Package: EquiTrends (via r-universe)

October 29, 2026

Type Package

Title Equivalence Testing for Pre-Trends in Difference-in-Differences Designs

Version 1.0.0

Maintainer Ties Bos <tc.bos@student.maastrichtuniversity.nl>

Description Testing for parallel trends is crucial in the Difference-in-Differences framework. To this end, this package performs equivalence testing in the context of Difference-in-Differences estimation. It allows users to test if pre-treatment trends in the treated group are "equivalent" to those in the control group. Here, "equivalence" means that rejection of the null hypothesis implies that a function of the pre-treatment placebo effects (maximum absolute, average or root mean squared value) does not exceed a pre-specified threshold below which trend differences are considered negligible. The package is based on the theory developed in Dette & Schumann (2024) <doi:10.1080/07350015.2024.2308121>.

License MIT + file LICENSE

Imports clubSandwich, nloptr, dplyr, rlang, plm, Rcpp (>= 1.0.12), RcppParallel, stats, VGAM

LinkingTo Rcpp, RcppArmadillo, RcppParallel

URL https://github.com/TiesBos/EquiTrends

BugReports https://github.com/TiesBos/EquiTrends/issues

RoxygenNote 7.3.2

Encoding UTF-8

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Ties Bos [aut, cre], Martin Schumann [ctb]

Depends R (>= 3.5.0)

Repository https://cran.r-universe.dev

Date/Publication 2024-09-04 02:51:14 UTC

RemoteUrl https://github.com/cran/EquiTrends

RemoteRef HEAD

RemoteSha da1101b9537ed394309a8a8cc9705105e277256e

Contents

EquiTrends-package 2
boot_optimization_function 3
EquiTrends_dataconstr 4
EquiTrends_inputcheck 5
maxEquivTest 6
maxTest_error 12
maxTestBoot_func 12
maxTestIU_func 14
maxTestIU_optim_func 16
meanEquivTest 17
meanTest_func 21
meanTest_optim_func 23
print.maxEquivTestBoot 24
print.maxEquivTestIU 24
print.meanEquivTest 25
print.rmsEquivTest 25
rmsEquivTest 26
rmsTest_error 30
rmsTest_func 30
sigma_hathat_c 32
sim_check 33
sim_paneldata 34
W_critical_value 35

Index **36**

`EquiTrends-package` *Equivalence Testing for Pre-Trends in Difference-in-Differences Designs*

Description

Testing for parallel trends is crucial in the Difference-in-Difference framework. EquiTrends is an R package for equivalence testing in the context of Difference-in-Differences estimation. It allows users to test if pre-treatment trends in the treated group are "equivalent" to those in the control group. Here, "equivalence" means that rejection of the null hypothesis implies that a function of the pre-treatment placebo effects (maximum absolute, average or root mean squared value) does not exceed a pre-specified threshold below which trend differences are considered

negligible. The package is based on the theory developed in Dette & Schumann (2024) <doi: 10.1080/07350015.2024.2308121>.

Details

The package contains the functions maxEquivTest to perform the testing procedure surrounding the maximum placebo coefficient (see equation (3.1) of Dette & Schumann (2024)), meanEquivTest to perform the testing procedure surrounding the mean placebo coefficient (see equation (3.2) of Dette & Schumann (2024)) and rmsEquivTest to perform the testing procedure surrounding the root mean squared placebo coefficient (see equation (3.3) and (3.4) of Dette & Schumann (2024)). Furthermore, the package contains the function sim_paneldata to simulate a paneldataset for such testing purposes.

Author(s)

Maintainer: Ties Bos <tc.bos@student.maastrichtuniversity.nl>

References

Dette H., & Schumann M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: [10.1080/07350015.2024.2308121](https://doi.org/10

`boot_optimization_function` *Finding the restricted placebo coefficients for the maximum equivalence test based on the bootstrap approaches*

Description

`boot_optimization_function` solves the optimization problem to find the restricted placebo coefficients, according to Dette & Schumann (2024).

Usage

```
boot_optimization_function(x, y, no_placebos, equiv_threshold, start_val)
```

Arguments

`x`	The double demeaned independent variables.
`y`	The double demeaned dependent variable.
`no_placebos`	The number of placebo coefficients.
`equiv_threshold`	The equivalence threshold for the test.
`start_val`	The starting values for the optimization.

Value

A numeric vector containing the restricted placebo coefficients

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

`EquiTrends_dataconstr` *Data Construction Function for EquiTrends*

Description

Data Construction Function for EquiTrends

Usage

```
EquiTrends_dataconstr(
  Y,
  ID,
  G,
  period,
  X,
  data,
  pretreatment_period,
  base_period,
  cluster
)
```

Arguments

`Y`	see maxEquivTest, meanEquivTest or rmsEquivTest
`ID`	see maxEquivTest, meanEquivTest or rmsEquivTest
`G`	see maxEquivTest, meanEquivTest or rmsEquivTest
`period`	see maxEquivTest, meanEquivTest or rmsEquivTest
`X`	see maxEquivTest, meanEquivTest or rmsEquivTest
`data`	see maxEquivTest, meanEquivTest or rmsEquivTest
`pretreatment_period`	see maxEquivTest, meanEquivTest or rmsEquivTest
`base_period`	see maxEquivTest, meanEquivTest or rmsEquivTest
`cluster`	see maxEquivTest, meanEquivTest or rmsEquivTest

Value

A list containing the structured data.frame object used in the equivalence testing procedures, the base period for the test, a logical value indicating whether the panel is balanced and the number of periods.

`EquiTrends_inputcheck` *Input Checks Function for EquiTrends*

Description

Input Checks Function for EquiTrends

Usage

```
EquiTrends_inputcheck(
  Y,
  ID,
  G,
  period,
  X,
  data,
  equiv_threshold,
  pretreatment_period,
  base_period,
  cluster,
  alpha
)
```

Arguments

`Y`	see maxEquivTest, meanEquivTest or rmsEquivTest
`ID`	see maxEquivTest, meanEquivTest or rmsEquivTest
`G`	see maxEquivTest, meanEquivTest or rmsEquivTest
`period`	see maxEquivTest, meanEquivTest or rmsEquivTest
`X`	see maxEquivTest, meanEquivTest or rmsEquivTest
`data`	see maxEquivTest, meanEquivTest or rmsEquivTest
`equiv_threshold`	see maxEquivTest, meanEquivTest or rmsEquivTest
`pretreatment_period`	see maxEquivTest, meanEquivTest or rmsEquivTest
`base_period`	see maxEquivTest, meanEquivTest or rmsEquivTest
`cluster`	see maxEquivTest, meanEquivTest or rmsEquivTest
`alpha`	see maxEquivTest, meanEquivTest or rmsEquivTest

Value

A list containing an error indicator and a message. If `error` is TRUE, `message` contains an error message. If `error` is FALSE, `message` is empty.

maxEquivTest *Equivalence Test for Pre-trends based on the Maximum Absolute Placebo Coefficient*

Description

This function performs an equivalence test for pre-trends based on the maximum absolute placebo coefficient from Dette & Schumann (2024). The test can be performed using the intersection-union approach (IU), a bootstrap procedure for spherical errors (Boot) and a wild bootstrap procedure (Wild).

Usage

```
maxEquivTest(
  Y,
  ID,
  G,
  period,
  X = NULL,
  data = NULL,
  equiv_threshold = NULL,
  pretreatment_period = NULL,
  base_period = NULL,
  type = c("IU", "Boot", "Wild"),
  vcov = NULL,
  cluster = NULL,
  alpha = 0.05,
  B = 1000
)
```

Arguments

Y	A numeric vector with the variable of interest. If data is supplied, Y should be a scalar indicating the column number or column-name character string that corresponds to the numeric dependent (outcome) variable in 'data'.
ID	A numeric vector identifying the different cross-sectional units in the dataset. If data is supplied, ID should be a scalar indicating the column number or column-name character string that corresponds to the cross-sectional units identifier in data.
G	A binary or logic vector (of the same dimension as Y and ID) indicating if the individual (e.g. as indicated by ID) receives treatment (e.g. 1 or TRUE) or not (0 or FALSE). If 'data' is supplied, G should be a scalar identifying the column number or column-name character string associated to G in data.
period	A numeric vector (of the same dimension as Y) indicating time. If data is supplied, period should be a scalar indicating the column number or column-name character string that corresponds to the time identifier in data.

`X` A vector, matrix, or data.frame containing the control variables. If `data` is supplied, `X` must be a vector of column numbers or column-name character strings that identifies the control variables' columns.

`data` An optional `data.frame` object containing the variables in Y, ID, G, T and, if supplied, X and cluster as its columns.

`equiv_threshold`
The scalar equivalence threshold (must be positive). The default is NULL, implying that the function must look for the minimum value for which the null hypothesis of "non-negligible differences" can still be rejected.

`pretreatment_period`
A numeric vector identifying the pre-treatment periods that should be used for testing. `pretreatment_period` must be a subset of the periods included through `period`. The default is to use all periods that are included in `period`.

`base_period` The pre-treatment period to compare the post-treatment observation to. The default is to take the last period of the pre-treatment period.

`type` The type of maximum test that should be performed. "IU" for the intersection-union test, "Boot" for the regular bootstrap procedure from Dette & Schumann (2024) and "Wild" for the Wild bootstrap procedure.

`vcov` If `type = "IU"`, the variance-covariance matrix that needs to be used. See *Details* for more details.

`cluster` If `vcov = "CL"`, a vector indicating which observations belong to the same cluster. `cluster` must be of the same length as the panel. If `data` is supplied, `cluster` must be either the column index or column name of this vector in the data.frame/matrix. The default (`cluster=NULL`) assumes every unit in ID is its own cluster. Only required if `vcov = "CL"` and `type = "IU"`.

`alpha` Significance level of the test. The default is 0.05. Only required if `equiv_threshold` is not specified.

`B` If type = Boot or type = Wild, the number of bootstrap samples used. The default is 1000.

Details

The `vcov` parameter specifies the variance-covariance matrix to be used in the function for `type = "IU"`. This parameter can take two types of inputs:

1. A character string specifying the type of variance-covariance matrix estimation. The options are:
   - `NULL`: The default variance-covariance matrix estimated by the plm function is used.
   - `"HC"`: A heteroscedasticity-robust (HC) covariance matrix is estimated using the `vcovHC` function from the `plm` package, vcovHC, with type `"HC1"` and method `"white1"` (see White, 1980).
   - `"HAC"`: A heteroscedasticity and autocorrelation robust (HAC) covariance matrix is estimated using the `vcovHC` function from the `plm` package, vcovHC, with type `"HC3"` and method `"arellano"` (see Arellano, 1987).
   - `"CL"`: A cluster-robust covariance matrix is estimated using the `vcovCR` function from the `clubSandwich` package with type `"CR0"` (see Lian & Zegers (1986)). The cluster variable is either `"ID"` or a custom cluster variable provided in the `data` dataframe.

2. A function that takes an plm object as input and returns a variance-covariance matrix. This allows for custom variance-covariance matrix estimation methods. For example, you could use the vcovHC function from the sandwich package with a specific method and type:

   ```
   function(x) {vcovHC(x, method = "white1", type = "HC2")}
   ```

If no vcov parameter is provided, the function defaults to using the variance-covariance matrix estimated by the plm::plm() function.

One should note that rows containing NA values are removed from the panel before the testing procedure is performed.

NOTE: Please be aware that including control variables (X) might lead to higher computation times for type = "Boot" and type = "Wild", due to unconstrained parameters in the optimization problem that estimates the constrained placebo coefficients.

On top of that, please be aware that the bootstrap procedures for the equivalence test based on the maximum absolute placebo coefficient apply a bootstrap procedure (as described by Dette & Schumann (2024)), leading to a stochastic critical value and minimum equivalence threshold. Therefore, the results may vary slightly between different runs of the function. For reproducibility of the bootstrap procedures, it is recommended to set a seed before using the function.

Value

If type = "IU", an object of class maxEquivTestIU with

- placebo_coefficients: A numeric vector of the estimated placebo coefficients,
- abs_placebo_coefficients: a numeric vector with the absolute values of estimated placebo coefficients,
- placebo_coefficients_se: a numeric vector with the standard errors of the placebo coefficients,
- significance_level: the chosen significance level of the test,
- base_period: the base period used in the testing procedure,
- placebo_names: the names corresponding to the placebo coefficients,
- num_individuals: the number of cross-sectional individuals in the panel used for testing,
- num_periods: the number of periods in the panel used for testing (if the panel is unbalanced, num_periods indicates the range of time periods across all individuals),
- num_observations: the total number of observations in the panel used for testing,
- is_panel_balanced: a logical value indicating whether the panel is balanced,
- equiv_threshold_specified: a logical value indicating whether an equivalence threshold was specified.
- if equiv_threshold_specified = TRUE, then additionally
  - IU_critical_values: a numeric vector with the individual critical values for each of the placebo coefficients,
  - reject_null_hypothesis: a logical value indicating whether the null hypothesis of negligible pre-trend differences can be rejected at the specified significance level alpha,
  - equiv_threshold: the equivalence threshold employed.
- if equiv_threshold_specified = FALSE, then additionally

  - minimum_equiv_thresholds: a numeric vector including for each placebo coefficient the minimum equivalence threshold for which the null hypothesis of negligible pre-trend differences can be rejected for the corresponding placebo coefficient individually,
  - minimum_equiv_threshold: a numeric scalar minimum equivalence threshold for which the null hypothesis of negligible pre-trend differences can be rejected for all placebo coefficients individually.

if type = "Boot" or type = "Wild", an object of class "maxEquivTestBoot" with

- placebo_coefficients: a numeric vector of the estimated placebo coefficients,
- abs_placebo_coefficients: a numeric vector with the absolute values of estimated placebo coefficients,
- max_abs_coefficient: the maximum absolute estimated placebo coefficient,
- B: the number of bootstrap samples used to find the critical value,
- significance_level: the chosen significance level of the test alpha,
- base_period: the base period used in the testing procedure,
- placebo_names: the names corresponding to the placebo coefficients,
- equiv_threshold_specified: a logical value indicating whether an equivalence threshold was specified.
- num_individuals: the number of cross-sectional individuals in the panel used for testing,
- num_periods: the number of pre-treatment periods in the panel used for testing (if the panel is unbalanced, num_periods represents the range in the number of time periods covered by different individuals),
- num_observations: the total number of observations in the panel used for testing,
- is_panel_balanced: a logical value indicating whether the panel is balanced.
- if equiv_threshold_specified = TRUE, then additionally
  - bootstrap_critical_value: the by bootstrap found critical value for the equivalence test based on the maximum absolute placebo coefficient,
  - reject_null_hypothesis: a logical value indicating whether the null hypothesis of negligible pre-trend differences can be rejected at the specified significance level alpha,
- if equiv_threshold_specified = FALSE, then additionally
  - minimum_equiv_threshold: a numeric scalar minimum equivalence threshold for which the null hypothesis of negligible pre-trend differences can be rejected for the bootstrap procedure.

Author(s)

Ties Bos

References

Arellano M (1987). "Computing Robust Standard Errors for Within-groups Estimators." *Oxford bulletin of Economics and Statistics*, 49(4), 431–434.

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

Liang, K.-Y., & Zeger, S. L. (1986). "Longitudinal data analysis using generalized linear models." *Biometrika*, 73(1), 13-22. doi:10.1093/biomet/73.1.13

White H (1980). "A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity." *Econometrica*, 48(4), 817–838.

See Also

print.maxEquivTestBoot print.maxEquivTestIU

Examples

```
# Generate a balanced panel dataset with 500 cross-sectional units (individuals),
# 5 time periods (labeled 1-5), a binary variable indicating which individual
# receives treatment and 2 control variables ("X_1" and "X_2") The error-terms are generated without
# heteroscedasticity,  autocorrelation, or any significant clusters.
# Furthermore, there are no fixed effects (lambda and eta are both vectors
# containing only 0) and no pre-trends present in the data (all values in
# beta are 0). See sim_paneldata() for more details.

sim_data <- sim_paneldata(N = 500, tt = 5, p = 2, beta = rep(0, 5),
                          gamma = rep(1, 2), het = 0, phi = 0, sd = 1,
                          burnins = 50)

# -----------------  IU Approach -----------------
# Perform the test with equivalent threshold specified as 1 based on
# pre-treatment periods 1-4 and homoscedastic error-terms:
  # To select variables, one can use the column names / numbers in the panel data
maxEquivTest(Y = "Y", ID = "ID", G = "G", period = 2, X= c(5,6),
              data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
              base_period = 4, type = "IU")
  # Alternatively, one can enter the variables separately:
data_Y <- sim_data$Y
data_ID <- sim_data$ID
data_G <- sim_data$G
data_period <- sim_data$period
data_X <- sim_data[, c(5, 6)]
maxEquivTest(Y = data_Y, ID = data_ID, G = data_G, period = data_period, X = data_X,
             equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4, type = "IU")

# Perform the test without specifying the equivalence threshold with heteroscedastic
# and autocorrelation robust variance-covariance matrix estimator:
maxEquivTest(Y = 3, ID = 1, G = 4, period = 2,
             data = sim_data, equiv_threshold = NULL, pretreatment_period = 1:4,
             base_period = 4, type = "IU", vcov = "HAC")

# Perform the test without specifying the equivalence threshold with a custom
# variance-covariance matrix estimator:
```

```
vcov_func <- function(x) {plm::vcovHC(x, method = "white1", type = "HC2")}
maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
             data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4, type = "IU", vcov = vcov_func)

# Perform the test using clustered standard errors based on a vector indicating
# the cluster. For instance, two clusters with the following rule: all
# individuals with an ID below 250 are in the same cluster.
cluster_ind <- ifelse(sim_data$ID < 250, 1, 2)
maxEquivTest(Y = data_Y, ID = data_ID, G = data_G, period = data_period, X = data_X,
               equiv_threshold = 1, pretreatment_period = 1:4,
               base_period = 4, type = "IU", vcov = "CL", cluster = cluster_ind)

# Note that the testing procedure can also handle unbalanced panels.
# Finally, one should note that the test procedure also works for unbalanced panels.
# To illustrate this, we generate an unbalanced panel dataset by randomly selecting
# 70% of the observations from the balanced panel dataset:
random_indeces <- sample(nrow(sim_data), 0.7*nrow(sim_data))
unbalanced_sim_data <- sim_data[random_indeces, ]
maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period", X = c(5, 6),
             data = unbalanced_sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
              base_period = 4, type = "IU", vcov = "HAC")

#-----------------  Bootstrap Approach -----------------

 # Perform the test with equivalence threshold specified as 1 based on
 # pre-treatment periods 1:4 (with base period 4) with the general bootstrap procedure:
 maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
             data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4, type = "Boot")

 # Perform the test with the equivalence threshold specified as 1 based on
 # pre-treatment periods 1:4 (with base period 4) with the wild bootstrap procedure:
 maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
             data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4, type = "Wild")

 # The bootstrap procedures can handle unbalanced panels:
 maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
             data = unbalanced_sim_data, equiv_threshold = 1,
             pretreatment_period = 1:4,
             base_period = 4, type = "Boot")
 maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
             data = unbalanced_sim_data, equiv_threshold = 1,
             pretreatment_period = 1:4,
             base_period = 4, type = "Wild")

 # Performing the test without specifying the equivalence threshold:
 maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
             data = sim_data, equiv_threshold = NULL, pretreatment_period = 1:4,
             base_period = 4, type = "Boot")

 maxEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
```

```
data = sim_data, equiv_threshold = NULL, pretreatment_period = 1:4,
base_period = 4, type = "Wild")
```

`maxTest_error` *Additional input checks for the maxEquivTest function*

Description

This function checks additonal inputs specific to the maxEquivTest function.

Usage

```
maxTest_error(type, equiv_threshold, vcov, B)
```

Arguments

`type`	the type of test for the maximum absolute placebo coefficient to be conducted; must be one of "IU", "Boot" or "Wild".
`equiv_threshold`	the equivalence threshold for the test. Must be a numeric scalar or NULL.
`vcov`	the variance-covariance matrix estimator. See `maxEquivTest` for more information.
`B`	the number of bootstrap iterations. Must be a numeric integer scalar.

Value

A list with two elements: `error` a logical value indicating whether an error was found, and `message` a character string with the error message. If no error was found, `error` is `FALSE` and `message` is empty.

`maxTestBoot_func` *An internal function of the EquiTrends Maximum Equivalence Testing procedure using the Bootstrap approaches.*

Description

This is a supporting function of the `maxEquivTest` function. It calculates the placebo coefficients and the absolute value of the placebo coefficients. It then calculates the critical value by bootstrap if an equivalence threshold is supplied for the test, according to Dette & Schumann (2024).

Usage

```
maxTestBoot_func(
  data,
  equiv_threshold,
  alpha,
  n,
  B,
  no_periods,
  base_period,
  type,
  original_names,
  is_panel_balanced
)
```

Arguments

`data`	The data.frame object containing the data for the test. Should be of the form what is returned by the EquiTrends_dataconstr function.
`equiv_threshold`	The equivalence threshold for the test.
`alpha`	The significance level for the test.
`n`	The number of cross-sectional individuals in the data.
`B`	The number of bootstrap replications.
`no_periods`	The number of periods in the data.
`base_period`	The base period for the test. Must be one of the unique periods in the data.
`type`	The type of bootstrap to be used. Must be one of "Boot" or "Wild".
`original_names`	The original names of the control variables in the data.
`is_panel_balanced`	A logical value indicating whether the panel data is balanced.

Value

an object of class "maxEquivTestBoot" with

`placebo_coefficients`	A numeric vector of the estimated placebo coefficients,
`abs_placebo_coefficients`	a numeric vector with the absolute values of estimated placebo coefficients,
`max_abs_coefficient`	the maximum absolute estimated placebo coefficient,
`bootstrap_critica_value`	the by bootstrap found critical value for the equivalence test based on the maximum absolute placebo coefficient,
`reject_null_hypothesis`	a logical value indicating whether the null hypothesis of negligible pre-trend differences can be rejected at the specified significance level `alpha`,

`B` the number of bootstrap samples used to find the critical value,

`significance_level` the chosen significance level of the test `alpha`,

`num_individuals` the number of cross-sectional individuals (n),

`num_periods` the number of periods (T),

`num_observations` the total number of observations (N),

`base_period` the base period in the data,

`placebo_names` the names corresponding to the placebo coefficients,

`equiv_threshold_specified` a logical value indicating whether an equivalence threshold was specified.

`is_panel_balanced` a logical value indicating whether the panel data is balanced.

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

`maxTestIU_func` *An internal function of the EquiTrends Maximum Equivalence Testing procedure using the Intersection Union approach.*

Description

This is a supporting function of the `maxEquivTest` function. It calculates the placebo coefficients and the absolute value of the placebo coefficients. It then calculates the critical value and p-values if an equivalence threshold is supplied for the test, according to Dette & Schumann (2024). If no equivalence threshold is supplied, it calculates the minimum equivalence threshold for which the null of non-negligible pre-trend differences can be rejected.

Usage

```
maxTestIU_func(
  data,
  equiv_threshold,
  vcov,
  cluster,
  alpha,
  n,
  no_periods,
  base_period,
  is_panel_balanced
)
```

Arguments

`data`	The data.frame object containing the data for the test. Should be of the form what is returned by the EquiTrends_dataconstr function.
`equiv_threshold`	The equivalence threshold for the test. If NULL, the minimum equivalence threshold for which the null hypothesis of non-negligible can be rejected is calculated.
`vcov`	The variance-covariance matrix estimator. See maxEquivTest for more information.
`cluster`	The cluster variable for the cluster-robust variance-covariance matrix estimator. See maxEquivTest for more information.
`alpha`	The significance level for the test.
`n`	The number of cross-sectional individuals in the data.
`no_periods`	The number of periods in the data.
`base_period`	The base period for the test. Must be one of the unique periods in the data.
`is_panel_balanced`	A logical value indicating whether the panel data is balanced.

Value

An object of class "maxEquivTestIU" containing:

`placebo_coefficients`	A numeric vector of the estimated placebo coefficients,
`abs_placebo_coefficients`	a numeric vector with the absolute values of estimated placebo coefficients,
`placebo_coefficient_se`	a numeric vector with the standard errors of the placebo coefficients,
`significance_level`	the chosen significance level of the test,
`num_individuals`	the number of cross-sectional individuals (n),
`num_periods`	the number of periods (T),
`num_observations`	the total number of observations (N),
`base_period`	the base period in the data,
`placebo_names`	the names corresponding to the placebo coefficients,
`equiv_threshold_specified`	a logical value indicating whether an equivalence threshold was specified.
`is_panel_balanced`	a logical value indicating whether the panel data is balanced.

Additionally, if `!(is.null(equiv_threshold))`

- `IU_critical_values`: a numeric vector with the individual critical values for each of the placebo coefficients,
- `reject_null_hypothesis`: a logical value indicating whether the null hypothesis of negligible pre-trend differences can be rejected at the specified significance level `alpha`,
- `equiv_threshold`: the equivalence threshold employed,

if `is.null(equiv_threshold)`

- `minimum_equiv_thresholds`: a numeric vector including for each placebo coefficient the minimum equivalence threshold for which the null hypothesis of negligible pre-trend differences can be rejected for the corresponding placebo coefficient individually,
- `minimum_equiv_threshold`: a numeric scalar minimum equivalence threshold for which the null hypothesis of negligible pre-trend differences can be rejected for all placebo coefficients individually.

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

`maxTestIU_optim_func` *Finding the minimum equivalence threshold for the equivalence test based on the IU procedure for the maximum placebo coefficient.*

Description

`maxTestIU_optim_func` solves the optimization problem to find the minimum equivalence threshold for which one can reject the null hypothesis of non-negligible pre-trend differences at a given significance level for the equivalence test based on the maximum placebo coefficient, especially for the Intersection Union type.

Usage

```
maxTestIU_optim_func(coef, sd, alpha)
```

Arguments

`coef`	The estimated absolute value of the mean placebo coefficients
`sd`	The estimated standard deviation of the mean of the placebo coefficients
`alpha`	The significance level

Value

The minimum equivalence threshold for which the null hypothesis of non-negligible differences can be rejected for the equivalence test based on the mean placebo coefficient.

`meanEquivTest` *Equivalence Test for Pre-trends based on the Mean Placebo Coefficient*

Description

This function performs an equivalence test for pre-trends based on the mean placebo coefficient from Dette & Schumann (2024).

Usage

```
meanEquivTest(
  Y,
  ID,
  G,
  period,
  X = NULL,
  data = NULL,
  equiv_threshold = NULL,
  pretreatment_period = NULL,
  base_period = NULL,
  vcov = NULL,
  cluster = NULL,
  alpha = 0.05
)
```

Arguments

`Y`	A numeric vector with the variable of interest. If `data` is supplied, `Y` should be a scalar indicating the column number or column-name character string that corresponds to the numeric dependent (outcome) variable in 'data'.
`ID`	A numeric vector identifying the different cross-sectional units in the dataset. If `data` is supplied, `ID` should be a scalar indicating the column number or column-name character string that corresponds to the cross-sectional units identifier in `data`.
`G`	A binary or logic vector (of the same dimension as `Y` and `ID`) indicating if the individual (e.g. as indicated by `ID`) receives treatment (e.g. 1 or TRUE) or not (0 or FALSE). f 'data' is supplied, `G` should be a scalar identifying the column number or column-name character string associated to `G` in `data`.
`period`	A numeric vector (of the same dimension as Y) indicating time. If `data` is supplied, `period` should be a scalar indicating the column number or column-name character string that corresponds to the time identifier in `data`.
`X`	A vector, matrix, or data.frame containing the control variables. If `data` is supplied, `X` must be a vector of column numbers or column-name character strings that identifies the control variables' columns.

`data` An optional `data.frame` object containing the variables in Y, ID, G, T and, if supplied, X and cluster as its columns.

`equiv_threshold`
The scalar equivalence threshold (must be positive). The default is NULL, implying that the function must look for the minimum value for which the null hypothesis of "non-negligible differences" can still be rejected.

`pretreatment_period`
A numeric vector identifying the pre-treatment periods that should be used for testing. `pretreatment_period` must be a subset of the periods included through `period`. The default is to use all periods that are included in `period`.

`base_period` The pre-treatment period to compare the post-treatment observation to. The default is to take the last period of the pre-treatment period.

`vcov` The variance-covariance matrix that needs to be used. See *Details* for more details.

`cluster` If `vcov = "CL"`, a vector indicating which observations belong to the same cluster. `cluster` must be of the same length as the panel. If `data` is supplied, `cluster` must be either the column index or column name of this vector in the data.frame/matrix. The default (`cluster=NULL`) assumes every unit in ID is its own cluster.

`alpha` Significance level of the test. The default is 0.05. Only required if `equiv_threshold` is not specified.

Details

The `vcov` parameter specifies the variance-covariance matrix to be used in the function. This parameter can take two types of inputs:

1. A character string specifying the type of variance-covariance matrix estimation. The options are:
   - `NULL`: The default variance-covariance matrix estimated by the plm function is used.
   - `"HC"`: A heteroscedasticity-robust (HC) covariance matrix is estimated using the `vcovHC` function from the `plm` package, vcovHC, with type `"HC1"` and method `"white1"` (see White, 1980).
   - `"HAC"`: A heteroscedasticity and autocorrelation robust (HAC) covariance matrix is estimated using the `vcovHC` function from the `plm` package, vcovHC, with type `"HC3"` and method `"arellano"` (see Arellano, 1987).
   - `"CL"`: A cluster-robust covariance matrix is estimated using the `vcovCR` function from the `clubSandwich` package with type `"CR0"` (see Lian & Zegers (1986)). The cluster variable is either `"ID"` or a custom cluster variable provided in the `data` dataframe.
2. A function that takes an plm object as input and returns a variance-covariance matrix. This allows for custom variance-covariance matrix estimation methods. For example, you could use the `vcovHC` function from the `sandwich` package with a specific method and type:

   ```
   function(x) {vcovHC(x, method = "white1", type = "HC2")}
   ```

If no `vcov` parameter is provided, the function defaults to using the variance-covariance matrix estimated by the plm::plm() function.

One should note that rows containing `NA` values are removed from the panel before the testing procedure is performed.

Value

An object of class "meanEquivTest" containing:

`placebo_coefficients`
: a numeric vector of the estimated placebo coefficients,

`abs_mean_placebo_coefs`
: the absolute value of the mean of the placebo coefficients,

`var_mean_placebo_coef`
: the estimated variance of the mean placebo coefficient,

`significance_level`
: the significance level of the test,

`base_period`
: the base period used in the testing procedure,

`num_individuals`
: the number of cross-sectional individuals in the panel used for testing,

`num_periods`
: the number of periods in the panel used for testing (if the panel is unbalanced, `num_periods` represents the range in the number of time periods covered by different individuals),

`num_observations`
: the total number of observations in the panel used for testing,

`is_panel_balanced`
: a logical value indicating whether the panel is balanced,

`equiv_threshold_specified`
: a logical value indicating whether an equivalence threshold was specified.

If `equiv_threshold_specified = FALSE`, then additionally `minimum_equiv_threshold`: the minimum equivalence threshold for which the null hypothesis of non-negligible (based on the equivalence threshold) trend-differences can be rejected.

If `equiv_threshold_specified = TRUE`, then additionally

- `mean_critical_value`: the critical value at the alpha level,
- `p_value`: the p-value of the test,
- `reject_null_hypothesis`: A logical value indicating whether to reject the null hypothesis,
- `equiv_threshold`: the equivalence threshold specified.

Author(s)

Ties Bos

References

Arellano M (1987). "Computing Robust Standard Errors for Within-groups Estimators." *Oxford bulletin of Economics and Statistics*, 49(4), 431–434.

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

Liang, K.-Y., & Zeger, S. L. (1986). "Longitudinal data analysis using generalized linear models." *Biometrika*, 73(1), 13-22. DOI: doi:10.1093/biomet/73.1.13

White H (1980). "A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity." *Econometrica*, 48(4), 817–838.

See Also

print.meanEquivTest

Examples

```
# Generate a balanced panel dataset with 500 cross-sectional units (individuals),
# 5 time periods (labeled 1-5), a binary variable indicating which individual
# receives treatment and 2 control variables ("X_1" and "X_2")
# The error-terms are generated without heteroscedasticity, autocorrelation,
# or any significant clusters. Furthermore, there are no fixed effects
# and no pre-trends present in the data (all values in beta are 0).
# See sim_paneldata() for more details.

sim_data <- sim_paneldata(N = 500, tt = 5, p = 2, beta = rep(0, 5),
                          gamma = rep(1, 2), het = 0, phi = 0, sd = 1,
                          burnins = 50)

# Perform the test with equivalent threshold specified as 1 based on
# pre-treatment periods 1-4 and assuming homoscedastic error-terms:
  # To select variables, one can use the column names / column numbers in the panel data:
  meanEquivTest(Y = "Y", ID = "ID", G = "G", period = 2, X = c(5, 6),
                data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
                base_period = 4)
  # Alternatively, one can use separate variables:
  data_Y <- sim_data$Y
  data_ID <- sim_data$ID
  data_G <- sim_data$G
  data_period <- sim_data$period
  data_X <- sim_data[, c(5, 6)]
  meanEquivTest(Y = data_Y, ID = data_ID, G = data_G, period = data_period, X = data_X,
                equiv_threshold = 1, pretreatment_period = 1:4,
                base_period = 4)

# Perform the test with a heteroscedastic and autocorrelation robust
# variance-covariance matrix estimator, and without specifying the equivalence threshold:
meanEquivTest(Y = "Y", ID = "ID", G = "G", period = "period", X = c(5, 6),
              data = sim_data, equiv_threshold = NULL, pretreatment_period = 1:4,
              base_period = 4, vcov = "HAC")

# Perform the test with an equivalence threshold of 1 and a custom
# variance-covariance matrix estimator:
vcov_func <- function(x) {plm::vcovHC(x, method = "white1", type = "HC2")}
meanEquivTest(Y = "Y", ID = "ID", G = "G", period = "period",
              data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
              base_period = 4, vcov = vcov_func)
```

```
# Perform the test using clustered standard errors based on a vector indicating
# the cluster. For instance, two clusters with the following rule: all
# individuals with an ID below 250 are in the same cluster:
cluster_ind <- ifelse(sim_data$ID < 250, 1, 2)
meanEquivTest(Y = data_Y, ID = data_ID, G = data_G, period = data_period, X = data_X,
               equiv_threshold = 1, pretreatment_period = 1:4,
               base_period = 4, vcov = "CL", cluster = cluster_ind)

# Note that the testing procedure can also handle unbalanced panels.
# Finally, one should note that the test procedure also works for unbalanced panels.
# To illustrate this, we generate an unbalanced panel dataset by randomly selecting
# 70% of the observations from the balanced panel dataset:
random_indeces <- sample(nrow(sim_data), 0.7*nrow(sim_data))
unbalanced_sim_data <- sim_data[random_indeces, ]
meanEquivTest(Y = "Y", ID = "ID", G = "G", period = "period", X = c(5, 6),
              data = unbalanced_sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
               base_period = 4, vcov = "HAC")
```

`meanTest_func` *An internal function of the EquiTrends Mean Equivalence Testing procedure*

Description

This is a supporting function of the `meanEquivTest` function. It calculates the placebo coefficients and the absolute value of the mean of the placebo coefficients. It then calculates the critical value and p-values if an equivalence threshold is supplied for the test, according to Dette & Schumann (2024). If equivalence threshold is not supplied, it calculates the minimum equivalence threshold for which the null of non-negligible pre-trend differences can be rejected.

Usage

```
meanTest_func(
  data,
  equiv_threshold,
  vcov,
  cluster,
  alpha,
  n,
  no_periods,
  base_period,
  is_panel_balanced
)
```

Arguments

`data`	The data.frame object containing the data for the test. Should be of the form what is returned by the EquiTrends_dataconstr function.
`equiv_threshold`	The equivalence threshold for the test. If NULL, the minimum equivalence threshold for which the null hypothesis can be rejected is calculated.
`vcov`	The variance-covariance matrix estimator. See meanEquivTest for more information.
`cluster`	The cluster variable for the cluster-robust variance-covariance matrix estimator. See meanEquivTest for more information.
`alpha`	The significance level for the test. Only required if no equivalence threshold is supplied.
`n`	The number of cross-sectional individuals in the data.
`no_periods`	The number of periods in the data.
`base_period`	The base period for the test. Must be one of the unique periods in the data.
`is_panel_balanced`	A logical value indicating whether the panel data is balanced.

Value

#' An object of class "meanEquivTest" containing:

`placebo_coefficients`	A numeric vector of the estimated placebo coefficients,
`abs_mean_placebo_coefs`	the absolute value of the mean of the placebo coefficients,
`var_mean_placebo_coef`	the estimated variance of the mean placebo coefficient,
`significance_level`	the significance level of the test,
`num_individuals`	the number of cross-sectional individuals in the data,
`num_periods`	the number of periods in the data,
`base_period`	the base period in the data,
`num_observations`	the total number of observations in the data,
`equiv_threshold_specified`	a logical value indicating whether an equivalence threshold was specified.
`is_panel_balanced`	a logical value indicating whether the panel data is balanced.

If `is.null(equiv_threshold)`, then additionally `minimum_equiv_threshold`: the minimum equivalence threshold for which the null hypothesis of non-negligible (based on the equivalence threshold) trend-differnces can be rejected.

if `!(is.null(equiv_threshold))`, then additionally

- `mean_critical_value`: the critical value at the alpha level,
- `p_value`: the p-value of the test,
- `reject_null_hypothesis`: A logical value indicating whether to reject the null hypothesis,
- `equiv_threshold`: the equivalence threshold specified.

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

`meanTest_optim_func` *Finding the minimum equivalence threshold for the mean equivalence test*

Description

`meanTest_optim_func` solves the optimization problem to find the minimum equivalence threshold for which one can reject the null hypothesis of non-negligible pre-trend differences at a given significance level for the equivalence test based on the mean placebo coefficient.

Usage

```
meanTest_optim_func(coef, sd, alpha)
```

Arguments

`coef`	The estimated absolute value of the mean placebo coefficients
`sd`	The estimated standard deviation of the mean of the placebo coefficients
`alpha`	The significance level

Value

The minimum equivalence threshold for which the null hypothesis of non-negligible differences can be rejected for the equivalence test based on the mean placebo coefficient.

print.maxEquivTestBoot *Print maxEquivTestBoot objects*

Description

Print maxEquivTestBoot objects

Usage

```
## S3 method for class 'maxEquivTestBoot'
print(x, ...)
```

Arguments

x	An object of class 'maxEquivTestBoot' containing the results of the maximum test based on the bootstrap procedure.
...	Further arguments passed to or from other methods.

Value

The function prints a summary of the results of the maximum test based on the bootstrap procedures.

print.maxEquivTestIU *Print maxEquivTestIU objects*

Description

Print method for objects of class 'maxEquivTestIU'.

Usage

```
## S3 method for class 'maxEquivTestIU'
print(x, ...)
```

Arguments

x	An object of class 'maxEquivTestIU' containing the results of the maximum test based on the intersection-union approach.
...	Further arguments passed to or from other methods.

Value

The function prints a summary of the results of the maximum test based on the intersection-union approach.

print.meanEquivTest *Print meanEquivTest objects*

Description

Print meanEquivTest objects

Usage

```
## S3 method for class 'meanEquivTest'
print(x, ...)
```

Arguments

x	An object of class 'meanEquivTest' containing the results of the maximum test based on the bootstrap procedure.
...	Further arguments passed to or from other methods.

Value

The function prints a summary of the results of the maximum test based on the bootstrap procedures.

print.rmsEquivTest *Print rmsEquivTest objects*

Description

Print rmsEquivTest objects

Usage

```
## S3 method for class 'rmsEquivTest'
print(x, ...)
```

Arguments

x	An object of class 'rmsEquivTest' containing the results of the maximum test based on the bootstrap procedure.
...	Further arguments passed to or from other methods.

Value

The function prints a summary of the results of the maximum test based on the bootstrap procedures.

`rmsEquivTest` *Equivalence Test for Pre-trends based on the RMS Placebo Coefficient*

Description

This function performs an equivalence test for pre-trends based on the root mean squared placebo coefficient from Dette & Schumann (2024).

Usage

```
rmsEquivTest(
  Y,
  ID,
  G,
  period,
  X = NULL,
  data = NULL,
  equiv_threshold = NULL,
  pretreatment_period = NULL,
  base_period = NULL,
  alpha = 0.05,
  no_lambda = 5
)
```

Arguments

`Y`	A numeric vector with the variable of interest. If `data` is supplied, `Y` should be a scalar indicating the column number or column-name character string that corresponds to the numeric dependent (outcome) variable in 'data'.
`ID`	A numeric vector identifying the different cross-sectional units in the dataset. If `data` is supplied, `ID` should be a scalar indicating the column number or column-name character string that corresponds to the cross-sectional units identifier in `data`.
`G`	A binary or logic vector (of the same dimension as `Y` and `ID`) indicating if the individual (e.g. as indicated by `ID`) receives treatment (e.g. 1 or TRUE) or not (0 or FALSE). f 'data' is supplied, `G` should be a scalar identifying the column number or column-name character string associated to `G` in `data`.
`period`	A numeric vector (of the same dimension as Y) indicating time. If `data` is supplied, `period` should be a scalar indicating the column number or column-name character string that corresponds to the time identifier in `data`.
`X`	A vector, matrix, or data.frame containing the control variables. If `data` is supplied, `X` must be a vector of column numbers or column-name character strings that identifies the control variables' columns.
`data`	An optional `data.frame` object containing the variables in Y, ID, G, T and, if supplied, X and cluster as its columns.

`equiv_threshold`
: The scalar equivalence threshold (must be positive). The default is NULL, implying that the function must look for the minimum value for which the null hypothesis of "non-negligible differences" can still be rejected.

`pretreatment_period`
: A numeric vector identifying the pre-treatment periods that should be used for testing. `pretreatment_period` must be a subset of the periods included through `period`. The default is to use all periods that are included in `period`.

`base_period`
: The pre-treatment period to compare the post-treatment observation to. The default is to take the last period of the pre-treatment period.

`alpha`
: Significance level of the test. The default is 0.05.

`no_lambda`
: Parameter specifying the number of incremental segments of the dataset over which a statistic is calculated. See *Details*. The default is 5.

Details

`no_lambda` determines the proportions lambda/`no.lambda` for lambda = 1,...,`no_lambda` of the cross-sectional units at which the placebo coefficients are estimated. The placebo coefficients are estimated for each of these proportions and the root mean squared (RMS) of the placebo coefficients is calculated, which are then used to construct the critical value at a significance level of `alpha`. See Dette & Schumann (2024, s. 4.2.3.) for more details.

One should note that rows containing `NA` values are removed from the panel before the testing procedure is performed.

Please be aware that the equivalence test based on the root mean squared placebo coefficient uses a randomization technique (as described by Dette & Schumann (2024)), leading to a stochastic critical value and minimum equivalence threshold. Therefore, the results may vary slightly between different runs of the function. For reproducibility, it is recommended to set a seed before using the function.

Value

An object of class "rmsEquivTest" containing:

`placebo_coefficients`
: A numeric vector of the estimated placebo coefficients,

`rms_placebo_coefs`
: the root mean squared value of the placebo coefficients,

`significance_level`
: the significance level of the test,

`base_period`
: the base period used in the testing procedure,

`num_individuals`
: the number of cross-sectional individuals in the panel used for testing,

`num_periods`
: the number of pre-treatment periods in the panel used for testing (if the panel is unbalanced, `num_periods` represents the range in the number of time periods covered by different individuals),

`num_observations`
: the total number of observations in the panel used for testing,

`is_panel_balanced`
: a logical value indicating whether the panel is balanced,

`equiv_threshold_specified`
: a logical value indicating whether an equivalence threshold was specified.

If `equiv_threshold_specified = FALSE`, then additionally `minimum_equiv_threshold`: the minimum equivalence threshold for which the null hypothesis of non-negligible (based on the equivalence threshold) trend-differences can be rejected.

If `equiv_threshold_specified = TRUE`, then additionally

- `rms_critical_value`: the critical value at the alpha level,
- `reject_null_hypothesis`: A logical value indicating whether to reject the null hypothesis,
- `equiv_threshold`: the equivalence threshold specified.

Author(s)

Ties Bos

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

See Also

`print.rmsEquivTest`

Examples

```
# Generate a balanced panel dataset with 500 cross-sectional units (individuals),
# 5 time periods (labeled 1-5), a binary variable indicating which individual
# receives treatment and 2 control variables ("X_1" and "X_2").
# The error-terms are generated without  heteroscedasticity,  autocorrelation,
# or any significant clusters. Furthermore, there are no fixed effects and
# no pre-trends present in the data (all values in  beta are 0).
# See sim_paneldata() for more details.

sim_data <- sim_paneldata(N = 500, tt = 5, p = 2, beta = rep(0, 5),
                          gamma = rep(1, 2), het = 0, phi = 0, sd = 1,
                          burnins = 50)

# Perform the equivalence test using an equivalence threshold of 1 with periods
# 1-4 as pre-treatment periods based on the RMS testing procedure:
#  - option 1: using column names in the panel
# One can use the names of the columns in the panel to specify the variables:
rmsEquivTest(Y = "Y", ID = "ID", G = "G", period = "period", X = c("X_1", "X_2"),
             data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4)

#  - option 2: using column numbers in the panel
```

```
# Alternatively, one can use the column numbers in the panel to specify the variables:
rmsEquivTest(Y = 3, ID = 1, G = 4, period = 2, X = c(5, 6),
             data = sim_data, equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4)

#  - option 3: using separate variables
# One can also use the variables directly without specifying the data variable:
data_Y <- sim_data$Y
data_ID <- sim_data$ID
data_G <- sim_data$G
data_period <- sim_data$period
data_X <- cbind(sim_data$X_1, sim_data$X_2)

rmsEquivTest(Y = data_Y, ID = data_ID, G = data_G, period = data_period, X = data_X,
             equiv_threshold = 1, pretreatment_period = 1:4,
             base_period = 4)

# The testing procedures can also be performed without specifying the
# equivalence threshold specified. Then, the minimum equivalence threshold is returned
# for which the null hypothesis of non-negligible trend-differences can be rejected.
# Again, the three possible ways of entering the data as above can be used:
rmsEquivTest(Y = "Y", ID = "ID", G = "G", period = "period", X = c("X_1", "X_2"),
             data = sim_data, equiv_threshold = NULL, pretreatment_period = 1:4,
             base_period = 4)

rmsEquivTest(Y = 3, ID = 1, G = 4, period = 2, X = c(5, 6),
             data = sim_data, equiv_threshold = NULL, pretreatment_period = 1:4,
             base_period = 4)

rmsEquivTest(Y = data_Y, ID = data_ID, G = data_G, period = data_period, X= data_X,
             equiv_threshold = NULL, pretreatment_period = 1:4,
             base_period = 4)

# Finally, one should note that the test procedure also works for unbalanced panels.
# To illustrate this, we generate an unbalanced panel dataset by randomly selecting
# 70% of the observations from the balanced panel dataset:

random_indeces <- sample(nrow(sim_data), 0.7*nrow(sim_data))
unbalanced_sim_data <- sim_data[random_indeces, ]
#  With Equivalence Threshold:
rmsEquivTest(Y = 3, ID = 1, G = 4, period = 2, X = c(5, 6),
             data = unbalanced_sim_data, equiv_threshold = 1,
             pretreatment_period = 1:4, base_period = 4)

#  Without Equivalence Threshold:
rmsEquivTest(Y = 3, ID = 1, G = 4, period = 2, X = c(5, 6),
             data = unbalanced_sim_data, equiv_threshold = NULL,
             pretreatment_period = 1:4, base_period = 4)
```

rmsTest_error *Additional input checks for the rmsEquivTest function*

Description

Additional input checks for the rmsEquivTest function

Usage

```
rmsTest_error(alpha, no_lambda)
```

Arguments

alpha	The significance level for the test. Must be one of 0.01, 0.025, 0.05, 0.1 or 0.2.
no_lambda	see rmsEquivTest

Value

A list with two elements: a logical object error indicating if an error is encountered and a message (a character string) corresponding to the error. If error is TRUE, message contains an error message. If error is FALSE, message is an empty string.

rmsTest_func *An internal function of the RMS Equivalence Testing procedure*

Description

This is a supporting function of the rmsEquivTest function. It calculates the placebo coefficients and the RMS of the placebo coefficients. It then calculates the critical value for the test and checks whether the null hypothesis can be rejected, according to Dette & Schumann (2024).

Usage

```
rmsTest_func(
  data,
  equiv_threshold,
  alpha,
  no_lambda,
  base_period,
  no_periods,
  is_panel_balanced
)
```

Arguments

`data`	The data.frame object containing the data for the test. Should be of the form what is returned by the EquiTrends_dataconstr function.
`equiv_threshold`	The equivalence threshold for the test. If NULL, the minimum equivalence threshold for which the null hypothesis can be rejected is calculated.
`alpha`	The significance level for the test. Must be one of 0.01, 0.025, 0.05, 0.1 or 0.2.
`no_lambda`	See rmsEquivTest.
`base_period`	The base period for the test. Must be one of the unique periods in the data.
`no_periods`	The number of periods in the data.
`is_panel_balanced`	A logical value indicating whether the panel data is balanced.

Value

An object of class "rmsEquivTest" containing:

`placebo_coefficients`	A numeric vector of the estimated placebo coefficients,
`rms_placebo_coefs`	the root mean squared value of the placebo coefficients,
`significance_level`	the significance level of the test,
`num_individuals`	the number of cross-sectional individuals in the data (n),
`num_periods`	the number of pre-treatment periods in the data (T),
`num_observations`	the number of observations in the data (N),
`base_period`	the base period in the data,
`equiv_threshold_specified`	a logical value indicating whether an equivalence threshold was specified.
`is_panel_balanced`	a logical value indicating whether the panel data is balanced.

If `is.null(equiv_threshold)`, then additionally `minimum_equiv_threshold`: the minimum equivalence threshold for which the null hypothesis of non-negligible (based on the equivalence threshold) trend-differnces can be rejected.

if `!(is.null(equiv_threshold))`, then additionally

- `rms_critical_value`: the critical value at the alpha level,
- `reject_null_hypothesis`: A logical value indicating whether to reject the null hypothesis,
- `equiv_threshold`: the equivalence threshold specified.

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

`sigma_hathat_c` *Calculating the constrained variance of the residuals for the Boostrap approaches in the EquiTrends Maximum Equivalence Testing procedure, according to Dette & Schumann (2024).*

Description

Calculating the constrained variance of the residuals for the Boostrap approaches in the EquiTrends Maximum Equivalence Testing procedure, according to Dette & Schumann (2024).

Usage

```
sigma_hathat_c(parameter, x, y, ID, time)
```

Arguments

`parameter`	The constrained coefficients.
`x`	The double demeaned independent variables.
`y`	The double demeaned dependent variable.
`ID`	The ID variable.
`time`	The time variable.

Value

The estimated constrained variance of the residuals.

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

sim_check *Checking input for the sim_paneldata function*

Description

Checking input for the sim_paneldata function

Usage

```
sim_check(N, tt, beta, p, gamma, eta, lambda, het, phi, sd, burnins)
```

Arguments

N	The number of cross-sectional units in the panel-data
tt	The number of time periods in the panel-data
beta	The vector of coefficients for the placebo variables. Must be of size tt.
p	The number of additional regressors
gamma	The vector of coefficients for the additional regressors
eta	The vector of fixed effects. Must be of size N.
lambda	The vector of time effects. Must be of size tt.
het	The heteroskedasticity parameter. Must be 0 or 1: `het = 1` indicates that the error terms are generated under heteroskedasticity, `het = 0` indicates the error terms are generated under homoscedasticity.
phi	The AR(1) parameter for the error terms. Must be in the interval [0,1).
sd	The standard deviation of the error terms. Must be a positive number.
burnins	The number of burn-ins for the AR(1) process. Must be a positive integer.

Value

A list with two elements: a logical object error indicating if an error is encountered and a message (a character string) corresponding to the error. If error is TRUE, message contains an error message. If error is FALSE, message is an empty string.

`sim_paneldata` *Simulating a panel data for a binary treatment*

Description

sim.paneldata generates a panel data set with N cross-sectional units and tt time periods. The data set includes a binary treatment variable, a set of placebo variables, and a set of additional regressors. The data set can be generated under homoskedasticity or heteroskedasticity, and/or AR(1) errors.

Usage

```
sim_paneldata(
  N = 500,
  tt = 5,
  beta = rep(0, tt),
  p = 1,
  gamma = rep(1, p),
  eta = rep(0, N),
  lambda = rep(0, tt),
  het = 0,
  phi = c(0),
  sd = 1,
  burnins = 100
)
```

Arguments

`N`	The number of cross-sectional units in the panel-data
`tt`	The number of time periods in the panel-data
`beta`	The vector of coefficients for the placebo variables. Must be of size tt.
`p`	The number of additional regressors
`gamma`	The vector of coefficients for the additional regressors
`eta`	The vector of fixed effects. Must be of size N.
`lambda`	The vector of time effects. Must be of size tt.
`het`	The heteroskedasticity parameter. Must be 0 or 1: `het = 1` indicates that the error terms are generated under heteroskedasticity, `het = 0` indicates the error terms are generated under homoscedasticity.
`phi`	The AR(1) parameter for the error terms. Must be in the interval [0,1).
`sd`	The standard deviation of the error terms. Must be a positive number.
`burnins`	The number of burn-ins for the AR(1) process. Must be a positive integer.

Value

A `data.frame` with the following columns:

`ID`	The cross-sectional unit identifier
`period`	The time period identifier
`Y`	The dependent variable
`G`	The binary treatment variable
`X_1, ..., X_p`	The additional regressors

Examples

```
sim_data <- sim_paneldata(N = 500, tt = 5, beta = rep(0, 5), p=1,
                          gamma = rep(0,1), het = 1, phi = 0.5, sd = 1,
                          burnins = 100)
```

W_critical_value *Calculating the critical value for the W distribution as construced in Dette & Schumann (2024).*

Description

Calculating the critical value for the W distribution as construced in Dette & Schumann (2024).

Usage

```
W_critical_value(significance_level)
```

Arguments

`significance_level`
: The significance level for the test. Must be one of 0.01, 0.025, 0.05, 0.1, 0.2, 0.8, 0.9, 0.95, 0.975, 0.99.

Value

A numeric scalar with the critical value for the W distribution at the given significance level.

References

Dette, H., & Schumann, M. (2024). "Testing for Equivalence of Pre-Trends in Difference-in-Differences Estimation." *Journal of Business & Economic Statistics*, 1–13. DOI: doi:10.1080/07350015.2024.2308121

Index

∗ **package**
EquiTrends-package, 2

boot_optimization_function, 3

EquiTrends *(*EquiTrends-package*)*, 2
EquiTrends-package, 2
EquiTrends_dataconstr, 4, *13*, *15*, *22*, *31*
EquiTrends_inputcheck, 5

maxEquivTest, *4, 5*, 6, *12*, *15*
maxTest_error, 12
maxTestBoot_func, 12
maxTestIU_func, 14
maxTestIU_optim_func, 16
meanEquivTest, *4, 5*, 17, *22*
meanTest_func, 21
meanTest_optim_func, 23

plm, *7, 8*, *18*
plm::plm(), *8*, *18*
print.maxEquivTestBoot, *10*, 24
print.maxEquivTestIU, *10*, 24
print.meanEquivTest, *20*, 25
print.rmsEquivTest, 25, *28*

rmsEquivTest, *4, 5*, 26, *30, 31*
rmsTest_error, 30
rmsTest_func, 30

sigma_hathat_c, 32
sim_check, 33
sim_paneldata, 34

vcovHC, *7*, *18*

W_critical_value, 35