Package 'bcaboot'

May 7, 2026

Title Bias Corrected Bootstrap Confidence Intervals

Version 0.2-3

VignetteBuilder knitr

Description Computation of bootstrap confidence intervals in an almost automatic fashion as described in Efron and Narasimhan (2020, <doi:10.1080/10618600.2020.1714633>).

Depends R (>= 3.5.0)

URL https://bnaras.github.io/bcaboot/, https://github.com/bnaras/bcaboot

BugReports https://github.com/bnaras/bcaboot/issues

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests glmnet, rmarkdown, knitr

Imports graphics, stats, utils

NeedsCompilation no

Author Bradley Efron [aut], Balasubramanian Narasimhan [aut, cre]

Maintainer Balasubramanian Narasimhan <naras@stat.stanford.edu>

Repository CRAN

Date/Publication 2021-05-09 05:10:19 UTC

Contents

---

bcaboot *Automatic Construction of Bootstrap Confidence Intervals*

Description

Bootstrap confidence intervals depend on three elements: (a) the cumulative distribution of the bootstrap replications, (b) the bias-correction, and (c) the acceleration number that measures the rate of change in the standard deviation of the estimate as the data changes. The first two of these depend only on the bootstrap distribution, and not how it is generated: parametrically or non-parametrically. Therefore, the only difference in a parametric bca analysis would lie in the nonparametric estimation of the acceleration, often a negligible error.

---

bcajack *Nonparametric bias-corrected and accelerated bootstrap confidence limits*

Description

This routine computes nonparametric confidence intervals for bootstrap estimates. For reproducibility, save or set the random number state before calling this routine.

Usage

```
bcajack(
  x,
  B,
  func,
  ...,
  m = nrow(x),
  mr = 5,
  K = 2,
  J = 10,
  alpha = c(0.025, 0.05, 0.1, 0.16),
  verbose = TRUE
)
```

Arguments

x	an $n \times p$ data matrix, rows are observed p-vectors, assumed to be independently sampled from target population. If p is 1 then x can be a vector.
B	number of bootstrap replications. It can also be a vector of B bootstrap replications of the estimated parameter of interest, computed separately.
func	function $\hat{\theta} = func(x)$ computing estimate of the parameter of interest; $func(x)$ should return a real value for any $n' \times p$ matrix x', n' not necessarily equal to n

`...`	additional arguments for `func`.
`m`	an integer less than or equal to n; the routine collects the n rows of `x` into `m` groups to speed up the jackknife calculations for estimating the acceleration value a; typically `m` is 20 or 40 and does not have to exactly divide n. However, warnings will be shown.
`mr`	if $m < n$ then `mr` repetions of the randomly grouped jackknife calculations are averaged.
`K`	a non-negative integer. If `K` > 0, bcajack also returns estimates of *internal standard error*, that is, of the variability due to stopping at `B` bootstrap replications rather than going on to infinity. These are obtained from a second type of jackknifing, taking an average of `K` separate jackknife estimates, each randomly splitting the `B` bootstrap replications into `J` groups.
`J`	the number of groups into which the bootstrap replications are split
`alpha`	percentiles desired for the bca confidence limits. One only needs to provide `alpha` values below 0.5; the upper limits are automatically computed
`verbose`	logical for verbose progress messages

Details

Bootstrap confidence intervals depend on three elements:

- the cdf of the B bootstrap replications t_i^*, $i = 1 \dots B$
- the bias-correction number $z_0 = \Phi(\sum_i^B I(t_i^* < t_0)/B)$ where $t_0 = f(x)$ is the original estimate
- the acceleration number a that measures the rate of change in σ_{t_0} as x, the data changes.

The first two of these depend only on the bootstrap distribution, and not how it is generated: parametrically or non-parametrically. Program bcajack can be used in a hybrid fashion in which the vector `tt` of B bootstrap replications is first generated from a parametric model.

So, in the diabetes example below, we might first draw bootstrap samples $y^* \sim N(X\hat{\beta}, \hat{\sigma}^2 I)$ where $\hat{\beta}$ and $\hat{\sigma}$ were obtained from `lm(y~X)`; each y^* would then provide a bootstrap replication `tstar = rfun(cbind(X, ystar))`. Then we could get bca intervals from `bcajack(Xy, tt, rfun ....)` with `tt`, the vector of B `tstar` values. The only difference from a full parametric bca analysis would lie in the nonparametric estimation of a, often a negligible error.

Value

a named list of several items

- **lims** : first column shows the estimated bca confidence limits at the requested alpha percentiles. These can be compared with the standard limits $\hat{\theta} + \hat{\sigma} z_\alpha$, third column. The second column `jacksd` gives the internal standard errors for the bca limits, quite small in the example. Column 4, `pct`, gives the percentiles of the ordered B bootstrap replications corresponding to the bca limits, eg the 897th largest replication equalling the .975 bca limit .557.
- **stats** : top line of stats shows 5 estimates: theta is $f(x)$, original point estimate of the parameter of interest; `sdboot` is its bootstrap estimate of standard error; `z0` is the bca bias correction value, in this case quite negative; `a` is the *acceleration*, a component of the bca limits (nearly

zero here); sdjack is the jackknife estimate of standard error for theta. Bottom line gives the internal standard errors for the five quantities above. This is substantial for z0 above.

- **B.mean** : bootstrap sample size B, and the mean of the B bootstrap replications $\hat{\theta}^*$
- **ustats** : The bias-corrected estimator `2 * t0 - mean(tt)`, and an estimate sdu of its sampling error
- **seed** : The random number state for reproducibility

References

DiCiccio T and Efron B (1996). Bootstrap confidence intervals. Statistical Science 11, 189-228

Efron B (1987). Better bootstrap confidence intervals. JASA 82 171-200

B. Efron and B. Narasimhan. Automatic Construction of Bootstrap Confidence Intervals, 2018.

Examples

```
data(diabetes, package = "bcaboot")
Xy <- cbind(diabetes$x, diabetes$y)
rfun <- function(Xy) {
  y <- Xy[, 11]
  X <- Xy[, 1:10]
  summary(lm(y~X) )$adj.r.squared
}
set.seed(1234)
## n = 442 = 34 * 13
bcajack(x = Xy, B = 1000, func = rfun, m = 34, verbose = FALSE)
```

---

bcajack2 *Nonparametric bias-corrected and accelerated bootstrap confidence limits*

Description

This function is a version of bcajack that allows all the recomputations of the original statistic function f to be carried out separately. This is an advantage if f is time-consuming, in which case the B replications for the nonparametric bca calculations might need to be done on a distributed basis.

To use bcajack2 in this mode, we first compute a list Blist via `Blist <- list(Y = Y, tt = tt, t0 = t0)`. Here tt is a vector of length B having i-th entry `tt[i] <- func(x[Ii,], ...)`, where x is the $n \times p$ data matrix and Ii is a bootstrap vector of (observation) indices. Y is a B by n count matrix, whose i-th row is the counts corresponding to Ii. For example if n = 5 and `Ii = (2, 5, 2, 1, 4)`, then `Yi = (1, 2, 0, 1, 1)`. Having computed Blist, bcajack2 is invoked as `bcajack2(Blist)` without need to enter the function $func$.

Usage

```
bcajack2(
  x,
  B,
  func,
  ...,
  m = nrow(x),
  mr,
  pct = 0.333,
  K = 2,
  J = 12,
  alpha = c(0.025, 0.05, 0.1, 0.16),
  verbose = TRUE
)
```

Arguments

x	an $n \times p$ data matrix, rows are observed p-vectors, assumed to be independently sampled from target population. If p is 1 then x can be a vector.
B	number of bootstrap replications. B can also be a vector of B bootstrap replications of the estimated parameter of interest, computed separately. If B is Blist as explained above, x is not needed.
func	function $\hat{\theta} = func(x)$ computing estimate of the parameter of interest; $func(x)$ should return a real value for any $n' \times p$ matrix x', n' not necessarily equal to n
...	additional arguments for func.
m	an integer less than or equal to n; the routine collects the n rows of x into m groups to speed up the jackknife calculations for estimating the acceleration value a; typically m is 20 or 40 and does not have to exactly divide n. However, warnings will be shown.
mr	if $m < n$ then mr repetions of the randomly grouped jackknife calculations are averaged.
pct	bcajack2 uses those count vectors nearest (1,1,...1) to estimate the gradient of the statistic, "nearest" being defined as those count vectors in the smallest pct of all B of them. Default value for 'pct is 1/3 (see appendix in Efron and Narasimhan for further details)
K	a non-negative integer. If K > 0, bcajack also returns estimates of *internal standard error*, that is, of the variability due to stopping at B bootstrap replications rather than going on to infinity. These are obtained from a second type of jackknifing, taking an average of K separate jackknife estimates, each randomly splitting the B bootstrap replications into J groups.
J	the number of groups into which the bootstrap replications are split
alpha	percentiles desired for the bca confidence limits. One only needs to provide alpha values below 0.5; the upper limits are automatically computed
verbose	logical for verbose progress messages

Value

a named list of several items

- **lims** : first column shows the estimated bca confidence limits at the requested alpha percentiles. These can be compared with the standard limits $\hat{\theta} + \hat{\sigma} z_\alpha$, third column. The second column `jacksd` gives the internal standard errors for the bca limits, quite small in the example. Column 4, `pct`, gives the percentiles of the ordered B bootstrap replications corresponding to the bca limits, eg the 897th largest replication equalling the .975 bca limit .557.
- **stats** : top line of stats shows 5 estimates: theta is $func(x)$, original point estimate of the parameter of interest; `sdboot` is its bootstrap estimate of standard error; `z0` is the bca bias correction value, in this case quite negative; `a` is the *acceleration*, a component of the bca limits (nearly zero here); `sdjack` is the jackknife estimate of standard error for theta. Bottom line gives the internal standard errors for the five quantities above. This is substantial for `z0` above.
- **B.mean** : bootstrap sample size B, and the mean of the B bootstrap replications $\hat{\theta}^*$
- **ustats** : The bias-corrected estimator `2 * t0 - mean(tt)`, and an estimate `sdu` of its sampling error
- **seed** : The random number state for reproducibility

Examples

```
data(diabetes, package = "bcaboot")
Xy <- cbind(diabetes$x, diabetes$y)
rfun <- function(Xy) {
  y <- Xy[, 11]
  X <- Xy[, 1:10]
  summary(lm(y~X) )$adj.r.squared
}
set.seed(1234)
bcajack2(x = Xy, B = 1000, func = rfun, m = 40, verbose = FALSE)
```

`bcapar` *Compute parametric bootstrap confidence intervals*

Description

bcapar computes parametric bootstrap confidence intervals for a real-valued parameter theta in a p-parameter exponential family. It is described in Section 4 of the reference below.

Usage

```
bcapar(
  t0,
  tt,
  bb,
```

```
  alpha = c(0.025, 0.05, 0.1, 0.16),
  J = 10,
  K = 6,
  trun = 0.001,
  pct = 0.333,
  cd = 0,
  func
)
```

Arguments

t0	Observed estimate of theta, usually by maximum likelihood.
tt	A vector of parametric bootstrap replications of theta of length B, usually large, say B = 2000
bb	A B by p matrix of natural sufficient vectors, where p is the dimension of the exponential family.
alpha	percentiles desired for the bca confidence limits. One only needs to provide alpha values below 0.5; the upper limits are automatically computed
J, K	Parameters controlling the jackknife estimates of Monte Carlo error: J jackknife folds, with the jackknife standard errors averaged over K random divisions of bb
trun	Truncation parameter used in the calculation of the acceleration a.
pct	Proportion of "nearby" b vectors used in the calculation of t., the gradient vector of theta.
cd	If cd is 1 the bca confidence density is also returned; see Section 11.6 in reference Efron and Hastie (2016) below
func	Function $\hat{\theta} = func(b)$. If this is not missing then output includes *abc* estimates; see reference DiCiccio and Efron (1992) below

Value

a named list of several items:

- **lims** : Bca confidence limits (first column) and the standard limits (fourth column). Also the abc limits (fifth column) if func is provided. The second column, jacksd, are the jackknife estimates of Monte Carlo error; pct, the third column are the proportion of the replicates tt less than each bcalim value
- **stats** : Estimates and their jackknife Monte Carlo errors: theta = $\hat{\theta}$; sd, the bootstrap standard deviation for $\hat{\theta}$; a the acceleration estimate; az another acceleration estimate that depends less on extreme values of tt; z0 the bias-correction estimate; A the big-A measure of raw acceleration; sdd delta method estimate for standard deviation of $\hat{\theta}$; mean the average of tt
- **abcstats** : The abc estimates of a and z0, returned if func was provided
- **ustats** : The bias-corrected estimator 2 * t0 - mean(tt). ustats gives ustat, an estimate sdu of its sampling error, and jackknife estimates of monte carlo error for both ustat and sdu. Also given is B, the number of bootstrap replications
- **seed** : The random number state for reproducibility

References

DiCiccio T and Efron B (1996). Bootstrap confidence intervals. Statistical Science 11, 189-228

T. DiCiccio and B. Efron. More accurate confidence intervals in exponential families. Biometrika (1992) p231-245.

Efron B (1987). Better bootstrap confidence intervals. JASA 82, 171-200

B. Efron and T. Hastie. Computer Age Statistical Inference. Cambridge University Press, 2016.

B. Efron and B. Narasimhan. Automatic Construction of Bootstrap Confidence Intervals, 2018.

Examples

```
data(diabetes, package = "bcaboot")
X <- diabetes$x
y <- scale(diabetes$y, center = TRUE, scale = FALSE)
lm.model <- lm(y ~ X - 1)
mu.hat <- lm.model$fitted.values
sigma.hat <- stats::sd(lm.model$residuals)
t0 <- summary(lm.model)$adj.r.squared
y.star <- sapply(mu.hat, rnorm, n = 1000, sd = sigma.hat)
tt <- apply(y.star, 1, function(y) summary(lm(y ~ X - 1))$adj.r.squared)
b.star <- y.star %*% X
set.seed(1234)
bcapar(t0 = t0, tt = tt, bb = b.star)
```

---

bcaplot *Plots of bca confidence limits*

Description

`bcaplot` uses the output of `bcajack`, `bcajack2`, or `bcapar` to plot bca and standard confidence limits for the parameter of interest.

Usage

```
bcaplot(
  vl,
  main = " ",
  xlab = "coverage",
  ylab = "limits",
  alpha = c(0.025, 0.05, 0.1, 0.16),
  ylim,
  xlim,
  add = 0,
  sub = "black=bca, green=standard",
  sw = 1,
  ...
)
```

Arguments

vl	output of bcajack, bcajack2, or bcapar
main	The main caption (can be empty)
xlab	The x axis label (supplied if not specified)
ylab	The y axis labels (supplied if not specified)
alpha	Coverages are $1-2\alpha$, e.g. alpha=c(.025,.05) plots intervals [.025,.975] and [.05,.95]. Default is alpha=c(.025,.05,.1,.16) giving coverages .95,.90,.80,.68
ylim	y axis plot limits set automatically if not provided
xlim	x axis plot limits set automatically if not provided
add	add=1 adds a new plot of bca limits (in red) to an existing plot
sub	subtitle (can be empty)
sw	sw=1 draws light vertical dashed lines showing the bca intervals
...	further args for plot

Details

confidence interval endpoints are plotted vertically versus two-sided coverages $1-2\alpha$. Bca limits in black, Standard limits in green (dashed.). If vl$lims includes the column "jacksd" of jackknife internal standard deviations then these are indicated by vertical red bars centered at the bca limit points.

---

diabetes *Blood and other measurements in diabetics*

Description

The diabetes data frame has 442 rows and 3 columns. These are the data used in the Efron et al "Least Angle Regression" paper.

Format

This data frame contains the following columns:

- **x** a matrix with 10 columns
- **y** a numeric vector
- **x2** a matrix with 64 columns

Details

The x matrix has been standardized to have unit L2 norm in each column and zero mean. The matrix x2 consists of x plus certain interactions.

Source

https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf

References

Efron, Hastie, Johnstone and Tibshirani (2003) "Least Angle Regression" (with discussion) *Annals of Statistics*

Index