R语言与生物统计学(2)

基础生物统计学与统计计算分析

http://cbb.sjtu.edu.cn/~mywu/bi217

本节内容

- 集合操作
- 向量、矩阵、data.frame、list
- 矩阵的运算
- 读写文件
- R绘图

集合（set）操作：交、并、差

```
x <- 0:100
y <- seq(0, 100, by=10)

intersect(x, y); intersect(y, x) # 交：x ∩ y
union(x, y); union(y, x) # 并：x ∪ y
setdiff(x, y); setdiff(y, x) # 差：x\y

all(y %in% x); x %in% y # %in% means “belong to”
is.element(x, y); is.element(y, x)
y <- sample(y) # permutation
mymat <- match(y, x) # return the indices for each y in x
x[mymat]
subset(x, x<50) # 子集
which(x<50)
```

match()函数的应用

在microarray的数据中，每个chip返回来的基因的顺序是不同的，需要把这些基因的表达数据按照相同的顺序排列起来，组成一个完整的数据集。

例如在chip1中，基因的顺序是(1,5,2,3,4)，但是在其他chip中，基因的顺序可能又不一样，这时候我们就需要match()函数，让所有的基因都按照我们预想的(1,2,3,4,5)排列

```
indices <- c(1,2,3,4,5)
mymatch <- match(indices, chip1$gene)
chip1 <- chip1[mymatch,]
```

向量

```
# vector：c(), seq(), rep()
# inner product
x <- 1:5; y <- 2:6
t(x) %*% y
sum( x * y)

# outer operation
outer(x, y, “+”); outer(x, y, ‘*’)
# outer product
z <- x %*% t(y)
z[4, 6] == x[4] * y[6]
```

向量的内积与外积

```
## inner product, resulting in a scalar
## since vector is column vector
x <- 1:10; y <- 11:20
(xy.inner <- t(x) %*% y)

## outer product, resulting in a matrix
## the result is a 5by3 matrix
x <- 1:5; y <- 1:3
xy.outer <- x %*% t(y)
```

向量乘法的应用

```
## 计算总和、均值、方差
X <- rnorm(100)
one <- rep(1, length(X))
X.sum <- t(one) %*% X
sum(X)
X.mean <- t(one) %*% X/length(X)
mean(X)
X.var <- t(X-X.mean) %*% (X-X.mean) / (length(X)-1)
var(X)
X - X.mean
## 向量的归一化处理
scale(X, center=T, scale=T)
```

矩阵(matrix)
数据框(data.frame)

- 矩阵的元素必须是单一的，通常为数值类型（numeric）$\mathbb{R}^{m\times n}$
- 数据框的元素可以是多种，例如可以包含数值类型、布尔变量、分类变量等等
- 矩阵和数据框常常可以相互转化，对应的R函数分别为data.frame()和as.matrix()

矩阵

```
## construction of a matrix, matrix() function
X <- matrix(1:20, nr=4, nc=5, byrow=F)
x <- 1:5
y <- 5:1
A <- cbind(x, y) # column binding
B <- rbind(x, y) # row binding

## an element of a matrix
X[4,3]
## subset of a matrix
X[1:3,] # return the rows 1-3
X[-4, c(3,2,4,5)] # return the columns 3, 2, 4, 5 except the 4th row
X[, sample(1:5)] #permute the columns
```

矩阵的运算

```
x <- matrix(1:25, 5, 5)
y <- matrix(1:25, 5, 5)
x + y; x-y
yt <- t(y) # transposition
x * y # element-wise multiplication
z <- x %*% y # matrix multiplication
z
z[3,5] == sum(x[3,] * y[,5])
all(z[,5] == x %*% y[,5])
```

矩阵的乘法

```
## 矩阵A、B相乘，满足的条件是A的列数必须与B的行数相等
## C=AB，则C的c[i,j] = sum(a[i,]*b[,j])
## 也就是说c[i,j]是向量a[i,]与b[,j]的点积
A <- matrix(1:16, 4, 4)
B <- matrix(1:20, 4, 5)
C <- A %*% B
M <- matrix(NA, 4, 5)
for (i in 1:4) for (j in 1:5) M[i,j] <- sum(A[i,]*B[,j])
all(M==C)

## 另一种看法则是c[,j]是A的列向量a[,k]的线性组合
## 其权重系数为b[k,j], c[,j]=sum(a[,k]*b[k,j])
X <- matrix(0, 4, 5)
for (k in 1:5)  for (m in 1:4) X[,k] <- X[,k] + A[,m]*B[m,k]
all(X==C)

## 此外，矩阵C还可以看作r个矩阵C[k]的和，其中每个矩阵C[k]
## 都是A的列向量和B的行向量的外积
D <- list()
for (r in 1:4) D[[r]] <- outer(A[,r], B[r,])
E <- matrix(0,4,5)
for (k in 1:4) E <- E + D[[k]]
all(E==C)
```

矩阵乘法的应用

```
x <- matrix(1:20, 4, 5)
rowMeans(x)
apply(x, 1, mean)
colMeans(x)
apply(x, 2, mean)
one1 <- rep(1, dim(x)[1])
x.means <- t(one1) %*% x / dim(x)[1]
one2 <- rep(1, dim(x)[2])
as.vector(x %*% one2 / dim(x)[2])
x.diff <- x - one1 %*% x.means
(x.scaled <- scale(x, center=T, scale=F))
x.cov <- t(x.diff) %*% x.diff / (dim(x)[1]-1)
cov(x)
```

list

```
## list常常作为函数的返回结果，尤其是当返回结果不只一个的时候
## list
xl <- list()
xl[[1]] <- c(1,3,5)
xl[[2]] <- "standard normal distribution"
xl[[1]][2]
x
f <- function(x) {
   mean <- mean(x);
   sd <- sd(x);
   return(list(m=mean, s=sd))
}
x <- rnorm(100)
xf <- f(x)
xf
xf$m; xf$s
```

控制结构

```
## if (…) { dosomething(); }
## else { doanotherthing(); }
## another kind of structure
c <- rnorm(100)
x <- ifelse(c>=0, 1, -1)
## while (condition) { dosomething }
for (i in 1:10) cat(i);
```

函数型语言

```
## 函数的定义
f <- function(x) {
  return(x^2+2*x+1)
}

## 调用函数
x <- rnorm(1000)
y <- f(x)
plot(x, y, type=“l”)
cor(x, y) ## correlation coefficient
```

apply(), sapply(), lapply()

```
## apply() used for one dimension of an array or data.frame
x <- matrix(1:20, 4, 5)
apply(x, 2, median)
apply(x, 1, sum)
## lapply() return a list
M <- list(m=3, n=c(TRUE, FALSE, TRUE), r=4:7)
lapply(M, quantile)
## sapply() used for each element of a vector
## the return result is a vector
x <- sapply(1:5, seq)
lapply(x, fivenum)
```

tapply与aggregate

```
## tapply() will return a table
data(warpbreaks)
warpbreaks
tapply(warpbreaks$breaks, warpbreaks[,-1], sum)
aggregate(warpbreaks[,1], by=c(warpbreaks[,-1]), sum)

## it can be treated as the weighed version of table()
## 比如我们在工作中，需要考察(吸烟, 肺癌)和血压的关系
## 我们会用tapply()统计(+,+), (+,-), (-,+), (-,-)的平均血压
## 而table()只能统计这四组的人数
```

因子水平factor

```
### 产生因子水平
## gl(n,p,length)
# n            -       因子水平数
# p            -       同一水平重复试验次数
# length       -       总样本大小
gender <- rep(c('male','female'), each=8)
gender <- factor(gender)
## 直接用gl函数就要简单的多了
(gender <- gl(2,8,labels=c('male','female'))
(gender2 <- gl(2,1,length=16, labels=c('male','female'))) # balanced
(gender3 <- gl(2,2,length=16, labels=c('male','female'))) # balanced
(gender4 <- gl(2,3,length=16, labels=c('male','female'))) # imbalanced
table(gender)
table(gender2)
table(gender3)
table(gender4)
```

变量的初始化

```
x <- numeric()
xl <- list()
M <- matrix(NA,3, 4)
D <- data.frame(M)
rownames(D) <- paste("R", 1:3, sep="")
colnames(D) <- paste("C", 1:4, sep="")
class(x)
class(str)
typeof(x)
typeof(string)
mode(x)
mode(string)
```

读写文件

- scan()函数可以读取单个向量
- read.table()函数可以读取以TAB作为分隔符的表，读入的结果为一个data.frame
- read.csv()可以读取以comma为分割符的表
- read.delim()等可以读取任意分隔符的表

- 写文件用write.table()函数将变量保存为文本文件；
- 还可以用save.image()和save()函数将变量存取为二进制文件*.RData
- *.RData文件可以用load()命令载入

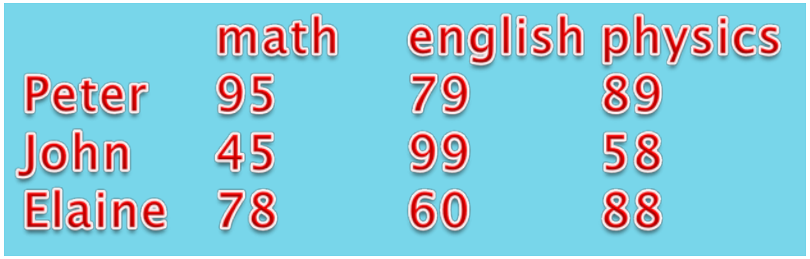
scan()

```
x <- scan(“number.txt”)
x
```

```
0.312
0.46
0.374
0.411
0.974
0.645
0.392
```

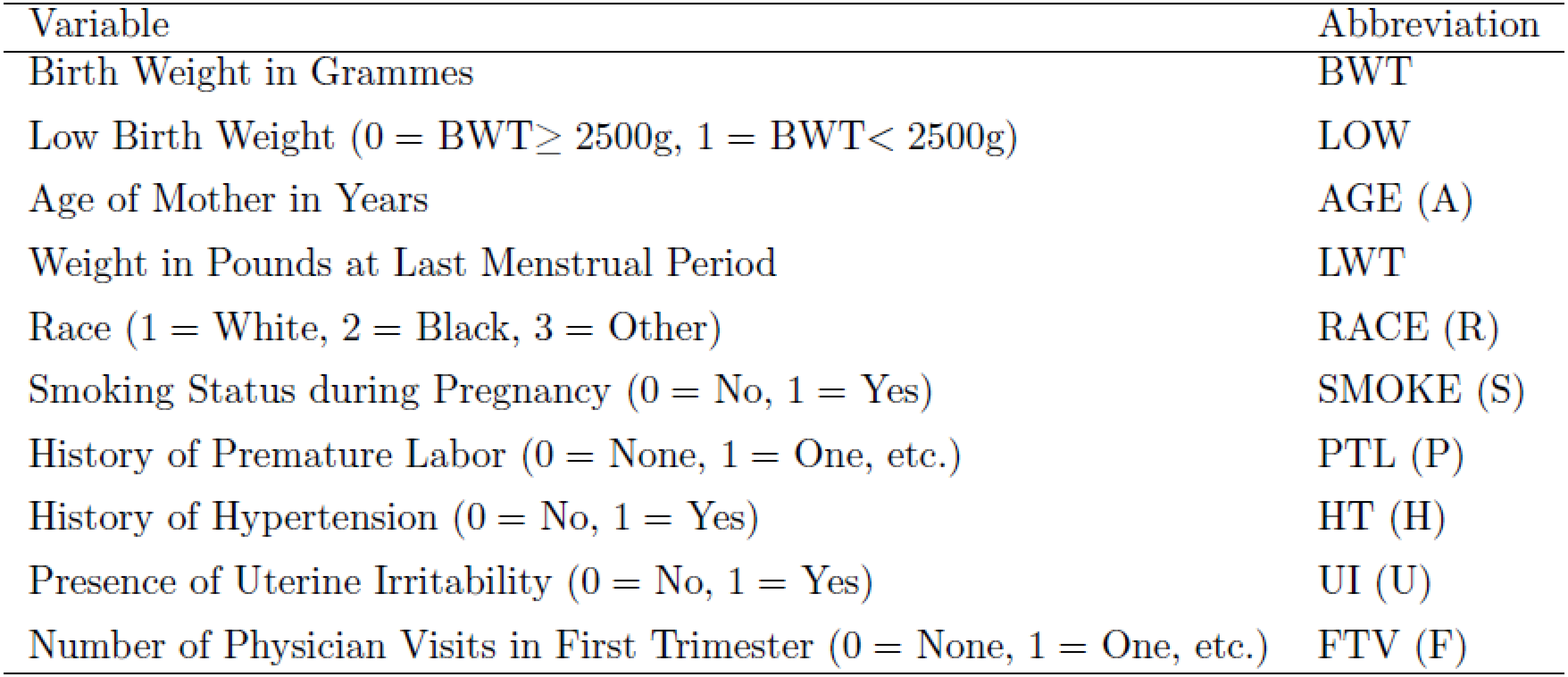
read.table()

```
x <- read.table(“scores.txt”, header=T, row.names=1)
x$math
names(x)
class(x)
attach(x)
math
x
```



	math	english	physics
Peter	95	79	89
John	45	99	58
Elaine	78	60	88

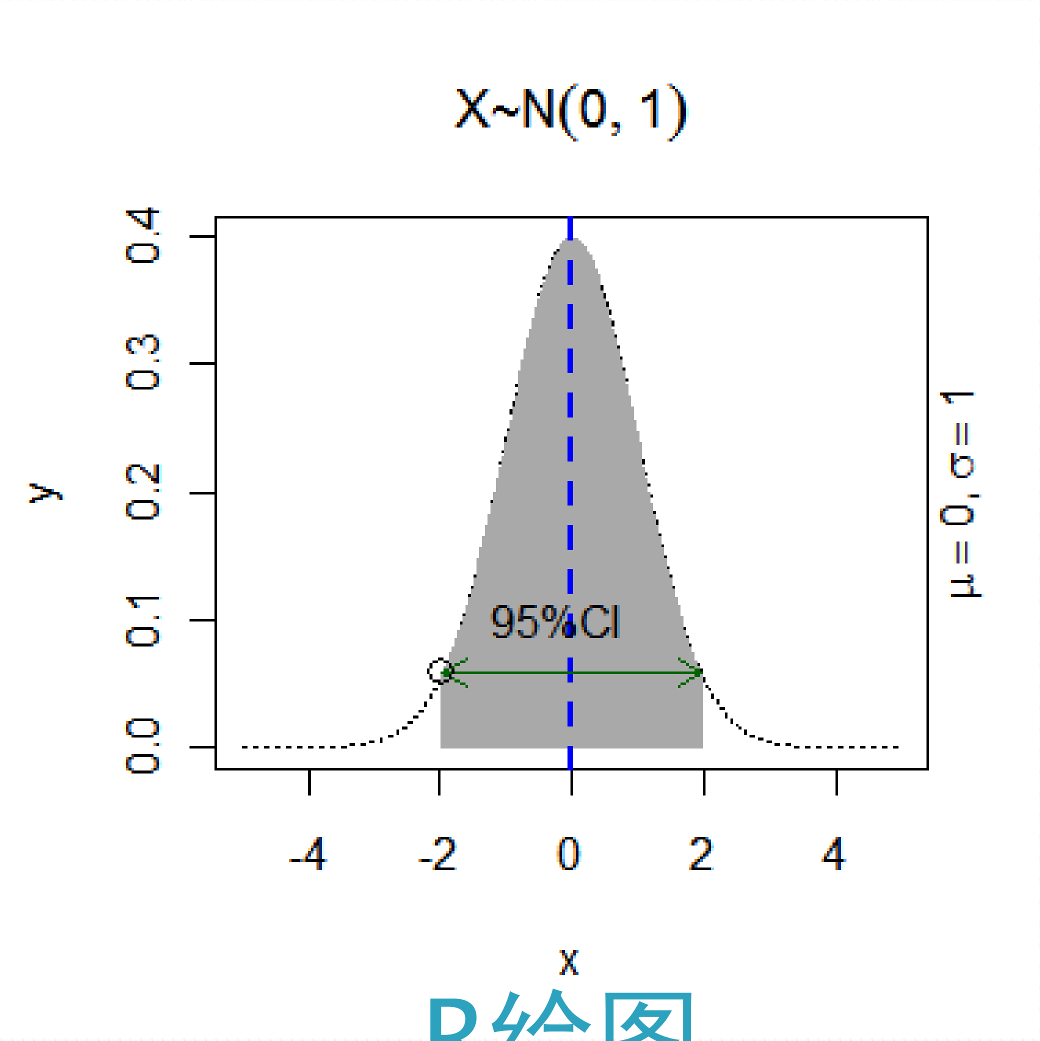
用read.table()读取birthweights.dat



Variable	Abbreviation
Birth Weight in Grammes	BWT
Low Birth Weight (0 = BWT$\geq$ 2500g, 1 = BWT$<$ 2500g)	LOW
Age of Mother in Years	AGE (A)
Weight in Pounds at Last Menstrual Period	LWT
Race (1 = White, 2 = Black, 3 = Other)	RACE (R)
Smoking Status during Pregnancy (0 = No, 1 = Yes)	SMOKE (S)
History of Premature Labor (0 = None, 1 = One, etc.)	PTL (P)
History of Hypertension (0 = No, 1 = Yes)	HT (H)
Presence of Uterine Irritability (0 = No, 1 = Yes)	UI (U)
Number of Physician Visits in First Trimester (0 = None, 1 = One, etc.)	FTV (F)

部分R code

```
bwt <- read.table(“birthweights.dat”,
                                header=T)
## 基本统计
summary(bwt)
str(bwt)
bwt$BWT
attach(bwt)
BWT
```



R绘图

基本绘图函数

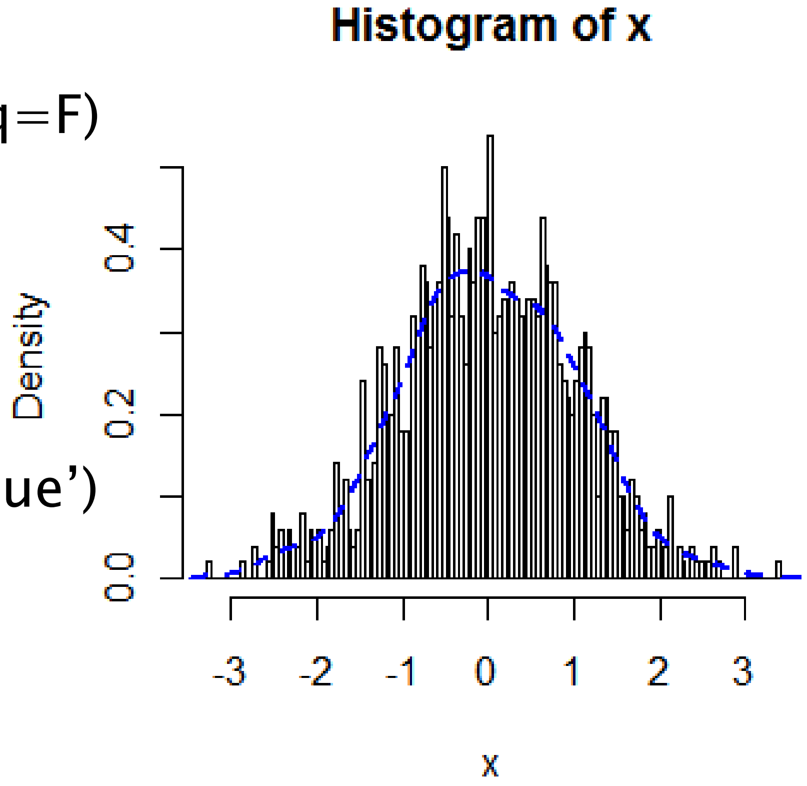
- 常用的绘图函数主要有
  - hist(): 直方图， 显示分布频率或密度
  - barplot()： 条形图，常用于多个样本分类频率的比较
  - boxplot()：盒状图，常用于多个组分布的比较
  - plot()：基本画图函数，散点图或者折线图等
  - points()：基本绘图函数，画散点，不能单独使用
  - lines()：折线图，不能单独使用
  - segments()：绘制线段，不能单独使用
  - arrows()：绘制箭头，箭头的形状大小要注意设置
  - rect()：绘制四边形的函数
  - polygon()：绘制多边形的函数
  - abline()：绘制直线
  - axis()：绘制坐标
  - box()：为图形添加外框

绘图函数（续）

  - text()：在指定坐标添加文字
  - mtext()：在坐标变上添加文字说明
  - legend()：添加图示说明
  - identify()：为数据点添加标注
- 比较复杂的绘图函数
  - contour()：类似等高线图
  - persp()：立体的侧面视图
  - image()：
  - curve()：可以用来绘制指定方程的曲线

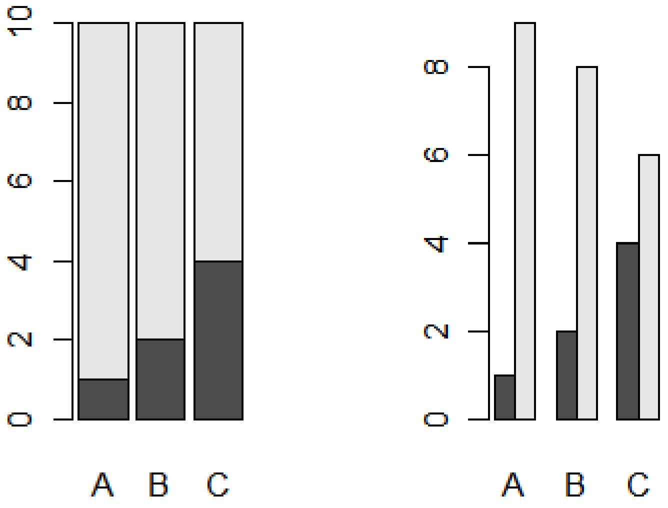
histogram

```
x <- rnorm(1000)
hx <- hist(x, nclass=120, freq=F)
names(hx)
hx$breaks
hx$mids
hx$counts
lines(density(x), lty=2, col=‘blue’)
```



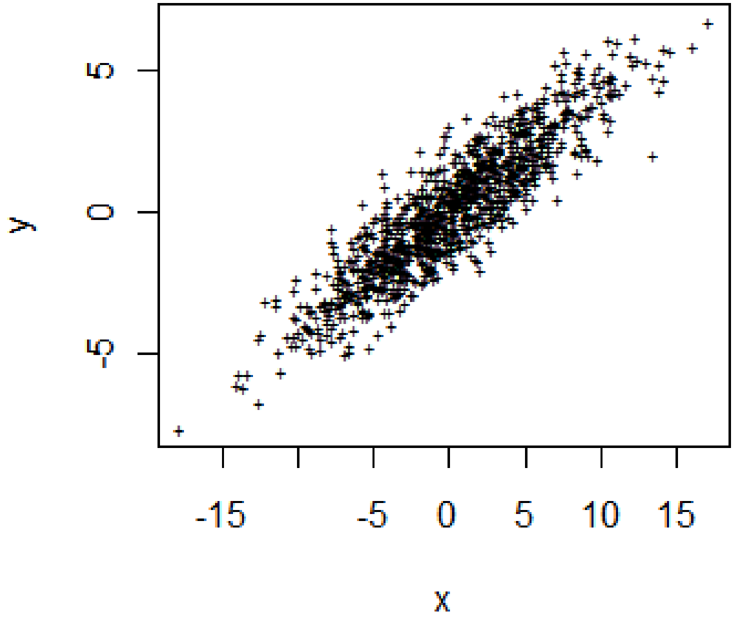
barplot

```
x <- matrix(c(1,9,2,8,4,6), 2, 3)
colnames(x) <- LETTERS[1:3]
par(mfrow=c(1,2))
barplot(x, beside=T,
        col=c(“red”,”blue”))
barplot(x, beside=F)
```



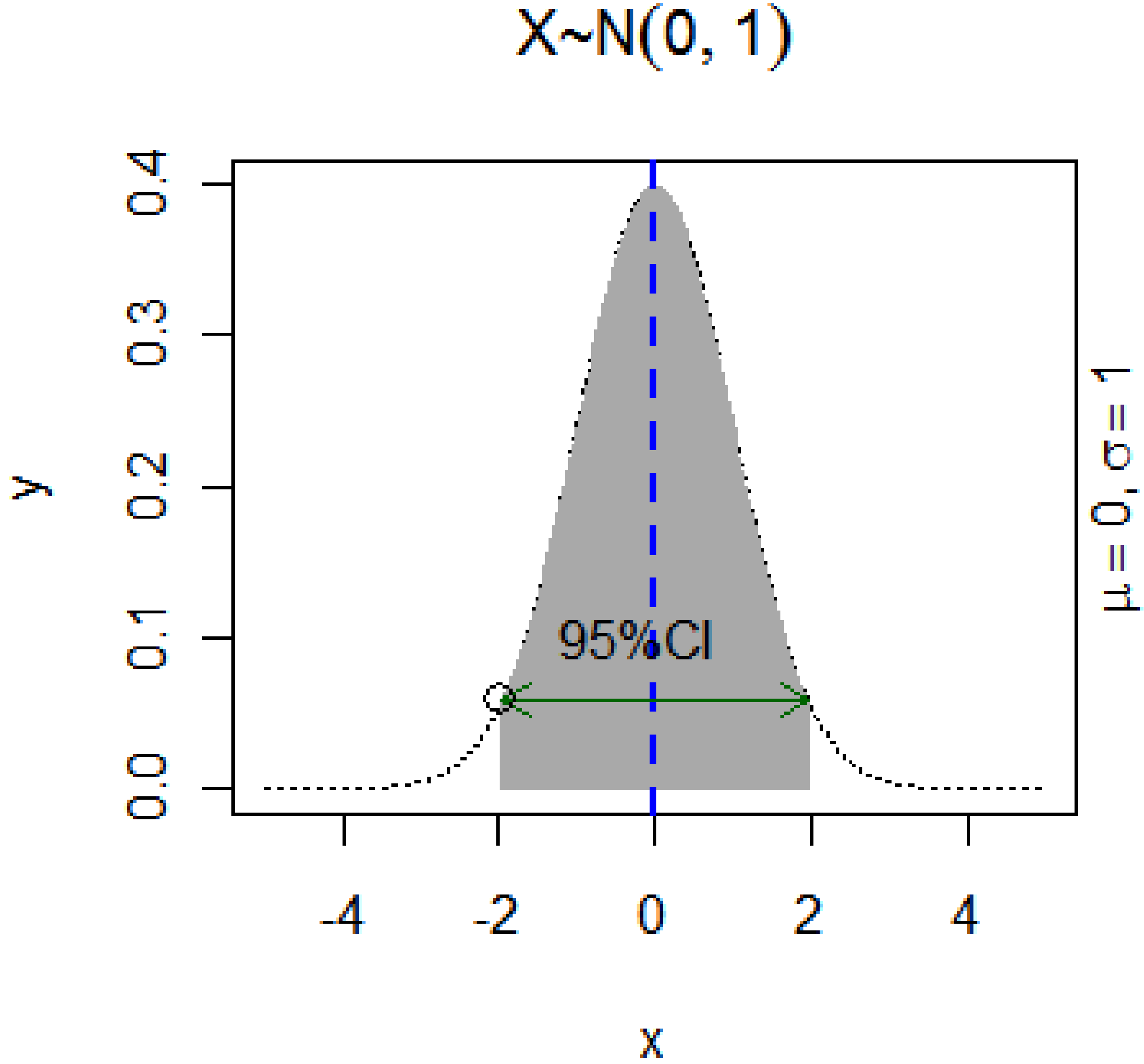
scatterplot

```
x <- 5*rnorm(1000)
y <- 0.4 * x + rnorm(1000)
plot(x, y, pch="+", cex=.6)
abline(c(0, .4))
```



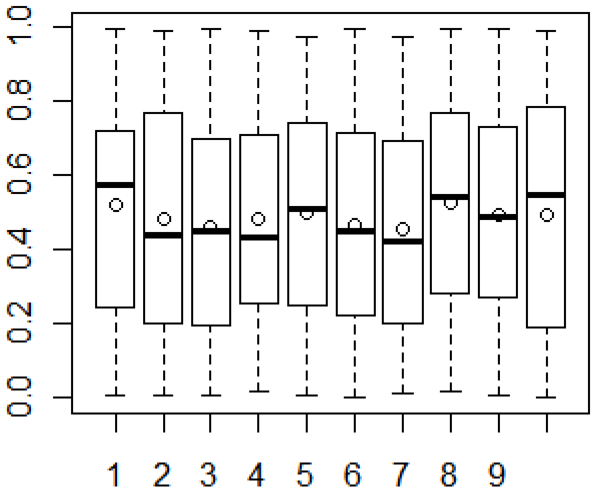
组合图：线段、箭头、图标

```
x <- seq(-5, 5, length=1001)
y <- dnorm(x)
plot(x, y, type='n')
lines(x,y,lty=3)
x0 <- qnorm(0.025)
x1 <- qnorm(0.975)
segments(x0,0,x0,dnorm(x0))
segments(x1,0,x1,dnorm(x1))
curve(dnorm, x0, x1, add=T, type="h", col='darkgray')
abline(v=0, lty=2, lwd=2, col='blue')
text(-.20, .1, "95%CI")
arrows(x0, dnorm(x0), x1, dnorm(x1), length=.1, code=3,
   col='darkgreen')
symbols(x0, dnorm(x0), circles=0.2, add=T, inches=F)
mtext(expression(paste(mu==0, ", ", sigma==1)), 4)
title(expression(paste(X,"~", N(0,1))))
```


X~N(0, 1)
y
0.0
0.1
0.2
0.3
0.4
95%CI
μ = 0, σ = 1
-4
-2
0
2
4
x

boxplot

```
x <- matrix(rnorm(1000), 100, 10)
boxplot(x)
x <- matrix(runif(1000), 100, 10)
xbox <- boxplot(x)
names(xbox)
xmean <- apply(x, 2, mean)
xmed <- apply(x, 2, median)
library(e1071)
xskew <- apply(x, 2, skewness)
xskew > 0; xmean > xmed
points(1:10, xmean)
```



辅助图

```
## axis()
## text()
## mtext()
## legend()
```

R simulation

```
xmeans <- NULL
for (i in 1:1000) {
  x <- rnorm(100, 0, 4)
   xmeans <- c(xmeans, mean(x))
}
xmeans
mean(xmeans)
var(xmeans)
sd(xmeans)
## 从结果你看到了什么结论？
```