第三讲：重采样方法

张伟平

统计与金融系

第三讲　重采样方法

3.1 重采样技术

■ Resampling is a computationally intensive statistical technique in which multiple new samples are drawn (generated) from the data sample or from the population inferred by the data sample.

■ 为什么使用重采样技术？

- 对于样本所来自的总体分布所知很少
- 传统检验方法所需的假设条件不满足
- 样本量很小，大样本统计推断不合适
- 概念上简单，可以简化计算

■ 应用场合:

- 估计一个估计的标准差、偏差等 (Jackknife, Bootstrap)
- 估计置信区间 (不假设正态分布)
- 随机化检验 (permutation tests)
- 模型选择 (cross-validation)

重采样技术历史

1935 *randomization tests*, Fisher 

1948 *cross-validation*, Kurtz 

1958 *jackknife*, Quenouille-Tukey 

1979 *bootstrap*, Efron 

重采样技术分类

■ In randomization we systematically shuffle observed data many times (no replacement)

- Unconcerned about parameters, used for testing hypotheses concerning distributions

■ In bootstrapping we draw samples with replacement from the observed data

- Focused primarily on parameters (e.g. confidence intervals, hypothesis testing)

3.2 Jackknife(刀切) 方法

■ Jackknife(刀切法) 是由 Quenouille(1949,1956) 提出的重采样方法.

■ Jackknife 可用于估计偏差，其类似于"leave-one-out" 的交叉验证方法，本质上和影响函数的想法相同.

■ 令 $x = (x_1, \ldots, x_n)$ 为观测到的样本, 定义第 i 个 Jackknife 样本为丢掉第 i 个样本后的剩余样本, 即

$$x_{(i)} = (x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n).$$

若 $\hat{\theta} = T_n(x)$, 则定义第 i 个 Jackknife 重复为 $\hat{\theta}_{(i)} = T_{n-1}(x_{(i)})$, $i = 1, \cdots, n$.

■ 假设参数 $\theta = T(F)$, 为分布 F 的函数. F_n 为 F 的

经验分布函数. 则 θ 的"plug-in" 估计为 $\hat{\theta} = T(F_n)$.

■ 称一个"plug-in" 估计 $\hat{\theta}$ 是平滑的, 如果数据的小幅变化相应于 $\hat{\theta}$ 的小幅变化.

偏差的 Jackknife 估计

Definition

θ 的一个估计量 $\hat{\theta}$ 的偏差定义为

$$bias(\hat{\theta}) = E\hat{\theta} - \theta.$$

■ 当 $\hat{\theta}$ 的分布未知或者形式很复杂使得期望的计算不可能 (从此分布中抽样变得很困难, Monte Carlo 方法不可

行), 以及在现实中, 我们也不知道 θ 的真值时 (需要估计), 这种情况下偏差是未知的.

- 如果 $\hat{\theta}$ 为一个平滑的 (plug-in) 估计量, 则其偏差 $bias(\hat{\theta})$ 如何估计?
- 记 $\hat{\theta}_{(i)} = T(F_{n-1}(x_{(i)})$, 以及

$$\overline{\hat{\theta}_{(\cdot)}} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}_{(i)}.$$

- 如果 $\hat{\theta}$ 为无偏估计，则

$$E\overline{\hat{\theta}_{(\cdot)}} = \frac{1}{n}\sum_{i=1}^{n}E\hat{\theta}_{(i)} = \theta$$

- 但是，假设

$$bias(\hat{\theta}) = E\hat{\theta} - \theta = \frac{a}{n} + \frac{b}{n^2} + O(\frac{1}{n^3})$$

则

$$
\begin{aligned}
E[\overline{\hat{\theta}_{(\cdot)}} - \hat{\theta}] &= \frac{a}{n(n-1)} + \frac{(2n-1)b}{n^2(n-1)^2} + O(\frac{1}{n^3}) \\
&= \frac{1}{n-1}\Big[\frac{a}{n} + \frac{b}{n^2} + O(\frac{1}{n^2})\Big] \approx \frac{1}{n-1} bias(\hat{\theta})
\end{aligned}
$$

考虑方差的估计

$$
\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2.
$$

■ 估计量 $\hat{\theta}$ 是 σ^2 的有偏估计, 偏差为 $bias(\hat{\theta}) = E\hat{\theta} - \sigma^2 = -\frac{\sigma^2}{n}$.

■ 每一个 Jackknife 估计是基于样本量 $n-1$ 的样本构造, 因此 Jackknife 重复 $\hat{\theta}_{(i)}$ 的偏差为 $-\frac{\sigma^2}{n-1}$. 所以

$$
E[\hat{\theta}_{(i)} - \hat{\theta}] = E[\hat{\theta}_{(i)} - \theta] - E[\hat{\theta} - \theta]
$$

$$= bias(\hat{\theta}_{(i)}) - bias(\hat{\theta}) = -\frac{\sigma^2}{n-1} - (-\frac{\sigma^2}{n}) = \frac{bias(\hat{\theta})}{n-1}.$$

- 这些例子启发我们可以得到偏差的一个合理估计

设 $X_1, \ldots, X_n$ $i.i.d$ F, $\hat{\theta}$ 为 $\theta = T(F)$ 的一个估计量，则偏差 $bias(\hat{\theta})$ 的 Jackknife 估计 (Quenouille) 为

$$bias_{jack} = (n-1)(\overline{\hat{\theta}_{(\cdot)}} - \hat{\theta})$$

Definition

θ 的 Jackknife 估计为

$$\hat{\theta}_{jack} = \hat{\theta} - bias_{jack}(\hat{\theta}) = n\hat{\theta} - (n-1)\overline{\hat{\theta}_{(\cdot)}}$$

对 Jackknife 估计, 我们有下面的结论:

Theorem 1. 设 $X_1, \ldots, X_n$ *i.i.d* F, $\hat{\theta}$ 为 $\theta = T(F)$ 的一个估计量, $\hat{\theta}_{jack}$ 为 θ 的 *Jackknife* 估计量, 若 $\hat{\theta}$ 的偏差 $bias(\hat{\theta}) = E_F\hat{\theta} - \theta = \sum_{k=1}^{\infty} \frac{b_k(\theta)}{n^k} = O(\frac{1}{n})$, 则 θ 的 *Jackknife* 估计 $\hat{\theta}_{jack}$ 的偏差 $bias(\hat{\theta}_{jack}) = E_F\hat{\theta}_{jack} - \theta = O(\frac{1}{n^2})$.

Proof.

$$
\begin{aligned}
bias(\hat{\theta}_{jack}) &= E_F\hat{\theta}_{jack} - \theta = E_F[n\hat{\theta} - (n-1)\overline{\hat{\theta}_{(\cdot)}}] - \theta \\
&= n[E_F\hat{\theta} - \theta] - (n-1)[E_F\overline{\hat{\theta}_{(\cdot)}} - \theta] \\
&= n[E_F\hat{\theta} - \theta] - (n-1)[\frac{1}{n}\sum_{i=1}^{n}(E_F\hat{\theta}_{(i)} - \theta)] \\
&= b_1(\theta) + \sum_{k=2}^{\infty}\frac{b_k(\theta)}{n^{k-1}} - \frac{n-1}{n}\sum_{i=1}^{n}\sum_{k=1}^{\infty}\frac{b_k(\theta)}{(n-1)^k}
\end{aligned}
$$

$$= b_1(\theta) + \sum_{k=1}^{\infty} \frac{b_{k+1}(\theta)}{n^k} - [b_1(\theta) + \sum_{k=1}^{\infty} \frac{b_{k+1}(\theta)}{(n-1)^k}]$$

$$= -\frac{b_2(\theta)}{n(n-1)} - \sum_{k=2}^{\infty} b_{k+1}(\theta)[\frac{1}{(n-1)^k} - \frac{1}{n^k}] = O(\frac{1}{n^2}).$$

$\square$

↑Example

例: 计算**patch**数据中比值参数的估计偏差的 Jackknife 估计.

↓Example

↑Code

```
data(patch, package = "bootstrap")
n <- nrow(patch)
y <- patch$y
z <- patch$z
theta.hat <- mean(y) / mean(z)
print (theta.hat)
#compute the jackknife replicates
theta.jack <- numeric(n)
for (i in 1:n)
    theta.jack[i] <- mean(y[-i]) / mean(z[-i])
bias <- (n - 1) * (mean(theta.jack) - theta.hat)
print(bias)  #jackknife estimate of bias
```

↓Code

标准差的 Jackknife 估计

■ 将 Jackknife 估计 $\hat{\theta}_{jack}$ 改写为

$$\hat{\theta}_{jack} = \frac{1}{n}\sum_{i=1}^{n}\tilde{\theta}_i = \frac{1}{n}\sum_{i=1}^{n}[n\hat{\theta} - (n-1)\hat{\theta}_{(-i)}]$$

■ 从而 Jackknife 估计 $\hat{\theta}_{jack}$ 的方差 $Var(\hat{\theta}_{jack})$ 的估计为

$$\begin{aligned}\widehat{Var}(\hat{\theta}_{jack}) &= \frac{1}{n}\widehat{Var}(\tilde{\theta}_i) = \frac{1}{n}\frac{1}{n-1}\sum_{i=1}^{n}(\tilde{\theta}_i - \hat{\theta}_{jack})^2 \\ &= \frac{1}{n}\frac{1}{n-1}\sum_{i=1}^{n}[(n\hat{\theta} - (n-1)\hat{\theta}_{(i)}) - (n\hat{\theta} - (n-1)\overline{\hat{\theta}_{(\cdot)}})]^2 \\ &= \frac{n-1}{n}\sum_{i=1}^{n}(\hat{\theta}_{(i)} - \overline{\hat{\theta}_{(\cdot)}})^2.\end{aligned}$$

- $\hat{\theta}$ 的标准差的 Jackknife 估计 (Tukey) 定义为

$$\hat{se}_{jack} = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{\theta}_{(i)} - \overline{\hat{\theta}_{(\cdot)}})^2}.$$

- 例如当 θ 为总体均值时, $\hat{\theta} = \bar{x}$, 其方差估计为

$$Var(\hat{\theta}) = \frac{\hat{\sigma}^2}{n} = \frac{1}{n(n-1)}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

记 $\theta_{(i)} = \frac{n\bar{x}-x_i}{n-1}$, 则 $\overline{\hat{\theta}_{(\cdot)}} = \frac{1}{n}\sum_i \hat{\theta}_{(i)} = \hat{\theta}$, $\hat{\theta}_{(i)} - \overline{\hat{\theta}_{(\cdot)}} = \frac{\bar{x}-x_i}{n-1}$. 因此有

$$\hat{se}_{jack} = \sqrt{Var(\hat{\theta})}.$$

计算**patch**数据中比值参数的估计标准差的 Jackknife 估计.

↑Example

↓Example

```
se <- sqrt((n-1) *
    mean((theta.jack - mean(theta.jack))^2))
print(se)
```

↑Code

↓Code

■ 在一些情况下, 可以证明 $\widehat{Var}(\hat{\theta}_{jack})$ 为 $Var(\hat{\theta}_{jack})$ 的相合估计

Theorem 2. 令 $\mu = EX_1, \sigma^2 = Var(X_1) < \infty$, 记 $\hat{\theta}_n = g(\bar{X})$, 其中 $\bar{X}$ 为样本均值, g 为连续函数, 在 μ 处有非零导数, 则

(1) $\frac{\hat{\theta}_n - g(\mu)}{\sigma_n} \rightsquigarrow N(0,1), \quad \sigma_n^2 = \frac{1}{n}[g'(\mu)]^2\sigma^2.$

(2) $\widehat{Var}(\hat{\theta}_{jack})/\sigma_n^2 \to 1, a.s.$

■ 当 $T(F)$ 不够光滑, 则相合性可能不成立.

Theorem 3. *(Efron,1982)* 设 $T(F) = F^{-1}(p)$ 为 p 分位数, f 为 F 的密度, 则 $\widehat{Var}(\hat{\theta}_{jack})$ 不是相合估计. 特别地, 对 $p = 1/2$ 有 $\widehat{Var}(\hat{\theta}_{jack})/\sigma_n^2 \rightsquigarrow (\frac{1}{2}\chi_2^2)^2$, 其中 $\sigma_n^2 = \frac{1}{4nf^2(m)}$, m 为总体中位数.

影响函数与 Jackknife 估计

- 回忆影响函数

$$L(x)=\lim_{\epsilon\to 0}\left[\frac{T((1-\epsilon)F+\epsilon\delta_x)-T(F)}{\epsilon}\right]$$

- 在**经验影响函数**中取 $\epsilon=-\frac{1}{n-1}$, 则

$$\begin{aligned}L_n(x_i) &\approx \left[\frac{T((1-\epsilon)F_n+\epsilon\delta_{x_i})-T(F_n)}{\epsilon}\right]\\ &=(n-1)(\hat{\theta}-\hat{\theta}_{(i)})\equiv l_i\end{aligned}$$

- 从而

$$\widehat{Var}(\hat{\theta}_{jack})=\frac{1}{n(n-1)}\left(\sum_{i=1}^{n}l_i^2-nb^2\right),\quad b=-\frac{1}{n}\sum_{i=1}^{n}l_i.$$

Jackknife 失效情形

■ 若估计量 $\hat{\theta}$ 不够平滑, Jackknife 方法就可能会失效. 中位数就是一个不平滑统计量的例子.

↑Example

例 (Jackknife 方法失效) 用 Jackknife 方法估计从 1,2,...,100 中随机抽取的 10 个数的中位数的标准差.

↓Example

↑Code

```
set.seed(123) #for the specific example given
#change the seed to see other examples
n <- 10
x <- sample(1:100, size = n)
```

```
#jackknife estimate of se
M <- numeric(n)
for (i in 1:n) {          #leave one out
    y <- x[-i]
    M[i] <- median(y)
}
Mbar <- mean(M)
print(sqrt((n-1)/n * sum((M - Mbar)^2)))
#bootstrap estimate of se
Mb <- replicate(1000, expr = {
      y <- sample(x, size = n, replace = TRUE)
        median(y) })
print(sd(Mb))
```

↓Code

本例中 Jackknife 估计和 Bootstrap 估计相差很远, 显然存在问题. 事实上, 由于中位数不是平滑的, Jackknife 方法失效了.

3.3 Bootstrap(自助) 方法

■ Bootstrap(自助) 方法是一类重采样方法, 常用来估计统计量 $T_n = g(X_1, \ldots, X_n)$ 的方差和分布。

■ 自助法是现代非参数统计中非常重要的一个想法，Casella & Berger 称其为“perhaps the single most important development in statistical methodology in recent times”.

■ Bootstrap 一词可以指非参数 Bootstrap, 也可以指参数 Bootstrap.

■ 参数 Bootstrap 是指总体分布完全已知, 利用 Monte Carlo 方法从此总体中抽样进行统计推断; 而非参数 Bootstrap 是指总体分布完全未知, 利用重采样方法从样本中

(再) 抽样进行统计推断.

■ 假设 $x = (x_1, \cdots, x_n)$ 为一个从总体分布 $F(x)$ 中观测到得样本, X^* 为从 x 中随机选择的一个样本, 则

$$P(X^* = x_i) = \frac{1}{n}, \quad i = 1, \cdots, n.$$

从 x 中有放回的重采样得到随机样本 $X_1^*, \cdots, X_n^*$. 显然随机变量 $X_1^*, \cdots, X_n^*$ 为 $i.i.d$ 的随机变量, 服从 $\{x_1, \cdots, x_n\}$ 上的均匀分布.

■ 经验分布函数 $F_n(x)$ 是 $F(x)$ 的估计, 它也是 $\{x_1, \cdots, x_n\}$ 上的均匀分布随机变量 X^* 的分布函数. 由于 F_n 逼近到 F, Bootstrap 重复抽样下的经验分布函数 F_n^* 是 F_n 的逼近. 这两种逼近可以表示为

$$F \to X \to F_n$$

$$F_n \to X^* \to F_n^*$$

Real World			**Bootstrap World**	
Unknown probability distribution	Observed random sample		Empirical distribution	Bootstrap sample

$$P \longrightarrow X = (X_1, \ldots, X_n) \quad \Longrightarrow \quad \hat{P} \longrightarrow X^* = (X_1^*, \ldots, X_n^*)$$

$$\downarrow \qquad\qquad\qquad\qquad \downarrow$$

$$\hat{\theta} = s(X) \qquad\qquad \hat{\theta}^* = s(X^*)$$

Statistic of interest — Bootstrap replication

■ 假设 $\theta = T(F)$ 是我们感兴趣的参数, $T(F_n)$ 为 $T(F)$ 的 plug-in 估计. 则估计 $T(F_n)$ 的分布特征的 Bootstrap 方法如下

1. 抽取相同样本容量的 $X_1^*, \ldots, X_n^*\ i.i.d \sim F_n$, 得到 F_n^*
2. 计算 $T_n^* = T(F_n^*)$
3. 重复步骤 1 和 2, B 次得到 $T_{n,1}^*, \ldots, T_{n,B}^*$, 最后得到经验分布函数 $F_{B,T}^*(t) = \frac{1}{B}\sum_{b=1}^{B} 1\{T_{n,b}^* \leq t\}$.

例 (F_n 与 Bootstrap 抽样) 假设我们观察到样本

$$x = \{2, 2, 1, 1, 5, 4, 4, 3, 1, 2\}$$

计算 F_n 和 Bootstrap 抽样.

↑Example

↓Example

从 x 中重采样依照选择 $1, 2, 3, 4, 5$ 的概率分别为 0.3, 0.3, 0.1, 0.2, 0.1 进行. 从而从 x 中随机选择的一个样本

X^*, 其分布函数就是经验分布函数, 即

$$F_{X^*}(x) = F_n(x) = \begin{cases} 0, & x < 1; \\ 0.3, & 1 \le x < 2; \\ 0.6, & 2 \le x < 3; \\ 0.7, & 3 \le x < 4; \\ 0.9, & 4 \le x < 5; \\ 1, & x \ge 5. \end{cases}$$

注意如果 F_n 没有靠近 F_X, 则重复抽样下的分布也不会靠近 F_X. 上例中的样本 x 实际上是从 $Poisson(2)$ 中随机产生的, 从 x 中大量重复抽样可以很好的估计 F_n, 但是不能很好的估计 F_X, 因为无论重复多少次重采样, 得到的 Bootstrap 样本都没有 0.

3.3.1 标准差的 Bootstrap 估计

■ 估计量 $\hat{\theta}$ 的标准差的 Bootstrap 估计, 是 Bootstrap 重复 $\hat{\theta}^{(1)},\ldots,\hat{\theta}^{(B)}$ 的样本标准差:

$$\hat{se}_B(\hat{\theta}^*) = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\theta}^{(b)} - \overline{\hat{\theta}^*})^2}.$$

其中 $\overline{\hat{\theta}^*} = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^{(b)}$.

■ 根据 Efron 和 Tibshirini(1993), 要得到标准差一个好的估计, 重复的次数 B 并非需要非常大. $B = 50$ 常常已经足够了, $B > 200$ 是很少见的 (置信区间除外).

↑Example

例 (标准差的 Bootstrap 估计) **bootstrap**包里的法律院校数据集**law**, 记录了 15 所法律院校入学考试的平均成绩 (LSAT) 和 GPA(乘了 100).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
LSAT	576	635	558	578	666	580	555	661	651	605	653	575	545	572	594
GPA	339	330	281	303	344	307	300	343	336	313	312	274	276	288	296

估计 LSAT 和 GPA 之间的相关系数, 并求样本相关系数的标准差的 Bootstrap 估计.

↓Example

1. 数据是成对的 $z_i = (x_i, y_i), i = 1, \ldots, 15$.

2. 可以通过样本相关系数估计相关系数

$$\hat{\tau} = \frac{n\sum_i x_i y_i - \sum_i x_i \sum_i y_i}{\sqrt{n\sum_i x_i^2 - (\sum_i x_i)^2}\sqrt{n\sum_i y_i^2 - (\sum_i y_i)^2}}.$$

3. Bootstrap 对这些数据对重采样来估计标准差. 因此, 算法如下

> 1. 对 Bootstrap 重复的第 b 次 $(b = 1, \ldots, B)$,
>   (a) 有放回的从 $z_1 \cdots, z_n$ 中抽样 $z^{*(b)} = (z_1^*, \ldots, z_n^*)$.
>   (b) 根据 $z^{*(b)}$ 计算 $\hat{\tau}^{(b)}$.
> 2. 使用 $\hat{\tau}^{(1)}, \ldots, \hat{\tau}^{(B)}$ 估计标准差.

样本相关系数为 $cor(LSAT, GPA) = 0.7763745$, 使用

Bootstrap 估计标准差的程序如下:

↑Code

```
library(bootstrap)    #for the law data
print(cor(law$LSAT, law$GPA))
#set up the bootstrap
B <- 200              #number of replicates
n <- nrow(law)        #sample size
R <- numeric(B)       #storage for replicates
#bootstrap estimate of standard error of R
for (b in 1:B) {
    #randomly select the indices
    i <- sample(1:n, size = n, replace = TRUE)
 LSAT <- law$LSAT[i]     #i is a vector of indices
    GPA <- law$GPA[i]
    R[b] <- cor(LSAT, GPA)
}
print(se.R <- sd(R))#output
hist(R, prob = TRUE)
```

↓Code

$se(\hat{\tau})$ 的 Bootstrap 估计为 0.1371913, 由渐近正态得到样本相关系数的标准差为 0.110. ($\sqrt{n}(\hat{\tau}-\tau) \rightsquigarrow N(0,(1-\tau^2)^2)$.)

例使用**boot**函数进行 Bootstrap 估计标准差

↑Example

↓Example

在 R 中, 包**boot**里的**boot**函数可以进行 Bootstrap 估计.

```
boot(data, statistic, R, sim = "ordinary", stype = c("i", "f", "w"),
        strata = rep(1,n), L = NULL, m = 0, weights = NULL,
      ran.gen = function(d, p) d, mle = NULL, simple = FALSE, ...,
          parallel = c("no", "multicore", "snow"),
          ncpus = getOption("boot.ncpus", 1L), cl = NULL)
```

boot 函数中的参数 *statistic* 是一个函数, 用来返回感兴趣的统计量值. 这个函数必须至少有两个参数, 其中

第一个是数据, 第二个表示 Bootstrap 抽样中的指标向量, 频率或者权重等. 因此我们首先写一个函数计算 $\hat{\theta}^{(b)}$. 用 $i = (i_1, \cdots, i_n)$ 表示指标向量, 则计算相关系数的程序为

↑Code

```
tau<-function(x,i){
xi<-x[i,]
cor(xi[,1],xi[,2])
}
```

↓Code

然后我们就可以使用**boot**函数进行 Bootstrap 估计:

↑Code

```
  library(boot)        #for boot function
obj <- boot(data = law, statistic = tau, R = 2000)
  obj
```

```
# alternative method for std.error
y <- obj$t
sd(y)
detach(package:boot)
```

↓Code

观测到的 $\hat{\theta}$ 值用 $t1*$ 标出. 2000 次重复下的 Bootstrap 标准差估计为 0.1326418.

和**boot**函数相似功能的函数是**bootstrap**包里的**bootstrap**函数. 使用此函数重复上述问题的程序如下

↑Code

```
 library(bootstrap)       #for boot function
 n <- 15
theta <- function(i,x){ cor(x[i,1],x[i,2]) }
results <- bootstrap(1:n,2000,theta,law)
sd(results$thetastar) #0.1325971
```

```
detach(package:bootstrap)
```

↓Code

两个函数的用法上有些差异, **bootstrap**包是收录了 Efron & Tibshirani 的书里的程序和数据. **boot** 包是收录了 Davson & Hinkley 的书里的程序和数据.

3.3.2 偏差的 Bootstrap 估计

■ 但是我们已经有了样本, 期望 $E\hat{\theta}$ 可以通过 Bootstrap 方法进行估计. 从而可以得到偏差的估计:

$$\widehat{bias}_B(\hat{\theta}) = E^*\hat{\theta}^* - \hat{\theta}.$$

E^* 表示 Bootstrap 经验分布.

■ 因此一个估计量的偏差的 Bootstrap 估计, 是通过使用 $\hat{\theta}$ 的 Bootstrap 重复来估计 $E\hat{\theta}$. 对一个有限样本 $x=(x_1,\cdots,x_n)$, 有 $\hat{\theta}(x)$ 的 B 个 i.i.d 估计量 $\hat{\theta}^{(b)}$. 则 $\{\hat{\theta}^{(b)}\}$ 的均值是期望值 $E\hat{\theta}^*$ 的无偏估计, 因此偏差的 Bootstrap 估计为

$$\widehat{bias}_B(\hat{\theta})=\overline{\hat{\theta}^*}-\hat{\theta}.$$

这里 $\overline{\hat{\theta}^*}=\frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^{(b)}$.

■ 一个经过偏差修正 (Bias-correction) 的估计量为

$$\tilde{\theta}=\hat{\theta}-\widehat{bias}_B(\hat{\theta}).$$

↑Example

例 (Bootstrap 偏差估计) 估计上例中样本相关系数的偏差

↓Example

↑Code

```
theta.hat <- cor(law$LSAT, law$GPA)
#bootstrap estimate of bias
B <- 2000   #larger for estimating bias
n <- nrow(law)
theta.b <- numeric(B)
for (b in 1:B) {
    i <- sample(1:n, size = n, replace = TRUE)
    LSAT <- law$LSAT[i]
    GPA <- law$GPA[i]
```

```
    theta.b[b] <- cor(LSAT, GPA)
}
bias <- mean(theta.b - theta.hat)
bias
```

↓Code

这个值和前例中的**boot**函数返回的结果非常相近.

↑Example

例 (Bootstrap 偏差估计) 假设 $x = (x_1, \cdots, x_{10}) \sim N(\mu, \sigma^2)$, 求 σ^2 的估计量 $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$ 的偏差

↓Example

↑Code

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

sigma2.hat<-(n-1)*var(x)/n
#bootstrap estimate of bias
B <- 2000   #larger for estimating bias
sigma2.b <- numeric(B)
for (b in 1:B) {
    i <- sample(1:n, size = n, replace = TRUE)
     sigma2.b[b] <-(n-1)*var(x[i])/n
}
bias <- mean(sigma2.b - sigma2.hat)
bias
```

↓Code

在这种情形下, $\hat{\sigma}^2$ 过低的估计了参数 σ^2.

例 (比值参数估计的偏差的 Bootstrap 估计) 以包**bootstrap**里的**patch**数据为例. 该数据是测量了 8 个人使用 3 种不同的药物后血液中某种荷尔蒙的含量. 这三种药物分别是安慰剂, 旧药品 (经过 FDA 审批的), 新药品 (某个新工厂相同的工艺下生产的, 按 FDA 规定, 新工厂生产的药品也要审批). 研究的目的是比较新药和旧药的等价性. 如果可以证明新药和旧药之间的等价性, 则对新药就不需要完全重新向 FDA 申请审批了. 等价性的标准是对比值参数

$$\theta = \frac{E(new) - E(old)}{E(old) - E(placebo)}.$$

若 $|\theta| \le 0.20$, 则新药和旧药就等价. 估计 θ 的统计量为 $\bar{Y}/\bar{Z}$. 这两个变量在**patch**数据中给出. 我们的目标是计算

此估计偏差的 Bootstrap 估计.

↓Example

↑Code

```
data(patch, package = "bootstrap")
patch
n <- nrow(patch)  #in bootstrap package
B <- 2000
theta.b <- numeric(B)
theta.hat <- mean(patch$y) / mean(patch$z)
#bootstrap
for (b in 1:B) {
    i <- sample(1:n, size = n, replace = TRUE)
     y <- patch$y[i]
     z <- patch$z[i]
     theta.b[b] <- mean(y) / mean(z)
     }
bias <- mean(theta.b) - theta.hat
```

```
se <- sd(theta.b)
print(list(est=theta.hat, bias = bias,
           se = se, cv = bias/se))
```

↓Code

3.4 Jackknife-after-Bootstrap

- 偏差和标准差的 Bootstrap 估计的方差可以使用 Jackknife 方法来估计.
- 注意到 $\hat{se}(\hat{\theta})$ 是 B 次 $\hat{\theta}$ 的 Bootstrap 重复统计量的样本标准差, 那么如果我们丢掉第 i 个样本, 则 Jackknife 算法就是对每个 i, 从剩余的 $n-1$ 个样本值中重采样 B

次, 来计算 $\hat{se}(\hat{\theta}_{(i)})$ (Bootstrap 过程), 即一个 Jackknife 重复. 最后我们得到

$$\hat{se}_{jack}(\hat{se}_B(\hat{\theta})) = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{se}_{B(i)}(\hat{\theta}) - \overline{\hat{se}_{B(\cdot)}(\hat{\theta})})^2}$$

其中 $\overline{\hat{se}_{B(\cdot)}(\hat{\theta})} = \frac{1}{n}\sum_{i=1}^{n}\hat{se}_{B(i)}(\hat{\theta})$. 即对每个 i, 我们将重复 Bootstrap 本身. 这当然是效率低下的, 庆幸的是有方法可以避免重复 Bootstrap.

■ *Jackknife-after-Bootstrap* 方法是对每个"leave-one-out" 的 Bootstrap 样本计算一个估计. 具体如下:

记 $x_i^* = (x_1^*, \ldots, x_n^*)$ 为一次 Bootstrap 抽样, $x_1^*, \cdots, x_B^*$ 表示样本大小为 B 的 Bootstrap 样本. 令 $J(i)$ 表示 Bootstrap 样本中不含 x_i 的那些样本指标; $B(i)$ 表示不含 x_i 的

Bootstrap 样本个数, 因此我们可以使用丢掉 $B-B(i)$ 个含有 x_i 的样本后其余的样本来计算一个 Jackknife 重复. 故标准差估计量的 Jackknife 估计为

$$\hat{se}_{jab}(\hat{se}_B(\hat{\theta})) = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{se}_{B(i)} - \overline{\hat{se}_{B(\cdot)}})^2},$$

其中

$$\hat{se}_{B(i)} = \sqrt{\frac{1}{B(i)}\sum_{j\in J(i)}[\hat{\theta}_{(j)} - \overline{\hat{\theta}_{(J(i))}}]^2},$$

$$\overline{\hat{\theta}_{(J(i))}} = \frac{1}{B(i)}\sum_{j\in J(i)}\hat{\theta}_{(j)}.$$

例 (Jackknife-after-Bootstrap) 对前例中标准差的 Bootstrap 估计 $\hat{se}_B(\hat{\theta})$, 使用 Jackknife-after-Bootstrap 方法估计其标准差.

↑Example

↓Example

↑Code

```
# initialize
data(patch, package = "bootstrap")
n <- nrow(patch)
y <- patch$y
z <- patch$z
B <- 2000
theta.b <- numeric(B)
# set up storage for the sampled indices
```

```r
 indices <- matrix(0, nrow = B, ncol = n)

# jackknife-after-bootstrap step 1: run the bootstrap
 for (b in 1:B) {
     i <- sample(1:n, size = n, replace = TRUE)
     y <- patch$y[i]
     z <- patch$z[i]
     theta.b[b] <- mean(y) / mean(z)
     #save the indices for the jackknife
     indices[b, ] <- i
     }

 #jackknife-after-bootstrap to est. se(se)
 se.jack <- numeric(n)
 for (i in 1:n) {
   #in i-th replicate omit all samples with x[i]
     keep <- (1:B)[apply(indices, MARGIN = 1,
                 FUN = function(k) {!any(k == i)})]
     se.jack[i] <- sd(theta.b[keep])
 }
```

```
print(sd(theta.b))
print(sqrt((n-1) * mean((se.jack - mean(se.jack))^2)))
```

↓Code

3.5 Bootstrap 置信区间

■ Bootstrap 方法也常用于构造目标参数的渐近置信区间, 常用方法包括标准正态 Bootstrap 置信区间, 基本的 Bootstrap 置信区间, Bootstrap 百分位数 (percentile) 置信区间和 Bootstrap t 置信区间.

3.5.1 Normal Interval

■ 标准正态 Bootstrap 置信区间是一种比较简单的方法. 假设 $\hat{\theta}$ 是参数 θ 的估计量, 以及估计量的标准差为 $se(\hat{\theta})$. 若 $\hat{\theta}$ 渐近到正态分布, 即

$$Z = \frac{\hat{\theta} - E\hat{\theta}}{se(\hat{\theta})}$$

渐近服从标准正态分布.

■ 若 $\hat{\theta}$ 为 θ 的无偏估计, 那么 θ 的一个渐近的 $100(1-\alpha)\%$ 标准正态 Bootstrap 置信区间为

$$\hat{\theta} \pm z_{\alpha/2}\hat{se}_B(\hat{\theta}),$$

其中 $z_{\alpha/2} = \Phi^{-1}(1-\alpha/2)$. 此区间容易计算, 但是有正态性假设或者 CLT 需成立. 以及 $\hat{\theta}$ 为 θ 的无偏估计.

3.5.2 Percentile Interval

■ 由形式

$$P(L \le \hat{\theta} \le U) = 1 - \alpha$$

知, 可以使用 Bootstrap 重复的样本百分位数来估计 L 和 U.

■ 而 $\hat{\theta}$ 为 θ 的估计, 因此就取 θ 的 $1-\alpha$ 置信区间上下界分别为 Bootstrap 重复的样本 $1-\alpha/2$ 百分位数 $\hat{\theta}^*_{[(B+1)(1-\alpha/2)]}$ 和 $\alpha/2$ 百分位数 $\hat{\theta}^*_{[(B+1)\alpha/2]}$.

■ Efron & Tibshirani 证明了百分位数区间相比于标准正态区间有着更好的理论覆盖率.

■ 下面我们还会介绍 *bias-corrected and accelerated*(BCa) 百分位数区间, 它是百分位数区间的一个修正

版本, 有着更好的理论性质以及在应用中有着更好的覆盖率.

3.5.3 Basic Interval

- 由

$$P(L < \hat{\theta} - \theta < U) = 1 - \alpha$$

在 $\hat{\theta} - \theta$ 的分布未知时, 由于 Bootstrap 重复 $\hat{\theta}^*$ 的样本分位数 $\hat{\theta}^*_{[(B+1)\alpha/2]}$ 和 $\hat{\theta}^*_{[(B+1)(1-\alpha/2)]}$ 满足

$$P(\hat{\theta}^*_{[(B+1)\alpha/2]} - \hat{\theta} \le \theta^* - \hat{\theta} \le \hat{\theta}^*_{[(B+1)(1-\alpha/2)]} - \hat{\theta}) \approx 1 - \alpha.$$

- 因此 $100(1-\alpha)\%$ 置信区间为

$$(2\hat{\theta} - \hat{\theta}^*_{[(B+1)(1-\alpha/2)]}, 2\hat{\theta} - \hat{\theta}^*_{[(B+1)\alpha/2]})$$

■ **boot**包里的函数**boot.ci**计算五种类型的置信区间: 基本的, 正态, 百分位数, 学生化和 BCa.

例 **patch**数据比值统计量的 Bootstrap 置信区间

↑Example

↓Example

本例说明如何使用**boot**和**boot.ci**函数得到正态, 基本的和百分位数 Bootstrap 置信区间. 下面的代码计算比值统计量的 95% 置信区间.

↑Code

```
library(boot)       #for boot and boot.ci
 data(patch, package = "bootstrap")
 theta.boot <- function(dat, ind) {
```

```
      #function to compute the statistic
      y <- dat[ind, 1]
      z <- dat[ind, 2]
      mean(y) / mean(z)
  }
  y <- patch$y
  z <- patch$z
  dat <- cbind(y, z)
boot.obj <- boot(dat, statistic = theta.boot, R = 2000)
  print(boot.obj)
  print(boot.ci(boot.obj,
                type = c("basic", "norm", "perc")))
```

↓Code

■ 注意当 $|\theta| < 0.2$ 时, 旧药和新药才被认为是等价的. 因此区间估计没有支持旧药和新药的等价性. 下面我们根据 Bootstrap 置信区间的定义计算置信区间, 和上面的结果相

对比.

↑Code

```
#calculations for bootstrap confidence intervals
  alpha <- c(.025, .975)
  #normal
 print(boot.obj$t0 + qnorm(alpha) * sd(boot.obj$t))
  #basic
  print(2*boot.obj$t0 -
      quantile(boot.obj$t, rev(alpha), type=1))
  #percentile
  print(quantile(boot.obj$t, alpha, type=6))
```

↓Code

↑Example

例 **patch** 数据中相关系数的 Bootstrap 置信区间

↓Example

对**law**数据, 计算相关统计量的 95% 置信区间.

↑Code

```
library(boot)
   data(law, package = "bootstrap")
   boot.obj <- boot(law, R = 2000,
    statistic = function(x, i){cor(x[i,1], x[i,2])})
  print(boot.ci(boot.obj, type=c("basic","norm","perc")))
```

↓Code

三种置信区间都包含了 $\rho = .76$(此时通过完整数据集**law82**计算的). 百分位数置信区间和正态置信区间的

差异在于样本相关系数的分布是不是靠近正态分布. 当统计量的分布很靠近正态时, 百分位数区间和正态区间就会一致.

3.5.4 The Bootstrap t interval

■ 即使当 $\hat{\theta}$ 的分布是正态分布, 且 $\hat{\theta}$ 为 θ 的无偏估计, $Z = (\hat{\theta} - \theta)/se(\hat{\theta})$ 的分布也不会一定是正态的, 这是因为我们估计了 $se(\hat{\theta})$.

■ 我们也不能说 Z 的分布是 t 分布, 因为 Bootstrap 估计 $\hat{se}(\hat{\theta})$ 的分布未知.

■ Bootstrap t 区间并没有使用 t 分布作为推断分布. 而是使用重采样方法得到一个"t 类型" 的统计量的分布.

■ 假设 $x = (x_1, \ldots, x_n)$ 为观测到得样本, 则 $100(1 -$

$\alpha)\%$ Bootstrap t 置信区间为

$$(\hat{\theta} - t^*_{1-\alpha/2}\hat{se}(\hat{\theta}), \hat{\theta} - t^*_{\alpha/2}\hat{se}(\hat{\theta}))$$

其中 $\hat{se}(\hat{\theta}), t^*_{\alpha/2}$ 和 $t^*_{1-\alpha/2}$ 由下面的方法计算:

■ **Bootstrap t 区间 (学生化的 Bootstrap 区间)**

1. 计算观测到得 $\hat{\theta}$.
2. 对每个重复, $b = 1, \cdots, B$:
   (a) 从 x 中有放回的抽样得到第 b 个样本 $x^{(b)} = (x_1^{(b)}, \ldots, x_n^{(b)})$.
   (b) 由第 b 个重采样样本计算 $\hat{\theta}^{(b)}$.
   (c) 计算标准差估计 $\hat{se}(\hat{\theta}^{(b)})$.(对每个 Bootstrap 样本 $x^{(b)}$, 再单独进行一个 Bootstrap 估计).

(d) 计算第 b 个重复下的"t" 统计量: $t^{(b)} = (\hat{\theta}^{(b)} - \hat{\theta})/\hat{se}(\hat{\theta}^{(b)})$.

3. 重复样本 $t^{(1)}, \ldots, t^{(B)}$ 的分布作为推断分布. 找出样本分位数 $t^*_{\alpha/2}$ 和 $t^*_{1-\alpha/2}$.

4. 计算 $\hat{se}(\hat{\theta})$, 即 Bootstrap 重复 $\{\hat{\theta}^{(b)}\}$ 的样本标准差.

5. 计算置信界 $(\hat{\theta} - t^*_{1-\alpha/2}\hat{se}(\hat{\theta}), \hat{\theta} - t^*_{\alpha/2}\hat{se}(\hat{\theta}))$.

■ Bootstrap t 区间的一个缺点是要再次使用 Bootstrap 方法得到标准差的估计 $\hat{se}(\hat{\theta}^{(b)})$. 这是在 Bootstrap 里面嵌套 Bootstrap. 若 $B = 1000$, 则 Bootstrap t 区间方法需要比别的方法 1000 倍的时间.

↑Example

例 (Bootstrap t 区间) 本例我们写一个函数来计算一元或者多元样本下 Bootstrap t 置信区间. 默认的置信水平为 95%, Bootstrap 重复数为 500, 估计标准差的重复次数默认为 100.

↓Example

↑Code

```
boot.t.ci <-
   function(x, B = 500, R = 100, level = .95, statistic){
       #compute the bootstrap t CI
       x <- as.matrix(x);  n <- nrow(x)
```

```
stat <- numeric(B); se <- numeric(B)
boot.se <- function(x, R, f) {
    #local function to compute the bootstrap
   #estimate of standard error for statistic f(x)
    x <- as.matrix(x); m <- nrow(x)
    th <- replicate(R, expr = {
        i <- sample(1:m, size = m, replace = TRUE)
        f(x[i, ])
        })
    return(sd(th))
}
for (b in 1:B) {
    j <- sample(1:n, size = n, replace = TRUE)
    y <- x[j, ]
    stat[b] <- statistic(y)
```

```
        se[b] <- boot.se(y, R = R, f = statistic)
    }
    stat0 <- statistic(x)
    t.stats <- (stat - stat0) / se
    se0 <- sd(stat)
    alpha <- 1 - level
  Qt <- quantile(t.stats, c(alpha/2, 1-alpha/2), type = 1)
    names(Qt) <- rev(names(Qt))
    CI <- rev(stat0 - Qt * se0)
}
```

↓Code

例:patch数据下比值统计量的 Bootstrap t 置信区间

↑Example

↓Example

程序如下:

↑Code

```
#boot package and patch data were loaded in Example 7.10
 #library(boot)       #for boot and boot.ci
 #data(patch, package = "bootstrap")
 dat <- cbind(patch$y, patch$z)
 stat <- function(dat) {
     mean(dat[, 1]) / mean(dat[, 2]) }
```

```
ci <- boot.t.ci(dat, statistic = stat, B=2000, R=200)
print(ci)
```

↓Code

3.5.5 BCa Interval

■ 对百分位数区间进行修正可以得到更好的 Bootstrap 置信区间, 其具有更好的理论性质和更好的实际覆盖率.

■ 对 $1-\alpha$ 置信区间, 使用两个因子来调整常用的 $\alpha/2$ 和 $1-\alpha/2$ 分位数: 一个偏差 (bias) 的修正, 一个偏度 (skewness) 的修正. 偏差的修正记为 b_0, 偏度或者" 加速" 修正记为 a. 更优的 Bootstrap 置信区间也常称为 BCa.

■ $100(1-\alpha)\%$ BCa 置信区间为: 先计算

$$\alpha_1 = \Phi\left(\hat{b}_0 + \frac{\hat{b}_0 + z_{\alpha/2}}{1 - \hat{a}(\hat{b}_0 + z_{\alpha/2})}\right),$$

$$\alpha_2 = \Phi\left(\hat{b}_0 + \frac{\hat{b}_0 + z_{1-\alpha/2}}{1 - \hat{a}(\hat{b}_0 + z_{1-\alpha/2})}\right)$$

其中 $z_\alpha = \Phi^{-1}(\alpha)$. $\hat{b}_0, \hat{a}$ 由下面的式子计算. 则 BCa 区间为

$$(\hat{\theta}^*_{\alpha_1}, \hat{\theta}^*_{\alpha_2}).$$

BCa 区间的下界和上界是 Bootstrap 重复的经验的 α_1 和 α_2 分位数.

■ 偏差修正因子实际上是测量 $\hat{\theta}$ 的重复 $\hat{\theta}^*$ 的中位数偏

差. 其估计为

$$\hat{b}_0 = \Phi^{-1}(\frac{1}{B}\sum_{b=1}^{B} I(\hat{\theta}^{(b)} < \hat{\theta})).$$

- 加速因子是从 Jackknife 重复中估计:

$$\hat{a} = \frac{\sum_{i=1}^{n}(\overline{\theta_{(\cdot)}} - \theta_{(i)})^3}{6\sum_{i=1}^{n}((\overline{\theta_{(\cdot)}} - \theta_{(i)})^2)^{3/2}}.$$

- $\hat{a}$ 之所以称为是加速因子, 是因为它估计的是相对于目标参数 θ, $\hat{\theta}$ 的标准差的变化率. 我们在使用标准正态 Bootstrap 置信区间时, 是假设方差为一个常数, 与 θ 无关. 但是很多时候方差都可能和 θ 有关. 加速因子的目的就是要考虑到估计量的方差可能会与目标参数有关, 因此对置信界进行调整.

BCa 方法的来源

假设 $\hat{\theta} \sim N\left(\theta - b_0\sigma_\theta, \sigma_\theta^2\right), \sigma_\theta = 1 + a\theta$ ，即允许偏差和方差是依赖 θ 的. 则 $\hat{\theta}$ 可以被表示为: $\hat{\theta} = \theta + \sigma_\theta\left(Z - b_0\right)$ 和 $1 + a\hat{\theta} = (1 + a\theta)\left(1 + a\left(Z - b_0\right)\right)$. 两边取对数得到:

$$\hat{\gamma} = \gamma + W$$

其中 $\hat{\gamma} = \log(1+a\hat{\theta}), \gamma = \log(1+a\theta), W = \log\left(1 + a\left(Z - b_0\right)\right)$.

根据观测 $\hat{\gamma}$, γ 的 $1-\alpha$ 置信区间为

$$\gamma \in \left(\hat{\gamma} - w_{1-\alpha/2}, \hat{\gamma} - w_{\alpha/2}\right)$$

变换回 θ 尺度下, 我们有:

$$\theta \in \left(\frac{\hat{\theta} - \left(z_{1-\alpha/2} - b_0\right)}{1 + a\left(z_{1-\alpha/2} - b_0\right)}, \frac{\hat{\theta} - \left(z_{\alpha/2} - b_0\right)}{1 + a\left(z_{\alpha/2} - b_0\right)}\right)$$

进一步计算后有:

$$\theta_{[\alpha]} = \hat{\theta} + \sigma_{\hat{\theta}} \frac{z_\alpha + b_0}{1 - a\left(z_\alpha + b_0\right)}$$

基于对 $\hat{\theta}$ 的假定:

$$F_{\hat{\theta}}(x) = \Phi((x-\theta)/\sigma_\theta + b_0))$$

因此 $\hat{\theta}^*$ 的 bootstrap CDF 为

$$F_{\hat{\theta}^*}(x) = \Phi((x-\hat{\theta})/\sigma_{\hat{\theta}} + b_0))$$

注意 b_0, a 分别满足 $P(\hat{\theta} \leq \theta) = \Phi(b_0)$ 和 $a = skew(l_{\hat{\theta}})/6$, 其中 $l_\theta = \partial f_\theta(x)/\partial\theta$ 为得分函数, 可以利用 bootstrap 方法进行估计.

■ BCa 的性质

BCa Bootstrap 置信区间有两个重要的理论性质:

- 一是**不变性**, 即若 θ 的置信区间为 $(\hat{\theta}^*_{\alpha_1}, \hat{\theta}^*_{\alpha_2})$, $g(\cdot)$ 为一一变换函数, 则 $g(\theta)$ 得置信区间为 $(g(\hat{\theta}^*_{\alpha_1}), g(\hat{\theta}^*_{\alpha_2}))$.
- 另外一个性质是**二阶精确性**, 即误差以 $1/n$ 的速度趋于 0.

■ Bootstrap t 置信区间是二阶精确的, 但是不具有不变性. Bootstrap 百分位数区间有不变性, 但是不是二阶精确的; 标准正态置信区间既没有不变性, 也没有二阶精确性.

例: BCa Bootstrap 置信区间

↑Example
↓Example

本例是写一个函数来计算 BCa Bootstrap 置信区间.

↑Code

```
boot.BCa <-
function(x, th0, th, stat, conf = .95) {
    # bootstrap with BCa bootstrap confidence interval
    # th0 is the observed statistic
    # th is the vector of bootstrap replicates
    # stat is the function to compute the statistic
    x <- as.matrix(x)
    n <- nrow(x) #observations in rows
```

```
N <- 1:n
alpha <- (1 + c(-conf, conf))/2
zalpha <- qnorm(alpha)
# the bias correction factor
z0 <- qnorm(sum(th < th0) / length(th))
# the acceleration factor (jackknife est.)
th.jack <- numeric(n)
for (i in 1:n) {
    J <- N[1:(n-1)]
    th.jack[i] <- stat(x[-i, ], J)
}
L <- mean(th.jack) - th.jack
a <- sum(L^3)/(6 * sum(L^2)^1.5)
# BCa conf. limits
adj.alpha <- pnorm(z0 + (z0+zalpha)/(1-a*(z0+zalpha)))
```

```
    limits <- quantile(th, adj.alpha, type=6)
    return(list("est"=th0, "BCa"=limits))
}
```

↓Code

↑Example

例 (BCa 置信区间) 计算**patch**数据中比值统计量的BCa 置信区间.

↓Example

↑Code

```
n <- nrow(patch)
```

```
B <- 2000
y <- patch$y
z <- patch$z
x <- cbind(y, z)
theta.b <- numeric(B)
theta.hat <- mean(y) / mean(z)
#bootstrap
for (b in 1:B) {
    i <- sample(1:n, size = n, replace = TRUE)
    y <- patch$y[i]
    z <- patch$z[i]
    theta.b[b] <- mean(y) / mean(z)
    }
#compute the BCa interval
stat <- function(dat, index) {
  mean(dat[index, 1]) / mean(dat[index, 2])  }
boot.BCa(x, th0 = theta.hat, th = theta.b, stat = stat)
```

↓Code

在结果中, $\alpha/2 = 0.025$ 和 $1 - \alpha/2 = 0.975$ 被分别调整为

0.0334 和 0.982.

例使用函数**boot.ci**计算上例中的 BCa Bootstrap 置信区间

↑Example

↓Example

```
#using x from previous example
 boot.obj <- boot(x, statistic = stat, R=2000)
 boot.ci(boot.obj, type=c("perc", "bca"))
```

↑Code

↓Code

此例中也计算了百分位数置信区间.

3.6 Bootstrap 与回归

- 线性回归模型

$$y_j = x_j'\beta + \epsilon_j, \qquad \epsilon_j \sim (0, \sigma^2) \Leftrightarrow Y = X\beta + \epsilon$$

  - β 的 LSE$\hat{\beta} = (X'X)^{-1}X'Y$, 其方差 $Var(\hat{\beta}) = \sigma^2(X'X)^{-1}$
  - 残差 $e = Y - X\hat{\beta} = (I - X(X'X)X')Y = (I - H)Y$, $var(e) = \sigma^2(I - H)$

- 基于模型的重抽样:

$$y_j^* = x_j'\hat{\beta} + \hat{\epsilon}_j^*, \qquad \epsilon_j^* \sim ECDF(e_1 - \bar{e}, \ldots, e_n - \bar{e})$$

其中 $\bar{e} = \frac{1}{n}\sum e_i$.

- 这种方法固定设计阵 X, 对残差进行重抽样, 因而依赖于模型是正确的.

- 利用重抽样数据 (x_j, y_j^*) 进行最小二乘估计, 得到 $\hat{\beta}^* = (X'X)^{-1}X'Y^*$, 重复该过程, 得到充分多的 $\hat{\beta}^*$, 进而得到 β 的置信区间.

■ 基于观测个体的重抽样

$$(x_j, y_j)^* \sim ECDF((x_1, y_1), \ldots, (x_n, y_n))$$

- 设计阵 X 发生变化, 如果估计对比其敏感, 则会出现问题

- n 一般需要较大

3.7 Bootstrap 与假设检验

■ 假设 $H_0 : \theta \in \Theta_0 \leftrightarrow H_1 : \theta \notin \Theta_0$, T 为检验统计量且较大时候拒绝零假设, p 值为

$$p = P(T(X) \geq T(x)|H_0) = \sup_{\theta \in \Theta_0} P(T(X) \geq T(x))$$

■ 使用 Bootstrap 方法估计 p 值

- 在 H_0 下得到总体分布的估计 $\hat{F}_0$
- 在 $\hat{F}_0$ 之下进行抽样, 得到 $T^* = T(x^*), X^* \sim \hat{F}_0$
- 估计 p 值为

$$\hat{p} = \frac{1}{B}\sum_{b=1}^{B} I(T(x^*) \geq T(x))$$

或者

$$\hat{p} = \frac{1}{B+1}[1 + \sum_{b=1}^{B} I(T(x^*) \geq T(x))]$$

3.8 Application: Cross Validation

交叉验证 (Cross Validation) 是一种分割数据方法, 其可以用来验证参数估计的稳健性, 分类算法的准确度, 模型的合理性等等. Jackknife 可以被视为是交叉验证的一种特例, 其主要用来估计偏差和估计量的标准差.

最简单的交叉验证方法是所谓的"holdout" 方法. 其将数据随机分为训练集 (training set) 和测试集 (testing set) 两个子集. 然后仅使用训练集样本进行建模, 然后通过测试

集来对模型进行评估. 其优点是 training/testing 比例不依赖于重复次数. 其缺点是依赖于数据的分割方式, 某些点可能从不进入到测试集中, 而某些点可能多次进入测试集. 这种方法呈现出"Mente Carlo" 波动性, 即随机分割不同, 结果会波动.

"K-fold" 交叉验证是对"holdout" 方法的改进, 其将数据分割为 K 个子集, 然后重复"holdout" 方法 K 次. 每次第 i 个子集被作为测试集来评估模型, 其余的 $K-1$ 个子集被作为训练集进行建模. 最后计算 K 次的平均误差. 其优点是对数据的分割方式依赖性不是很强, 每个点仅有一次在测试集中, 而在训练集中有 $K-1$ 次. 因此估计的方差会随着 K 的增加而减少. 缺点是计算的时间复杂度增加.

"leave-one-out" 交叉验证是"K-fold" 交叉验证的一个

特例 ($K = n$), 仅使用一个点作为测试集, 其余的点作为训练集. 其缺点是计算的时间复杂度可能会比较高.

例 18 模型选择问题 包**DAAG**里的**ironslag**数据描述了两种方法 (chemical, magnetic) 测量含铁量的 53 次结果. 散点图显示 chemical 和 magnetic 变量是正相关的, 但是关系可能不是线性的. 从散点图上可以看出, 二次多项式, 或者可能一个指数的, 或对数模型能更好的拟合数据.

本例我们使用交叉验证来进行模型选择. 通过交叉验证来估计模型的预测误差. 候选的模型有

1. 线性模型: $Y = \beta_0 + \beta_1 X + e$.
2. 二次的: $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + e$.
3. 指数: $log(Y) = log(\beta_0) + \beta_1 X + e$.
4. log-log: $log(Y) = \beta_0 + \beta_1 log(X) + e$.

四种模型的参数估计程序如下

↑Code

```
par(ask = TRUE)

library(DAAG); attach(ironslag)
a <- seq(10, 40, .1)     #sequence for plotting fits

L1 <- lm(magnetic ~ chemical)
plot(chemical, magnetic, main="Linear", pch=16)
yhat1 <- L1$coef[1] + L1$coef[2] * a
lines(a, yhat1, lwd=2)

L2 <- lm(magnetic ~ chemical + I(chemical^2))
plot(chemical, magnetic, main="Quadratic", pch=16)
```

```
yhat2 <- L2$coef[1] + L2$coef[2] * a + L2$coef[3] * a^2
lines(a, yhat2, lwd=2)

L3 <- lm(log(magnetic) ~ chemical)
plot(chemical, magnetic, main="Exponential", pch=16)
logyhat3 <- L3$coef[1] + L3$coef[2] * a
yhat3 <- exp(logyhat3)
lines(a, yhat3, lwd=2)

L4 <- lm(log(magnetic) ~ log(chemical))
plot(log(chemical), log(magnetic), main="Log-Log", pch=16)
logyhat4 <- L4$coef[1] + L4$coef[2] * log(a)
lines(log(a), logyhat4, lwd=2)
```

↓Code

然后我们使用交叉验证来对每个模型的预测误差进行估计.

算法如下

1. 对 $k=1,\dots,n$, 令 (x_k, y_k) 为检验样本, 使用其余样本对模型参数进行估计. 然后计算预测误差.
   (a) 使用其余的样本对模型进行拟合.
   (b) 计算预测值: $\hat{y}_k = \hat{\beta}_0 + \beta_1 x_k$.
   (c) 计算预测误差: $e_k = y_k - \hat{y}_k$.
2. 计算均方预测误差: $\sigma_e^2 = \frac{1}{n}\sum_{i=1}^n e_k^2$.

计算程序如下

↑Code

```
n <- length(magnetic)
e1 <- e2 <- e3 <- e4 <- numeric(n)
```

```
# for n-fold cross validation
# fit models on leave-one-out samples
for (k in 1:n) {
    y <- magnetic[-k]
    x <- chemical[-k]

    J1 <- lm(y ~ x)
    yhat1 <- J1$coef[1] + J1$coef[2] * chemical[k]
    e1[k] <- magnetic[k] - yhat1

    J2 <- lm(y ~ x + I(x^2))
    yhat2 <- J2$coef[1] + J2$coef[2] * chemical[k] +
            J2$coef[3] * chemical[k]^2
    e2[k] <- magnetic[k] - yhat2
```

```
    J3 <- lm(log(y) ~ x)
    logyhat3 <- J3$coef[1] + J3$coef[2] * chemical[k]
    yhat3 <- exp(logyhat3)
    e3[k] <- magnetic[k] - yhat3

    J4 <- lm(log(y) ~ log(x))
  logyhat4 <- J4$coef[1] + J4$coef[2] * log(chemical[k])
    yhat4 <- exp(logyhat4)
    e4[k] <- magnetic[k] - yhat4
}
c(mean(e1^2), mean(e2^2), mean(e3^2), mean(e4^2))
```

↓Code

结果表明二次多项式回归的预测误差最小. 我们可以使

用**plot(L2)**进行模型诊断.