Package 'TMTI'

May 7, 2026

Type Package

Title Too Many, Too Improbable (TMTI) Test Procedures

Version 1.0.3

Author Phillip B. Mogensen [aut, cre]

Maintainer Phillip B. Mogensen <pbm@math.ku.dk>

Description Methods for computing joint tests, controlling the Familywise Error Rate (FWER) and getting lower bounds on the number of false hypotheses in a set. The methods implemented here are described in Mogensen and Markussen (2021) <doi:10.48550/arXiv.2108.04731>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.0

Imports stats, Rcpp

Suggests parallel

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-10-29 09:30:02 UTC

Contents

`adjust_LocalTest` *Adjust all p-values using a Closed Testing Procedure and a user-defined local test which satisfies the quadratic shortcut given in Mogensen and Markussen (2021)*

Description

Adjust all p-values using a Closed Testing Procedure and a user-defined local test which satisfies the quadratic shortcut given in Mogensen and Markussen (2021)

Usage

```
adjust_LocalTest(
  LocalTest,
  pvals,
  alpha = 0.05,
  is.sorted = FALSE,
  EarlyStop = FALSE,
  verbose = FALSE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  direction = "increasing",
  parallel.direction = "breadth",
  AdjustAll = FALSE,
  ...
)
```

Arguments

LocalTest	A function specifying a local test.
pvals	vector of p-values.
alpha	significance level. Defaults to 0.05.
is.sorted	Logical, indicating whether the supplied p-values are already sorted. Defaults to FALSE.
EarlyStop	Logical; set to TRUE to stop as soon as a hypothesis can be accepted at level alpha. This speeds up the procedure, but now only provides upper bounds on the adjusted p-values that are below alpha.
verbose	Logical; set to TRUE to print progress. Defaults to FALSE.
mc.cores	Number of cores to parallelize onto.
chunksize	Integer indicating the size of chunks to parallelize. E.g., if setting chunksize = mc.cores, each time a parallel computation is set up, each worker will perform only a single task. If mc.cores > chunksize, some threads will be inactive.
direction	String that is equal to either "increasing"/"i", "decreasing"/"d" or "binary"/"b". Determines the search direction. When set to"increasing", the function computes the exact adjusted p-value for all those hypotheses that can be rejected (while controlling the FWER), but is potentially slower than "decreasing". "decreasing"identifies all hypotheses that can be rejected with FWER control, but does not compute the actual adjusted p-values. "binary" performs a binary search for the number of hypotheses that can be rejected with FWER control. Defaults to "increasing". Note that 'binary' does not work with parallel.direction == 'breadth'.
parallel.direction	A string that is either "breadth" or "depth" (or abbreviated to "b" or "d), indicating in which direction to parallelize. Breadth-first parallelization uses a more efficient C++ implementation to adjust each p-value, but depth-first parallelization potentially finishes faster if using early stopping (EarlyStop = TRUE) and very few hypotheses can be rejected.
AdjustAll	Logical, indicating whether to adjust all p-values (TRUE) or only those that are marginally significant (FALSE). Defaults to FALSE.
...	Additional arguments.

Value

a data.frame containing adjusted p-values and their respective indices. If direction == 'decreasing' or 'binary', an integer describing the number of hypotheses that can be rejected with FWER control is returned.

Examples

```
p = sort(runif(100)) # Simulate and sort p-values
p[1:10] = p[1:10]**3 # Make the bottom 10 smaller, such that they correspond to false hypotheses
adjust_LocalTest(
  LocalTest = function(x) {
    min(c(1, length(x) * min(x)))
```

```
  },
  p, alpha = 0.05, is.sorted = TRUE
)
```

`adjust_TMTI` *Adjust all p-values using a Closed Testing Procedeure and the TMTI family of tests.*

Description

Adjust all p-values using a Closed Testing Procedeure and the TMTI family of tests.

Usage

```
adjust_TMTI(
  pvals,
  alpha = 0.05,
  B = 1000,
  gammaList = NULL,
  tau = NULL,
  K = NULL,
  is.sorted = FALSE,
  EarlyStop = FALSE,
  verbose = FALSE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  direction = "increasing",
  parallel.direction = "breadth",
  AdjustAll = FALSE,
  ...
)
```

Arguments

`pvals`	vector of p-values.
`alpha`	significance level. Defaults to 0.05.
`B`	Number of bootstrap replications. Only relevant if length(pvals) > 100 and no gammaList is supplied.
`gammaList`	A list of functions. These functions should be the CDFs of the chosen TMTI test for different m.
`tau`	Number between 0 and 1 or NULL, describing the truncation level.
`K`	Integer between >1 and m describing the truncation index.
`is.sorted`	Logical, indicating whether the supplied p-values are already sorted. Defaults to FALSE.

EarlyStop	Logical; set to TRUE to stop as soon as a hypothesis can be accepted at level alpha. This speeds up the procedure, but now only provides upper bounds on the adjusted p-values that are below alpha.
verbose	Logical; set to TRUE to print progress. Defaults to FALSE.
mc.cores	Number of cores to parallelize onto.
chunksize	Integer indicating the size of chunks to parallelize. E.g., if setting chunksize = mc.cores, each time a parallel computation is set up, each worker will perform only a single task. If mc.cores > chunksize, some threads will be inactive.
direction	String that is equal to either "increasing"/"i", "decreasing"/"d" or "binary"/"b". Determines the search direction. When set to"increasing", the function computes the exact adjusted p-value for all those hypotheses that can be rejected (while controlling the FWER), but is potentially slower than "decreasing". "decreasing"identifies all hypotheses that can be rejected with FWER control, but does not compute the actual adjusted p-values. "binary" performs a binary search for the number of hypotheses that can be rejected with FWER control. Defaults to "increasing". Note that 'binary' does not work with parallel.direction == 'breadth'.
parallel.direction	A string that is either "breadth" or "depth" (or abbreviated to "b" or "d), indicating in which direction to parallelize. Breadth-first parallelization uses a more efficient C++ implementation to adjust each p-value, but depth-first parallelization potentially finishes faster if using early stopping (EarlyStop = TRUE) and very few hypotheses can be rejected.
AdjustAll	Logical, indicating whether to adjust all p-values (TRUE) or only those that are marginally significant (FALSE). Defaults to FALSE.
...	Additional arguments.

Value

a data.frame containing adjusted p-values and their respective indices. If direction == 'decreasing' or 'binary', an integer describing the number of hypotheses that can be rejected with FWER control is returned.

Examples

```
p = sort(runif(100)) # Simulate and sort p-values
p[1:10] = p[1:10]**3 # Make the bottom 10 smaller, such that they correspond to false hypotheses
adjust_TMTI(p, alpha = 0.05, is.sorted = TRUE)
```

CTP_LocalTest *A Closed Testing Procedure for any local test satisfying the conditions of Mogensen and Markussen (2021) using an O(n^2) shortcut.*

Description

A Closed Testing Procedure for any local test satisfying the conditions of Mogensen and Markussen (2021) using an O(n^2) shortcut.

Usage

```
CTP_LocalTest(
  LocalTest,
  pvals,
  alpha = 0.05,
  is.sorted = FALSE,
  EarlyStop = FALSE,
  ...
)

localTest_CTP(localTest, pvals, alpha = 0.05, is.sorted = FALSE, ...)
```

Arguments

LocalTest	A function which defines the choice of local test to use.
pvals	A vector of p-values.
alpha	Level to perform each intersection test at. Defaults to 0.05.
is.sorted	Logical, indicating whether the supplied p-values are already is.sorted. Defaults to FALSE.
EarlyStop	Logical indicating whether to exit as soon as a non-significant p-value is found. Defaults to FALSE.
...	Additional arguments.
localTest	A function specifying a local test (deprecated).

Value

A data.frame containing adjusted p-values and the original index of the p-values.

Examples

```
## Simulate some p-values
## The first 10 are from false hypotheses, the next 10 are from true
pvals = c(
  rbeta(10, 1, 20), ## Mean value of .05
  runif(10)
)
## Perform the CTP using a local Bonferroni test
CTP_LocalTest(function(x) {
  min(c(length(x) * min(x), 1))
}, pvals)
```

CTP_TMTI *A Closed Testing Procedure for the TMTI using an O(n^2) shortcut*

Description

A Closed Testing Procedure for the TMTI using an O(n^2) shortcut

Usage

```
CTP_TMTI(
  pvals,
  alpha = 0.05,
  B = 1000,
  gammaList = NULL,
  tau = NULL,
  K = NULL,
  is.sorted = FALSE,
  EarlyStop = FALSE,
  ...
)

TMTI_CTP(
  pvals,
  alpha = 0.05,
  B = 1000,
  gammaList = NULL,
  tau = NULL,
  K = NULL,
  is.sorted = FALSE,
  ...
)
```

Arguments

pvals	A vector of p-values.
alpha	Level to perform each intersection test at. Defaults to 0.05.
B	Number of bootstrap replications if gamma needs to be approximated. Not used if specifying a list of functions using the gammaList argument or if length(pvals) <= 100. Defaults to 1000.
gammaList	A list of pre-specified gamma functions. If NULL, gamma functions will be approximated via bootstrap, assuming independence. Defaults to NULL.
tau	Numerical (in (0,1)); threshold to use in tTMTI. If set to NULL, then either TMTI (default) or rtTMTI is used.
K	Integer; Number of smallest p-values to use in rtTMTI. If se to NULL, then either TMTI (default) or tTMTI is used.

is.sorted	Logical, indicating the p-values are pre-sorted. Defaults to FALSE.
EarlyStop	Logical indicating whether to exit as soon as a non-significant p-value is found. Defaults to FALSE.
...	Additional arguments.

Value

A data.frame containing adjusted p-values and the original index of the p-values.

Examples

```
## Simulate some p-values
## The first 10 are from false hypotheses, the next 10 are from true
pvals = c(
  rbeta(10, 1, 20), ## Mean value of .05
  runif(10)
)
CTP_TMTI(pvals)
```

FullCTP_C *Leading NA*

Description

Tests a user-specified subset in a CTP, using a user-supplied local test

Usage

```
FullCTP_C(LocalTest, f, pvals, EarlyStop, alpha)
```

Arguments

LocalTest	A function that returns a double in (0, 1).
f	A function that iterates LocalTest over the relevant test tree. In practice, this is called as TestSet_C.
pvals	A vector of p-values.
EarlyStop	Logical indicating whether to exit as soon as a non-significant p-value is found.
alpha	Significance level. This is only used if EarlyStop = TRUE

FWER_set_C *Leading NA*

Description

Computes a the number of hypotheses that can be rejected with FWER control by using a binary search

Usage

```
FWER_set_C(LocalTest, pvals, alpha, low, high, verbose)
```

Arguments

LocalTest	A function that returns a double in (0, 1).
pvals	A vector of p-values.
alpha	A double indicating the significance level
low	integer denoting the starting point for the search. Should start at zero.
high	integer denoting the end point of the search. Should end at pvals.size() - 1.
verbose	boolean, indicating whether to print progress.

Value

The number of hypotheses that can be rejected with kFWER control at a user specific k.

---

gamma_bootstrapper *Function to bootstrap the Cumulative Distribution Functions (CDFs) of the TMTI statistics.*

Description

Function to bootstrap the Cumulative Distribution Functions (CDFs) of the TMTI statistics.

Usage

```
gamma_bootstrapper(m, n = Inf, B = 1000, mc.cores = 1L, tau = NULL, K = NULL)
```

Arguments

m	Number of tests.
n	Number (or Inf) indicating what kind of minimum to consider. Defaults to Inf, corresponding to the global minimum.
B	Number of bootstrap replicates. Rule of thumb is to use at least 10 * m.
mc.cores	Integer denoting the number of cores to use when using parallelization, Defaults to 1, corresponding to single-threaded computations.
tau	Numerical (in (0,1)); threshold to use in tTMTI. If set to NULL, then either TMTI (default) or rtTMTI is used.
K	Integer; Number of smallest p-values to use in rtTMTI. If se to NULL, then either TMTI (default) or tTMTI is used.

Value

An approximation of the function $\gamma^m(x)$ under the assumption that all p-values are independent and exactly uniform.

Examples

```
## Get an approximation of gamma
gamma_function = gamma_bootstrapper(10)
## Evaluate it in a number, say .2
gamma_function(.2)
```

---

gamma_bootstrapper_Ttest

Compute a list of TMTI CDFs for one- and two-sample test scenarios

---

Description

Compute a list of TMTI CDFs for one- and two-sample test scenarios

Usage

```
gamma_bootstrapper_Ttest(
  Y,
  X = NULL,
  n = Inf,
  B = 1000,
  mc.cores = 1L,
  tau = NULL,
  K = NULL,
  verbose = FALSE
)
```

Arguments

Y	A d*m matrix of m response variables with d observations. Can contain missing values in places.
X	Null if one-sample, a vector with only two unique values if two-sample.
n	Number (or Inf) indicating what kind of minimum to consider. Defaults to Inf, corresponding to the global minimum.
B	Number of bootstrap replicates. Rule of thumb is to use at least 10 * m.
mc.cores	Integer denoting the number of cores to use when using parallelization, Defaults to 1, corresponding to single-threaded computations.
tau	Numerical (in (0,1)); threshold to use in tTMTI. If set to NULL, then either TMTI (default) or rtTMTI is used.
K	Integer; Number of smallest p-values to use in rtTMTI. If se to NULL, then either TMTI (default) or tTMTI is used.
verbose	Logical, indicating whether or not to print progress.

Value

A list of bootstrapped TMTI CDFs that can be used directly in the CTP_TMTI function.

Examples

```
d = 100
m = 3

X = sample(LETTERS[1:2], d, replace = TRUE)
Y = matrix(rnorm(d * m), nrow = d, ncol = m)
pvalues = apply(Y, 2, function(y) t.test(y ~ X)$p.value)

gammaFunctions = gamma_bootstrapper_Ttest(Y, X) # Produces a list of CDFs
CTP_TMTI(pvalues, gammaList = gammaFunctions) # Adjusted p-values using the bootstrapped CDFs
```

---

kFWER_LocalTest *kFWER_LocalTest. Computes the largest rejection set possible with kFWER control.*

Description

kFWER_LocalTest. Computes the largest rejection set possible with kFWER control.

Usage

```
kFWER_LocalTest(LocalTest, pvals, k, alpha = 0.05, verbose = FALSE)
```

Arguments

LocalTest	A function that returns a p-value for a joint hypothesis test.
pvals	A vector p-values.
k	An integer denoting the desired k at which to control the kFWER.
alpha	Significance level.
verbose	Logical, indicating whether or not to print progress.

Value

The number of marginal hypotheses that can be rejected with kFWER control.

Examples

```
nfalse = 50
m = 100
pvals = c (
  sort(runif(nfalse, 0, 0.05 / m)),
  sort(runif(m - nfalse, 0.1, 1))
)
kFWER_LocalTest (
  LocalTest = function (x) min(x) * length(x),
  pvals = pvals,
  k = 5,
  alpha = 0.05,
  verbose = FALSE
)
```

---

kFWER_set_C *Leading NA*

Description

Computes a confidence set for the number of false hypotheses among a subset of using a binary search

Usage

```
kFWER_set_C(LocalTest, pvals, k, alpha, low, high, verbose)
```

Arguments

LocalTest	A function that returns a double in (0, 1).
pvals	A vector of p-values.
k	integer denoting the k to control the kFWER at.
alpha	A double indicating the significance level
low	integer denoting the starting point for the search. Should start at zero.
high	integer denoting the end point of the search. Should end at pvals.size() - 1.
verbose	boolean, indicating whether to print progress.

Value

The number of hypotheses that can be rejected with kFWER control at a user specific k.

`kFWER_TMTI` *kFWER_TMTI. Computes the largest rejection set possible with kFWER control.*

Description

kFWER_TMTI. Computes the largest rejection set possible with kFWER control.

Usage

```
kFWER_TMTI(
  pvals,
  k,
  alpha = 0.05,
  B = 1000,
  gammaList = NULL,
  tau = NULL,
  K = NULL,
  verbose = FALSE
)
```

Arguments

`pvals`	A vector p-values.
`k`	An integer denoting the desired k at which to control the kFWER.
`alpha`	Significance level.
`B`	Number of bootstrap replications if gamma needs to be approximated. Not used if specifying a list of functions using the gammaList argument or if length(pvals) <= 100. Defaults to 1000.
`gammaList`	A list of pre-specified gamma functions. If NULL, gamma functions will be approximated via bootstrap, assuming independence. Defaults to NULL.
`tau`	Numerical (in (0,1)); threshold to use in tTMTI. If set to NULL, then either TMTI (default) or rtTMTI is used.
`K`	Integer; Number of smallest p-values to use in rtTMTI. If se to NULL, then either TMTI (default) or tTMTI is used.
`verbose`	Logical, indicating whether or not to print progress.

Value

The number of marginal hypotheses that can be rejected with kFWER control.

Examples

```
nfalse = 50
m = 100
pvals = c (
  sort(runif(nfalse, 0, 0.05 / m)),
  sort(runif(m - nfalse, 0.1, 1))
)
kFWER_TMTI (
  pvals = pvals,
  k = 5,
  alpha = 0.05,
  verbose = FALSE
)
```

MakeY_C *Leading NA*

Description

Returns the transformed p-values (Y) from pre-sorted p-values and pre-truncated p-values. If not truncation is used, set m_full = m

Usage

```
MakeY_C(pvals, m)
```

Arguments

pvals	A NumericVector containing the truncated sorted p-values. It is important that this vector: 1) contains only the truncated p-values (i.e, those that fall below the truncation point) and 2) is sorted.
m	The total (i.e., non-truncated) number of p-values.

MakeZ_C *Leading NA*

Description

Returns the TMTI_infinity statistic from pre-sorted, pre-truncated vector of p-values. If no truncation is used, set m_full = m

Usage

```
MakeZ_C(pvals, m)
```

Arguments

pvals	A NumericVector containing the truncated sorted p-values. It is important that this vector: 1) contains only the truncated p-values (i.e, those that fall below the truncation point) and 2) is sorted.
m	The total (i.e., non-truncated) number of p-values.

MakeZ_C_nsmall *Leading NA*

Description

Returns the transformed p-values (Y) from pre-sorted p-values and pre-truncated p-values when n < m - 1

Usage

```
MakeZ_C_nsmall(pvals, n, m)
```

Arguments

pvals	A NumericVector containing the truncated sorted p-values. It is important that this vector: 1) contains only the truncated p-values (i.e, those that fall below the truncation point) and 2) is sorted.
n	A positive number (or Inf) indicating which type of local minimum to consider. Defaults to Infm, corresponding to the global minimum.
m	The total (i.e., non-truncated) number of p-values.

---

rtTMTI_CDF *Computes the analytical version of the rtMTI_infty CDF. When m>100, this should not be used.*

Description

Computes the analytical version of the rtMTI_infty CDF. When m>100, this should not be used.

Usage

```
rtTMTI_CDF(x, m, K)
```

Arguments

x	Point in which to evaluate the CDF.
m	Number of independent tests to combine.
K	Integer; the truncation point to use.

Value

The probability that the test statistic is at most x assuming independence under the global null hypothesis.

Examples

```
rtTMTI_CDF(0.05, 100, 10)
```

`TestSet_C` *Leading NA*

Description

Tests a user-specified subset in a CTP, using a user-supplied local test

Usage

```
TestSet_C(
  LocalTest,
  pSub,
  pRest,
  alpha,
  is_subset_sequence,
  EarlyStop,
  verbose
)
```

Arguments

`LocalTest`	A function that returns a double in (0, 1).
`pSub`	A vector with the p-values of the set to be tested.
`pRest`	A vector containing the remaining p-values.
`alpha`	Double indicating the significance level.
`is_subset_sequence`	Logical indicating whether the supplied subset of p_values corresponds to the pSub.size() smallest overall p-values.
`EarlyStop`	Logical indicating whether to exit as soon as a non-significant p-value is found.
`verbose`	Logical indicating whether to print progress.

`TestSet_LocalTest` *Test a subset of hypotheses in its closure using a user-specified local test*

Description

Test a subset of hypotheses in its closure using a user-specified local test

Usage

```
TestSet_LocalTest(
  LocalTest,
  pvals,
  subset,
  alpha = 0.05,
  EarlyStop = FALSE,
  verbose = FALSE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  is.sorted = FALSE,
  ...
)

TestSet_localTest(
  localTest,
  pvals,
  subset,
  alpha = 0.05,
  EarlyStop = FALSE,
  verbose = FALSE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  is.sorted = FALSE,
  ...
)
```

Arguments

`LocalTest`	Function which defines a combination test.
`pvals`	Numeric vector of p-values.
`subset`	Numeric vector; the subset to be tested.
`alpha`	Numeric; the level to test at, if stopping early. Defaults to 0.05.
`EarlyStop`	Logical; set to TRUE to stop as soon as a hypothesis can be accepted at level alpha. This speeds up the procedure, but now only provides lower bounds on the p-values for the global test.
`verbose`	Logical; set to TRUE to print progress.

mc.cores	Number of cores to parallelize onto.
chunksize	Integer indicating the size of chunks to parallelize. E.g., if setting chunksize = mc.cores, each time a parallel computation is set up, each worker will perform only a single task. If mc.cores > chunksize, some threads will be inactive.
is.sorted	Logical, indicating whether the supplied p-values are already is.sorted. Defaults to FALSE.
...	Additional arguments.
localTest	A function specifying a local test (deprecated).

Value

The adjusted p-value for the test of the hypothesis that there are no false hypotheses among the selected subset.

Examples

```
## Simulate p-values; 10 from false hypotheses, 10 from true
pvals = sort(c(
  rbeta(10, 1, 20), # Mean value of .1
  runif(10)
))
## Test whether the highest 10 contain any false hypotheses using a Bonferroni test
TestSet_LocalTest(function(x) {
  min(c(1, length(x) * min(x)))
}, pvals, subset = 11:20)
```

TestSet_TMTI *Test a subset of hypotheses in its closure using the TMTI*

Description

Test a subset of hypotheses in its closure using the TMTI

Usage

```
TestSet_TMTI(
  pvals,
  subset,
  alpha = 0.05,
  tau = NULL,
  K = NULL,
  EarlyStop = FALSE,
  verbose = FALSE,
  gammaList = NULL,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  is.sorted = FALSE,
  ...
)
```

Arguments

pvals	Numeric vector of p-values.
subset	Numeric vector; the subset to be tested.
alpha	Numeric; the level to test at, if stopping early. Defaults to 0.05.
tau	Numeric; the treshold to use if using rTMTI. Set to NULL for TMTI or rtTMTI. Defaults to NULL.
K	Integer; The number of p-values to use if using rtTMTI. Set to NULL for TMTI or tTMTI. Defaults to NULL.
EarlyStop	Logical; set to TRUE to stop as soon as a hypothesis can be accepted at level alpha. This speeds up the procedure, but now only provides lower bounds on the p-values for the global test.
verbose	Logical; set to TRUE to print progress.
gammaList	List of functions. Must be such that the ith element is the gamma function for sets of size i. Set to NULL to bootstrap the functions assuming independence. Defaults to NULL.
mc.cores	Number of cores to parallelize onto.
chunksize	Integer indicating the size of chunks to parallelize. E.g., if setting chunksize = mc.cores, each time a parallel computation is set up, each worker will perform only a single task. If mc.cores > chunksize, some threads will be inactive.
is.sorted	Logical, indicating the p-values are pre-sorted. Defaults to FALSE.
...	Additional arguments.

Value

The adjusted p-value for the test of the hypothesis that there are no false hypotheses among the selected subset.

Examples

```
## Simulate p-values; 10 from false hypotheses, 10 from true
pvals = sort(c(
  rbeta(10, 1, 20), # Mean value of .1
  runif(10)
))
## Test whether the highest 10 contain any false hypotheses
TestSet_TMTI(pvals, subset = 11:20)
```

TMTI *Computes the TMTI test for a joint hypothesis given input p-values.*

Description

A package to compute TMTI tests, perform closed testing procedures with quadratic shortcuts and to generate confidence sets for the number of false hypotheses among m tested hypotheses.

Usage

```
TMTI(
  pvals,
  n = Inf,
  tau = NULL,
  K = NULL,
  gamma = NULL,
  B = 1000,
  m_max = 100,
  is.sorted = FALSE,
  ...
)
```

Arguments

pvals	A vector of pvalues.
n	A positive number (or Inf) indicating which type of local minimum to consider. Defaults to Inf, corresponding to the global minimum.
tau	Number between 0 and 1 or NULL, describing the truncation level.
K	Integer between >1 and m describing the truncation index.
gamma	Function; function to be used as the gamma approximation. If NULL, then the gamma function will be bootstrapped assuming independence. Defaults to NULL.
B	Numeric; number of bootstrap replicates to be used when estimating the gamma function. If a gamma is supplied, this argument is ignored. Defaults to 1e3.
m_max	Integer; the highest number of test for which the analytical computation of the TMTI CDF is used. When m is above m_max it will be bootstrapped or user supplied instead.
is.sorted	Logical, indicating whether the supplied p-values are already is.sorted. Defaults to FALSE.
...	Additional parameters.

Value

A p-value from the TMTI test

Author(s)

Phillip B. Mogensen <pbm@math.ku.dk>

Examples

```
## Simulate some p-values
## The first 10 are from false hypotheses, the next 10 are from true
pvals = c(
  rbeta(10, 1, 20), ## Mean value of .05
  runif(10)
```

```
)
TMTI(pvals)
```

`TMTI_CDF` *Computes the analytical version of the TMTI_infty CDF. When m>100, this should not be used.*

Description

Computes the analytical version of the TMTI_infty CDF. When m>100, this should not be used.

Usage

```
TMTI_CDF(x, m)
```

Arguments

`x`	Point in which to evaluate the CDF.
`m`	Number of independent tests to combine.

Value

The probability that the test statistic is at most x assuming independence under the global null hypothesis.

Examples

```
TMTI_CDF(0.05, 100)
```

`TopDown_C` *Leading NA*

Description

Computes a confidence set for the number of false hypotheses among all hypotheses

Usage

```
TopDown_C(LocalTest, pvals, alpha)
```

Arguments

`LocalTest`	A function that returns a double in (0, 1).
`pvals`	A vector of p-values.
`alpha`	A double indicating the significance level

TopDown_C_binary *Leading NA*

Description

Computes a confidence set for the number of false hypotheses among all hypotheses using a binary search

Usage

```
TopDown_C_binary(LocalTest, pvals, alpha, low, high, verbose)
```

Arguments

LocalTest	A function that returns a double in (0, 1).
pvals	A vector of p-values.
alpha	A double indicating the significance level
low	integer denoting the starting point for the search. Should start at zero.
high	integer denoting the end point of the search. Should end at pvals.size() - 1.
verbose	boolean, indicating whether to print progress.

TopDown_C_binary_subset *Leading NA*

Description

Computes a confidence set for the number of false hypotheses among a subset of using a binary search

Usage

```
TopDown_C_binary_subset(LocalTest, pSub, pRest, alpha, low, high, verbose)
```

Arguments

LocalTest	A function that returns a double in (0, 1).
pSub	A vector of p-values from the subset of interest.
pRest	A vector of the remaining p-values.
alpha	A double indicating the significance level
low	integer denoting the starting point for the search. Should start at zero.
high	integer denoting the end point of the search. Should end at pvals.size() - 1.
verbose	boolean, indicating whether to print progress.

`TopDown_LocalTest` *TopDown LocalTest algorithm for estimating a 1-alpha confidence set for the number of false hypotheses among a set.*

Description

TopDown LocalTest algorithm for estimating a 1-alpha confidence set for the number of false hypotheses among a set.

Usage

```
TopDown_LocalTest(
  LocalTest,
  pvals,
  subset = NULL,
  alpha = 0.05,
  verbose = FALSE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  direction = "binary",
  ...
)

TopDown_localTest(
  localTest,
  pvals,
  subset = NULL,
  alpha = 0.05,
  verbose = TRUE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  ...
)
```

Arguments

`LocalTest`	A function specifying a local test.
`pvals`	A vector of p-values.
`subset`	Numeric vector specifying a subset a p-values to estimate a confidence set for the number of false hypotheses for. Defaults to NULL corresponding to estimating a confidence set for the number of false hypotheses in the entire set.
`alpha`	Level in [0,1] at which to generate confidence set. Defaults to 0.05.
`verbose`	Logical, indicating whether or not to write out the progress. Defaults to TRUE.
`mc.cores`	Integer specifying the number of cores to parallelize onto.

chunksize	Integer indicating the size of chunks to parallelize. E.g., if setting chunksize = mc.cores, each time a parallel computation is set up, each worker will perform only a single task. If mc.cores > chunksize, some threads will be inactive.
direction	A string indicating whether to perform a binary search ('binary'/'b') or decreasing ('decreasing'/'d') search. Defaults to 'binary', which has better computational complexity.
...	Additional parameters.
localTest	A function specifying a local test (deprecated).

Value

A 1-alpha bound lower for the number of false hypotheses among the specified subset of the supplied p-values

Examples

```
## Simulate some p-values
## The first 10 are from false hypotheses, the next 10 are from true
pvals = c(
  rbeta(10, 1, 20), ## Mean value of .05
  runif(10)
)
## Estimate the confidence set using a local Bonferroni test
TopDown_LocalTest(function(x) {
  min(c(1, length(x) * min(x)))
}, pvals)
```

`TopDown_TMTI` *TopDown TMTI algorithm for estimating a 1-alpha confidence set for the number of false hypotheses among a set.*

Description

TopDown TMTI algorithm for estimating a 1-alpha confidence set for the number of false hypotheses among a set.

Usage

```
TopDown_TMTI(
  pvals,
  subset = NULL,
  alpha = 0.05,
  gammaList = NULL,
  verbose = TRUE,
  tau = NULL,
  K = NULL,
```

```
  is.sorted = FALSE,
  mc.cores = 1L,
  chunksize = 4 * mc.cores,
  direction = "binary",
  ...
)
```

Arguments

pvals	A vector of p-values.
subset	Numeric vector specifying a subset a p-values to estimate a confidence set for the number of false hypotheses for. Defaults to NULL corresponding to estimating a confidence set for the number of false hypotheses in the entire set.
alpha	Level in [0,1] at which to generate confidence set. Defaults to 0.05.
gammaList	List of pre-specified gamma functions. If NULL, the functions will be approximated by bootstrap assuming independence. Defaults to NULL.
verbose	Logical, indicating whether or not to write out the progress. Defaults to TRUE.
tau	Numerical (in (0,1)); threshold to use in tTMTI. If set to NULL, then either TMTI (default) or rtTMTI is used.
K	Integer; Number of smallest p-values to use in rtTMTI. If se to NULL, then either TMTI (default) or tTMTI is used.
is.sorted	Logical, indicating whether the supplied p-values are already is.sorted. Defaults to FALSE.
mc.cores	Number of cores to parallelize onto.
chunksize	Integer indicating the size of chunks to parallelize. E.g., if setting chunksize = mc.cores, each time a parallel computation is set up, each worker will perform only a single task. If mc.cores > chunksize, some threads will be inactive.
direction	A string indicating whether to perform a binary search ('binary'/'b') or decreasing ('decreasing'/'d') search. Defaults to 'binary', which has better computational complexity.
...	Additional parameters.

Value

A 1-alpha lower bound for the number of false hypotheses among the set of supplied p-values

Examples

```
## Simulate some p-values
## The first 10 are from false hypotheses, the next 10 are from true
pvals = c(
  rbeta(10, 1, 20), ## Mean value of .05
  runif(10)
)
TopDown_TMTI(pvals)
```

tTMTI_CDF *Computes the analytical version of the tTMTI_infty CDF. When m>100, this should not be used.*

Description

Computes the analytical version of the tTMTI_infty CDF. When m>100, this should not be used.

Usage

```
tTMTI_CDF(x, m, tau)
```

Arguments

x	Point in which to evaluate the CDF.
m	Number of independent tests to combine.
tau	The truncation point of the tTMTI procedure.

Value

The probability that the test statistic is at most x assuming independence under the global null hypothesis.

Examples

```
tTMTI_CDF(0.05, 100, 0.05)
```

Index