Package ‘bootstrap’

June 17, 2019

Version  2019.6
Date     2019-06-15
Title    Functions for the Book ``An Introduction to the Bootstrap"
Author   S original, from StatLib, by Rob Tibshirani. R port by
          Friedrich Leisch.
Maintainer Scott Kostyshak <scott.kostyshak@gmail.com>
Depends  stats, R (>= 2.10.0)
LazyData TRUE
Description Software (bootstrap, cross-validation, jackknife) and data
      for the book ``An Introduction to the Bootstrap'' by B. Efron and
      R. Tibshirani, 1993, Chapman and Hall. This package is
      primarily provided for projects already based on it, and for
      support of the book. New projects should preferentially use the
      recommended package ``boot``.
License  BSD_3_clause + file LICENSE
URL      https://gitlab.com/scottkosty/bootstrap
BugReports https://gitlab.com/scottkosty/bootstrap/issues
NeedsCompilation yes
Repository CRAN
Date/Publication 2019-06-17 09:40:08 UTC

\textbf{R topics documented:}

\texttt{abcnon} \dotfill 2
\texttt{abcpar} \dotfill 3
\texttt{bcanon} \dotfill 5
\texttt{bootpred} \dotfill 6
\texttt{bootstrap} \dotfill 7
\texttt{boott} \dotfill 9
\texttt{cell} \dotfill 11
\texttt{cholost} \dotfill 12
abcnon

Nonparametric ABC Confidence Limits

Description

See Efron and Tibshirani (1993) for details on this function.

Usage

```r
abcnon(x, tt, epsilon=0.001,
       alpha=c(0.025, 0.05, 0.1, 0.16, 0.84, 0.9, 0.95, 0.975))
```

Arguments

- `x`: the data. Must be either a vector, or a matrix whose rows are the observations
- `tt`: function defining the parameter in the resampling form `tt(p, x)`, where `p` is the vector of proportions and `x` is the data
- `epsilon`: optional argument specifying step size for finite difference calculations
- `alpha`: optional argument specifying confidence levels desired

Value

list with following components

- `limits`: The estimated confidence points, from the ABC and standard normal methods
- `stats`: list consisting of `t0=observed value of tt, sighat=infinitesimal jackknife estimate of standard error of tt, bhat=estimated bias`
- `constants`: list consisting of `a=acceleration constant, z0=bias adjustment, cq=curvature component`
abcpar

\texttt{tt.inf} \quad \text{approximate influence components of} \ \tt
\texttt{pp} \quad \text{matrix whose rows are the resampling points in the least favourable family. The abc confidence points are the function tt evaluated at these points}
\texttt{call} \quad \text{The deparsed call}

References


Examples

\begin{verbatim}
# compute abc intervals for the mean
x <- rnorm(10)
theta <- function(p, x) {sum(p * x) / sum(p)}
results <- abcnon(x, theta)

# compute abc intervals for the correlation
x <- matrix(rnorm(20), ncol=2)
theta <- function(p, x)
{
  x1m <- sum(p * x[, 1]) / sum(p)
  x2m <- sum(p * x[, 2]) / sum(p)
  num <- sum(p * (x[, 1] - x1m) * (x[, 2] - x2m))
  den <- sqrt(sum(p * (x[, 1] - x1m)^2)) * sqrt(sum(p * (x[, 2] - x2m)^2))
  return(num/den)
}
results <- abcnon(x, theta)
\end{verbatim}

\begin{verbatim}
abcpar

\emph{Parametric ABC Confidence Limits}

\end{verbatim}

Description

See Efron and Tibshirani (1993) for details on this function.

Usage

\begin{verbatim}
abcpar(y, tt, S, etahat, mu, n=rep(1,length(y)), lambda=0.001,
       alpha=c(0.025, 0.05, 0.1, 0.16))
\end{verbatim}
Arguments

- \( y \) vector of data
- \( tt \) function of expectation parameter \( \mu \) defining the parameter of interest
- \( S \) maximum likelihood estimate of the covariance matrix of \( x \)
- \( \etahat \) maximum likelihood estimate of the natural parameter \( \eta \)
- \( \mu \) function giving expectation of \( x \) in terms of \( \eta \)
- \( n \) optional argument containing denominators for binomial (vector of length \( \text{length}(x) \))
- \( \lambda \) optional argument specifying step size for finite difference calculation
- \( \alpha \) optional argument specifying confidence levels desired

Value

- list with the following components
  - \( \text{call} \) the call to \( abcpar \)
  - \( \text{limits} \) The nominal confidence level, ABC point, quadratic ABC point, and standard normal point.
  - \( \text{stats} \) list consisting of observed value of \( tt \), estimated standard error and estimated bias
  - \( \text{constants} \) list consisting of \( a=\text{acceleration constant}, z0=\text{bias adjustment}, cq=\text{curvature component} \)
  - \( \text{asym.05} \) asymmetry component

References


Examples

```r
# binomial
# x is a p-vector of successes, n is a p-vector of
# number of trials
## Not run:
S <- matrix(0,nrow=p,ncol=p)
S[row(S)==col(S)] <- x*(1-x)/n
mu <- function(eta,n) n/(1+exp(eta))
etahat <- log(x/(n-x))
#suppose p=2 and we are interested in mu2-mu1
tt <- function(mu)[mu[2]-mu[1]]
x <- c(2,4); n <- c(12,12)
a <- abcpar(x, tt, S, etahat, n)
## End(Not run)
```
**bcanon**  

---  

**Nonparametric BCa Confidence Limits**

**Description**

See Efron and Tibshirani (1993) for details on this function.

**Usage**

```r
bcanon(x, nboot, theta, ...,  
   alpha=c(0.025, 0.05, 0.1, 0.16, 0.84, 0.9, 0.95, 0.975))
```

**Arguments**

- `x`: a vector containing the data. To bootstrap more complex data structures (e.g. bivariate data) see the last example below.
- `nboot`: number of bootstrap replications
- `theta`: function defining the estimator used in constructing the confidence points
- `...`: additional arguments for `theta`
- `alpha`: optional argument specifying confidence levels desired

**Value**

list with the following components

- `confpoints`: estimated BCa confidence limits
- `z0`: estimated bias correction
- `acc`: estimated acceleration constant
- `u`: jackknife influence values
- `call`: The deparsed call

**References**


Examples

# bca limits for the mean
# (this is for illustration;
# since "mean" is a built in function,
# bcanon(x,100,mean) would be simpler!)
x <- rnorm(20)
theta <- function(x){mean(x)}
results <- bcanon(x,100,theta)

# To obtain bca limits for functions of more
# complex data structures, write theta
# so that its argument x is the set of observation
# numbers and simply pass as data to bcanon
# the vector 1,2,...n.
# For example, find bca limits for
# the correlation coefficient from a set of 15 data pairs:
xdata <- matrix(rnorm(30),ncol=2)
n <- 15
theta <- function(x,xdata){ cor(xdata[,1],xdata[,2]) }
results <- bcanon(1:n,100,theta,xdata)

bootpred

**Bootstrap Estimates of Prediction Error**

Description

See Efron and Tibshirani (1993) for details on this function.

Usage

```r
bootpred(x,y,nboot,theta.fit,theta.predict,err.meas,...)
```

Arguments

- `x` a matrix containing the predictor (regressor) values. Each row corresponds to an observation.
- `y` a vector containing the response values
- `nboot` the number of bootstrap replications
- `theta.fit` function to be cross-validated. Takes `x` and `y` as an argument. See example below.
- `theta.predict` function producing predicted values for `theta.fit`. Arguments are a matrix `x` of predictors and fit object produced by `theta.fit`. See example below.
- `err.meas` function specifying error measure for a single response `y` and prediction `yhat`. See examples below
- `...` any additional arguments to be passed to `theta.fit`
Value

list with the following components

app.err the apparent error rate - that is, the mean value of err.meas when theta.fit is applied to x and y, and then used to predict y.

optim the bootstrap estimate of optimism in app.err. A useful estimate of prediction error is app.err+optim

err.632 the ".632" bootstrap estimate of prediction error.

call The deparsed call

References


Examples

# bootstrap prediction error estimation in least squares
# regression
x <- rnorm(85)
y <- 2*x + 5*rnorm(85)
theta.fit <- function(x,y){lsfit(x,y)}
theta.predict <- function(fit,x){
  cbind(1,x)*fit$coef
}
sq.err <- function(y,yhat) { (y-yhat)^2}
results <- bootpred(x,y,20,theta.fit,theta.predict,
  err.meas=sq.err)

# for a classification problem, a standard choice
# for err.meas would simply count up the
# classification errors:
miss.clas <- function(y,yhat){ 1*(yhat!=y)}
# with this specification, bootpred estimates
# misclassification rate
Arguments

- **x**: a vector containing the data. To bootstrap more complex data structures (e.g. bivariate data) see the last example below.
- **nboot**: The number of bootstrap samples desired.
- **theta**: function to be bootstrapped. Takes x as an argument, and may take additional arguments (see below and last example).
- **...**: any additional arguments to be passed to theta
- **func**: (optional) argument specifying the functional the distribution of thetahat that is desired. If func is specified, the jackknife after-bootstrap estimate of its standard error is also returned. See example below.

Value

list with the following components:

- **thetastar**: the nboot bootstrap values of theta
- **func.thetastar**: the functional func of the bootstrap distribution of thetastar, if func was specified
- **jack.boot.val**: the jackknife-after-bootstrap values for func, if func was specified
- **jack.boot.se**: the jackknife-after-bootstrap standard error estimate of func, if func was specified
- **call**: the deparsed call

References


Examples

```r
# 100 bootstraps of the sample mean
# (this is for illustration; since "mean" is a built-in function, bootstrap(x,100,mean) would be simpler!)
x <- rnorm(20)
theta<- function(x){mean(x)}

results <- bootstrap(x,100,theta)

# as above, but also estimate the 95th percentile
# of the bootstrap dist'n of the mean, and
# its jackknife-after-bootstrap standard error
perc95 <- function(x){quantile(x, .95)}
```
results <- bootstrap(x,100,theta, func=perc95)

# To bootstrap functions of more complex data structures,
# write theta so that its argument x
# is the set of observation numbers
# and simply pass as data to bootstrap the vector 1,2,...n.
# For example, to bootstrap
# the correlation coefficient from a set of 15 data pairs:
xdata <- matrix(rnorm(30),ncol=2)
n <- 15
theta <- function(x,xdata) { cor(xdata[x,1],xdata[x,2]) } 
results <- bootstrap(1:n,20,theta,xdata)

boott  

---

**boott**  

**Bootstrap-t Confidence Limits**

---

**Description**

See Efron and Tibshirani (1993) for details on this function.

**Usage**

```r
boott(x,theta, ..., sdfun=sdfunboot, nbootsd=25, nboott=200,
      VS=FALSE, v.nbootg=100, v.nboott=25, v.nboott=200,
      perc=c(.001,.01,.025,.05,.10,.50,.90,.95,.975,.99,.999))
```

**Arguments**

- **x**  
  a vector containing the data. Nonparametric bootstrap sampling is used. To
  bootstrap from more complex data structures (e.g. bivariate data) see the last
  example below.

- **theta**  
  function to be bootstrapped. Takes x as an argument, and may take additional
  arguments (see below and last example).

- **...**  
  any additional arguments to be passed to theta

- **sdfun**  
  optional name of function for computing standard deviation of theta based on
  data x. Should be of the form: sdmean <- function(x,nbootsd,theta,...) where
  nbootsd is a dummy argument that is not used. If theta is the mean, for
  example, sdmean <- function(x,nbootsd,theta,...) 
  {sqrt(var(x)/length(x))}. If sdfun is missing, then boott uses an inner
  bootstrap loop to estimate the standard deviation of theta(x)

- **nbootsd**  
  The number of bootstrap samples used to estimate the standard deviation of
  theta(x)

- **nboott**  
  The number of bootstrap samples used to estimate the distribution of the boot-
  strap T statistic. 200 is a bare minimum and 1000 or more is needed for reli-
  able α% confidence points, α > .95 say. Total number of bootstrap samples is
  nboott*nbootsd.
VS

If TRUE, a variance stabilizing transformation is estimated, and the interval is constructed on the transformed scale, and then is mapped back to the original theta scale. This can improve both the statistical properties of the intervals and speed up the computation. See the reference Tibshirani (1988) given below. If FALSE, variance stabilization is not performed.

v.nbootg

The number of bootstrap samples used to estimate the variance stabilizing transformation g. Only used if VS=TRUE.

v.nbootsd

The number of bootstrap samples used to estimate the standard deviation of theta(x). Only used if VS=TRUE.

v.nboott

The number of bootstrap samples used to estimate the distribution of the bootstrap T statistic. Only used if VS=TRUE. Total number of bootstrap samples is v.nbootg*v.nbootsd + v.nboott.

perc

Confidence points desired.

Value

list with the following components:

cnfpoints Estimated confidence points
theta, g theta and g are only returned if VS=TRUE was specified. (theta[i],g[i]), i=1,length(theta) represents the estimate of the variance stabilizing transformation g at the points theta[i].
call The deparsed call

References


Examples

# estimated confidence points for the mean
x <- rchisq(20,1)
theta <- function(x)(mean(x))
results <- boot(x,theta)
# estimated confidence points for the mean,
# using variance-stabilization bootstrap-T method
results <- boot(x,theta,VS=TRUE)
results$confpoints # gives confidence points
# plot the estimated var stabilizing transformation
plot(results$theta,results$g)
# use standard formula for stand dev of mean
# rather than an inner bootstrap loop
sdmean <- function(x, ...)
  (sqrt(var(x)/length(x)))
results <- boot(x, theta, sdfun = sdmean)

# To bootstrap functions of more complex data structures,
# write theta so that its argument x
# is the set of observation numbers
# and simply pass as data to boot the vector 1, 2, \ldots n.
# For example, to bootstrap
# the correlation coefficient from a set of 15 data pairs:
xdata <- matrix(rnorm(30), ncol = 2)
n <- 15
theta <- function(x, xdata) {
  cor(xdata[, 1], xdata[, 2])
}
results <- boot(1:n, theta, xdata)

---

cell

### Cell Survival data

**Description**

Data on cell survival under different radiation doses.

**Usage**

data(cell)

**Format**

A data frame with 14 observations on the following 2 variables.

- **dose** a numeric vector, unit rads/100
- **log.surv** a numeric vector, (natural) logarithm of proportion

**Details**

There are regression situations where the covariates are more naturally considered fixed rather than random. This cell survival data are an example. A radiologist has run an experiment involving 14 bacterial plates. The plates where exposed to different doses of radiation, and the proportion of surviving cells measured. Greater doses lead to smaller survival proportions, as would be expected. The investigator expressed some doubt as to the validity of observation 13.

So there is some interest as to the influence of observation 13 on the conclusions.

Two different theoretical models as to radiation damage were available, one predicting a linear regression,

\[
\mu_i = \text{E}(y_i | z_i) = \beta_1 z_i
\]

and the other predicting a quadratic regression,

\[
\mu_i = \text{E}(y_i | z_i) = \beta_1 z_i + \beta_2 z_i^2
\]

Hypothesis tests on \(\beta_2\) is of interest.
**Source**


**Examples**

```r
plot(cell[,2:1],pch=c(rep(1,12),17,1),
     col=c(rep("black",12),"red","black"),
     cex=c(rep(1,12),2,1))
```

---

**The Cholostyramine Data**

**Description**

$n = 164$ men took part in an experiment to see if the drug cholostyramine lowered blood cholesterol levels. The men were supposed to take six packets of cholostyramine per day, but many actually took much less.

**Usage**

```r
data(cholost)
```

**Format**

A data frame with 164 observations on the following 2 variables.

- `z` Compliance, a numeric vector
- `y` Improvement, a numeric vector

**Details**

In the book, this is used as an example for curve fitting, with two methods, traditional least-squares fitting and modern `loess`. In the book is considered linear and polynomial models for the dependence of Improvement upon Compliance.

**Source**


**Examples**

```r
str(cholost)
summary(cholost)
plot(y ~ z, data=cholost, xlab="Compliance", ylab="Improvement")
abline(lm(y ~ z, data=cholost), col="red")
```
crossval

K-fold Cross-Validation

Description

See Efron and Tibshirani (1993) for details on this function.

Usage

crossval(x, y, theta.fit, theta.predict, ..., ngroup=n)

Arguments

x a matrix containing the predictor (regressor) values. Each row corresponds to an observation.
y a vector containing the response values
theta.fit function to be cross-validated. Takes x and y as an argument. See example below.
theta.predict function producing predicted values for theta.fit. Arguments are a matrix x of predictors and fit object produced by theta.fit. See example below.
... any additional arguments to be passed to theta.fit
ngroup optional argument specifying the number of groups formed. Default is ngroup=sample size, corresponding to leave-one out cross-validation.

Value

list with the following components

cv.fit The cross-validated fit for each observation. The numbers 1 to n (the sample size) are partitioned into ngroup mutually disjoint groups of size "leave.out". leave.out, the number of observations in each group, is the integer part of n/ngroup. The groups are chosen at random if ngroup < n. (If n/leave.out is not an integer, the last group will contain > leave.out observations). Then theta.fit is applied with the kth group of observations deleted. for k=1, 2, ngroup. Finally, the fitted value is computed for the kth group using theta.predict.
ngroup The number of groups
leave.out The number of observations in each group
groups A list of length ngroup containing the indices of the observations in each group. Only returned if leave.out > 1.
call The deparsed call

References

Examples

```r
# cross-validation of least squares regression
# note that crossval is not very efficient, and being a
general purpose function, it does not use the
# Sherman-Morrison identity for this special case
x <- rnorm(85)
y <- 2*x +.5*rnorm(85)
theta.fit <- function(x,y) {lsfit(x,y)}
theta.predict <- function(fit,x){
    cbind(1,x) %*% fit$coef
}
results <- crossval(x,y,theta.fit,theta.predict,n=6)
```

---

**diabetes**

**Blood Measurements on 43 Diabetic Children**

**Description**

Measurements on 43 diabetic children of log-Cpeptide (a blood measurement) and age (in years). Interest is predicting the blood measurement from age.

**Usage**

```r
data(diabetes)
```

**Format**

A data frame with 43 observations on the following 3 variables.

- **obs** a numeric vector
- **age** a numeric vector
- **logCpeptide** a numeric vector

**Source**


**Examples**

```r
plot(logCpeptide ~ age, data=diabetes)
```
**Description**

The hormone data. Amount in milligrams of anti-inflammatory hormone remaining in 27 devices, after a certain number of hours (hrs) of wear.

**Usage**

```r
data(hormone)
```

**Format**

A data frame with 27 observations on the following 3 variables.

- **Lot**: a character vector
- **hrs**: a numeric vector
- **amount**: a numeric vector

**Details**

The hormone data. Amount in milligrams of anti-inflammatory hormone remaining in 27 devices, after a certain number of hours (hrs) of wear. The devices were sampled from 3 different manufacturing lots, called A, B and C. Lot C looks like it had greater amounts of remaining hormone, but it also was worn the least number of hours.

The book uses this as an example for regression analysis.

**Source**


**Examples**

```r
str(hormone)
if(interactive())par(ask=TRUE)
with(hormone, stripchart(amount ~ Lot))
with(hormone, plot(amount ~ hrs, pch=Lot))
abline( lm(amount ~ hrs, data=hormone, col="red2"))
```
**jackknife**  

*Jackknife Estimation*

**Description**

See Efron and Tibshirani (1993) for details on this function.

**Usage**

```r
jackknife(x, theta, ...)
```

**Arguments**

- `x`: a vector containing the data. To jackknife more complex data structures (e.g., bivariate data) see the last example below.
- `theta`: function to be jackknifed. Takes `x` as an argument, and may take additional arguments (see below and last example).
- `...`: any additional arguments to be passed to `theta`

**Value**

list with the following components

- `jack.se`: The jackknife estimate of standard error of `theta`. The leave-one out jackknife is used.
- `jack.bias`: The jackknife estimate of bias of `theta`. The leave-one out jackknife is used.
- `jack.values`: The n leave-one-out values of `theta`, where n is the number of observations. That is, `theta` applied to `x` with the 1st observation deleted, `theta` applied to `x` with the 2nd observation deleted, etc.
- `call`: The deparsed call

**References**


**Examples**

```r
# jackknife values for the sample mean
# (this is for illustration; # since "mean" is a
# built in function, jackknife(x,mean) would be simpler!)
x <- rnorm(20)
theta <- function(x){mean(x)}
results <- jackknife(x,theta)
```
# To jackknife functions of more complex data structures,
# write theta so that its argument x
# is the set of observation numbers
# and simply pass as data to jackknife the vector 1,2,...,n.
# For example, to jackknife
# the correlation coefficient from a set of 15 data pairs:

```r
xdata <- matrix(rnorm(30), ncol=2)
n <- 15
theta <- function(x, xdata) { cor(xdata[x, 1], xdata[x, 2]) }
results <- jackknife(1:n, theta, xdata)
```

---

### Description

The law school data. A random sample of size \( n = 15 \) from the universe of 82 USA law schools. Two measurements: LSAT (average score on a national law test) and GPA (average undergraduate grade-point average). `law82` contains data for the whole universe of 82 law schools.

### Usage

```r
data(law)
```

### Format

A data frame with 15 observations on the following 2 variables.

- **LSAT**  a numeric vector
- **GPA**   a numeric vector

### Details

In the book for which this package is support software, this example is used to bootstrap the correlation coefficient.

### Source


### See Also

`law82`.  

---

`law`
Examples

str(law)
if(interactive())par(ask=TRUE)
plot(law)
theta <- function(ind) cor(law[ind,1], law[ind,2])
theta(1:15) # sample estimate
law.boot <- bootstrap(1:15, 2000, theta)
sd(law.boot$thetastar) # bootstrap standard error
hist(law.boot$thetastar)
# bootstrap t confidence limits for the correlation coefficient:
theta <- function(ind) cor(law[ind,1], law[ind,2])
boott(1:15, theta, VS=FALSE)$confpoints
boott(1:15, theta, VS=TRUE)$confpoints
# Observe the difference! See page 162 of the book.
# abcnon(as.matrix(law), function(p,x) cov.wt(x, p, cor=TRUE)$cor[1,2] )$limits
# The above cannot be used, as the resampling vector can take negative values!

law82

Data for Universe of USA Law Schools

Description

This is the universe of 82 USA law schools for which the data frame law provides a sample of size 15. See documentation for law for more details.

Usage

data(law82)

Format

A data frame with 82 observations on the following 3 variables.

School a numeric vector
LSAT a numeric vector
GPA a numeric vector

Source


Examples

plot(law82[,2:3])
cor(law82[,2:3])
Description

Five sets of levels of luteinizing hormone for each of 48 time periods

Usage

data(lutenhorm)

Format

A data frame with 48 observations on the following 5 variables.

- **V1**: a numeric vector
- **V2**: a numeric vector
- **V3**: a numeric vector
- **V4**: a numeric vector
- **V5**: a numeric vector

Details

Five sets of levels of luteinizing hormone for each of 48 time periods, taken from Diggle (1990). These are hormone levels measured on a healthy woman in 10 minute intervals over a period of 8 hours. The luteinizing hormone is one of the hormones that orchestrate the menstrual cycle and hence it is important to understand its daily variation.

This is a time series. The book gives only one time series, which correspond to V4. I don’t know what are the other four series, the book doesn’t mention them. They could be block bootstrap replicates?

Source


Examples

str(lutenhorm)
matplot(lutenhorm)
Description

A small randomized experiment were done with 16 mouse, 7 to treatment group and 9 to control group. Treatment was intended to prolong survival after a test surgery.

Usage

data(mouse.c)

Format

The format is: num [1:9] 52 104 146 10 50 31 40 27 46

Details

The treatment group is is dataset mouse.t. mouse.c is the control group. The book uses this example to illustrate bootstrapping a sample mean. Measurement unit is days of survival following surgery.

Source


Examples

str(mouse.c)
if(interactive())par(ask=TRUE)
stripchart(list(treatment=mouse.t, control=mouse.c))
cat("bootstrapping the difference of means, treatment - control:\n")
cat("bootstrapping is done independently for the two groups:"
) mouse.boot.c <- bootstrap(mouse.c, 2000, mean)
mouse.boot.t <- bootstrap(mouse.t, 2000, mean)
mouse.boot.diff <- mouse.boot.t$thetastar - mouse.boot.c$thetastar
hist(mouse.boot.diff)
abline(v=0, col="red2")
.sd(mouse.boot.diff)
**mouse.t**

*Experiment with mouse*

**Description**

A small randomized experiment were done with 16 mouse, 7 to treatment group and 9 to control group. Treatment was intended to prolong survival after a test surgery.

**Usage**

`data(mouse.t)`

**Format**

The format is: `num [1:7] 94 197 16 38 99 141 23`

**Details**

The control group is dataset `mouse.c`. This dataset is the treatment group. The book uses this for exemplifying bootstrapping the sample mean. Measurement unit is days of survival following surgery.

**Source**


**Examples**

`str(mouse.t)`

`stripchart(list(treatment=mouse.t, control=mouse.c))`

---

**patch**

*The Patch Data*

**Description**

Eight subjects wore medical patches designed to infuse a naturally-occuring hormone into the blood stream.

**Usage**

`data(patch)`
Format

A data frame with 8 observations on the following 6 variables.

- **subject**: a numeric vector
- **placebo**: a numeric vector
- **oldpatch**: a numeric vector
- **newpatch**: a numeric vector
- **z**: a numeric vector, oldpatch - placebo
- **y**: a numeric vector, newpatch - oldpatch

Details

Eight subjects wore medical patches designed to infuse a certain naturally-occurring hormone into the bloodstream. Each subject had his blood levels of the hormone measured after wearing three different patches: a placebo patch, an "old" patch manufactured at an older plant, and a "new" patch manufactured at a newly opened plant.

The purpose of the study was to show **bioequivalence**. Patches from the old plant was already approved for sale by the FDA (food and drug administration). Patches from the new facility would not need a full new approval, if they could be shown bioequivalent to the patches from the old plant.

Bioequivalence was defined as

\[
\frac{|E(\text{new}) - E(\text{old})|}{E(\text{old}) - E(\text{placebo})} \leq 0.20
\]

The book uses this to investigate bias of ratio estimation.

Source


Examples

```r
c
theta <- function(ind){
  Y <- patch[ind,"y"]
  Z <- patch[ind,"z"]
  mean(Y)/mean(Z) }
patch.boot <- bootstrap(1:8, 2000, theta)
names(patch.boot)
hist(patch.boot$thetastar)
abline(v=-c(-0.2, 0.2), col="red2")
theta(1:8) # sample plug-in estimator
abline(v=theta(1:8), col="blue")
# The bootstrap bias estimate:
mean(patch.boot$thetastar) - theta(1:8)
sd(patch.boot$thetastar) # bootstrapped standard error
```
### Rainfall

**Rainfall Data**

<table>
<thead>
<tr>
<th>Description</th>
<th>Rainfall data. The yearly rainfall, in inches, in Nevada City, California, USA, 1873 through 1978. An example of time series data.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td><code>data(Rainfall)</code></td>
</tr>
<tr>
<td>Format</td>
<td>The format is: Time-Series [1:106] from 1873 to 1978: 80 40 65 46 68 32 58 60 61 60 ...</td>
</tr>
<tr>
<td>Examples</td>
<td><code>str(Rainfall)</code> <code>plot(Rainfall)</code></td>
</tr>
</tbody>
</table>

### scor

**Open/Closed Book Examination Data**

<table>
<thead>
<tr>
<th>Description</th>
<th>This is data form mardia, Kent and Bibby on 88 students who took examinations in 5 subjects. Some where with open book and other with closed book.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td><code>data(scor)</code></td>
</tr>
</tbody>
</table>
Details

The book uses this for bootstrap in principal component analysis.

Source


Examples

```r
str(scor)
if(interactive())par(ask=TRUE)
plot(scor)
# The parameter of interest (theta) is the fraction of variance explained
# by the first principal component.
# For principal components analysis svd is better numerically than
# eigen-decomposition, but for bootstrapping the latter is _much_ faster.
theta <- function(ind) {
  vals <- eigen(var(scor[ind,]), symmetric=TRUE, only.values=TRUE)$values
  vals[1] / sum(vals)
} 
scor.boot <- bootstrap(1:88, 500, theta) 
sd(scor.boot$thetastar) # bootstrap standard error
hist(scor.boot$thetastar)
abline(v=theta(1:88), col="red")
abline(v=mean(scor.boot$thetastar), col="blue")
```

spatial

Spatial Test Data

Description

Twenty-six neurologically impaired children have each taken two tests of spatial perception, called "A" and "B".

Usage

data(spatial)

Format

A data frame with 26 observations on the following 2 variables.

A  a numeric vector
B  a numeric vector

Details

In the book this is used as a test data set for bootstrapping confidence intervals.
**stamp**

**Source**


**Examples**

```r
str(spatial)
plot(spatial)
abline(0, 1, col="red2")
```

---

**Data on Thickness of Stamps**

**Description**

Thickness in millimeters of 485 postal stamps, printed in 1872. The stamp issue of that year was thought to be a "philatelic mixture", that is, printed on more than one type of paper. It is of historical interest to determine how many different types of paper were used.

**Usage**

```r
data(stamp)
```

**Format**

A data frame with 485 observations on the following variable.

**Thickness** Thickness in millimeters, a numeric vector

**Details**

In the book, this is used to exemplify determination of number of modes. It is also used for kernel density estimation.

**Note**

The main example in the book is on page 227. See also the CRAN package diptest for an alternative method.

**Source**


**Examples**

```r
summary(stamp)
with(stamp, {hist(Thickness); plot(density(Thickness), add=TRUE))
```
Tooth Strength Data

Description

Thirteen accident victims have had the strength of their teeth measured. It is desired to predict teeth strength from measurements not requiring destructive testing. Four such variables have been obtained for each subject, \((D1, D2)\) are difficult to obtain, \((E1, E2)\) are easy to obtain.

Usage

data(tooth)

Format

A data frame with 13 observations on the following 6 variables.

- **patient** a numeric vector
- **D1** a numeric vector
- **D2** a numeric vector
- **E1** a numeric vector
- **E2** a numeric vector
- **strength** a numeric vector

Details

Do the easy to obtain variables give as good prediction as the difficult to obtain ones?

Source


Examples

```r
str(tooth)
mod.easy <- lm(strength ~ E1+E2, data=tooth)
summary(mod.easy)
mod.diffi <- lm(strength ~ D1+D2, data=tooth)
summary(mod.diffi)
if(interactive())par(ask=TRUE)
theta <- function(ind) {
  easy <- lm(strength ~ E1+E2, data=tooth, subset=ind)
  diffi <- lm(strength ~ D1+D2, data=tooth, subset=ind)
  (sum(resid(easy)^2) - sum(resid(diffi)^2))/13
}
tooth.boot <- bootstrap(1:13, 2000, theta)
hist(tooth.boot$thetastar)
```
abline(v=0, col="red2")
qqnorm(tooth.boot$thetastar)
qqline(tooth.boot$thetastar, col="red2")
Index

*Topic datasets

- cell, 11
- cholost, 12
- diabetes, 14
- hormone, 15
- law, 17
- law82, 18
- lutenhorm, 19
- mouse.c, 20
- mouse.t, 21
- patch, 21
- Rainfall, 23
- scor, 23
- spatial, 24
- stamp, 25
- tooth, 26

*Topic htest

- abcnon, 2
- abcpar, 3

*Topic nonparametric

- abcnon, 2
- bcanon, 5
- bootpred, 6
- bootstrap, 7
- boott, 9
- crossval, 13
- jackknife, 16
- hormone, 15
- jackknife, 16
- law, 17, 18
- law82, 17, 18
- loess, 12
- lutenhorm, 19
- mouse.c, 20, 21
- mouse.t, 20, 21
- patch, 21
- Rainfall, 23
- scor, 23
- spatial, 24
- stamp, 25
- tooth, 26

abcnon, 2
abcpar, 3

bcanon, 5
bootpred, 6
bootstrap, 7
boott, 9

cell, 11
cholost, 12
crossval, 13
diabetes, 14