
> daf1[, 2]
[1] 28 18 23 20 29 30
> daf1[1,1]
[1] wang feng
Levels: chen yu li zhenying liu jun wa
> daf1$age
[1] 28 18 23 20 29 30
> daf1[, "age"]
[1] 28 18 23 20 29 30
```

- R中自带的一个数据iris

```
> head(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
```

```
> iris$Sepal.Length
 [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4 5.1
[19] 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4 5.2 5.5 4.9 5.0
[37] 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.6 5.3 5.0 7.0 6.4 6.9 5.5
[55] 6.5 6.2 5.9 6.5 5.4 4.7 4.7 6.7 6.3 6.6 5.8 6.1 6.9 3.3 6.7 3.0 6.8 6.7
[73] 8.0 6.7 6.0 5.7 5.5 5.5 5.8 6.0 5.4 6.0 6.7 6.3 5.6 5.5 5.1 5.8 3.4 4.1
[91] 6.5 5.7 5.7 6.2 5.1 5.7 6.3 5.8 7.1 6.3 6.5 7.6 4.9 7.3 7.7 4.8 6.0 4.9
[109] 8.0 5.7 5.8 6.4 6.5 7.7 7.7 6.0 6.9 5.6 7.7 6.3 6.7 7.2 7.7 4.9 6.5 5.9
[127] 4.7 7.9 6.4 6.3 6.1 7.7 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8 6.7 3.0 6.8 6.7
[145] 2.0 5.9

> iris$
```



```
> str(iris)
'data.frame': 150 obs. of 5 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species : Factor w/ 3 levels "setosa","versicolor",...: 1 1
1 ...
```

```
> str(iris[2])
'data.frame': 150 obs. of 1 variable:
 $ Sepal.Width: num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1
> str(iris[,2])
num [1:150] 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
> str(iris[["Sepal.Width"]])
num [1:150] 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
```

```
> c(class(iris[2]),class(iris$Sepal.Width))
[1] "data.frame" "numeric"
```

```
> c(class(lst1["first"]),class(lst1[["first"]]))
[1] "list" "integer"
```

➤ 列联表

table()

频数统计:

```
> table(iris$Sepal.Width)
```

```
 2 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9  3 3.1 3.2
1   3   4   3   8   5   9  14  10 26  11  13
3.3 3.4 3.5 3.6 3.7 3.8 3.9   4 4.1 4.2 4.4
6  12   6   4   3   6   2   1   1   1   1
```

```
> table(iris$Species)
```

```
      setosa versicolor virginica
      50         50         50
```

```
> sum(table(iris$Sepal.Width))
```

```
[1] 150
```

```
> class(table(iris$Sepal.Width))
```

```
[1] "table"
```

```
> table(iris$Sepal.Width,iris$Species)
```

	setosa	versicolor	virginica
2	0	1	0
2.2	0	2	1
2.3	1	3	0
2.4	0	3	0
2.5	0	4	4
2.6	0	3	2
2.7	0	5	4
2.8	0	6	8
2.9	1	7	2
3	6	8	12
3.1	4	3	4

大于两个因子时: **ftable()**

计算频率而非频数

prob.table()

```
prop.table(counts,1) # 按行计算频率  
prop.table(counts,2) # 按列计算频率  
prop.table(counts)   # 计算占总和的比例
```

计算各维度上的统计量，如总和、均值、标准差、分位数等。同样适用于矩阵数据

```
colSums # 按列求和  
colMeans # 按列求均值  
rowSums # 按行求和  
rowMeans # 按行求均值
```

apply()

```
> apply(iris[,1:3], 2, sum)  
Sepal.Length Sepal.Width Petal.Length  
      876.5      458.6      563.7  
> apply(iris[,1:3], 2, mean)  
Sepal.Length Sepal.Width Petal.Length  
    5.843333    3.057333    3.758000
```

根据若干个分类变量，将一个连续变量分成若干个部分，分别计算各个部分的统计量：

aggregate()

```
> aggregate(iris$Sepal.Length,by=list(iris$Species),mean)
  Group.1      x
1   setosa 5.006
2 versicolor 5.936
3  virginica 6.588
> aggregate(Sepal.Length~Species,iris,mean)
  Species Sepal.Length
1   setosa      5.006
2 versicolor      5.936
3  virginica      6.588
```

◆ 管道操作及{dplyr}包 (拓展, 不讲)

%>%、%T>%、%\$%、%<>%

<https://4va.github.io/biodatasci/r-dplyr-yeast.html>

JMU2017 [Home](#) [Setup](#) [Data](#) [Lessons](#)

Review

- Our data
- Reading in data
- The dplyr package
- dplyr verbs

Data Manipulation with dplyr

Data analysis involves a large amount of [janitor work](#) – munging and cleaning data to facilitate downstream data analysis. This lesson demonstrates techniques for advanced data manipulation and analysis with the split-apply-combine strategy. We will use the dplyr

◆运算符的优先级

()
^
:
* /
+ -
== > <
!
& |
|| &&
=
先
↓
后

```
> 1:3^2+4  
[1] 5 6 7 8 9 10 11 12 13  
> (1:3)^2+4  
[1] 5 8 13  
> 1:3*2+4  
[1] 6 8 10  
> 1:(3*2)+4  
[1] 5 6 7 8 9 10  
> 1:(3*2+4)  
[1] 1 2 3 4 5 6 7 8 9 10
```

◆流程控制

for	循环控制 for(i in seq) { }
while(cond)	满足条件循环
replicate	表达式的重复运算
if(cond)	cond为真时，运行if后面的表达式，否则运行
if {} else {}	else后的表达式
ifelse	同上，更加灵活
break	循环中止
Stop	函数中止

```
> iris$Petal.Length
[1] 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 1.5 1.6 1.4 1.1 1.2 1.5
[17] 1.3 1.4 1.7 1.5 1.7 1.5 1.0 1.7 1.9 1.6 1.6 1.5 1.4 1.6 1.6 1.5
[33] 1.5 1.4 1.5 1.2 1.3 1.4 1.3 1.5 1.3 1.3 1.3 1.3 1.6 1.9 1.4 1.6 1.4
[49] 1.5 1.4 4.7 4.5 4.9 4.0 4.6 4.5 4.7 3.3 4.6 3.9 3.5 4.2 4.0 4.7
[65] 3.6 4.4 4.5 4.1 4.5 3.9 4.8 4.0 4.9 4.7 4.3 4.4 4.8 5.0 4.5 3.5
[81] 3.8 3.7 3.9 5.1 4.5 4.5 4.7 4.4 4.1 4.0 4.4 4.6 4.0 3.3 4.2 4.2
[97] 4.2 4.3 3.0 4.1 6.0 5.1 5.9 5.6 5.8 6.6 4.5 6.3 5.8 6.1 5.1 5.3
[113] 5.5 5.0 5.1 5.3 5.5 6.7 6.9 5.0 5.7 4.9 6.7 4.9 5.7 6.0 4.8 4.9
[129] 5.6 5.8 6.1 6.4 5.6 5.1 5.6 6.1 5.6 5.5 4.8 5.4 5.6 5.1 5.1 5.9
[145] 5.7 5.2 5.0 5.2 5.4 5.1
```

```
> ifelse(iris$Petal.Length >= 5, 1,
+       ifelse(iris$Petal.Length > 3, 2, 3))
[1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[33] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2
[65] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[97] 2 2 3 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 2 2
[129] 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```


➤逻辑比较

- `>`, `<`, `==`, `!=`, `>=`, `<=`
- `identical()`
- `&` 向量化的逻辑与
 `&&` 逻辑与
- `|` 向量化的逻辑或
 `||` 逻辑或
- `!` 逻辑非
- `xor()` 异或
- `all()` 当所有值为TRUE, 结果才是TRUE
 `any()` 只要有值为TRUE, 结果就是TRUE

➤ apply 系列

- **apply()** 把数组、矩阵按照指定的维度分成若干个部分，再应用指定的函数。
- **tapply()** 把一个向量按照若干个因子分成若干个部分，再应用指定的函数。
- **split()** 按照若干个因子，将向量或数据框分成若干个部分，产生一个列表。
- **lapply()** / **sapply()** 对列表的每个元素应用指定的函数。
sapply 是 **lapply** 的特化，更加容易使用。
- **by()** 把数据框根据指定的因子分成若干个部分，再应用指定的函数。
- **mapply()** 用不同的参数组合运行同一个函数。也就是以向量化的方式运行这个函数

```
> tapply(iris$Sepal.Length,iris$Species,mean)
      setosa versicolor virginica 
      5.006      5.936      6.588
```

```
> lapply(1st,function(x) apply(x,2,mean))
$`setosa`
Sepal.Length Sepal.Width Petal.Length Petal.Width
      5.006      3.428      1.462      0.246

$versicolor
Sepal.Length Sepal.Width Petal.Length Petal.Width
      5.936      2.770      4.260      1.326

$virginica
Sepal.Length Sepal.Width Petal.Length Petal.Width
      6.588      2.974      5.552      2.026
```

```
> sapply(1st,function(x) apply(x,2,mean))
      setosa versicolor virginica 
Sepal.Length  5.006      5.936      6.588 
Sepal.Width   3.428      2.770      2.974 
Petal.Length  1.462      4.260      5.552 
Petal.Width   0.246      1.326      2.026
```

```
> by(iris[-5],iris$Species,function(x) apply(x,2,mean))
iris$Species: setosa
Sepal.Length Sepal.Width Petal.Length Petal.Width
      5.006      3.428      1.462      0.246
-----
iris$Species: versicolor
Sepal.Length Sepal.Width Petal.Length Petal.Width
      5.936      2.770      4.260      1.326
-----
iris$Species: virginica
Sepal.Length Sepal.Width Petal.Length Petal.Width
      6.588      2.974      5.552      2.026
```

Part 3

用R实现基础统计

这里不系统讲统计学知识，只讲在R里的基本运用

- 概率统计 (Probability)

随机取样, `sample()`

```
> sample(10,5)
[1] 10  3  6  2  8
```

从10个球中有序地依次取出5个, 每个事件的概率:

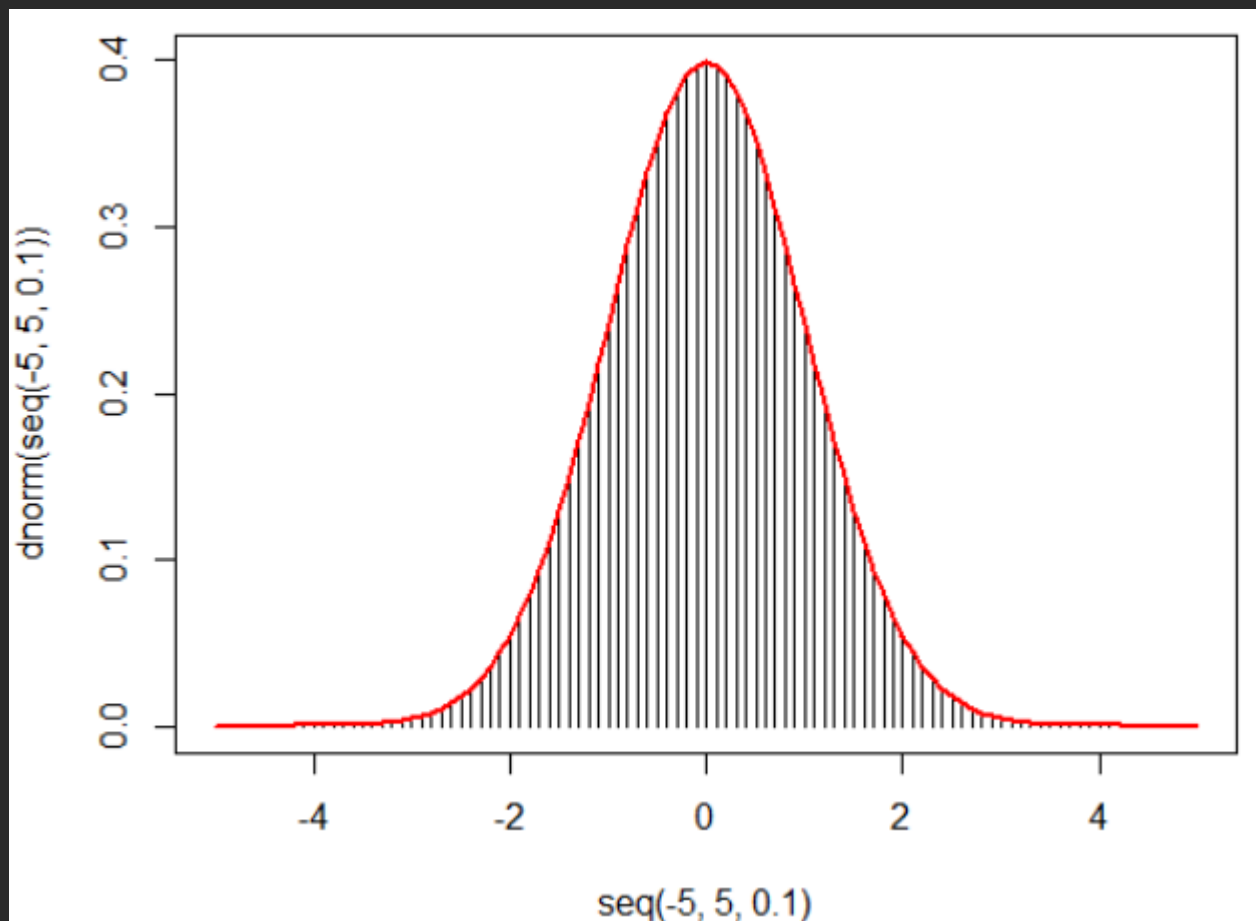
```
> 1/prod(10:6)
[1] 3.306878e-05
```

一次性取出5个球为一个组合, 则概率为:

```
> prod(5:1)/prod(10:6) 或 > 1/choose(10,5)
[1] 0.003968254          [1] 0.003968254
```

- 数据分布密度

```
> plot(seq(-5,5,.1),dnorm(seq(-5,5,.1)),type="h")  
> lines(seq(-5,5,.1),dnorm(seq(-5,5,.1)),type="l",col=2,lwd=2)
```



- 数据检验

正态W检验方法`shapiro.test()`，提供了W的统计量和检验P值，我们当 $P > 0.05$ 时候，我们可以认为这组数据是正态的。（一般适用条件是所测数据量在 3 and 5000 之间）

```
> shapiro.test(iris$Sepal.Length)

      Shapiro-Wilk normality test

data:  iris$Sepal.Length
W = 0.97609, p-value = 0.01018
```

```
> shapiro.test(rnorm(100))

      Shapiro-Wilk normality test

data:  rnorm(100)
W = 0.98298, p-value = 0.2253
```

◆一些基本数学函数

三角和幂函数:

- `sin`, `cos`, `tan`, `asin`, `acos`, `atan`
- `log`, `log10`, `exp`, `sqrt`, `log(x, base)`

截尾函数(`round`, `ceiling`):

```
> ceiling(c(2.400, 3.5868))  
[1] 3 4  
> round(c(2.400, 3.5868),2)  
[1] 2.40 3.59
```

算术操作符(`%%`, `%/%`):

```
> c(1:10)%%2 ## 求余数  
[1] 1 0 1 0 1 0 1 0 1 0  
> c(1:10)%/%2 # 求整数  
[1] 0 1 1 2 2 3 3 4 4 5
```

集合:

`union()`
`intersect()`
`setdiff()`

```
> union(1:5,2:6)  
[1] 1 2 3 4 5 6  
> intersect(1:5,2:6)  
[1] 2 3 4 5  
> setdiff(1:5,2:6)  
[1] 1  
> setdiff(2:6,1:5)  
[1] 6
```


◆ 一些常用的基础统计函数使用格式

数值计算

`log(x)`

`log10(x)`

`exp(x)`

`sin(x)`

`cos(x)`

`tan(x)`

`asin(x)`

`acos(x)`

`min(x)`

`max(x)`

`range(x)`

`length(x)`

统计检验

`mean(x)`

`sd(x)`

`var(x)`

`median(x)`

`quantile(x,p)`

`cor(x,y)`

`t.test()`

`lm(y ~ x)`

`wilcox.test()`

`kruskal.test()`

统计检验

`lm(y ~ f+x)`

`lm(y ~ x1+x2+x3)`

`bartlett.test`

`binom.test`

`fisher.test`

`chisq.test`

`glm(y ~ x1+x2+x3,
 binomial)`

`friedman.test`

查看功能和用法: ? + 函数名

- 编写一个能返回基本统计量的函数

计算函数：最大值`max`，最小值`min`，区间`range`，和`sum`，均值`mean`，中位数`median`，偏度`skewness`，峰度`kurtosis`，分位数`quantile`，标准差`sd`，标准误`se`

```
simple.stats = function(x) {  
  library(moments)  
  library(sciplot)  
  list(max = max(x),  
        min = min(x),  
        sum = sum(x),  
        mean = mean(x),  
        median = median(x),  
        skewness = skewness(x),  
        kurtosis = kurtosis(x),  
        quantile = quantile(x),  
        sd = sd(x),  
        se = se(x))  
}
```

```
> simple.stats(iris$Petal.Length)  
$`max`  
[1] 6.9  
  
$min  
[1] 1  
  
$sum  
[1] 563.7  
  
$mean  
[1] 3.758  
  
$median  
[1] 4.35  
  
$skewness  
[1] -0.2721277  
  
$kurtosis  
[1] 1.604464  
  
$quantile  
      0%  25%  50%  75% 100%  
1.00 1.60 4.35 5.10 6.90  
  
$sd  
[1] 1.765298  
  
$se  
[1] 0.144136
```

◆ 相关分析

- 相关程度的度量:

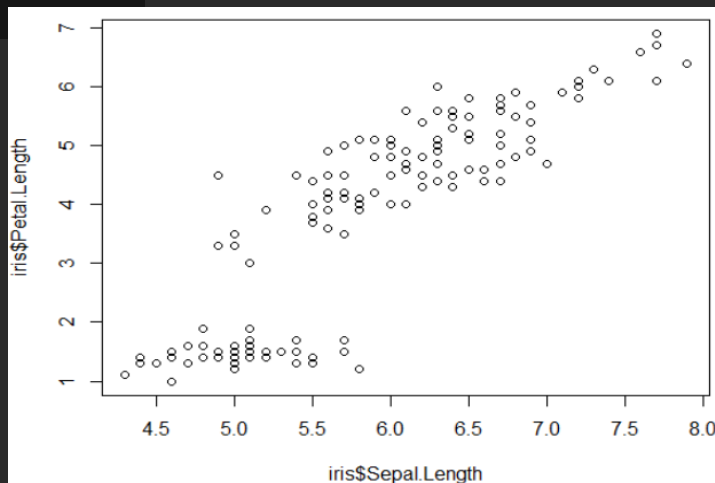
Pearson 相关系数 (**Pearson's correlation coefficient r**)

Kendall 秩相关系数 (**Kendall's rank correlation τ**)

Spearman 秩相关系数 (**Spearman rank correlation coefficient**
或**Spearman's ρ**)

- **cor()** `cor(x, y = NULL, use = "everything",
method = c("pearson", "kendall", "spearman"))`

```
> cor(iris$Sepal.Length, iris$Petal.Length)  
[1] 0.8717538
```



◆ t检验

• t.test()

单样本，独立样本方差齐、不齐，配对样本

```
iris[c(1:50,101:150),c(1,5)]->newiris  
newiris$Species<-factor(newiris$Species,  
                        levels=c("setosa","virginica"))  
bartlett.test(newiris$Sepal.Length~newiris$Species)
```

```
> bartlett.test(newiris$Sepal.Length~newiris$Species)
```

Bartlett test of homogeneity of variances

data: newiris\$Sepal.Length by newiris\$Species

Bartlett's K-squared = 15.987, df = 1, p-value = 6.379e-05

```
t.test(newiris$Sepal.Length~newiris$Species,  
      paired=F,var.equal=F)
```

Welch Two Sample t-test

data: newiris\$Sepal.Length by newiris\$Species

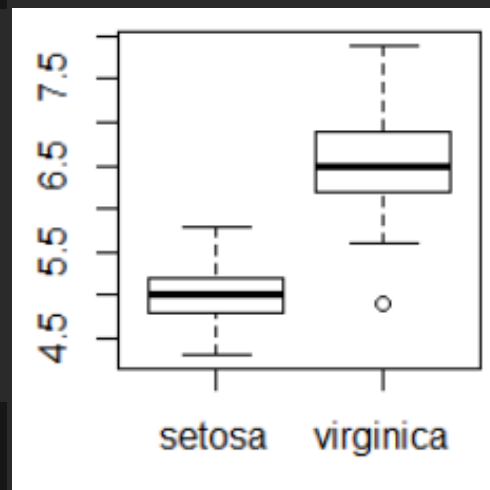
t = -15.386, df = 76.516, p-value < 2.2e-16

alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:

-1.78676 -1.37724

sample estimates:

mean in group setosa	mean in group virginica
5.006	6.588



- 针对线性模型（符合近正态分布、方差齐性）的方差齐性检验，用Levene检验：
- `leveneTest()` {car}

格式： `leveneTest (y ~ X1 * X2, data=data)`

◆ 一般线性模型 (General Linear models)

- 适用的假设前提

- 1) 变量的独立性，非独立的变量会有很多意想不到的结果出现；
- 2) 正态的残差分布；
- 3) 方差齐性；
- 4) 小的采样误差。

- 线性回归

线性模型：自变量是连续变量（即数值型）；

ANOVA：变量是因子型的（如不同加氮处理）；

ANCOVA：既有因子变量又有连续变量。

➤线性回归

lm()

Usage

```
lm(formula, data, subset, weights, na.action,  
    method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE,  
    singular.ok = TRUE, contrasts = NULL, offset, ...)
```

```
> lm.model1<- lm(Sepal.Length~Petal.Length,data=iris)
```



```
> summary(lm.model)
```

```
Call:
lm(formula = Sepal.Length ~ Petal.Length, data = iris)
```

模型公式

```
Residuals:
```

Min	1Q	Median	3Q	Max
-1.24675	-0.29657	-0.01515	0.27676	1.00269

模型残差分布的分位数

```
Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	4.30660	0.07839	54.94	<2e-16 ***
Petal.Length	0.40892	0.01889	21.65	<2e-16 ***

模型系数

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

显著水平

```
Residual standard error: 0.4071 on 148 degrees of freedom
```

残差标准误

```
Multiple R-squared:  0.76,    Adjusted R-squared:  0.7583
```

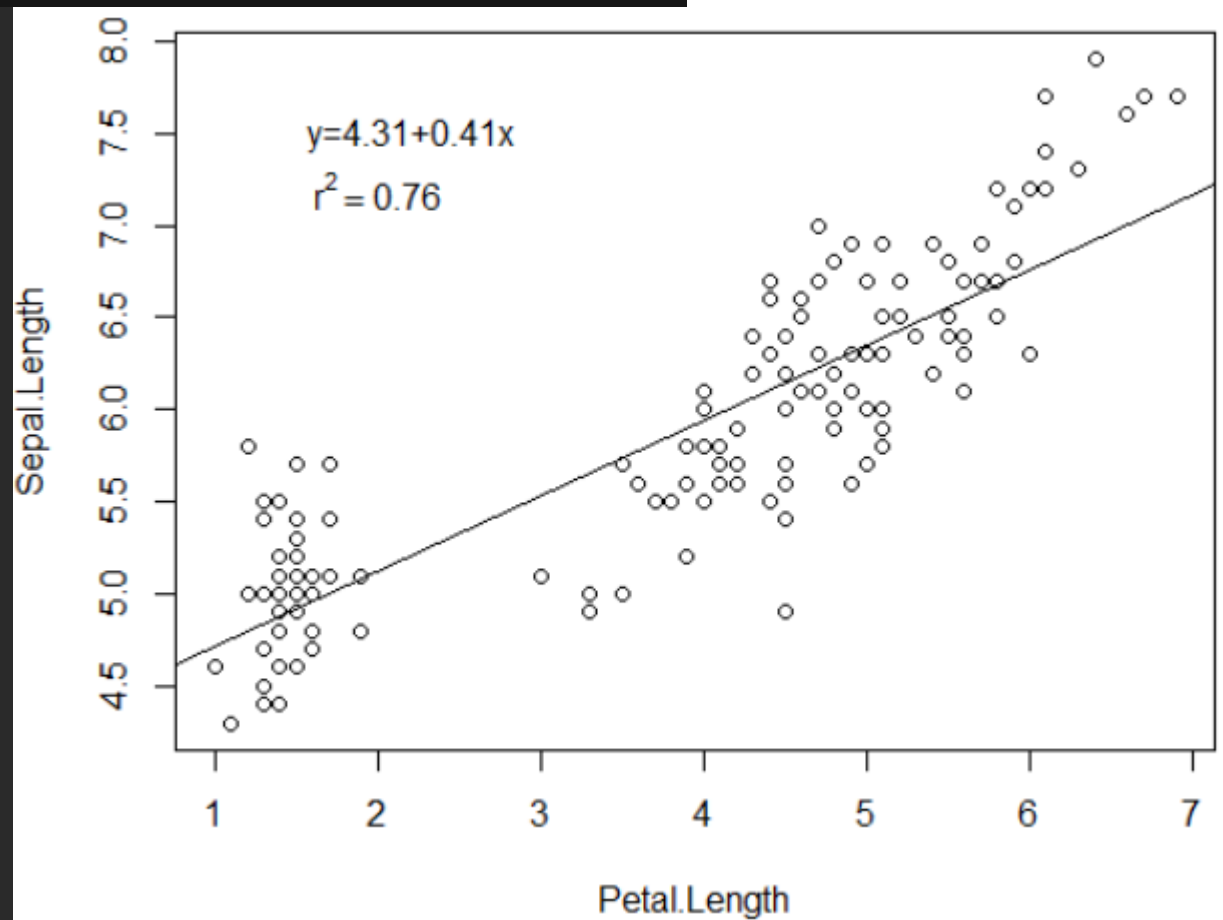
模型的拟合解释能力

```
F-statistic: 468.6 on 1 and 148 DF,  p-value: < 2.2e-16
```

模型整体评价

```
plot(Sepal.Length~Petal.Length,data=iris)
abline(a=4.30660, b=0.40892)
```

```
text(2.2,7.5,"y=4.31+0.41x")
text(2,7.2,expression(r^2==0.76))
```



可以方差分析表的形式显示结果

```
> summary.aov(lm.model)
              Df Sum Sq Mean Sq F value Pr(>F)
Petal.Length   1  77.64   77.64   468.6 <2e-16 ***
Residuals    148  24.53    0.17
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

提取结果

```
> class(summary(lm.model))
[1] "summary.lm"
```

```
> summary(lm.model)[[4]]
              Estimate Std. Error t value      Pr(>|t|)
(Intercept)  4.3066034  0.07838896  54.93890 2.426713e-100
Petal.Length  0.4089223  0.01889134  21.64602  1.038667e-47
```

```
> summary(lm.model)$r.squared
[1] 0.7599546
```

```
> coef(lm.model)
(Intercept) Petal.Length
  4.3066034   0.4089223
```

- 模型评价

有很多的评判标准，其中比如高的拟合 R^2 ，相对较少的因子，低的AIC值（相对于同一的应变变量Y值）。

```
> AIC(lm.model)
[1] 160.0404
```

- 残差

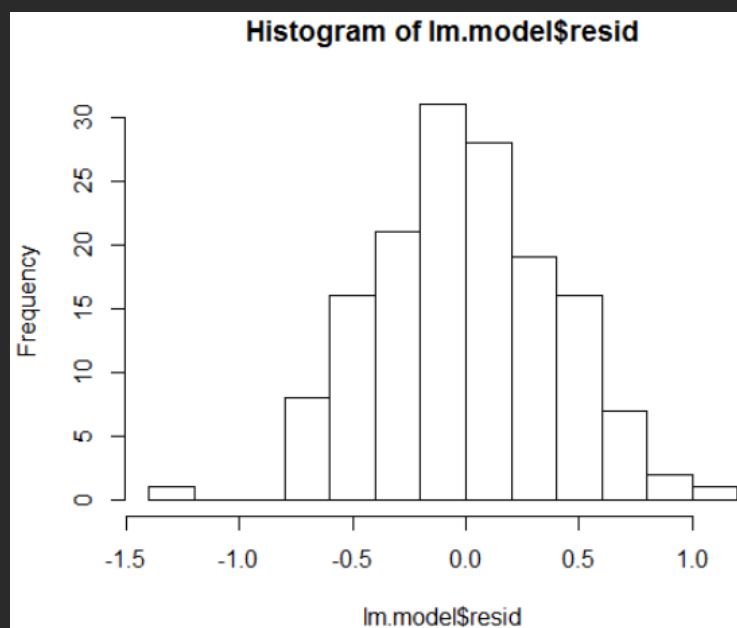
模型的预测值与实际值之间的差值，即余下模型没有解释的部分。好的模型，残差的散点图是符合正态分布的。

```
> shapiro.test(lm.model$resid)

Shapiro-Wilk normality test

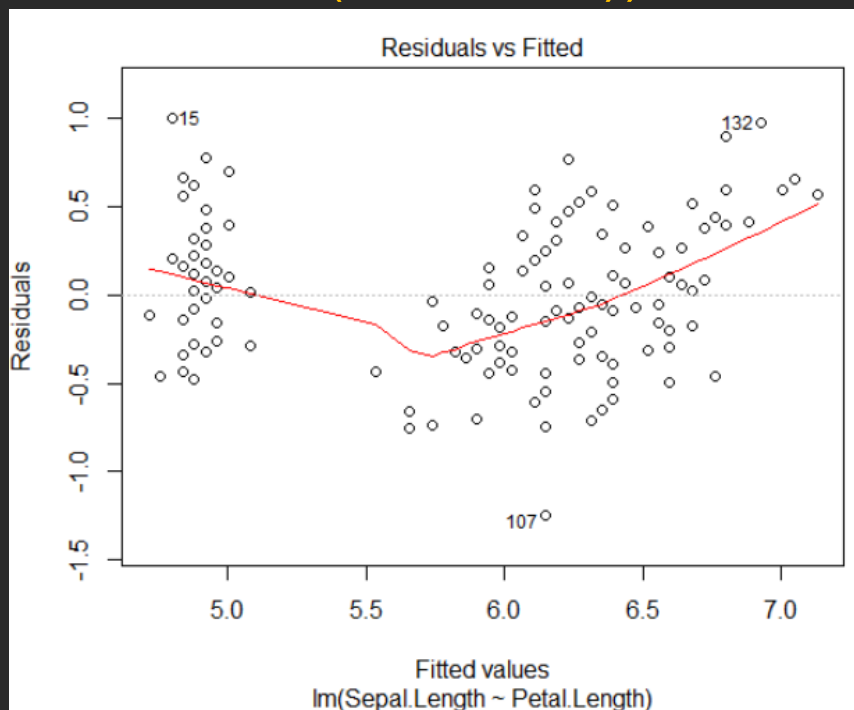
data:  lm.model$resid
W = 0.99298, p-value = 0.6767
```

```
> hist(lm.model$resid)
```

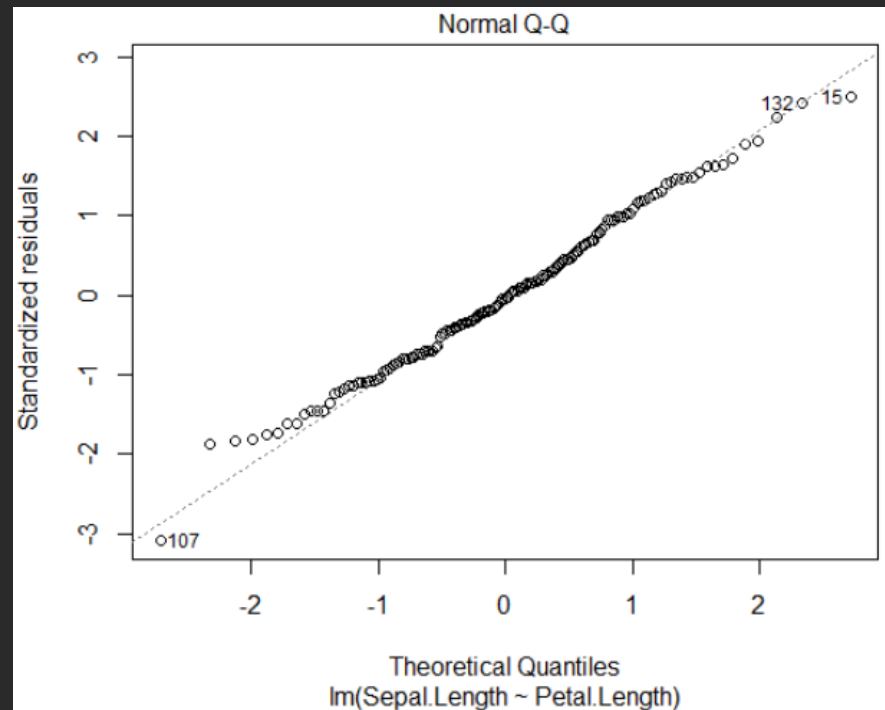


- 可看下残差的和预测值关系，来评判其模型拟合效果

```
plot(fitted(lm.model),  
     resid(lm.model))
```



```
qqnorm(lm.model$resid)
```



➤ 方差分析

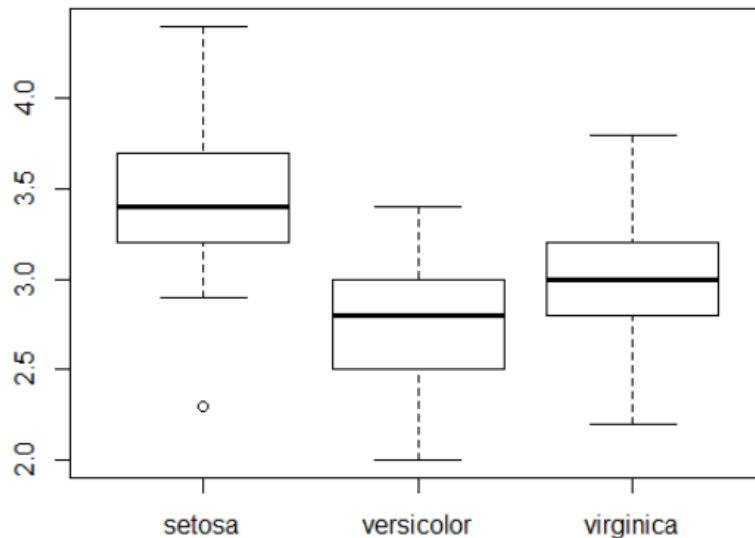
aov()

```
> dis.aov<- aov(Sepal.Width~Species,data=iris)
> summary(dis.aov)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
Species	2	11.35	5.672	49.16	<2e-16	***
Residuals	147	16.96	0.115			

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

boxplot()



`summary(aov())`的结果表示的是在所有个水平上具有差异(如上面的Species有3个水平), 但并不是指各水平两两具有差异, 看两两间是否有差异则需要多重比较。

其原理用的是每组数据进行t检验那样, 但是当因素水平较多时, 检验同时进行, 容易造成误差, 就需要对显著性进行调整, 即 P 值进行调整;

在R中有自动调整 P 值的函数, 在`pairwise.t.test()`内可以设置。

Description

Calculate pairwise comparisons between group levels with corrections for multiple testing

Usage

```
pairwise.t.test(x, g, p.adjust.method = p.adjust.methods,
               pool.sd = !paired, paired = FALSE,
               alternative = c("two.sided", "less", "greater"),
               ...)
```

两两比较

```
> pairwise.t.test(iris$Sepal.Width,iris$Species)

Pairwise comparisons using t tests with pooled SD

data:  iris$Sepal.Width and iris$Species

      setosa versicolor
versicolor < 2e-16 -
virginica  9.1e-10 0.0031

P value adjustment method: holm
```

各组的方差齐性检验

```
> bartlett.test(iris$Sepal.Width,iris$Species)

Bartlett test of homogeneity of variances

data:  iris$Sepal.Width and iris$Species
Bartlett's K-squared = 2.0911, df = 2, p-value = 0.3515
```


TukeyHSD()

```
> TukeyHSD(aov(Sepal.Width~Species,data=iris))
```

Tukey multiple comparisons of means

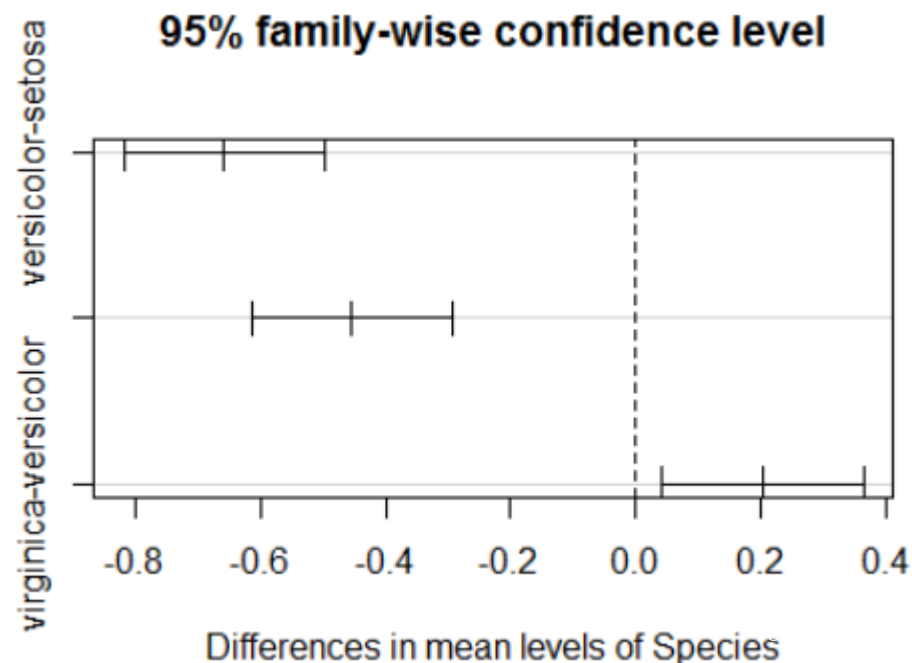
95% family-wise confidence level

Fit: aov(formula = Sepal.Width ~ Species, data = iris)

\$`Species`

	diff	lwr	upr	p adj
versicolor-setosa	-0.658	-0.81885528	-0.4971447	0.0000000
virginica-setosa	-0.454	-0.61485528	-0.2931447	0.0000000
virginica-versicolor	0.204	0.04314472	0.3648553	0.0087802

```
> plot(TukeyHSD(  
+ aov(Sepal.Width~Species,data=iris)  
+ ))
```



{multcomp}包

glht()

```
> summary(glht(aov(Sepal.Width~Species,data=iris), linfct = mcp(Species ="Tukey")))
```

Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts

Fit: aov(formula = Sepal.Width ~ Species, data = iris)

Linear Hypotheses:

	Estimate	Std. Error	t value	Pr(> t)	
versicolor - setosa == 0	-0.65800	0.06794	-9.685	<1e-04	***
virginica - setosa == 0	-0.45400	0.06794	-6.683	<1e-04	***
virginica - versicolor == 0	0.20400	0.06794	3.003	0.0087	**

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Adjusted p values reported -- single-step method)

- 非参数检验

两组独立数据可用Wilcoxon秩和检验：

`wilcox.test(y ~ X, data)` 或 `wilcox(y1, y2)`

多于两个水平时可用Kruskal-Wallis检验：

`kruskal.test(y ~ A, data)` 或 `kruskal.test(y ~ A|B, data)`

- 双因素方差分析

不考虑交互作用

```
fit <- aov (y ~ A + B, data=data)
```

考虑交互作用

```
fit <- aov(y ~ A + B + A:B, data=data) 或
```

```
fit <- aov (y ~ A * B, data=data)
```

(A、B为两个分类变量)

➤多元线性回归模型筛选、方差分解

```
modelfit0 <- lm (y ~ x1 + x2 + x3 + ... +xn)
```

AIC()

update()

drop1()

add1()

step()

stepAIC {MASS}

varpart {vegan}

Description

Select a formula-based model by AIC.

Usage

```
step(object, scope, scale = 0,  
      direction = c("both", "backward", "forward"),  
      trace = 1, keep = NULL, steps = 1000, k = 2, ...)
```

Description

Performs stepwise model selection by AIC.

Usage

```
stepAIC(object, scope, scale = 0,  
         direction = c("both", "backward", "forward"),  
         trace = 1, keep = NULL, steps = 1000, use.start = FALSE,  
         k = 2, ...)
```

➤ 嵌套模型比较

- 似然比检验 likelihood ratio test

lrtest {lmtest}, {epicalc}

```
data("USDistLag")
library(dplyr) ## 运用前面提到的管道操作符写代码
usd1<- USDistLag %>%
  stats::lag(k=-1) %>% ## 同一个函数出现在不同包且冲突时
  cbind(USDistLag, .) %>%
  na.contiguous
colnames(usd1) <- c("con", "gnp", "con1", "gnp1")
fm1 <- lm(con ~ gnp + gnp1, data = usd1)
fm2 <- lm(con ~ gnp + con1 + gnp1, data = usd1)
library(lmtest)
lrtest(fm2, fm1)
```

```
> lrtest(fm2, fm1)
Likelihood ratio test

Model 1: con ~ gnp + con1 + gnp1
Model 2: con ~ gnp + gnp1
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -56.069
2    4 -65.871 -1  19.605  9.524e-06 ***
---
```

➤混合效应模型

应变变量

~

固定因子

+

随机因子

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \varphi_k + \varepsilon_{ijk} \quad (\text{双因子试验})$$

lmer {lme4}

Modelfit = lmer (y ~ A * B + (1|RI), data)

lme {nlme}

Modelfit = lme (y ~ A * B, random = ~1|RI, data)

◆ 广义线性模型 (Generalized Linear models)

glm()

Fitting Generalized Linear Models

Description

`glm` is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.

Usage

```
glm(formula, family = gaussian, data, weights, subset,  
     na.action, start = NULL, etastart, mustart, offset,  
     control = list(...), model = TRUE, method = "glm.fit",  
     x = FALSE, y = TRUE, singular.ok = TRUE, contrasts = NULL, .
```


分布族 (family)	连接函数
binomial	logit, probit, cloglog
gaussian	identity
Gamama	Identity, inverse, log
Inverse.gaussian	$1/\mu^2$
poisson	Identity, log, sqrt
quasi	logit,probit,cloglog,identity,inverse ,log, $1/\mu^2$,sqrt

- 在不改变应变量 y 和自变量 x 之间的线性关系，通过一个连接函数linkfunction 将自变量线性连接到应变量 y 的关系中。
- glm模型的获得的系数和通过summary()获得的结果并不是原始的响应变量所获得的参数，而是转换后的模型所获得的参数，因此需要通过连接函数的反函数去获得对应的原始Y值的参数，或者使用predict()里面的参数设置成为type="responses"。

```
> summary(glm(Sepal.Length~Sepal.Width+Petal.Length,family = gaussian(link=identity), data=iris))
```

Call:

```
glm(formula = Sepal.Length ~ Sepal.Width + Petal.Length, family = gaussian(link = identity),  
     data = iris)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-0.96159	-0.23489	0.00077	0.21453	0.78557

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	2.24914	0.24797	9.07	7.04e-16	***
Sepal.Width	0.59552	0.06933	8.59	1.16e-14	***
Petal.Length	0.47192	0.01712	27.57	< 2e-16	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.11108)

Null deviance: 102.168 on 149 degrees of freedom
Residual deviance: 16.329 on 147 degrees of freedom
AIC: 101.03

零模型的总体方差和
模型解释的剩余方差

Number of Fisher Scoring iterations: 2

模型解释能力

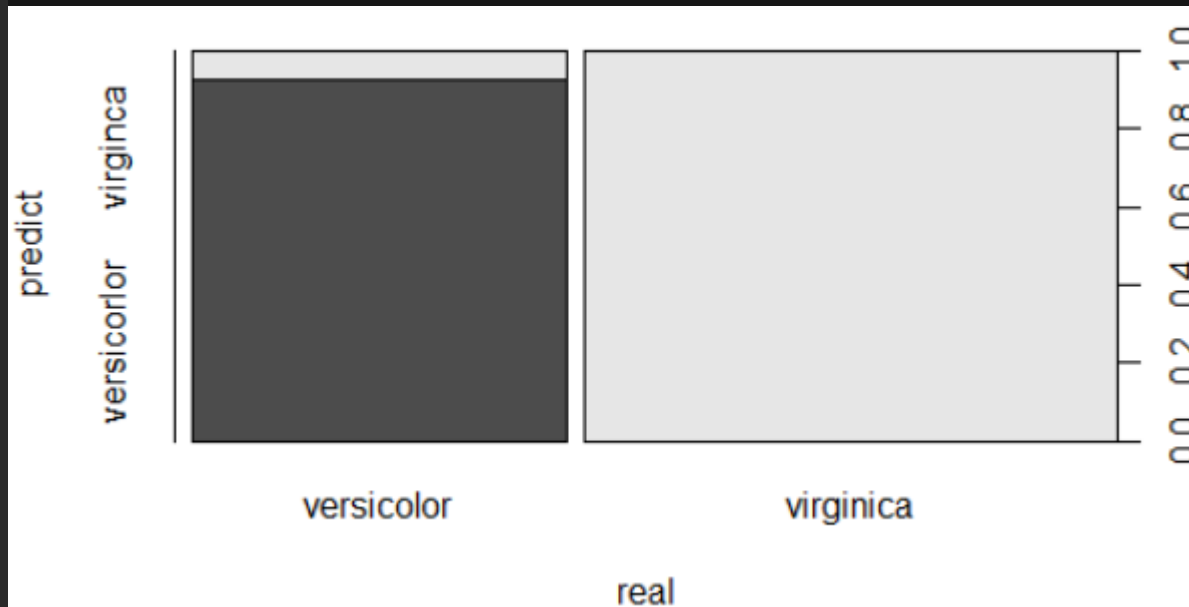
```
mod.pos<-  
  glm(Sepal.Length~Species,family=poisson,data=iris)  
mod.pos2<-  
  glm(Sepal.Length~1,family=poisson,data=iris)
```

```
anova(glm1, glm2, test="Chisq")
```

```
> anova(mod.pos, mod.pos2, test="Chisq")  
Analysis of Deviance Table  
  
Model 1: Sepal.Length ~ Species  
Model 2: Sepal.Length ~ 1  
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
1         147      6.4251  
2         149     17.3620 -2   -10.937 0.004218 **  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

二项式分布中**Logistic**回归时最重要也是最典型的回归模型，特别是对应的响应变量是二分类的

```
ir<- iris[51:150,]  
levels(ir$Species)[1] <- ""  
split <- sample(100,100*(2/3))  
ir_train <- ir[split,]  
ir_test <- ir[-split,]  
fit <- glm(Species ~.,family=binomial(link="logit"),data=ir_train)  
real <- ir_test$Species  
predict <- predict(fit,type="response",newdata=ir_test)  
res <- data.frame(real,  
                  predict =ifelse(predict>0.5,"virginica","versicolor"))  
plot(res)
```



➤ 对数-逻辑斯蒂拟合方程

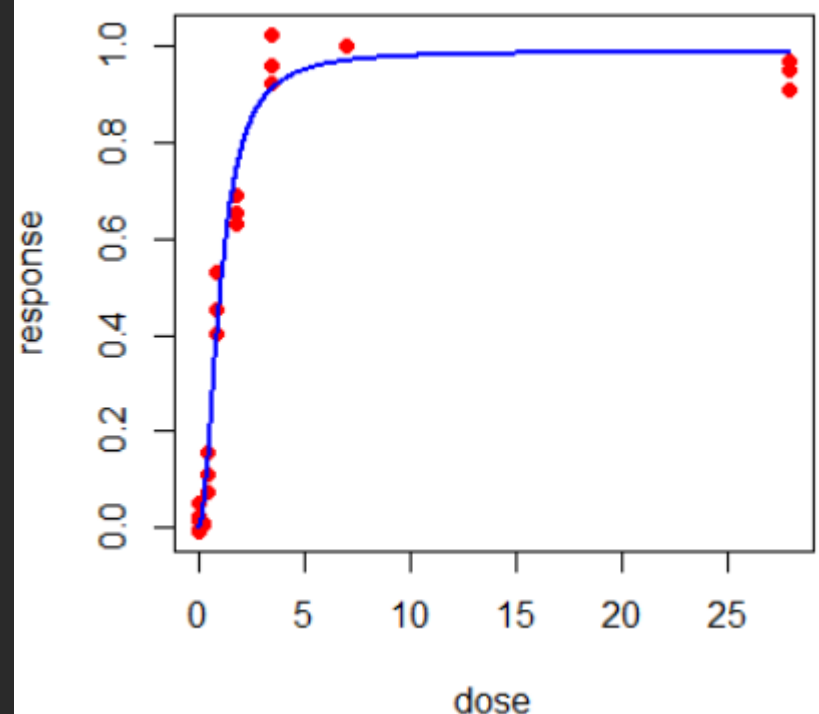
{drc}

drm() + LL.4()

```
ori.data<-read.table(text="dose,  r1, r2, r3
0.01, 0.024101146, -0.002737418, 0.01538146
0.03, 0.050177795, -0.004632554, 0.017227235
0.05, 0.026866851, -0.005264266, 0.013330599
0.11, 0.011062821, 0.011791956, 0.005742412
0.22, 0.013038325, 0.009054538, 0.004922067
0.44, 0.157250099, 0.074542009, 0.110131255
0.88, 0.401027262, 0.451379238, 0.528506973
1.75, 0.68945081, 0.62855338, 0.653814602
3.5, 0.960529435, 0.921, 1.021
7, 1, 1, 1
28, 0.906598183, 0.968866, 0.947558"
,sep=",",header=T)

library(drc)
require(reshape2)
melt.data <- melt(ori.data, id = c("dose"),
                  value.name = "response")[, -2]
model <- drm(response ~ dose, data = melt.data,
             fct = LL.4(names = c("Slope",
                                   "Lower Limit",
                                   "Upper Limit",
                                   "EC50"))))
```

```
# summary(model)
min <- range(ori.data$dose)[1]
max <- range(ori.data$dose)[2]
line.data<-data.frame(d.predict=
                      seq(min, max,
                          length.out = 1000))
line.data$p.predict<-predict(model,
                             newdata = line.data)
plot(response ~ dose,data=melt.data,
     col=2,pch=20,cex=1.5)
lines(line.data$d.predict, line.data$p.predict,
      col=4,lwd=2)
```



◆非线性回归(nonlinear regression models)

nls()

Description

Determine the nonlinear (weighted) least-squares estimates of the parameters of a nonlinear model.

Usage

```
nls(formula, data, start, control, algorithm,  
     trace, subset, weights, na.action, model,  
     lower, upper, ...)
```

包括分断回归、多项式回归、指数函数回归等。

指数函数: $y \sim a * \exp(b * x)$

```
mod.nls<- nls(y ~ a * exp(b*x),  
              data=data,  
              start=list(a=a, b=b))
```

```
summary(mod.nls)
```

- PCA、NMDS、RDA、heatmap

简单实现的例子见之前上传群里的PFD文件。

学习教材：

《Numerical Ecology with R》

《数量生态学》

https://4va.github.io/biodatasci/r-stats.html#logistic_regression

Descriptive statistics

Missing data

EDA

Exercise set 1

Continuous variables

T-tests

Wilcoxon test

Linear models

ANOVA

Linear regression

Multiple regression

Exercise set 2

Discrete variables

Contingency tables

Logistic regression

Essential Statistics with R

This workshop will provide hands-on instruction and exercises covering basic statistical analysis in R. This will cover descriptive statistics, *t*-tests, linear models, chi-square, clustering, dimensionality reduction, and resampling strategies. We will also cover methods for “tidying” model results for downstream visualization and summarization.

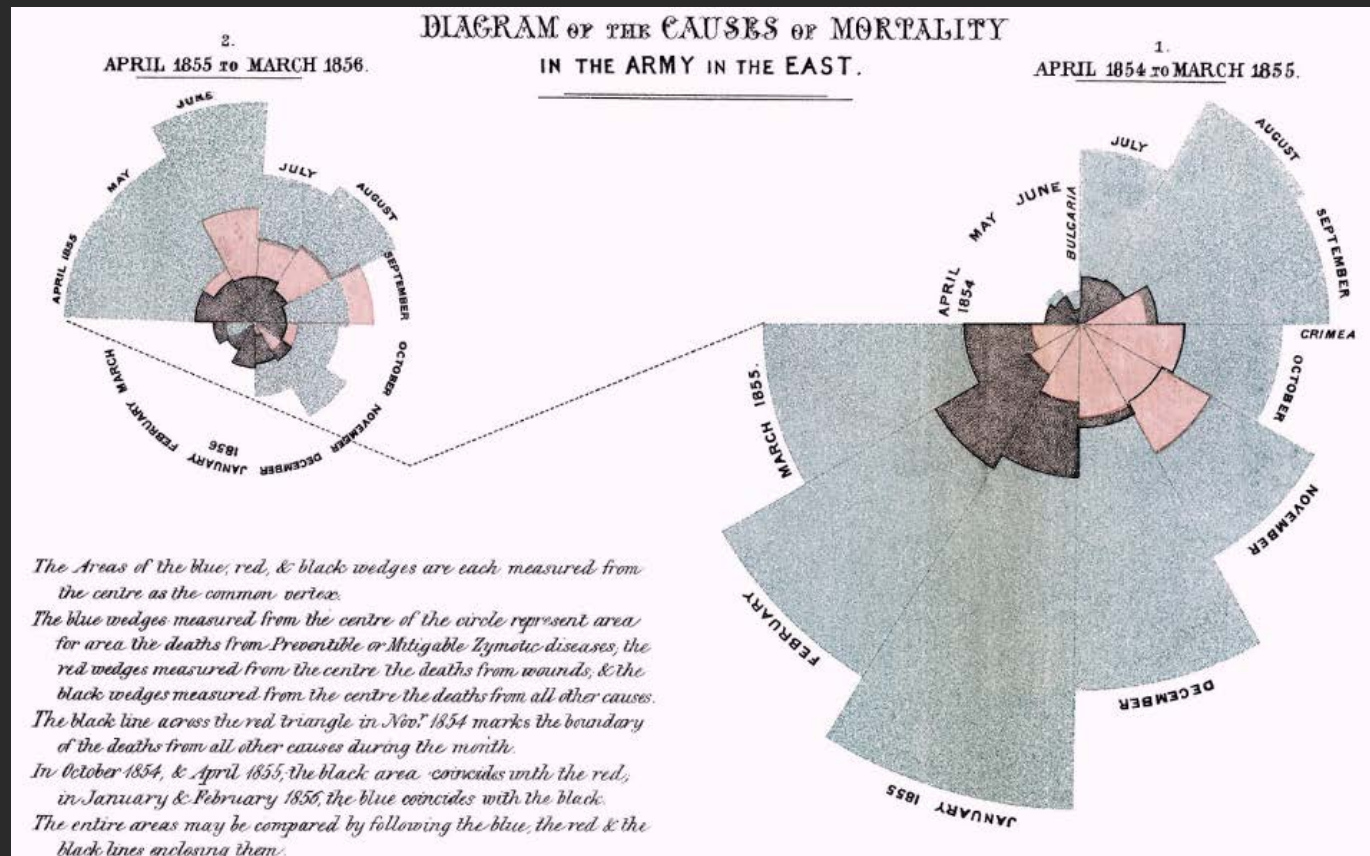
Prerequisites: **Familiarity with R is required** (including working with [data frames](#), installing/using packages, importing data, and saving results); familiarity with [dplyr](#) and [ggplot2](#) packages is highly recommended.

You must [complete the basic R setup here](#) *prior to class*. This includes installing R, RStudio, and the required packages. Please [contact one of the instructors](#) *prior to class* if you are having difficulty with any of the setup. Please bring your laptop and charger cable to class.

Handouts: Download and print out these handouts and bring them to class:

Part 4

图形基础与绘制



高等函数：在图形设备上新建一幅图形。

- 直方图
- 柱状图
- 饼形图
- 散点图
- 箱线图
- 地图
- ...

低级函数：在现有图形上添加图形元素。

- 添加点、线等

◆ 绘图配置

par()

adj (adjust)	字符位置调整，取值0-1，默认为0.5. 函数text, mtext, title常常使用
ask	默认FALSE，控制Rconsole（控制台）图形输出，一次输出一张
bg (background)	绘图背景，默认为“transparent”（透明），可设成“white”等其他色
bty (box type)	绘图边界，默认“o”（边界闭合），“l”，“7”，“c”，“u”，或“]”，“n”抑制边界

cex	绘图字体放大，默认为1
col	颜色设定，针对添加的文本
family	设定字体类型， "serif", "sans" and "mono", 默认为"arial", "GB1"中文
fg(forebackground)	前景色设定，默认为"black"
fin	默认为6.999999 6.999999，分别是图形的长度和高度，
font	字体类型， 1=普通字体， 2=粗体， 3=斜体， 4=粗斜体， 5=符号字体
las	默认为0，刻度标记一直与坐标轴平行， 1一直水平（常用）
lty(line type)	线型，默认为1（实线）， 2=虚线, 3=点线, 4=破折线, 5=长破折线, 6=twodash
lwd(line width)	正整数，默认为1
mai	默认为5,4,4,2+0.1 (下、左、上、右)

```
layout(matrix(1:2,ncol=2))
```

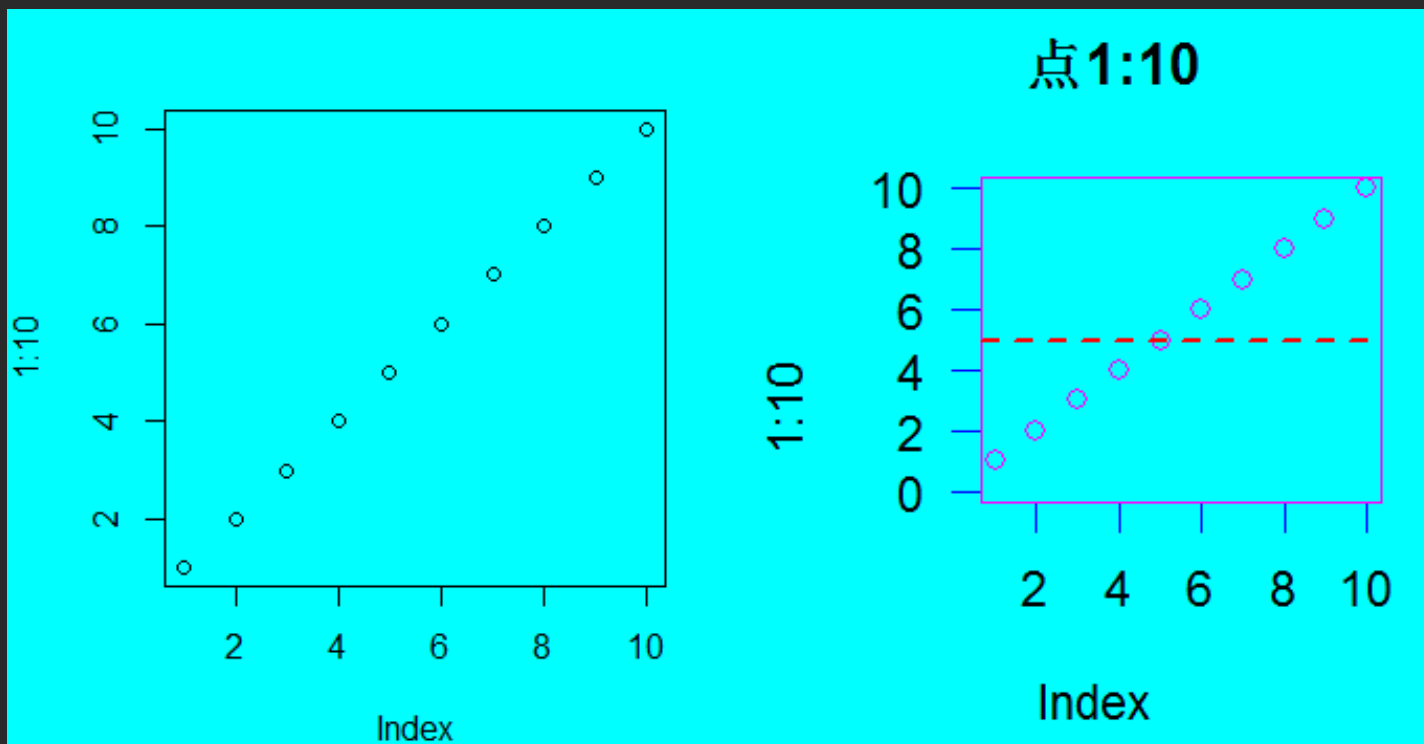
```
par(bg=5)
```

```
plot(1:10)
```

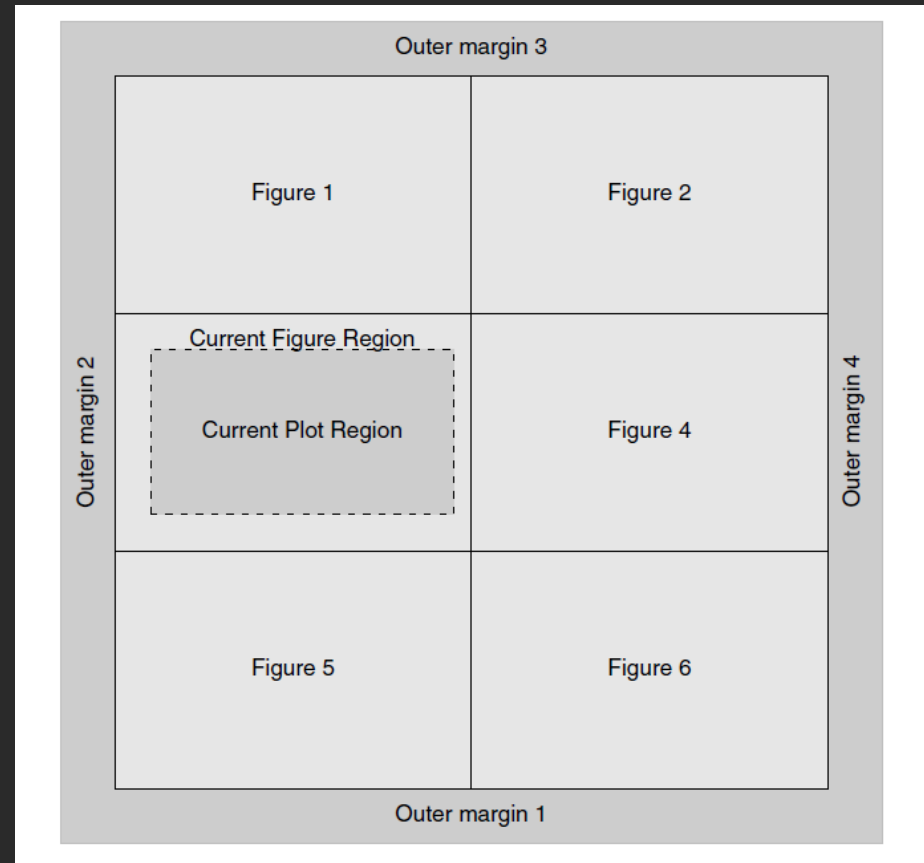
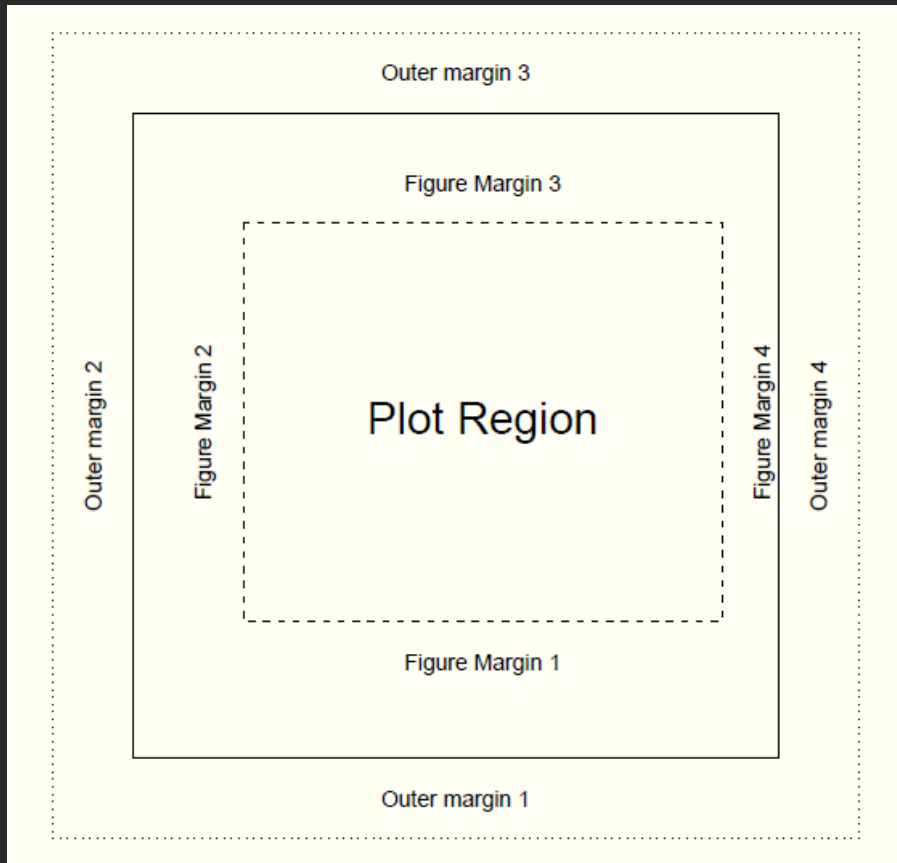
```
par(adj=0.2, fg=4, cex=1.4, col=6, las=1, lty=2)
```

```
plot(1:10, main="点1:10", ylim=c(0,10))
```

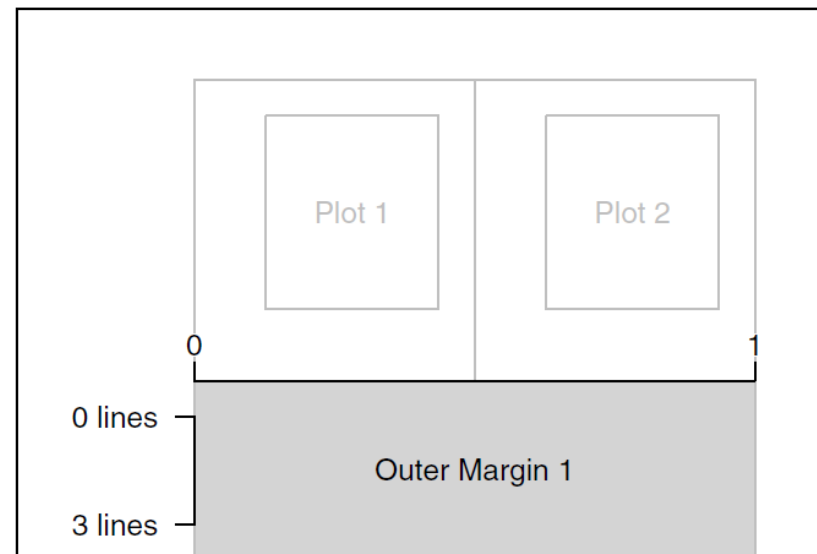
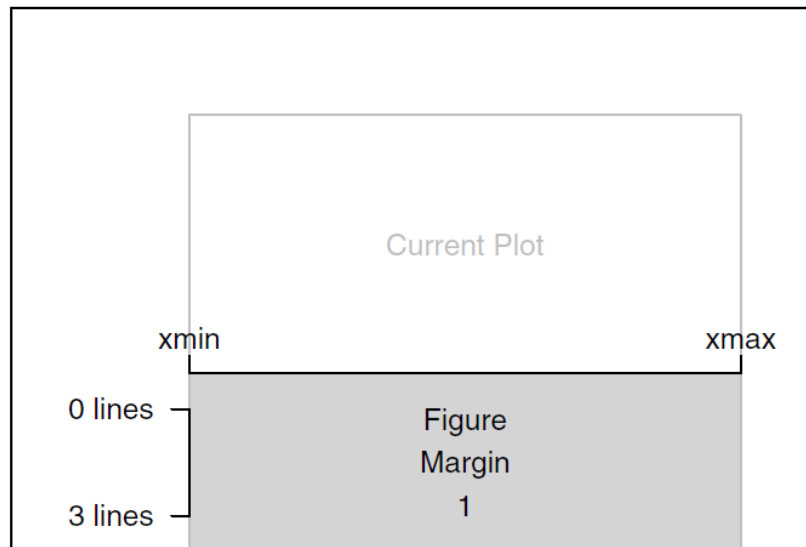
```
abline(h=5,col=2,lwd=2)
```



• 边margins



• 坐标系



mgp

`mgp`

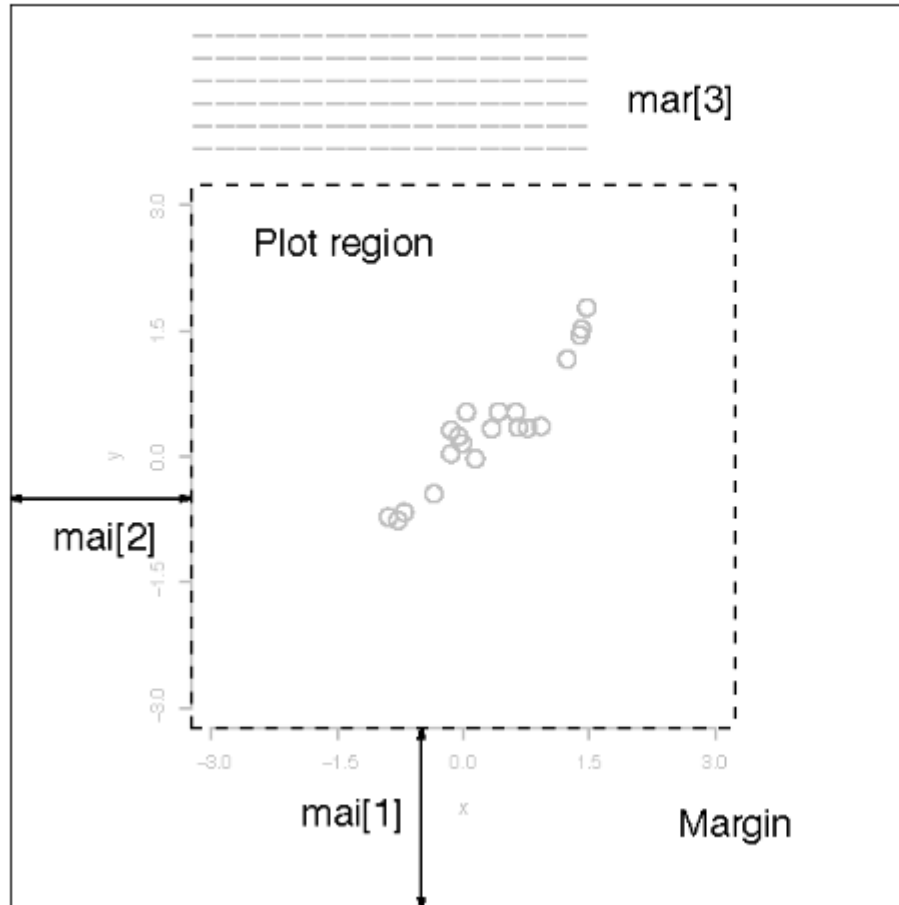
The margin line (in `mex` units) for the axis title, axis labels and axis line.

默认 `c(3,1,0)`

mai

```
mai
```

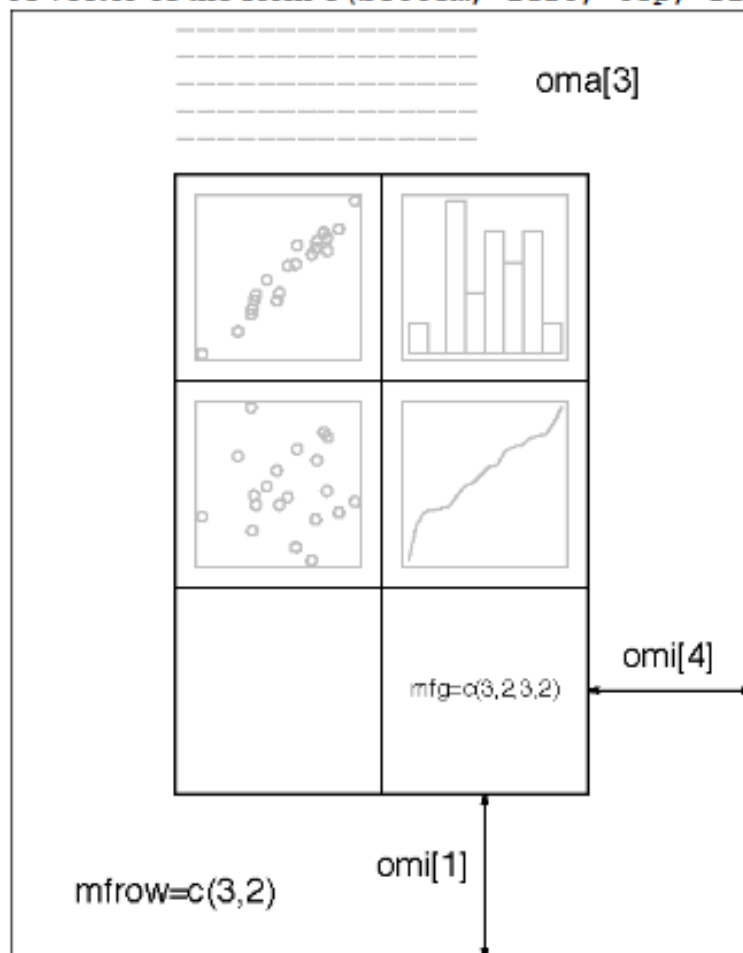
A numerical vector of the form `c(bottom, left, top, right)` which gives the margin size specified in inches.



oma

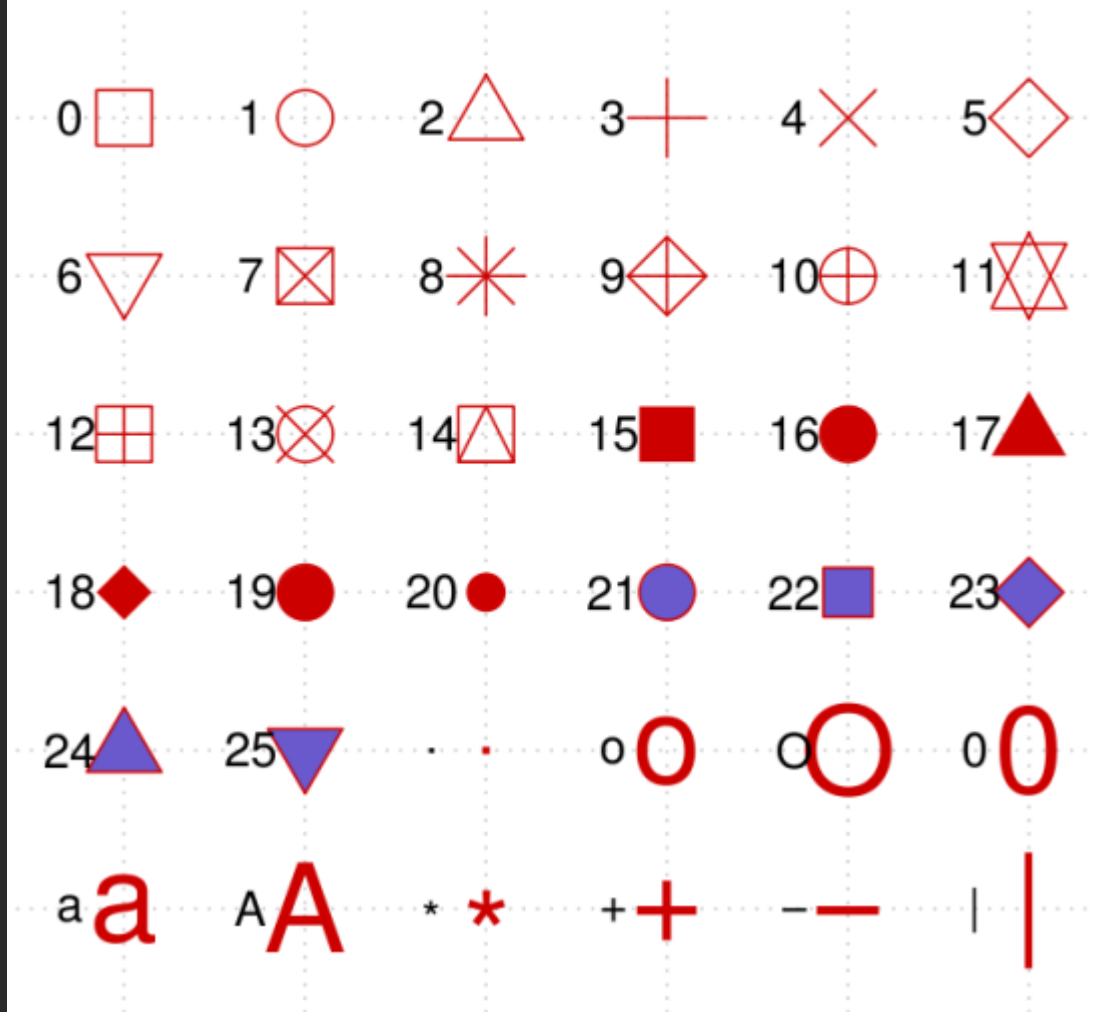
oma

A vector of the form `c(bottom, left, top, right)` giving the size of the outer margins in lines of text.



pch

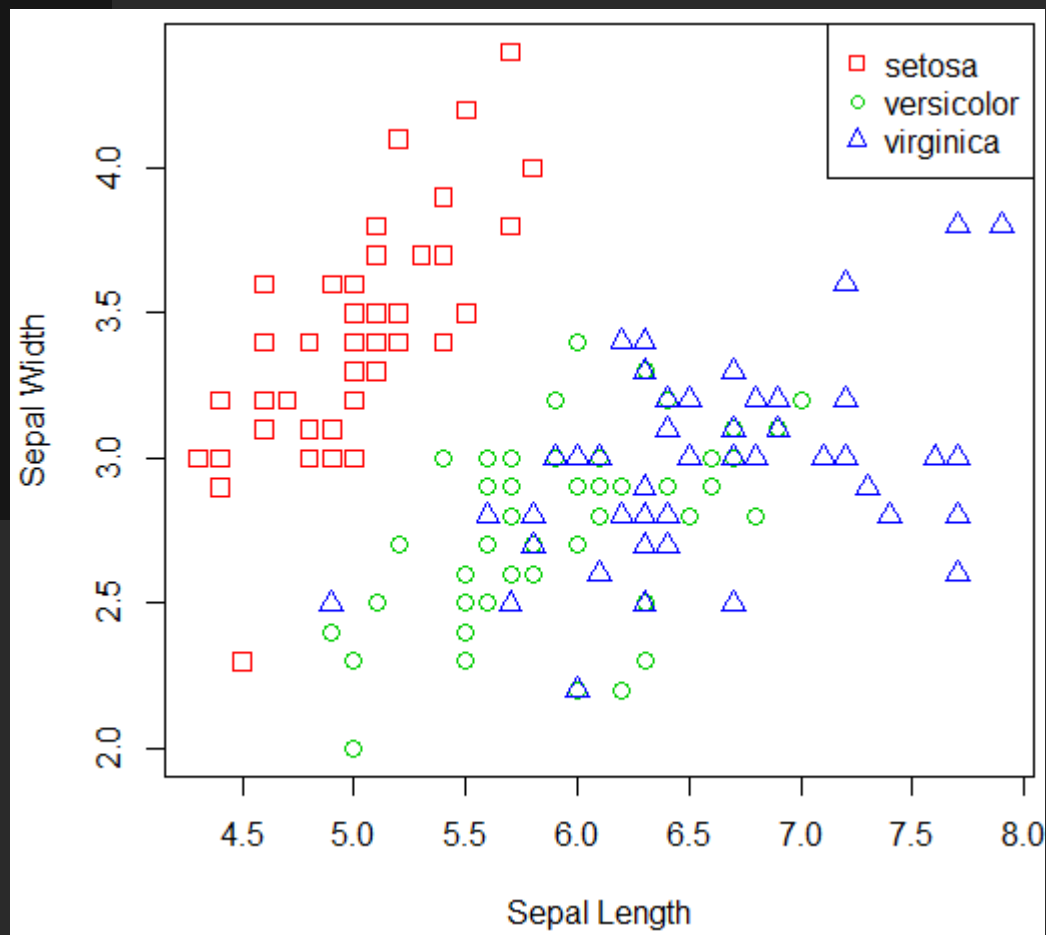
Plot symbols in R;
col = " red3 ", bg = " slateblue3 "



• 散点图

```
plot(iris$Sepal.Length, iris$Sepal.Width,  
     col=(2:4)[iris$Species],  
     pch=(0:2)[iris$Species],  
     xlab="Sepal Length",  
     ylab="Sepal Width",  
     cex=1.2)  
legend("topright",  
       legend=levels(iris$Species),  
       pch=0:2,  
       col=2:4)
```

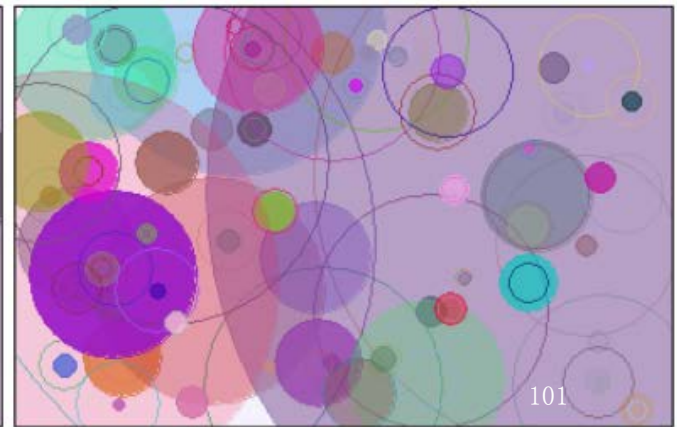
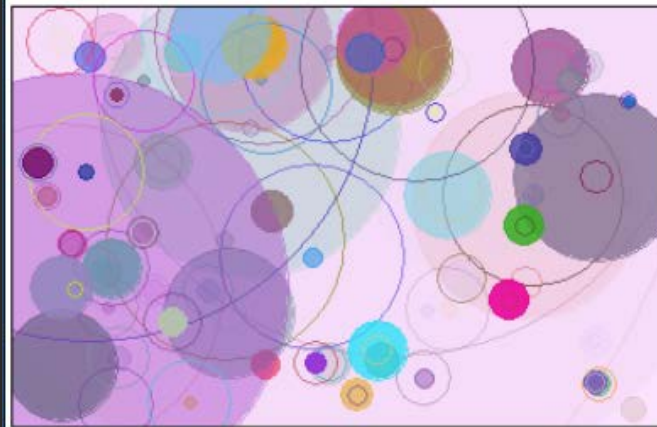
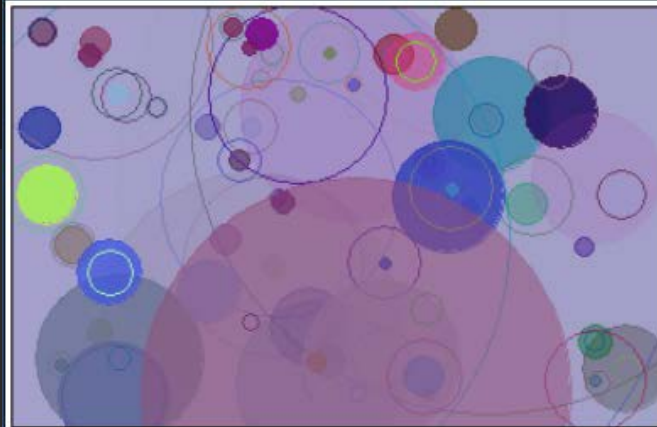
points()



```

par(mar = c(0.2, 0.2, 0.2, 0.2), mfrow = c(2, 2))
for (n in c(63, 60, 76, 74)) {
  set.seed(521) ###
  plot.new()
  size = c(replicate(n, 1/rbeta(2, 1.5, 4)))
  center = t(replicate(n, runif(2)))
  center = center[rep(1:n, each = 2), ]
  color = apply(replicate(2 * n, sample(c(0:9,
LETTERS[1:6])), 8, TRUE)), 2, function(x)
  sprintf("#%s",paste(x, collapse = "")))
  points(center, cex = size, pch = rep(20:21, n),
  col = color)
  box()
}

```



• 画线

```
plot(x, y, type="l", ...)
```

```
lines(x, y = NULL, type = "l", ...)
```

```
abline(a = NULL, b = NULL, h = NULL, v = NULL, reg = NULL, coef =  
NULL, untf = FALSE, ...)
```

```
segments(x0, y0, x1 = x0, y1 = y0, col = par("fg"), lty =  
par("lty"), lwd = par("lwd"), ...)
```

```
arrows(x0, y0, x1 = x0, y1 = y0, length = 0.25, angle = 30, code =  
2, col = par("fg"), lty = par("lty"), lwd = par("lwd"), ...)
```

```
grid(nx = NULL, ny = nx, col = "lightgray", lty = "dotted", lwd =  
par("lwd"), equilogs = TRUE)
```

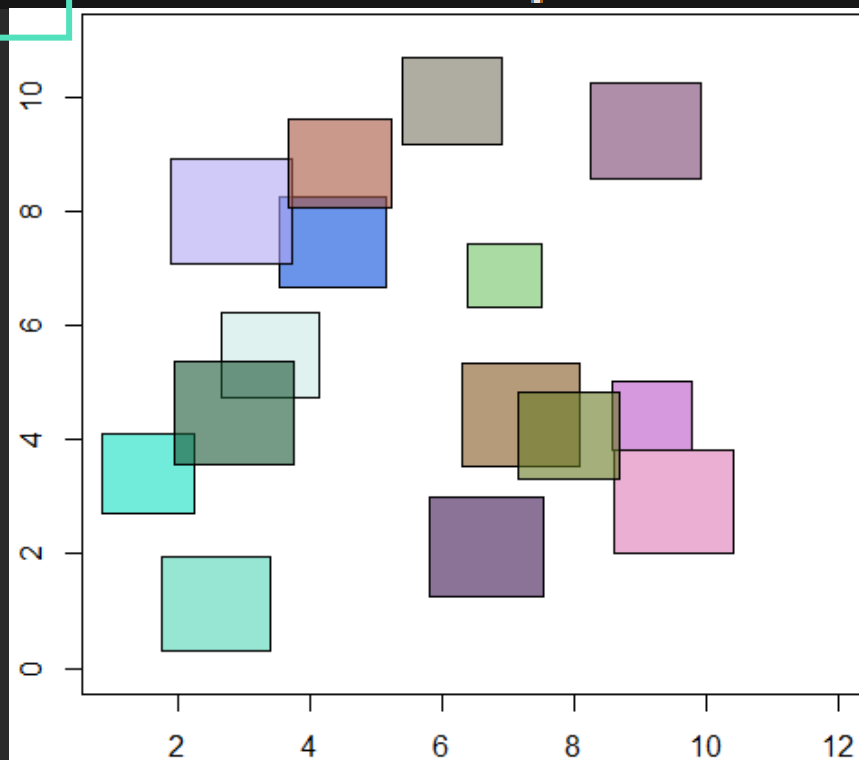
```
curve(expr, from = NULL, to = NULL, n = 101, add = FALSE, type =  
"l", ylab = NULL, log = NULL, xlim = NULL, ...)
```

- 画多边形

rect()

polygon()

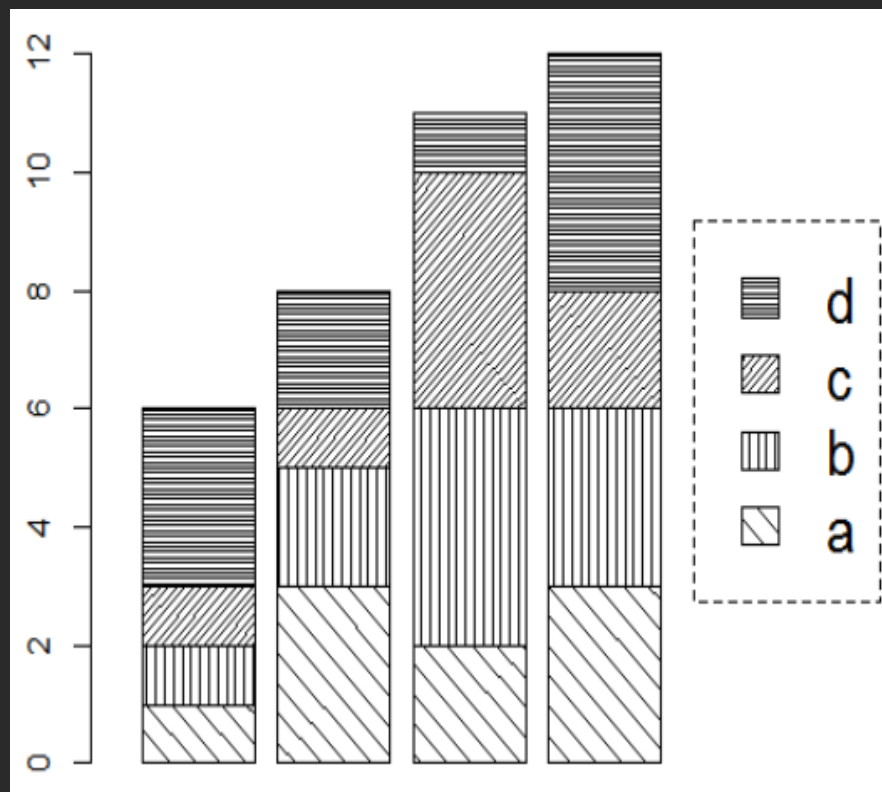
```
set.seed(123)
x<-runif(n=15,min=1,max=10)
y<-runif(n=15,min=1,max=10)
side<-runif(15,min=.5,max=1)
samps<-replicate(4,runif(15,min=0,max=1))
cols<-rgb(samps,alpha=.6)
plot(0:11,type="n",ann=F)
rect(col=cols,x-side,y-side,x+side,y+side)
```



• 柱状图

barplot()

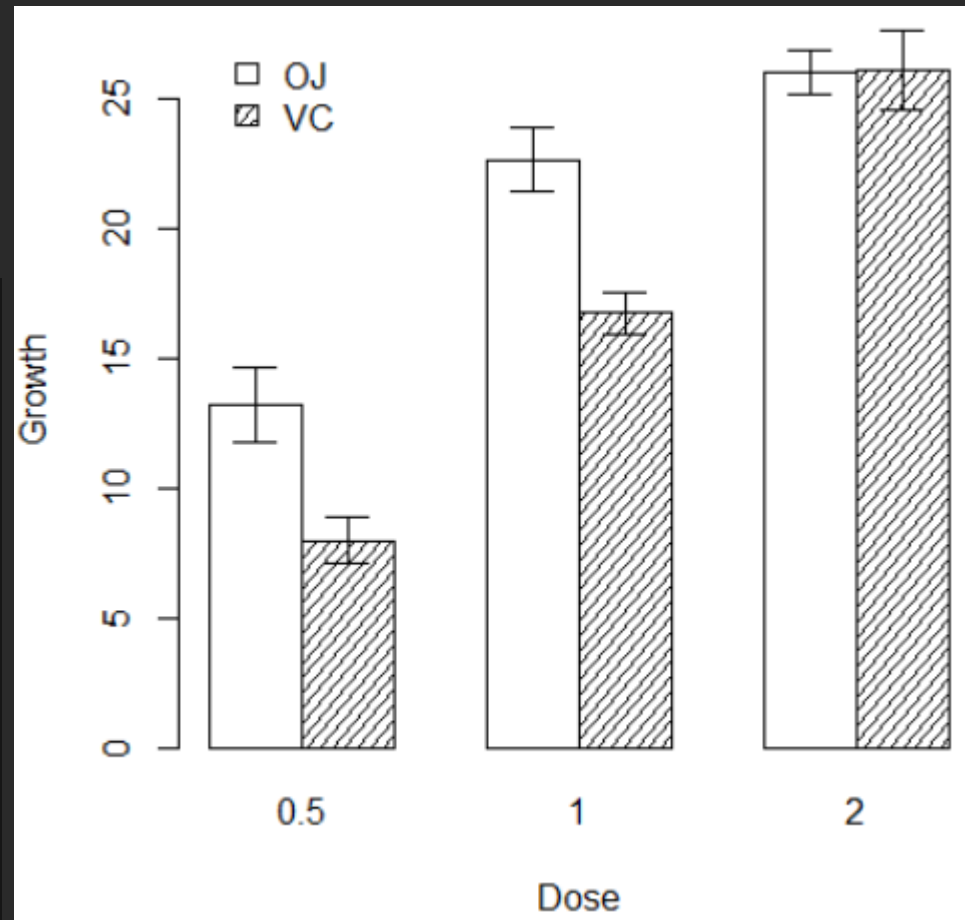
```
barplot(matrix(sample(1:4, 16, replace=T),
               ncol=4),
        angle=c(135,90,45,0),
        density=1:4*10,
        col=1,
        legend.text=letters[1:4],
        xlim=c(0,6.5),
        args.legend=list(x="right",
                        box.lty=2,
                        cex=1.8)
        )
```



- 柱形图+误差线

bargraph.CI()

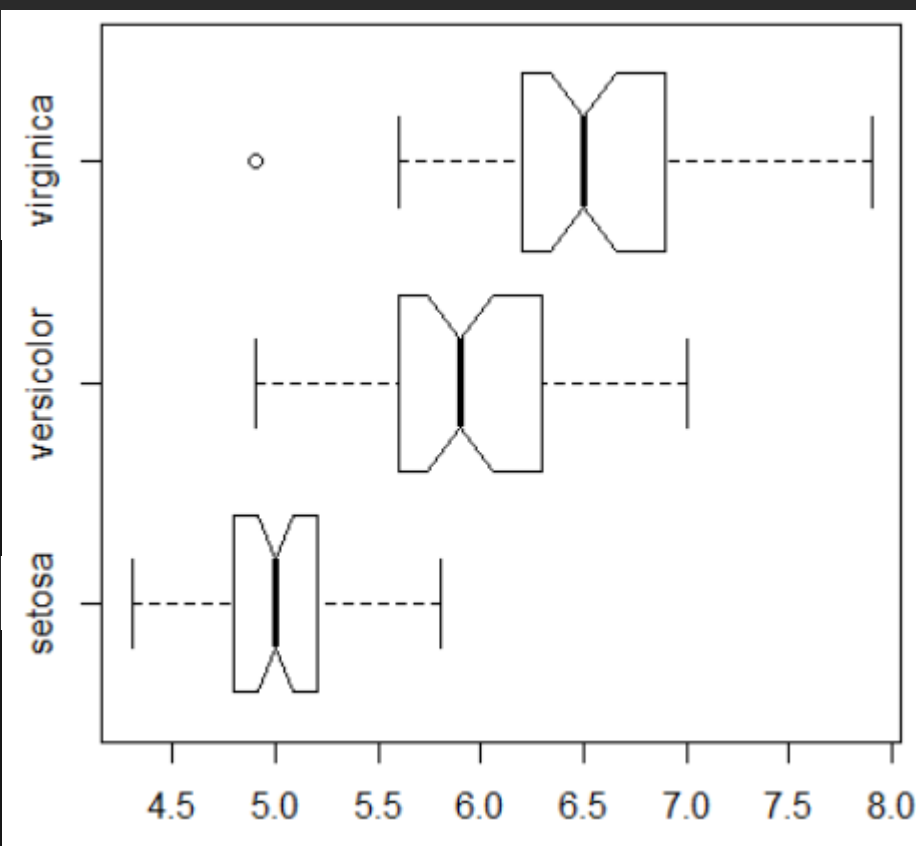
```
data(ToothGrowth)
library(sciplot)
bargraph.CI(x.factor=dose,
  response=len,
  group = supp,
  data = ToothGrowth,
  density = c(0,20),
  angle = 45,
  col="black",
  legend = TRUE,
  x.legend=1,
  xlab = "Dose",
  ylab = "Growth")
```



- 箱线图
`boxplot()`

```
boxplot(Sepal.Length~Species,  
        notch=T,  
        horizontal=T,  
        data=iris)
```

`notch` if `notch` is `TRUE`, a notch is drawn in each side of the boxes. If the notches of two plots do not overlap this is 'strong evidence' that the two medians differ (Chambers *et al*, 1983, p. 62). See `boxplot.stats` for the calculations used.

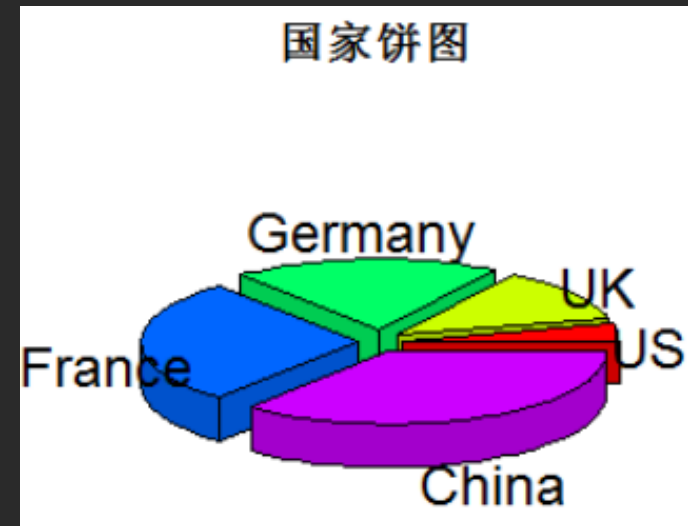
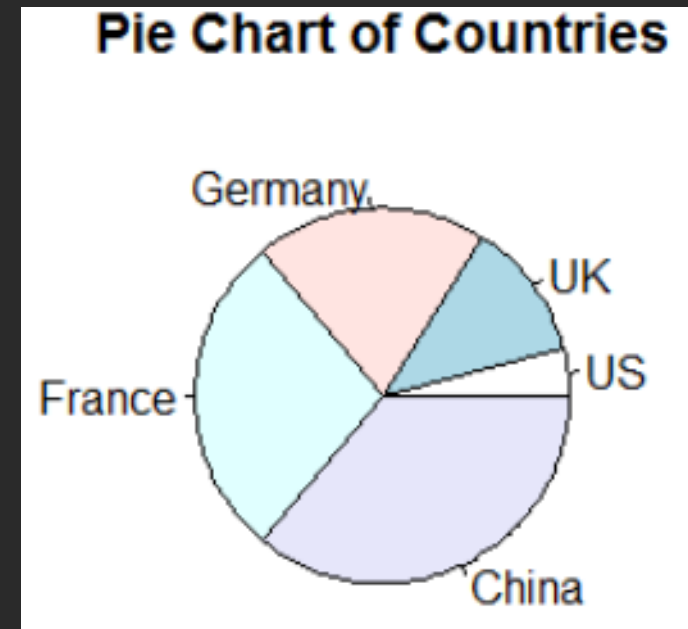


- 饼图

pie()

pie3D()

```
slices <- seq(1,9,2)
lbls <- c("US", "UK",
          "Germany", "France",
          "China")
pie(slices,
    labels = lbls,
    main="Pie Chart of Countries")
library(plotrix)
pie3D(slices,
      labels=lbls,
      explode=0.1,
      main="国家饼图 ",
      col=rainbow(5))
```

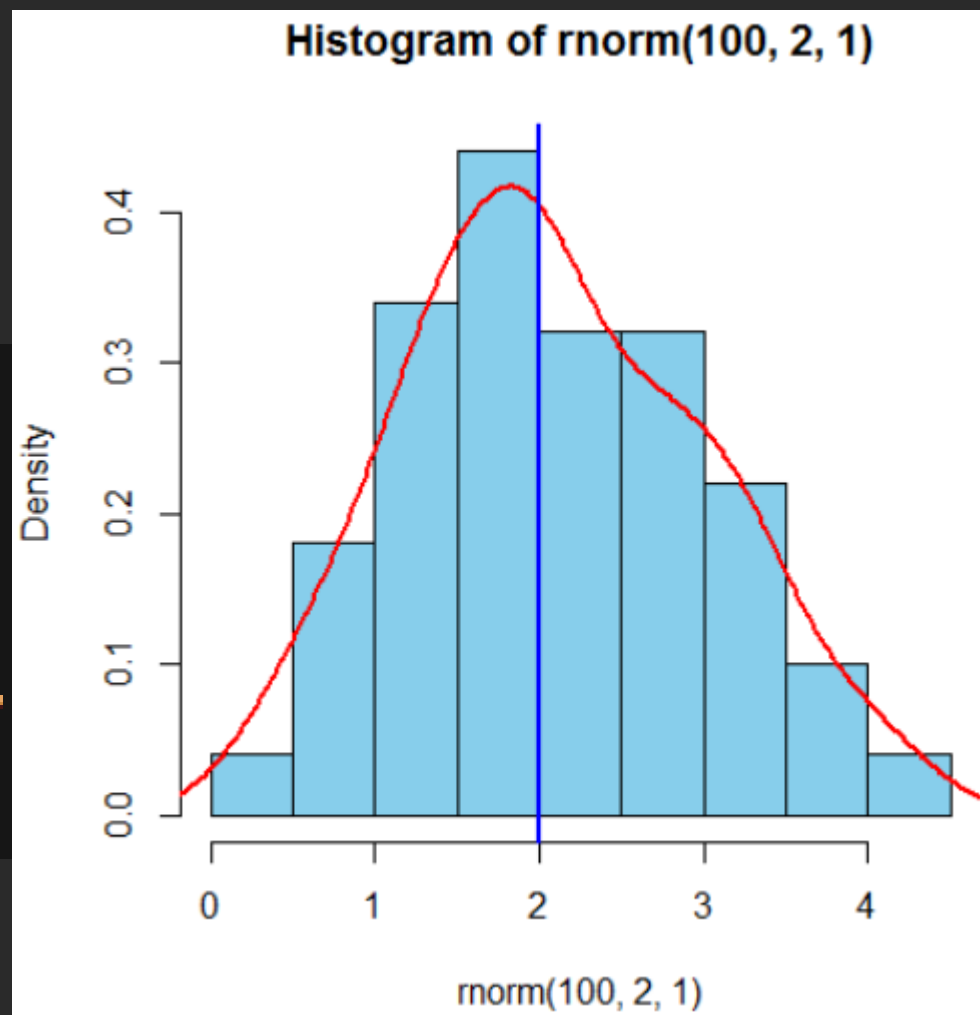


- 直方图和一元密度估计

hist()

density()

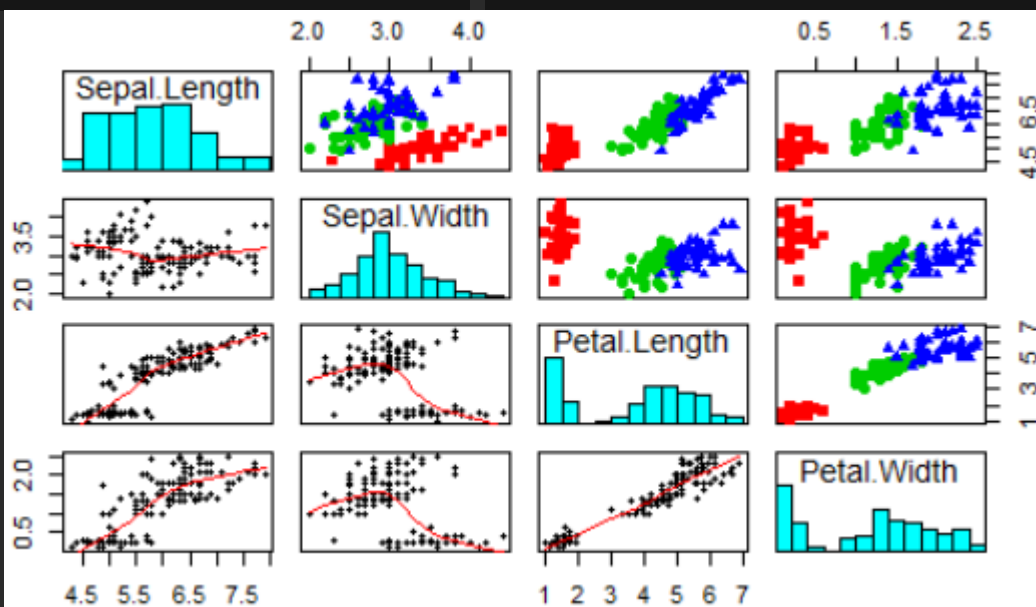
```
set.seed(4)
hist(rnorm(100,2,1),
     freq=FALSE,
     col="skyblue")
set.seed(4)
lines(density(rnorm(100,2,1)),
      col=2,lwd=2)
abline(v=2,col=4,lwd=2)
```



- 散点图矩阵
`pairs()`

```
panel.hist <- function(x, ...)
{
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(usr[1:2], 0, 1.5) )
  h <- hist(x, plot = FALSE)
  breaks <- h$breaks; nB <- length(breaks)
  y <- h$counts; y <- y/max(y)
  rect(breaks[-nB], 0, breaks[-1], y, col=5, ...)
}
```

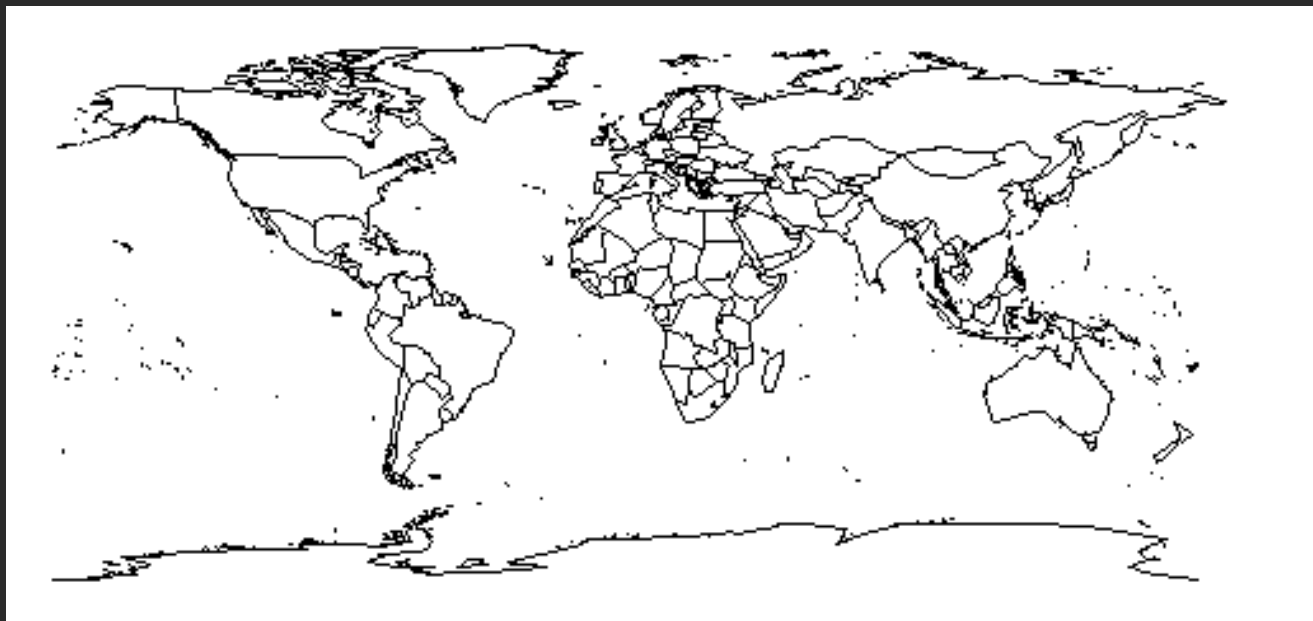
```
idx=as.integer(iris$Species)
pairs(iris[1:4],
      upper.panel=function(x,y,...) {
        points(x,y,
               pch=c(15:17)[idx],
               col = idx+1)
      },
      pch = 20,
      oma = c(2, 2, 2, 2),
      lower.panel = panel.smooth,
      diag.panel = panel.hist
)
```



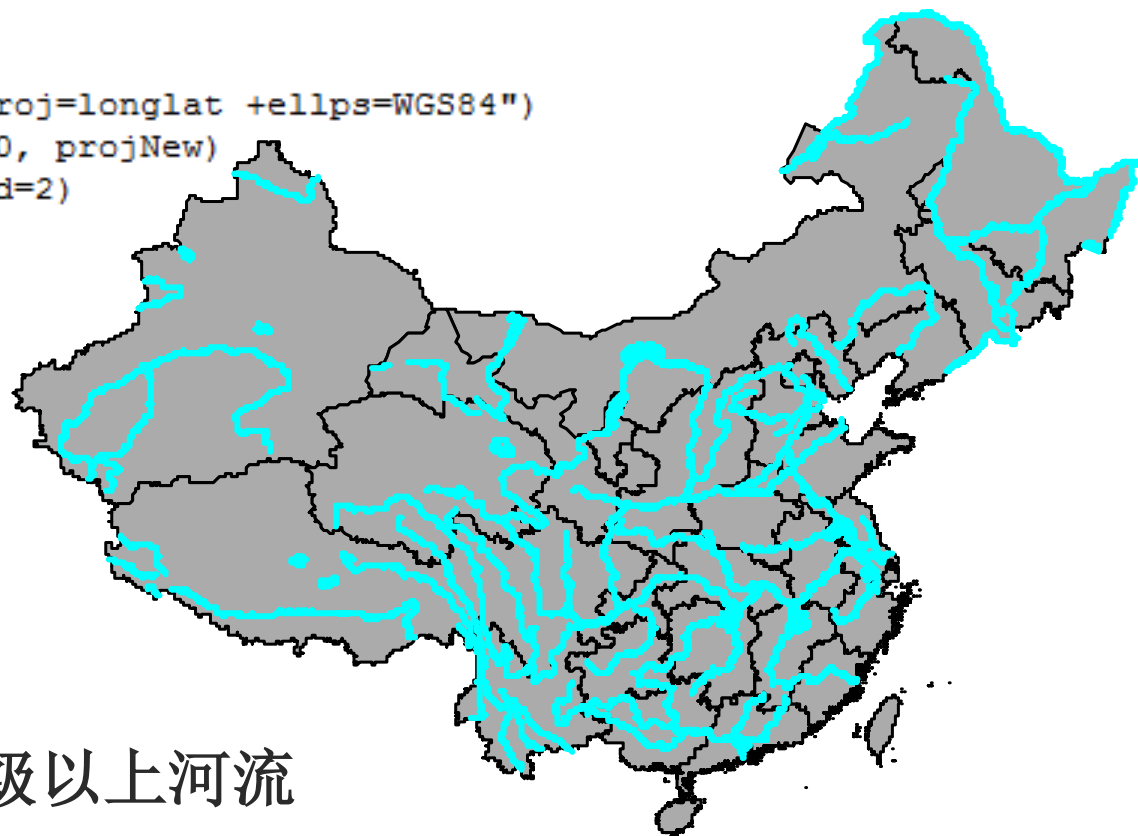
- 地图

map()

```
> map("world")
```



```
readShapeLines(file.choose())->xxx # "国界与省界_arc.shp"  
readShapeLines(file.choose())->rvrs0 # "三级以上河流_arc.shp"  
  
projNew <- CRS("+proj=merc +lat_0=45n +lon_0=100e")  
proj4string(xxx) <- CRS("+proj=longlat +ellps=WGS84")  
xProj <- spTransform(xxx, projNew)  
plot(xProj,col="darkgrey")  
  
proj4string(rvrs0) <- CRS("+proj=longlat +ellps=WGS84")  
rvrs0Proj <- spTransform(rvrs0, projNew)  
plot(rvrs0Proj,add=T,col=5,lwd=2)
```



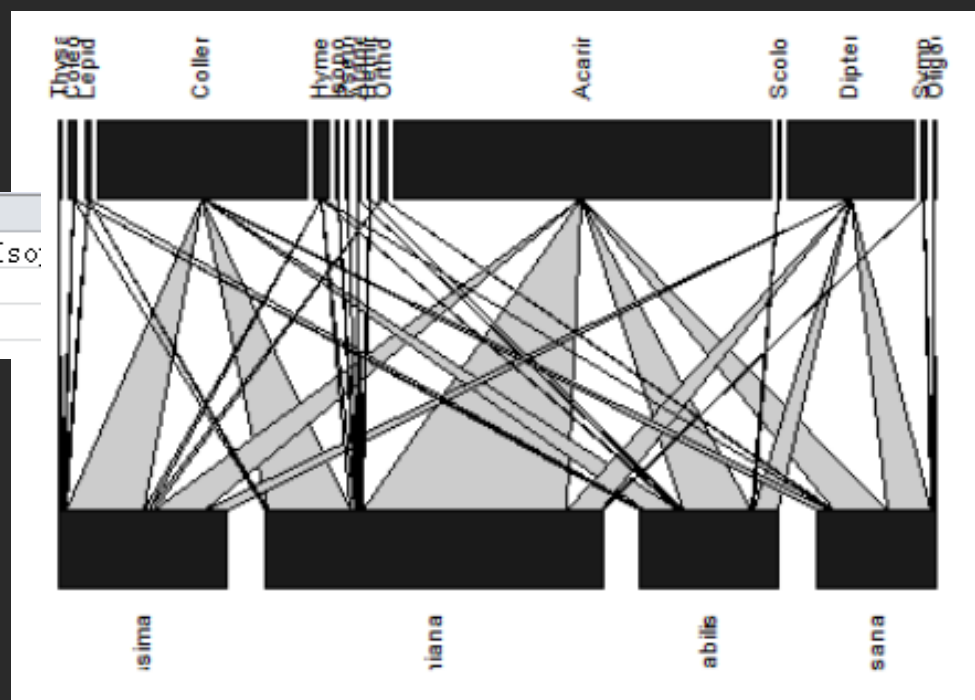
国界与省界 + 三级以上河流

- 二分网图

plotweb() {bipartite}

```
below<-read.csv('below.csv');rownames(below)<-below[,1];below<-below[, -1]
plotweb(below,arrow="up.center",text.rot=90)
```

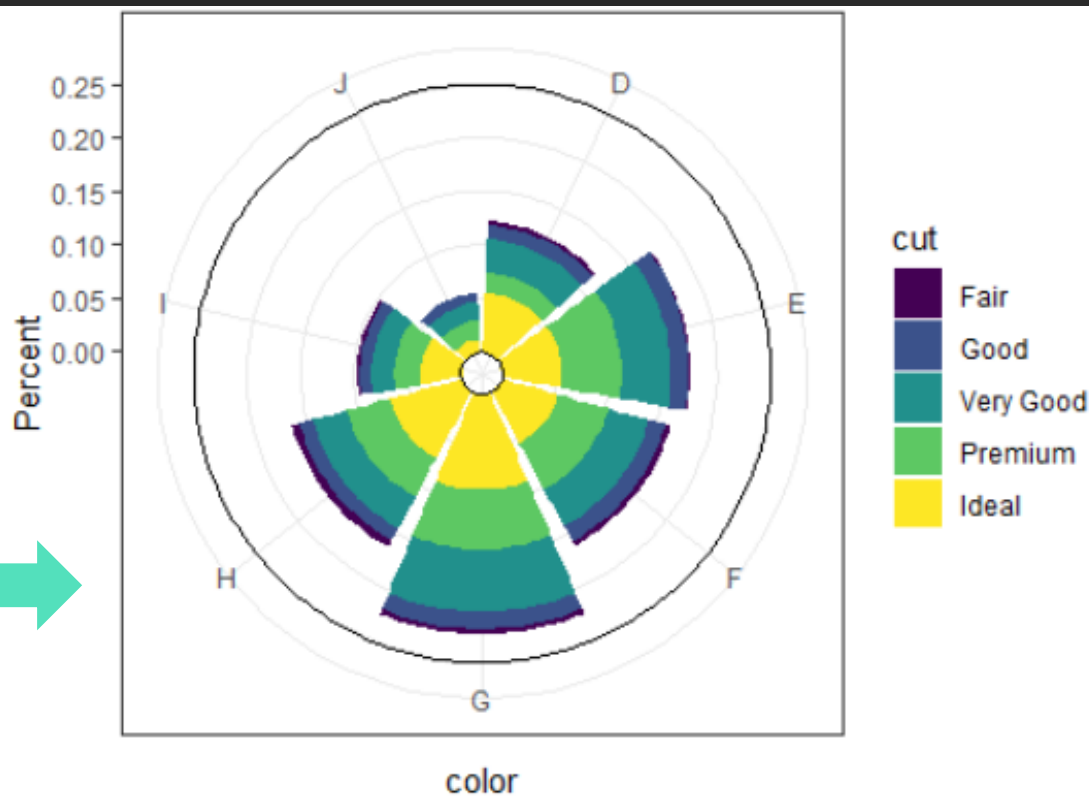
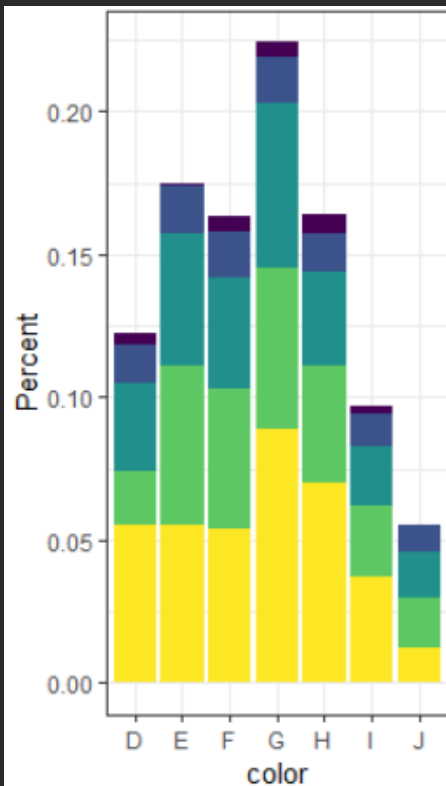
A	B	C	D	E
	Acarina	Collembola	Diptera	Hymenoptera
L. ormosa	37	10	33	1
Q. variab	49	24	18	2
Q. sentis	27	56	15	4



- 玫瑰花图

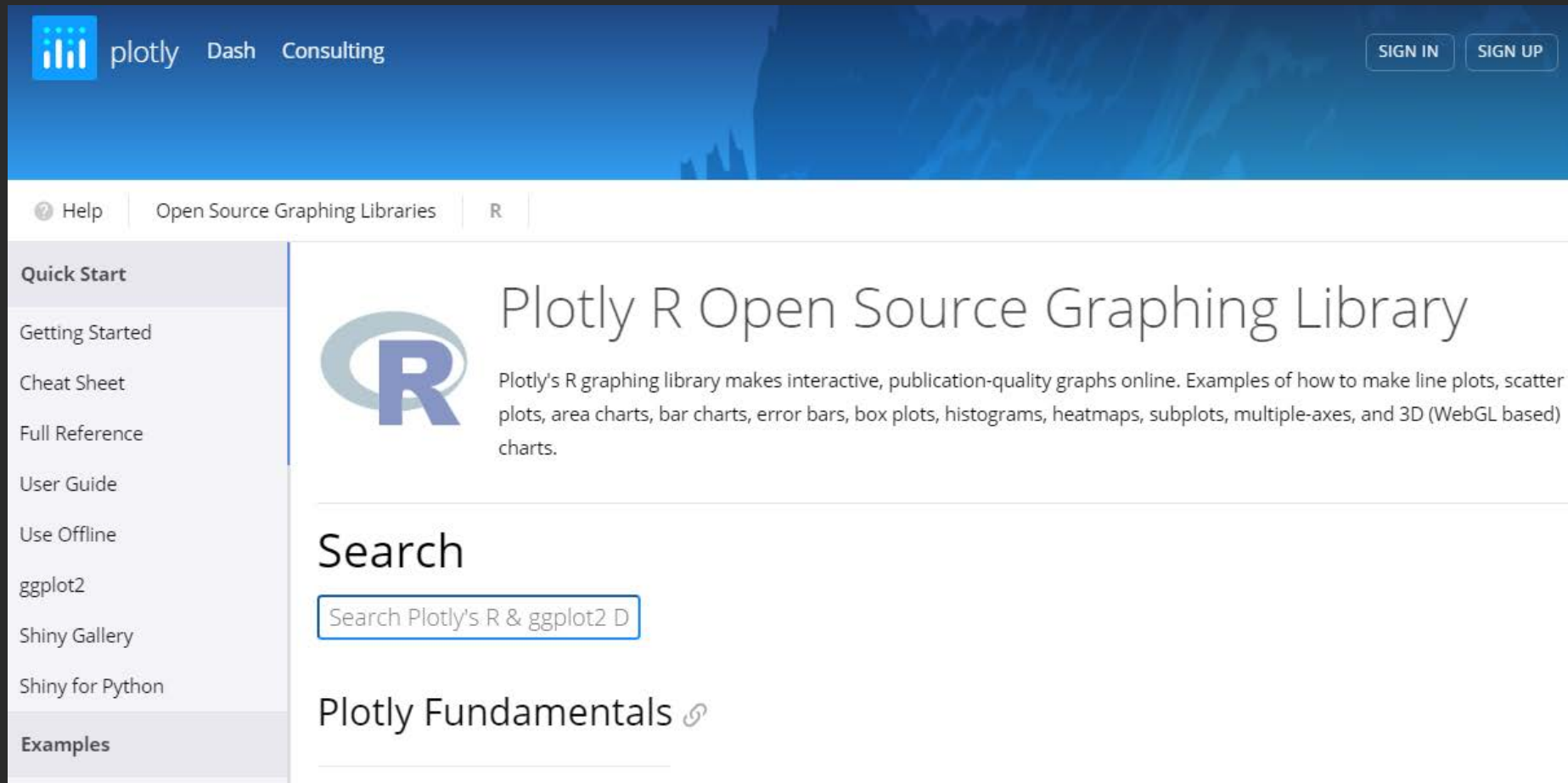
ggplot() +
geom_bar() +
coord_polar()

```
library(ggplot2)
subdmd<-diamonds[sample(nrow(diamonds),1000),]
ggplot(subdmd,aes(x=color))+
  geom_bar(aes(y=..count../sum(..count..),fill=cut))+
  coord_polar()+
  theme_bw()+
  ylim(-.02,.25)+
  ylab("Percent")+
  geom_hline(yintercept=0)+geom_hline(yintercept=.25)
```



- 网页形式的可交互图表: **Plotly**

<https://plot.ly/r/>



The screenshot shows the Plotly R Open Source Graphing Library website. The header features the Plotly logo, navigation links for 'Dash' and 'Consulting', and 'SIGN IN' and 'SIGN UP' buttons. The main navigation bar includes 'Help', 'Open Source Graphing Libraries', and 'R'. A left sidebar contains a 'Quick Start' section with links to 'Getting Started', 'Cheat Sheet', 'Full Reference', 'User Guide', 'Use Offline', 'ggplot2', 'Shiny Gallery', and 'Shiny for Python', followed by an 'Examples' section. The main content area is titled 'Plotly R Open Source Graphing Library' and includes the R logo, a description of the library's capabilities, a search bar, and a link to 'Plotly Fundamentals'.

plotly Dash Consulting

SIGN IN SIGN UP

Help Open Source Graphing Libraries R

Quick Start

- Getting Started
- Cheat Sheet
- Full Reference
- User Guide
- Use Offline
- ggplot2
- Shiny Gallery
- Shiny for Python

Examples

Plotly R Open Source Graphing Library

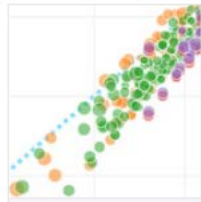
Plotly's R graphing library makes interactive, publication-quality graphs online. Examples of how to make line plots, scatter plots, area charts, bar charts, error bars, box plots, histograms, heatmaps, subplots, multiple-axes, and 3D (WebGL based) charts.

Search

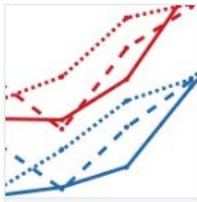
Search Plotly's R & ggplot2 D

Plotly Fundamentals [↗](#)

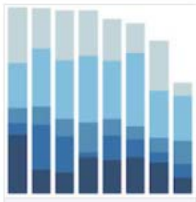
Basic Charts [↗](#)



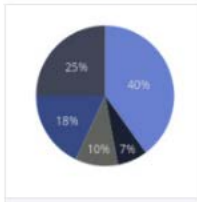
Scatter and
Line Plots



Line Plots



Bar Charts



Pie Charts

Maps [↗](#)



Choropleth
Maps



Bubble Maps



Lines on
Maps

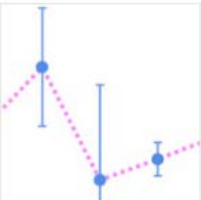


Scattermapbox

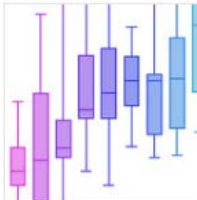


[More Ma](#)

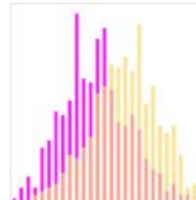
Statistical Charts [↗](#)



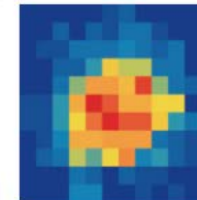
Error Bars



Box Plots

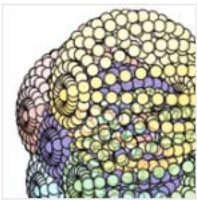


Histograms

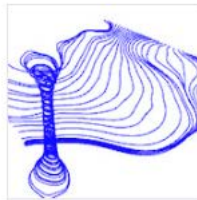


2D
Histograms

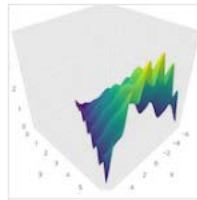
3D Charts [↗](#)



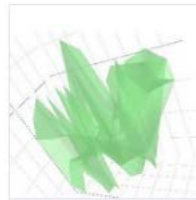
3D Scatter
Plots



3D Line Plots



3D Surface
Plots

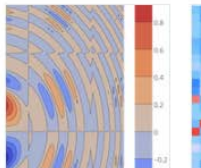


3D Mesh
Plots

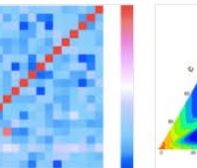


More 3D
Charts

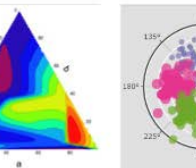
Scientific Charts [↗](#)



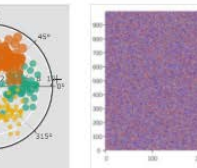
Contour
Plots



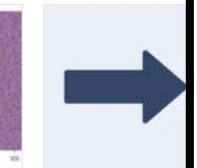
Heatmaps



Ternary Plots



Polar Charts

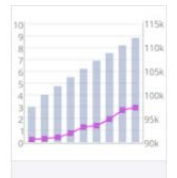


WebGL
Heatmaps

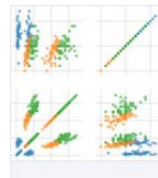


More
Scientific
Charts

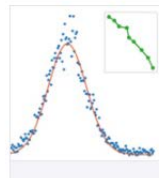
Multiple Axes, Subplots, and Insets [↗](#)



Multiple Axes



Subplots



Inset Plots



Mixed
Subplots



More
Subplots

其它可参考的R快速学习网络资源推荐

<https://www.statmethods.net/index.html>



R Tutorial R Interface Data Input Data Management Statistics
Advanced Statistics Graphs Advanced Graphs

SITE CONTENTS

Learning R

R Tutorial

R Interface

Data Input

Data Management

Statistics

Advanced Statistics

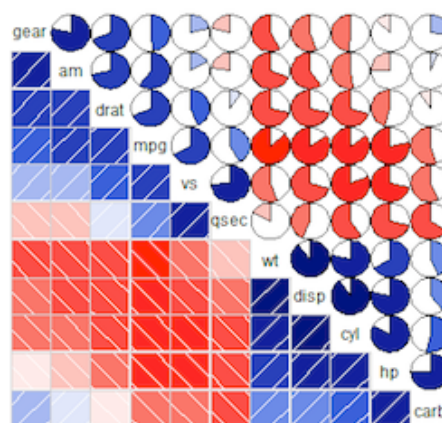
Graphs

Advanced Graphs

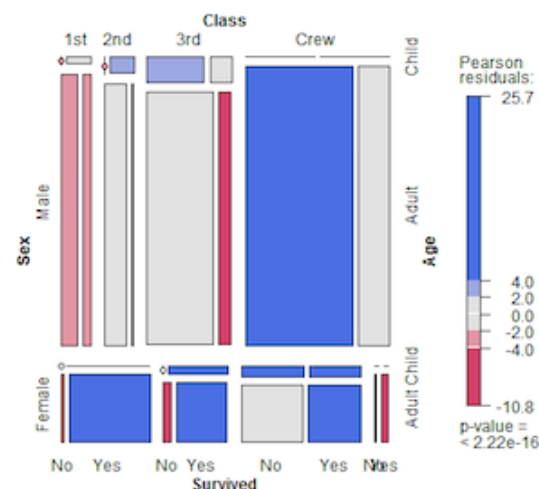
ABOUT THIS SITE

About Quick-R

Correlations Among Auto Characteristics



Who Survived the Titanic?



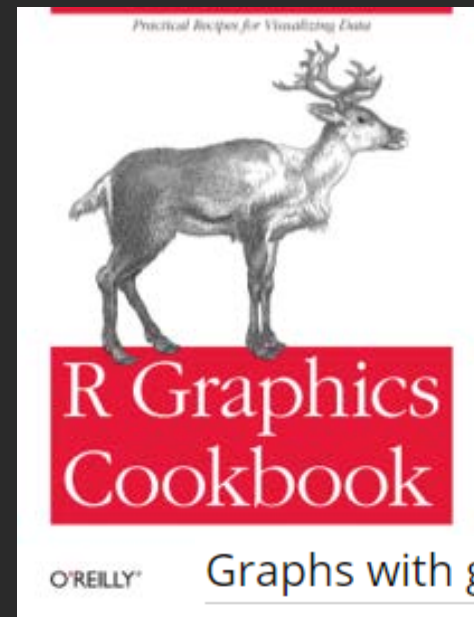
R is an elegant and comprehensive statistical and graphical programming language. Unfortunately, it can also have a [steep learning curve](#). I created this website for both current R users, and experienced users of other statistical packages (e.g., **SAS**, **SPSS**, **Stata**) who would like to transition to R. My goal is to help you quickly access this language in your work

<http://www.cookbook-r.com/>

Welcome to the Cookbook for R. The goal of the

Most of the code in these pages can be copied

1. Basics
2. Numbers
3. Strings
4. Formulas
5. Data input and output
6. Manipulating data
7. Statistical analysis
8. Graphs
9. Scripts and functions
10. Tools for experiments



Graphs with ggplot2

1. Bar and line graphs (ggplot2)
2. Plotting means and error bars (ggplot2)
3. Plotting distributions (ggplot2) - Histograms
4. Scatterplots (ggplot2)
5. Titles (ggplot2)
6. Axes (ggplot2) - Control axis text, labels, and ticks
7. Legends (ggplot2)
8. Lines (ggplot2) - Add lines to a graph.
9. Facets (ggplot2) - Slice up data and graph
10. Multiple graphs on one page (ggplot2)
11. Colors (ggplot2)

- <https://r4ds.had.co.nz/>

[Welcome](#)

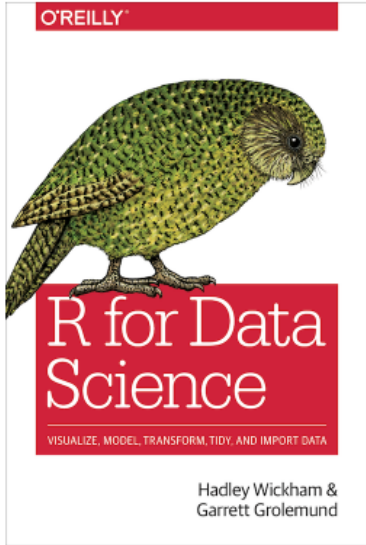
- 1 Introduction
- I Explore
 - 2 Introduction
 - 3 Data visualisation
 - 4 Workflow: basics
 - 5 Data transformation
 - 6 Workflow: scripts
 - 7 Exploratory Data Analysis
 - 8 Workflow: projects
- II Wrangle
 - 9 Introduction
 - 10 Tibbles
 - 11 Data import
 - 12 Tidy data
 - 13 Relational data

R for Data Science

Welcome

This is the website for “**R for Data Science**”. This book will teach you how to do data science with R: You’ll learn how to get your data into R, get it into the most useful structure, transform it, visualise it and model it. In this book, you will find a practicum of skills for data science. Just as a chemist learns how to clean test tubes and stock a lab, you’ll learn how to clean data and draw plots—and many other things besides. These are the skills that allow data science to happen, and here you will find the best practices for doing each of these things with R. You’ll learn how to use the grammar of graphics, literate programming, and reproducible research to save time. You’ll also learn how to manage cognitive resources to facilitate discoveries when wrangling, visualising, and exploring data.

This website is (and will always be) **free to use**, and is licensed under the [Creative Commons Attribution-NonCommercial-NoDerivs 3.0](#) License. If you’d like a **physical copy** of the book, you can order it from [amazon](#); it was published by O’Reilly in January 2017. If you’d like to support the book, you can also order it from [Amazon](#) or [O’Reilly](#).





谢谢

共勉，学无止境