Package ‘wildrwolf’

March 2, 2023

Type Package

Title Fast Computation of Romano-Wolf Corrected p-Values for Linear Regression Models

Version 0.6.1

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

Description Fast Routines to Compute Romano-Wolf corrected p-Values (Romano and Wolf (2005a) <DOI:10.1198/016214504000000539>, Romano and Wolf (2005b) <DOI:10.1111/j.1468-0262.2005.00615.x>) for objects of type 'fixest' and 'fixest_multi' from the 'fixest' package via a wild (cluster) bootstrap.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.1

Imports fixest, fwildclusterboot, dreamerr, fabricatr, MASS

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

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

Suggests covr, testthat (>= 3.0.0), rmarkdown, ggplot2, RStata, microbenchmark

Config/testthat/edition 3

NeedsCompilation no

Author Alexander Fischer [aut, cre]

Repository CRAN

Date/Publication 2023-03-02 12:20:02 UTC

R topics documented:

fwer_sim — *Simulate data as in Clarke, Romano & Wolf (2019) to simulate family wise error rates (FWERs)*

Description

Simulate data as in Clarke, Romano & Wolf (2019) to simulate family wise error rates (FWERs)

Usage

```
fwer_sim(rho, N, s, B, G = 20)
```

Arguments

`rho` The correlation between the outcome variables

`N` The number of observations

`s` The number of dependent variables

`B` The number of bootstrap draws e

`G` The number of clusters. If NULL, no clustering.

Value

A 'data.frame' containing unadjusted p-values & p-values adjusted using the methods by Holm and Romano & Wolf (2005), with the following columns

get_rwolf_pval — *compute Romano-Wolf adjusted p-values based on bootstrapped t-statistics*

Description

compute Romano-Wolf adjusted p-values based on bootstrapped t-statistics

Usage

```
get_rwolf_pval(t_stats, boot_t_stats)
```

Arguments

`t_stats` A vector of length S - where S is the number of tested hypotheses - containing the original, non-bootstrappe t-statisics

`boot_t_stats` A (B x S) matrix containing the bootstrapped t-statistics

Value

A vector of Romano-Wolf corrected p-values

run_fwer_sim *Family Wise Error Rate Simulations*

Description

Run a MC simulation study on family-wise error rates (FWERs) for the Holm and Romano & Wolf Methods multiple hypothesis adjustment methods given true null effects

Usage

```
run_fwer_sim(
  n_sims = 100,
  rho = c(0, 0.25, 0.5, 0.75),
  seed = 114411,
  B = 499,
  N = 1000,
  s = 6,
  G = 20
)
```

Arguments

n_sims	The number of Monte Carlo iterations. 100 by default.
rho	The correlation between the outcome variables. Vectorized c(0, 0.25, 0.5, .75) by default
seed	A random seed.
B	The number of bootstrap draws. 499 by default.
N	The number of observations. 1000 by default.
s	The number of dependent variables. 6 by default.
G	The number of clusters. If NULL, no clustering. 20 by default

Value

A data frame containing familiy wise rejection rates for uncorrected pvalues and corrected pvalues using Holm's and the Romano-Wolf method.

reject_5	The family wise rejection rate at a 5% level
reject_10	The family wise rejection rate at a 10% level
rho	The correlation between the outcome variables. See function argument‘rho‘ for more information.

Examples

```
# N, B, n_sims, chosen so that the example runs quicker
# for a higher quality simulation, increase all values
res <- run_fwer_sim(
  seed = 123,
  n_sims = 10,
  B = 199,
  N = 100,
  s = 10,
  rho = 0
)
```

rwolf *Romano-Wolf multiple hypotheses adjusted p-values*

Description

Function implements the Romano-Wolf multiple hypothesis correction procedure for objects of type ‘fixest_multi‘ (‘fixest_multi‘ are objects created by ‘fixest::feols()‘ that use ‘feols()‘ multiple-estimation interface). The null hypothesis is always imposed on the bootstrap dgp.

Usage

```
rwolf(
  models,
  param,
  B,
  R = NULL,
  r = 0,
  p_val_type = "two-tailed",
  weights_type = "rademacher",
  engine = "R",
  nthreads = 1,
  bootstrap_type = "fnw11",
  ...
)
```

Arguments

models	An object of type ‘fixest_multi‘ or a list of objects of type ‘fixest‘, estimated via ordinary least squares (OLS)
param	The regression parameter to be tested
B	The number of bootstrap iterations

`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.

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

`p_val_type` Character vector of length 1. Type of hypothesis test By default "two-tailed". Other options include "equal-tailed" (for one-sided tests), ">" and "<" (for two-sided tests).

`weights_type` character or function. The character string specifies the type of bootstrap 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).

`engine` Should the wild cluster bootstrap run via ‘fwildclusterboot's‘ R implementation or via ‘WildBootTests.jl‘? 'R' by default. The other option is ‘WildBootTests.jl‘. Running the bootstrap through ‘WildBootTests.jl‘ might significantly reduce the runtime of ‘rwolf()‘ for complex problems (e.g. problems with more than 500 clusters).

`nthreads` Integer. The number of threads to use when running the bootstrap.

`bootstrap_type` Either "11", "13", "31", "33", or "fnw11". "fnw11" by default. See ‘?fwildclusterboot::boottest‘ for more details

`...` additional function values passed to the bootstrap function.

Value

A data.frame containing the following columns:

`model` Index of Models

`Estimate` The estimated coefficient of ‘param‘ in the respective model.

`Std. Error` The estimated standard error of ‘param‘ in the respective model.

`t value` The t statistic of ‘param‘ in the respective model.

`Pr(>|t|)` The uncorrected pvalue for ‘param‘ in the respective model.

`RW Pr(>|t|)` The Romano-Wolf corrected pvalue of hypothesis test for ‘param‘ in the respective model.

Setting Seeds and Random Number Generation

To guarantee reproducibility, please set a global random seeds via ‘set.seed()‘.

References

Clarke, Romano & Wolf (2019), STATA Journal. IZA working paper: https://ftp.iza.org/dp12845.pdf

Examples

```
library(fixest)
library(wildrwolf)

set.seed(12345)

N <- 1000
X1 <- rnorm(N)
Y1 <- 1 + 1 * X1 + rnorm(N)
Y2 <- 1 + 0.01 * X1 + rnorm(N)
Y3 <- 1 + 0.01 * X1 + rnorm(N)
Y4 <- 1 + 0.01 * X1 + rnorm(N)

B <- 999
# intra-cluster correlation of 0 for all clusters
cluster <- rep(1:50, N / 50)

data <- data.frame(Y1 = Y1,
                   Y2 = Y2,
                   Y3 = Y3,
                   Y4 = Y4,
                   X1 = X1,
                   cluster = cluster)

res <- feols(c(Y1, Y2, Y3) ~ X1, data = data, cluster = ~ cluster)
res_rwolf <- rwolf(models = res, param = "X1", B = B)
res_rwolf
```

Index