Package 'fwildclusterboot'

January 3, 2022

Title Fast Wild Cluster Bootstrap Inference for Linear Regression Models

Version 0.7

Date 2022-01-03

Description Implementation of the fast algorithm for wild cluster bootstrap inference developed in Roodman et al (2019, STATA Journal) for linear regression models <doi:10.1177/1536867X19830877>, which makes it feasible to quickly calculate bootstrap test statistics based on a large number of bootstrap draws even for large samples - as long as the number of bootstrapping clusters is not too large. Multiway clustering, regression weights, bootstrap weights, fixed effects and subcluster bootstrapping are supported. Further, both restricted (WCR) and unrestricted (WCU) bootstrap are supported. Methods are provided for a variety of fitted models, including 'lm()', 'feols()' (from package 'fixest') and 'felm()' (from package 'lfe').

URL https://s3alfisc.github.io/fwildclusterboot/

BugReports https://github.com/s3alfisc/fwildclusterboot/issues/

License GPL-3

Imports collapse, Formula, Rcpp, dreamerr, Matrix, Matrix.utils, generics, gtools, dqrng

Suggests fixest, lfe, data.table, fabricatr, tinytest, covr, knitr, rmarkdown, broom, modelsummary, RStata

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

LinkingTo Rcpp, RcppEigen

VignetteBuilder knitr

Language en-US

NeedsCompilation yes

Author Alexander Fischer [aut, cre],
David Roodman [aut],
Achim Zeileis [ctb] (Author of included sandwich fragments),
Nathaniel Graham [ctb] (Contributor to included sandwich fragments),
Susanne Koell [ctb] (Contributor to included sandwich fragments),
Laurent Berge [ctb] (Author of included fixest fragments),
Sebastian Krantz [ctb]

Maintainer Alexander Fischer <alexander-fischer1801@t-online.de>

Repository CRAN

Date/Publication 2022-01-03 19:00:02 UTC

R topics documented:

`.onLoad` *setting options for nthreads when package is loaded*

Description

setting options for nthreads when package is loaded

Usage

```
.onLoad(libname, pkgname)
```

Arguments

`libname`	library name
`pkgname`	package name

Value

Changes number of threads used.

`boottest` *Fast wild cluster bootstrap inference*

Description

`boottest` is a S3 method that allows for fast wild cluster bootstrap inference for objects of class lm, fixest and felm by implementing the fast wild bootstrap algorithm developed in Roodman et al., 2019.

Usage

```
boottest(object, ...)
```

Arguments

`object`	An object of type lm, fixest of felm
`...`	other arguments

Value

An object of class `boottest`.

References

Roodman et al., 2019, "Fast and wild: Bootstrap inference in STATA using boottest", The STATA Journal. (https://journals.sagepub.com/doi/full/10.1177/1536867X19830877)

Cameron, A. Colin, Jonah B. Gelbach, and Douglas L. Miller. "Bootstrap-based improvements for inference with clustered errors." The Review of Economics and Statistics 90.3 (2008): 414-427.

MacKinnon, James G., and Matthew D. Webb. "The wild bootstrap for few (treated) clusters." The Econometrics Journal 21.2 (2018): 114-135.

MacKinnon, James. "Wild cluster bootstrap confidence intervals." L'Actualite economique 91.1-2 (2015): 11-33.

Webb, Matthew D. Reworking wild bootstrap based inference for clustered errors. No. 1315. Queen's Economics Department Working Paper, 2013.

See Also

boottest.lm, boottest.fixest and boottest.felm

`boottest.felm` *Fast wild cluster bootstrap inference for object of class felm*

Description

`boottest.felm` is a S3 method that allows for fast wild cluster bootstrap inference for objects of class felm by implementing the fast wild bootstrap algorithm developed in Roodman et al., 2019 and implemented in the STATA package `boottest`.

Usage

```
## S3 method for class 'felm'
boottest(
  object,
  param,
  B,
  clustid,
  bootcluster = "max",
  fe = NULL,
  conf_int = NULL,
  seed = NULL,
  R = NULL,
  beta0 = 0,
  sign_level = NULL,
  type = "rademacher",
  impose_null = TRUE,
  p_val_type = "two-tailed",
  tol = 1e-06,
  maxiter = 10,
  na_omit = TRUE,
  nthreads = getBoottest_nthreads(),
  ssc = boot_ssc(adj = TRUE, fixef.K = "none", cluster.adj = TRUE, cluster.df =
    "conventional"),
  ...
)
```

Arguments

`object`	An object of class felm
`param`	A character vector of length one. The name of the regression coefficient for which the hypothesis is to be tested
`B`	Integer. The number of bootstrap iterations. When the number of clusters is low, increasing B adds little additional runtime.
`clustid`	A character vector containing the names of the cluster variables

`bootcluster` A character vector. Specifies the bootstrap clustering variable or variables. If more than one variable is specified, then bootstrapping is clustered by the intersections of clustering implied by the listed variables. To mimic the behavior of stata's boottest command, the default is to cluster by the intersection of all the variables specified via the `clustid` argument, even though that is not necessarily recommended (see the paper by Roodman et al cited below, section 4.2). Other options include "min", where bootstrapping is clustered by the cluster variable with the fewest clusters.

`fe` A character vector of length one which contains the name of the fixed effect to be projected out in the bootstrap. Note: if regression weights are used, fe needs to be NULL.

`conf_int` A logical vector. If TRUE, boottest computes confidence intervals by p-value inversion. If FALSE, only the p-value is returned.

`seed` An integer. Allows the user to set a random seed. If you want to set a "global" seed, set it via `dqrng::dqset.seed()`. For Mammen weights, you have to use `set.seed()` instead.

`R` Hypothesis Vector giving linear combinations of coefficients. Must be either NULL or a vector of the same length as `param`. If NULL, a vector of ones of length param.

`beta0` A numeric. Shifts the null hypothesis H0: param = beta0 vs H1: param != beta0

`sign_level` A numeric between 0 and 1 which sets the significance level of the inference procedure. E.g. sign_level = 0.05 returns 0.95% confidence intervals. By default, sign_level = 0.05.

`type` character or function. The character string specifies the type of boostrap to use: One of "rademacher", "mammen", "norm" and "webb". Alternatively, type can be a function(n) for drawing wild bootstrap factors. "rademacher" by default. For the Rademacher distribution, if the number of replications B exceeds the number of possible draw ombinations, 2^(#number of clusters), then `boottest()` will use each possible combination once (enumeration).

`impose_null` Logical. Controls if the null hypothesis is imposed on the bootstrap dgp or not. Null imposed (`WCR`) by default. If FALSE, the null is not imposed (`WCU`)

`p_val_type` Character vector of length 1. Type of p-value. By default "two-tailed". Other options include "equal-tailed", ">" and "<".

`tol` Numeric vector of length 1. The desired accuracy (convergence tolerance) used in the root finding procedure to find the confidence interval. 1e-6 by default.

`maxiter` Integer. Maximum number of iterations used in the root finding procedure to find the confidence interval. 10 by default.

`na_omit` Logical. If TRUE, `boottest()` omits rows with missing variables in the cluster variable that have not previously been deleted when fitting the regression object (e.g. if the cluster variable was not used when fitting the regression model).

`nthreads` The number of threads. Can be: a) an integer lower than, or equal to, the maximum number of threads; b) 0: meaning all available threads will be used; c) a number strictly between 0 and 1 which represents the fraction of all threads to use. The default is to use 1 core.

ssc	An object of class boot_ssc.type obtained with the function boot_ssc. Represents how the small sample adjustments are computed. The defaults are adj = TRUE, fixef.K = "none", cluster.adj = "TRUE", cluster.df = "conventional". You can find more details in the help file for boot_ssc(). The function is purposefully designed to mimic fixest's ssc function.
...	Further arguments passed to or from other methods.

Value

An object of class boottest

p_val	The bootstrap p-value.
conf_int	The bootstrap confidence interval.
param	The tested parameter.
N	Sample size. Might differ from the regression sample size if the cluster variables contain NA values.
B	Number of Bootstrap Iterations.
clustid	Names of the cluster Variables.
N_G	Dimension of the cluster variables as used in boottest.
sign_level	Significance level used in boottest.
type	Distribution of the bootstrap weights.
impose_null	Whether the null was imposed on the bootstrap dgp or not.
R	The vector "R" in the null hypothesis of interest Rbeta = beta0.
beta0	The scalar "beta0" in the null hypothesis of interest Rbeta = beta0.
point_estimate	R'beta. A scalar: the constraints vector times the regression coefficients.
p_test_vals	All p-values calculated while calculating the confidence interval.
t_stat	The 'original' regression test statistics.
test_vals	All t-statistics calculated while calculating the confidence interval.
t_boot	All bootstrap t-statistics.
regression	The regression object used in boottest.
call	Function call of boottest.

Confidence Intervals

boottest computes confidence intervals by inverting p-values. In practice, the following procedure is used:

- Based on an initial guess for starting values, calculate p-values for 26 equal spaced points between the starting values.
- Out of the 26 calculated p-values, find the two pairs of values x for which the corresponding p-values px cross the significance level sign_level.
- Feed the two pairs of x into an numerical root finding procedure and solve for the root. boottest currently relies on stats::uniroot and sets an absolute tolerance of 1e-06 and stops the procedure after 10 iterations.

Standard Errors

`boottest` does not calculate standard errors.

References

Roodman et al., 2019, "Fast and wild: Bootstrap inference in STATA using boottest", The STATA Journal. (https://journals.sagepub.com/doi/full/10.1177/1536867X19830877)

Cameron, A. Colin, Jonah B. Gelbach, and Douglas L. Miller. "Bootstrap-based improvements for inference with clustered errors." The Review of Economics and Statistics 90.3 (2008): 414-427.

MacKinnon, James G., and Matthew D. Webb. "The wild bootstrap for few (treated) clusters." The Econometrics Journal 21.2 (2018): 114-135.

MacKinnon, James. "Wild cluster bootstrap confidence intervals." L'Actualite economique 91.1-2 (2015): 11-33.

Webb, Matthew D. Reworking wild bootstrap based inference for clustered errors. No. 1315. Queen's Economics Department Working Paper, 2013.

Examples

```
if(requireNamespace("lfe")){
library(fwildclusterboot)
library(lfe)
data(voters)
felm_fit <- felm(proposition_vote ~ treatment + ideology1 + log_income
           | Q1_immigration,
           data = voters)
boot1 <- boottest(felm_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = "group_id1")
boot2 <- boottest(felm_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"))
boot3 <- boottest(felm_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"),
                  fe = "Q1_immigration")
boot4 <- boottest(felm_fit,
                  B = 999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"),
                  fe = "Q1_immigration",
                  sign_level = 0.2,
                  seed = 8,
                  beta0 = 2)
# test treatment + ideology1 = 2
boot5 <- boottest(felm_fit,
                  B = 9999,
```

```
                  clustid = c("group_id1", "group_id2"),
                  param = c("treatment", "ideology1"),
                  R = c(1, 1),
                  beta0 = 2)
summary(boot1)
plot(boot1)
}
```

---

boottest.fixest *Fast wild cluster bootstrap inference for object of class fixest*

Description

boottest.fixest is a S3 method that allows for fast wild cluster bootstrap inference for objects of class fixest by implementing the fast wild bootstrap algorithm developed in Roodman et al., 2019 and implemented in the STATA package boottest.

Usage

```
## S3 method for class 'fixest'
boottest(
  object,
  clustid,
  param,
  B,
  bootcluster = "max",
  fe = NULL,
  sign_level = NULL,
  conf_int = NULL,
  seed = NULL,
  R = NULL,
  beta0 = 0,
  type = "rademacher",
  impose_null = TRUE,
  p_val_type = "two-tailed",
  tol = 1e-06,
  maxiter = 10,
  na_omit = TRUE,
  nthreads = getBoottest_nthreads(),
  ssc = boot_ssc(adj = TRUE, fixef.K = "none", cluster.adj = TRUE, cluster.df =
    "conventional"),
  ...
)
```

Arguments

object	An object of class fixest

clustid	A character vector containing the names of the cluster variables
param	A character vector. The name of the regression coefficients for which the hypothesis is to be tested
B	Integer. The number of bootstrap iterations. When the number of clusters is low, increasing B adds little additional runtime.
bootcluster	A character vector. Specifies the bootstrap clustering variable or variables. If more than one variable is specified, then bootstrapping is clustered by the intersections of clustering implied by the listed variables. To mimic the behavior of stata's boottest command, the default is to cluster by the intersection of all the variables specified via the clustid argument, even though that is not necessarily recommended (see the paper by Roodman et al cited below, section 4.2). Other options include "min", where bootstrapping is clustered by the cluster variable with the fewest clusters.
fe	A character vector of length one which contains the name of the fixed effect to be projected out in the bootstrap. Note: if regression weights are used, fe needs to be NULL.
sign_level	A numeric between 0 and 1 which sets the significance level of the inference procedure. E.g. sign_level = 0.05 returns 0.95% confidence intervals. By default, sign_level = 0.05.
conf_int	A logical vector. If TRUE, boottest computes confidence intervals by p-value inversion. If FALSE, only the p-value is returned.
seed	An integer. Allows the user to set a random seed. If you want to set a "global" seed, set it via dqrng::dqset.seed(). For Mammen weights, you have to use set.seed() instead.
R	Hypothesis Vector giving linear combinations of coefficients. Must be either NULL or a vector of the same length as param. If NULL, a vector of ones of length param.
beta0	A numeric. Shifts the null hypothesis H0: param = beta0 vs H1: param != beta0
type	character or function. The character string specifies the type of boostrap to use: One of "rademacher", "mammen", "norm" and "webb". Alternatively, type can be a function(n) for drawing wild bootstrap factors. "rademacher" by default. For the Rademacher distribution, if the number of replications B exceeds the number of possible draw ombinations, 2^(#number of clusters), then boottest() will use each possible combination once (enumeration).
impose_null	Logical. Controls if the null hypothesis is imposed on the bootstrap dgp or not. Null imposed (WCR) by default. If FALSE, the null is not imposed (WCU)
p_val_type	Character vector of length 1. Type of p-value. By default "two-tailed". Other options include "equal-tailed", ">" and "<".
tol	Numeric vector of length 1. The desired accuracy (convergence tolerance) used in the root finding procedure to find the confidence interval. 1e-6 by default.
maxiter	Integer. Maximum number of iterations used in the root finding procedure to find the confidence interval. 10 by default.
na_omit	Logical. If TRUE, boottest() omits rows with missing variables in the cluster variable that have not previously been deleted when fitting the regression object (e.g. if the cluster variable was not used when fitting the regression model).

nthreads	The number of threads. Can be: a) an integer lower than, or equal to, the maximum number of threads; b) 0: meaning all available threads will be used; c) a number strictly between 0 and 1 which represents the fraction of all threads to use. The default is to use 1 core.
ssc	An object of class boot_ssc.type obtained with the function boot_ssc. Represents how the small sample adjustments are computed. The defaults are adj = TRUE, fixef.K = "none", cluster.adj = "TRUE", cluster.df = "conventional". You can find more details in the help file for boot_ssc(). The function is purposefully designed to mimic fixest's ssc function.
...	Further arguments passed to or from other methods.

Value

An object of class boottest

p_val	The bootstrap p-value.
conf_int	The bootstrap confidence interval.
param	The tested parameter.
N	Sample size. Might differ from the regression sample size if the cluster variables contain NA values.
B	Number of Bootstrap Iterations.
clustid	Names of the cluster Variables.
N_G	Dimension of the cluster variables as used in boottest.
sign_level	Significance level used in boottest.
type	Distribution of the bootstrap weights.
impose_null	Whether the null was imposed on the bootstrap dgp or not.
R	The vector "R" in the null hypothesis of interest Rbeta = beta0.
beta0	The scalar "beta0" in the null hypothesis of interest Rbeta = beta0.
point_estimate	R'beta. A scalar: the constraints vector times the regression coefficients.
p_test_vals	All p-values calculated while calculating the confidence interval.
t_stat	The 'original' regression test statistics.
test_vals	All t-statistics calculated while calculating the confidence interval.
t_boot	All bootstrap t-statistics.
regression	The regression object used in boottest.
call	Function call of boottest.

Confidence Intervals

boottest computes confidence intervals by inverting p-values. In practice, the following procedure is used:

- Based on an initial guess for starting values, calculate p-values for 26 equal spaced points between the starting values.

- Out of the 26 calculated p-values, find the two pairs of values x for which the corresponding p-values px cross the significance sign_level sign_level.
- Feed the two pairs of x into an numerical root finding procedure and solve for the root. boottest currently relies on `stats::uniroot` and sets an absolute tolerance of 1e-06 and stops the procedure after 10 iterations.

Standard Errors

`boottest` does not calculate standard errors.

References

Roodman et al., 2019, "Fast and wild: Bootstrap inference in STATA using boottest", The STATA Journal. (https://journals.sagepub.com/doi/full/10.1177/1536867X19830877)

Cameron, A. Colin, Jonah B. Gelbach, and Douglas L. Miller. "Bootstrap-based improvements for inference with clustered errors." The Review of Economics and Statistics 90.3 (2008): 414-427.

MacKinnon, James G., and Matthew D. Webb. "The wild bootstrap for few (treated) clusters." The Econometrics Journal 21.2 (2018): 114-135.

MacKinnon, James. "Wild cluster bootstrap confidence intervals." L'Actualite economique 91.1-2 (2015): 11-33.

Webb, Matthew D. Reworking wild bootstrap based inference for clustered errors. No. 1315. Queen's Economics Department Working Paper, 2013.

Examples

```
if(requireNamespace("fixest")){
library(fwildclusterboot)
library(fixest)
data(voters)
feols_fit <-feols(proposition_vote ~ treatment + ideology1 + log_income,
           fixef =  "Q1_immigration",
           data = voters)
boot1 <- boottest(feols_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = "group_id1")
boot2 <- boottest(feols_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"))
boot3 <- boottest(feols_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"),
                  fe = "Q1_immigration")
boot4 <- boottest(feols_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"),
                  fe = "Q1_immigration",
```

```
                  sign_level = 0.2,
                  seed = 8,
                  beta0 = 2)
# test treatment + ideology1 = 2
boot5 <- boottest(feols_fit,
                  B = 9999,
                  clustid = c("group_id1", "group_id2"),
                  param = c("treatment", "ideology1"),
                  R = c(1, 1),
                  beta0 = 2)
summary(boot1)
plot(boot1)
}
```

`boottest.lm` *Fast wild cluster bootstrap inference for object of class lm*

Description

`boottest.lm` is a S3 method that allows for fast wild cluster bootstrap inference for objects of class lm by implementing the fast wild bootstrap algorithm developed in Roodman et al., 2019.

Usage

```
## S3 method for class 'lm'
boottest(
  object,
  clustid,
  param,
  B,
  bootcluster = "max",
  conf_int = NULL,
  seed = NULL,
  R = NULL,
  beta0 = 0,
  sign_level = NULL,
  type = "rademacher",
  impose_null = TRUE,
  p_val_type = "two-tailed",
  tol = 1e-06,
  maxiter = 10,
  na_omit = TRUE,
  nthreads = getBoottest_nthreads(),
  ssc = boot_ssc(adj = TRUE, fixef.K = "none", cluster.adj = TRUE, cluster.df =
    "conventional"),
  ...
)
```

Arguments

object	An object of class lm
clustid	A character vector containing the names of the cluster variables
param	A character vector of length one. The name of the regression coefficient for which the hypothesis is to be tested
B	Integer. The number of bootstrap iterations. When the number of clusters is low, increasing B adds little additional runtime.
bootcluster	A character vector. Specifies the bootstrap clustering variable or variables. If more than one variable is specified, then bootstrapping is clustered by the intersections of clustering implied by the listed variables. To mimic the behavior of stata's boottest command, the default is to cluster by the intersection of all the variables specified via the clustid argument, even though that is not necessarily recommended (see the paper by Roodman et al cited below, section 4.2). Other options include "min", where bootstrapping is clustered by the cluster variable with the fewest clusters.
conf_int	A logical vector. If TRUE, boottest computes confidence intervals by p-value inversion. If FALSE, only the p-value is returned.
seed	An integer. Allows the user to set a random seed. If you want to set a "global" seed, set it via dqrng::dqset.seed(). For Mammen weights, you have to use set.seed() instead.
R	Hypothesis Vector giving linear combinations of coefficients. Must be either NULL or a vector of the same length as param. If NULL, a vector of ones of length param.
beta0	A numeric. Shifts the null hypothesis H0: param = beta0 vs H1: param != beta0
sign_level	A numeric between 0 and 1 which sets the significance level of the inference procedure. E.g. sign_level = 0.05 returns 0.95% confidence intervals. By default, sign_level = 0.05.
type	character or function. The character string specifies the type of boostrap to use: One of "rademacher", "mammen", "norm" and "webb". Alternatively, type can be a function(n) for drawing wild bootstrap factors. "rademacher" by default. For the Rademacher distribution, if the number of replications B exceeds the number of possible draw ombinations, 2^(#number of clusters), then boottest() will use each possible combination once (enumeration).
impose_null	Logical. Controls if the null hypothesis is imposed on the bootstrap dgp or not. Null imposed (WCR) by default. If FALSE, the null is not imposed (WCU)
p_val_type	Character vector of length 1. Type of p-value. By default "two-tailed". Other options include "equal-tailed", ">" and "<".
tol	Numeric vector of length 1. The desired accuracy (convergence tolerance) used in the root finding procedure to find the confidence interval. 1e-6 by default.
maxiter	Integer. Maximum number of iterations used in the root finding procedure to find the confidence interval. 10 by default.
na_omit	Logical. If TRUE, boottest() omits rows with missing variables in the cluster variable that have not previously been deleted when fitting the regression object (e.g. if the cluster variable was not used when fitting the regression model).

nthreads	The number of threads. Can be: a) an integer lower than, or equal to, the maximum number of threads; b) 0: meaning all available threads will be used; c) a number strictly between 0 and 1 which represents the fraction of all threads to use. The default is to use 1 core.
ssc	An object of class boot_ssc.type obtained with the function boot_ssc. Represents how the small sample adjustments are computed. The defaults are adj = TRUE, fixef.K = "none", cluster.adj = "TRUE", cluster.df = "conventional". You can find more details in the help file for boot_ssc(). The function is purposefully designed to mimic fixest's ssc function.
...	Further arguments passed to or from other methods.

Value

An object of class boottest

p_val	The bootstrap p-value.
conf_int	The bootstrap confidence interval.
param	The tested parameter.
N	Sample size. Might differ from the regression sample size if the cluster variables contain NA values.
B	Number of Bootstrap Iterations.
clustid	Names of the cluster Variables.
N_G	Dimension of the cluster variables as used in boottest.
sign_level	Significance level used in boottest.
type	Distribution of the bootstrap weights.
impose_null	Whether the null was imposed on the bootstrap dgp or not.
R	The vector "R" in the null hypothesis of interest Rbeta = beta0.
beta0	The scalar "beta0" in the null hypothesis of interest Rbeta = beta0.
point_estimate	R'beta. A scalar: the constraints vector times the regression coefficients.
p_test_vals	All p-values calculated while calculating the confidence interval.
t_stat	The 'original' regression test statistics.
test_vals	All t-statistics calculated while calculating the confidence interval.
t_boot	All bootstrap t-statistics.
regression	The regression object used in boottest.
call	Function call of boottest.

Confidence Intervals

boottest computes confidence intervals by inverting p-values. In practice, the following procedure is used:

- Based on an initial guess for starting values, calculate p-values for 26 equal spaced points between the starting values.

- Out of the 26 calculated p-values, find the two pairs of values x for which the corresponding p-values px cross the significance level sign_level.
- Feed the two pairs of x into an numerical root finding procedure and solve for the root. boottest currently relies on `stats::uniroot` and sets an absolute tolerance of 1e-06 and stops the procedure after 10 iterations.

Standard Errors

`boottest` does not calculate standard errors.

References

Roodman et al., 2019, "Fast and wild: Bootstrap inference in STATA using boottest", The STATA Journal. (https://journals.sagepub.com/doi/full/10.1177/1536867X19830877)

Cameron, A. Colin, Jonah B. Gelbach, and Douglas L. Miller. "Bootstrap-based improvements for inference with clustered errors." The Review of Economics and Statistics 90.3 (2008): 414-427.

MacKinnon, James G., and Matthew D. Webb. "The wild bootstrap for few (treated) clusters." The Econometrics Journal 21.2 (2018): 114-135.

MacKinnon, James. "Wild cluster bootstrap confidence intervals." L'Actualite economique 91.1-2 (2015): 11-33.

Webb, Matthew D. Reworking wild bootstrap based inference for clustered errors. No. 1315. Queen's Economics Department Working Paper, 2013.

Examples

```
library(fwildclusterboot)
data(voters)
lm_fit <-lm(proposition_vote ~ treatment + ideology1 + log_income + Q1_immigration,
         data = voters)
boot1 <- boottest(lm_fit,
                  B = 9999,
                  param = "treatment",
                   clustid = "group_id1")
boot2 <- boottest(lm_fit,
                  B = 9999,
                  param = "treatment",
                 clustid = c("group_id1", "group_id2"))
boot3 <- boottest(lm_fit,
                  B = 9999,
                  param = "treatment",
                  clustid = c("group_id1", "group_id2"),
                  sign_level = 0.2,
                  seed = 8,
                  beta0 = 2)
# test treatment + ideology1 = 2
boot4 <- boottest(lm_fit,
                  B = 9999,
                  clustid = c("group_id1", "group_id2"),
                  param = c("treatment", "ideology1"),
                  R = c(1, 1),
```

```
                  beta0 = 2)
summary(boot1)
plot(boot1)
```

---

boot_algo2 *Fast wild cluster bootstrap algorithm*

Description

function that implements the fast bootstrap algorithm as described in Roodman et al (2019)

Usage

```
boot_algo2(
  preprocessed_object,
  boot_iter,
  point_estimate,
  impose_null,
  beta0,
  sign_level,
  param,
  p_val_type,
  nthreads,
  type,
  full_enumeration,
  small_sample_correction
)
```

Arguments

preprocessed_object	A list: output of the preprocess2 function.
boot_iter	number of bootstrap iterations
point_estimate	The point estimate of the test parameter from the regression model.
impose_null	If TRUE, the null is not imposed on the bootstrap distribution. This is what Roodman et al call the "WCU" bootstrap. With impose_null = FALSE, the null is imposed ("WCR").
beta0	Shifts the null hypothesis.
sign_level	The significance level.
param	name of the test parameter.
p_val_type	type Type of p-value. By default "two-tailed". Other options: "equal-tailed", ">", "<"
nthreads	The number of threads. Can be: a) an integer lower than, or equal to, the maximum number of threads; b) 0: meaning all available threads will be used; c) a number strictly between 0 and 1 which represents the fraction of all threads to use. The default is to use 50\ set permanently the number of threads used within this package using the function ...

type character or function. The character string specifies the type of boostrap to use: One of "rademacher", "mammen", "norm" and "webb". Alternatively, type can be a function(n) for drawing wild bootstrap factors. "rademacher" by default.

full_enumeration
Is full enumeration employed? Full enum. is used if N_G^2 < boot_iter for Mammen and Rademacher weights

small_sample_correction
The small sample correction to be applied. See ssc().

Value

A list of ...

---

boot_ssc *set the small sample correction factor applied in* bootest()

Description

set the small sample correction factor applied in bootest()

Usage

```
boot_ssc(
  adj = TRUE,
  fixef.K = "none",
  cluster.adj = TRUE,
  cluster.df = "conventional"
)
```

Arguments

adj Logical scalar, defaults to TRUE. If TRUE, applies a small sample correction of (N-1) / (N-k) where N is the number of observations and k is the number of estimated coefficients excluding any fixed effects projected out in either fixest::feols() or lfe::felm().

fixef.K Character scalar, equal to 'none': the fixed effects parameters are discarded when calculating k in (N-1) / (N-k).

cluster.adj Logical scalar, defaults to TRUE. If TRUE, a cluster correction G/(G-1) is performed, with G the number of clusters.

cluster.df Either "conventional" or "min" default. Controls how "G" is computed for multiway clustering if cluster.adj = TRUE. Note that the covariance matrix in the multiway clustering case is of the form V = V_1 + V_2 - V_12. If "conventional", then each summand G_i is multiplied with a small sample adjustment G_i / (G_i - 1). If "min", all summands are multiplied with the same value, min(G) / (min(G) - 1)

Examples

```
boot_ssc(adj = TRUE, cluster.adj = TRUE)
boot_ssc(adj = TRUE, cluster.adj = TRUE, cluster.df = "min")
```

`cpp_get_nb_threads` *Get maximum number of threads on hardware for open mp support*

Description

Get maximum number of threads on hardware for open mp support

Usage

```
cpp_get_nb_threads()
```

Value

The maximum number of threads supported.

`create_data` *Simulate Data*

Description

Function simulates data for tests and examples with clustering variables and fixed-effects.

Usage

```
create_data(N, N_G1, icc1, N_G2, icc2, numb_fe1, numb_fe2, seed, weights)
```

Arguments

`N`	number of observations
`N_G1`	A scalar. number of clusters for clustering variable 1
`icc1`	A scalar between 0 and 1. intra-cluster correlation for clustering variable 1
`N_G2`	A scalar. number of clusters for clustering variable 2
`icc2`	A scalar between 0 and 1. intra-cluster correlation for clustering variable 2
`numb_fe1`	A scalar. Number of fixed effect for first factor variable
`numb_fe2`	A scalar. Number of fixed effect for second factor variable
`seed`	An integer. Set the random seed
`weights`	Possible regression weights to be used in estimation

Value

A simulated `data.frame` with specified numbers of clusters, intra-cluster correlations and dimensionality of fixed effects.

eigenMapMatMult *Matrix Multiplication via Eigen*

Description

Matrix Multiplication via Eigen

Usage

```
eigenMapMatMult(A, B, nthreads)
```

Arguments

A	A matrix.
B	A matrix.
nthreads	Integer. Number of threads to use for matrix multiplication.

Value

A matrix

get_ssc *Compute small sample adjustment factors*

Description

Compute small sample adjustment factors

Usage

```
get_ssc(boot_ssc_object, N, k, G, vcov_sign)
```

Arguments

boot_ssc_object	An object of type 'boot_ssc.type'
N	The number of observations
k	The number of estimated parameters
G	The number of clusters
vcov_sign	A vector that helps create the covariance matrix

Value

A small sample adjustment factor

glance.boottest *S3 method to glance at objects of class boottest*

Description

S3 method to glance at objects of class boottest

Usage

```
## S3 method for class 'boottest'
glance(x, ...)
```

Arguments

x	object of type boottest
...	Further arguments passed to or from other methods.

Value

A single row summary "glance" of an object of type boottest - lists characteristics of the input regression model

plot.boottest *Plot the bootstrap distribution of t-statistics*

Description

Plot the bootstrap distribution of t-statistics

Usage

```
## S3 method for class 'boottest'
plot(x, ...)
```

Arguments

x	An object of type boottest
...	Further arguments passed to or from other methods.

Value

A plot of bootstrap t-statistics under different null hypotheses

summary.boottest *S3 method to summarize objects of class boottest*

Description

S3 method to summarize objects of class boottest

Usage

```
## S3 method for class 'boottest'
summary(object, digits = 3, ...)
```

Arguments

object	object of type boottest
digits	rounding of output. 3 by default
...	Further arguments passed to or from other methods.

Value

Returns result summaries for objects of type boottest

tidy.boottest *S3 method to summarize objects of class boottest into tidy data.frame*

Description

S3 method to summarize objects of class boottest into tidy data.frame

Usage

```
## S3 method for class 'boottest'
tidy(object, ...)
```

Arguments

object	object of type boottest
...	Further arguments passed to or from other methods.

Value

A tidy data.frame with estimation results for objects of type boottest

`voters` *Random example data set*

Description

Random example data set

Usage

```
data(voters)
```

Format

An object of class `data.frame` with 300 rows and 13 columns.

Examples

```
data(voters)
```

Index