Package ‘MSCCT’

July 29, 2025

Title Multiple Survival Crossing Curves Tests

Version 1.0.2

Description Tests of comparison of two or more survival curves. Allows for comparison of more than two survival curves whether the proportional hazards hypothesis is verified or not.

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.5)

LazyData true

Imports boot, stats, survival

Suggests survRM2, testthat (>= 3.0.0), TSHRC

Config/testthat/edition 3

URL https://github.com/HMinP/MSCCT

BugReports https://github.com/HMinP/MSCCT/issues

NeedsCompilation no

Author Hugo MINA PASSI [aut, cre, cph],
Olayidé BOUSSARI [aut]

Maintainer Hugo MINA PASSI <hugo.minapassi@gmail.com>

Repository CRAN

Date/Publication 2025-07-29 19:20:27 UTC

Contents

`data_not_PH` *A simulation with three groups without the Proportional Hazards hypothesis*

Description

This dataframe represents a simulation of a study where the three survival curves cross each other.

Usage

```
data_not_PH
```

Format

`data_not_PH`**:**

A data frame with 600 rows and 3 columns:

time Time of events in months with truncation at 60 months

status Indicator of censorship. 1 denotes an event, 0 denotes a censor

arm Integer from 0 to 2. Indicates the group the patient belongs to

---

`data_under_PH` *A simulation with three groups under the Proportional Hazards hypothesis*

Description

This dataframe represents a simulation of a study under the Proportional Hazards hypothesis. All three survival curves follows an exponential distribution with different parameters.

Usage

```
data_under_PH
```

Format

`data_under_PH`**:**

A data frame with 600 rows and 3 columns:

time Time of events in months with truncation at 60 months

status Indicator of censorship. 1 denotes an event, 0 denotes a censor

arm Integer from 0 to 2. Indicates the group the patient belongs to

---

`multi_lr` *(Weighted) Log-rank test for comparison of two or more survival curves.*

Description

Performs a global log-rank test for comparing two or more survival curves.

Usage

```
multi_lr(df, weights, test = c("lr", "gw", "fh"), rho = 1, gamma = 0)
```

Arguments

`df`	A data frame with columns : • `time` : positive numbers, time-to-event; • `status` : vector of integer, 0 or 1. 0 is (right) censoring, 1 is event; • `arm` : a factor or object that can be coerced to one. The group the patient belongs to. Must have at least two levels.
`weights`	An object that can be coerced to a matrix. The weights used for the tests. Can be omitted (see Details);
`test`	If `weights` is omitted, specify the test to perform. Possible values are `lr` for log-rank, `gw` for Gehan-Wilcoxon, and `fh` for Flemming-Harrington;
`rho`, `gamma`	The parameters for Flemming-Harrington test. Default is (rho,gamma)=(1,0), which is also called the Peto-Peto test.

Details

`weights` contains the chosen weights for the test. It must be a vector, a matrix or an object that can be coerced to a matrix, like a data frame (passed as argument to `as.matrix`). Can be omitted.

If not given (default), then perform either a log-rank test, a Gehan-Wilcoxon test or a Fleming-Harrington test depending on the choice of `test`.

If `weights` is a one-dimension vector, its length must be equal to the number of distinct time of event and throws an error if it is not true. In this case, `multi_lr()` performs a weighted log-rank test with the specified weights.

If `weights` is a matrix (or a two-dimension object), its number of rows must be equal to the number of distinct time of event and throws an error if it is not true. In this case, `multi_lr()` performs as many tests as the number of columns in `weights`. The first test is a weighted log-rank test with weights the first column of `weights`, the second test is a weighted log-rank test with weights the second column of `weights`, and so on.

Value

An object of class `multi_lr` containing:

- `U` : Statistics of tests;
- `p` : The corresponding p-values;
- `degree` : Degrees of freedom of the statistics of tests;
- The argument `test`, changed to "chosen" if weights are given.

Examples

```
# Log-rank test
multi_lr(data_not_PH)

# Gehan-Wilcoxon test
multi_lr(data_not_PH, test="gw")

# It is possible to run several tests with different weights at a time
evt_time = unique(data_not_PH$time[data_not_PH$status == 1])
nb_evt_time = length(evt_time)
weights = matrix(runif(nb_evt_time*3), ncol=3)
multi_lr(data_not_PH, weights=weights)
```

`multi_rmst` *Test of RMST for comparing two or more survival curves*

Description

Performs the test of Restricted Mean Survival Time for two or more survival curves by comparing the difference of areas under survival curves.

Usage

```
multi_rmst(df, tau, method = p.adjust.methods, nboot = 500)
```

Arguments

`df`	A dataframe with columns : • `time` : positive numbers, time-to-event; • `status` : vector of integer, 0 or 1. 0 is (right) censoring, 1 is event; • `arm` : a factor or object that can be coerced to one. The group the patient belongs to. Must have at least two levels.
`tau`	The truncation time, default is the lowest of the max(time) of the groups;
`method`	The correction used for the p-values. Must be in p.adjust.methods. Default is the Holm correction. Unused if number of groups equals two.
`nboot`	Number of bootstrap samples;

Details

For each group, the Restricted Mean Survival Time at time tau (RMST(tau)) is computed as the area under the survival curve between time 0 and tau. The test of RMST is a pairwise multiple comparison test. For each pair of groups, it tests whether the difference between the RMST(tau) is zero or not. If the difference is not null, then the survival curves cannot be equal.

For exactly two groups, a single test is performed. For more than two survival curves, it compares each survival curve to every other curves and tests the global null hypothesis "all curves are equal" against the hypothesis "the curves are not all equal".

Value

An object of class `multi_rmst` containing :

- `rmst_mat` RMST estimation for each arm;
- `results` A matrix. Each row represents a comparison of two curves and contains the difference of RMST, its standard deviation, the p-value and the adjusted p-value;
- `p` The p-value of the global test;
- `nb_tests` The number of performed tests;
- The parameters `tau`, `method` and `nboot`.

References

Royston, P., & Parmar, M. K. (2013). Restricted mean survival time: an alternative to the hazard ratio for the design and analysis of randomized trials with a time-to-event outcome. BMC medical research methodology, 13, 1-15.

Examples

```
multi_rmst(data_under_PH, tau = 36, nboot = 300)
multi_rmst(data_not_PH, tau = 36, method = "BH", nboot = 300)
```

`multi_ts` *Two-staged test for comparison of two or more survival curves.*

Description

Performs a two-stage test for each pair of survival curves and apply a correction in case of several comparisons.

Usage

```
multi_ts(df, method = p.adjust.methods, eps = 0.1, nboot = 100)
```

Arguments

`df`	A dataframe with columns : • `time` : positive numbers, time-to-event; • `status` : vector of integer, 0 or 1. 0 is (right) censoring, 1 is event; • `arm` : a factor or object that can be coerced to one. The group the patient belongs to. Must have at least two levels.
`method`	The correction used for the p-values. Must be in p.adjust.methods. Default is the Holm correction. Unused if number of groups equals two.
`eps`	A number such that 0 < `eps` < 0.5. See reference for more information;
`nboot`	A positive integer, number of bootstrap sample for the second stage;

Details

For a two-stage test, the first stage is a log-rank test. If the first test is significant, then the whole procedure stops and we conclude that the survival curves are different. If it is not significant, then the survival curves are either equal or crossing each other and the log-rank test can't conclude the difference. A second test is performed to distinguish these two cases.

For multiple curves comparison, the two-stage test is a pairwise test. A two-stage test is performed for each pair of curves.

Value

An object of class `multi_ts` containing:

- `results` : A matrix. Each row represents a comparison of two curves and contains the p-values for both stage, the global p-value and the adjusted global p-value;
- `p` : The p-value for the global test;
- `nb_tests` : The number of performed Two-Stage tests;
- the parameters `eps`, `method` and `nboot`.

References

- Qiu, P., & Sheng, J. (2008). A two-stage procedure for comparing hazard rate functions. Journal of the Royal Statistical Society Series B: Statistical Methodology, 70(1), 191-208. Chen, Zhongxue & Huang, Hanwen & Qiu, Peihua. (2017).
- Chen, Z., Huang, H., & Qiu, P. (2017). An improved two-stage procedure to compare hazard curves. Journal of Statistical Computation and Simulation, 87(9), 1877-1886.

Examples

```
# test with a quarter of the data frame data_not_PH
ind = c(1:100, 401:500, 801:900)
multi_ts(data_not_PH[ind,], method = "BH", eps = 0.1, nboot = 10)
```

`print.multi_lr` *Print method for the multiple log-rank test*

Description

Print method for the multiple log-rank test

Usage

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

Arguments

`x`	An object of class `multi_lr` as returned by `multi_lr()`;
`...`	For compatibility with the `print` method, unused and to be ignored.

Value

None

Examples

```
x = multi_lr(data_not_PH)
print(x)
```

`print.multi_rmst` *Print method for the multiple test of RMST*

Description

Print method for the multiple test of RMST

Usage

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

Arguments

`x`	An object of class `multi_rmst` as returned by `multi_rmst()`;
`...`	For compatibility with the `print` method, unused and to be ignored.

Value

None

Examples

```
x = multi_rmst(data_under_PH, tau = 36, nboot = 300)
print(x)
```

`print.multi_ts` *Print method for the multiple Two-Stage test*

Description

Print method for the multiple Two-Stage test

Usage

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

Arguments

`x`	An object of class `multi_ts` as returned by `multi_ts()`;
`...`	For compatibility with the `print` method, unused and to be ignored.

Value

None

Examples

```
# test with a quarter of the data frame data_not_PH
ind = c(1:100, 401:500, 801:900)
x = multi_ts(data_not_PH[ind,], method = "BH", eps = 0.1, nboot = 10)
print(x)
```

Index