Package 'TwoTimeScales'

July 22, 2025

Title Analysis of Event Data with Two Time Scales

Type Package

Version 1.0.0

URL https://github.com/AngelaCar/TwoTimeScales,
https://angelacar.github.io/TwoTimeScales/

Description Analyse time to event data with two time scales by estimating a smooth hazard that varies over two time scales and also, if covariates are available, to estimate a proportional hazards model with such a two-dimensional baseline hazard.
Functions are provided to prepare the raw data for estimation, to estimate and to plot the two-dimensional smooth hazard.
Extension to a competing risks model are implemented. For details about the method please refer to Carollo et al. (2024) <doi:10.1002/sim.10297>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports colorspace, Epi, fields, JOPS, LMMsolver, popEpi, reshape2, spam, ucminf, viridis

Depends R (>= 3.5.0)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

BugReports https://github.com/AngelaCar/TwoTimeScales/issues

NeedsCompilation no

Author Angela Carollo [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0002-5625-6553>),
Paul H.C. Eilers [aut],
Jutta Gampe [aut]

Maintainer Angela Carollo <carollo@demogr.mpg.de>

Repository CRAN

Date/Publication 2024-12-23 10:50:10 UTC

Contents

`covariates_plot` *Plot of the covariates' effects*

Description

`covariates_plot()` produces a plot of the covariates' effects ($\hat{\beta}$) with confidence intervals, or of the Hazard Ratios ($\exp(\hat{\beta})$) with confidence intervals.

Usage

```
covariates_plot(
  fitted_model,
  confidence_lev = 0.95,
  plot_options = list(),
  ...
)
```

Arguments

`fitted_model`	A list returned by the function `fit2ts` or `fit1ts`.
`confidence_lev`	The level of confidence for the CIs. Default is 0.95 ($\alpha = 0.05$).
`plot_options`	A list of options for the plot:

- `HR` A Boolean. If `TRUE` the HRs with their CIs will be plotted. Default is `FALSE` (plot the `beta` with their CIs).
- `symmetric_ci` A Boolean. Default is `TRUE`. If a plot of the HRs is required (`HR == TRUE`), then plot symmetrical Confidence Intervals, based on the SEs for the HRs calculated by delta method. If `FALSE`, then CIs are obtained by exponentiating the CIs for the betas.
- `main` The title of the plot.
- `ylab` The label for the y-axis.
- `ylim` A vector with two elements defining the limits for the y-axis.
- `col_beta` The color for the plot of the covariates' effects.
- `pch` The symbol for plotting the point estimates.
- `cex_main` The magnification factor for the main of the plot.

`...`	further arguments passed to plot()

Value

A plot of the covariates' effects. The different covariates are plotted on the x-axis, and on the y-axis the effects on the coefficient- or on the HR-scale are plotted. The main estimate is represented by a point and the CIs are added as vertical bars.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)
x1 <- c(0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0)

fakedata <- as.data.frame(cbind(id, u, s, ev, x1))
covs <- subset(fakedata, select = c("x1"))
fakedata2ts <- prepare_data(u = fakedata$u,
                            s_out = fakedata$s,
                            ev = fakedata$ev,
                            ds = .5,
                            individual = TRUE,
                            covs = covs)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1 ,1.5 ,.5),
                              seq(1 ,1.5 ,.5)))
# Covariates plot with default options
covariates_plot(fakemod)

# Plot the hazard ratios instead
covariates_plot(fakemod,
                plot_options = list(
                HR = TRUE))

# Change confidence level
covariates_plot(fakemod,
                confidence_lev = .99)
```

`cumhaz2ts` *Cumulative hazard over two time scales*

Description

Computes the cumulative hazard surface over two time scales from a fitted model. The function is also called internally from `plot()` if the user wants to plot the cumulative hazard from a fitted model.

Usage

```
cumhaz2ts(
  fitted_model,
  plot_grid = NULL,
```

```
  cause = NULL,
  midpoints = FALSE,
  where_slices = NULL,
  direction = c("u", "s", NULL),
  tmax = NULL
)
```

Arguments

fitted_model	(optional) The output of the function fit2ts. This is an object of class 'haz2ts' or 'haz2tsLMM'.
plot_grid	(optional) A list containing the parameters to build a new finer grid of intervals over u and s for plotting. This must be of the form: • plot_grid = list(c(umin, umax, du), c(smin, smax, ds)) where umin, umax and smin, smax are the minimum and maximum values desired for the grid-points over u and s respectively, and du, ds are distances between two adjacent points over u and s respectively. Specifying a new denser grid is used to evaluate the B-spline bases used for estimation on such grid and plot the estimated surfaces with a greater level of detail. If not specified, the plotting is done using the same B-splines bases as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.
cause	a character string with a short name for the cause (optional).
midpoints	A Boolean. Default is FALSE. If TRUE, the estimated quantities are evaluated at the midpoints of the rectangles (or parallelograms) of the grids, rather than at each grid-point.
where_slices	A vector of values for the cutting points of the desired slices of the surface. This option is included mostly for the plotting function. When using plot.haz2ts(), the user selects which_plot = "cumhaz" and cumhaz_slices = TRUE, then where_slices indicates the location of the cutting points over the u time.
direction	If cross-sectional one-dimensional curves are plotted, this indicates whether the cutting points are located on the u time, or on the s time. For plots of the cumulative hazards, only cutting points over the u time are meaningful.
tmax	The maximum value of t that should be plotted.

Value

A list with the following elements: * Haz a list of estimated hazard and associated SEs (obtained from the function get_hazard_2d); * CumHaz the cumulated hazard estimate over u and s; * cause (if provided) the short name for the cause.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
```

```
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(u = fakedata$u,
                            s_out = fakedata$s,
                            ev = fakedata$ev,
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1 ,1.5 ,.5),
                              seq(1 ,1.5 ,.5)))

# Obtain the fake cumulated hazard
fakecumhaz2ts <- cumhaz2ts(fakemod)
```

`cuminc2ts` *Cumulative incidence surface over two time scales*

Description

Cumulative incidence surface over two time scales

Usage

```
cuminc2ts(haz = list(), ds, cause = NULL)
```

Arguments

`haz`	a list of cause-specific hazards
`ds`	the distance between two consecutive intervals over the `s` time scale. This has to be equal for all cause-specific hazards
`cause`	is an optional vector of short names for the causes. It should be of the same length as the number of cause-specific cumulated hazards provided.

Value

a list with one cumulative incidence matrix for each cause-specific hazard (named if a vector of short names is passed to `cause`).

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(u = fakedata$u,
                            s_out = fakedata$s,
                            ev = fakedata$ev,
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1 ,1.5 ,.5),
                              seq(1 ,1.5 ,.5)))

# Obtain the fake cumulated hazard
fakecumhaz2ts <- cumhaz2ts(fakemod)
# Fake cumulative incidence function 2ts
fakecif2ts <- cuminc2ts(haz = list(fakecumhaz2ts$Haz$hazard),
                        ds = .5)
```

`exposures_events_1d` *Bin data on one time scale*

Description

`exposure_events_1d()` computes aggregated measures of exposure times and event counts starting from individual records of time at entry, time at exit and event's indicator, over one time scale (`s`).

Usage

```
exposures_events_1d(s_in = NULL, s_out, ev, bins)
```

Arguments

`s_in`	A vector of (possibly left truncated) times at entry. If this is not provided by the user, the function will consider a value of 0 for all observations.
`s_out`	A vector of times at event or censoring.
`ev`	A vector of events' indicators (possible values 0/1).
`bins`	A vector of interval breaks for discretization (see also `make_bins()`).

Details

The time scale `s` is divided into bins of equal size, which are provided as input to the function. Then, the time-at-risk for each individual is split according to these bins, and an event indicator is placed in the bin where the exit time is located. Finally, the individual contributions are summed in each bin to provide a vector of total exposure time and total event counts. See also `prepare_data()` to conveniently prepare individual data for the analysis with one, or two time scales.

Value

A list with the following elements:

- `R` A matrix of dimension n by ns containing the exposure times for each individual separately.
- `r` A vector of exposure times.
- `Y` A matrix of dimension n by ns containing the event counts for each individual separately
- `y` A vector of event counts.

If the length of the input vectors do not match, an error message is returned.

Author(s)

Angela Carollo <carollo@demogr.mpg.de> and Paul Eilers <p.eilers@erasmus.nl>

Examples

```
# ---- Bin colon cancer data by time since recurrence ----
# First create vector of bins
bins1ts <- make_bins(s_in = reccolon2ts$entrys, s_out = reccolon2ts$timesr, ds = 30)
bindata <- exposures_events_1d(s_in = reccolon2ts$entrys,
s_out = reccolon2ts$timesr, ev = reccolon2ts$status, bins = bins1ts$bins_s)
```

---

exposures_events_2d *Bin data on two time scales*

Description

`exposures_events_2d()` computes individual or aggregated matrices of exposure times and event counts starting from individual records of time at entry in the process (measured over the first time scale), duration at entry in the process (measured over the second time scale), duration at exit from the process (measured over the second time scale), and event's indicator.

Usage

```
exposures_events_2d(u, s_in = NULL, s_out, ev, bins_list, individual = FALSE)
```

Arguments

u	A vector of fixed times at entry in the process, measured over the first time scale.
s_in	A vector of (possibly left truncated) times at entry. If this is not provided by the user, the function will consider a value of 0 for all observations.
s_out	A vector of times at event or censoring.
ev	A vector of events' indicators (possible values 0/1).
bins_list	is a list with the following (necessary) elements (usually prepared by make_bins()): • bins_u a vector of extreme values for the bins over the u axis • bins_s a vector of extreme values for the bins over the s axis
individual	A Boolean. Default is FALSE: if FALSE computes the matrices R and Y collectively for all observations; if TRUE computes the matrices R and Y separately for each individual record.

Details

The fixed-time variable u and the second time scale s are divided into nu and ns intervals, respectively. The extremes of these intervals are provided as input to the function. First, the fixed-time at entry is located in one of the nu bins that cover the whole range of u. Then, the time-at-risk for each individual is split according to the ns bins that span the whole range of values for s, and an event indicator is placed in the bin where the exit time is located. This is done by calling the function exposure_events_1d. If individual matrices of exposure and events are required, then the function returns two arrays of dimension nu by ns by n. If aggregated results are preferred, the individual contributions are summed in each bin to provide a matrix of total exposure time and a matrix of total event counts, both of dimensions nu by ns. See also prepare_data() to conveniently prepare individual data for the analysis with one, or two time scales.

Value

A list with the following elements:

- R an array of exposure times: if individual == TRUE, then R is an array of dimension nu by ns by n, otherwise is an array of dimension nu by ns
- Yan array of event counts: if individual == TRUE, then Y is an array of dimension nu by ns by n, otherwise is an array of dimension nu by ns

Author(s)

Angela Carollo <carollo@demogr.mpg.de>

Examples

```
# ---- Bin colon cancer data by time at randomization and time since recurrence ----
# First create vectors of bins (using function `make_bins()`)
bins <- make_bins(u = reccolon2ts$timer, s_out = reccolon2ts$timesr,
du = 30, ds = 30)
# Now bin data (note: the s_in argument is omitted because data are not left truncated)
bindata2d <- exposures_events_2d(u = reccolon2ts$timer,
s_out = reccolon2ts$timesr, ev = reccolon2ts$status, bins = bins)
```

`exposures_events_Lexis` *Bin data on the Lexis diagram*

Description

`exposures_events_Lexis()` computes aggregated matrices of exposure times and event counts over two time scales, on the Lexis diagram.

The time scales are `t` and `s`. This function uses functions from the package `popEpi` and from the package `Epi`, and code shared by Bendix Carstensen on the website bendixcarstensen.com. See also `prepare_data()` to conveniently prepare individual data for the analysis with one, or two time scales.

Usage

```
exposures_events_Lexis(t_in = NULL, t_out, s_in = NULL, s_out, ev, bins_list)
```

Arguments

`t_in`	(optional) A vector of entry times on the time scale `t`.
`t_out`	(optional) A vector of exit times on the time scale `t`.
`s_in`	(optional) A vector of entry times on the time scale `s`.
`s_out`	A vector of exit times on the time scale `s`.
`ev`	A vector of event indicators (possible values 0/1).
`bins_list`	A list with the following (necessary) elements: • `bins_t` a vector of extreme values for the bins over the `t` axis. • `nt` the number of bins over `t`. • `bins_s` a vector of extreme values for the bins over the `t` axis. • `ns` the number of bins over `s`.

Value

A list with the following elements:

- `R` an array of exposure times of dimension nt by ns
- `Y` an array of event counts of dimension nt by ns

Author(s)

Angela Carollo <carollo@demogr.mpg.de>

References

Carstensen B, Plummer M, Laara E, Hills M (2022). Epi: A Package for Statistical Analysis in Epidemiology. R package version 2.47.1, https://CRAN.R-project.org/package=Epi.

Miettinen J, Rantanen M, Seppa K (2023). popEpi: Functions for Epidemiological Analysis using Population Data. R package version 0.4.11, https://cran.r-project.org/package=popEpi.

Examples

```
# ---- Bin colon cancer data by time since randomization and time since recurrence ----
# First create vectors of bins (using function `make_bins()`)
bins <- make_bins(t_out = reccolon2ts$timedc, s_out = reccolon2ts$timesr,
dt = 90, ds = 90)
# Now bin data (note: the t_in and s_in arguments are omitted because data are not left truncated)
bindata2d <- exposures_events_Lexis(t_out = reccolon2ts$timedc,
s_out = reccolon2ts$timesr, ev = reccolon2ts$status, bins = bins)
```

---

fit1ts *Fit a smooth hazard model with one time scale*

Description

`fit1ts()` fits a smooth hazard model with one time scale.

Three methods are implemented for the search of the optimal smoothing parameter (and therefore optimal model): a numerical optimization of the AIC or BIC of the model, a search for the minimum AIC or BIC of the model over a grid of $\log_{10}$ values for the smoothing parameter and the estimation using a sparse mixed model representation of P-splines. Construction of the B-splines basis and of the penalty matrix is incorporated within the function. If a matrix of covariates is provided, the function will estimate a model with covariates.

Usage

```
fit1ts(
  data1ts = NULL,
  y = NULL,
  r = NULL,
  Z = NULL,
  bins = NULL,
  Bbases_spec = list(),
  Wprior = NULL,
  pord = 2,
  optim_method = c("ucminf", "grid_search", "LMMsolver"),
  optim_criterion = c("aic", "bic"),
  lrho = 0,
  ridge = 0,
  control_algorithm = list(),
  par_gridsearch = list()
)
```

Arguments

data1ts	(optional) an object created by the function prepare_data(). Providing this input is the easiest way to use the function fit1ts. However, the user can also provide the input data together with a list of bins, as explained by the following parameters' descriptions.
y	A vector of event counts of length ns, or an array of dimension ns by n.
r	A vector of exposure times of length ns, or an array of dimension ns by n.
Z	(optional) A regression matrix of covariates of dimension n by p.
bins	a list with the specification for the bins. This is created by the function prepare_data. Alternatively, a list with the following elements can be provided: * bins_s is a vector of intervals for the time scale s. * mids is a vector with the midpoints of the intervals over s. * ns is the number of bins over s.
Bbases_spec	A list with the specification for the B-splines basis with the following elements: • bdeg The degree of the B-splines basis. Default is 3 (for cubic B-splines). • nseg_s The number of segments for the B-splines over s. Default is 10. • min_s (optional) The lower limit of the domain of Bs. Default is min(bins_s). • max_s (optional) The upper limit of the domain of Bs. Default is max(bins_s).
Wprior	An optional vector of a-priori weights.
pord	The order of the penalty. Default is 2.
optim_method	The method to be used for optimization: "ucminf" (default) for the numerical optimization of the AIC (or BIC), "grid_search" for a grid search of the minimum AIC (or BIC) over a grid of $\log_{10}(\varrho_s)$ values, and "LMMsolver" to solve the model as sparse linear mixed model using the package LMMsolver.
optim_criterion	The criterion to be used for optimization: "aic" (default) or "bic". BIC penalized model complexity more strongly than AIC, so that its usage is recommended when a smoother fit is preferable (see also Camarda, 2012).
lrho	A number if optim_method == "ucminf", default is 0. A vector of values for $\log_{10}(\varrho_s)$ if optim_method == "grid_search". In the latter case, if a vector is not provided, a default sequence of values is used for $\log_{10}(\varrho_s)$.
ridge	A ridge penalty parameter: default is 0.
control_algorithm	A list with optional values for the parameters of the iterative processes: • maxiter The maximum number of iteration for the IWSL algorithm. Default is 20. • conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5. • verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process. • monitor_ev A Boolean. Default is FALSE. If TRUE monitors the evaluation of the model over the log_10(rho_s) values.
par_gridsearch	A list of parameters for the grid_search:

- plot_aic A Boolean. Default is FALSE. If TRUE, plot the AIC values over the grid of log_10(rhos) values.
- plot_bic A Boolean. Default is FALSE. If TRUE, plot the BIC values over the grid of log_10(rhos) values.
- return_aic A Boolean. Default is TRUE. Return the AIC values.
- return_bic A Boolean. Default is TRUE. Return the BIC values.
- mark_optimal A Boolean. Default is TRUE. If the plot of the AIC or BIC values is returned, marks the optimal log_10(rho_s) in the plot.
- main_aic The title of the AIC plot. Default is "AIC grid".
- main_bic The title of the BIC plot. Default is "BIC grid".

Details

Some functions from the R-package LMMsolver are used here. We refer the interested readers to https://biometris.github.io/LMMsolver/ for more detail on LMMsolver and its usage.

Value

An object of class haz1ts, or of class haz1tsLMM. For objects of class haz1ts this is

- optimal_model A list with:
  - alpha The vector of estimated P-splines coefficients of length cs.
  - SE_alpha The vector of estimated Standard Errors for the alpha coefficients, of length cs.
  - beta The vector of estimated covariate coefficients of length p (if model with covariates).
  - se_beta The vector of estimated Standard Errors for the beta coefficients of length p (if model with covariates).
  - eta or eta0. The vector of values of the (baseline) linear predictor (log-hazard) of length ns.
  - H The hat-matrix.
  - Cov The full variance-covariance matrix.
  - deviance The deviance.
  - ed The effective dimension of the model.
  - aic The value of the AIC.
  - bic The value of the BIC.
  - Bbases a list with the B-spline basis Bs (this is a list for compatibility with functions in 2d).
- optimal_logrho The optimal value of log10(rho_s).
- P_optimal The optimal penalty matrix P.
- AIC (if par_gridsearch$return_aic == TRUE) The vector of AIC values.
- BIC (if par_gridsearch$return_bic == TRUE) The vector of BIC values.

Objects of class haz1tsLMM have a slight different structure. They are a list with:

- optimal_model an object of class LMMsolve
- AIC_BIC a list with, among other things, the AIC and BIC values and the ED of the model

- n_events the number of events
- ns the number of bins over the s-axis
- cs the number of B-splines over the s-axis
- covariates an indicator for PH model

References

Boer, Martin P. 2023. "Tensor Product P-Splines Using a Sparse Mixed Model Formulation." Statistical Modelling 23 (5-6): 465–79. https://doi.org/10.1177/1471082X231178591.#'

Examples

```
## preparing data - no covariates
dt1ts <- prepare_data(data = reccolon2ts,
                      s_in = "entrys",
                      s_out = "timesr",
                      events = "status",
                      ds = 180)

## fitting the model with fit1ts() - default options, that is ucminf optimization

mod1 <- fit1ts(dt1ts)

## fitting with LMMsolver
mod2 <- fit1ts(dt1ts,
               optim_method = "LMMsolver")

## preparing the data - covariates

dt1ts_cov <- prepare_data(data = reccolon2ts,
                      s_in = "entrys",
                      s_out = "timesr",
                      events = "status",
                      ds = 180,
                      individual = TRUE,
                      covs = c("rx", "node4", "sex"))

## fitting the model with fit1ts() - grid search over only two log_10(rho_s) values

mod3 <- fit1ts(dt1ts_cov,
               optim_method = "grid_search",
               lrho = c(1, 1.5))
```

---

fit1tsmodel_ucminf *Numerical optimization of the 1ts model*

Description

`fit1tsmodel_ucminf()` performs a numerical optimization of the AIC or BIC of the one time scale model.

It finds the optimal values of $\log_{10}(\varrho_s)$ and returns the estimated optimal model. See also `ucminf::ucminf()`.

Usage

```
fit1tsmodel_ucminf(
  r,
  y,
  Z = NULL,
  lrho = 0,
  Bs,
  Ds,
  Wprior = NULL,
  optim_criterion = c("aic", "bic"),
  control_algorithm = list()
)
```

Arguments

`r`	A vector of exposure times of length ns, or an array of dimension ns by n.
`y`	A vector of event counts of length ns, or an array of dimension ns by n.
`Z`	(optional) A regression matrix of covariates of dimension n by p.
`lrho`	A starting value for $\log_{10}(\varrho_s)$. Default is 0.
`Bs`	A matrix of B-splines for the time scale `s`.
`Ds`	The difference matrix of the penalty.
`Wprior`	An optional vector of a-priori weights.
`optim_criterion`	The criterion to be used for optimization: `"aic"` (default) or `"bic"`.
`control_algorithm`	A list with optional values for the parameters of the iterative processes: • `maxiter` The maximum number of iteration for the IWSL algorithm. Default is 20. • `conv_crit` The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is `1e-5`. • `verbose` A Boolean. Default is `FALSE`. If `TRUE` monitors the iteration process. • `monitor_ev` A Boolean. Default is `FALSE`. If `TRUE` monitors the evaluation of the model over the `log_10(rho_s)` values.

Value

An object of class `haz1ts` with the following elements:

- `optimal_model` A list containing the results of the optimal model.

- optimal_logrho The optimal value of log10(rho_s).
- P_optimal The optimal penalty matrix P.

References

Nielsen H, Mortensen S (2024). *ucminf: General-Purpose Unconstrained Non-Linear Optimization*. R package version 1.2.2, https://CRAN.R-project.org/package=ucminf

---

fit2ts *Fit a smooth hazard model with two time scales*

Description

fit2ts() fits a smooth hazard model with two time scales.

Three methods are implemented for the search of the optimal smoothing parameters (and therefore optimal model): a numerical optimization of the AIC or BIC of the model, a search for the minimum AIC or BIC of the model over a grid of log_10 values for the smoothing parameters, and a solution that uses a sparse mixed model representation of the P-spline model to estimate the smoothing parameters. Construction of the B-splines bases and of the penalty matrix is incorporated within the function. If a matrix of covariates is provided, the function will estimate a model with covariates.

Usage

```
fit2ts(
  data2ts = NULL,
  Y = NULL,
  R = NULL,
  Z = NULL,
  bins = NULL,
  Bbases_spec = list(),
  pord = 2,
  optim_method = c("ucminf", "grid_search", "LMMsolver"),
  optim_criterion = c("aic", "bic"),
  lrho = c(0, 0),
  Wprior = NULL,
  ridge = 0,
  control_algorithm = list(),
  par_gridsearch = list()
)
```

Arguments

data2ts	(optional) an object of class created by the function prepare_data(). Proving this input is the easiest way to use the function fit2ts. However, the user can also provide the input data together with a list of bins, as explained by the following parameters' descriptions.
Y	A matrix (or 3d-array) of event counts of dimension nu by ns (or nu by ns by n).

R — A matrix (or 3d-array) of exposure times of dimension nu by ns (or nu by ns by n).

Z — (optional) A regression matrix of covariates values of dimensions n by p.

bins — a list with the specification for the bins. This is created by the function prepare_data. If a list prepared externally from such function if provided, it should contain the following elements: * bins_u A vector of bins extremes for the time scale u. * midu A vector with the midpoints of the bins over u. * nu The number of bins over u. * bins_s A vector of bins extremes for the time scale s. * mids A vector with the midpoints of the bins over s. * ns The number of bins over s.

Bbases_spec — A list with the specification for the B-splines basis with the following elements:

- bdeg The degree of the B-splines basis. Default is 3 (for cubic B-splines).
- nseg_u The number of segments for the B-splines over u. Default is 10.
- min_u (optional) The lower limit of the domain of Bu. Default is min(bins_u).
- max_u (optional) The upper limit of the domain of Bu. Default is max(bins_u).
- nseg_s The number of segments for the B-splines over s. Default is 10.
- min_s (optional) The lower limit of the domain of Bs. Default is min(bins_s).
- max_s (optional) The upper limit of the domain of Bs. Default is max(bins_s).

pord — The order of the penalty. Default is 2.

optim_method — The method to be used for optimization: "ucminf" (default) for the numerical optimization of the AIC (or BIC), "grid_search" for a grid search of the minimum AIC (or BIC) over a grid of log_10(rho_u) and log_10(rho_s) values, or "LMMsolver" to solve the model as sparse linear mixed model using the package LMMsolver.

optim_criterion — The criterion to be used for optimization: "aic" (default) or "bic". BIC penalized model complexity more strongly than AIC, so that its usage is recommended when a smoother fit is preferable (see also Camarda, 2012).

lrho — A vector of two elements if optim_method == "ucminf". Default is c(0,0). A list of two vectors of values for log_10(rho_u) and log_10(rho_s) if optim_method == "grid_search". In the latter case, if a list with two vectors is not provided, a default sequence of values is used for both log_10(rho_u) and log_10(rho_s).

Wprior — An optional matrix of a-priori weights.

ridge — A ridge penalty parameter: default is 0. This is useful when, in some cases the algorithm shows convergence problems. In this case, set to a small number, for example 1e-4.

control_algorithm — A list with optional values for the parameters of the iterative processes:

- maxiter The maximum number of iteration for the IWSL algorithm. Default is 20.
- conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5.
- verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process.

- monitor_ev A Boolean. Default is FALSE. If TRUE monitors the evaluation of the model over the log_10(rho_s) values.

par_gridsearch A list of parameters for the grid_search:

- plot_aic A Boolean. Default is FALSE. If TRUE, plot the AIC values over the grid of log_10(rho_u) and log_10(rho_s) values.
- plot_bic A Boolean. Default is FALSE. If TRUE, plot the BIC values over the grid of log_10(rho_u) and log_10(rho_s) values.
- return_aic A Boolean. Default is TRUE. Return the AIC values.
- return_bic A Boolean. Default is TRUE. Return the BIC values.
- col The color palette to be used for the AIC/BIC plot. Default is grDevices::gray.colors(n=10).
- plot_contour A Boolean. Default is TRUE. Adds white contour lines to the AIC/BIC plot.
- mark_optimal A Boolean. Default is TRUE. If the plot of the AIC or BIC values is returned, marks the optimal combination of log_10(rho_u) and log_10(rho_s) in the plot.
- main_aic The title of the AIC plot. Default is "AIC grid".
- main_bic The title of the BIC plot. Default is "BIC grid".

Details

Some functions from the R-package LMMsolver are used here. We refer the interested readers to https://biometris.github.io/LMMsolver/ for more details on LMMsolver and its usage.

Value

An object of class haz2ts, or of class haz2tsLMM. For objects of class haz2ts this is

- optimal_model A list with :
  - Alpha The matrix of estimated P-splines coefficients of dimension c_u by c_s.
  - Cov_alpha The variance-covariance matrix of the Alpha coefficients, of dimension $c_u c_s$ by $c_u c_s$.
  - beta The vector of length p of estimated covariates coefficients (if model with covariates).
  - Cov_beta The variance-covariance matrix of the beta coefficients, of dimension p by p (if model with covariates).
  - SE_beta The vector of length p of estimated Standard Errors for the beta coefficients (if model with covariates)..
  - Eta or Eta0 The matrix of values of the (baseline) linear predictor (log-hazard) of dimension n_u by n_s.
  - H The hat-matrix.
  - deviance The deviance.
  - ed The effective dimension of the model.
  - aic The value of the AIC.
  - bic The value of the BIC.
  - Bbases a list with the B-spline bases Bu and Bs
- optimal_logrho A vector with the optimal values of log10(rho_u) and log10(rho_s).

- P_optimal The optimal penalty matrix P.
- AIC (if par_gridsearch$return_aic == TRUE) The matrix of AIC values.
- BIC (if par_gridsearch$return_bic == TRUE) The matrix of BIC values.

Objects of class haz2tsLMM have a slight different structure. They are a list with:

- optimal_model an object of class LMMsolve
- AIC_BIC a list with, among other things, the AIC and BIC values and the ED of the model
- n_events the number of events
- nu the number of bins over the u-axis
- ns the number of bins over the s-axis
- cu the number of B-splines over the u-axis
- cs the number of B-splines over the s-axis
- covariates an indicator for PH model

References

Boer, Martin P. 2023. "Tensor Product P-Splines Using a Sparse Mixed Model Formulation." Statistical Modelling 23 (5-6): 465–79. https://doi.org/10.1177/1471082X231178591. Carollo, Angela, Paul H. C. Eilers, Hein Putter, and Jutta Gampe. 2023. "Smooth Hazards with Multiple Time Scales." arXiv Preprint: https://arxiv.org/abs/http://arxiv.org/abs/2305.09342v1

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(data = fakedata,
                            u = "u",
                            s_out = "s",
                            ev = "ev",
                            ds = .5)
# Fit a fake model - not optimal smoothing
fit2ts(fakedata2ts,
       optim_method = "grid_search",
       lrho = list(seq(1, 1.5, .5), seq(1, 1.5, .5)))
# For more examples please check the vignettes!!! Running more complicated examples
# here would imply longer running times...
```

`fit2tsmodel_ucminf` *Numerical optimization of the 2ts model*

Description

`fit2tsmodel_ucminf()` performs a numerical optimization of the AIC or BIC of the two time scales model.

It finds the optimal values of `log_10(rho_u)` and `log_10(rho_s)` and returns the estimated optimal model. See also `ucminf::ucminf()`.

Usage

```
fit2tsmodel_ucminf(
  Y,
  R,
  Z = NULL,
  optim_criterion = c("aic", "bic"),
  lrho = c(0, 0),
  Bu,
  Bs,
  Iu,
  Is,
  Du,
  Ds,
  Wprior = NULL,
  ridge = 0,
  control_algorithm = list()
)
```

Arguments

`Y`	A matrix (or 3d-array) of event counts of dimension nu by ns (or nu by ns by n).
`R`	A matrix (or 3d-array) of exposure times of dimension nu by ns (or nu by ns by n).
`Z`	(optional) A regression matrix of covariates values of dimensions n by p.
`optim_criterion`	The criterion to be used for optimization: `"aic"` (default) or `"bic"`.
`lrho`	A vector of two elements, the initial values for $\log_{10}(\varrho_u)$ and $\log_{10}(\varrho_s)$.
`Bu`	A matrix of B-splines for the u time scale of dimension nu by cu.
`Bs`	A matrix of B-splines for the s time scale of dimension ns by cs.
`Iu`	An identity matrix of dimension nbu by nbu.
`Is`	An identity matrix of dimension nbs by nbs.
`Du`	The difference matrix over u.
`Ds`	The difference matrix over s.

`Wprior` An optional matrix of a-priori weights.

`ridge` A ridge penalty parameter: default is 0. This is useful when, in some cases the algorithm shows convergence problems. In this case, set to a small number, for example `1e-4`.

`control_algorithm`
A list with optional values for the parameters of the iterative processes:

- `maxiter` The maximum number of iteration for the IWSL algorithm. Default is 20.
- `conv_crit` The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is `1e-5`.
- `verbose` A Boolean. Default is `FALSE`. If `TRUE` monitors the iteration process.
- `monitor_ev` A Boolean. Default is `FALSE`. If `TRUE` monitors the evaluation of the model over the `log_10(rho_s)` values.

Value

An object of class `haz2ts` with the following elements:

- `optimal_model` A list containing the results of the optimal model.
- `optimal_logrho` A vector with the optimal values of `log10(rho_u)` and `log10(rho_s)`.
- `P_optimal` The optimal penalty matrix P.

References

Nielsen H, Mortensen S (2024). *ucminf: General-Purpose Unconstrained Non-Linear Optimization*. R package version 1.2.2, https://CRAN.R-project.org/package=ucminf

---

`getAIC_BIC_LMM` *Calculates AIC and BIC from object fitted via LMMsolver*

Description

`getAIC_BIC_LMM` is an utility function that takes an object of class `'LMMsolve'` fitted via `fit1ts()` or `fit2ts()` and calculates AIC, BIC and ED.

Usage

```
getAIC_BIC_LMM(fit, offset)
```

Arguments

`fit` An object of class `"LMMsolve"`

`offset` The vector of exposure times from dataLMM

Value

A list with: * ED effective dimension of the full model; * EDbase effective dimension of the baseline hazard only; * Dev deviance; * AIC the aic; * BIC the bic; * n_beta the number of estimated covariate parameters (if PH model).

`get_aic_fit_1d` *Return the AIC of 1ts model*

Description

`get_aic_fit_1d()` fits the 1ts model with or without individual level covariates and it returns the AIC of the model. See also `fit1tsmodel_ucminf()` and `fit1ts()`.

Usage

```
get_aic_fit_1d(
  lrho,
  r,
  y,
  Z = NULL,
  Bs,
  Ds,
  Wprior = NULL,
 control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE, monitor_ev =
    FALSE)
)
```

Arguments

`lrho`	A starting value for $\log_{10}(\varrho_s)$. Default is 0.
`r`	A vector of exposure times of length ns, or an array of dimension ns by n.
`y`	A vector of event counts of length ns, or an array of dimension ns by n.
`Z`	(optional) A regression matrix of covariates of dimension n by p.
`Bs`	A matrix of B-splines for the time scale s.
`Ds`	The difference matrix of the penalty.
`Wprior`	An optional vector of a-priori weights.
`control_algorithm`	A list with optional values for the parameters of the iterative processes: • `maxiter` The maximum number of iteration for the IWSL algorithm. Default is 20. • `conv_crit` The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is `1e-5`. • `verbose` A Boolean. Default is `FALSE`. If `TRUE` monitors the iteration process. • `monitor_ev` A Boolean. Default is `FALSE`. If `TRUE` monitors the evaluation of the model over the `log_10(rho_s)` values.

Value

The `aic` value of the fitted model.

`get_aic_fit_2d` *Return the AIC of 2ts model*

Description

`get_aic_fit_2d()` fits the 2ts model with or without individual level covariates and it returns the AIC of the model. See also `fit2tsmodel_ucminf()` and `fit2ts()`.

Usage

```
get_aic_fit_2d(
  lrho,
  R,
  Y,
  Z = NULL,
  Bu,
  Bs,
  Iu,
  Is,
  Du,
  Ds,
  Wprior = NULL,
  ridge = 0,
 control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE, monitor_ev =
    FALSE)
)
```

Arguments

`lrho`	A vector of two elements, the initial values for $\log_{10}(\varrho_u)$ and $\log_{10}(\varrho_s)$.
`R`	A matrix (or 3d-array) of exposure times of dimension nu by ns (or nu by ns by n).
`Y`	A matrix (or 3d-array) of event counts of dimension nu by ns (or nu by ns by n).
`Z`	(optional) A regression matrix of covariates values of dimensions n by p.
`Bu`	A matrix of B-splines for the u time scale of dimension nu by cu.
`Bs`	A matrix of B-splines for the s time scale of dimension ns by cs.
`Iu`	An identity matrix of dimension nbu by nbu.
`Is`	An identity matrix of dimension nbs by nbs.
`Du`	The difference matrix over u.
`Ds`	The difference matrix over s.

`Wprior` An optional matrix of a-priori weights.

`ridge` A ridge penalty parameter: default is 0. This is useful when, in some cases the algorithm shows convergence problems. In this case, set to a small number, for example `1e-4`.

`control_algorithm`
A list with optional values for the parameters of the iterative processes:

- `maxiter` The maximum number of iteration for the IWSL algorithm. Default is 20.
- `conv_crit` The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is `1e-5`.
- `verbose` A Boolean. Default is `FALSE`. If `TRUE` monitors the iteration process.
- `monitor_ev` A Boolean. Default is `FALSE`. If `TRUE` monitors the evaluation of the model over the `log_10(rho_s)` values.

Value

The `aic` value of the fitted model.

`get_bic_fit_1d` *Return the BIC of 1ts model*

Description

`get_bic_fit_1d()` fits the 1ts model with or without individual level covariates and it returns the BIC of the model. See also `fit1tsmodel_ucminf()` and `fit1ts()`.

Usage

```
get_bic_fit_1d(
  lrho,
  r,
  y,
  Z = NULL,
  Bs,
  Ds,
  Wprior = NULL,
 control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE, monitor_ev =
    FALSE)
)
```

Arguments

`lrho` A starting value for $\log_{10}(\varrho_s)$. Default is 0.

`r` A vector of exposure times of length ns, or an array of dimension ns by n.

y A vector of event counts of length ns, or an array of dimension ns by n.

Z (optional) A regression matrix of covariates of dimension n by p.

Bs A matrix of B-splines for the time scale s.

Ds The difference matrix of the penalty.

Wprior An optional vector of a-priori weights.

control_algorithm
A list with optional values for the parameters of the iterative processes:

- maxiter The maximum number of iteration for the IWSL algorithm. Default is 20.
- conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5.
- verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process.
- monitor_ev A Boolean. Default is FALSE. If TRUE monitors the evaluation of the model over the log_10(rho_s) values.

Value

the bic value of the fitted model.

---

get_bic_fit_2d *Return the BIC of 2ts model*

Description

get_bic_fit_2d() fits the 2ts model with or without individual level covariates and it returns the BIC of the model. See also fit2tsmodel_ucminf() and fit2ts().

Usage

```
get_bic_fit_2d(
  lrho,
  R,
  Y,
  Z = NULL,
  Bu,
  Bs,
  Iu,
  Is,
  Du,
  Ds,
  Wprior = NULL,
  ridge = 0,
 control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE, monitor_ev =
    FALSE)
)
```

Arguments

lrho	A vector of two elements, the initial values for $\log_{10}(\varrho_u)$ and $\log_{10}(\varrho_s)$.
R	A matrix (or 3d-array) of exposure times of dimension nu by ns (or nu by ns by n).
Y	A matrix (or 3d-array) of event counts of dimension nu by ns (or nu by ns by n).
Z	(optional) A regression matrix of covariates values of dimensions n by p.
Bu	A matrix of B-splines for the u time scale of dimension nu by cu.
Bs	A matrix of B-splines for the s time scale of dimension ns by cs.
Iu	An identity matrix of dimension nbu by nbu.
Is	An identity matrix of dimension nbs by nbs.
Du	The difference matrix over u.
Ds	The difference matrix over s.
Wprior	An optional matrix of a-priori weights.
ridge	A ridge penalty parameter: default is 0. This is useful when, in some cases the algorithm shows convergence problems. In this case, set to a small number, for example 1e-4.
control_algorithm	A list with optional values for the parameters of the iterative processes: • maxiter The maximum number of iteration for the IWSL algorithm. Default is 20. • conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5. • verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process. • monitor_ev A Boolean. Default is FALSE. If TRUE monitors the evaluation of the model over the log_10(rho_s) values.

Value

The bic value of the fitted model.

---

get_hazard_1d *Get estimated (log-)hazard values with 1 time scale*

Description

get_hazard_1d() takes as input the results of a model estimated by fit1ts and it returns the estimated values of the smooth log-hazard and the smooth hazard together with their standard errors.

If the model includes covariates, then only the baseline (log-)hazard is returned. It is possible to provide values that define a new grid for evaluation of the estimated hazard. If not specified, the hazard is evaluated on the same grid used for the binning of the data, and therefore the estimation of the model. The function will check if the parameters for the new grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

Usage

```
get_hazard_1d(fitted_model, plot_grid = NULL)
```

Arguments

fitted_model	is an object of class "haz1ts", the output of the function fit1ts().
plot_grid	(optional) A named vector containing the parameters to build a new grid of intervals over s for plotting the estimated hazard on a finer grid. This must be of the form: plot_grid = c(smin, smax, ds), where smin, smax are the minimum and maximum values desired for the intervals over s, and ds is the distance between intervals over s. If not specified, the plotting is done using the same B-splines basis as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

Value

A list with the following elements:

- new_plot_grid A list of parameters that specify the new grid, of the form list("ints", "smin", "smax", "ds")
- hazard A vector containing the estimated hazard values.
- loghazard A vector containing the estimated log-hazard values.
- log10hazard A vector containing the estimated log10-hazard values.
- SE_hazard A vector containing the estimated SEs for the hazard.
- SE_loghazard A vector containing the estimated SEs for the log-hazard.
- SE_log10hazard A vector containing the estimated SEs for the log10-hazard.

Examples

```
## preparing data - no covariates
dt1ts <- prepare_data(
  data = reccolon2ts,
  s_in = "entrys",
  s_out = "timesr",
  events = "status",
  ds = 180
)

## fitting the model with fit1ts() - default options

mod1 <- fit1ts(dt1ts)
# Obtain 1d hazard
get_hazard_1d(mod1)
# Change grid
get_hazard_1d(mod1,
  plot_grid = c(smin = 0, smax = 2730, ds = 30)
```

```
)
```

`get_hazard_1d_LMM` *Get estimated (log-)hazard values with 1 time scale*

Description

`get_hazard_1d_LMM()` takes as input the results of a model estimated by `fit1ts` and it returns the estimated values of the smooth log-hazard and the smooth hazard together with their standard errors.

If the model includes covariates, then only the baseline (log-)hazard is returned. It is possible to provide values that define a new grid for evaluation of the estimated hazard. If not specified, the hazard is evaluated on the same grid used for the binning of the data, and therefore the estimation of the model. The function will check if the parameters for the new grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

Usage

```
get_hazard_1d_LMM(fitted_model, plot_grid = NULL)
```

Arguments

`fitted_model`	is an object of class `"haz1tsLMM"`, the output of the function `fit1ts()`.
`plot_grid`	(optional) A named vector containing the parameters to build a new grid of intervals over `s` for plotting the estimated hazard on a finer grid. This must be of the form: `plot_grid = c(smin, smax, ds)`, where `smin`, `smax` are the minimum and maximum values desired for the intervals over `s`, and `ds` is the distance between intervals over `s`. If not specified, the plotting is done using the same B-splines basis as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

Value

A list with the following elements:

- `new_plot_grid` A list of parameters that specify the new grid, of the form list("ints", "smin", "smax", "ds")
- `hazard` A vector containing the estimated hazard values.
- `loghazard` A vector containing the estimated log-hazard values.
- `log10hazard` A vector containing the estimated log10-hazard values.
- `SE_hazard` A vector containing the estimated SEs for the hazard.
- `SE_loghazard` A vector containing the estimated SEs for the log-hazard.
- `SE_log10hazard` A vector containing the estimated SEs for the log10-hazard.

Examples

```
## preparing data - no covariates
dt1ts <- prepare_data(
  data = reccolon2ts,
  s_in = "entrys",
  s_out = "timesr",
  events = "status",
  ds = 180
)

## fitting the model with fit1ts()

mod1 <- fit1ts(dt1ts,
  optim_method = "LMMsolver"
)
# Obtain 1d hazard
get_hazard_1d_LMM(mod1)
# Change grid
get_hazard_1d_LMM(mod1,
  plot_grid = c(smin = 0, smax = 2730, ds = 30)
)
```

`get_hazard_2d` *Get estimated (log-)hazard surface with 2 time scales*

Description

`get_hazard_2d()` takes as input the results of a model estimated by `fit2ts` and it returns the estimated smooth log-hazard and the smooth hazard together with their standard errors.

It is possible to provide values that define a new grid for evaluation of the estimated hazard. If not specified, the hazard is evaluated on the same grid used for the binning of the data, and therefore the estimation of the model. The function will check if the parameters for the new grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

Usage

```
get_hazard_2d(
  fitted_model,
  plot_grid = NULL,
  where_slices = NULL,
  direction = c("u", "s", NULL),
  tmax = NULL,
  midpoints = FALSE
)
```

Arguments

fitted_model	is an object of class "haz2ts", the output of the function fit2ts().
plot_grid	(optional) A list containing the parameters to build a new finer grid of intervals over u and s for plotting. This must be of the form: plot_grid = list(c(umin, umax, du), c(smin, smax, ds)), where umin, umax and smin, smax are the minimum and maximum values desired for the intervals over u and s respectively, and du, ds are distances between intervals over u and s respectively. Specifying a new denser grid is used to evaluate the B-spline bases used for estimation on such grid and plot the estimated surfaces with a greater level of details. If not specified, the plotting is done using the same B-splines bases as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.
where_slices	A vector of values for the cutting points of the desired slices of the surface. If which_plot == "slices", please provide this argument. Please also provide this argument in case which_plot = "survival or which_plot = "cumhaz and surv_slices = TRUE or cumhaz_slices = TRUE, respectively.
direction	If which_plot == "slices", indicates the direction for cutting the surface. If u, then the surface will be cut at the selected values of u (indicated by where_slices), hence obtaining one-dimensional curves over s. If s, then the surface will be cut at the selected values of s (indicated by where_slices), hence obtaining one-dimensional curves over u.
tmax	The maximum value of t that should be plotted.
midpoints	A Boolean. Default is FALSE. If TRUE, the estimated quantities are evaluated at the midpoints of the rectangles (or parallelograms) of the grids, rather than at each grid-point.

Value

A list with the following elements:

- new_plot_grid A list of parameters that specify the new grid, of the form list("intu", "umin", "umax", "du", "ints", "smin", "smax", "ds")
- nBu The B-spline basis for u, evaluated over the new grid.
- nBs The B-spline basis for s, evaluated over the new grid.
- hazard A matrix containing the estimated hazard values.
- loghazard A matrix containing the estimated log-hazard values.
- log10hazardA matrix containing the estimated log10-hazard values.
- SE_hazard A matrix containing the estimated SEs for the hazard.
- SE_loghazard A matrix containing the estimated SEs for the log-hazard.
- SE_log10haz A matrix containing the estimated SEs for the log10-hazard.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(data = fakedata,
                            u = "u",
                            s_out = "s",
                            ev = "ev",
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1 ,1.5 ,.5),
                              seq(1 ,1.5 ,.5)))
# Obtain 2d hazard
get_hazard_2d(fakemod)

get_hazard_2d(fakemod,
          plot_grid = list(c(umin = 3, umax = 8.5, du = .1),
                           c(smin = 0, smax = 7.1, ds = .1)))
```

---

get_hazard_2d_LMM *Get estimated (log-)hazard surface with 2 time scales*

Description

`get_hazard_2d_LMM()` takes as input an object of class `'haz2tsLMM'` and it returns the estimated smooth log-hazard, the log10-hazard and the hazard surface together with their standard errors.

It is possible to provide values that define a new grid for evaluation of the estimated hazard. If not specified, the hazard is evaluated on the same grid used for the binning of the data, and therefore the estimation of the model. The function will check if the parameters for the new grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

Usage

```
get_hazard_2d_LMM(
  fitted_model,
  plot_grid = NULL,
  where_slices = NULL,
  direction = c("u", "s", NULL),
  tmax = NULL
)
```

Arguments

fitted_model	is an object of class 'haz2tsLMM' the output of the function fit2ts().
plot_grid	A list containing the parameters to build a new finer grid of intervals over u and s for plotting. This must be of the form: plot_grid = list(c(umin, umax, du), c(smin, smax, ds)), where umin, umax and smin, smax are the minimum and maximum values desired for the intervals over u and s respectively, and du, ds are distances between intervals over u and s respectively. Specifying a new denser grid is used to evaluate the B-spline bases used for estimation on such grid and plot the estimated surfaces with a greater level of details. If not specified, the plotting is done using the same B-splines bases as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned. While for objects of class 'haz2ts' this is an optional input, we strongly recommend to provide the plotting grid for object of class 'haz2tsLMM', given that evaluation of the B-splines works a bit differently in LMMsolver.
where_slices	A vector of values for the cutting points of the desired slices of the surface. If which_plot == "slices", please provide this argument. Please also provide this argument in case which_plot = "survival or which_plot = "cumhaz and surv_slices = TRUE or cumhaz_slices = TRUE, respectively.
direction	If which_plot == "slices", indicates the direction for cutting the surface. If u, then the surface will be cut at the selected values of u (indicated by where_slices), hence obtaining one-dimensional curves over s. If s, then the surface will be cut at the selected values of s (indicated by where_slices), hence obtaining one-dimensional curves over u.
tmax	The maximum value of t that should be plotted.

Value

A list with the following elements:

- new_plot_grid A list of parameters that specify the new grid, of the form list("intu", "umin", "umax", "du", "ints", "smin", "smax", "ds")
- hazard A matrix containing the estimated hazard values.
- loghazard A matrix containing the estimated log-hazard values.
- log10hazardA matrix containing the estimated log10-hazard values.
- SE_hazard A matrix containing the estimated SEs for the hazard.
- SE_loghazard A matrix containing the estimated SEs for the log-hazard.
- SE_log10haz A matrix containing the estimated SEs for the log10-hazard.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(
```

```
  5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
  4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96
)
s <- c(
  0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
  7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00
)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1) #'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(data = fakedata,
                            u = "u",
                            s_out = "s",
                            ev = "ev",
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
  optim_method = "LMMsolver"
)

# Get hazard
get_hazard_2d_LMM(fakemod)

# Use a finer grid of points
get_hazard_2d_LMM(fakemod,
                  plot_grid = list(c(umin = 3, umax = 8.5, du = .1),
                                   c(smin = 0, smax = 7.1, ds = .1)))
```

`get_hr` *Get the Hazard Ratios with their Standard Errors*

Description

`get_hr()` takes as input the results of a model with covariates estimated by `fit2ts` or `fit1ts` and returns the estimated hazard ratios together with their standard errors.

Usage

```
get_hr(fitted_model)
```

Arguments

`fitted_model`	A list returned by the function `fit2ts` or `fit1ts`.

Value

A list with the following elements:

- `HR` A vector of hazard ratios (calculated as $\exp(\hat{\beta})$).

- SE_HR A vector of Standard Errors for the hazard ratios calculated via the delta method.
- beta A vector of the estimated $\hat{\beta}$ coefficients.
- SE_beta A vector of the Standard Errors for the beta coefficients.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)
x1 <- c(0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0)

fakedata <- as.data.frame(cbind(id, u, s, ev, x1))
fakedata2ts <- prepare_data(data = fakedata,
                            u = "u",
                            s_out = "s",
                            ev = "ev",
                            ds = .5,
                            individual = TRUE,
                            covs = "x1")
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1, 1.5, .5),
                              seq(1, 1.5, .5)))
get_hr(fakemod)
```

GLAM_1d_covariates *Fit the 1d GLAM with covariates*

Description

GLAM_1d_covariates() fits a GLAM for the hazard with one time scale, with covariates.

Usage

```
GLAM_1d_covariates(
  R,
  Y,
  Bs,
  Z = Z,
  Wprior = NULL,
  P,
  control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE)
)
```

Arguments

R	A 2d-array of dimensions ns by n containing exposure times.
Y	A 2d-array of dimensions ns by n containing event indicators.
Bs	A matrix of B-splines for the s time scale of dimension ns by cs.
Z	A regression matrix of covariates values of dimensions n by p.
Wprior	An optional vector of length ns of a-priori weights.
P	The penalty matrix of dimension cs by cs.
control_algorithm	A list with optional values for the parameters of iterative processes: *maxiter The maximum number of iterations for the IWSL algorithm, default is 20 . * conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1, default is 1e-5. * verbose A Boolean. Default is FALSE. If TRUE, monitor the iteration process.

Value

A list with the following elements:

- alpha The vector of estimated P-splines coefficients of length cs.
- SE_alpha The vector of estimated Standard Errors for the alpha coefficients, of length cs.
- beta The vector of length p of estimated covariates coefficients.
- se_beta The vector of length p of estimated Standard Errors for the beta coefficients.
- eta0 The vector of values of the baseline linear predictor (log-hazard).
- H The hat-matrix.
- Cov The full variance-covariance matrix.
- deviance The deviance.
- ed The effective dimension of the model.
- aic The value of the AIC.
- bic The value of the BIC.
- Bbases a list with the B-spline basis Bs (this is a list for compatibility with functions in 2d).

GLAM_2d_covariates *Fit the 2d GLAM with covariates*

Description

GLAM_2d_covariates() fits a GLAM for the hazard with two time scales, with covariates.

Usage

```
GLAM_2d_covariates(
  R,
  Y,
  Bu,
  Bs,
  Z,
  Wprior = NULL,
  P,
  ridge = 0,
  control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE)
)
```

Arguments

R	A 3d-array of dimensions nu by ns by n containing exposure times.
Y	A 3d-array of dimensions nu by ns by n containing event indicators.
Bu	A matrix of B-splines for the u time scale of dimension nu by cu.
Bs	A matrix of B-splines for the s time scale of dimension ns by cs.
Z	(optional) A regression matrix of covariates values of dimensions n by p.
Wprior	An optional matrix of a-priori weights.
P	The penalty matrix of dimension cucs by cucs.
ridge	A ridge penalty parameter: default is 0.
control_algorithm	A list with optional values for the parameters of the iterative processes: • maxiter The maximum number of iteration for the IWSL algorithm. Default is 20. • conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5. • verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process.

Value

A list with the following elements:

- Alpha The matrix of estimated P-splines coefficients of dimension cu by cs.
- Cov_alpha The variance-covariance matrix of the Alpha coefficients, of dimension cucs by cucs.
- beta The vector of length p of estimated covariates coefficients.
- Cov_beta The variance-covariance matrix of the beta coefficients, of dimension p by p.
- SE_beta The vector of length p of estimated Standard Errors for the beta coefficients.
- Eta0 The matrix of values of the baseline linear predictor (log-hazard) of dimension nu by ns.
- H The hat-matrix.

- deviance The deviance.
- ed The effective dimension of the model.
- aic The value of the AIC.
- bic The value of the BIC.
- Bbases a list with the B-spline bases Bu and Bs.

GLAM_2d_no_covariates *Fit the 2d GLAM without covariates*

Description

GLAM_2d_no_covariates() fits a GLAM for the hazard with two time scales, without covariates.

Usage

```
GLAM_2d_no_covariates(
  R,
  Y,
  Bu,
  Bs,
  Wprior = NULL,
  P,
  ridge = 0,
  control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE)
)
```

Arguments

R	A matrix of exposure times of dimension nu by ns.
Y	A matrix of event counts of dimension nu by ns.
Bu	A matrix of B-splines for the u time scale of dimension nu by cu.
Bs	A matrix of B-splines for the s time scale of dimension ns by cs.
Wprior	An optional matrix of a-priori weights.
P	The penalty matrix of dimension cucs by cucs.
ridge	A ridge penalty parameter: default is 0.
control_algorithm	A list with optional values for the parameters of the iterative processes: • maxiter The maximum number of iteration for the IWSL algorithm. Default is 20. • conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5. • verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process.

Value

A list with the following elements:

- `Alpha` The matrix of estimated P-splines coefficients of dimension cu by cs.
- `Cov_alpha` The variance-covariance matrix of the `Alpha` coefficients, of dimension cucs by cucs.
- `Eta0` The matrix of values of the baseline linear predictor (log-hazard) of dimension nu by ns.
- `H` The hat-matrix.
- `deviance` The deviance.
- `ed` The effective dimension of the model.
- `aic` The value of the AIC.
- `bic` The value of the BIC.
- `Bbases` a list with the B-spline bases `Bu` and `Bs`.

`grid_search_1d` *Grid search for the optimal 1ts model*

Description

`grid_search_1d()` performs a grid search for the minimum AIC or BIC of the one time scale model.

It finds the optimal values of `log_10(rho_s)` and returns the estimated optimal model.

Usage

```
grid_search_1d(
  r,
  y,
  Z = NULL,
  lrho,
  Bs,
  Ds,
  Wprior = NULL,
  optim_criterion = c("aic", "bic"),
 control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE, monitor_ev =
    FALSE),
 par_gridsearch = list(plot_aic = FALSE, plot_bic = FALSE, return_aic = TRUE, return_bic
    = TRUE, mark_optimal = TRUE, main_aic = "AIC grid", main_bic = "BIC grid")
)
```

Arguments

r	A vector of exposure times of length ns, or an array of dimension ns by n.
y	A vector of event counts of length ns, or an array of dimension ns by n.
Z	(optional) A regression matrix of covariates of dimension n by p.
lrho	A vector of log_10(rho_s) values.
Bs	A matrix of B-splines for the time scale s.
Ds	The difference matrix of the penalty.
Wprior	An optional vector of a-priori weights.
optim_criterion	The criterion to be used for optimization: "aic" (default) or "bic". BIC penalized model complexity more strongly than AIC, so that its usage is recommended when a smoother fit is preferable (see also Camarda, 2012).
control_algorithm	A list with optional values for the parameters of the iterative processes: • maxiter The maximum number of iteration for the IWSL algorithm. Default is 20. • conv_crit The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is 1e-5. • verbose A Boolean. Default is FALSE. If TRUE monitors the iteration process. • monitor_ev A Boolean. Default is FALSE. If TRUE monitors the evaluation of the model over the log_10(rho_s) values.
par_gridsearch	A list of parameters for the grid_search: • plot_aic A Boolean. Default is FALSE. If TRUE, plot the AIC values over the grid of log_10(rhos) values. • plot_bic A Boolean. Default is FALSE. If TRUE, plot the BIC values over the grid of log_10(rhos) values. • return_aic A Boolean. Default is TRUE. Return the AIC values. • return_bic A Boolean. Default is TRUE. Return the BIC values. • mark_optimal A Boolean. Default is TRUE. If the plot of the AIC or BIC values is returned, marks the optimal log_10(rho_s) in the plot. • main_aic The title of the AIC plot. Default is "AIC grid". • main_bic The title of the BIC plot. Default is "BIC grid".

Value

An object of class h1tsfit with the following elements:

- optimal_model A list containing the results of the optimal model.
- optimal_logrho The optimal value of log10(rho_s).
- P_optimal The optimal penalty matrix P.
- AIC (if par_gridsearch$return_aic == TRUE) The vector of AIC values.
- BIC (if par_gridsearch$return_bic == TRUE) The vector of BIC values.

References

Camarda, C. G. (2012). "MortalitySmooth: An R Package for Smoothing Poisson Counts with P-Splines." Journal of Statistical Software, 50(1), 1–24. https://doi.org/10.18637/jss.v050.i01

`grid_search_2d` *Grid search for the optimal 2ts model*

Description

`grid_search_2d()` performs a grid search for the minimum AIC or BIC of the two time scales model.

It finds the optimal values of `log_10(rho_u)` and `log_10(rho_s)` and returns the estimated optimal model.

Usage

```
grid_search_2d(
  lru,
  lrs,
  R,
  Y,
  Bu,
  Bs,
  Z = NULL,
  Iu,
  Is,
  Du,
  Ds,
  Wprior = NULL,
  ridge = 0,
  optim_criterion = c("aic", "bic"),
 control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE, monitor_ev =
    FALSE),
 par_gridsearch = list(plot_aic = FALSE, plot_bic = FALSE, return_aic = TRUE, return_bic
   = TRUE, col = grey.colors(n = 10), plot_contour = FALSE, mark_optimal = TRUE,
    main_aic = "AIC grid", main_bic = "BIC grid")
)
```

Arguments

`lru`	A vector of `log_10(rho_u)` values.
`lrs`	A vector of `log_10(rho_s)` values.
`R`	A matrix (or 3d-array) of exposure times of dimension nu by ns (or nu by ns by n).
`Y`	A matrix (or 3d-array) of event counts of dimension nu by ns (or nu by ns by n).

Bu A matrix of B-splines for the u time scale of dimension nu by cu.

Bs A matrix of B-splines for the s time scale of dimension ns by cs.

Z (optional) A regression matrix of covariates values of dimensions n by p.

Iu An identity matrix of dimension nbu by nbu.

Is An identity matrix of dimension nbs by nbs.

Du The difference matrix over u.

Ds The difference matrix over s.

Wprior An optional matrix of a-priori weights.

ridge A ridge penalty parameter: default is 0. This is useful when, in some cases the algorithm shows convergence problems. In this case, set to a small number, for example `1e-4`.

optim_criterion

The criterion to be used for optimization: `"aic"` (default) or `"bic"`. BIC penalized model complexity more strongly than AIC, so that its usage is recommended when a smoother fit is preferable (see also Camarda, 2012).

control_algorithm

A list with optional values for the parameters of the iterative processes:

- `maxiter` The maximum number of iteration for the IWSL algorithm. Default is 20.
- `conv_crit` The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is `1e-5`.
- `verbose` A Boolean. Default is `FALSE`. If `TRUE` monitors the iteration process.
- `monitor_ev` A Boolean. Default is `FALSE`. If `TRUE` monitors the evaluation of the model over the `log_10(rho_s)` values.

par_gridsearch A list of parameters for the grid_search:

- `plot_aic` A Boolean. Default is `FALSE`. If `TRUE`, plot the AIC values over the grid of `log_10(rho_u)` and `log_10(rho_s)` values.
- `plot_bic` A Boolean. Default is `FALSE`. If `TRUE`, plot the BIC values over the grid of `log_10(rho_u)` and `log_10(rho_s)` values.
- `return_aic` A Boolean. Default is `TRUE`. Return the AIC values.
- `return_bic` A Boolean. Default is `TRUE`. Return the BIC values.
- `col` The color palette to be used for the AIC/BIC plot. Default is `grDevices::gray.colors(n=10)`.
- `plot_contour` A Boolean. Default is `TRUE`. Adds white contour lines to the AIC/BIC plot.
- `mark_optimal` A Boolean. Default is `TRUE`. If the plot of the AIC or BIC values is returned, marks the optimal combination of `log_10(rho_u)` and `log_10(rho_s)` in the plot.
- `main_aic` The title of the AIC plot. Default is `"AIC grid"`.
- `main_bic` The title of the BIC plot. Default is `"BIC grid"`.

Value

An object of class `h2tsfit` with the following elements:

- `optimal_model` A list containing the results of the optimal model.
- `optimal_logrho` The optimal couple of `log_10(rho_u)` and `log_10(rho_s)` values.
- `P_optimal` The optimal penalty matrix P.
- `AIC` (if `par_gridsearch$return_aic == TRUE`) The vector of AIC values.
- `BIC` (if `par_gridsearch$return_bic == TRUE`) The vector of BIC values.

References

Camarda, C. G. (2012). "MortalitySmooth: An R Package for Smoothing Poisson Counts with P-Splines." Journal of Statistical Software, 50(1), 1–24. https://doi.org/10.18637/jss.v050.i01

`haz2tsLMM_summary` *Summary function for object of class 'haz2tsLMM'*

Description

Summary function for object of class 'haz2tsLMM'

Usage

```
haz2tsLMM_summary(x, ...)
```

Arguments

`x`	an object of class 'haz2tsLMM' returned by the function `fit2ts()`
`...`	further arguments

Value

a printed summary of the fitted model, including optimal smoothing paramters, the effective dimension ED and the AIC/BIC. For model with covariates, a regression table is also returned.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(
  5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
  4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96
)
s <- c(
  0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
  7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00
)
```

```
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1) #'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(data = fakedata,
                            u = "u",
                            s_out = "s",
                            ev = "ev",
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
  optim_method = "LMMsolver"
)
summary(fakemod)
```

`haz2ts_summary` *Summary function for object of class 'haz2ts'*

Description

Summary function for object of class 'haz2ts'

Usage

```
haz2ts_summary(x, ...)
```

Arguments

`x`	an object of class 'haz2ts' returned by the function `fit2ts()`
`...`	further arguments

Value

a printed summary of the fitted model

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(data = fakedata,
                            u = "u",
                            s_out = "s",
                            ev = "ev",
```

```
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1 ,1.5 ,.5),
                              seq(1 ,1.5 ,.5)))
summary(fakemod)
```

`imageplot_2ts` *Image Plot of 2ts hazard*

Description

`imageplot_2ts()` plots an image of the two time scales hazard (or survival or cumulative hazard) with contour lines. This is the default call implemented in plot.haz2ts.

Usage

```
imageplot_2ts(x, y, z, plot_options = list(), ...)
```

Arguments

`x`	The coordinates for the x-axis. This is a vector of intervals over the `u` axis (default), a matrix with the corner points of the parallelograms over the `t` time scale, or a vector of intervals for the `t` time scale.
`y`	The coordinates for the y-axis. This is a vector of intervals over the `s` time scale (default), or a matrix with the corner points of the parallelograms over the `s` time scale.
`z`	The values of the surface to plot, organized in a matrix with dimensions compatible with those of `x` and `y`. The default is to plot the hazard.
`plot_options`	A list of options for the plot: • `loghazard` A Boolean. Default is `FALSE`. If `FALSE` the function returns a plot of the hazard surface, if `TRUE` the function returns a plot of the log-hazard surface. • `log10hazard` A Boolean. Default is `FALSE`. If `TRUE`, then a log10-hazard surface is plotted. • `original` A Boolean. Default is `TRUE`. Plot the (log-)hazard in the (t,s)-plane. If `FALSE`, the (log-)hazard will be plotted in the (u,s)-plane. • `rectangular_grid` A Boolean. Default is `FALSE`. If `TRUE`, a rectangular grid is used for plotting also in the (t,s)-plane as opposed to the grid of parallelograms used as default in the (t,s)-plane. • `col_palette` A function defining the color palette. The default palette is `rev(viridis::plasma())`. • `n_shades` The number of color shades to plot, default is 50.

- breaks The vector of breaks for the color legend. If n_shades is provided, this should be of length n_shades + 1. Otherwise, n_shades will be recalculated accordingly.
- show_legend A Boolean. Default is TRUE. If FALSE no legend will be plotted, useful for multi-panel figures with common legend. Works only for plots on rectangular grid!
- tmax The maximum value of t that should be plotted.
- main The title of the plot.
- xlab The label of the first time axis (plotted on the x axis).
- ylab The label of the second time axis (plotted on the y axis).
- xlim A vector with two elements defining the limits of the time scale on the x axis.
- ylim A vector with two elements defining the limits of the time scale on the y axis.
- cex_main The magnification to be used for the main title, default is 1.2.
- cex_lab The magnification to be used for the axis labels, default is 1.
- contour_lines A Boolean. Default is FALSE. If TRUE white contour lines are added to the surfaces.
- contour_col The color for the contour lines. Default is white.
- contour_cex The magnification to be used for the contour lines. Default is .8.
- contour_nlev The number of contour levels. Default is 10.

... Further arguments to image.plot or image

Value

An image plot of an estimated surface.

---

imageplot_SE *Image Plot of Standard Errors for the 2ts hazard*

Description

imageplot_SE() plots an image of the SEs of the two time scales hazard with contour lines.

Usage

```
imageplot_SE(x, y, z, plot_options = list(), ...)
```

Arguments

x The coordinates for the x-axis. This is a vector of intervals over the u axis (default), a matrix with the corner points of the parallelograms over the t time scale, or a vector of intervals for the t time scale.

y The coordinates for the y-axis. This is a vector of intervals over the s time scale (default), or a matrix with the corner points of the parallelograms over the s time scale.

z The values of the surface to plot, organized in a matrix with dimensions compatible with those of x and y. These can be the SEs for the hazard, the SEs for the log-hazard or the SEs for the log10-hazard.

plot_options A list of options for the plot:

- loghazard A Boolean. Default is FALSE. If FALSE the function returns a plot of the standard errors of the hazard surface, if TRUE the function returns a plot of the standard errors of the log-hazard surface.
- log10hazard A Boolean. Default is FALSE. If TRUE, the function returns a plot of the standard errors of the log10-hazard surface
- original A Boolean. Default is TRUE. Plot the (log-)hazard in the (t,s)-plane. If FALSE, the (log-)hazard will be plotted in the (u,s)-plane.
- rectangular_grid A Boolean. Default is FALSE. If TRUE, a rectangular grid is used for plotting also in the (t,s)-plane as opposed to the grid of parallelograms used as default in the (t,s)-plane.
- col_palette A function defining the color palette. The default palette is rev(colorspace::sequential_hcl(n = 50, "Red-Purple")).
- n_shades The number of color shades to plot, default is 50.
- breaks The vector of breaks for the color legend. If n_shades is provided, this should be of length n_shades + 1. Otherwise, n_shades will be recalculated accordingly.
- show_legend A Boolean. Default is TRUE. If FALSE no legend will be plotted, useful for multi-panel figures with common legend. Works only for plots on rectangular grid!
- tmax The maximum value of t that should be plotted.
- main The title of the plot.
- xlab The label of the first time axis (plotted on the x axis).
- ylab The label of the second time axis (plotted on the y axis).
- xlim A vector with two elements defining the limits of the time scale on the x axis.
- ylim A vector with two elements defining the limits of the time scale on the y axis.
- cex_main The magnification to be used for the main title, default is 1.2.
- cex_lab The magnification to be used for the axis labels, default is 1.
- contour_lines A Boolean. Default is FALSE. If TRUE white contour lines are added to the surfaces.
- contour_col The color for the contour lines. Default is white.
- contour_cex The magnification to be used for the contour lines. Default is .8.
- contour_nlev The number of contour levels. Default is 10.

... Further arguments to image.plot or image

Value

An image plot of the SEs for the (log-) hazard surface.

`iwls_1d` *Iterative Weighted Least Squares algorithm for 1ts model*

Description

`iwls_1d()` fits the 1ts model with IWLS algorithm.

Usage

```
iwls_1d(
  r,
  y,
  Bs,
  P,
  Wprior = NULL,
  control_algorithm = list(maxiter = 20, conv_crit = 1e-05, verbose = FALSE)
)
```

Arguments

`r`	A vector of exposure times of length ns.
`y`	A vector of event counts of length ns.
`Bs`	A matrix of B-splines for the `s` time scale of dimension ns by cs.
`P`	The penalty matrix of dimension cs by cs.
`Wprior`	An optional vector of length ns of a-priori weights.
`control_algorithm`	A list with optional values for the parameters of the iterative processes: • `maxiter` The maximum number of iteration for the IWSL algorithm. Default is 20. • `conv_crit` The convergence criteria, expressed as difference between estimates at iteration i and i+1. Default is `1e-5`. • `verbose` A Boolean. Default is `FALSE`. If `TRUE` monitors the iteration process.

Value

A list with the following elements:

- `alpha` The vector of estimated P-splines coefficients of length cs.
- `SE_alpha` The vector of estimated Standard Errors for the `alpha` coefficients, of length cs.
- `H` The hat-matrix.

- Cov The full variance-covariance matrix.
- deviance The deviance.
- ed The effective dimension of the model.
- aic The value of the AIC.
- bic The value of the BIC.
- Bbases a list with the B-spline basis Bs (this is a list for compatibility with functions in 2d).

---

make_bins *Construct bins over one or more time axes*

Description

make_bins() constructs the bins over the time axes and saves the extremes of the bins in a vector.

Usage

```
make_bins(
  t_in = NULL,
  t_out = NULL,
  u = NULL,
  s_in = NULL,
  s_out,
  min_t = NULL,
  max_t = NULL,
  min_u = NULL,
  max_u = NULL,
  min_s = NULL,
  max_s = NULL,
  dt = NULL,
  du = NULL,
  ds
)
```

Arguments

t_in	(optional) A vector of entry times on the time scale t.
t_out	(optional) A vector of exit times on the time scale t.
u	(optional) A vector of fixed-times at entry in the process.
s_in	(optional) A vector of entry times on the time scale s.
s_out	A vector of exit times on the time scale s.
min_t	(optional) A minimum value for the bins over t. If NULL, the minimum of t_in will be used.
max_t	(optional) A maximum value for the bins over t. If NULL, the maximum of t_out will be used.

min_u	(optional) A minimum value for the bins over u. If NULL, the minimum of u will be used.
max_u	(optional) A maximum value for the bins over u. If NULL, the maximum of u will be used.
min_s	(optional) A minimum value for the bins over s. If NULL, the minimum of s_in will be used.
max_s	(optional) A maximum value for the bins over s. If NULL, the maximum of s_out will be used.
dt	(optional) A scalar giving the length of the intervals on the t time scale.
du	(optional) A scalar giving the length of the intervals on the u axis.
ds	A scalar giving the length of the intervals on the s time scale.

Details

It allows construction of bins over the time scales t and s and/or over the fixed-time axis u. The time scale s is always required. See also prepare_data() to conveniently prepare individual data for the analysis with one, or two time scales.

A few words about constructing the grid of bins. There is no 'golden rule' or optimal strategy for setting the number of bins over each time axis, or deciding on the bins' width. It very much depends on the data structure, however, we try to give some directions here. First, in most cases, more bins is better than less bins. A good number is about 30 bins. However, if data are scarce, the user might want to find a compromise between having a larger number of bins, and having many bins empty. Second, the chosen width of the bins (that is du and ds) does depend on the time unit over which the time scales are measured. For example, if the time is recorded in days, as in the example below, and several years of follow-up are available, the user can split the data in bins of width 30 (corresponding to about one month), 60 (about two months), 90 (about three months), etc. If the time scale is measured in years, then appropriate width could be 0.25 (corresponding to a quarter of a year), or 0.5 (that is half year). However, in some cases, time might be measure in completed years, as is often the case for age. In this scenario, an appropriate bin width is 1.

Finally, it is always a good idea to plot your data first, and explore the range of values over which the time scale(s) are recorded. This will give insight about reasonable values for the arguments min_s, min_u, max_s and max_u (that in any case are optional).

Value

A list with the following elements:

- bins_t if t_out is provided, this is a vector of bins extremes for the time scale t
- midt if t_out is provided, this is a vector with the midpoints of the bins over t
- nt if t_out is provided, this is the number of bins over t
- bins_u if u is provided, this is a vector of bins extremes for u axis
- midu if u is provided, this is a vector with the midpoints of the bins over u
- nu if u is provided, this is the number of bins over u
- bins_s is a vector of bins extremes for the time scale s
- mids is a vector with the midpoints of the bins over s
- ns is the number of bins over s

Examples

```
# Make bins for colon cancer data by time at randomization and time since recurrence
bins <- make_bins(u = reccolon2ts$timer, s_out = reccolon2ts$timesr,
                  du = 30, ds = 30)
# Make bins for colon cancer data only over time since recurrence
bins <- make_bins(s_out = reccolon2ts$timesr, ds = 60)
```

`plot.haz1ts` *Plot method for a haz1ts object.*

Description

`plot.haz1ts()` is a plot method for objects of class `haz1ts`.

Usage

```
## S3 method for class 'haz1ts'
plot(
  x,
  which_plot = c("hazard", "covariates"),
  plot_grid = NULL,
  plot_options = list(),
  ...
)
```

Arguments

`x`	The output of the function `fit1ts`.
`which_plot`	The type of plot required. Can be one of `"hazard"` (default) or `"covariates"`.
`plot_grid`	(optional) A named vector containing the parameters to build a new grid of intervals over `s` for plotting the estimated hazard on a finer grid. This must be of the form: `plot_grid = c(smin, smax, ds)`, where `smin`, `smax` are the minimum and maximum values desired for the intervals over `s`, and `ds` is the distance between intervals over `s`. If not specified, the plotting is done using the same B-splines basis as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.
`plot_options`	A list with all possible options for any of the plots: • `loghazard` A Boolean. Default is `FALSE`. If `FALSE` the function returns a plot of the hazard curve, if `TRUE` the function returns a plot of the log-hazard curve. • `log10hazard` A Boolean. Default is `FALSE`. If `TRUE` it returns a plot of the log10-hazard curve. • `col` The color of the curve plotted. Default is `"black"`.

- add_CI A Boolean. If TRUE (default) the confidence bands will be added.
- col_CI The color for the confidence bands. The default is the same color of the curve, with a 50% transparancy level.
- main The title of the plot.
- xlab The label of the time axis (plotted on the x axis).
- ylab The label of the y-axis (hazard, log-hazard or log10-hazard).
- xlim A vector with two elements defining the limits of the time scale on the x axis.
- ylim A vector with two elements defining the limits of function plotted on the y axis (hazard, log-hazard or log10-hazard).
- xmin The minimum value on the x-axis.
- ymin The minimum value on the y-axis.
- cex_main The magnification to be used for the main title, default is 1.2 .
- cex_lab The magnification to be used for the axis labels, default is 1 .
- HR A Boolean. If TRUE the HRs with their CIs will be plotted. Default is FALSE (plot the beta with their CIs).
- symmetric_CI A Boolean. Default is TRUE. If a plot of the HRs is required (HR == TRUE), then plot symmetrical Confidence Intervals, based on the SEs for the HRs calculated by delta method. If FALSE, then CIs are obtained by exponentiating the CIs for the betas.
- confidence The level of confidence for the CIs. Default is .95 (alpha = 0.05).
- col_beta The color for the plot of the covariates' effects.
- pch The symbol for plotting the point estimates.

... Further arguments to plot.

Value

A plot of the type required.

Examples

```
## preparing data - no covariates
dt1ts <- prepare_data(data = reccolon2ts,
                      s_in = "entrys",
                      s_out = "timesr",
                      events = "status",
                      ds = 180)

## fitting the model with fit1ts() - default options

mod1 <- fit1ts(dt1ts)

plot(mod1)
```

`plot.haz1tsLMM` *Plot method for a haz1ts object.*

Description

`plot.haz1tsLMM()` is a plot method for objects of class `haz1tsLMM`.

Usage

```
## S3 method for class 'haz1tsLMM'
plot(
  x,
  which_plot = c("hazard", "covariates"),
  plot_grid = NULL,
  plot_options = list(),
  ...
)
```

Arguments

`x`	The output of the function `fit1ts`.
`which_plot`	The type of plot required. Can be one of `"hazard"` (default) or `"covariates"`.
`plot_grid`	(optional) A named vector containing the parameters to build a new grid of intervals over `s` for plotting the estimated hazard on a finer grid. This must be of the form: `plot_grid = c(smin, smax, ds)`, where `smin`, `smax` are the minimum and maximum values desired for the intervals over `s`, and `ds` is the distance between intervals over `s`. If not specified, the plotting is done using the same B-splines basis as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.
`plot_options`	A list with all possible options for any of the plots: • `loghazard` A Boolean. Default is `FALSE`. If `FALSE` the function returns a plot of the hazard curve, if `TRUE` the function returns a plot of the log-hazard curve. • `log10hazard` A Boolean. Default is `FALSE`. If `TRUE` it returns a plot of the log10-hazard curve. • `col` The color of the curve plotted. Default is `"black"`. • `add_CI` A Boolean. If `TRUE` (default) the confidence bands will be added. • `col_CI` The color for the confidence bands. The default is the same color of the curve, with a 50% transparency level. • `main` The title of the plot. • `xlab` The label of the time axis (plotted on the x axis). • `ylab` The label of the y-axis (hazard, log-hazard or log10-hazard).

- `xlim` A vector with two elements defining the limits of the time scale on the x axis.
- `ylim` A vector with two elements defining the limits of function plotted on the y axis (hazard, log-hazard or log10-hazard).
- `xmin` The minimum value on the x-axis.
- `ymin` The minimum value on the y-axis.
- `cex_main` The magnification to be used for the main title, default is 1.2 .
- `cex_lab` The magnification to be used for the axis labels, default is 1 .
- `HR` A Boolean. If `TRUE` the HRs with their CIs will be plotted. Default is `FALSE` (plot the `beta` with their CIs).
- `symmetric_CI` A Boolean. Default is `TRUE`. If a plot of the HRs is required (`HR == TRUE`), then plot symmetrical Confidence Intervals, based on the SEs for the HRs calculated by delta method. If `FALSE`, then CIs are obtained by exponentiating the CIs for the betas.
- `confidence` The level of confidence for the CIs. Default is .95 (alpha = 0.05).
- `col_beta` The color for the plot of the covariates' effects.
- `pch` The symbol for plotting the point estimates.

`...` Further arguments to plot.

Details

The function `obtainSmoothTrend` from the R-package `LMMsolver` is used here. We refer the interested readers to https://biometris.github.io/LMMsolver/ for more details on `LMMsolver` and its usage.

Value

A plot of the type required.

Examples

```
## preparing data - no covariates
dt1ts <- prepare_data(data = reccolon2ts,
                      s_in = "entrys",
                      s_out = "timesr",
                      events = "status",
                      ds = 180)

## fitting the model with fit1ts() - default options

mod1 <- fit1ts(dt1ts,
optim_method = "LMMsolver")
plot(mod1)
```

plot.haz2ts *Plot method for a haz2ts object.*

Description

plot.haz2ts() is the plot method for objects of class haz2ts. It produces several kinds of plots of the fitted model with two time scales (see fit2ts()), either in the original (t,s) plane, while respecting the constraint imposed by the relation of the two time scales, or in the transformed (u,s) plane.

Usage

```
## S3 method for class 'haz2ts'
plot(
  x,
  plot_grid = NULL,
  which_plot = c("hazard", "covariates", "SE", "slices", "survival", "cumhaz"),
  where_slices = NULL,
  direction = c(NULL, "u", "s"),
  plot_options = list(),
  ...
)
```

Arguments

x	The output of the function fit2ts. This is an object of class "haz2ts".
plot_grid	(optional) A list containing the parameters to build a new finer grid of intervals over u and s for plotting. This must be of the form: plot_grid = list(c(umin, umax, du), c(smin, smax, ds)), where umin, umax and smin, smax are the minimum and maximum values desired for the intervals over u and s respectively, and du, ds are distances between intervals over u and s respectively. Specifying a new denser grid is used to evaluate the B-spline bases used for estimation on such grid and plot the estimated surfaces with a greater level of details. If not specified, the plotting is done using the same B-splines bases as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.
which_plot	The type of plot required. Can be one of "hazard" (default), "covariates", "SE", "slices", "survival" or "cumhaz" (see details section).
where_slices	A vector of values for the cutting points of the desired slices of the surface. If which_plot == "slices", please provide this argument. Please also provide this argument in case which_plot = "survival or which_plot = "cumhaz and surv_slices = TRUE or cumhaz_slices = TRUE, respectively.

`direction` If `which_plot == "slices"`, indicates the direction for cutting the surface. If `u`, then the surface will be cut at the selected values of `u` (indicated by `where_slices`), hence obtaining one-dimensional curves over `s`. If `s`, then the surface will be cut at the selected values of `s` (indicated by `where_slices`), hence obtaining one-dimensional curves over `u`.

`plot_options` A list with all possible options for any of the plots:

- `loghazard` A Boolean. Default is `FALSE`. If `FALSE` the function returns a plot of the hazard surface, if `TRUE` the function returns a plot of the log-hazard surface.
- `log10hazard` A Boolean. Default is `FALSE`. If `TRUE`, then a log_10 hazard surface is plotted.
- `cut_extrapolated` A Boolean. Default is `TRUE`. Cuts away the extrapolated area of the (log-)hazard surface before plotting.
- `rectangular_grid` A Boolean. Default is `FALSE`. If `TRUE`, a rectangular grid is used for plotting also in the (t,s)-plane as opposed to the grid of parallelograms used as default in the (t,s)-plane.
- `original` A Boolean. Default is `TRUE`. Plot the (log-)hazard (and/or the SEs) in the (t,s)-plane. If `FALSE`, the (log-)hazard (and/or the SEs) will be plotted in the (u,s)-plane.
- `tmax` The maximum value of `t` that should be plotted.
- `surv_slices` A Boolean. Default is `FALSE`. If `TRUE` and `which_plot == "survival"`, plot survival curves over the time `s` for selected values of `u`, that are cross-sections of the 2D survival surface.
- `cumhaz_slices` A Boolean. Default is `FALSE`. If `TRUE` and `which_plot == "cumhaz"`, plot cumulative hazards curves over the time `s` for selected values of `u`, that are cross-sections of the 2D cumulative hazard surface.
- `midpoints` A Boolean. Default is `FALSE`. If `TRUE`, the estimated quantities (hazard, survival, etc.) will be evaluated in the mid-points of the bins rather than at the extremes. Set to `TRUE` if plotting estimated number of events.
- `col_palette` A function defining the color palette. The default palette is `viridis::rev(plasma())`. Specifying the color palette as a function allows for greater flexibility than passing the palette as a vector.
- `n_shades` The number of color shades to plot, default is 50.
- `breaks` The vector of breaks for the color legend. If `n_shades` is provided, this should be of length `n_shades + 1`.
- `show_legend` A Boolean. Default is `TRUE`. If `FALSE` no legend will be plotted, useful for multi-panel figures with common legend. Works only for plots on rectangular grid (i.e. transformed (u,s) plane)
- `main` The title of the plot.
- `xlab` The label of the first time axis (plotted on the x axis).
- `ylab` The label of the second time axis (plotted on the y axis).
- `xlim` A vector with two elements defining the limits of the time scale on the x axis.
- `ylim` A vector with two elements defining the limits of the time scale on the y axis.

- contour_lines A Boolean. Default is FALSE. If TRUE white contour lines are added to the surfaces.
- contour_col The color for the contour lines. Default is white.
- contour_cex The magnification to be used for the contour lines. Default is .8.
- contour_nlev The number of contour levels desired. Default is 10.
- cex_main The magnification to be used for the main title, default is 1.2 .
- cex_lab The magnification to be used for the axis labels, default is 1 .
- HR A Boolean. If TRUE the HRs with their CIs will be plotted. Default is FALSE (plot the beta with their CIs).
- symmetric_CI A Boolean. Default is TRUE. If a plot of the HRs is required (HR == TRUE), then plot symmetrical Confidence Intervals, based on the SEs for the HRs calculated by delta method. If FALSE, then CIs are obtained by exponentiating the CIs for the betas.
- confidence The level of confidence for the CIs. Default is .95 (alpha = 0.05).
- col_beta The color for the plot of the covariates' effects.
- pch The symbol for plotting the point estimates.
- lwd The line width.

... Further arguments to image.plot or image

Details

The vignette "visualization" presents and discusses all the different plotting options for the fitted model over two time scales. In most of the cases, the user will want to visualize the hazard surface over the two time scales. This can be plotted on the hazard scale, the log-hazard scale or the log10-hazard scale, by switching to TRUE the corresponding argument in plot_options. The survival and cumulative hazard functions can be plotted as two-dimensional surfaces over u and s or t and s. However, it is also very informative to plot them as one-dimensional curves over s (cross-sections or slices). This is done by selecting which_plot = "survival" and surv_slices = TRUE in plot_options. Additionally, a vector of values for the cutting points over the u-axis should be passed to the argument where_slices, together with setting direction = u. Similar plot is obtained for the cumulative hazard by selecting which_plot = "cumhaz", cumhaz_slices = TRUE, see examples section. Please, notice that for the survival function and the cumulative hazard, only cross-sections of the surface for selected values of u (over the s time) can be plotted.

Value

A plot of the fitted model.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
```

```
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(u = fakedata$u,
                            s_out = fakedata$s,
                            ev = fakedata$ev,
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
                  lrho = list(seq(1 ,1.5 ,.5),
                              seq(1 ,1.5 ,.5)))

 # plot the hazard surface
 plot(fakemod)

 # plot the survival function as one-dimension curves over `s`
 plot(fakemod,
      which_plot = "survival",
      direction = "u",
      where_slices = c(4, 6, 8),
      plot_options = list(
      surv_slices = TRUE
      ))

# Plot cross-sections of the hazard over `s` for selected values of `u`

plot(fakemod,
  which_plot = "slices",
  where_slices = c(4, 6, 8),
  direction = "u",
  plot_options = list(
    main = "Cross-sections of the hazard",
    xlab = "Time",
    ylab = "Hazard"
  )
)
```

`plot.haz2tsLMM` *Plot method for a haz2tsLMM object.*

Description

`plot.haz2tsLMM()` is the plot method for objects of class `haz2tsLMM`. It produces plots of the fitted model with two time scales (see `fit2ts()`), fitted via LMMsolver. The two-dimensional plots are limited to the transformed plane, that is only plots over `u` and `s` axes are produced.

Usage

```
## S3 method for class 'haz2tsLMM'
plot(
  x,
  plot_grid = NULL,
  which_plot = c("hazard", "covariates", "SE", "slices", "survival", "cumhaz"),
  where_slices = NULL,
  direction = c(NULL, "u", "s"),
  plot_options = list(),
  ...
)
```

Arguments

x	The output of the function fit2ts. This is an object of class "haz2tsLMM".
plot_grid	A list containing the parameters to build a new finer grid of intervals over u and s for plotting. This must be of the form: plot_grid = list(c(umin, umax, du), c(smin, smax, ds)), where umin, umax and smin, smax are the minimum and maximum values desired for the intervals over u and s respectively, and du, ds are distances between intervals over u and s respectively. Specifying a new denser grid is used to evaluate the B-spline bases used for estimation on such grid and plot the estimated surfaces with a greater level of details. If not specified, the plotting is done using the same B-splines bases as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned. While for objects of class 'haz2ts' this is an optional input, we strongly recommend to provide the plotting grid for object of class 'haz2tsLMM', given that evaluation of the B-splines works a bit differently in LMMsolver.
which_plot	The type of plot required. Can be one of "hazard" (default), "covariates", "SE", "slices", "survival" or "cumhaz" (see details section).
where_slices	A vector of values for the cutting points of the desired slices of the surface. If which_plot == "slices", please provide this argument. Please also provide this argument in case which_plot = "survival or which_plot = "cumhaz and surv_slices = TRUE or cumhaz_slices = TRUE, respectively.
direction	If which_plot == "slices", indicates the direction for cutting the surface. If u, then the surface will be cut at the selected values of u (indicated by where_slices), hence obtaining one-dimensional curves over s. If s, then the surface will be cut at the selected values of s (indicated by where_slices), hence obtaining one-dimensional curves over u.
plot_options	A list with all possible options for any of the plots: • loghazard A Boolean. Default is FALSE. If FALSE the function returns a plot of the hazard surface, if TRUE the function returns a plot of the log-hazard surface. • log10hazard A Boolean. Default is FALSE. If TRUE, then a log10 hazard surface is plotted.

- cut_extrapolated A Boolean. Default is TRUE. Cuts away the extrapolated area of the (log-)hazard surface before plotting.
- tmax The maximum value of t that should be plotted.
- surv_slices A Boolean. Default is FALSE. If TRUE and which_plot == "survival", plot survival curves over the time s for selected values of u, that are cross-sections of the 2D survival surface.
- cumhaz_slices A Boolean. Default is FALSE. If TRUE and which_plot == "cumhaz", plot cumulative hazards curves over the time s for selected values of u, that are cross-sections of the 2D cumulative hazard surface.
- midpoints A Boolean. Default is FALSE. If TRUE, the estimated quantities (hazard, survival, etc.) will be evaluated in the mid-points of the bins rather than at the extremes. Set to TRUE if plotting estimated number of events.
- col_palette A function defining the color palette. The default palette is viridis::rev(plasma()). Specifying the color palette as a function allows for greater flexibility than passing the palette as a vector. We provide an example on how to create a function from any color palette below.
- n_shades The number of color shades to plot, default is 50.
- breaks The vector of breaks for the color legend. If n_shades is provided, this should be of length n_shades + 1.
- show_legend A Boolean. Default is TRUE. If FALSE no legend will be plotted, useful for multi-panel figures with common legend. Works only for plots on rectangular grid (i.e. transformed (u,s) plane).
- main The title of the plot.
- xlab The label of the first time axis (plotted on the x axis).
- ylab The label of the second time axis (plotted on the y axis).
- xlim A vector with two elements defining the limits of the time scale on the x axis.
- ylim A vector with two elements defining the limits of the time scale on the y axis.
- contour_lines A Boolean. Default is FALSE. If TRUE contour lines are added to the surfaces.
- contour_col The color for the contour lines. Default is white.
- contour_cex The magnification to be used for the contour lines. Default is .8.
- contour_nlev The number of contour levels desired. Default is 10.
- cex_main The magnification to be used for the main title, default is 1.2 .
- cex_lab The magnification to be used for the axis labels, default is 1 .
- HR A Boolean. If TRUE the HRs with their CIs will be plotted. Default is FALSE (plot the beta with their CIs).
- symmetric_CI A Boolean. Default is TRUE. If a plot of the HRs is required (HR == TRUE), then plot symmetrical Confidence Intervals, based on the SEs for the HRs calculated by delta method. If FALSE, then CIs are obtained by exponentiating the CIs for the betas.
- confidence The level of confidence for the CIs. Default is .95 (alpha = 0.05).

- col_beta The color for the plot of the covariates' effects.
- pch The symbol for plotting the point estimates.
- lwd The line width.

... Further arguments to image.plot or image

Details

The vignette "visualization" presents and discusses all the different plotting options for the fitted model over two time scales. In most of the cases, the user will want to visualize the hazard surface over the two time scales. This can be plotted on the hazard scale, the log-hazard scale or the log10-hazard scale, by switching to TRUE the corresponding argument in plot_options. The survival and cumulative hazard functions can be plotted as two-dimensional surfaces over u and s or t and s. However, it is also very informative to plot them as one-dimensional curves over s (cross-sections or slices). This is done by selecting which_plot = "survival" and surv_slices = TRUE in plot_options. Additionally, a vector of values for the cutting points over the u-axis should be passed to the argument where_slices, together with setting direction = u. Similar plot is obtained for the cumulative hazard by selecting which_plot = "cumhaz", cumhaz_slices = TRUE, see examples section. Please, notice that for the survival function and the cumulative hazard, only cross-sections of the surface for selected values of u (over the s time) can be plotted.

The function obtainSmoothTrend from the R-package LMMsolver is used here. We refer the interested readers to https://biometris.github.io/LMMsolver/ for more details on LMMsolver and its usage.

Value

A plot of the fitted model.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)#'

fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(u = fakedata$u,
                            s_out = fakedata$s,
                            ev = fakedata$ev,
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "LMMsolver")

 # plot the hazard surface
 plot(fakemod)
 # plot the survival function as one-dimension curves over `s`
 plot(fakemod,
```

```
      which_plot = "survival",
      direction = "u",
      where_slices = c(4, 6, 8),
      plot_options = list(
      surv_slices = TRUE
      ))


# Plot cross-sections of the hazard over `s` for selected values of `u`

plot(fakemod,
  which_plot = "slices",
  where_slices = c(4, 6, 8),
  direction = "u",
  plot_options = list(
    main = "Cross-sections of the hazard",
    xlab = "Time",
    ylab = "Hazard"
  )
)
```

`plot_slices` *Plot slices of the (log-) hazard*

Description

`plot_slices()` plots slices of the (log-)hazard with two time scales, at selected values of one of the two time dimensions.

Usage

```
plot_slices(x, y, direction, plot_options = list())
```

Arguments

`x`	A vector of values for the x-axis. This is a vector of values over the axis opposite to the one where the sliced are cut.
`y`	A matrix of (log-)hazard values.
`direction`	Either `"u"` or `"s"`.
`plot_options`	A list of options for the plot: • `loghazard` A Boolean. Default is `FALSE`. If `FALSE` the function returns a plot of cross-sections from the hazard surface, if `TRUE` the function returns a plot of cross-sections from the log-hazard surface. • `log10hazard` A Boolean. Default is `FALSE`. If `TRUE` returns a plot of cross-sections from the log10-hazard surface.

- `col_palette` A function defining the color palette. The default palette is `grDevices::gray.colors()`.
- `n_shades` The number of color shades to plot, default is 50.
- `main` The title of the plot.
- `xlab` The label of the first time axis (plotted on the x axis).
- `ylab` The label of the second time axis (plotted on the y axis).
- `xlim` A vector with two elements defining the limits of the time scale on the x axis.
- `ylim` A vector with two elements defining the limits of the time scale on the y axis.
- `cex_main` The magnification to be used for the main title, default is `1.2`.
- `cex_lab` The magnification to be used for the axis labels, default is `1`.
- `lwd` The line's width. Default is `2`.

Value

A plot of the slices of the hazard cut at selected points.

`prepare_data` *Prepare raw data by binning them in 1d or 2d*

Description

`prepare_data()` prepares the raw individual time-to-event data for hazard estimation in 1d or 2d.

Given the raw data, this function first constructs the bins over one or two time axes and then computes the aggregated (or individual) vectors or matrices of exposure times and events indicators. A data.frame with covariates values can be provided by the user.

Usage

```
prepare_data(
  data = NULL,
  t_in = NULL,
  t_out = NULL,
  u = NULL,
  s_in = NULL,
  s_out,
  events,
  min_t = NULL,
  max_t = NULL,
  min_u = NULL,
  max_u = NULL,
  min_s = NULL,
  max_s = NULL,
  dt = NULL,
```

```
  du = NULL,
  ds,
  individual = FALSE,
  covs = NULL
)
```

Arguments

data	A data frame.
t_in	(optional) A vector of entry times on the time scale t.
t_out	(optional) A vector of exit times on the time scale t.
u	(optional) A vector of fixed-times at entry in the process.
s_in	(optional) A vector of entry times on the time scale s.
s_out	A vector of exit times on the time scale s.
events	A vector of event's indicators (possible values 0/1).
min_t	(optional) A minimum value for the bins over t. If NULL, the minimum of t_in will be used.
max_t	(optional) A maximum value for the bins over t. If NULL, the maximum of t_out will be used.
min_u	(optional) A minimum value for the bins over u. If NULL, the minimum of u will be used.
max_u	(optional) A maximum value for the bins over u. If NULL, the maximum of u will be used.
min_s	(optional) A minimum value for the bins over s. If NULL, the minimum of s_in will be used.
max_s	(optional) A maximum value for the bins over s. If NULL, the maximum of s_out will be used.
dt	(optional) A scalar giving the length of the intervals on the t time scale.
du	(optional) A scalar giving the length of the intervals on the u axis.
ds	A scalar giving the length of the intervals on the s time scale.
individual	A Boolean. Default is FALSE: if FALSE computes the matrices R and Y collectively for all observations; if TRUE computes the matrices R and Y separately for each individual record.
covs	A data.frame with the variables to be used as covariates. The function will create dummy variables for any factor variable passed as argument in covs. If a variable of class character is passed as argument, it will be converted to factor.

Details

A few words about constructing the grid of bins. Bins are containers for the individual data. There is no 'golden rule' or optimal strategy for setting the number of bins over each time axis, or deciding on the bins' width. It very much depends on the data structure, however, we try to give some directions here. First, in most cases, more bins is better than less bins. A good number is about 30 bins. However, if data are scarce, the user might want to find a compromise between having a larger

number of bins, and having many bins empty. Second, the chosen width of the bins (that is du and ds) does depend on the time unit over which the time scales are measured. For example, if the time is recorded in days, as in the example below, and several years of follow-up are available, the user can split the data in bins of width 30 (corresponding to about one month), 60 (about two months), 90 (about three months), etc. If the time scale is measured in years, then appropriate width could be 0.25 (corresponding to a quarter of a year), or 0.5 (that is half year). However, in some cases, time might be measure in completed years, as is often the case for age. In this scenario, an appropriate bin width is 1.

Finally, it is always a good idea to plot the data first, and explore the range of values over which the time scale(s) are recorded. This will give insight about reasonable values for the arguments min_s, min_u, max_s and max_u (that in any case are optional).

Regarding names of covariates or levels of categorical covariates/factors: When using "LMMsolver" to fit a model with covariates that which have names (or factor labels) including a symbol such as "+", "-", "<" or ">" will result in an error. To avoid this, the responsible names (labels) will be rewritten without mathematical symbols. For example: "Lev+5FU" (in the colon cancer data) is replaced by "Lev&5FU".

Value

A list with the following elements:

- bins a list:
  - bins_t if t_out is provided, this is a vector of bins extremes for the time scale t.
  - mid_t if t_out is provided, this is a vector with the midpoints of the bins over t.
  - nt if t_out is provided, this is the number of bins over t.
  - bins_u if u is provided, this is a vector of bins extremes for u axis.
  - midu if u is provided, this is a vector with the midpoints of the bins over u.
  - nu if u is provided, this is the number of bins over u.
  - bins_s is a vector of bins extremes for the time scale s.
  - mids is a vector with the midpoints of the bins over s.
  - ns is the number of bins over s.
- bindata:
  - r or R an array of exposure times: if binning the data over one time scale only this is a vector. If binning the data over two time scales and if individual == TRUE then R is an array of dimension nu by ns by n, otherwise it is an array of dimension nu by ns
  - y or Y an array of event counts: if binning the data over one time scale only this is a vector. If binning the data over two time scales and if individual == TRUE then Y is an array of dimension nu by ns by n, otherwise it is an array of dimension nu by ns
  - Z A matrix of covariates' values to be used in the model, of dimension n by p

Author(s)

Angela Carollo <carollo@demogr.mpg.de>

Examples

```
# Bin data over s = time since recurrence only, with intervals of length 30 days
# aggregated data (no covariates)
# The following example provide the vectors of data directly from the dataset
binned_data <- prepare_data(s_out = reccolon2ts$timesr, events = reccolon2ts$status, ds = 30)
# Visualize vector of event counts
print(binned_data$bindata$y)
# Visualize midpoints of the bins
print(binned_data$bins$mids)
# Visualize number of bins
print(binned_data$bins$ns)

# Now, the same thing is done by providing a dataset and the name of all relevant variables
binned_data <- prepare_data(data = reccolon2ts, s_out = "timesr", events = "status", ds = 30)
# Visualize vector of event counts
print(binned_data$bindata$y)

# Now using ds = .3 and the same variable measured in years
binned_data <- prepare_data(s_out = reccolon2ts$timesr_y, events = reccolon2ts$status, ds = .3)
# Visualize vector of exposure timess
print(binned_data$bindata$r)


# Bin data over u = time at recurrence and s = time since recurrence, measured in days
# aggregated data (no covariates)
# Note that if we do not provide du this is taken to be equal to ds
binned_data <- prepare_data(
  u = reccolon2ts$timer, s_out = reccolon2ts$timesr,
  events = reccolon2ts$status, ds = 30
)

# Visualize matrix of event counts
print(binned_data$bindata$Y)

# Visualize midpoints of bins over u
print(binned_data$bins$midu)


# Bin data over u = time at recurrence and s = time since recurrence, measured in day
# individual-level data required
# we provide two covariates: nodes (numerical) and rx (factor)
covs <- subset(reccolon2ts, select = c("nodes", "rx"))
binned_data <- prepare_data(
  u = reccolon2ts$timer, s_out = reccolon2ts$timesr,
  events = reccolon2ts$status, ds = 30, individual = TRUE, covs = covs
)

# Visualize structure of binned data
print(str(binned_data$bindata))

# Alternatevely:
binned_data <- prepare_data(
```

```
  data = reccolon2ts,
  u = "timer", s_out = "timesr",
  events = "status", ds = 30, individual = TRUE, covs = c("nodes", "rx")
)
```

`prepare_data_LMMsolver` *Process data to fit model with LMMsolver*

Description

Process data to fit model with LMMsolver

Usage

```
prepare_data_LMMsolver(Y = Y, R = R, Z = NULL, bins = bins)
```

Arguments

`Y`	A matrix (or 3d-array) of event counts of dimension nu by ns (or nu by ns by n).
`R`	A matrix (or 3d-array) of exposure times of dimension nu by ns (or nu by ns by n).
`Z`	(optional) A regression matrix of covariates values of dimensions n by p.
`bins`	a list with the specification for the bins. This is created by the function `prepare_data`. If a list prepared externally from such function if provided, it should contain the following elements: * `bins_u` A vector of bins extremes for the time scale `u`. * `midu` A vector with the midpoints of the bins over `u`. * `nu` The number of bins over `u`. * `bins_s` A vector of bins extremes for the time scale `s`. * `mids` A vector with the midpoints of the bins over `s`. * `ns` The number of bins over `s`.

Value

A dataset in long form to fit the model with LMMsolver

`print.data2ts` *Print method for a* `data2ts` *object*

Description

Print method for an object of class `data2ts`

Usage

```
## S3 method for class 'data2ts'
print(x, ...)
```

Arguments

x	of class data2ts, as prepared by prepare_data
...	Further arguments to print

Value

No return value

Author(s)

Angela Carollo <carollo@demogr.mpg.de>

Examples

```
# Bin the colon cancer data over s (time since recurrence)

dt1ts <- prepare_data(data = reccolon2ts,
                      s_in = "entrys",
                      s_out = "timesr",
                      events = "status",
                      ds = 180)

print(dt1ts)

# Bin the colon cancer data over u (time at recurrence) and s (time since recurrence)
dt2ts <- prepare_data(data = reccolon2ts,
                      u = "timer",
                      s_in = "entrys",
                      s_out = "timesr",
                      events = "status",
                      ds = 180)
print(dt2ts)
```

---

reccolon2ts *Data from the chemotherapy for stace B/C colon cancer study*

Description

This dataset is a reduced version of the dataset colon from the package survival (Therneau, T.M. et al., 2022). Each observation is a transition from recurrence of colon cancer to death or censoring. The time scales are time from randomization to recurrence, time from randomization to death or censoring and time from recurrence of the cancer to death or censoring. Only observations about individuals with a recurrence of the cancer are selected. Additionally, 7 individuals with exit times from the risk set equal to entry times in the recurrence state (0 exposure time) were dropped from the sample. In the original dataset, all times of recurrence are known precisely, so that after recurrence all individuals are followed right from entry in the state, without left truncation. To be able to illustrate how to include left truncated times in the model, artificial left truncated entry in the 'recurrence' state are introduced for 40 individuals.

Usage

```
data(reccolon2ts)
```

Format

reccolon2ts A data.table with 461 rows and 25 columns::

id patient's id

study 1 for all patients

rx Treatment - Obs(ervation), Lev(amisole), Lev(amisole)+5-FU

sex 1=male, 0=female

age Age at transplant in years

obstruct obstruction of colon by tumor: 1=yes

perfor perforation of colon: 1=yes

adhere adherence to nearby organs: 1=yes

nodes number of lymph nodes with detectable cancer

status censoring status: 0=censoring, 1=event

differ differentiation of tumour: 1=well, 2=moderate, 3=poor

extent extent of local spread: 1=submucosa, 2=muscle, 3=serosa, 4=contigous structures

surg time from surgery to registration: 0=short, 1=long

node4 more than 4 positive lymph nodes

etype 2 for all patients (2=death)

timedc time in days from randomization to death or censoring

timer time in days from randomization to recurrence

timesr time in days from recurrence to death or censoring

entrys artificial entry time on the time since recurrence scale. For most of the individual this is 0 (no left truncation). For 40 individuals a random number between 1 and the exit time on the time since recurrence scale (timesr) is simulated.

entryt time in days from randomization to observation in the recurrence state. If the individual is observed from entry in the recurrence state this is equal to the time at recurrence. If the entry in the recurrence state is not observed from the beginning, left truncation is observed. This is not present in the original data, but has been here introduced artificially for 40 individuals. This is done by first increasing the time at recurrence by a random number between 1 and the exit time on the time since recurrence scale. Then, the time at recurrence is added to the artificial entry time.

timedc_y time in years from randomization to death or censoring

timer_y time in years from randomization to recurrence

entrys_y left truncated entry in the recurrence state measured in years since recurrence

entryt_y left truncated entry in the recurrence state measured in years since randomization

timesr_y time in years from recurrence to death or censoring

Source

Therneau, T. (2023). A Package for Survival Analysis in R. R package version 3.5-3, https://CRAN.R-project.org/package=survival

References

Moertel, C.G, et al. (1995). Fluorouracil plus Levamisole as Effective Adjuvant Theraphy after Resection of Stage III Colon Carcinoma: A Final Report. Annals of Internal Medicine, 122:321-326

Moerel, C.G., et al. (1990). Levamisole and Fluorouracil for Adjvant Theraphy of Resected Colon Carcinoma. The New England Journal of Medicine, 322:352-8

Examples

```
data(reccolon2ts)
rm(reccolon2ts)
```

`surv2ts` *Survival function with two time scales*

Description

Computes the survival matrix, that contains the probability of not experiencing an event (of any cause) by time `s` and fixed entry time `u`. The survival function can be obtained from one fitted model with only one event type, or combining information from several cause-specific hazard in a competing risks model. In the first case, a fitted object of class `'haz2ts'` or `'haz2tsLMM'` can be passed directly as argument to the function. In the competing risks framework, the user should provide a list of cause-specific cumulative hazard matrices. The function is also called internally from `plot()` if the user wants to plot the cumulative hazard from a fitted model.

Usage

```
surv2ts(
  cumhaz = NULL,
  fitted_model = NULL,
  plot_grid = NULL,
  cause = NULL,
  midpoints = FALSE,
  where_slices = NULL,
  direction = c("u", "s", NULL),
  tmax = NULL
)
```

Arguments

`cumhaz`	(optional) a list with all the cause-specific cumulated hazard matrices (minimum one element needs to be supplied). If more than one cause-specific cumulated hazard is provided, then they should all be matrices of the same dimension.
`fitted_model`	(optional) The output of the function `fit2ts`. This is an object of class `'haz2ts'` or `'haz2tsLMM'`.

plot_grid (optional) A list containing the parameters to build a new finer grid of intervals over u and s for plotting. This must be of the form:

- plot_grid = list(c(umin, umax, du), c(smin, smax, ds)) where umin, umax and smin, smax are the minimum and maximum values desired for the grid-points over u and s respectively, and du, ds are distances between points over u and s respectively. Specifying a new denser grid is used to evaluate the B-spline bases used for estimation on such grid and plot the estimated surfaces with a greater level of details. If not specified, the plotting is done using the same B-splines bases as for the estimation. The function will check if the parameters for the grid provided by the user are compatible with those originally used to construct the B-splines for estimating the model. If not, the grid will be adjusted accordingly and a warning will be returned.

cause a character string with a short name for the cause (optional).

midpoints A Boolean. Default is FALSE. If TRUE, the estimated quantities are evaluated at the midpoints of the rectangles (or parallelograms) of the grids, rather than at each grid-point.

where_slices A vector of values for the cutting points of the desired slices of the surface. This option is included mostly for the plotting function. When using plot.haz2ts(), the user selects which_plot = "survival" and surv_slices = TRUE, then where_slices indicates the location of the cutting points over the u time.

direction If cross-sectional one-dimensional curves are plotted, this indicates whether the cutting points are located on the u time, or on the s time. For plots of the survival function, only cutting points over the u time are meaningful.

tmax The maximum value of t that should be plotted.

Value

a matrix containing the values of the survival function over s and u.

Examples

```
# Create some fake data - the bare minimum
id <- 1:20
u <- c(5.43, 3.25, 8.15, 5.53, 7.28, 6.61, 5.91, 4.94, 4.25, 3.86, 4.05, 6.86,
       4.94, 4.46, 2.14, 7.56, 5.55, 7.60, 6.46, 4.96)
s <- c(0.44, 4.89, 0.92, 1.81, 2.02, 1.55, 3.16, 6.36, 0.66, 2.02, 1.22, 3.96,
       7.07, 2.91, 3.38, 2.36, 1.74, 0.06, 5.76, 3.00)
ev <- c(1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1)


fakedata <- as.data.frame(cbind(id, u, s, ev))
fakedata2ts <- prepare_data(u = fakedata$u,
                            s_out = fakedata$s,
                            ev = fakedata$ev,
                            ds = .5)
# Fit a fake model - not optimal smoothing
fakemod <- fit2ts(fakedata2ts,
                  optim_method = "grid_search",
```

```
                    lrho = list(seq(1 , 1.5, .5),
                                seq(1 , 1.5, .5)))

# Obtain the fake cumulated hazard
fakecumhaz2ts <- cumhaz2ts(fakemod)
# Fake survival curve
fakesurv2ts <- surv2ts(fitted_model = fakemod)
```

Index