Package ‘MSPRT’

January 20, 2025

Type Package

Title A Modified Sequential Probability Ratio Test (MSPRT)

Version 3.0

Date 2020-11-11

Author Sandipan Pramanik [aut, cre],
Valen E. Johnson [aut],
Anirban Bhattacharya [aut]

Maintainer Sandipan Pramanik <sandy.pramanik@gmail.com>

Description
Given the maximum available sample size (N) for an experiment, and the target levels of Type I and II error probabilities, this package designs a modified SPRT (MSPRT). For any designed MSPRT
the package can also obtain its operating characteristics and implement the test for a given sequentially observed data. The MSPRT is defined in a manner very similar to Wald's initial proposal.
The proposed test has shown evidence of reducing the average sample size required to perform statistical hypothesis tests at specified levels of significance and power. Currently, the package implements one-sample proportion tests, one and two-sample z tests, and one and two-sample t tests. A brief user guidance for this package is provided below. One can also refer to the supplemental information for the same.

Imports nleqslv, ggplot2, ggpubr, foreach, iterators, parallel,
doParallel, datasets, graphics, grDevices, methods, stats,
utils

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2020-11-13 10:20:05 UTC

Contents

`MSPRT-package` *A Modified Sequential Probability Ratio Test (MSPRT)*

Description

Given the maximum available sample size (N) for an experiment, and the target levels of Type I and II error probabilities, this package designs a modified SPRT (MSPRT). For any designed MSPRT the package can also obtain its operating characteristics and implement the test for a given sequentially observed data. The MSPRT is defined in a manner very similar to Wald's initial proposal. The proposed test has shown evidence of reducing the average sample size required to perform statistical hypothesis tests at specified levels of significance and power. Currently, the package implements one-sample proportion tests, one and two-sample z tests, and one and two-sample t tests. A brief user guidance for this package is provided below. One can also refer to the supplemental information for the same.

Details

Package:	MSPRT
Type:	Package
Version:	3.0
Date:	11-11-2020
License:	GPL>=2

Author(s)

Sandipan Pramanik [aut, cre], Valen E. Johnson [aut], Anirban Bhattacharya [aut]

Maintainer: Sandipan Pramanik <sandy.pramanik@gmail.com>

---

design.MSPRT *Designing the MSPRT*

Description

Given the maximum available sample size and prespecified Type I & II error probabilities, this function designs/obtains the corresponding MSPRT.

Usage

```
design.MSPRT(test.type, side = "right", theta0, theta1 = T,
             Type1.target = 0.005, Type2.target = 0.2,
             N.max, N1.max, N2.max,
             sigma = 1, sigma1 = 1, sigma2 = 1,
             batch.size, batch1.size, batch2.size,
             nReplicate = 1e+06, verbose = T, seed = 1)
```

Arguments

`test.type` Character. Type of test. Currently, the package only allows

- `oneProp` for one-sample proportion tests
- `oneZ` for one-sample z tests
- `oneT` for one-sample t tests
- `twoZ` for two-sample z tests
- `twoT` for two-sample t tests.

`side` Character. Direction of the composite alternative hypothesis. `right` for $H_1 : \theta > \theta_0$ (**default**), and `left` for $H_1 : \theta < \theta_0$.

`theta0` Numeric. Hypothesized value of effect size (θ_0) under H_0.

Default: 0.5 in one-sample proportion tests, and 0 for others.

`theta1` Logical, numeric or list (two components with names `'right'` and `'left'`).

- If `FALSE`, no comparison is done under the alternative hypothesis.
- If `TRUE` (**Default**), comparison is done at the fixed-design alternative effect size (θ_a).

- If numeric, this can only be in case of one-sided tests (that is, `side = "right"` or `"left"`). The comparison is done at the specified numeric value of the alternative effect size.
- If list, this can only be in case of two-sided tests (that is, `side = "both"`). The list has to be of the form `list("right" =` θ_1`, "left" =` θ_2`)`. Then the comparison is done at alternative effect sizes θ_1 and θ_2.

Note: In case of two-sided tests at a given level of significance, there are two effect sizes under H_1 (one on the right of H_0 and one on the left) that corresponds to the same Type II error probability (or power). This list provides users with the ability where he/she can replace θ_1 and θ_2 by any effect sizes from each side in the form of a list as mentioned above, and can get the designed MSPRT together with its operating characteristics at those effect sizes.

`Type1.target` Numeric within [0,1]. Prespecified level of Type I error probability.

Default: 0.005.

The MSPRT exactly maintains its Type I error probability at this value.

`Type2.target` Numeric within [0,1]. Prespecified level of Type 2 error probability.

Default: 0.2.

The MSPRT approximately maintains its Type II error probability at this value at the corresponding fixed-design alternative (θ_a).

`N.max` Positive integer. Maximum available sample size in one-sample tests.

`N1.max` Positive integer. Maximum available sample size from Group-1 in two-sample tests.

`N2.max` Positive integer. Maximum available sample size from Group-2 in two-sample tests.

`sigma` Positive numeric. Known standard deviation in one-sample z tests.

Default: 1.

`sigma1` Positive numeric. Known standard deviation for Group-1 in two-sample z tests.

Default: 1.

`sigma2` Positive numeric. Known standard deviation for Group-2 in two-sample z tests.

Default: 1.

`batch.size` Integer vector. A vector denoting the number of observations that are planned to be observed at each sequential step in one-sample tests.

Default:

- Proportion and z tests: `rep(1, N.max)`.
- t tests: `c(2, rep(1, N.max-1))`.

Default values mean the sequential analysis is performed after observing each observation. This corresponds to a sequential MSPRT. If any batch size is more than 1 (or more than 2 in the 1st step for t test) it corresponds to a group sequential MSPRT.

Note: First batch size for t tests needs to be at least 2. The length of batch.size equals to the maximum number of planned sequential analyses.

`batch1.size` Integer vector. A vector denoting the number of observations that are planned to be observed from Group-1 at each sequential step in two-sample tests.

Default:

- z tests: `rep(1, N1.max)`.
- t tests: `c(2, rep(1, N1.max-1))`.

Default values mean the sequential analysis is performed after observing each observation from Group-1.

`batch2.size` Integer vector. A vector denoting the number of observations that are planned to be observed from Group-2 at each sequential step in two-sample tests.

Default:

- z tests: `rep(1, N2.max)`.
- t tests: `c(2, rep(1, N2.max-1))`.

Default values mean the sequential analysis is performed after observing each observation from Group-2.

`nReplicate` Positive integer. Total number of replications to be used in Monte Carlo simulation for calculating the termination threshold and the operating characteristics of the MSPRT.

Default: 1,000,000.

`verbose` Logical. If `TRUE` (**default**), returns messages of the current proceedings. Otherwise it doesn't.

`seed` Integer. Random number generating seed.

Default: 1.

Value

List. The list has the following named components in case of one-sided one-sample tests:

`TypeI.attained` Numeric in [0,1]. Type I error probability attained by the designed MSPRT.

`Type2.attained` Numeric in [0,1]. Type II error probability attained by the designed MSPRT at the specified alternative effect size `theta1`. Returned only if `theta1` is `TRUE` or numeric.

`N` List.

- If `theta1 = FALSE`, the list has one component named `H0`. It stores an integer vector of length `nReplicate`. This is the vector of sample size required by the MSPRT for each of `nReplicate` Monte Carlo simulations under H_0.
- If `theta1` is `TRUE` or numeric, the list has two components named `H0` and `H1`. Each of these stores an integer vector of length `nReplicate`. The stored vector under `H0` is the same as in `theta1 = FALSE`. The `H1` component stored the vector of sample size required by the MSPRT for each of `nReplicate` Monte Carlo simulations under the specified alternative effect size.

`EN` Numeric vector.

- If `theta1 = FALSE`, the vector is of length 1. It is the number of samples required on average by the MSPRT under H_0.
- If `theta1` is `TRUE` or numeric, the vector is of length 2. They are the number of samples required on average by the MSPRT under H_0 (first component) and the specified alternative effect size (second component), respectively.

`UMPBT` or `theta.UMPBT`
: The UMPBT alternative. `UMPBT` in case of one-sample proportion test and `theta.UMPBT` in case of all the other tests. Their types are the same as their output from `UMPBT.alt` function.

  Note: Not returned in t tests as it depends on the data.

`theta1`
: Returned only if `theta1` is anything but `FALSE`. Stores the effect size under H_1 where the operating characteristic of the MSPRT is obtained. Of the same type as the argument `theta1`.

`Type2.fixed.design`
: Numeric in [0,1]. Type II error probability attained by the fixed design test with sample size `N.max` and Type I error probability `Type1.target` at the alternative effect size `theta1`.

`RejectH0.threshold`
: Positive numeric. Threshold for rejecting H_0 in the MSPRT.

`RejectH1.threshold`
: Positive numeric. Threshold for accepting H_1 in the MSPRT.

`termination.threshold`
: Positive numeric. Termination threshold of the MSPRT.

In case of one-sided two-sample tests the above components are returned with following modifications:

`N:`
: List.

  - If `theta1 = FALSE` the list has one component named `H0`.
  - `theta1` is `TRUE` or numeric, the list has two components named `H0` and `H1`. Each of the named components `H0` and `H1` contains a list with two components named `Group1` and `Group2`. Each of these contains the same vector corresponding to Group-1 and Group-2. In each of these, it contains the sample size required by the MSPRT in each of `nReplicate` Monte Carlo simulations under the respective effect size for the respective group.

`EN`
: List.

  - If `theta1 = FALSE` the list has one component named `H0`.
  - If `theta1` is `TRUE` or numeric, the list has two components named `H0` and `H1`.

  Each of the named components `H0` or `H1` contains a list with two components named `Group1` and `Group2`. In each of these, it contains the sample size required on average by the MSPRT under the respective effect size for the respective group.

In case of two-sided tests the above components are returned with following modifications:

`Type2.attained`
: Numeric vector of length 2 with both elements in [0,1]. The first and second component is the Type II error probability of the MSPRT at the specified alternative effect sizes `theta1$right` and `theta1$left`, respectively.

N — This is the same as in one-sided tests if `theta1` is `FALSE`. If `theta1` is `TRUE` or a two-component list with names `right` and `left`, this is a list with three components with names `H0`, `right` and `left` instead of a two-component list with names `H0` and `H1`. Quantities stored under these components are the same as in one-sided tests except the quantities under `right` and `left` are the same performance of the designed MSPRT at the specified alternative effect sizes `theta1$right` and `theta1$left`, respectively.

EN — Numeric vector. The same as in one-sided tests if `theta1` is `FALSE`. If `theta1` is `TRUE` or a two-component list with names `right` and `left`, this is a numeric vector of length 3, where the first, second and third components are the average required sample size under H_0, and at the specified alternative effect sizes `theta1$right` and `theta1$left`, respectively.

Additionally, the output list also contains the provided arguments of design.MSPRT, and

nAnalyses — Positive integer. This is the maximum number of sequential analyses that is planned. This equals to the `length(batch.size)` in one-sample tests, and to the `length(batch1.size)` and `length(batch2.size)` in two-sample tests.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
##### one-sample proportion test #####

## right-sided
#design.MSPRT(test.type = 'oneProp', side = 'right',
#             N.max = 20)

## left-sided
#design.MSPRT(test.type = 'oneProp', side = 'right',
#             N.max = 20)

## two-sided
#design.MSPRT(test.type = 'oneProp', side = 'both',
#             N.max = 20)


##### one-sample z test #####

## right-sided
#design.MSPRT(test.type = 'oneZ', side = 'right',
#             N.max = 20)
```

```
## left-sided
#design.MSPRT(test.type = 'oneZ', side = 'right',
#             N.max = 20)

## two-sided
#design.MSPRT(test.type = 'oneZ', side = 'both',
#              N.max = 20)


##### one-sample t test #####

## right-sided
#design.MSPRT(test.type = 'oneT', side = 'right',
#             N.max = 20)

## left-sided
#design.MSPRT(test.type = 'oneT', side = 'right',
#             N.max = 20)

## two-sided
#design.MSPRT(test.type = 'oneT', side = 'both',
#             N.max = 20)


##### two-sample z test #####

## right-sided
#design.MSPRT(test.type = 'twoZ', side = 'right',
#             N1.max = 20, N2.max = 20)

## left-sided
#design.MSPRT(test.type = 'twoZ', side = 'left',
#             N1.max = 20, N2.max = 20)

## two-sided
#design.MSPRT(test.type = 'twoZ', side = 'both',
#             N1.max = 20, N2.max = 20)


##### two-sample t test #####

## right-sided
#design.MSPRT(test.type = 'twoT', side = 'right',
#             N1.max = 20, N2.max = 20)

## left-sided
#design.MSPRT(test.type = 'twoT', side = 'left',
#             N1.max = 20, N2.max = 20)

## two-sided
#design.MSPRT(test.type = 'twoT', side = 'both',
#             N1.max = 20, N2.max = 20)
```

design.MSPRT_oneProp *Internal MSPRT function: Designing the MSPRT for one-sample proportion tests*

Description

design.MSPRT calls this function for designing the MSPRT in one-sample proportion tests. Users please refer to design.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

design.MSPRT_oneT *Internal MSPRT function: Designing the MSPRT for one-sample t tests*

Description

design.MSPRT calls this function for designing the MSPRT in one-sample t tests. Users please refer to design.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

design.MSPRT_oneZ *Internal MSPRT function: Designing the MSPRT for one-sample z tests*

Description

design.MSPRT calls this function for designing the MSPRT in one-sample z tests. Users please refer to design.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

design.MSPRT_twoT — *Internal MSPRT function: Designing the MSPRT for two-sample t tests*

Description

design.MSPRT calls this function for designing the MSPRT in two-sample t tests. Users please refer to design.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

design.MSPRT_twoZ — *Internal MSPRT function: Designing the MSPRT for two-sample z tests*

Description

design.MSPRT calls this function for designing the MSPRT in two-sample z tests. Users please refer to design.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

effectiveN.oneProp — *Calculating effective maximum sample size to be used in designing the MSPRT in one-sample proportion test*

Description

Given a maximum sample size that is planned to use, this function obtains the maximum sample size (N) that is suggested to use in designing the MSPRT for one-sample proportion tests.

Usage

```
effectiveN.oneProp(N, side = "right", Type1 = 0.005, theta0 = 0.5,
                   plot.it = T)
```

Arguments

N	Positive integer. Maximum sample that is intended to use.
side	Character. Direction of the composite alternative hypothesis. `right` for $H_1 : \theta > \theta_0$ (**default**), and `left` for $H_1 : \theta < \theta_0$.
Type1	Numeric in [0,1]. Prespecified Type I error probability. **Default:** 0.005.
theta0	Numeric. Hypothesized value of effect size (θ_0) under H_0. **Default:** 0.5.
plot.it	Logical. If `TRUE` (**default**), returns a plot. Otherwise it doesn't.

Value

Positive integer. This is suggested to use in `OCandASN.MSPRT` as the maximum availeble sample size (N) to design the MSPRT for one-sample proportion tests.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
effectiveN.oneProp(N = 30)
```

---

fixed_design.alt *Fixed-design alternative*

Description

Given a sample size and prespecified Type I & II error probabilities, this function obtains the fixed-design alternative (θ_a) for testing the point null hypothesis $H_0 : \theta = \theta_0$.

Usage

```
fixed_design.alt(test.type, side = "right", theta0,
                 N, N1, N2, Type1 = 0.005, Type2 = 0.2,
                 sigma = 1, sigma1 = 1, sigma2 = 1)
```

Arguments

test.type	Character. Type of test. Currently, the package only allows • oneProp for one-sample proportion tests • oneZ for one-sample z tests • oneT for one-sample t tests • twoZ for two-sample z tests • twoT for two-sample t tests.
side	Character. Direction of the composite alternative hypothesis. right for $H_1 : \theta > \theta_0$ (**default**), and left for $H_1 : \theta < \theta_0$.
theta0	Numeric. Hypothesized value of effect size (θ_0) under H_0. **Default:** 0.5 in one-sample proportion tests, and 0 for others.
N	Positive integer. Sample size in one-sample tests.
N1	Positive integer. Sample size from Group-1 in two-sample tests.
N2	Positive integer. Sample size from Group-2 in two-sample tests.
Type1	Numeric in [0,1]. Prespecified Type I error probability. **Default:** 0.005.
Type2	Numeric in [0,1]. Prespecified Type II error probability. **Default:** 0.2.
sigma	Positive numeric. Known standard deviation in one-sample z tests. **Default:** 1.
sigma1	Positive numeric. Known standard deviation for Group-1 in two-sample z tests. **Default:** 1.
sigma2	Positive numeric. Known standard deviation for Group-2 in two-sample z tests. **Default:** 1.

Value

Numeric. The fixed-design alternative effect size (θ_a).

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
##### one-sample proportion test #####

## right-sided
fixed_design.alt(test.type = "oneProp", N = 30)

## left-sided
fixed_design.alt(side = "left", test.type = "oneProp", N = 30)
```

```
##### one-sample z test #####

## right-sided
fixed_design.alt(test.type = "oneZ", N = 30)

## left-sided
fixed_design.alt(side = "left", test.type = "oneZ", N = 30)


##### one-sample t test #####

## right-sided
fixed_design.alt(test.type = "oneT", N = 30)

## left-sided
fixed_design.alt(side = "left", test.type = "oneT", N = 30)


##### two-sample z test #####

## right-sided
fixed_design.alt(test.type = "twoZ", N1 = 30, N2 = 30)

## left-sided
fixed_design.alt(side = "left", test.type = "twoZ", N1 = 30, N2 = 30)


##### two-sample t test #####

## right-sided
fixed_design.alt(test.type = "twoT", N1 = 30, N2 = 30)

## left-sided
fixed_design.alt(side = "left", test.type = "twoT", N1 = 30, N2 = 30)
```

implement.MSPRT *Implementing the MSPRT*

Description

This function implements the MSPRT for a sequentially observed data.

Usage

```
implement.MSPRT(obs, obs1, obs2, design.MSPRT.object,
                termination.threshold, test.type, side = "right",
                theta0, Type1.target = 0.005, Type2.target = 0.2,
                N.max, N1.max, N2.max,
                sigma = 1, sigma1 = 1, sigma2 = 1,
```

```
batch.size, batch1.size, batch2.size,
verbose = T, plot.it = 2)
```

Arguments

obs
: Numeric vector. The vector of data in the order they are sequentially observed for one-sample tests. Note: Its length can't exceed the length of `batch.size`.

obs1
: Numeric vector. The vector of data in the order they are sequentially observed from Group-1 for two-sample tests. Note: Its length can't exceed the length of `batch1.size`.

obs2
: Numeric vector. The vector of data in the order they are sequentially observed from Group-2 for two-sample tests. Note: Its length can't exceed the length of `batch2.size`.

design.MSPRT.object
: List. The output returned from `design.MSPRT` corresponding to the MSPRT for which the operating characteristics are desired.

termination.threshold
: Positive numeric. Termination threshold of the designed MSPRT.

test.type
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

side
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

theta0
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

Type1.target
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

Type2.target
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

N.max
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

N1.max
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

N2.max
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

sigma
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

sigma1
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

sigma2
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

batch.size
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

batch1.size
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

batch2.size
: Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

verbose
: Logical. If TRUE (**default**), returns messages of the current proceedings. Otherwise it doesn't.

plot.it
: 0, 1 or 2 (**default**).
  - if `plot.it=0`, no plot is returned.
  - if `plot.it=1`, only the ggplot object required to get a comparison plot is returned, but it's not plotted.
  - if `plot.it=2`, a comparison plot and the corresponding ggplot object is returned.

Details

If design.MSPRT.object is provided, one can only additionally provide nReplicate, nCore, verbose and seed (Easier option). Otherwise, just like in design.MSPRT, all the other arguments together with termination.threshold (obtained from design.MSPRT) needs to be provided adequately.

Value

List. The list has the following named components in case of one-sided one-sample tests:

- n: Positive integer. Number of samples required to reach the decision.
- decision: Character. The decision reached. The possibilities are 'accept', 'reject' and 'continue'. They respectively correspond to accepting H_0, rejecting H_0 and continue sampling.
- RejectH0.threshold: Positive numeric. Threshold for rejecting H_0 in the MSPRT.
- RejectH1.threshold: Positive numeric. Threshold for accepting H_1 in the MSPRT.
- LR: Numeric vector. Vector of weighted likelihood ratios (proportion tests) or likelihood ratios (z tests) or Bayes factor (t tests) that are computed at each step of sequential analysis until either a decision is reached or the maximum available number of samples (N.max in one-sample tests, or N1.max and N2.max in two-sample tests) has been used.
- UMPBT alternative: This stores the UMPBT alternative(s) as
  - UMPBT for proportion tests. Of the same type as it is returned by UMPBT.alt in these tests.
  - theta.UMPBT for z and t tests. This is a numeric in case of z tests and a numeric vector in case of t tests. For t tests the UMPBT alternative depends on the data. So the numeric vector returned in this case contains the UMPBT alternative computed at step of sequential analysis and is based on all data observed until that step.

In case of two-sample tests, the n output above is replaced by n1 and n2. They are positive integers and refer to the number of samples from Group-1 and 2 required to reach the decision.

In case of two-sided tests at level of significance α, the MSPRT carries out a right and a left sided test simultaneously at level of significance $\alpha/2$. In this case the outputs are same as above with following changes in components in the returned list:

- LR: List. It has two components named right and left corresponding to the right and left sided tests of size $\alpha/2$. Each of these components stores the vector of weighted likelihood ratios (proportion tests) or likelihood ratios (z tests) or Bayes factor (t tests) that are computed at each step of sequential analysis until either a decision is reached or the maximum available number of samples (N.max in one-sample tests, or N1.max and N2.max in two-sample tests) has been used for that sided test.

`UMPBT` or `theta.UMPBT`
: List with two components named `right` and `left` corresponding to the right and left sided tests of size $\alpha/2$. Each of these contains the UMPBT alternative (of the same type as the output from `UMPBT.alt` for the test with respective sides.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
#################### one-sample proportion test ####################

#### right sided ####
### design
#design.oneprop.right = design.MSPRT(test.type = 'oneProp', side = 'right',
#                                    N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0.5  # change effect size to experiment
#y = rbinom(20, 1, theta.gen)
#implement.oneprop.right = implement.MSPRT(obs = y,
#                                          design.MSPRT.object = design.oneprop.right)

#### left sided ####
### design
#design.oneprop.left = design.MSPRT(test.type = 'oneProp', side = 'left',
#                                   N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0.5  # change effect size to experiment
#y = rbinom(20, 1, theta.gen)
#implement.oneprop.left = implement.MSPRT(obs = y,
#                                          design.MSPRT.object = design.oneprop.left)

#### both sided ####
### design
#design.oneprop.both = design.MSPRT(test.type = 'oneProp', side = 'both',
#                                   N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0.5  # change effect size to experiment
#y = rbinom(20, 1, theta.gen)
#implement.oneprop.both = implement.MSPRT(obs = y,
```

```
#                                         design.MSPRT.object = design.oneprop.both)


#################### one-sample z test ####################

#### right sided ####
### design
#design.onez.right = design.MSPRT(test.type = 'oneZ', side = 'right',
#                                 N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y = rnorm(20, theta.gen, design.onez.right$sigma)
#implement.onez.right = implement.MSPRT(obs = y,
#                                       design.MSPRT.object = design.onez.right)

#### left sided ####
### design
#design.onez.left = design.MSPRT(test.type = 'oneZ', side = 'left',
#                                N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y = rnorm(20, theta.gen, design.onez.left$sigma)
#implement.onez.left = implement.MSPRT(obs = y,
#                                      design.MSPRT.object = design.onez.left)

#### both sided ####
### design
#design.onez.both = design.MSPRT(test.type = 'oneZ', side = 'both',
#                                N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y = rnorm(20, theta.gen, design.onez.both$sigma)
#implement.onez.both = implement.MSPRT(obs = y,
#                                      design.MSPRT.object = design.onez.both)


#################### one-sample t test ####################

#### right sided ####
### design
#design.onet.right = design.MSPRT(test.type = 'oneT', side = 'right',
#                                 N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y = rnorm(20, theta.gen, 1)
```

```
#implement.onet.right = implement.MSPRT(obs = y,
#                                       design.MSPRT.object = design.onet.right)

#### left sided ####
### design
#design.onet.left = design.MSPRT(test.type = 'oneT', side = 'left',
#                                N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y = rnorm(20, theta.gen, 1)
#implement.onet.left = implement.MSPRT(obs = y,
#                                      design.MSPRT.object = design.onet.left)

#### both sided ####
### design
#design.onet.both = design.MSPRT(test.type = 'oneT', side = 'both',
#                                N.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y = rnorm(20, theta.gen, 1)
#implement.onet.both = implement.MSPRT(obs = y,
#                                      design.MSPRT.object = design.onet.both)


#################### two-sample z test ####################

#### right sided ####
### design
#design.twoz.right = design.MSPRT(test.type = 'twoZ', side = 'right',
#                                 N1.max = 20, N2.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y1 = rnorm(20, theta.gen/2, design.twoz.right$sigma1)
#y2 = rnorm(20, -theta.gen/2, design.twoz.right$sigma2)
#implement.twoz.right = implement.MSPRT(obs1 = y1, obs2 = y2,
#                                       design.MSPRT.object = design.twoz.right)

#### left sided ####
### design
#design.twoz.left = design.MSPRT(test.type = 'twoZ', side = 'left',
#                                N1.max = 20, N2.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y1 = rnorm(20, theta.gen/2, design.twoz.left$sigma1)
#y2 = rnorm(20, -theta.gen/2, design.twoz.left$sigma2)
```

```
#implement.twoz.left = implement.MSPRT(obs1 = y1, obs2 = y2,
#                                      design.MSPRT.object = design.twoz.left)

#### both sided ####
### design
#design.twoz.both = design.MSPRT(test.type = 'twoZ', side = 'both',
#                                N1.max = 20, N2.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y1 = rnorm(20, theta.gen/2, design.twoz.both$sigma1)
#y2 = rnorm(20, -theta.gen/2, design.twoz.both$sigma2)
#implement.twoz.both = implement.MSPRT(obs1 = y1, obs2 = y2,
#                                      design.MSPRT.object = design.twoz.both)


#################### two-sample t test ####################

#### right sided ####
### design
#design.twot.right = design.MSPRT(test.type = 'twoT', side = 'right',
#                                 N1.max = 20, N2.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y1 = rnorm(20, theta.gen/2, 1)
#y2 = rnorm(20, -theta.gen/2, 1)
#implement.twot.right = implement.MSPRT(obs1 = y1, obs2 = y2,
#                                       design.MSPRT.object = design.twot.right)

#### left sided ####
### design
#design.twot.left = design.MSPRT(test.type = 'twoT', side = 'left',
#                                N1.max = 20, N2.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
#y1 = rnorm(20, theta.gen/2, 1)
#y2 = rnorm(20, -theta.gen/2, 1)
#implement.twot.left = implement.MSPRT(obs1 = y1, obs2 = y2,
#                                      design.MSPRT.object = design.twot.left)

#### both sided ####
### design
#design.twot.both = design.MSPRT(test.type = 'twoT', side = 'both',
#                                N1.max = 20, N2.max = 20)

### implementation
#set.seed(1)
#theta.gen = 0  # change effect size to experiment
```

```
#y1 = rnorm(20, theta.gen/2, 1)
#y2 = rnorm(20, -theta.gen/2, 1)
#implement.twot.both = implement.MSPRT(obs1 = y1, obs2 = y2,
#                                      design.MSPRT.object = design.twot.both)
```

---

implement.MSPRT_oneProp *Internal MSPRT function: Implementing the MSPRT in one-sample proportion tests*

Description

implement.MSPRT calls this function for implementing the MSPRT in one-sample proportion tests. Users please refer to implement.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

---

implement.MSPRT_oneT *Internal MSPRT function: Implementing the MSPRT in one-sample t tests*

Description

implement.MSPRT calls this function for implementing the MSPRT in one-sample t tests. Users please refer to implement.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

---

implement.MSPRT_oneZ *Internal MSPRT function: Implementing the MSPRT in one-sample z tests*

Description

implement.MSPRT calls this function for implementing the MSPRT in one-sample z tests. Users please refer to implement.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

implement.MSPRT_twoT — *Internal MSPRT function: Implementing the MSPRT in two-sample t tests*

Description

implement.MSPRT calls this function for implementing the MSPRT in two-sample t tests. Users please refer to implement.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

implement.MSPRT_twoZ — *Internal MSPRT function: Implementing the MSPRT in two-sample z tests*

Description

implement.MSPRT calls this function for implementing the MSPRT in two-sample z tests. Users please refer to implement.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

Nstar — *Sample size required to achieve higher significance*

Description

Given the sample size that is available at a lower level of significance, this function calculates the sample size that is required for achieving a higher level of significance so that a desired level of Type II error probability is maintained at a desired effect size.

Usage

```
Nstar(test.type, N, N1, N2,
      N.increment = 1, N1.increment = 1, N2.increment = 1,
      lower.signif = 0.05, higher.signif = 0.005, theta0,
      side = "right", Type2.target = 0.2, theta,
      sigma = 1, sigma1 = 1, sigma2 = 1, plot.it = T)
```

Arguments

`test.type`	Character. Type of test. Currently, the package only allows • `oneProp` for one-sample proportion tests • `oneZ` for one-sample z tests • `oneT` for one-sample t tests • `twoZ` for two-sample z tests • `twoT` for two-sample t tests.
`N`	Positive integer. Sample size available at the lower level of significance in one-sample tests.
`N1`	Positive integer. Sample size available from Group-1 at the lower level of significance in two-sample tests.
`N2`	Positive integer. Sample size available from Group-2 at the lower level of significance in two-sample tests.
`N.increment`	Positive integer. Increment in sample size allowed while searching for the sample size that is required for achieving the higher level of significance.
`N1.increment`	Positive integer. Increment in sample size from Group-1 allowed while searching for the sample size that is required for achieving the higher level of significance.
`N2.increment`	Positive integer. Increment in sample size from Group-2 allowed while searching for the sample size that is required for achieving the higher level of significance.
`lower.signif`	Numeric within [0,1]. Lower level of significance. **Default** 0.05.
`higher.signif`	Numeric within [0,1]. Higher level of significance. **Default:** 0.005.
`theta0`	Numeric. Hypothesized value of effect size (θ_0) under H_0. **Default:** 0.5 in one-sample proportion tests, and 0 for others.
`side`	Character. Direction of the composite alternative hypothesis. `right` for $H_1 : \theta > \theta_0$ (**default**), and `left` for $H_1 : \theta < \theta_0$.
`Type2.target`	Numeric within [0,1]. Prespecified level of Type 2 error probability. **Default:** 0.2.
`theta`	Numeric. Effect size value where `Type2.target` Type II error probability is desired at both levels of significance. **Default:** Fixed-design alternative (θ_a) at the lower level of significance; that is, the effect size where the fixed design test with N samples and level of significance `lower.signif` has the Type II error probability `Type2.target`.
`sigma`	Positive numeric. Known standard deviation in one-sample z tests. **Default:** 1.
`sigma1`	Positive numeric. Known standard deviation for Group-1 in two-sample z tests. **Default:** 1.
`sigma2`	Positive numeric. Known standard deviation for Group-2 in two-sample z tests. **Default:** 1.
`plot.it`	Logical. If `TRUE` (**default**), returns a plot. Otherwise it doesn't.

Value

- One-sample tests: Numeric. The required sample size.
- Two-sample tests: Numeric vector of length 2. The first and second components store the sample sizes required respectively from Group 1 and 2 for achieving the higher level of significance.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
##### one-sample proportion test #####

## right-sided
Nstar(test.type = "oneProp", N = 30)

## left-sided
Nstar(test.type = "oneProp", side = "left", N = 30)


##### one-sample z test #####

## right-sided
Nstar(test.type = "oneZ", N = 30)

## left-sided
Nstar(test.type = "oneZ", side = "left", N = 30)


##### one-sample t test #####

## right-sided
Nstar(test.type = "oneT", N = 30)

## left-sided
Nstar(test.type = "oneT", side = "left", N = 30)


##### two-sample z test #####

## right-sided
Nstar(test.type = "twoZ", N1 = 30, N2 = 30)

## left-sided
Nstar(test.type = "twoZ", side = "left", N1 = 30, N2 = 30)
```

```
##### two-sample t test #####

## right-sided
Nstar(test.type = "twoT", N1 = 30, N2 = 30)

## left-sided
Nstar(test.type = "twoT", side = "left", N1 = 30, N2 = 30)
```

OCandASN.MSPRT *Operating characteristics (OC) and Average Sample Number (ASN) of a designed MSPRT*

Description

This function obtains the operating characteristics, that is the probability of accepting H_0 and the sample size required on average for reaching a decision, for a designed MSPRT at the specified effect size(s).

Usage

```
OCandASN.MSPRT(theta, design.MSPRT.object,
               termination.threshold, test.type, side = "right",
               theta0, Type1.target = 0.005, Type2.target = 0.2,
               N.max, N1.max, N2.max,
               sigma = 1, sigma1 = 1, sigma2 = 1,
               batch.size, batch1.size, batch2.size,
               nReplicate = 1e+06, nCore = max(1, detectCores() - 1),
               verbose = T, seed = 1)
```

Arguments

`theta` Numeric vector. Vector of effect size(s) where the operating characteristics of the MSPRT is desired.

`design.MSPRT.object`
: List. The output returned from `design.MSPRT` corresponding to the MSPRT for which the operating characteristics are desired.

`termination.threshold`
: Positive numeric. Termination threshold of the designed MSPRT.

`test.type` Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

`side` Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

`theta0` Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

`Type1.target` Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

`Type2.target` Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

`N.max` Same as in `design.MSPRT`. Not required if `design.MSPRT.object` is provided.

N1.max	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
N2.max	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
sigma	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
sigma1	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
sigma2	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
batch.size	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
batch1.size	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
batch2.size	Same as in design.MSPRT. Not required if design.MSPRT.object is provided.
nReplicate	Positive integer. Total number of replications to be used in Monte Carlo simulation for calculating the termination threshold and the operating characteristics of the MSPRT. **Default:** 1,000,000.
verbose	Logical. If TRUE (**default**), returns messages of the current proceedings. Otherwise it doesn't.
nCore	Positive integer. Total number of cores available for computation. Can be anything ≥ 1. **Default:** detectCores() - 1. That is, 1 less than the total number of available cores.
seed	Integer. Random number generating seed. **Default:** 1.

Details

If design.MSPRT.object is provided, one can only additionally provide nReplicate, nCore, verbose and seed (Easier option). Otherwise, just like in design.MSPRT, all the other arguments together with termination.threshold (obtained from design.MSPRT) needs to be provided adequately.

Value

Data frame.

- One-sample tests: The data frame has 3 columns named theta, acceptH0.prob and EN, and the number of rows equals to the number of effect sizes (length of theta) where the operating characteristics are evaluated. Each row corresponds to a particular value of theta (effect size). The columns respectively contain the value of a particular theta (effect size), and the probability of accepting the H_0 and the average sample size required by the MSPRT for reaching a decision thereat.
- Two-sample tests: The data frame has 4 columns named theta, acceptH0.prob, EN1 and EN2, and the number of rows equals to the number of effect sizes (length of theta) where the operating characteristics are evaluated. Each row corresponds to a particular value of theta (effect size). The columns respectively contain the value of a particular theta (effect size), and the probability of accepting the H_0 at that effect size, and the average sample size from Group-1 & 2 that is required by the MSPRT for reaching a decision thereat.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
#################### one-sample proportion test ####################

#### right sided ####
### design
#design.oneprop.right <- design.MSPRT(test.type = 'oneProp', side = 'right',
#                                      N.max = 20)

### OC and ASN
#OC.oneprop.right <- OCandASN.MSPRT(theta = seq(design.oneprop.right$theta0, 1,
#                                               length.out = 3),
#                                    design.MSPRT.object = design.oneprop.right)

#### left sided ####
### design
#design.oneprop.left = design.MSPRT(test.type = 'oneProp', side = 'left',
#                                    N.max = 20)

### OC and ASN
#OC.oneprop.left = OCandASN.MSPRT(theta = seq(0, design.oneprop.right$theta0,
#                                             length.out = 3),
#                                  design.MSPRT.object = design.oneprop.left)

#### both sided ####
### design
#design.oneprop.both = design.MSPRT(test.type = 'oneProp', side = 'both',
#                                    N.max = 20)

### OC and ASN
#OC.oneprop.both = OCandASN.MSPRT(theta = seq(0, 1, length.out = 3),
#                                  design.MSPRT.object = design.oneprop.both)


#################### one-sample z test ####################

#### right sided ####
### design
#design.onez.right = design.MSPRT(test.type = 'oneZ', side = 'right',
#                                  N.max = 20)

### OC and ASN
#OC.onez.right = OCandASN.MSPRT(theta = seq(design.onez.right$theta0,
#                                   design.onez.right$theta0 + 3*design.onez.right$sigma,
```

```
#                                            length.out = 3),
#                                design.MSPRT.object = design.onez.right)

#### left sided ####
### design
#design.onez.left = design.MSPRT(test.type = 'oneZ', side = 'left',
#                                 N.max = 20)

### OC and ASN
#OC.onez.left = OCandASN.MSPRT(theta = seq(design.onez.left$theta0 - 3*design.onez.left$sigma,
#                                           design.onez.left$theta0,
#                                           length.out = 3),
#                               design.MSPRT.object = design.onez.left)

#### both sided ####
### design
#design.onez.both = design.MSPRT(test.type = 'oneZ', side = 'both',
#                                 N.max = 20)

### OC and ASN
#OC.onez.both = OCandASN.MSPRT(theta = seq(design.onez.both$theta0 - 3*design.onez.both$sigma,
#                                     design.onez.both$theta0 + 3*design.onez.both$sigma,
#                                           length.out = 3),
#                               design.MSPRT.object = design.onez.both)


#################### one-sample t test ####################

#### right sided ####
### design
#design.onet.right = design.MSPRT(test.type = 'oneT', side = 'right',
#                                  N.max = 20)

### OC and ASN
#OC.onet.right = OCandASN.MSPRT(theta = seq(design.onet.right$theta0, 1,
#                                            length.out = 3),
#                                design.MSPRT.object = design.onet.right)

#### left sided ####
### design
#design.onet.left = design.MSPRT(test.type = 'oneT', side = 'left',
#                                 N.max = 20)

### OC and ASN
#OC.onet.left = OCandASN.MSPRT(theta = seq(-1, design.onet.left$theta0,
#                                           length.out = 3),
#                               design.MSPRT.object = design.onet.left)

#### both sided ####
### design
#design.onet.both = design.MSPRT(test.type = 'oneT', side = 'both',
#                                 N.max = 20)
```

```
### OC and ASN
#OC.onet.both = OCandASN.MSPRT(theta = seq(-1, 1, length.out = 3),
#                              design.MSPRT.object = design.onet.both)


################### two-sample z test ###################

#### right sided ####
### design
#design.twoz.right = design.MSPRT(test.type = 'twoZ', side = 'right',
#                                 N1.max = 20, N2.max = 20)

### OC and ASN
#OC.twoz.right = OCandASN.MSPRT(theta = seq(design.twoz.right$theta0,
#                                           design.twoz.right$theta0 + 2,
#                                           length.out = 3),
#                               design.MSPRT.object = design.twoz.right)

#### left sided ####
### design
#design.twoz.left = design.MSPRT(test.type = 'twoZ', side = 'left',
#                                N1.max = 20, N2.max = 20)

### OC and ASN
#OC.twoz.left = OCandASN.MSPRT(theta = seq(design.twoz.left$theta0 - 2,
#                                          design.twoz.left$theta0,
#                                          length.out = 3),
#                              design.MSPRT.object = design.twoz.left)

#### both sided ####
### design
#design.twoz.both = design.MSPRT(test.type = 'twoZ', side = 'both',
#                                N1.max = 20, N2.max = 20)

### OC and ASN
#OC.twoz.both = OCandASN.MSPRT(theta = seq(design.twoz.both$theta0 - 2,
#                                          design.twoz.both$theta0 + 2,
#                                          length.out = 3),
#                             design.MSPRT.object = design.twoz.both)


################### two-sample t test ###################

#### right sided ####
### design
#design.twot.right = design.MSPRT(test.type = 'twoT', side = 'right',
#                                 N1.max = 20, N2.max = 20)

### OC and ASN
#OC.twot.right = OCandASN.MSPRT(theta = seq(design.twot.right$theta0,
#                                           design.twot.right$theta0 + 2,
#                                           length.out = 3),
#                               design.MSPRT.object = design.twot.right)
```

```
#### left sided ####
### design
#design.twot.left = design.MSPRT(test.type = 'twoT', side = 'left',
#                                N1.max = 20, N2.max = 20)

### OC and ASN
#OC.twot.left = OCandASN.MSPRT(theta = seq(design.twot.left$theta0 - 2,
#                                          design.twot.left$theta0,
#                                          length.out = 3),
#                              design.MSPRT.object = design.twot.left)

#### both sided ####
### design
#design.twot.both = design.MSPRT(test.type = 'twoT', side = 'both',
#                                N1.max = 20, N2.max = 20)

### OC and ASN
#OC.twot.both = OCandASN.MSPRT(theta = seq(design.twot.both$theta0 - 2,
#                                          design.twot.both$theta0 + 2,
#                                          length.out = 3),
#                              design.MSPRT.object = design.twot.both)
```

`OCandASN.MSPRT_oneProp` *Internal MSPRT function: OC and ASN of a designed MSPRT in one-sample proportion tests*

Description

`OCandASN.MSPRT` calls this function for obtaining the OC and ASN of a designed MSPRT for one-sample proportion tests. Users please refer to `OCandASN.MSPRT`.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

`OCandASN.MSPRT_oneT` *Internal MSPRT function: OC and ASN of a designed MSPRT in one-sample t tests*

Description

`OCandASN.MSPRT` calls this function for obtaining the OC and ASN of a designed MSPRT for one-sample t tests. Users please refer to `OCandASN.MSPRT`.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

OCandASN.MSPRT_oneZ — *Internal MSPRT function: OC and ASN of a designed MSPRT in one-sample z tests*

Description

OCandASN.MSPRT calls this function for obtaining the OC and ASN of a designed MSPRT for one-sample z tests. Users please refer to OCandASN.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

OCandASN.MSPRT_twoT — *Internal MSPRT function: OC and ASN of a designed MSPRT in two-sample t tests*

Description

OCandASN.MSPRT calls this function for obtaining the OC and ASN of a designed MSPRT for two-sample t tests. Users please refer to OCandASN.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

OCandASN.MSPRT_twoZ — *Internal MSPRT function: OC and ASN of a designed MSPRT in two-sample z tests*

Description

OCandASN.MSPRT calls this function for obtaining the OC and ASN of a designed MSPRT for two-sample z tests. Users please refer to OCandASN.MSPRT.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

`Type2.fixed_design` *Type II error probability of fixed design tests*

Description

Obtains the Type II error probability of fixed-design tests for testing the point null hypothesis $H_0 : \theta = \theta_0$.

Usage

```
Type2.fixed_design(theta, test.type, side = "right", theta0,
                   N, N1, N2, Type1 = 0.005,
                   sigma = 1, sigma1 = 1, sigma2 = 1)
```

Arguments

`theta`	Numeric. Effect size where the Type II error probability is desired.
`test.type`	Character. Type of test. Currently, the package only allows • `oneProp` for one-sample proportion tests • `oneZ` for one-sample z tests • `oneT` for one-sample t tests • `twoZ` for two-sample z tests • `twoT` for two-sample t tests.
`side`	Character. Direction of the composite alternative hypothesis. `right` for $H_1 : \theta > \theta_0$ (**default**), and `left` for $H_1 : \theta < \theta_0$.
`theta0`	Numeric. Hypothesized value of effect size (θ_0) under H_0. **Default:** 0.5 in one-sample proportion tests, and 0 for others.
`N`	Positive integer. Sample size in one-sample tests.
`N1`	Positive integer. Sample size from Group-1 in two-sample tests.
`N2`	Positive integer. Sample size from Group-2 in two-sample tests.
`Type1`	Numeric in [0,1]. Prespecified Type I error probability. **Default:** 0.005.
`sigma`	Positive numeric. Known standard deviation in one-sample z tests. **Default:** 1.
`sigma1`	Positive numeric. Known standard deviation for Group-1 in two-sample z tests. **Default:** 1.
`sigma2`	Positive numeric. Known standard deviation for Group-2 in two-sample z tests. **Default:** 1.

Value

Numeric in [0,1]. The Type II error probability of the fixed-design test at the specified effect size value `theta`.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
##### one-sample proportion test #####

## right-sided
Type2.fixed_design(theta = seq(0, 1, length.out = 10),
                   test.type = "oneProp", N = 30)

## left-sided
Type2.fixed_design(theta = seq(0, 1, length.out = 10), side = "left",
                   test.type = "oneProp", N = 30)


##### one-sample z test #####

## right-sided
Type2.fixed_design(theta = seq(0, 1, length.out = 10),
                   test.type = "oneZ", N = 30)

## left-sided
Type2.fixed_design(theta = seq(-1, 0, length.out = 10), side = "left",
                   test.type = "oneZ", N = 30)


##### one-sample t test #####

## right-sided
Type2.fixed_design(theta = seq(0, 1, length.out = 10),
                   test.type = "oneT", N = 30)

## left-sided
Type2.fixed_design(theta = seq(-1, 0, length.out = 10), side = "left",
                   test.type = "oneT", N = 30)


##### two-sample z test #####

## right-sided
Type2.fixed_design(theta = seq(0, 1, length.out = 10),
                   test.type = "twoZ", N1 = 30, N2 = 30)

## left-sided
Type2.fixed_design(theta = seq(-1, 0, length.out = 10), side = "left",
                   test.type = "twoZ", N1 = 30, N2 = 30)
```

```
##### two-sample t test #####

## right-sided
Type2.fixed_design(theta = seq(0, 1, length.out = 10),
                   test.type = "twoT", N1 = 30, N2 = 30)

## left-sided
Type2.fixed_design(theta = seq(-1, 0, length.out = 10), side = "left",
                   test.type = "twoT", N1 = 30, N2 = 30)
```

UMPBT.alt *UMPBT alternative*

Description

Given a sample size and prespecified Type I & II error probabilities, this function obtains the objective alternative in the Uniformly Most Powerful Bayesian Test (UMPBT).

Usage

```
UMPBT.alt(test.type, side = "right", theta0,
          N, N1, N2, Type1 = 0.005,
          sigma = 1, sigma1 = 1, sigma2 = 1,
          obs, sd.obs, obs1, obs2, pooled.sd)
```

Arguments

`test.type`	Character. Type of test. Currently, the package only allows • `oneProp` for one-sample proportion tests • `oneZ` for one-sample z tests • `oneT` for one-sample t tests • `twoZ` for two-sample z tests • `twoT` for two-sample t tests.
`side`	Character. Direction of the composite alternative hypothesis. `right` for $H_1 : \theta > \theta_0$ (**default**), and `left` for $H_1 : \theta < \theta_0$.
`theta0`	Numeric. Hypothesized value of effect size (θ_0) under H_0. **Default:** 0.5 in one-sample proportion tests, and 0 for others.
`N`	Positive integer. Sample size in one-sample tests.
`N1`	Positive integer. Sample size from Group-1 in two-sample tests.
`N2`	Positive integer. Sample size from Group-2 in two-sample tests.
`Type1`	Numeric in [0,1]. Prespecified Type I error probability. **Default:** 0.005.
`sigma`	Positive numeric. Known standard deviation in one-sample z tests. **Default:** 1.

`sigma1`	Positive numeric. Known standard deviation for Group-1 in two-sample z tests. **Default:** 1.
`sigma2`	Positive numeric. Known standard deviation for Group-2 in two-sample z tests. **Default:** 1.
`obs`	Numeric vector. The vector of observations based on which the UMPBT alternative in one-sample t test is determined. Either `obs` or `sd.obs` is required.
`sd.obs`	Positive numeric. The standard deviation (with divisor n-1) of observations based on which the UMPBT alternative in one-sample t test is determined. Either `obs` or `sd.obs` is required.
`obs1`	Numeric vector. The vector of observations from Group-1 based on which the UMPBT alternative in two-sample t test is determined. Either both `obs1` and `obs2`, or `pooled.sd` is required.
`obs2`	Numeric vector. The vector of observations from Group-2 based on which the UMPBT alternative in two-sample t test is determined. Either both `obs1` and `obs2`, or `pooled.sd` is required.
`pooled.sd`	Positive numeric. The pooled standard deviation of observations from Group-1 and 2 based on which the UMPBT alternative in two-sample t test is determined. Either both `obs1` and `obs2`, or `pooled.sd` is required.

Value

List with two named components `theta` and `mix.prob` in one-sample proportion test. In this case, the UMPBT alternative is a mixture distribution of two points. `theta` contains the two points (effect sizes) and `mix.prob` contains their respective mixing probabilities.

Numeric in case of all the other tests. It is the UMPBT alternative effect size.

Author(s)

Sandipan Pramanik, Valen E. Johnson and Anirban Bhattacharya

References

Johnson, V. E. (2013a). Revised standards for statistical evidence.Proceed-ings of the National Academy of Sciences, 110(48):19313-19317. [Article]

Johnson, V. E. (2013b). Uniformly most powerful Bayesian tests.TheAnnals of Statistics, 41(4):1716-1741. [Article]

Pramanik S., Johnson V. E. and Bhattacharya A. (2020+). A Modified Sequential Probability Ratio Test. [Arxiv]

Examples

```
##### one-sample proportion test #####

## right-sided
UMPBT.alt(test.type = "oneProp", N = 30)

## left-sided
```

```
UMPBT.alt(side = "left", test.type = "oneProp", N = 30)


##### one-sample z test #####

## right-sided
UMPBT.alt(test.type = "oneZ", N = 30)

## left-sided
UMPBT.alt(side = "left", test.type = "oneZ", N = 30)


##### one-sample t test #####

## observed data

set.seed(1)
x = rnorm(n = 30, mean = 0, sd = 1.5)

## right-sided
UMPBT.alt(test.type = "oneT", N = 30, obs = x)

## left-sided
UMPBT.alt(side = "left", test.type = "oneT", N = 30, obs = x)


##### two-sample z test #####

## right-sided
UMPBT.alt(test.type = "twoZ", N1 = 30, N2 = 30)

## left-sided
UMPBT.alt(side = "left", test.type = "twoZ", N1 = 30, N2 = 30)


##### two-sample t test #####

## observed data

set.seed(1)
x1 = rnorm(n = 30, mean = 0, sd = 1.5)
x2 = rnorm(n = 30, mean = 0, sd = 1.5)

## right-sided
UMPBT.alt(test.type = "twoT", N1 = 30, N2 = 30,
          obs1 = x1, obs2 = x2)

## left-sided
UMPBT.alt(side = "left", test.type = "twoT", N1 = 30, N2 = 30,
          obs1 = x1, obs2 = x2)
```

Index