Package ‘ClusterBootstrap’

August 21, 2025

Title Analyze Clustered Data using the Cluster Bootstrap

Date 2025-08-21

Version 2.0.0

Description Provides functionality for the analysis of clustered data using the cluster bootstrap.

Depends R (>= 4.1)

Imports stats, utils, graphics, parallel, data.table

Suggests dplyr

License GPL-3 | file LICENSE

URL `https://github.com/mathijsdeen/ClusterBootstrap`

BugReports `https://github.com/mathijsdeen/ClusterBootstrap/issues`

Maintainer Mathijs Deen <`dev@mathijsdeen.com`>

LazyData true

RoxygenNote 7.3.2

Encoding UTF-8

NeedsCompilation no

Author Mathijs Deen [aut, cre],
Mark de Rooij [aut]

Repository CRAN

Date/Publication 2025-08-21 09:30:01 UTC

Contents

`clusbootglm` *Fit generalized linear models with the cluster bootstrap*

Description

Fit a generalized linear model with the cluster bootstrap for analysis of clustered data.

Usage

```
clusbootglm(
  model,
  data,
  clusterid,
  family = gaussian,
  B = 5000,
  confint.level = 0.95,
  n.cores = 1
)
```

Arguments

`model`	generalized linear model to be fitted with the cluster bootstrap. This should either be a formula (or be able to be interpreted as one) or a `glm` / `lm` object. From the `(g)lm` objects, the formula will be used.
`data`	dataframe that contains the data.
`clusterid`	variable in data that identifies the clusters.
`family`	error distribution to be used in the model, e.g. `gaussian` or `binomial`.
`B`	number of bootstrap samples.
`confint.level`	level of confidence interval.
`n.cores`	number of CPU cores to be used.

Details

Some useful methods for the obtained `clusbootglm` class object are `summary.clusbootglm`, `coef.clusbootglm`, and `clusbootsample`.

Value

`clusbootglm` produces an object of class `"clusbootglm"`, containing the following relevant components:

`coefficients`	A matrix of `B` rows, containing the parameter estimates for all bootstrap samples.
`bootstrap.matrix`	n*B matrix, of which each column represents a bootstrap sample; each value in a column represents a unit of `subjectid`.
`lm.coefs`	Parameter estimates from a single (generalized) linear model.
`boot.coefs`	Mean values of the paramater estimates, derived from the bootstrap coefficients.
`boot.sds`	Standard deviations of cluster bootstrap parameter estimates.
`ci.level`	User defined confidence interval level.
`percentile.interval`	Confidence interval based on percentiles, given the user defined confidence interval level.
`parametric.interval`	Confidence interval based on `lm.coefs` and column standard deviations of `coefficients`, given the user defined confidence interval level.
`BCa.interval`	Confidence interval based on percentiles with bias correction and acceleration, given the user defined confidence interval level.
`samples.with.NA.coef`	Cluster bootstrap sample numbers with at least one coefficient being `NA`.
`failed.bootstrap.samples`	For each of the coefficients, the number of failed bootstrap samples are given.

Author(s)

Mathijs Deen, Mark de Rooij

Examples

```
## Not run:
data(opposites)
clusbootglm(SCORE~Time*COG,data=opposites,clusterid=Subject)
## End(Not run)
```

`clusbootsample` *Return data for specified bootstrap sample*

Description

Returns the full data frame for a specified bootstrap sample in a `clusbootglm` object.

Usage

```
clusbootsample(object, samplenr)
```

Arguments

`object`	object of class `clusbootglm`, created with the `clusbootglm` function.
`samplenr`	sample number for which the data frame should be returned.

Author(s)

Mark de Rooij, Mathijs Deen

Examples

```
## Not run:
data(opposites)
cbglm.1 <- clusbootglm(SCORE~Time*COG,data=opposites,clusterid=Subject)
clusbootsample(cbglm.1, samplenr=1)
## End(Not run)
```

`clusterBootstrap` *Cluster Bootstrap*

Description

Performs bootstrapping on hierarchically structured data using clustered or nested resampling at any level of the hierarchy. Allows bootstrapping of arbitrary statistics computed from the resampled dataset.

Usage

```
clusterBootstrap(df, clusters, replace, stat_fun, B = 5000, ...)
```

Arguments

`df`	A data frame. The original dataset.
`clusters`	A character vector of variable names that define the nested structure of the data, ordered from highest to lowest level.
`replace`	A logical vector indicating whether sampling should be with replacement at each level. Should be of the same length as `clusters`.
`stat_fun`	A function that takes a data frame (a bootstrap sample) and returns a numeric vector of statistics.
`B`	Integer. The number of bootstrap samples to generate.
`...`	Additional arguments passed to `stat_fun`.

Value

`clusterBootstrap` returns an object of class `clusterBootstrap`, containing the following elements:

`call`	The function call
`args`	Arguments passed to the function
`estimates`	A list with the following elements: • `originalEstimates`: a `data.frame` with one row, containing the return of `stat_fun` on the original data. • `bootstrapEstimates`: a `data.frame` with B rows, containing the return of `stat_fun` on each of the bootstrap samples. • `bootstrapSE`: the bootstrap standard error(s) for all rows in `bootstrapEstimates`.

Author(s)

Mathijs Deen

See Also

`clusterResample` for the underlying resampling mechanism. `confint.clusterBootstrap` `for cluster bootstrap confidence intervals.`

Examples

```
## Not run:
library(dplyr)
medData <- medication |>
filter(time %% 1 == 0, time < 4)
bootFun <- function(d) lm(pos ~ treat*time, data = d)$coefficients

# Resampling on the person level only
clusterBootstrap(df       = medData,
                 clusters = "id",
                 replace  = TRUE,
                 stat_fun = bootFun,
                 B        = 5000)

# Resampling on the person level and the repeated measures level
clusterBootstrap(df       = medData,
                 clusters = c("id", "time"),
                 replace  = c(TRUE, TRUE),
                 stat_fun = bootFun,
                 B        = 5000)

# Not resampling at one level
# (e.g., by design all classes in a probed school are included,
# but not all students in a class)
set.seed(2025)
n_school  <- 30
n_class   <- 8
```

```
n_student <- 15

demo <- expand.grid(
school  = paste0("S", 1:n_school),
class   = paste0("C", 1:n_class),
student = paste0("P", 1:n_student)) |>
  mutate(score1 = rnorm(n()),
         score2 = rnorm(n())) |>
  arrange(school, class, student) |>
  slice(1:(n() - 3)) # slightly unbalanced data
bootFun2 <- function(d) lm(score1 ~ score2, data = d)$coef
clusterBootstrap(df       = demo,
                 clusters = c("school", "class", "student"),
                 replace  = c(TRUE, FALSE, TRUE),
                 stat_fun = bootFun2,
                 B        = 1000)

## End(Not run)
```

`clusterResample` *Cluster Resampling*

Description

Performs hierarchical (clustered or nested) resampling of a data frame across one or more grouping variables. Each level of grouping can be resampled with or without replacement.

Usage

```
clusterResample(df, clusters, replace)
```

Arguments

`df`	A data frame or data table. The original dataset to be resampled.
`clusters`	A character vector of variable names that define the nested structure of the data. The order should be from highest (outermost) to lowest (innermost) level.
`replace`	A logical vector, of the same length as `clusters`, indicating whether to sample with replacement at each level.

Details

This function supports arbitrary nesting depth, and preserves the original hierarchical structure during resampling. At each level, sampling is done conditionally within the grouping structure defined by the higher levels.

Value

A resampled data.table with the same column structure as `df`, potentially with repeated or dropped rows depending on `replace`.

Author(s)

Mathijs Deen

See Also

`clusterBootstrap` that uses the current function.

Examples

```
## Not run:
set.seed(123)
df <- expand.grid(
  school = paste0("S", 1:5),
  class  = paste0("C", 1:5),
  student = paste0("P", 1:5)
)
df$score <- rnorm(nrow(df))

resampled <- clusterResample(df, clusters = c("school", "class", "student"),
                              replace = c(TRUE, TRUE, FALSE))

## End(Not run)
```

`coef.clusbootglm` *Obtain coefficients from cluster bootstrap object*

Description

Returns the coefficients of an object of class `clusbootglm`.

Usage

```
## S3 method for class 'clusbootglm'
coef(object, estimate.type = "bootstrap", ...)
```

Arguments

`object`	object of class `clusbootglm`.
`estimate.type`	type of coefficient (`bootstrap` or `GLM`).
`...`	other arguments.

Author(s)

Mathijs Deen

Examples

```
## Not run:
data(opposites)
cbglm.1 <- clusbootglm(SCORE~Time*COG,data=opposites,clusterid=Subject)
coef(cbglm.1, estimate.type="bootstrap")
## End(Not run)
```

`confint.clusbootglm` *Confidence intervals for cluster bootstrap model parameters*

Description

Computes confidence intervals for one or more parameters in a fitted GLM with the cluster bootstrap.

Usage

```
## S3 method for class 'clusbootglm'
confint(object, parm = "all", level = 0.95, interval.type = "BCa", ...)
```

Arguments

`object`	object of class `clusbootglm`.
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. Defaults to all parameters.
`level`	the required confidence level
`interval.type`	type of confidence level. Options are `BCa`, `percentile`, and `parametric`.
`...`	other arguments.

Author(s)

Mathijs Deen

Examples

```
## Not run:
data(opposites)
cbglm.1 <- clusbootglm(SCORE~Time*COG,data=opposites,clusterid=Subject)
confint(cbglm.1,parm=c("Time","COG"), level=.90, interval.type="percentile")
## End(Not run)
```

`confint.clusterBootstrap` *Confidence intervals for clusterBootstrap objects*

Description

Computes confidence intervals for estimates obtained via clustered bootstrap resampling. Supported interval types are: percentile (default), normal approximation (parametric), and bias-corrected (BC).

Usage

```
## S3 method for class 'clusterBootstrap'
confint(
  object,
  parm = NULL,
  level = 0.95,
  type = c("percentile", "parametric", "bc"),
  ...
)
```

Arguments

`object`	An object of class `"clusterBootstrap"`, as returned by `clusterBootstrap`.
`parm`	A character vector of parameter names to compute confidence intervals for. If `NULL` (default), intervals are computed for all parameters.
`level`	Confidence level, e.g., `0.95` for a 95% confidence interval.
`type`	Type of confidence interval. One of `"percentile"`, `"parametric"`, or `"bc"`. The default is `"percentile"`.
`...`	Currently ignored. Included for method compatibility.

Details

- **Percentile**: uses the empirical quantiles of the bootstrap estimates.
- **Parametric**: uses the bootstrap standard error and assumes normality.
- **Bias-corrected (BC)**: adjusts for bias in the bootstrap distribution. Note: acceleration (BCa) is not implemented.

Value

A `data.frame` with one row per parameter and the following columns:

term The name of the parameter.

type The type of confidence interval used.

conf.low The lower bound of the confidence interval.

conf.high The upper bound of the confidence interval.

See Also

clusterBootstrap, confint

Examples

```
## Not run:
set.seed(2025)
n_school  <- 30
n_class   <- 8
n_student <- 15

demo <- expand.grid(school  = paste0("S", 1:n_school),
                    class   = paste0("C", 1:n_class),
                    student = paste0("P", 1:n_student)) |>
  mutate(score1 = rnorm(n()),
  score2 = rnorm(n())) |>
  arrange(school, class, student) |>
  slice(1:(n() - 3)) # slightly unbalanced data
bootFun2 <- function(d) lm(score1 ~ score2, data = d)$coef
clusterBootstrap(df       = demo,
                 clusters = c("school", "class", "student"),
                 replace  = c(TRUE, FALSE, TRUE),
                 stat_fun = bootFun2,
                 B        = 1000) |>
  confint()

## End(Not run)
```

---

emm *Calculate estimated marginal means for a cluster bootstrap GLM*

Description

Returns the estimated marginal means of a `clusbootglm` object. This function works with a maximum of one between-subjects and one within-subjects variable.

Usage

```
emm(object, confint.level = 0.95)
```

Arguments

`object`	object of class `clusbootglm`.
`confint.level`	level of the confidence interval.

Value

`emm` returns an object of class `clusbootemm`, containing the following components:

`grid`	Grid with estimated marginal means for each combination of levels of the variables.
`bootstrapsample.emm`	p*B matrix, with p being the number of estimates and B being the number of bootstrap samples.

Author(s)

Mathijs Deen

Examples

```
## Not run:
medication <- medication[medication$time %% 1 == 0,]
medication$time_f <- as.factor(medication$time)
set.seed(1)
model.1 <- clusbootglm(pos~time_f*treat, clusterid = id, data = medication)
emm.1 <- emm(object = model.1)
summary(object = emm.1)
## End(Not run)
```

`medication` *Medication data*

Description

The `medication` dataframe consists of 1242 observations within 73 individuals that were part of a placebo controlled clinical trial, as reported in Tomarken, Shelton, Elkins, and Anderson (1997).

The data were retrieved from the accompanied website of Singer & Willett (2003), at https://stats.idre.ucla.edu/other/examples

Usage

```
medication
```

Format

the following variables are available:

- `id`: subject indicator
- `treat`: either placebo (0) or antidepressant (1)
- `time`: number of days since trial start.
- `pos`: positive affect. Higher scores indicate a more positive mood.

References

- Singer, J.D., & Willett, J.B. (2003). *Applied longitudinal data analysis. Modeling change and event occurence.* NY: Oxford University Press, Inc.
- Tomarken, A.J., Shelton, R.C., Elkins, L., & Anderson, T (1997). *Sleep deprivation and anti-depressant medication: Unique effects on positive and negative affect.* Poster session presented at the 9th annual meeting of the American Psychological Society, Washington, DC.

`opposites` *Opposites naming data*

Description

The `opposites` dataframe consists of 144 observations within 36 individuals that completed an inventory that assesses their performance on a timed cognitive task called "opposites naming".

The dataset does not contain the empirical data within 35 individuals from the experiment by Willett (1988), but a simulation based on the multilevel model from Singer & Willett (2003) within 36 individuals.

Usage

```
opposites
```

Format

the following variables are available:

- `Subject`: subject indicator
- `Time`: a time variable, ranging 0-3
- `COG`: cognitive skill, measured once (at time=0)
- `SCORE`: score on opposites naming task

References

- Willett, J.B. (1988). Questions and answers in the measurement of change. In: E. Rothkopf (Ed.), *Review of research in education (1988-89)* (pp. 345-422). Washington, DC: American Educational Research Association.
- Singer, J.D., & Willett, J.B. (2003). *Applied longitudinal data analysis. Modeling change and event occurence.* NY: Oxford University Press, Inc.

`plot.clusbootemm` *Plot estimated marginal means for a cluster bootstrap GLM*

Description

Plots the estimated marginal means of an `clusbootglm` object. Works with one within-subjects and/or one between-subjects variable.

Usage

```
## S3 method for class 'clusbootemm'
plot(
  x,
  within,
  between,
  pch,
  lty,
  pcol,
  lcol,
  ylim,
  ylab = "Estimated marginal mean",
  xlab = "Within subject",
  ...
)
```

Arguments

`x`	object of class `clusbootemm`.
`within`	within-subjects variable. Should be numeric or numerically labeled factor.
`between`	between-subjects variable.
`pch`	point character. Length must be equal to the number of between-subjects levels.
`lty`	linetype. Length must be equal to the number of between-subjects levels.
`pcol`	point color. Length must be equal to the number of between-subjects levels.
`lcol`	line color. Length must be equal to the number of between-subjects levels.
`ylim`	limits of the y axis. If omitted, it will be based on the lowest and highest values within the confidence intervals of the estimated marginal means.
`ylab`	label for y-axis.
`xlab`	label for x-axis.
`...`	other arguments to be passed to the `plot` function (see `par`).

Author(s)

Mathijs Deen

Examples

```
## Not run:
medication <- medication[medication$time %% 1 == 0,]
medication$time_f <- as.factor(medication$time)
set.seed(1)
model.1 <- clusbootglm(pos~time_f*treat, clusterid = id, data = medication)
emm.1 <- emm(object = model.1)
plot(x = emm.1, within = time_f, between = treat, pch = c(15,17), lty = c(1,2),
     lcol = c("blue", "red"), pcol = c("blue","red"), )
## End(Not run)
```

`plot.clusbootptest` *Plot results of a permutation test*

Description

Plot results of a permutation test performed with ptest

Usage

```
## S3 method for class 'clusbootptest'
plot(x, pcol = "red", pty = 1, mfrow = c(1, 1), ...)
```

Arguments

`x`	object of class `clusbootptest`
`pcol`	color of vertical line indicating the observed Welch t test statistic
`pty`	type of vertical line indicating the observed Welch t test statistic
`mfrow`	vector of length 2 indicating the numbers of rows and columns in which the histograms will be drawn on the device.
`...`	other arguments to be passed into the `hist` function.

Author(s)

Mathijs Deen, Mark de Rooij

Examples

```
## Not run:
medication <- medication[medication$time %% 1 == 0,]
set.seed(1)
permtest.1 <- ptest(data = meds, outcome = pos, within = time, between = treat,
                    at.within = c(0,2,4,6), at.between = c(0,1), pn = 2000)
plot(permtest.1, pcol = "red", pty=2, mfrow = c(2,2), breaks="FD")
## End(Not run)
```

`predict.clusbootglm` *Predict method for cluster bootstrap GLM*

Description

Returns the predicted values for an `clusbootglm` object.

Usage

```
## S3 method for class 'clusbootglm'
predict(
  object,
  stat = mean,
  newdata = NULL,
  interval = FALSE,
  confint.level = NULL,
  keep.bootstrap.matrix = FALSE,
  ...
)
```

Arguments

`object`	Object of class `clusbootglm`.
`stat`	Center statistic of choice. Defaults to `mean`.
`newdata`	Optional data frame in which to look for variables with which to predict. If omitted, observations from the data value of the `clusbootglm` object are used.
`interval`	Boolean, indicating whether a confidence interval should be returned.
`confint.level`	Level of the confidence interval. Should be in [0, 1]. Defaults to .95 when `interval` = TRUE.
`keep.bootstrap.matrix`	Boolean, indicating whether the n * B bootstrap matrix should be returned. If TRUE, the return value for `predict.clusbootglm` becomes a list (see 'Value' below).
`...`	additional arguments passed to the function defined in the `stat` parameter.

Value

If `keep.bootstrap.matrix` is FALSE, `predict.clusbootglm` returns a matrix, containing the predicted values by evaluating the regression parameters in `newdata` (which defaults to the data value in `object`). If `keep.bootstrap.matrix` is TRUE, the function returns a list containing:

`predictions`	Matrix containing predicted values by evaluating the regression parameters in `object$data`.
`bootstrapmatrix`	A n * B matrix with the predictions within all bootstrap samples.

Author(s)

Mathijs Deen

Examples

```
## Not run:
medication <- medication[medication$time %% 1 == 0,]
medication$time <- as.factor(medication$time)
set.seed(1)
model.1 <- clusbootglm(pos~time*treat, clusterid = id, data = medication)
predict(object = model.1, interval = TRUE)
## End(Not run)
```

`ptest` *Permutation test for group differences at within-subject levels*

Description

Perform permutation tests for differences between two groups at given within-subject levels in a long-formatted dataframe

Usage

```
ptest(
  data,
  outcome,
  within,
  between,
  at.within,
  at.between,
  pn = 1000,
  progress.bar = TRUE
)
```

Arguments

`data`	dataframe that contains the data in long format.
`outcome`	outcome variable (i.e., the variable for which the difference should be tested).
`within`	within-subject variable.
`between`	between-subjects variable.
`at.within`	determine for which within-subject levels (e.g., which timepoint) the difference should be tested.
`at.between`	determine the groups in the difference test (should always be of length 2).
`pn`	the number of permutations that should be performed.
`progress.bar`	indicates whether a progress bar will be shown.

Details

In every permutation cycle, the outcome variable gets permutated and the Welch t test statistic is calculated.

Value

`ptest` produces an object of class `"clusbootptest"`, containing the following relevant components:

`perm.statistics`	A matrix of `length(at.within)` rows and pn columns, containing the Welch t-test statics for all permutations within the `at.within` level in the columns. The first column contains the t statistic for the observed data.
`pvalues`	Data frame containing the p values for every `at.within` level.

Author(s)

Mathijs Deen, Mark de Rooij

See Also

A useful method for the obtained `clusbootptest` class object is `plot.clusbootptest`.

Examples

```
## Not run:
meds <- medication[medication$time %% 1 == 0,]
set.seed(1)
permtest.1 <- ptest(data = meds, outcome = pos, within = time, between = treat,
                    at.within = c(0,2,4,6), at.between = c(0,1), pn = 2000)
permtest.1$pvalues
## End(Not run)
```

`summary.clusbootemm` *Summarize estimated marginal means for cluster bootstrap GLM into a grid*

Description

Returns the summary of the EMM for a `clusbootglm` class object.

Usage

```
## S3 method for class 'clusbootemm'
summary(object, ...)
```

Arguments

`object`	object of class `clusbootemm`.
`...`	other arguments.

Author(s)

Mathijs Deen

Examples

```
## Not run:
medication <- medication[medication$time %% 1 == 0,]
medication$time_f <- as.factor(medication$time)
set.seed(1)
model.1 <- clusbootglm(pos~time_f*treat, clusterid=id, data=medication)
emm.1 <- emm(object = model.1)
summary(object = emm.1)
## End(Not run)
```

`summary.clusbootglm` *Summarize output of cluster bootstrap GLM*

Description

Returns the summary of an object of class `clusbootglm`.

Usage

```
## S3 method for class 'clusbootglm'
summary(object, estimate.type = "bootstrap", interval.type = "BCa", ...)
```

Arguments

`object`	object of class `clusbootglm`.
`estimate.type`	specify which type of estimate should be returned, either bootstrap means (default) or GLM estimates from model fitted on original data.
`interval.type`	which confidence interval should be used. Options are `parametric`, `percentile`, and `BCa` intervals.
`...`	other arguments.

Author(s)

Mathijs Deen

Examples

```
## Not run:
data(opposites)
cbglm.1 <- clusbootglm(SCORE~Time*COG,data=opposites,clusterid=Subject)
summary(cbglm.1, interval.type="percentile")
## End(Not run)
```

Index