

Package ‘nph’

August 23, 2019

Title Planning and Analysing Survival Studies under Non-Proportional Hazards

Version 1.9

Description

Piecewise constant hazard functions are used to flexibly model survival distributions with non-proportional hazards and to simulate data from the specified distributions. Also, a function to calculate weighted log-rank tests for the comparison of two hazard functions is included. Finally, a function to calculate a test using the maximum of a set of test statistics from weighted log-rank tests is provided. This test utilizes the asymptotic multivariate normal joint distribution of the separate test statistics. The correlation is estimated from the data.

Date 2019-08-21

Maintainer Robin Ristl <robin.ristl@meduniwien.ac.at>

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Depends R (>= 3.3.0)

Imports stats, graphics, mvtnorm, ggplot2

Suggests knitr, shiny, shinycssloaders, formatR, styler, rmarkdown

VignetteBuilder knitr

Author Robin Ristl [aut, cre],
Nicolas Ballarini [ctb]

Repository CRAN

Date/Publication 2019-08-23 10:50:02 UTC

R topics documented:

logrank.maxtest	2
logrank.test	4
m2r	6
nph_gui	6
pchaz	7
plot.mixpch	8
plot_diagram	9
plot_shhr	10
plot_subgroups	10
pop_pchaz	11
rSurv_conditional_fun	13
rSurv_fun	14
sample_conditional_fun	15
sample_fun	17
subpop_pchaz	18

Index	21
--------------	-----------

logrank.maxtest	<i>Maximum log-rank test</i>
-----------------	------------------------------

Description

Calculates a test for the comparison of two groups based on the maximum of test statistics of a set of weighted log-rank tests

Usage

```
logrank.maxtest(time, event, group, alternative = c("two.sided", "less",
"greater"), rho = c(0, 0, 1), gamma = c(0, 1, 0), weights = NULL)
```

Arguments

time	Vector of observed event/censored times
event	logical vector indicating if an event was observed (TRUE) or the time is censored (FALSE)
group	Vector of group allocations
alternative	Either of "two.sided", "less" or "greater", specifies if two-sided or respective one-sided p-values are calculated. In any case the z test statistic of each included weighted log-rank test is based on the (weighted) sum of expected minus observed events in the group corresponding to the first factor level of group. Hence a small value of the test statistic corresponds to a lower (weighted average) hazard rate in the first group.
rho	Vector of parameter values rho for a set of weighting functions in the rho-gamma family

gamma	Vector of parameter values gamma for a set of weighting functions in the rho-gamma family
weights	Optional matrix, each column containing a different weighting vector for the data

Details

To perform a maximum-type combination test, a set of m different weight functions $w_1(t), \dots, w_m(t)$ is specified and the correspondingly weighted logrank statistics z_1, \dots, z_m are calculated. The maximum test statistic is $z_{max} = \max_{i=1, \dots, m} z_i$. If at least one of the selected weight functions results in high power, we may expect a large value of z_{max} . Under the least favorable configuration in H_0 , approximately $(Z_1, \dots, Z_m) \sim N_m(0, \Sigma)$. The p-value of the maximum test, $P_{H_0}(Z_{max} > z_{max}) = 1 - P(Z_1 \leq z_{max}, \dots, Z_m \leq z_{max})$, is calculated based on this multivariate normal approximation via numeric integration.

This approach automatically corrects for multiple testing with different weights and does so efficiently since the correlation between the different tests is incorporated in Σ . For actual calculations, Σ is replaced by an estimate. Note that $cov(w_i(t)d_{t,ctr}, w_j(t)d_{t,ctr}) = w_i(t)w_j(t)var(d_{t,ctr})$, at least approximately assuming weights are converging in probability to a non-random function. Thus the i, j -the element of Σ is estimated as

$$\hat{cov}(Z_i, Z_j) = \sum_{t \in \mathcal{D}} w_i(t)w_j(t)var(d_{t,ctr}) / \sqrt{\sum_{t \in \mathcal{D}} w_i^2(t)var(d_{t,ctr}) \sum_{t \in \mathcal{D}} w_j^2(t)var(d_{t,ctr})}$$

Value

A list with elements:

`pmult` The two sided p-value for the null hypothesis of equal hazard functions in both groups, based on the multivariate normal approximation for the z-statistics of differently weighted log-rank tests.

`p.Bonf` The two sided p-value for the null hypothesis of equal hazard functions in both groups, based on a Bonferroni multiplicity adjustment for differently weighted log-rank tests.

`tests` Data frame with z-statistics and two-sided unadjusted p-values of the individual weighted log-rank tests

`korr` Estimated correlation matrix for the z-statistics of the differently weighted log-rank tests.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

See Also

[logrank.test](#)

Examples

```

A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
B <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.1, 0.6, 0.1), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.04, 0.04), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
dat <- sample_fun(A, B, r0 = 0.5, eventEnd = 30,
  lambdaRecr = 0.5, lambdaCens = 0.25 / 365,
  maxRecrCalendarTime = 2 * 365,
  maxCalendar = 4 * 365)
logrank.maxtest(dat$y, dat$event, dat$group)

```

logrank.test

Weighted log-rank test

Description

Calculates a weighted log-rank test for the comparison of two groups.

Usage

```

logrank.test(time, event, group, alternative = c("two.sided", "less",
  "greater"), rho = 0, gamma = 0, weights = NULL)

```

Arguments

time	Vector of observed event/censored times
event	logical vector indicating if an event was observed (TRUE) or the time is censored (FALSE)
group	Vector of group allocations
alternative	Either of "two.sided", "less" or "greater", specifies if two-sided or respective one-sided p-values are calculated. In any case the z test statistic of each included weighted log-rank test is based on the (weighted) sum of expected minus observed events in the group corresponding to the first factor level of group. Hence a small value of the test statistic corresponds to a lower (weighted average) hazard rate in the first group.
rho	Parameter to calculate weights in the rho-gamma family
gamma	Parameter to calculate weights in the rho-gamma family
weights	Optional vector of weights

Details

For a given sample, let \mathcal{D} be the set of unique event times. For a time-point $t \in \mathcal{D}$, let $n_{t,ctr}$ and $n_{t,trt}$ be the number of patients at risk in the control and treatment group and let $d_{t,ctr}$ and $d_{t,trt}$ be the respective number of events. The expected number of events in the control group is calculated under the least favorable configuration in H_0 , $\lambda_{ctr}(t) = \lambda_{trt}(t)$, as $e_{t,ctr} = (d_{t,ctr} + d_{t,trt}) \frac{n_{t0}}{n_{t0} + n_{t1}}$. The conditional variance of $d_{t,ctr}$ is calculated from a hypergeometric distribution as $var(d_{t,ctr}) = \frac{n_{t0}n_{t1}(d_{t0} + d_{t1})(n_{t0} + n_{t1} - d_{t0} - d_{t1})}{(n_{t0} + n_{t1})^2(n_{t0} + n_{t1} - 1)}$. Further define a weighting function $w(t)$. The weighted logrank test statistic for a comparison of two groups is

$$z = \sum_{t \in \mathcal{D}} w(t)(d_{t,ctr} - e_{t,ctr}) / \sqrt{\sum_{t \in \mathcal{D}} w(t)^2 var(d_{t,ctr})}$$

Under the the least favorable configuration in H_0 , the test statistic is asymptotically standard normally distributed and large values of z are in favor of the alternative.

The function consider particular weights in the Fleming-Harrington $\rho - \gamma$ family $w(t) = \hat{S}(t)^\rho(1 - \hat{S}(t))^\gamma$. Here, $\hat{S}(t) = \prod_{s \in \mathcal{D}: s \leq t} 1 - \frac{d_{t,ctr} + d_{t,trt}}{n_{t,ctr} + n_{t,trt}}$ is the pooled sample Kaplan-Meier estimator. Weights $\rho = 0, \gamma = 0$ correspond to the standard logrank test with constant weights $w(t) = 1$. Choosing $\rho = 0, \gamma = 1$ puts more weight on late events, $\rho = 1, \gamma = 0$ puts more weight on early events and $\rho = 1, \gamma = 1$ puts most weight on events at intermediate time points.

Value

A list with elements:

`D` A data frame event numbers, numbers at risk and expected number of events for each event time

`test` A data frame containing the z and chi-squared statistic for the one-sided and two-sided test, respectively, of the null hypothesis of equal hazard functions in both groups and the p-value for the one-sided test.

`var` The estimated variance of the sum of expected minus observed events in the first group.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

References

Thomas R Fleming and David P Harrington. Counting processes and survival analysis, volume 169. John Wiley & Sons, 2011

See Also

[logrank.maxtest](#)

Examples

```

A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
B <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.1, 0.6, 0.1), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.04, 0.04), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
dat <- sample_fun(A, B, r0 = 0.5, eventEnd = 30,
  lambdaRecr = 0.5, lambdaCens = 0.25 / 365,
  maxRecrCalendarTime = 2 * 365,
  maxCalendar = 4 * 365)
logrank.test(dat$y, dat$event, dat$group)

```

m2r

*Transform median time into rate***Description**

This helper function calculates the hazard rate per day of an exponential distribution from the median given in months.

Usage

```
m2r(x)
```

Arguments

x The median time in months to be transformed into rate

nph_gui

*Launch a GUI (shiny app) for the nph package***Description**

Launch a GUI (shiny app) for the nph package

Usage

```
nph_gui()
```

Details

The packages shinycssloaders, formatR and styler are required for correct display of the GUI. The package rmarkdown with access to pandoc is required to save reports.

pchaz

*Calculate survival for piecewise constant hazard***Description**

Calculates hazard, cumulative hazard, survival and distribution function based on hazards that are constant over pre-specified time-intervals.

Usage

```
pchaz(Tint, lambda)
```

Arguments

Tint vector of length $k + 1$, for the boundaries of k time intervals (presumably in days) with piecewise constant hazard. The boundaries should be increasing and the first one should be 0, the last one should be larger than the assumed trial duration.

lambda vector of length k with the piecewise constant hazards for the intervals specified via **Tint**.

Details

Given k time intervals $[t_{j-1}, t_j), j = 1, \dots, k$ with $0 = t_0 < t_1 \dots < t_k$, the function assume constant hazards λ_j at each interval. The resulting hazard function is $\lambda(t) = \sum_{j=1}^k \lambda_j 1_{t \in [t_{j-1}, t_j)}$, the cumulative hazard function is $\Lambda(t) = \int_0^t \lambda(s) ds = \sum_{j=1}^k ((t_j - t_{j-1}) \lambda_j 1_{t > t_j} + (t - t_{j-1}) \lambda_j 1_{t \in [t_{j-1}, t_j)})$ and the survival function $S(t) = e^{-\Lambda(t)}$. The output includes the functions values calculated for all integer time points between 0 and the maximum of **Tint**. Additionally, a list with functions is also given to calculate the values at any arbitrary point t .

Value

A list with class `mixpch` containing the following components:

haz Values of the hazard function over discrete times **t**.

cumhaz Values of the cumulative hazard function over discrete times **t**.

S Values of the survival function over discrete times **t**.

F Values of the distribution function over discrete times **t**.

t Time points for which the values of the different functions are calculated.

Tint Input vector of boundaries of time intervals.

lambda Input vector of piecewise constant hazards.

funs A list with functions to calculate the hazard, cumulative hazard, survival, pdf and cdf over arbitrary continuous times.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>, Nicolas Ballarini

See Also

[subpop_pchaz](#), [pop_pchaz](#), [plot.mixpch](#)

Examples

```
pchaz(Tint = c(0, 40, 100), lambda=c(.02, .05))
```

plot.mixpch

Plot mixpch Objects

Description

Plots survival and other functions stored in mixpch objects versus time.

Usage

```
## S3 method for class 'mixpch'
plot(x, fun = c("S", "F", "haz", "cumhaz"),
     add = FALSE, ylab = fun, xlab = "Time", ...)
```

Arguments

x	an object of class mixpch.
fun	character string in c("S", "F", "haz", "cumhaz") indicating which function to plot. Select "S" for the survival function, "F" for the distribution function, "haz" for the hazard function or "cumhaz" for the cumulative hazard function.
add	logical, indicates if the drawing should be added to an existing plot.
ylab	label of the y-axis
xlab	label of the x-axis
...	further arguments passed to the plotting functions

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

See Also

[pchaz](#), [subpop_pchaz](#), [pop_pchaz](#)

Examples

```
A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
plot(A)
plot(A, "haz", add = TRUE)
```

plot_diagram	<i>Draw a state space figure</i>
--------------	----------------------------------

Description

A figure that shows the states and the possible transitions between them.

Usage

```
plot_diagram(A, B, A_subgr_labels = "", B_subgr_labels = "",
  which = c("Both", "Experimental", "Control"),
  treatment_labels = c("Experimental", "Control"), colors = "default",
  show.rate = TRUE)
```

Arguments

A	An object of class mixpch, resembling the survival function in treatment group 0
B	An object of class mixpch, resembling the survival function in treatment group 1
A_subgr_labels	A character vector with the same length as A\$p. It indicates names for the subgroups in A
B_subgr_labels	A character vector with the same length as B\$p. It indicates names for the subgroups in B
which	Which treatment arm should be shown? One of "Both", "Experimental", "Control".
treatment_labels	A character vector of length 2 indicating the treatment labels.
colors	Either a vector of length two with colors for A and B, or "default".
show.rate	A logical indicating whether the rate should be shown in the diagram

plot_shhr	<i>Plot of survival, hazard and hazard ratio of two groups as a function of time</i>
-----------	--

Description

A convenience function that uses the generic plot function in the `nph` package to plot the three functions in a layout of 3 columns and 1 row.

Usage

```
plot_shhr(A, B, main = "", xmax = NULL, ymax_haz = NULL,
          ymax_hr = NULL)
```

Arguments

A	An object of class <code>mixpch</code> , resembling the survival function in treatment group 0
B	An object of class <code>mixpch</code> , resembling the survival function in treatment group 1
main	An overall title for the plot
xmax	A maximum value for the x-axis. The plot is drawn using <code>xlim = c(0, xmax)</code>
ymax_haz	A maximum value for the y-axis for the hazards plot. The plot is drawn using <code>ylim = c(0, ymax_haz)</code>
ymax_hr	A maximum value for the y-axis for the hazards ratio plot. The plot is drawn using <code>ylim = c(0, ymax_hr)</code>

plot_subgroups	<i>Draw a population composition plot</i>
----------------	---

Description

A figure that shows the composition of the population under study though time

Usage

```
plot_subgroups(A, B, colors = "default", max_time = max(A$Tint),
              position = c("stack", "fill"), title = "")
```

Arguments

A	An object of class <code>mixpch</code> , resembling the survival function in treatment group 0
B	An object of class <code>mixpch</code> , resembling the survival function in treatment group 1
colors	Either a vector of length four with colors for A and B and subgroup 1 and 2, or "default".
max_time	the maximum value for the x-axis.
position	Either "stack" or "fill". By default (stack), the total population decreases through time. If position="fill", the size of the population is rescaled to show conditional percentages.
title	The text for the title.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>, Nicolas Ballarini

See Also

[pop_pchaz](#)

Examples

```
A <- pop_pchaz(Tint = c(0, 90, 365),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
B <- pop_pchaz(Tint = c(0, 90, 365),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.1, 0.6, 0.1), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.04, 0.04), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
plot_subgroups(A, B, title = "position='stack'")
plot_subgroups(A, B, position='fill', title = "position='fill'")
```

pop_pchaz

Calculate survival for piecewise constant hazards with change after random time and mixture of subpopulations

Description

Calculates hazard, cumulative hazard, survival and distribution function based on hazards that are constant over pre-specified time-intervals

Usage

```
pop_pchaz(Tint, lambdaMat1, lambdaMat2, lambdaProgMat, p,
  timezero = FALSE, int_control = list(rel.tol =
    .Machine$double.eps^0.4, abs.tol = 1e-09),
  discrete_approximation = FALSE)
```

Arguments

Tint	vector of length $k + 1$, for the boundaries of k time intervals (presumably in days) with piecewise constant hazard. The boundaries should be increasing and the first one should be \emptyset , the last one should be larger than the assumed trial duration.
lambdaMat1	matrix of dimension m -by- k , each row contains the vector of piecewise constant hazards for one subpopulation before the changing event happens, for the intervals specified via Tint.
lambdaMat2	matrix of dimension m -by- k , each row contains the vector piecewise constant hazards for one subpopulation after the changing event has happened, for the intervals specified via Tint.
lambdaProgMat	matrix of dimension m -by- k , each row contains the vector of piecewise constant hazards for one subpopulation for the changing event, for the intervals specified via Tint.
p	vector of length m for relative sizes (proportions) of the subpopulations. They should sum up to 1.
timezero	logical, indicating whether after the changing event the timecount, governing which interval in Tint and which according value in lambda2 is used, should restart at zero. This argument is either of length 1 (applying the same to all subgroups) or the same length as the number of subgroups.
int_control	A list with additional parameters to be passed to the integrate function.
discrete_approximation	if TRUE, the function uses an approximation based on discretizing the time, instead of integrating. This speeds up the calculations

Details

Given m subgroups with relative sizes p_1, \dots, p_m and subgroup-specific survival functions $S_l(t)$, the marginal survival function is the mixture $S(t) = \sum_{l=1}^m p_l S_l(t)$. Note that the respective hazard function is not a linear combination of the subgroup-specific hazard functions. It may be calculated by the general relation $\lambda(t) = -\frac{dS(t)}{dt} \frac{1}{S(t)}$. In each subgroup, the hazard is modelled as a piecewise constant hazard, with the possibility to also model disease progression. Therefore, each row of the hazard rates is used in [subpop_pchaz](#). See [pchaz](#) and [subpop_pchaz](#) for more details. The output includes the function values calculated for all integer time points between 0 and the maximum of Tint.

Note: this function may be very slow in cases where many time points need to be calculated. If this happens, use `discrete_approximation = TRUE`.

Value

A list with class `mixpch` containing the following components:

`haz` Values of the hazard function.

`cumhaz` Values of the cumulative hazard function.

`S` Values of the survival function.

`F` Values of the distribution function.

`t` Time points for which the values of the different functions are calculated.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>, Nicolas Ballarini

See Also

[pchaz](#), [subpop_pchaz](#), [plot.mixpch](#)

Examples

```
pop_pchaz(Tint = c(0, 40, 100),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2),
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2),
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2),
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
```

`rSurv_conditional_fun` *Draw conditional random survival times from mixpch object.*

Description

Draws independent random survival times from `mixpch` objects conditional on observed time.

Usage

```
rSurv_conditional_fun(x, y)
```

Arguments

<code>x</code>	An object of class <code>mixpch</code>
<code>y</code>	A vector of observed right censored times

Details

Note that the `mixpch` object stores the survival function up to some time `T`. For random times equal or larger `T`, the value `T` is returned.

Value

A vector of random survival times, conditional on the observed censored times.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

See Also

[rSurv_fun](#), [sample_fun](#), [sample_conditional_fun](#)

Examples

```
A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
rSurv_conditional_fun(x = A, y = c(10,15,9,2,1))
```

rSurv_fun

Draw random survival times from mixpch object.

Description

Draws independent random survival times from mixpch objects.

Usage

```
rSurv_fun(n, x)
```

Arguments

n	Number of random draws
x	An object of class mixpch

Details

The mixpch object stores the survival function up to some time T. For random times equal or larger T, the value T is returned.

Value

A vector of random survival times.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

See Also

[rSurv_conditional_fun](#), [sample_fun](#), [sample_conditional_fun](#)

Examples

```
A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
rSurv_fun(n = 10, x = A)
```

sample_conditional_fun

Draw conditional survival times based on study settings

Description

Simulates data for a randomized controlled survival study conditional on observed interim data.

Usage

```
sample_conditional_fun(dat, A, B, r0 = 0.5, eventEnd, lambdaRecr,
  lambdaCens, maxRecrCalendarTime, maxCalendar)
```

Arguments

dat	A data frame with the same structure and column names as the output of sample_fun , containing the data to condition on
A	An object of class <code>mixpch</code> , resembling the survival function in treatment group 0
B	An object of class <code>mixpch</code> , resembling the survival function in treatment group 1
r0	Allocation ratio to group 1 (must be a number between 0 and 1)
eventEnd	Number of events, after which the study stops
lambdaRecr	Rate per day for recruiting patients, assuming recruiting follows a Poisson process
lambdaCens	Rate per day for random censoring, assuming censoring times are exponential
maxRecrCalendarTime	Maximal duration of recruitment in days
maxCalendar	Maximal total study duration in days, after which the study stops

Details

For simulating the data, patients are allocated randomly to either group (unrestricted randomization).

Value

A data frame with each line representing data for one patient and the following columns:

group Treatment group

inclusion Start of observation in terms of calendar time

y Observed survival/censored time

yCalendar End of observation in terms of calendar time.

event logical, TRUE indicates the observation ended with an event, FALSE corresponds to censored times

adminCens logical, True indicates that the observation is subject to administrative censoring, i.e. the subject was observed until the end of the study without an event.

cumEvents Cumulative number of events over calendar time of end of observation

The data frame is ordered by yCalendar

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

See Also

[rSurv_fun](#), [rSurv_conditional_fun](#), [sample_fun](#)

Examples

```
A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
B <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.1, 0.6, 0.1), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.04, 0.04), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
datinterim <- sample_fun(A, B, r0 = 0.5, eventEnd = 30, lambdaRecr = 1,
  lambdaCens = 0.25 / 365,
  maxRecrCalendarTime = 3 * 365,
  maxCalendar = 4 * 365)
datcond <- sample_conditional_fun(datinterim, A, B, r0 = 0.5, eventEnd = 60,
  lambdaRecr = 1, lambdaCens = 0.25 / 365, maxRecrCalendarTime = 3 * 365,
  maxCalendar = 4 * 365)
```

sample_fun	<i>Draw survival times based on study settings</i>
------------	--

Description

Simulates data for a randomized controlled survival study.

Usage

```
sample_fun(A, B, r0 = 0.5, eventEnd, lambdaRecr, lambdaCens,
           maxRecrCalendarTime, maxCalendar)
```

Arguments

A	An object of class <code>mixpch</code> , resembling the survival function in treatment group 0
B	An object of class <code>mixpch</code> , resembling the survival function in treatment group 1
r_0	Allocation ratio to group 0 (must be a number between 0 and 1)
eventEnd	Number of events, after which the study stops
lambdaRecr	Rate per day for recruiting patients, assuming recruiting follows a Poisson process
lambdaCens	Rate per day for random censoring, assuming censoring times are exponential
maxRecrCalendarTime	Maximal duration of recruitment in days
maxCalendar	Maximal total study duration in days, after which the study stops

Details

For simulating the data, patients are allocated randomly to either group (unrestricted randomization).

Value

A data frame with each line representing data for one patient and the following columns:

group	Treatment group
inclusion	Start of observation in terms of calendar time
y	Observed survival/censored time
yCalendar	End of observation in terms of calendar time.
event	logical, TRUE indicates the observation ended with an event, FALSE corresponds to censored times
adminCens	logical, True indicates that the observation is subject to administrative censoring, i.e. the subject was observed until the end of the study without an event.
cumEvents	Cumulative number of events over calendar time of end of observation

The data frame is ordered by `yCalendar`

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>

See Also

[rSurv_fun](#), [rSurv_conditional_fun](#), [sample_conditional_fun](#)

Examples

```
A <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.2, 0.6, 0.2), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.4, 0.4), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
B <- pop_pchaz(Tint = c(0, 90, 1500),
  lambdaMat1 = matrix(c(0.2, 0.1, 0.4, 0.1), 2, 2) / 365,
  lambdaMat2 = matrix(c(0.5, 0.1, 0.6, 0.1), 2, 2) / 365,
  lambdaProg = matrix(c(0.5, 0.5, 0.04, 0.04), 2, 2) / 365,
  p = c(0.8, 0.2),
  timezero = FALSE, discrete_approximation = TRUE)
plot(A)
plot(B, add = TRUE)
dat <- sample_fun(A, B, r0 = 0.5, eventEnd = 30, lambdaRecr = 0.5,
  lambdaCens = 0.25 / 365, maxRecrCalendarTime = 2 * 365,
  maxCalendar = 4 * 365)
```

subpop_pchaz

Calculate survival for piecewise constant hazards with change after random time

Description

Calculates hazard, cumulative hazard, survival and distribution function based on hazards that are constant over pre-specified time-intervals

Usage

```
subpop_pchaz(Tint, lambda1, lambda2, lambdaProg, timezero = FALSE,
  int_control = list(rel.tol = .Machine$double.eps^0.4, abs.tol = 1e-09),
  discrete_approximation = FALSE)
```

Arguments

Tint vector of length $k + 1$, for the boundaries of k time intervals (presumably in days) with piecewise constant hazard. The boundaries should be increasing and the first one should be 0, the last one should be larger than the assumed trial duration.

lambda1	vector of length k for piecewise constant hazards before the changing event happens, for the intervals specified via T.
lambda2	vector of length k for piecewise constant hazards after the changing event has happened, for the intervals specified via T.
lambdaProg	vector of length k for piecewise constant hazards for the changing event, for the intervals specified via T.
timezero	logical, indicating whether after the changeing event the timecount, governing which interval in Tint and which according value in lambda2 is used, should restart at zero.
int_control	A list with the rel.tol and abs.tol paramaters to be passed to the integrate function.
discrete_approximation	if TRUE, the function uses an approximation based on discretizing the time, instead of integrating. This speeds up the calculations

Details

We assume that the time to disease progression T_{PD} is governed by a separate process with hazard function $\eta(t)$, which does not depend on the hazard function for death $\lambda(t)$. $\eta(t)$, too, may be modelled as piecewise constant or, for simplicity, as constant over time. We define $\lambda_{prePD}(t)$ and $\lambda_{postPD}(t)$ as the hazard functions for death before and after disease progression. Conditional on $T_{PD} = s$, the hazard function for death is $\lambda(t|T_{PD} = s) = \lambda_{prePD}(t)I_{t \leq s} + \lambda_{postPD}(t)I_{t > s}$. The conditional survival function is $S(t|T_{PD} = s) = \exp(-\int_0^t \lambda(t|T_{PD} = s)ds)$. The unconditional survival function results from integration over all possible progression times as $S(t) = \int_0^t S(t|T_{PD} = s)dP(T_{PD} = s)$. The output includes the function values calculated for all integer time points between 0 and the maximum of Tint. Additionally, a list with functions is also given to calculate the values at any arbitrary point t .

Value

A list with class `mixpch` containing the following components:

haz Values of the hazard function.

cumhaz Values of the cumulative hazard function.

S Values of the survival function.

F Values of the distribution function.

t Time points for which the values of the different functions are calculated.

Tint Input vector of boundaries of time intervals.

lambda1 Input vector of piecewise constant hazards before the changing event happen.

lambda2 Input vector of piecewise constant hazards after the changing event happen.

lambdaProg Input vector of piecewise constant hazards for the changing event.

funs A list with functions to calculate the hazard, cumulative hazard, survival, and cdf over arbitrary continuous times.

Author(s)

Robin Ristl, <robin.ristl@meduniwien.ac.at>, Nicolas Ballarini

See Also

[pchaz](#), [pop_pchaz](#), [plot.mixpch](#)

Examples

```
subpop_pchaz(Tint = c(0, 40, 100), lambda1 = c(0.2, 0.4), lambda2 = c(0.1, 0.01),  
lambdaProg = c(0.5, 0.4), timezero = FALSE, discrete_approximation = TRUE)  
subpop_pchaz(Tint = c(0, 40, 100), lambda1 = c(0.2, 0.4), lambda2 = c(0.1, 0.01),  
lambdaProg = c(0.5, 0.4), timezero = TRUE, discrete_approximation = TRUE)
```

Index

integrate, [12](#), [19](#)

logrank.maxtest, [2](#), [5](#)

logrank.test, [3](#), [4](#)

m2r, [6](#)

nph_gui, [6](#)

pchaz, [7](#), [8](#), [12](#), [13](#), [20](#)

plot.mixpch, [8](#), [8](#), [13](#), [20](#)

plot_diagram, [9](#)

plot_shhr, [10](#)

plot_subgroups, [10](#)

pop_pchaz, [8](#), [11](#), [11](#), [20](#)

print.wlogrank (logrank.test), [4](#)

print.wlogrank_max (logrank.maxtest), [2](#)

rSurv_conditional_fun, [13](#), [15](#), [16](#), [18](#)

rSurv_fun, [14](#), [14](#), [16](#), [18](#)

sample_conditional_fun, [14](#), [15](#), [15](#), [18](#)

sample_fun, [14–16](#), [17](#)

subpop_pchaz, [8](#), [12](#), [13](#), [18](#)