

Package ‘sGBJ’

July 23, 2025

Type Package

Title Survival Extension of the Generalized Berk-Jones Test

Version 0.1.1

Description Implements an extension of the Generalized Berk-Jones (GBJ) statistic for survival data, sGBJ. It computes the sGBJ statistic and its p-value for testing the association between a gene set and a time-to-event outcome with possible adjustment on additional covariates. Detailed method is available at Villain L, Ferte T, Thiebaut R and Hejblum BP (2021) <[doi:10.1101/2021.09.07.459329](https://doi.org/10.1101/2021.09.07.459329)>.

License GPL (>= 3)

Depends R (>= 3.5.0)

Imports GBJ, stats, survival

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

URL <https://github.com/lauravillain/sGBJ>

BugReports <https://github.com/lauravillain/sGBJ/issues>

NeedsCompilation no

Author Laura Villain [aut, cre],
Thomas Ferte [aut],
Rodolphe Thiebault [aut],
Boris P. Hejblum [aut]

Maintainer Laura Villain <sistm.soft.maintain@gmail.com>

Repository CRAN

Date/Publication 2025-04-14 10:40:02 UTC

Contents

.epsilon_matrix	2
.survival_scores	3
ls_test_results	3
sGBJ	3
sGBJ_scores	4
surv_calc_scores_stats	5
Index	7

.epsilon_matrix	<i>.epsilon_matrix</i>
-----------------	------------------------

Description

Compute the epsilon matrix by permutation for the sGBJ_scores() function.

Usage

.epsilon_matrix(Z, nperm, surv, factor_matrix, covariates = NULL, dat)

Arguments

- Z the score vector returned by .survival_scores() function.
- nperm number of permutations performed to estimate the epsilon matrix. Default is 300.
- surv a [Surv](#) object of length n
- factor_matrix a n x p data.frame of the expression for the particular gene set of interest being tested
- covariates a n x 1 matrix of the covariates to adjust upon. Default is NULL
- dat data used to fit survival model returned by .survival_scores() function.

Value

The epsilon matrix.

<code>.survival_scores</code>	<code>.survival_scores</code>
-------------------------------	-------------------------------

Description

Compute the survival score

Usage

```
.survival_scores(factor_matrix, covariates = NULL, surv)
```

Arguments

- `factor_matrix` a `n x p` data.frame of the expression for the particular gene set of interest being tested
- `covariates` a matrix `n x l` of the covariates to adjust. Default is `NULL`
- `surv` a [Surv](#) object of length `n`

Value

A list of length 3 with the updated `factor_matrix` (same as `factor_matrix` but removing columns for which survival model failed to converge), the Z matrix and the data used to fit survival model.

<code>ls_test_results</code>	<i>A data file used for testing sGBJ</i>
------------------------------	--

Description

A data file used for testing sGBJ

<code>sGBJ</code>	<i>Compute the sGBJ statistic and its p-value quantifying a gene set expression association with survival</i>
-------------------	---

Description

This function is the main function of the sGBJ package to perform Gene Set Analysis in the context of time-to-event outcome.

Usage

```
sGBJ(surv, factor_matrix, covariates = NULL, nperm = 300)
```

Arguments

surv	a Surv object of length n
factor_matrix	a n x p data.frame of the expression for the particular gene set of interest being tested
covariates	a n x 1 matrix of the covariates to adjust upon. Default is NULL
nperm	number of permutations performed to estimate the epsilon matrix. Default is 300.

Value

The sGBJ statistic and its associated p-value associated

Examples

```
n <- 100
surv_data <- data.frame(Time = runif(n = n, min = 0, max = 100),
                        event = rbinom(n = n, size = 1, prob = 0.5))
surv <- survival::Surv(time = surv_data$Time, event = surv_data$event)

factor_matrix <- data.frame(P1 = rnorm(n = n),
                             P2 = rnorm(n = n))

sGBJ::sGBJ(surv, factor_matrix, nperm = 2)
```

sGBJ_scores	<i>Compute the sGBJ statistic along with its p-value quantifying the association between a gene set and survival outcome</i>
-------------	--

Description

Compute the sGBJ statistic along with its p-value quantifying the association between a gene set and survival outcome

Usage

```
sGBJ_scores(surv, factor_matrix, covariates = NULL, nperm = 300)
```

Arguments

surv	a Surv object of length n
factor_matrix	a n x p data.frame of the expression for the particular gene set of interest being tested
covariates	a n x 1 matrix of the covariates to adjust upon. Default is NULL
nperm	number of permutations performed to estimate the epsilon matrix. Default is 300.

Value

a list containing the sGBJ statistic estimation and its associated p-value

Examples

```
n <- 100
surv_data <- data.frame(Time = runif(n = n, min = 0, max = 100),
                        event = rbinom(n = n, size = 1, prob = 0.5))
surv <- survival::Surv(time = surv_data$Time, event = surv_data$event)

factor_matrix <- data.frame(P1 = rnorm(n = n),
                             P2 = rnorm(n = n))

sGBJ::sGBJ_scores(surv, factor_matrix, nperm = 2)

# with covariates

covariates <- data.frame(age = runif(n = n, 60, 90))

sGBJ_scores(surv, factor_matrix, nperm = 2, covariates = covariates)
```

surv_calc_scores_stats

surv_calc_scores_stats

Description

An adaptation of GBJ::calc_scores_stats() to survival context. Wrapper of sGBJ_scores() function.

Usage

```
surv_calc_scores_stats(null_model, factor_matrix, nperm = 300)
```

Arguments

null_model	An R cox model fitted with survival::coxph().
factor_matrix	An n x p matrix with each factor as one column. There should be no missing data.
nperm	Number of permutations (default is 300)

Value

A list with the elements:

test_stats	The p score test statistics.
cor_mat	The p x p matrix giving the pairwise correlation of every test statistic pairs.

Examples

```
n <- 100
surv_data <- data.frame(Time = runif(n = n, min = 0, max = 100),
                        event = rbinom(n = n, size = 1, prob = 0.5))
surv <- survival::Surv(time = surv_data$Time, event = surv_data$event)

factor_matrix <- data.frame(P1 = rnorm(n = n),
                             P2 = rnorm(n = n))

covariates <- data.frame(age = runif(n = n, 60, 90))

null_model <- survival::coxph(surv ~ age, data = covariates, x = TRUE)
surv_reg_stats <- surv_calc_scores_stats(null_model = null_model,
                                         factor_matrix = factor_matrix,
                                         nperm = 2)#nperm = 300)

GBJ::GBJ(test_stats=surv_reg_stats$test_stats, cor_mat=surv_reg_stats$cor_mat)
```

Index

* **data**

- ls_test_results, [3](#)
- .epsilon_matrix, [2](#)
- .survival_scores, [3](#)

ls_test_results, [3](#)

sGBJ, [3](#)

sGBJ_scores, [4](#)

Surv, [2-4](#)

surv_calc_scores_stats, [5](#)