Package 'boot.pval'

July 22, 2025

Title Bootstrap p-Values

Version 0.7.0

Description Computation of bootstrap p-values through inversion of confidence intervals, including convenience functions for regression models and tests of location.

License MIT + file LICENSE

URL https://github.com/mthulin/boot.pval, https://mthulin.github.io/boot.pval/

BugReports https://github.com/mthulin/boot.pval/issues

Encoding UTF-8

RoxygenNote 7.3.2

RdMacros Rdpack

Depends R (>= 4.1.0)

Imports boot, Rdpack, car, stats, lme4, survival, rms, gt, flextable

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Måns Thulin [aut, cre]

Maintainer Måns Thulin <mans@statistikkonsult.com>

Repository CRAN

Date/Publication 2025-03-05 13:30:06 UTC

Contents

`boot.pval` *Compute Bootstrap p-values*

Description

Compute bootstrap p-values through confidence interval inversion, as described in Hall (1992) and Thulin (2024).

Usage

```
boot.pval(
  boot_res,
  type = "perc",
  theta_null = 0,
  pval_precision = NULL,
  alternative = "two.sided",
  ...
)
```

Arguments

`boot_res`	An object of class "boot" containing the output of a bootstrap calculation.
`type`	A vector of character strings representing the type of interval to base the test on. The value should be one of "norm", "basic", "stud", "perc" (the default), and "bca".
`theta_null`	The value of the parameter under the null hypothesis.
`pval_precision`	The desired precision for the p-value. The default is 1/R, where R is the number of bootstrap samples in `boot_res`.
`alternative`	A character string specifying the alternative hypothesis. Must be one of "two.sided" (default), "greater", or "less".
`...`	Additional arguments passed to `boot.ci`.

Details

p-values can be computed by inverting the corresponding confidence intervals, as described in Section 14.2 of Thulin (2024) and Section 3.12 in Hall (1992). This function computes p-values in this way from "boot" objects. The approach relies on the fact that:

- the p-value of the two-sided test for the parameter theta is the smallest alpha such that theta is not contained in the corresponding 1-alpha confidence interval,
- for a test of the parameter theta with significance level alpha, the set of values of theta that aren't rejected by the two-sided test (when used as the null hypothesis) is a 1-alpha confidence interval for theta.

Value

A bootstrap p-value.

References

Hall P (1992). *The Bootstrap and Edgeworth Expansion*. Springer, New York. ISBN 9781461243847.
Thulin M (2024). *Modern Statistics with R*. Chapman & Hall/CRC Press, Boca Raton. ISBN 9781032512440, https://www.modernstatisticswithr.com/.

See Also

boot_t_test() for bootstrap t-tests, boot_median_test() for bootstrap tests for medians, boot_summary() for bootstrap tests for coefficients of regression models.

Examples

```
# Hypothesis test for the city data
# H0: ratio = 1.4
library(boot)
ratio <- function(d, w) sum(d$x * w)/sum(d$u * w)
city.boot <- boot(city, ratio, R = 99, stype = "w", sim = "ordinary")
boot.pval(city.boot, theta_null = 1.4)

# Studentized test for the two sample difference of means problem
# using the final two series of the gravity data.
diff.means <- function(d, f)
{
  n <- nrow(d)
  gp1 <- 1:table(as.numeric(d$series))[1]
  m1 <- sum(d[gp1,1] * f[gp1])/sum(f[gp1])
  m2 <- sum(d[-gp1,1] * f[-gp1])/sum(f[-gp1])
  ss1 <- sum(d[gp1,1]^2 * f[gp1]) - (m1 *  m1 * sum(f[gp1]))
  ss2 <- sum(d[-gp1,1]^2 * f[-gp1]) - (m2 *  m2 * sum(f[-gp1]))
  c(m1 - m2, (ss1 + ss2)/(sum(f) - 2))
}
grav1 <- gravity[as.numeric(gravity[,2]) >= 7, ]
grav1.boot <- boot(grav1, diff.means, R = 99, stype = "f",
                   strata = grav1[ ,2])
boot.pval(grav1.boot, type = "stud", theta_null = 0)
```

---

boot_median_test *Bootstrap Median Test*

Description

Performs one- and two-sample bootstrap median tests and computes the corresponding bootstrap confidence interval.

Usage

```
boot_median_test(x, ...)

## Default S3 method:
boot_median_test(
  x,
  y = NULL,
  alternative = c("two.sided", "less", "greater"),
  mu = 0,
  paired = FALSE,
  var.equal = FALSE,
  conf.level = 0.95,
  R = 9999,
  type = "stud",
  ...
)

## S3 method for class 'formula'
boot_median_test(formula, data, subset, na.action, ...)

## S3 method for class 'data.frame'
boot_median_test(x, formula, ...)

## S3 method for class 'matrix'
boot_median_test(x, formula, ...)
```

Arguments

x	a (non-empty) numeric vector of data values.
...	Additional arguments passed to boot, such as parallel for parallel computations. See ?boot::boot for details.
y	an optional (non-empty) numeric vector of data values.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
mu	a number indicating the true value of the mean (or difference in means if you are performing a two sample test).
paired	a logical indicating whether you want a paired t-test.
var.equal	a logical variable indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.
conf.level	confidence level of the interval.
R	The number of bootstrap replicates. The default is 9999.
type	A vector of character strings representing the type of interval to base the test on. The value should be one of "norm", "basic", "bca", perc", and "stud" (the default).

formula	a formula of the form lhs ~ rhs where lhs is a numeric variable giving the data values and rhs either 1 for a one-sample or paired test or a factor with two levels giving the corresponding groups. If lhs is of class "Pair" and rhs is 1, a paired test is done, see Examples.
data	an optional matrix or data frame (or similar: see model.frame) containing the variables in the formula formula. By default the variables are taken from environment(formula).
subset	an optional vector specifying a subset of observations to be used.
na.action	a function which indicates what should happen when the data contain NAs.

Details

p-values can be computed by inverting the corresponding confidence intervals, as described in Section 14.2 of Thulin (2024) and Section 3.12 in Hall (1992). This function computes p-values for the Median Test in this way. The approach relies on the fact that:

- the p-value of the two-sided test for the parameter theta is the smallest alpha such that theta is not contained in the corresponding 1-alpha confidence interval,
- for a test of the parameter theta with significance level alpha, the set of values of theta that aren't rejected by the two-sided test (when used as the null hypothesis) is a 1-alpha confidence interval for theta. Consequently, the p-value will be consistent with the confidence interval, in the sense that the null hypothesis is rejected if and only if the null parameter values is not contained in the confidence interval.

Value

A list with class "htest") containing the following components:

statistic	the value of the test statistic.
R	the number of bootstrap replicates used.
p.value	the bootstrap p-value for the test.
conf.int	a bootstrap confidence interval for the median appropriate to the specified alternative hypothesis.
estimate	the estimated median or difference in medians depending on whether it was a one-sample test or a two-sample test.
null.value	the specified hypothesized value of the median or median difference depending on whether it was a one-sample test or a two-sample test.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating what type of median test was performed.
data.name	a character string giving the name(s) of the data.

References

Hall P (1992). *The Bootstrap and Edgeworth Expansion*. Springer, New York. ISBN 9781461243847.
Thulin M (2024). *Modern Statistics with R*. Chapman & Hall/CRC Press, Boca Raton. ISBN 9781032512440, https://www.modernstatisticswithr.com/.

See Also

boot_t_test() for bootstrap t-tests, boot_summary() for bootstrap tests for coefficients of regression models.

Examples

```
## Not run:
# Generate example data:
# x is the variable of interest
# y is the grouping variable
example_data <- data.frame(x = rnorm(40), y = rep(c(1,2), 20))

# Two-sample test:
boot_median_test(x ~ y, data = example_data, R = 999)

# Two-sample test using the pipe:
example_data |> boot_median_test(x ~ y, R = 999)

# With a directed alternative hypothesis:
example_data |> boot_median_test(x ~ y, R = 999, alternative = "greater")

# One-sample test:
boot_median_test(example_data$x, R = 999)

# One-sample test using the pipe:
example_data |> boot_median_test(x ~ 1, R = 999)

# With a directed alternative hypothesis:
example_data |> boot_median_test(x ~ 1, R = 999, mu = 0.5, alternative = "less")

# Paired test:
boot_median_test(example_data$x[example_data$y==1],
            example_data$x[example_data$y==2],
            paired = TRUE, R = 999)

# Paired test using the pipe (after reshaping to wide format):
example_data$id <- rep(1:20, rep(2, 20))
example_data2 <- reshape(example_data, direction = "wide",
                         idvar = "id", timevar = "y")
example_data2 |> boot_median_test(Pair(x.1, x.2) ~ 1)

## End(Not run)
```

---

boot_summary *Summarising Regression Models Using the Bootstrap*

Description

Summaries for regression models, including "lm", "glm", "glm.nb", "nls", "rlm", "polr", and "merMod" ("lmer", "glmer") objects, using the bootstrap for p-values and confidence intervals.

Usage

```
boot_summary(
  model,
  type = "perc",
  method = NULL,
  conf.level = 0.95,
  R = 999,
  coef = "raw",
  pval_precision = NULL,
  adjust.method = "none",
  ...
)
```

Arguments

model	An object fitted using e.g. "lm", "glm", "glm.nb", "nls", "rlm", "polr", lmer", or "glmer".
type	A vector of character strings representing the type of interval to base the test on. The value should be one of "norm", "basic", "bca", and "perc" (the default). "bca" is not supported for "lmer" and "glmer" models.
method	The method used for bootstrapping. For "lm" and "nls" objects use either "residual" (for resampling of scaled and centred residuals, the default) or "case" (for case resampling). For "glm" objects, use "case" (the default). For "merMod" objects (mixed models) use either "parametric" (the default) or "semiparametric".
conf.level	The confidence level for the confidence intervals. The default is 0.95.
R	The number of bootstrap replicates. The default is 999.
coef	A string specifying whether to use exponentiated coefficients in the summary table. Either "exp" (for exponentiated coefficients, i.e. odds ratios in the case of a logistic regression model) or "raw" (for coefficients on their original scale). The default is "raw", which is recommended for linear models.
pval_precision	The desired precision for the p-value. The default is 1/R.
adjust.method	Adjustment of p-values for multiple comparisons using `p.adjust`. The default is "none", in which case the p-values aren't adjusted. The other options are "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", and "fdr"; see `?p.adjust` for details on these methods.
...	Additional arguments passed to `Boot` or `bootMer`, such as `parallel` for parallel computations. See `?car::Boot` and `?lme4::bootMer` for details.

Details

p-values can be computed by inverting the corresponding confidence intervals, as described in Section 14.2 of Thulin (2024) and Section 3.12 in Hall (1992). This function computes p-values for coefficients of regression models in this way. The approach relies on the fact that:

- the p-value of the two-sided test for the parameter theta is the smallest alpha such that theta is not contained in the corresponding 1-alpha confidence interval,

- for a test of the parameter theta with significance level alpha, the set of values of theta that aren't rejected by the two-sided test (when used as the null hypothesis) is a 1-alpha confidence interval for theta.

The function can be used with "lm", "glm", "glm.nb", "nls", "rlm", and "merMod" ("lmer", "glmer") objects. In addition, it should work for any regression model such that: `residuals(object, type="pearson")` returns Pearson residuals; `fitted(object)` returns fitted values; `hatvalues(object)` returns the leverages, or perhaps the value 1 which will effectively ignore setting the hatvalues. In addition, the `data` argument should contain no missing values among the columns actually used in fitting the model.

Value

A data frame containing coefficient estimates, bootstrap confidence intervals, and bootstrap p-values.

References

Hall P (1992). *The Bootstrap and Edgeworth Expansion*. Springer, New York. ISBN 9781461243847.
Thulin M (2024). *Modern Statistics with R*. Chapman & Hall/CRC Press, Boca Raton. ISBN 9781032512440, https://www.modernstatisticswithr.com/.

See Also

`boot_t_test()` for bootstrap t-tests, `boot_median_test()` for bootstrap tests for medians.

Examples

```
# Bootstrap summary of a linear model for mtcars:
model <- lm(mpg ~ hp + vs, data = mtcars)
boot_summary(model, R = 99)
# (Values for R greater than 99 are recommended for most applications.)

# Adjust p-values for multiplicity using Holm's method:
boot_summary(model, R = 99, adjust.method = "holm")
```

---

boot_t_test *Bootstrap t-Test*

Description

Performs one- and two-sample bootstrap t-tests and computes the corresponding bootstrap confidence interval.

Usage

```
boot_t_test(x, ...)

## Default S3 method:
boot_t_test(
  x,
  y = NULL,
  alternative = c("two.sided", "less", "greater"),
  mu = 0,
  paired = FALSE,
  var.equal = FALSE,
  conf.level = 0.95,
  R = 9999,
  type = "stud",
  ...
)

## S3 method for class 'formula'
boot_t_test(formula, data, subset, na.action, ...)

## S3 method for class 'data.frame'
boot_t_test(x, formula, ...)

## S3 method for class 'matrix'
boot_t_test(x, formula, ...)
```

Arguments

x	a (non-empty) numeric vector of data values.
...	Additional arguments passed to boot, such as parallel for parallel computations. See ?boot::boot for details.
y	an optional (non-empty) numeric vector of data values.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
mu	a number indicating the true value of the mean (or difference in means if you are performing a two sample test).
paired	a logical indicating whether you want a paired t-test.
var.equal	a logical variable indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.
conf.level	confidence level of the interval.
R	The number of bootstrap replicates. The default is 9999.
type	A vector of character strings representing the type of interval to base the test on. The value should be one of "norm", "basic", "bca", perc", and "stud" (the default).

formula	a formula of the form lhs ~ rhs where lhs is a numeric variable giving the data values and rhs either 1 for a one-sample or paired test or a factor with two levels giving the corresponding groups. If lhs is of class "Pair" and rhs is 1, a paired test is done, see Examples.
data	an optional matrix or data frame (or similar: see model.frame) containing the variables in the formula formula. By default the variables are taken from environment(formula).
subset	an optional vector specifying a subset of observations to be used.
na.action	a function which indicates what should happen when the data contain NAs.

Details

p-values can be computed by inverting the corresponding confidence intervals, as described in Section 14.2 of Thulin (2024) and Section 3.12 in Hall (1992). This function computes p-values for the t-test in this way. The approach relies on the fact that:

- the p-value of the two-sided test for the parameter theta is the smallest alpha such that theta is not contained in the corresponding 1-alpha confidence interval,
- for a test of the parameter theta with significance level alpha, the set of values of theta that aren't rejected by the two-sided test (when used as the null hypothesis) is a 1-alpha confidence interval for theta. Consequently, the p-value will be consistent with the confidence interval, in the sense that the null hypothesis is rejected if and only if the null parameter values is not contained in the confidence interval.

Value

A list with class "htest") containing the following components:

statistic	the value of the t-statistic.
R	the number of bootstrap replicates used.
p.value	the bootstrap p-value for the test.
conf.int	a bootstrap confidence interval for the mean appropriate to the specified alternative hypothesis.
estimate	the estimated mean or difference in means depending on whether it was a one-sample test or a two-sample test.
null.value	the specified hypothesized value of the mean or mean difference depending on whether it was a one-sample test or a two-sample test.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating what type of t-test was performed.
data.name	a character string giving the name(s) of the data.

References

Hall P (1992). *The Bootstrap and Edgeworth Expansion*. Springer, New York. ISBN 9781461243847.
Thulin M (2024). *Modern Statistics with R*. Chapman & Hall/CRC Press, Boca Raton. ISBN 9781032512440, https://www.modernstatisticswithr.com/.

See Also

boot_median_test() for bootstrap tests for medians, boot_summary() for bootstrap tests for coefficients of regression models.

Examples

```
# Generate example data:
# x is the variable of interest
# y is the grouping variable
example_data <- data.frame(x = rnorm(40), y = rep(c(1,2), 20))

# Two-sample (Welch) test:
boot_t_test(x ~ y, data = example_data, R = 999)

# Two-sample (Welch) test using the pipe:
example_data |> boot_t_test(x ~ y, R = 999)

# With a directed alternative hypothesis:
example_data |> boot_t_test(x ~ y, R = 999, alternative = "greater")

# One-sample test:
boot_t_test(example_data$x, R = 999)

# One-sample test using the pipe:
example_data |> boot_t_test(x ~ 1, R = 999)

# With a directed alternative hypothesis:
example_data |> boot_t_test(x ~ 1, R = 999, mu = 0.5, alternative = "less")

# Paired test:
boot_t_test(example_data$x[example_data$y==1],
            example_data$x[example_data$y==2],
            paired = TRUE, R = 999)

# Paired test using the pipe (after reshaping to wide format):
example_data$id <- rep(1:20, rep(2, 20))
example_data2 <- reshape(example_data, direction = "wide",
                         idvar = "id", timevar = "y")
example_data2 |> boot_t_test(Pair(x.1, x.2) ~ 1)
```

---

censboot_summary *Summarising Survival Regression Models Using the Bootstrap*

Description

Summaries for Cox proportional hazards and accelerated failure time models, using the bootstrap for p-values and confidence intervals.

Usage

```
censboot_summary(
  model,
  type = "perc",
  sim = "ordinary",
  strata = NULL,
  coef = "exp",
  conf.level = 0.95,
  R = 999,
  pval_precision = NULL,
  adjust.method = "none",
  ...
)
```

Arguments

model	An object fitted using "survival::coxph", "survival::survreg", or "rms::psm".
type	A vector of character strings representing the type of interval to base the test on. The value should be one of "norm", "basic", and "perc" (the default).
sim	The method used for bootstrapping. See ?boot::censboot for details. Currently only "ordinary" (case resampling) is supported.
strata	The strata used in the calls to survfit. It can be a vector or a matrix with 2 columns. If it is a vector then it is assumed to be the strata for the survival distribution, and the censoring distribution is assumed to be the same for all observations. If it is a matrix then the first column is the strata for the survival distribution and the second is the strata for the censoring distribution. When sim = "ordinary", only one set of strata is used to stratify the observations. This is taken to be the first column of strata when it is a matrix.
coef	A string specifying whether to use exponentiated coefficients in the summary table. Either "exp" (for exponentiated coefficients, i.e. hazard ratios in the case of a Cox PH model) or "raw" (for coefficients on their original scale). The default is "exp".
conf.level	The confidence level for the confidence intervals. The default is 0.95.
R	The number of bootstrap replicates. The default is 999.
pval_precision	The desired precision for the p-value. The default is 1/R.
adjust.method	Adjustment of p-values for multiple comparisons using p.adjust. The default is "none", in which case the p-values aren't adjusted. The other options are "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", and "fdr"; see ?p.adjust for details on these methods.
...	Additional arguments passed to censboot, such as parallel for parallel computations. See ?boot::censboot for details.

Details

p-values can be computed by inverting the corresponding confidence intervals, as described in Section 14.2 of Thulin (2024) and Section 3.12 in Hall (1992). This function computes p-values in this way from "coxph" or "survreg" objects. The approach relies on the fact that:

- the p-value of the two-sided test for the parameter theta is the smallest alpha such that theta is not contained in the corresponding 1-alpha confidence interval,
- for a test of the parameter theta with significance level alpha, the set of values of theta that aren't rejected by the two-sided test (when used as the null hypothesis) is a 1-alpha confidence interval for theta.

Value

A data frame containing coefficient estimates, bootstrap confidence intervals, and bootstrap p-values.

References

Hall P (1992). *The Bootstrap and Edgeworth Expansion*. Springer, New York. ISBN 9781461243847.
Thulin M (2024). *Modern Statistics with R*. Chapman & Hall/CRC Press, Boca Raton. ISBN 9781032512440, https://www.modernstatisticswithr.com/.

Examples

```
library(survival)
# Weibull AFT model:
# Note that model = TRUE is required for use with censboot_summary:
model <- survreg(formula = Surv(time, status) ~ age + sex, data = lung,
                 dist = "weibull", model = TRUE)
censboot_summary(model, R = 99)
# (Values for R greater than 99 are recommended for most applications.)

# Cox PH model:
model <- coxph(formula = Surv(time, status) ~ age + sex, data = lung,
               model = TRUE)
# Table with hazard ratios:
censboot_summary(model, R = 99)
censboot_summary(model, coef = "raw", R = 99)
```

---

summary_to_flextable *Convert Bootstrap Summary Tables to flextable Objects*

Description

Converts tables created using `boot_summary` and `censboot_summary` to nicely formatted `flextable` tables.

Usage

```
summary_to_flextable(summary_table, decimals = 3, conf = "95 % CI")
```

Arguments

summary_table	A table created using boot_summary or censboot_summary.
decimals	The number of decimals to print for estimates and confidence intervals. The default is 3.
conf	The text at the top of the confidence interval column in the gt table. The default is "95 % CI".

Value

A flextable object.

Examples

```
# Bootstrap summary of a linear model for mtcars:
model <- lm(mpg ~ hp + vs, data = mtcars)
boot_summary(model, R = 99) |> summary_to_flextable()

# Export to Word:
## Not run:
boot_summary(model, R = 99) |>
   summary_to_flextable() |>
   flextable::save_as_docx(path = "my_table.docx")

## End(Not run)
```

---

summary_to_gt *Convert Bootstrap Summary Tables to gt Objects*

Description

Converts tables created using boot_summary and censboot_summary to nicely formatted gt tables.

Usage

```
summary_to_gt(summary_table, decimals = 3, conf = "95 % CI")
```

Arguments

summary_table	A table created using boot_summary or censboot_summary.
decimals	The number of decimals to print for estimates and confidence intervals. The default is 3.
conf	The text at the top of the confidence interval column in the gt table. The default is "95 % CI".

Value

A gt table.

Examples

```
# Bootstrap summary of a linear model for mtcars:
model <- lm(mpg ~ hp + vs, data = mtcars)
boot_summary(model, R = 99) |> summary_to_gt()
```

Index