Package ‘RPEXE.RPEXT’

October 12, 2022

Title Reduced Piecewise Exponential Estimate/Test Software

Version 0.0.2

Author Gang Han [aut, cre],
Yu Zhang [aut]

Maintainer Gang Han <hangang.true@gmail.com>

URL https://github.com/hangangtrue/RPEXE.RPEXT

BugReports https://github.com/hangangtrue/RPEXE.RPEXT/issues

Description This reduced piecewise exponential survival software implements the likelihood ratio test and backward elimination procedure in Han, Schell, and Kim (2012 <doi:10.1080/19466315.2012.698945>, 2014 <doi:10.1002/sim.5915>), and Han et al. (2016 <doi:10.1111/biom.12590>). Inputs to the program can be either times when events/censoring occur or the vectors of total time on test and the number of events. Outputs of the programs are times and the corresponding p-values in the backward elimination. Details about the model and implementation are given in Han et al. 2014. This program can run in R version 3.2.2 and above.

Depends R (>= 3.2.2)

License GPL-3

Imports stats, graphics

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2020-05-21 06:30:03 UTC
R topics documented:

- bisection
- data2
- df
- exact_pvalue
- gamlik
- km
- kmvalue
- km_blacksolid
- km_combine
- km_log
- km_red
- km_redsolid
- loopcuts
- loopcuts_cut
- loopcuts_onestep
- loopcuts_t_c
- loopcuts_umbrella
- loopcuts_umbrella_cuttimes_mono
- loopcut_onestep_data
- pava_dfr
- pava_dfrd
- pava_ifr
- pexeest
- pexeest_times_censoring
- RPEXE.RPEXT
- RPEXEv1_2
- simple
- t100
- totaltest
- umbrella

Index

<table>
<thead>
<tr>
<th>bisection</th>
<th>Bisection algorithm in Beta distribution</th>
</tr>
</thead>
</table>

Description

Running bisection algorithm to search for a2, the minimizer of \((\log((a2)^{dea1}*(1-a2)^{dea2}-delta))^2\)

Usage

bisec(delta, dea1, dea2, upbd, lowbd)
Arguments

- **delta**: Test statistic in Han et al. (2012), \( \text{delta} = \left( \frac{\text{ttot1}}{\text{ttot1}+\text{ttot2}} \right)^{\text{dea1}} \left( \frac{\text{ttot2}}{\text{ttot1}+\text{ttot2}} \right)^{\text{dea2}} \);
- **dea1**: first parameter in Beta distribution (number of events from the first arm);
- **dea2**: second parameter in Beta distribution (number of events from the second arm);
- **upbd**: upper bound of \( a_2 \);
- **lowbd**: lower bound of \( a_2 \);

Value

- \( a_2 \)

Examples

- `bisec(-74.4824, 33, 98, 1, 0.252)`

---

**Description**

A breast cancer clinical trial dataset in Adelson et al. (2016).

**Usage**

`data(data2)`

**Details**

- first column - times: time to event
- second column - censor: censoring status; 0=censored, 1=event.
- third column - group: labels the single agent arm and combination arm

**References**

A dataset containing predictions for chemo-sensitivity and pathological response from Hatzis (2011)

**Usage**

```r
data(df)
```

**Details**

- `validate`: Validation status
- `drfs`: Censoring status; 0=censored, 1=event.
- `drfs.time`: Time to event or censoring
- `er.status`: ER status, P=positive, N=negative
- `chemo.pred`: Prediction for chemo sensitivity from the ACES predictor, sensitive or insensitive
- `pre.N`: Prediction of nodal status
- `pCR.RD`: pathological complete response (pCR) or residual disease (RD)
- `pre.grade`: prediction of tumor grade
- `pre.T`: T stage prediction
- `dlda30`: DLDA30 prediction for the pathological response.

**References**


---

This function computes the exact p-value from the likelihood ratio test

**Usage**

```r
exact_pvalue(ttot1,ttot2,dea1,dea2,mono)
```
**gamllik**

**Arguments**

- `ttot1`: total time on test 1
- `ttot2`: total time on test 2
- `dea1`: number of death 1
- `dea2`: number of death 2
- `mono`: 0: 2-sided hypothesis: \( H_0: \lambda_1 \) is equal to \( \lambda_2 \); \( H_1: \lambda_1 \) is not equal to \( \lambda_2 \) 1: 1-sided hypothesis: \( H_0: \lambda_1 \) is greater than or equal to \( \lambda_2 \); \( H_1: \lambda_1 \) is less than \( \lambda_2 \) 2: 1-sided hypothesis: \( H_0: \lambda_1 \) is less than or equal to \( \lambda_2 \); \( H_1: \lambda_1 \) is greater than \( \lambda_2 \)

**Value**

- \( a2 \): Beta distribution quantile computed using bisec.R \( pval \): p-value

**Examples**

```r
exact_pvalue(1, 302.04, 2, 25, 1)
```

---

**gamllik**

*Log likelihood from the gamma distribution*

**Description**

A function computing the log likelihood from the gamma distribution under an order restriction

**Usage**

```r
gamllik(structtime, structttot, structdeaths, time_die, ttot, deaths)
```

**Arguments**

- `structtime`: change-point times to be used to compute the likelihood value
- `structttot`: total time on test (tot) between each time point and the previous time point (or 0) corresponding to `structtime`
- `structdeaths`: number of deaths corresponding to `structttot`
- `time_die`: all event and censoring times from small to large
- `ttot`: total time on test corresponding to `time_die`
- `deaths`: the number of deaths corresponding to "ttot"

**Value**

log of the likelihood
Examples

time_die <- c(0.05, 0.08, 0.38, 0.41, 0.64)
ttot <- c(9.2, 5.8, 52.1, 5.8, 40.0)
deaths <- c(1, 1, 1, 1, 1)
structtime <- c(0.05, 0.64)
structttot <- c(9.2, 40.0)
structdeaths <- c(1, 5)
gamllik(structtime, structttot, structdeaths, time_die, ttot, deaths)

km (Kaplan-Meier curve)

Description

This function plots the Kaplan-Meier curve without returning outputs

Usage

km(time, censor, plotcens)

Arguments

time a vector of event or censoring time
censor a vector indicating censoring: 0 = censored; 1 = uncensored
plotcens 0: don’t add censored data symbol to the output curve 1: add censored data symbol to the output curve

Value

Kaplan-Meier curve only

Examples

t1 <- c(2, 3, 4, 5, 5, 7, 10, 12, 15)
c1 <- c(0, 0, 1, 0, 1, 0, 0)
km(t1, c1, 0)
kmvalue

Obtain values for Kaplan-Meier plotting

Description
Obtain values for Kaplan-Meier plotting

Usage
kmvalue(x)

Arguments
x
Nx2 data matrix, first column represents survival time of the i-th subject, second column represents censored flag (0 if not censored, 1 if censored)

Value
Values used for Kaplan-Meier plotting

Examples
t1 <- c(2,3,4,5,5,7,10,12,15)
c1 <- c(0,0,1,0,0,1,0,0)
x1<-cbind(t1,c1)
kmvalue(x1)

km_blacksolid
Kaplan-Meier curve

Description
This function plots the Kaplan-Meier curve without returning outputs

Usage
km_blacksolid(time, censor, plotcens)

Arguments
time
a vector of event or censoring time
censor
a vector indicating censoring: 0 = censored; 1 = uncensored
plotcens
0: don’t add censored data symbol to the output curve 1: add censored data symbol to the output curve
km_combine

Value
Kaplan-Meier curve only

Examples
```r
t1 <- c(2,3,4,5,5,7,10,12,15)
c1 <- c(0,0,1,0,0,1,0,0)
km_blacksolid(t1,c1,0)
```

km_combine

Comparing two Kaplan Meier curves in one plot

Description
The function compares two Kaplan Meier curves in one plot

Usage
```r
km_combine(x1, x2, pos = 0)
```

Arguments
- `x1` : Nx2 data matrix, first column represents survival time of the i-th subject, second column represents censored flag (0 if not censored, 1 if censored)
- `x2` : Nx2 data matrix, first column represents survival time of the i-th subject, second column represents censored flag (0 if not censored, 1 if censored)
- `pos` : The position of the legend. Can be 0 or 1. The legend will be on the topright if set to 0. The legend will be on the bottomleft if set to 1. Default is 0.

Value
A combined Kaplan Meier curve

Examples
```r
t1 <- c(2,3,4,5,5,7,10,12,15)
c1 <- c(0,0,1,0,0,1,0,0)
t2 <- c(1,3,5,4,8,10,9,11)
c2 <- c(0,0,0,1,0,0,0)
x1<-cbind(t1,c1)
x2<-cbind(t2,c2)
km_combine(x1,x2)
km_combine(x1,x2,pos=1)
```
km_log

Plot a Kaplan Meier curve in log scale

Description
The function plots a Kaplan Meier curve in log scale

Usage
km_log(time, censor, plotcens)

Arguments
- time: time of observed event
- censor: a vector indicating censored or not at the given times, 0 = censored; 1 = uncensored
- plotcens: 0: add censored data to the output curve
  1: don’t add censored data to the output curve

Value
A Kaplan Meier curve in log scale

Examples
```R
t1 <- c(2,3,4,5.5,7,10,12,15)
c1 <- c(0,0,1,0,0,1,0,0)
km_log(t1,c1,0)
```

km_red

Plot a Kaplan Meier curve in red

Description
The function plots a Kaplan Meier curve in red

Usage
km_red(time, censor, plotcens)

Arguments
- time: time of observed event
- censor: a vector indicating censored or not at the given times, 0 = censored; 1 = uncensored
- plotcens: 0: add censored data to the output curve
  1: don’t add censored data to the output curve
km_redsolid

Value

A red Kaplan Meier curve

Examples

t1 <- c(2, 3, 4, 5, 5, 7, 10, 12, 15)
c1 <- c(0, 0, 1, 0, 0, 1, 0, 0)
km_red(t1, c1, 0)

km_redsolid

Plot a Kaplan Meier curve in red solid line

Description

The function plots a Kaplan Meier curve in red solid line

Usage

km_redsolid(time, censor, plotcens)

Arguments

time time of observed event
censor a vector indicating censored or not at the given times, 0 = censored; 1 = uncensored
plotcens 0: add censored data to the output curve
1: don’t add censored data to the output curve

Value

A red solid Kaplan Meier curve

Examples

t1 <- c(2, 3, 4, 5, 5, 7, 10, 12, 15)
c1 <- c(0, 0, 1, 0, 0, 1, 0, 0)
km_redsolid(t1, c1, 0)
loopcuts

Change-point p-values with backward elimination

Description

A function that iterates to compute the p-values from the backward elimination procedure (Han et al. 2014)

Usage

loopcuts(time, censor, cuttimes, mono)

Arguments

time
  a sequence of time
censor
  a vector indicating censored or not at the given times, 0 = censored; 1 = uncensored
cuttimes
  unique, sorted, possible times to make the cuts, including 0 and the ending time
mono
  0: 2-sided hypothesis: H0: lam1 is equal to lam2; H1: lam1 is not equal to lam2
  1: 1-sided hypothesis: H0: lam1 is greater than or equal to lam2; H1: lam1 is less than lam2
  2: 1-sided hypothesis: H0: lam1 is less than or equal to lam2; H1: lam1 is greater than lam2

Value

the times in the backward elimination procedure and the corresponding p-values for each change-point in the iteration

Examples

data(loopcuts_t_c)
data(loopcuts_cut)
time = loopcuts_t_c[,1]
censor = loopcuts_t_c[,2]
loopcuts(time, censor, loopcuts_cut, 1)

loopcuts_cut

Example data for loopcuts_cuttimes

Description

Example data for loopcuts_cuttimes

Usage

data(loopcuts_cut)
loopcuts_onestep  

Description
This function computes the p-values at the current time points in input "time"

Usage
loopcuts_onestep(time, censor, cuttimes, mono)

Arguments

- **time**: a sequence of time
- **censor**: a vector indicating censored or not at the given times, 0 = censored; 1 = uncensored
- **cuttimes**: unique, sorted, possible times to make the cuts, including 0 and the ending time
- **mono**: 0: 2-sided hypothesis: H0: lam1 is equal to lam2; H1: lam1 is not equal to lam2
  1: 1-sided hypothesis: H0: lam1 is greater than or equal to lam2; H1: lam1 is less than lam2
  2: 1-sided hypothesis: H0: lam1 is less than or equal to lam2; H1: lam1 is greater than lam2

Value
P-values at for all time points in "time"

Examples

```r
data(loopcuts_t_c)
time = loopcuts_t_c[,1]
censor = loopcuts_t_c[,2]
loopcuts_onestep(time, censor, 28.03013699, 1)
```

loopcuts_t_c  

Example data for loopcut_times_censoring

Description
Example data for loopcut_times_censoring

Usage
data(loopcuts_t_c)
loopcuts_umbrella 

Change-point p-values with backward elimination under umbrella alternative order restriction

Description

A function that iterates to compute the p-values from the backward elimination procedure (Han et al. 2014) with umbrella alternative order restriction.

Usage

loopcuts_umbrella(time, censor, cuttimes, mono)

Arguments

time a sequence of time
censor a vector indicating censored or not at the given times, 0 = censored; 1 = uncensored
cuttimes unique, sorted, possible times to make the cuts, including 0 and the ending time
mono 0: 2-sided hypothesis: H0: lam1 is equal to lam2; H1: lam1 is not equal to lam2
1: 1-sided hypothesis: H0: lam1 is greater than or equal to lam2; H1: lam1 is less than lam2
2: 1-sided hypothesis: H0: lam1 is less than or equal to lam2; H1: lam1 is greater than lam2

Value

the times in the backward elimination procedure and the corresponding p-values for each change-point in the iteration

Examples

data(loopcuts_t_c)
data(loopcuts_umbrella_cuttimes_mono)
time = loopcuts_t_c[,1]
censor = loopcuts_t_c[,2]
cuttimes = loopcuts_umbrella_cuttimes_mono[,1]
mono = loopcuts_umbrella_cuttimes_mono[,2]
loopcuts_umbrella(time, censor, cuttimes, mono)
loopcuts_umbrella_cuttimes_mono

Example data for loopcut_umbrella

Description
Example data for loopcut_umbrella

Usage
data(loopcuts_umbrella_cuttimes_mono)

loopcut_onestep_data  Example data for loopcut_onestep

Description
Example data for loopcut_onestep

Usage
data(loopcut_onestep_data)

pava_dfr  PAVA order restriction under decreasing failure rate (DFR)

Description
This function imposes the PAVA DFR order restriction by eliminating change-points violating the restriction

Usage
pava_dfr(time_die,ttot,deaths)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_die</td>
<td>event times</td>
</tr>
<tr>
<td>ttot</td>
<td>the total time on test (ttot) corresponding to the event times</td>
</tr>
<tr>
<td>deaths</td>
<td>the number of deaths at each event time</td>
</tr>
</tbody>
</table>
Value
time2: the event times after PAVA
ttot2: the corresponding ttot
deaths2: the corresponding number of deaths

Examples
data(pava_dfrd)
t_d = pava_dfrd[,1]
t = pava_dfrd[,2]
d = pava_dfrd[,3]
pava_dfr(t_d, t, d)

Description
Example data for pava

Usage
data(pava_dfrd)

pava_ifr
PAVA order restriction under increasing failure rate (IFR)

Description
This function imposes the PAVA IFR order restriction by eliminating change-points violating the restriction

Usage
pava_ifr(time_die,ttot,deaths)

Arguments
time_die event times
ttot the total time on test (ttot) corresponding to the event times
deaths the number of deaths at each event time

Value
time2 the event times after PAVA
ttot2 the corresponding ttot after PAVA
deaths2 the corresponding number of deaths after PAVA
Examples

data(pava_dfrd)
t_d = pava_dfrd[,1]
t = pava_dfrd[,2]
d = pava_dfrd[,3]
pava_ifr(t_d, t, d)

\[ pexeest \]

RPEXE estimate given change-points

Description

This function estimates the survival probability at \( tx \) when a piecewise exponential distribution is fitted to \((times,cens)\) \( cens = 0 \) for censored, \( cens = 1 \) for uncensored. the change point is \( tchange \) and \( lamest \) is the estimated parameters

Usage

\texttt{pexeest(times, cens, tchange, tx)}

Arguments

- \texttt{times} All the event/censoring times used to fit the model
- \texttt{cens} censoring status used to fit the model
- \texttt{tchange} Change-points
- \texttt{tx} Time points to estimate the survival probability

Value

\texttt{lamest Lambda estimates for time periods divided by the change-points}

Examples

data(pexeest_times_censoring)
data(t100)
times = pexeest_times_censoring[,1]
cens = pexeest_times_censoring[,2]
pexeest(times, cens, 28.03014, t100)
Example data for pexeest_times_censoring

Description

Example data for pexeest_times_censoring

Usage

data(pexeest_times_censoring)

Reduced Piecewise Exponential Estimate/Test Software

Description

This reduced piecewise exponential survival software implements the likelihood ratio test and backward elimination procedure in Han, Schell, and Kim (2012, 2014), and Han et al. (2016). Inputs to the program can be either times when events/censoring occur or the vectors of total time on test and the number of events. Outputs of the programs are times and the corresponding p-values in the backward elimination. Details about the model and implementation are given in Han et al. 2014. This program can run in R version 3.2.2 and above.

References


**RPEXE main function**

**Description**

This is the RPEXE main function taking inputs including time, censoring, change-point candidates, order restriction, critical value, and display position. This function produces the RPEXE estimate. The prediction of the survival probability will be made on 100 equally spaced time points within the range of the event times based on the piecewise exponential estimate determined by all the change points.

**Usage**

RPEXEv1_2(times, censoring, cuttimes = NULL, monotone = 0, criticalp = -1, pos = 0)

**Arguments**

times A sequence of times where the events occur
censoring A sequence of dichotomous values indicating censored or not (0=censored and 1=not censored)
cuttimes A vector of unique, sorted, possible times to make the cuts. When it’s set to NULL, it’s the Default value, which is sorted event times from small to large.
monotone An input having indicating the monotonicity assumption – 0: no monotonic assumption (default) – 1: failure rate is decreasing over time – 2: failure rate is increasing over time – 3: monotonic failure rate – 4: failure rate is increasing and then decreasing – 5: failure rate is decreasing and then increasing – 6: failure rate is increasing and then decreasing with the peak removed first – 7: failure rate is decreasing and then increasing with the peak removed first
criticalp The critical (naive) p-value cutoff where all p-values in the backward elimination that are lower than this will be regarded as being significant. For example, at type I error rate 0.05, the critical p-value was 0.004 in the real example of Han et al. (2014). Default == -1 (equivalent to NA).
pos The position of the legend. Can be 0 or 1. The legend will be on the topright if set to 0. The legend will be on the bottomleft if set to 1. Default is 0.

**Value**

times: event/censoring times taking out from the backward elimination pvalues: p-values corresponding to "times" times_c: significant change-points pvalues_c: critical p-values that are smaller than the critical p-value trend: trend information struct: structure information for multiple order restrictions changet: change-point time of trend for umbrella alternatives.
Examples

t1 <- c(2,3,4,5,5,7,10,12,15)
c1 <- c(0,0,1,0,0,1,0,0)
RPEXEv1_2(t1, c1, monotone = 1, criticalp = 0.05, pos = 0)

---

t100

Example data for pexeest_tx

---

Description

Example data for pexeest_tx

Usage

data(t100)
**totaltest**  
*total time on test*

**Description**

Function `totaltest` computes total-time-on-test.

**Usage**

`totaltest(time,censor)`

**Arguments**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>event/censoring times</td>
</tr>
<tr>
<td>censor</td>
<td>censoring status</td>
</tr>
</tbody>
</table>

**Value**

time_die time points where events occur (in ascending order)  
ttot total time on test corresponding to each time point in "time_die"  
deaths number of death corresponding to each time point in "time_die"

**Examples**

```r
t1 <- c(2,3,4,5.5,7,10,12,15)  
c1 <- c(0,0,1,0,0,1,0,0)  
totaltest(t1,c1)
```

---

**umbrella**  
*Umbrella alternative.*

**Description**

Using the umbrella alternative to merge certain entries to make the sequence of ttot/deaths to increase then decrease or to decrease then increase. Note that the pava function imposes non-decreasing or non-increasing order. This function directly uses function `pava()`.

**Usage**

`umbrella(time_die,ttot,deaths,indi)`

**Arguments**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_die</td>
<td>a sequence of times where deaths happened.</td>
</tr>
<tr>
<td>ttot</td>
<td>the total time on test between each time point and the previous time point (or 0).</td>
</tr>
<tr>
<td>deaths</td>
<td>the number of deaths at each time point.</td>
</tr>
<tr>
<td>indi</td>
<td>an indicator indi == 0: monotonic failure rate (either decrease or increase) indi == 1: denoting the failure rate increase then decrease indi == 2: denoting the failure rate decrease then increase</td>
</tr>
</tbody>
</table>
Value

time2 == the merged time_die after the umbrella alternative order restriction; struct == a structure saves the partition information; label == a note about how the failure rate varies; indx == the position where the change point value is.

Examples

data(pava_dfrd)
t_d = pava_dfrd[,1]
t = pava_dfrd[,2]
d = pava_dfrd[,3]
umbrella(t_d, t, d, 2)
Index

* datasets
  - data2, 3
  - df, 4
  - loopcut_onestep_data, 14
  - loopcuts_cut, 11
  - loopcuts_t_c, 12
  - loopcuts_umbrella_cuttimes_mono, 14
  - pava_dfrd, 15
  - pexeest_times_censoring, 17
  - simple, 19
  - t100, 19
  - bisec, 2
  - data2, 3
  - df, 4
  - exact_pvalue, 4
  - gamllik, 5
  - km, 6
  - km_blacksolid, 7
  - km_combine, 8
  - km_log, 9
  - km_red, 9
  - km_redsolid, 10
  - kmvalue, 7
  - loopcut_onestep_data, 14
  - loopcuts, 11
  - loopcuts_cut, 11
  - loopcuts_onestep, 12
  - loopcuts_t_c, 12
  - loopcuts_umbrella, 13
  - loopcuts_umbrella_cuttimes_mono, 14
  - pava_dfr, 14
  - pava_dfrd, 15
  - pava_ifr, 15

  - pexeest, 16
  - pexeest_times_censoring, 17
  - RPEXE.RPEXT, 17
  - RPEXEv1_2, 18
  - simple, 19
  - t100, 19
  - totaltest, 20
  - umbrella, 20