Package 'TimeDepFrail'

January 17, 2025

Type Package

Title Time Dependent Shared Frailty Cox Model

Version 0.0.1

Description Fits time-dependent shared frailty Cox model (specifically the adapted Paik et al.'s Model) based on the paper ``Centre-Effect on Survival After Bone Marrow Transplantation: Application of Time-Dependent Frailty Models", by C.M. Wintrebert, H. Putter, A.H. Zwinderman and J.C. van Houwelingen (2004) <doi:10.1002/bimj.200310051>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.5.0)

LazyData true

URL https://alessandragni.github.io/TimeDepFrail/

NeedsCompilation no

Author Alessandra Ragni [aut, cre] (<https://orcid.org/0000-0002-3647-7340>),
Giulia Romani [aut],
Chiara Masci [aut] (<https://orcid.org/0000-0002-9208-3194>)

Maintainer Alessandra Ragni <alessandra.ragni@polimi.it>

Repository CRAN

Date/Publication 2025-01-17 20:10:02 UTC

Contents

AdPaikModel *Adapted Paik et al.'s Model: Time-Dependent Shared Frailty Cox Model*

Description

Function for applying the 'Adapted Paik et al.'s Model', an innovative Cox Model with time-dependent frailty, shared by individuals belonging to the same group/cluster.

To generate time-dependence, the temporal domain must be divided into a certain number of intervals. For more information about the time-domain, its relationship with the follow-up and its internal subdivision refer to Details.

The model log-likelihood function depends on a certain number of parameters and it is maximized with respect to all of them, using a reinterpretation of the 'Powell's method in multidimension', that is a multi-dimensional optimization method based on repeated one-dimensional optimization of the log-likelihood function (with respect to one parameter at the time). In this context, the one-dimensional optimization is performed through the 'optimize' R function. For more information about the unknown model parameters, their type and numerosity refer to Details.

Several quantities are estimated at the end of the optimization phase:

- parameters, their standard error and confidence interval;
- baseline hazard;
- frailty dispersion (standard deviation and variance);
- posterior frailty estimates, their variance and confidence interval;
- Akaike Information Criterion (AIC).

Usage

```
AdPaikModel(
  formula,
  data,
  time_axis,
  categories_range_min,
  categories_range_max,
  flag_fullsd = TRUE,
  n_extrarun = 60,
  tol_ll = 1e-06,
  tol_optimize = 1e-06,
  h_dd = 0.001,
  print_previous_ll_values = c(TRUE, 3),
  level = 0.95,
  verbose = FALSE
)
```

Arguments

formula	Formula object having on the left hand side the @time_to_event variable, that is the time-instant in which the individual failed. On the right hand side, it has the regressors and the cluster variable.
data	Dataset in which all variables of the formula object must be found and contained. This dataset can also contain other variables, but they will not be considered. It can be either a dataframe or a matrix, but in both cases the name of each column must correspond to the name of the formula variables. It is not necessary to attach it (in case of a data.frame)
time_axis	Temporal domain
categories_range_min	Vector containing the minimum value assumable by each parameter category.
categories_range_max	Vector containing the maximum value assumable by each parameter category.
flag_fullsd	Logical. If TRUE, the full frailty standard deviation is computed, otherwise the partial one that keeps into account only the time-dependent component. Defaults to TRUE.
n_extrarun	Total number of runs (iterations) are obtained summing to the number of parameters and n_extrarun.
tol_ll	Tolerance on the log-likelihood value.
tol_optimize	Internal tolerance for the one-dimensional optimization through 'optimize' R function.
h_dd	Discretization step used for the numerical approximation of the second derivative of the log-likelihood function.
print_previous_ll_values	If we want to print the previous values of the log-likelihood function. This can be useful for controlling that the optimization procedure is proceeding in a monotone way and it does not oscillate. This argument is composed of two elements: TRUE/FALSE if we want or not to print the previous values and how many values we want to print on the console. Default is (TRUE, 3), so that only the previous 3 values of the log-likelihood are printed.
level	A numeric value representing the confidence level for the optimal parameters (default is 0.95 for 95% confidence).
verbose	Logical. If TRUE, detailed progress messages will be printed to the console. Defaults to FALSE.

Details

Two observation needs to made about the time-domain:

- The time domain may coincide with the follow-up or it may be contained in it. Indeed, the left boundary can be greater than the beginning of the follow-up and, for instance, it can coincide with the time-instants in which the events begin to happen; conversely, the right boundary of the two must be the same.

- The partition of the time domain into intervals can be made according to two selected criteria: (1) using an already existent partition of the follow-up (2) using the shape of the baseline hazard function as reference: divide the time-domain according to regions in which it has a peak or a plateau.

The parameters with respect to which the log-likelihood function must be optimized are:

- baseline log-hazard (number of parameters = number of intervals of the time-domain)
- data regressors
- μ_1, ν: parameters of the gamma distribution of α_j (time-independent/constant) (2 parameters)
- γ_k: parameters of the gamma distribution of ϵ_{jk} (time-dependent) (number of parameters = number of intervals) Another model parameter is μ_2 and it is get imposing the constraint that $\mu_1 + \mu_2 = 1$. As it can be notice, some parameters can be grouped into the same category (regressors, baseline log-hazard and so on) and we can easily constraint them assigning each category both a minimum and maximum range. The category vector is structured as follows: (baseline log-hazard, regressors, μ_1, ν, γ_k) with dimension (n_intervals, n_regressors, 1, 1, n_intervals).

The output of the model call 'AdPaikModel(...)' is a S3 object of class 'AdPaik', composed of the following quantities:

- formula: formula object provided in input by the user and specifying the relationship between the time-to-event, the covariates of the dataset (regressors) and the cluster variable.
- Regressors: categorical vector of length R, with the name of the regressors. They could be different from the original covariates of the dataset in case of categorical covariates. Indeed, each categorical covariate with n levels needs to be transformed into (n-1) dummy variables and, therefore, (n-1) new regressors.
- NRegressors: number of regressors (R)
- ClusterVariable: name of the variable with respect to which the individuals can be grouped.
- NClusters: number of clusters/centres (also indicated with N).
- NIntervals: number of intervals of the time-domain, also called with L.
- NParameters: number of parameters of the model. It can be computed as: $n_p = 2L + R + 2$.
- ParametersCategories: Numerical vector of length 5, containing the numerosity of each parameter category.
- ParametersRange: S3 object of class 'ParametersRange', containing ParametersRangeMin and ParametersRangeMax, two numerical vectors of length n_p, giving the minimum and the maximum range of each parameter, respectively.
- Loglikelihood: value of the maximized log-likelihood function, at the optimal estimated parameters.
- AIC: 'Akaike Information Criterion': it can be computed as $AIC = 2n_p - 2ll_{optimal}$. It quantifies the loss of information related to the model fitting and output. The smaller, the less the loss of information and the better the model accuracy.
- Status: Logical value. TRUE if the model reaches convergence, FALSE otherwise.
- NRun: Number of runs necessary to reach convergence. If the model does not reach convergence, such number is equal to the maximum number of imposed runs.

- OptimalParameters: numerical vector of length n_p, containing the optimal estimated parameters (the parameters that maximize the log-likelihood function).
- StandardErrorParameters: numerical vector of length n_p, corresponding to the standard error of each estimated parameters.
- ParametersCI: S3 object of class 'ParametersCI', composed of two numerical vector of length equal to n_p: the left and right 95\ interval of each estimated parameter of given level.
- BaselineHazard: numerical vector of length equal to L, containing the baseline hazard step-function.
- FrailtyDispersion: S3 object of class 'FrailtyDispersion', containing two numerical vectors of length equal to L with the standard deviation and the variance of the frailty. numerical vector of length equal to L (i.e. number of intervals of the time-domain), reporting the standard deviation of the frailty.
- PosteriorFrailtyEstimates: S3 object of class 'PFE.AdPaik'. See details.
- PosteriorFrailtyVariance: S3 object of class 'PFV.AdPaik'. See details.
- PosteriorFrailtyCI: S3 object of class 'PFCI.AdPaik'. See details.

The object of class 'PFE.AdPaik' contains the Posterior Frailty Estimates computed with the procedure indicated in the reference paper and it is composed of three elements:

- 'alpha': posterior frailty estimates for $\alpha_j, \forall j$. It is a vector of length equal to the number of centres.
- 'eps': posterior frailty estimates for $\epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).
- 'Z': posterior frailty estimates for $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).

The object of class 'PFV.AdPaik' contains the Posterior Frailty Variances computed as indicated in the reference papaer and it is composed of three elements:

- 'alphaVar': posterior frailty variance for $\alpha_j, \forall j$. It is a vector of length equal to the number of centres.
- 'epsVar': posterior frailty variance for $\epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).
- 'ZVar': posterior frailty variance for $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).

The object of class 'PFCI.AdPaik' contains the Posterior Frailty Confidence Interval and it is composed of two elements:

- left confidence interval for the estimated $\hat{Z}_{jk}, \forall j, k$. Matrix of dimension (N, L).
- right confidence interval for the estimated $\hat{Z}_{jk}, \forall j, k$. Matrix of dimension (N, L).

Value

S3 object of class 'AdPaik', composed of several elements. See Details.

Source

...

Examples

```
# Consider the 'Academic Dropout dataset'
data(data_dropout)

# Define the variables needed for the model execution
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0, 1 - eps, 1, 10)


# Call the main model
result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)
```

AdPaik_1D *One-dimensional analysis of log-likelihood function*

Description

Function for studying the log-likelihood function from the point of view of a single parameter and, therefore, in a single direction. It performs both the optimization of the log-likelihood with respect to this parameter and the evaluation of the log-likelihood in several samples of the same parameter, while the other parameters can assume a constant assigned value or can vary in their range.

Usage

```
AdPaik_1D(
  formula,
  data,
  time_axis,
  index_param_to_vary,
  flag_optimal_params = FALSE,
  optimal_params = NULL,
  categories_range_min,
  categories_range_max,
  n_iter = 5,
  tol_optimize = 1e-06,
  flag_plot = FALSE,
  n_points = 150,
  cex = 0.7,
  cex_max = 0.8,
  color_bg = "black",
  color_max_bg = "red",
  pch = 21
)
```

Arguments

formula	Formula object indicating the response variable, the covariates and the cluster variable.
data	Dataset in which the variables of the formula object are located.
time_axis	Partitioned time-domain.
index_param_to_vary	Index of the parameter, in the parameter vector, with respect to which the log-likelihood function is maximized in a one-dimensional way. The index s provided to identify the parameter under consideration inside the vector, avoiding providing its name or value.
flag_optimal_params	Are the other parameters extracted from the optimal vector of parameters? If so, the flag should be equal to TRUE. Otherwise, the flag is equal to FALSE.
optimal_params	Vector of optimal parameters, determined through an entire multi-dimensional maximization of the log-likelihood function. The default value (NULL) indicates that no vector is provided and the parameters are randomly extracted in their range.
categories_range_min	Vector containing the minimum value assumed by each parameter category.
categories_range_max	Vector containing the maximum value assumed by each parameter category.
n_iter	Number of times the one-dimensional analysis with respect to the indicated parameter must be executed. Default value is 5. See details for more information.
tol_optimize	Tolerance used in the optimize R function for the one-dimensional optimization of the log-likelihood function.
flag_plot	Logical value for plotting the trend of the log-likelihood function with respect to the parameter under consideration. A plot for each iteration (n_iter) is reported. Defaults to FALSE. Be careful that if the optimal parameters are provided, then the trend may be always the same and therefore it may be sufficient to set n_iter = 1. On the other hand, if optimal parameters are not provided, then it is recommended to impose a higher n_iter.
n_points	Number of internal points in which the log-likelihood function must be evaluated, to plot it.
cex	Dimension of the points in the plot.
cex_max	Dimension of the optimal point in the plot.
color_bg	Color used in the plot for the points.
color_max_bg	Color used for the optimal point in the plot.
pch	Shape to be used for the points.

Details

The one-dimensional analysis of the log-likelihood function can be performed in two ways, with two different aims and results:

- Keeping fixed the other parameters (all the parameters in the vector, except for the one under consideration) to their optimal value (flag_optimal_params = TRUE), determined through the multi-dimensional optimization. In this way, the optimized value of the parameter coincides with the one get with the general and global approach and, therefore, there is no need to repeat this procedure several times (n_iter = 1). However, this approach is really useful if we want to check the trend the log-likelihood function and to observe if it increases, decreases or is constant.
- Letting the other parameters vary in their range (flag_optimal_params = FALSE). The optimized parameter value will always assume a different value, because it depends on the value of the other parameters, and it is suggested to repeat the procedure several times (n_iter $\geq$ 5), so that it is possible to identify a precise existence region for such parameter

Value

If the flag for the plot has been activated, the function returns both the plot of the one-dimensional log-likelihood function and a class S3 object. Otherwise, only a S3 object of class 'AdPaik_1D'. This class object is composed of:

- numerical vector of length @n_iter containing the optimal estimated parameter.
- numerical vector of length @n_iter containing the associated one-dimensional optimized log-likelihood value

Examples

```
# Consider the 'Academic Dropout dataset'
data(data_dropout)
# Define the variables needed for the model execution
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10

# Identify a parameter existence range
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0.5, 1 - eps, 1, 10)
index_param_to_vary <- 1
analysis_1D_opt <- AdPaik_1D(formula, data_dropout,
                             time_axis, index_param_to_vary,
                             flag_optimal_params = FALSE,
                             optimal_params = NULL,
                             flag_plot = TRUE,
                             categories_range_min, categories_range_max,
                             n_iter = 5)


# or Study the log-likelihood behaviour
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0.4, 1 - eps, 1, 10)
index_param_to_vary <- 14
# Call the main model
result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)
```

```
analysis_1D_opt <- AdPaik_1D(formula, data_dropout, time_axis,
                             index_param_to_vary, flag_optimal_params = TRUE,
                            flag_plot = TRUE, optimal_params = result$OptimalParameters,
                             categories_range_min, categories_range_max, n_iter = 1)
```

`bas_hazard` *Baseline hazard step-function*

Description

The method computes the baseline hazard step-function in each interval of the time-domain, using the estimated parameters $\phi_k, \forall k$

Usage

```
bas_hazard(optimal_params, time_axis)
```

Arguments

`optimal_params`	Numerical vector of length equal to the number of model parameters, containing the optimal estimated parameters.
`time_axis`	Numerical vector of temporal domain.

Value

Numerical vector of length equal to the number of intervals of the time-domain, with the value of the baseline hazard step-function.

`check.categories_params`
Check correctness of parameters categories

Description

The function controls that the provided parameters categories have a length equal to the number of categories required by the model parameters. For the current model, the number of categories is 5 because there are five blocks of unkown parameters ($\phi_k \forall k, \beta_r \forall r, \mu_1, \nu, \gamma_k \forall k$).

Moreover, it also controls that the minimum value of a parameter category is actually less than or eqaul to the maximum value for the same category.

Usage

```
check.categories_params(
  n_categories,
  categories_range_min,
  categories_range_max
)
```

Arguments

`n_categories`	Number of categories expected by the model. For the current model, they are 5.
`categories_range_min`	Numerical vector of length 5, containing the minimum ranges for the parameters beloning to those categories.
`categories_range_max`	Numerical vector of length equal to 5, containing the maximum ranges for the parameters belonging to those categories.

Value

An error if the any condition is not satisfied.

`check.centre` *Check correctness for the cluster variable*

Description

The function controls that the provided cluster variable is a vector, with at least two levels. Indeed, it is not possible to apply the Time-Dependent Shared Frailty Cox Model with no clusters.

Usage

```
check.centre(centre)
```

Arguments

`centre`	Numerical vector of length equal to the number of individuals in the study, containing the individual grouo/cluster membership.

Value

An error if any condition is not satisfied

`check.C_mult` *Check positiveness of the multiplicative constant C*

Description

The method controls the multiplicative constant C is non-negative (i.e. positive).

Usage

```
check.C_mult(C_mult)
```

Arguments

C_mult	Multiplicative constant

Value

An error if the condition is not satisfied.

---

check.dataset *Check presence of null or nan element value in the dataset*

Description

The method controls that the dataset does not contain 'NULL', 'null', 'NaN' or 'nan' value.

Usage

```
check.dataset(data)
```

Arguments

data	Dataset (dataframe)

Value

An error if any condition is not satisfied.

---

check.flag_optimal_params *Check coherence between flag for optimal parameters and optimal parameters*

Description

The function controls that one of the following condition is satisfied:

- if the flag for the optimal parameters is activated, then the optimal parameters should be provided in input
- if the flag is not activated, then the optimal parameters should not be provided and the parameter vector should be NU

Usage

```
check.flag_optimal_params(optimal_params, flag_optimal_params)
```

Arguments

optimal_params	Either a numerical vector of length equal to the number of model parameters or a NULL value.
flag_optimal_params	Logical value. Did the user want to provide optimal parameters vector? If so, the variable should be TRUE; otherwise (no optimal parameters), FALSE.

Value

An error if any condition is not satisfied.

---

check.formula_terms *Check correctness of formula terms*

Description

The function controls that the terms composing the formula object provided in input to the main model are correct. They must include:

- response variable on the left hand side
- covariates (numerical or categorical) on the right hand side
- cluster variable (categorical) on the right hand side and specified by the function 'cluster()'

Moreover, it controls that the covariates are contained in the dataset provided.

Usage

```
check.formula_terms(formula, data)
```

Arguments

formula	Formula object specifying the relationship between the time-to-event, the covariates and the cluster variables.
data	Dataset in which these variables can be found.

Value

An error if any condition is not satified.

check.frailty_dispersion *Check correctness of frailty standard deviation*

Description

The function controls that the frailty standard deviation vector has a length equal to the number of inyervals of the time domain and that its elements are non-negative.

Usage

```
check.frailty_dispersion(frailty_dispersion, n_intervals)
```

Arguments

frailty_dispersion	Frailty dispersion
n_intervals	Number of intervals of the time-domain

Value

An error if any condition is not satisfied.

check.index *Check existence of provided input index*

Description

The method controls that the provided input index exists: it cannot be greater than the maximum number of parameters of the current model.

Usage

```
check.index(index, n_params)
```

Arguments

index	Index with respect to which the user wants to study the one dimensional behaviour of the log-likelihood function.
n_params	Number of parameters of the model

Value

An error if any condition is not satisfied.

check.pchtype_colorbg *Check correctness of plot variables pch and color*

Description

The function controls that the input variables 'pch_type' and 'color_bg' have the correct structure, they have the same dimension of the number of clusters in the dataset and they have meaningful elements.

These variables are used for the plot of the posterio frailty estimates: the estimates for each faculty are plotted through a symbol, having color and shape indicated by the variables (for the k-th faculty, consider the k-th element of both vectors).

Usage

```
check.pchtype_colorbg(centre_codes, pch_type, color_bg)
```

Arguments

centre_codes	Numerical vector of length equal to the number of centres/clusters in the dataset and containing the distinct centres/clusters. They correspond to the levels of the numerical vector of individual group membership.
pch_type	Numerical vector of length equal to the number of centres and containing the point shape for each faculty.
color_bg	Numerical vector of length equal to the number of centres and containing the color of the point for each faculty.

Value

An error if any condition is not satisfied.

check.poslegend *Check correctness of legend position*

Description

The function controls that the provided position of the legend is correct. It can be either a vector of length 2, giving the x and y coordinates, or a string, giving the exact position among different possibilities.

Usage

```
check.poslegend(pos_legend)
```

Arguments

pos_legend	Either a numerical vector of length 2, with the x and y coordinates, or a string with the exact position.

Value

An error if any condition is not satisfied.

---

check.post_frailty_centre *Check numerosity of posterior frailty estimates*

Description

The function controls that a time-dependent posterior frailty estimate is computed for each centre

Usage

```
check.post_frailty_centre(post_frailty_est, centre_codes)
```

Arguments

post_frailty_est	An S3 class object containing the posterior frailty estimates $\hat{\alpha}_j, \hat{\epsilon}_{jk}, \hat{Z}_{jk}, \forall j, k$
centre_codes	Numerical vector of length equal to the number of distinct centres/clusters in the study

Value

An error if any condition is not satisfied.

---

check.pos_frailty_sd *Check positiviness of the frailty standard deviation*

Description

The method controls that the frailty standard deviation vector has non-negative elements

Usage

```
check.pos_frailty_sd(sd, n_intervals)
```

Arguments

sd	Numerical vector of length equal to the number of intervals, containing the frailty standard deviation
n_intervals	Number of intervals of the time-domain

Value

An error if any condition is not satisfied.

---

check.range_params *Check correctness of input parameters*

Description

The function controls that the input parameter vector have a length equal to the theoretical one required by the model and that each parameter properly belongs to its range.

Usage

```
check.range_params(optimal_params, params_range_min, params_range_max)
```

Arguments

optimal_params	Numerical vector of length equal to the number of model parameters. For the 'Adapted Paik et al.'s Model' it can be computed as: $n_p = 2L + R + 2$, where L stands for the number of intervals of the time domain and R the number of regressors of the dataset.
params_range_min	Numerical vector of length equal to the number of model parameters (n_p) and containing the minimum range for each parameter.
params_range_max	Numerical vector of length equal to the number of model parameters (n_p) and containing the maximum range for each parameter.

Value

An error if any condition is not satisfied.

`check.result.AdPaik` *Check structure of the 'AdPaikModel' output*

Description

The function controls that the structure of the input variable is coherent with the one returned by the 'AdPaikModel' execution.

Usage

```
check.result.AdPaik(result)
```

Arguments

`result`	S3 object of class 'AdPaik', composed of several elements. See details.

Details

The output of the model call 'AdPaikModel(...)' is a S3 object of class 'AdPaik', composed of 18 quantities:

- formula: formula object provided in input by the user and specifying the relationship between the time-to-event, the covariates of the dataset (regressors) and the cluster variable.
- Regressors: categorical vector of length R, with the name of the regressors. They could be different from the original covariates of the dataset in case of categorical covariates. Indeed, each categorical covariate with n levels needs to be transformed into (n-1) dummy variables and, therefore, (n-1) regressors.
- NRegressors: number of regressors (R)
- ClusterVariable: name of the variable with respect to which the individuals can be grouped.
- NClusters: number of clusters/groups/centres
- NIntervals: number of intervals of the time-domain, also called with L. It corresponds to the length of the time-domain minus 1.
- NParameters: number of parameters of the model. It can be computed as: $n_p = 2L + R + 2$.
- ParametersCategories: Numerical vector of length 5, containing the numerosity of each parameter category.
- ParametersRangeMin: Numerical vector of length n_p, giving the minimum range of each parameter.
- ParametersRangeMax: Numerical vector of length n_p, giving the maximum range of each parameter.
- Loglikelihood: value of the maximized log-likelihood function, at the optimal estimated parameters.
- AIC: 'Akaike Information Criterion': it can be computed as $AIC = 2n_p - 2ll_{optimal}$. It gives an idea of the loss of information related to the model fitting and output. The smaller it is, the less loss of information and the better model accuracy.

- Status: Logical value. Does the model reach convergence? If so, the variable is TRUE, otherwise FALSE.
- NRun: Number of runs necessary to reach convergence. If the model does not reach convergence, such number is equal to the maximum number of imposed runs.
- OptimalParameters: numerical vector of length n_p, containing the optimal estimated parameters or, in other words, the parameters that maximizes the log-likelihood function.
- StandardErrorParameters: numerical vector of length n_p, corresponding to the standard error of each estimated parameters.
- ParametersCI: S3 object of class 'ParametersCI', composed of two numerical vector of length equal to n_p: the left and right confidence interval of each estimated parameter.
- FrailtyStandardDeviation: numerical vector of length equal to L (i.e. number of intervals of the time-domain), reporting the standard deviation of the frailty.
- PosteriorFrailtyEstimates: S3 object of class 'PFE.AdPaik'. See details.
- PosteriorFrailtyVariance: S3 object of class 'PFV.AdPaik'. See details.
- PosteriorFrailtyCI: S3 object of class 'PFCI.AdPaik'. See details.

The object of class 'PFE.AdPaik' contains the Posterior Frailty Estimates computed with the procedure indicated in the reference paper and it is composed of three elements:

- 'alpha': posterior frailty estimates for $\alpha_j, \forall j$. It is a vector of length equal to the number of groups/centres.
- 'eps': posterior frailty estimates for $\epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).
- 'Z': posterior frailty estimates for $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).

The object of class 'PFV.AdPaik' contains the Posterior Frailty Variances computed as indicated in the reference papaer and it is composed of three elements:

- 'alphaVar': posterior frailty variance for $\alpha_j, \forall j$. It is a vector of length equal to the number of groups/centres.
- 'epsVar': posterior frailty variance for $\epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).
- 'ZVar': posterior frailty variance for $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$. Matrix of dimension (N, L).

The object of class 'PFCI.AdPaik' contains the Posterior Frailty Confidence Interval and it is composed of two elements:

- left confidence interval for the estimated $\hat{Z}_{jk}, \forall j, k$. Matrix of dimension (N, L).
- right confidence interval for the estimated $\hat{Z}_{jk}, \forall j, k$. Matrix of dimension (N, L).

Value

An error if any condition is not satisfied.

check.structure_paramsCI

Check structure for the Parameters Confidence Interval

Description

The function controls that the structure of the Parameters Confidence Intervals coincides with the theoretical one.

Usage

```
check.structure_paramsCI(parametersCI)
```

Arguments

parametersCI	S3 object of class 'ParametersCI', composed of two elements: • left confidence interval: numerical vector of length equal to the number of parameters in the model • right confidence interval: numerical vector of length equal to the number of parameters in the model

Value

An error if any condition is not satisfied.

check.structure_post_frailty_CI

Check structure of Posterior Frailty Confidence Interval

Description

The function controls that the structure of the 'Posterior Frailty Confidence Interval' coincides with the theoretical one.

Usage

```
check.structure_post_frailty_CI(post_frailty_CI, n_intervals, n_centres)
```

Arguments

post_frailty_CI	Posterior frailty estimates S3 object of class 'PFCI.AdPaik', composed of two elements: • left confidence interval for the estimated $\hat{Z}_{jk}, \forall j, k$ • right confidence interval for the estimated $\hat{Z}_{jk}, \forall j, k$
n_intervals	Number of intervals of the time-domain
n_centres	Number of centres/clusters.

Value

An error if any condition is not satisfied.

`check.structure_post_frailty_est` *Check structure of Posterior Frailty Estimates*

Description

The function controls that the structure of the 'Posterior Frailty Estimates' coincides with the theoretical one.

Usage

```
check.structure_post_frailty_est(post_frailty_est, n_intervals, n_centres)
```

Arguments

`post_frailty_est`	Posterior frailty estimates S3 object of class 'PFE.AdPaik', composed of three elements: • 'alpha': posterior frailty estimates for $\alpha_j, \forall j$. It is a vector of length equal to the number of centres. • 'eps': posterior frailty estimates for $\epsilon_{jk}, \forall j, k$. It is a matrix of dimension (n_centres, n_intervals). • 'Z': posterior frailty estimates for $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$. It is a matrix of dimension (n_centres, n_intervals)
`n_intervals`	Number of intervals of the time-domain
`n_centres`	Number of centres/clusters.

Value

An error if any condition is not satisfied.

check.structure_post_frailty_var *Check structure of Posterior Frailty Variances*

Description

The function controls that the structure of the 'Posterior Frailty Variances' coincides with the theoretical one.

Usage

```
check.structure_post_frailty_var(post_frailty_var, n_intervals, n_centres)
```

Arguments

post_frailty_var	Posterior frailty variances S3 object of class 'PFV.AdPaik', composed of three elements: • 'alphaVar': posterior frailty variance for $\alpha_j, \forall j$. It is a vector of length equal to the number of centres. • 'epsVar': posterior frailty variance for $\epsilon_{jk}, \forall j, k$. It is a matrix of dimension (n_centres, n_intervals). • 'ZVar': posterior frailty variance for $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$. It is a matrix of dimension (n_centres, n_intervals)
n_intervals	Number of intervals of the time-domain
n_centres	Number of centres/clusters.

Value

An error if any condition is not satisfied.

check.time_axis *Check correctness of time domain subdivision*

Description

The function controls that the time domain is a vector and it has at least 2 elements and that all of them are not negative. Moreover, it checks that all its elements are non-negative and properly ordered, in an ascending way.

Usage

```
check.time_axis(time_axis)
```

Arguments

time_axis	Numerical vector of temporal domain subdivision.

Value

An error is returned if any condition is not satisfied.

---

check.value_post_frailty *Check non-negativeness of the posterior frailty estimates*

Description

The function controls that all posterior frailty estimates are non-negative. Indeed, by construction the realizations of a gamma distribution are non negative.

Usage

```
check.value_post_frailty(post_frailty_est, n_centres, n_intervals)
```

Arguments

post_frailty_est	An S3 class object containing the posterior frailty estimates: $\hat{\alpha}_j, \hat{\epsilon}_{jk}, \hat{Z}_{jk}, \forall j, k$
n_centres	Number of groups/clusters.
n_intervals	Number of intervals of the time domain

Value

An error if any condition is not satisfied.

---

coef.AdPaik *Extracts the optimal parameters of each cateogry for the 'Adapted Paik et al.' Model*

Description

Extracts the optimal parameters ϕ, β, μ_1, ν, γ obtained with the time-dependent frailty model proposed in the 'Adapted Paik et al.' framework.

Usage

```
## S3 method for class 'AdPaik'
coef(object, ...)
```

Arguments

object	An S3 object of class AdPaik, returned by the main model function (AdPaikModel). This object contains all the optimal parameter estimates.
...	Additional arguments to be passed to other methods.

Details

The coef.AdPaik function extracts the coefficients from the OptimalParameters field in object.

The function validates the structure of object and ensures compatibility with the expected model output. It throws an error if the object is malformed or inconsistent.

Value

A named list containing the categories of optimal parameters.

Examples

```
# Example using the 'Academic Dropout' dataset
data(data_dropout)

# Define the formula and time axis for the model
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0, 1 - eps, 1, 10)


# Run the main model
result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)

# Extract the coefficients
coef(result)
```

---

coefse *Extracts the standard errors computed for each cateogry for the 'Adapted Paik et al.' Model*

Description

Extracts the standard errors for ϕ, β, μ_1, ν, γ obtained with the time-dependent frailty model proposed in the 'Adapted Paik et al.' framework.

Usage

```
coefse(object)
```

Arguments

object	An S3 object of class AdPaik, returned by the main model function (AdPaikModel). This object contains all the optimal parameter estimates.

Details

The coefse function extracts the standard errors for the estimated parameters from the StandardErrorParameters field in object.

The function validates the structure of object and ensures compatibility with the expected model output. It throws an error if the object is malformed or inconsistent.

Value

A named list containing the categories of the standard errors for the optimal parameters.

Examples

```
# Example using the 'Academic Dropout' dataset
data(data_dropout)

# Define the formula and time axis for the model
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0, 1 - eps, 1, 10)


# Run the main model
result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)

# Extract the coefficients
coefse(result)
```

---

confint.AdPaik *Extracts the confidence intervals computed for each cateogry for the 'Adapted Paik et al.' Model*

Description

Extracts the confidence intervals for ϕ, β, μ_1, ν, γ obtained with the time-dependent frailty model proposed in the 'Adapted Paik et al.' framework.

Usage

```
## S3 method for class 'AdPaik'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

object	An S3 object of class AdPaik, returned by the main model function (AdPaikModel). This object contains all the optimal parameter estimates.
parm	A specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. Defaults to NULL, and all parameters are considered. Changing it is not supported for this model. It will be ignored.
level	The confidence level required. Defaults to 0.95.
...	Additional arguments to be passed to other methods.

Details

The confint.AdPaik function extracts the standard errors for the estimated parameters from the ParametersCI field in object.

The function validates the structure of object and ensures compatibility with the expected model output. It throws an error if the object is malformed or inconsistent.

Value

A named list containing the categories of the standard errors for the optimal parameters.

Examples

```
# Example using the 'Academic Dropout' dataset
data(data_dropout)

# Define the formula and time axis for the model
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0, 1 - eps, 1, 10)


# Run the main model
result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)

# Extract the coefficients
confint(result)
```

---

data_dropout *Data Dropout Dataset*

Description

This dataset is extracted from an administrative database provided by a non-specified university and tracks students enrolled in 2012 over three academic years (or 6 semesters). We are interested in understanding what factors lead to students dropping out. Dropout students with a time-instant in the first semester have been removed, for internal reasons (the university cannot take preventive action to reduce or avoid their withdrawal). The students are followed for at most 3 academic years or, equivalently, 6 semesters (follow-up periods), from the first day of lecture up to the time-instant of withdrawal (i.e. survival event) or the end of the academic year.

Usage

```
data_dropout
```

Format

A data frame with 4448 rows and 4 columns:

Gender Categorical covariate (Male or Female).

CFUP Standardized numerical covariate indicating the number of credits \ passed by the students by the end of the first semester.

time_to_event Time (in semesters) at which a student decides to leave the university. \ A value greater than 6.0 indicates the student did not drop out during the follow-up (e.g. 6.1 semesters)

group Categorical variable indicating the student's course of study, with 16 different levels from CosA, CosB, ... , CosP.

Source

Data for demonstration purposes.

---

extract_dummy_variables

Transform categorical covariate into dummy variables

Description

This function produces for a categorical variable of the dataset (covariate) the associated dummy variables: for n levels of the covariate, (n-1) dummy binary variables are generated. The chosen reference value is the first one of the list of extracted levels and cannot be changed (the first one in alphabetical order). Therefore, if an individual has null value for all dummy variables, then his/her belonging level is the reference one.

Each dummy variable has a name, corresponding to the name of the covariate + name of the level.

Usage

```
extract_dummy_variables(covariate, covariate_name)
```

Arguments

covariate	Categorical dataset covariate, with at least 2 levels.
covariate_name	Name of the covariate, for assigning each dummy variable a proper name.

Details

The S3 class object 'DummyData' contains the variables related to the transformation of a single categorical covariate present in the dataset into (n-1) binary covariates, stored in a matrix. To be precise:

- DummyMatrix: binary matrix of dimension (n_individuals, n-1), where each column corresponds to one level of the original categorical covariate.
- DummyVariablesName: categorical vector of length (n-1), reporting the names of the dummy variables and, therefore, the new name of each regressor.
- DummyVariablesNumber: number of dummy variables (n-1).

Value

S3 object of class 'DummyData', composed of three elements. See details.

---

extract_event_data *Extracting variables for Posterior Frailty Estimates computation*

Description

Function for extracting from the dataset quantities necessary to the evaluation of the posterior frailty estimates.

Usage

```
extract_event_data(dataset, time_to_event, centre, time_axis, phi, betar)
```

Arguments

dataset	Dataset containing the covariates/regressors. Their numerosity is indicated with R.
time_to_event	Time-instant in the follow-up in which an individual fails or faces the event. If an individual does not face the event, the time-instant assumes a default value.
centre	Categorical vector indicating the group/cluster membership. The number of distinct group is indicated with N.
time_axis	Numerical vector of the temporal domain. Its length is (L+1), where L indicates the number of intervals of the time-domain.
phi	Numerical vector of length L, of estimated baseline log-hazard.
betar	Numerical vector of length R, of estimated regressors.

Details

The S3 class obejct 'EventData' contains the variables necessary for the estimate of the posterior frailty and that can be extracted or computed starting from the dataset.

- N_ik: matrix of dimension (N, L), containing the number of event in each interval k and group i.
- N_i: numerical vector of length L, with the number of event in each group i. It can be computed as: $N_i = \sum_{k=1}^{L} N_{ik}$.
- e_ijk: matrix of dimension (n_individuals, L) with the evaluation of the temporal integral, for each individual j, group i and interval k.
- Y_risk: binary matrix of dimension (n_individuals, L) reporting for each individual, in each interval, his/her risk of facing the event. For an individual, the risk is equal to 1 in an interval k if, in that interval, he/she has not faced the event yet; otherwise, it is equal to 0.
- cum_hazard_group: matrix of dimension (N, L), where each element in position (i,k) indicates the computed cumulative hazard for all individuals belonging to group i and at interval k.
- sum_cum_hazard_group: numerical vector of length N, giving the sum of the computed cumulative hazard for all intervals k and for all individuals belonging to group i. It can be computed from the previous element, summing with respect to the interval k.

Value

S3 object of class 'EventData', composed of six elements. See details.

---

frailty_sd *Frailty standard deviation and Variance for the 'Adapted Paik et al.'s Model'*

Description

The function computes both the standard deviation and the variance of the time-dependent frailty of the 'Adapted Paik et al.'s Model'.

Recalling the frailty structure $Z_{jk} = \alpha_j + \epsilon_{jk}$ as being composed by a constant group-dependent term (α_j) and a time and group dependent term (ϵ_{jk}), the frailty standard deviation (and variance) can be computed in two different way:

- Considering only the time-dependent spread of the clusters/groups/centre: $sd(Z_{jk}) = \mu_2 * \gamma_k$. In this case, the flag_fullsd should be FALSE.
- Considering both the time-dependent and constant spread of the clusters: $sd(Z_{jk}) = \mu_1 * \nu + \mu_2 * \gamma_k$. The new added term only moves upward the other case and the flag_fullsd should be TRUE.

The final case only depends on what we want to observe.

Usage

```
frailty_sd(result, flag_fullsd = TRUE)
```

Arguments

result	S3 object of class 'AdPaik' returned by the main model output, that contains all the information for the computation of the frailty standard deviation.
flag_fullsd	Logical value. Do we want to compute the full frailty standard deviation? If so, the flag must be TRUE, otherwise, FALSE.

Value

S3 class object 'FrailtyDispersion' containing both two numerical vectors of length equal to the number of intervals of the time-domain:

- FrailtyVariance
- FrailtyStandardDevation

Examples

```
# Consider the 'Academic Dropout dataset'
data(data_dropout)

# Define the variables needed for the model execution
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0, 1 - eps, 1, 10)

# Call the main model

result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)

frailty_sd(result, TRUE)
frailty_sd(result, FALSE)
```

---

frailty_sd.AdPaik *Frailty standard deviation and Variance for the 'Adapted Paik et al.'s Model'*

Description

The function computes both the standard deviation and the variance of the time-dependent frailty of the 'Adapted Paik et al.'s Model'.

Recalling the frailty structure $Z_{jk} = \alpha_j + \epsilon_{jk}$ as being composed by a constant group-dependent term (α_j) and a time and group dependent term (ϵ_{jk}), the frailty standard deviation (and variance) can be computed in two different way:

- Considering only the time-dependent spread of the clusters/groups/centre: $sd(Z_{jk}) = \mu_2 * \gamma_k$. In this case, the flag_fullsd should be FALSE.

- Considering both the time-dependent and constant spread of the clusters: $sd(Z_{jk}) = \mu_1 * \nu + \mu_2 * \gamma_k$. The new added term only moves upward the other case and the flag_fullsd should be TRUE.

The final case only depends on what we want to observe.

Usage

```
frailty_sd.AdPaik(result, flag_fullsd)
```

Arguments

result	S3 object of class 'AdPaik' returned by the main model output, that contains all the information for the computation of the frailty standard deviation.
flag_fullsd	Logical value. Do we want to compute the full frailty standard deviation? If so, the flag must be TRUE, otherwise, FALSE.

Value

S3 class object 'FrailtyDispersion' containing both two numerical vectors of length equal to the number of intervals of the time-domain:

- FrailtyVariance
- FrailtyStandardDevation

Examples

```
# Consider the 'Academic Dropout dataset'
data(data_dropout)

# Define the variables needed for the model execution
formula <- time_to_event ~ Gender + CFUP + cluster(group)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
eps <- 1e-10
categories_range_min <- c(-8, -2, eps, eps, eps)
categories_range_max <- c(-eps, 0, 1 - eps, 1, 10)


# Call the main model
result <- AdPaikModel(formula, data_dropout, time_axis,
                      categories_range_min, categories_range_max, TRUE)

frailty_sd(result, TRUE)
frailty_sd(result, FALSE)
```

`ll_AdPaik_1D` *One-dimensional log-likelihood function to be optimized.*

Description

Model log-likelihood function to be optimized only with respect to a parameter. To correctly identify this parameter inside the model and inside the vector of all parameter, it is necessary to provide also the position (index) of this parameter in the vector.

This function is internally used by the main function @AdPaikModel to perform, as said, the one-dimensional optimization through 'optimize'. It cannot be used to evaluate the log-likelihood function at a vector of parameter and at the provided data. For this purpose, we have to use another implemented function, called @ll_AdPaik_eval.

Usage

```
ll_AdPaik_1D(
  x,
  index,
  params,
  dataset,
  centre,
  time_axis,
  dropout_matrix,
  e_matrix
)
```

Arguments

`x`	Value of the parameter, with respect to which the log-likelihood function has to be optimized.
`index`	Index of the parameter inside the parameter vector. For instance, if we need to optimize the log-likelihood function with respect to the first regressor, then @x will be generic but @index will be equal to (n_intervals + 1) because in the parameter vector the first regressor appears after the baseline log-hazard group (n_intervals elements).
`params`	Parameter vector.
`dataset`	Matrix containing only the formula regressors, that is the regressors appearing in the formula object provided by the user and eventually modified if they are categorical (nd therefore transformed into dummy variables).
`centre`	Individual membership to the clusters.
`time_axis`	Temporal domain.
`dropout_matrix`	Binary matrix indicating in which interval of the time domain an individual failed. For an individual, the sum of the row elements must be equal to 1 (if he/she failed) or 0 (if he/she does not failed). It has dimension equal to (n_individuals, n_intervals).

e_matrix Matrix of dimension (n_individual, n_intervals), where each element contains the evaluation of the temporal integral, performed through the function @param time_int_eval.

Details

This function firstly divides the individuals according to their group/cluster membership, extracting group customized dataset and other variables, and then compute the group log-likelihood function through the function @ll_AdPaik_centre_1D. The produced group log-likelihood value is summed together the other values into a unique result, that corresponds to the overall (and final) log-likelihood value.

Value

Overall log-likelihood function

---

ll_AdPaik_centre_1D *One-dimensional group log-likelihood function.*

Description

This function simply implements the group log-likelihood function, following the definition. It is internally used by @ll_AdPaik_1D and, therefore, it requires as first and second argument the parameter according to which the global log-likelihood is one-dimensionally optimized and its position inside the vector of parameters.

Usage

```
ll_AdPaik_centre_1D(
  param_onedim,
  index_param_onedim,
  params,
  dataset,
  dropout_matrix,
  e_matrix
)
```

Arguments

param_onedim One dimensional parameter, with respect to which the log-likelihood function must be optmize.

index_param_onedim Index of the previous parameter inside the parameter vector.

params Parameter vector.

dataset Matrix of dataset regressors, with a number of rows equal to the number of individuals in a cluster.

dropout_matrix	Binary matrix indicating in which interval of the time domain and individual failed. For an individual, the sum of the row elements must be equal to 1 (if he/she failed) or 0 (if he/she does not failed). It has dimension equal to (n_individuals, n_intervals)
e_matrix	Matrix of dimension (n_individual, n_intervals), where each element contains the evaluation of the temporal integral, performed through the function @param time_int_eval.

Value

Centre log-likelihood function.

`ll_AdPaik_centre_eval` *Evaluation of model group log-likelihood*

Description

Evaluation of model group log-likelihood at the provided parameter vector and data. This function is internally called by 'll_AdPaik_eval' to evaluate the log-likelihood function, considering all and only the individuals belonging to a group.

Usage

```
ll_AdPaik_centre_eval(params, dataset, dropout_matrix, e_matrix)
```

Arguments

params	Parameter vector.
dataset	Matrix of dataset regressors, with a number of rows equal to the number of individuals in a cluster.
dropout_matrix	Binary matrix indicating in which interval of the time domain and individual failed. For an individual, the sum of the row elements must be equal to 1 (if he/she failed) or 0 (if he/she does not failed). It has dimension equal to (n_individuals, n_intervals)
e_matrix	Matrix of dimension (n_individual, n_intervals), where each element contains the evaluation of the temporal integral, performed through the function @time_int_eval.

Value

Group log-likelihood evaluation

`ll_AdPaik_eval` *Evaluation of model log-likelihood*

Description

Evaluation of the log-likelihood function at the provided parameter vector and data.

Usage

```
ll_AdPaik_eval(params, dataset, centre, time_axis, dropout_matrix, e_matrix)
```

Arguments

`params`	Parameter vector
`dataset`	Matrix of dimension equal to (number of individuals in the study, number of regressors), where only the regressors indicated in the formula object are considered.
`centre`	Vector of length equal to the number of individuals in the study, where each element corresponds to the individual cluster membership.
`time_axis`	Temporal domain
`dropout_matrix`	Binary matrix indicating in which interval of the time domain and individual failed. For an individual, the sum of the row elements must be equal to 1 (if he/she failed) or 0 (if he/she does not failed). It has dimension equal to (n_individuals, n_intervals)
`e_matrix`	Matrix of dimension (n_individual, n_intervals), where each element contains the evaluation of the temporal integral, performed through the function @time_int_eval.

Details

The function divides the individuals according to their group/cluster membership and then evaluates the group log-likelihood through another implemented function, but using all and only the individuals belonging to that group. Then the results are summed together to return the overall log-likelihood value.

Value

Overall log-likelihood function value at the provided parameters and data

n_nodes	*Nodes and weights for the Gauss_hermite quadrature formula for the 'Centre-Specific Frailty Model with Power Parameter'. The nodes and weights have been extracted from the 'Handbook of Mathematical functions' pag 940.*

Description

Nodes and weights for the Gauss_hermite quadrature formula for the 'Centre-Specific Frailty Model with Power Parameter'. The nodes and weights have been extracted from the 'Handbook of Mathematical functions' pag 940.

Usage

```
n_nodes
```

Format

An object of class `numeric` of length 1.

n_nodesG	*Nodes and weights for the Gauss-Hermite quadrature formula, for the 'Stochastic Time-Dependent Centre-Specific Frailty Model'. For the G function, the chosen nodes should not contain the zero (node) since it appears at the denominator of a fraction. Also in this case, the nodes and weights have been extracted from the 'Handbook of Mathematical functions', pag 940.*

Description

Nodes and weights for the Gauss-Hermite quadrature formula, for the 'Stochastic Time-Dependent Centre-Specific Frailty Model'. For the G function, the chosen nodes should not contain the zero (node) since it appears at the denominator of a fraction. Also in this case, the nodes and weights have been extracted from the 'Handbook of Mathematical functions', pag 940.

Usage

```
n_nodesG
```

Format

An object of class `numeric` of length 1.

params_CI *Confidence interval for the optimal estimated parameters*

Description

The function provides the confidence interval for each estimated parameter, using the standard error computed through another method and provided as second argument to the current function.

Usage

```
params_CI(optimal_params, se_params, level)
```

Arguments

optimal_params	Numerical vector of optimal estimated parameters. Its length is equal to the number of model parameters.
se_params	Numerical vector containing the standard error associated to each estimated parameter.
level	A numeric value representing the confidence level.

Value

A S3 object of class 'ParametersCI', composed of two numerical vector of length equal to the number of model parameters:

- ParamsCI_left: left confidence interval for each parameter
- ParamsCI_right: right confidence interval for each parameter

params_se.AdPaik *Standard error of the parameters*

Description

Function for computing the standard error of each optimal parameter, estimated through the constraint multi-dimensional optimization. The procedure for the computation is based on the numerical approximation of the second derivative of the log-likelihood function, by the 'centered finite difference scheme' with an accuracy of the second order.

Usage

```
params_se.AdPaik(
  optimal_params,
  params_range_min,
  params_range_max,
  dataset,
  centre,
  time_axis,
  dropout_matrix,
  e_matrix,
  h_dd
)
```

Arguments

optimal_params	Numerical vector of optimal parameters. Its length (i.e. number of parameters) is equal to n_p.
params_range_min	Numerical vector of length equal to n_p, containing the minimum range of each parameter.
params_range_max	Numerical vector of length equal to n_p, containing the maximum range of each parameter.
dataset	Dataset containing the value of the regressors for all individuals in the study.
centre	vector containing the group membership of each individual and that induces the clustering subdivision.
time_axis	Temporal domain. Its number of intervals corresponds to the length of the time-domain minus 1
dropout_matrix	Binary matrix of dimension (n_individuals, n_intervals). The sum of the elements of each row must be (1), if the associated individual failed in a precise interval, and (0) if the individual did not fail in the @time-axis. Therefore, if an individual failed in the time-domain, the interval in which he failed will have value (1) and the others (0).
e_matrix	Matrix of dimension (n_individuals, n_intervals) where each element contains the resolution of the temporal integral for that individual in that interval, thorugh the 'e_time_fun' function.
h_dd	Discretization step for the numerical approximation of the second derivative fo the loglikelihood function.

Details

The standrd error of each parameter is computed as the inverse of the square root of the 'Information matrix', that in turn is computed as the opposite of the 'Hessian matrix'. Only its diagonal is built and its elements are separatey evaluated through a numerical approximation of the second derivative of the log-likelihood function.

The function requires the optimal parameter vector and other parameters-related variables, to check:

- the right numerosity of the parameter vector
- the correct range existence of each parameter (i.e. each parameter lies in its range).

Value

Vector of parameter standard error, of length equal to the number of model parameters.

`plot_bas_hazard` *Plot the Baseline Hazard Step-Function*

Description

This function plots the baseline hazard step-function based on the estimated parameters from the Adapted Paik et al.'s model.

Usage

```
plot_bas_hazard(
  result,
  xlim = c(min(result$TimeDomain), max(result$TimeDomain)),
  ylim = c(0, max(result$BaselineHazard)),
  xlab = "Time",
  ylab = "Values",
  main = "Baseline hazard step-function",
  color = "black",
  pch = 21,
  bg = "black",
  cex_points = 0.7
)
```

Arguments

`result`	S3 object of class 'AdPaik', returned by the method call 'AdPaikModel(...)'.
`xlim`	A numeric vector specifying the x-axis limits. Default is set to the interval min-max of the time-domain.
`ylim`	A numeric vector specifying the y-axis limits. Default is 0 to the maximum value of the baseline hazard.
`xlab, ylab`	String giving the x and y axis name. Default values are 'x' and 'y'.
`main`	Title of the plot. Default title is 'Baseline hazard step-function'.
`color`	Color used for plotting the horizontal segments of the step-function. Default one is 'black'.
`pch`	Symbol for marking the boundaries of each segment. Default is a dot (value 21).
`bg`	Color for the boundary symbols. Default matches the plot color ('black').
`cex_points`	Size of the boundary symbols. Default is 0.7.

Details

The function plots a horizontal segment for each interval of the time domain, representing the baseline hazard. The boundaries of each segment are marked with colored dots, and subsequent segments are intentionally left unconnected to reflect the discrete nature of the intervals.

Value

Plot of the baseline hazard step-function and value of the function in each interval.

Examples

```
# Import data
data(data_dropout)

# Define the variables needed for the model execution
eps_paik <- 1e-10
categories_range_min <- c(-8, -2, eps_paik, eps_paik, eps_paik)
categories_range_max <- c(-eps_paik, 0.4, 1 - eps_paik, 1, 10)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
formula <- time_to_event ~ Gender + CFUP + cluster(group)


# Call the main model function
result <- AdPaikModel(formula, data_dropout, time_axis, categories_range_min, categories_range_max)

plot_bas_hazard(result)
```

---

plot_frailty_sd *Plot for the Frailty Standard Deviation or Variance*

Description

This function generates a plot of either the frailty standard deviation or the frailty variance for the intervals in the time-domain.

Usage

```
plot_frailty_sd(
  result,
  flag_variance = FALSE,
  flag_sd_external = FALSE,
  frailty_sd = NULL,
  xlim = c(min(result$TimeDomain), max(result$TimeDomain)),
  ylim = NULL,
  xlab = "Time",
  ylab = "Values",
  main = NULL,
```

```
  pch = 21,
  color_bg = "blue",
  cex_points = 0.7
)
```

Arguments

result	An S3 object of class 'AdPaik', returned by the main model call 'AdPaikModel(...)'.
flag_variance	A boolean flag indicating whether to plot the frailty variance (TRUE) or the frailty standard deviation (FALSE). Default is FALSE.
flag_sd_external	A logical flag indicating whether the user is providing an external frailty standard deviation vector.
frailty_sd	A numerical vector representing the evaluated frailty standard deviation, with length equal to the number of time-domain intervals. Its elements must be non-negative. Default is NULL.
xlim	A numeric vector specifying the range for the x-axis (intervals). If NULL, default is set to the interval min-max of the time-domain.
ylim	A numeric vector specifying the range for the y-axis (intervals). If NULL, default is 0 to the maximum value of the frailty variance/standard deviation.
xlab	A string for the x-axis label. Default is 'Intervals'.
ylab	A string for the y-axis label. Default is 'Values'.
main	A string for the plot title. Default title is 'Frailty Standard Deviation' or 'Frailty Variance' according to the produced plot (flag_variance).
pch	A numeric or character symbol used for plotting the frailty standard deviation values. Default is a dot (21).
color_bg	A string specifying the color used for the symbols. Default is 'blue'.
cex_points	A numeric value indicating the size of the plotting symbols. Default is 0.7.

Details

The plot represents the values of the frailty standard deviation or variance for each time interval (represented by its mid point). It connects these points to illustrate the trend of the chosen metric.

This function supports two modes of operation:

- Plotting the frailty standard deviation or variance retrieved from the main model (contained in the S3 object of class 'AdPaik').
- Plotting a user-provided vector of frailty standard deviations, which can be computed using the method frailty.sd. This allows for flexibility in analysis without re-optimizing the log-likelihood function. For instance, users can compare frailty standard deviations computed with different model specifications (e.g., including only time-dependent terms).

The output will differentiate between these two cases, ensuring the correct values are plotted regardless of the source.

Value

A plot displaying either the frailty standard deviation or variance across the specified intervals.

Examples

```
# Import data
data(data_dropout)

# Define the variables needed for the model execution
eps_paik <- 1e-10
categories_range_min <- c(-8, -2, eps_paik, eps_paik, eps_paik)
categories_range_max <- c(-eps_paik, 0.4, 1 - eps_paik, 1, 10)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
formula <- time_to_event ~ Gender + CFUP + cluster(group)

# Call the main model function

result <- AdPaikModel(formula, data_dropout, time_axis, categories_range_min, categories_range_max)

plot_frailty_sd(result)
```

---

plot_ll_1D *Plot the One-Dimensional Log-Likelihood Function*

Description

This function plots the trend of the log-likelihood function concerning a single parameter specified by its index in the parameter vector. It generates samples of the parameter, evaluates them in the log-likelihood function, and displays the results along with the maximum point of the one-dimensional log-likelihood function.

Usage

```
plot_ll_1D(
  param_1D,
  index_param_1D,
  ll_1D,
  params,
  param_range_min,
  param_range_max,
  dataset,
  centre,
  time_axis,
  dropout_matrix,
  e_matrix,
  n_points = 150,
  cex = 0.7,
```

```
  cex_max = 0.8,
  color_bg = "black",
  color_max_bg = "red",
  pch = 21
)
```

Arguments

param_1D	A numeric value representing the optimal parameter determined by maximizing the log-likelihood function for the specified parameter.
index_param_1D	An integer representing the index of the optimal parameter within the parameter vector.
ll_1D	A numeric value of the log-likelihood function evaluated at the optimal parameter param_1D, with the other parameters held constant.
params	A numeric vector of length equal to the number of parameters minus one, containing the fixed values for the other parameters.
param_range_min	A numeric value indicating the minimum allowable value for the parameter param_1D.
param_range_max	A numeric value indicating the maximum allowable value for the parameter param_1D.
dataset	A data frame or matrix containing individual covariates.
centre	A numeric vector indicating individual cluster membership; its length must match the number of individuals in the dataset.
time_axis	A numeric vector corresponding to the subdivisions of the temporal domain.
dropout_matrix	A binary matrix indicating which interval of the time domain an individual failed. Each row should sum to 1 (if failed) or 0 (if not failed), with dimensions (n_individuals, n_intervals).
e_matrix	A matrix of dimensions (n_individuals, n_intervals), where each element contains the evaluation of the temporal integral performed by the function time_int_eval.
n_points	An integer specifying the number of points at which to evaluate the log-likelihood function. A value that is neither too small nor too high is recommended; the default is 150.
cex	A numeric value specifying the size of the points used for the graphical representation of the log-likelihood function. Default is 0.7.
cex_max	A numeric value indicating the size of the optimal point (the one maximizing the log-likelihood function). Default is 0.8.
color_bg	A string specifying the color for the points representing the log-likelihood trend. Default is 'black'.
color_max_bg	A string specifying the color for the optimal point provided as the first argument. Default is 'red'.
pch	A numeric or character symbol representing the shape of the plotted points. Default is a circle (21).

Value

A plot displaying the trend of the log-likelihood function concerning a single parameter, including the maximum point.

`plot_ll_1D.AdPaik` *Plot the One-Dimensional Log-Likelihood Function*

Description

This function plots the trend of the log-likelihood function with respect to a single parameter specified by its index in the parameter vector. It generates samples of the parameter, evaluates them in the log-likelihood function, and displays the results along with the maximum point of the one-dimensional log-likelihood function.

Usage

```
plot_ll_1D.AdPaik(
  param_1D,
  index_param_1D,
  ll_1D,
  params,
  param_range_min,
  param_range_max,
  dataset,
  centre,
  time_axis,
  dropout_matrix,
  e_matrix,
  n_points = 150,
  cex = 0.7,
  cex_max = 0.8,
  color_bg = "black",
  color_max_bg = "red",
  pch = 21
)
```

Arguments

`param_1D`	A numeric value representing the optimal parameter determined by maximizing the log-likelihood function for the specified parameter.
`index_param_1D`	An integer representing the index of the optimal parameter within the parameter vector.
`ll_1D`	A numeric value of the log-likelihood function evaluated at the optimal parameter `param_1D`, with other parameters held constant.
`params`	A numeric vector of length equal to the number of parameters minus one, containing fixed values for the other parameters.

`param_range_min`
: A numeric value indicating the minimum allowable value for the parameter `param_1D`.

`param_range_max`
: A numeric value indicating the maximum allowable value for the parameter `param_1D`.

`dataset` A data frame or matrix containing individual covariates.

`centre` A numeric vector indicating individual cluster membership; its length must match the number of individuals in the dataset.

`time_axis` A numeric vector corresponding to the subdivisions of the temporal domain.

`dropout_matrix` A binary matrix indicating which interval of the time domain an individual failed. Each row should sum to 1 (if failed) or 0 (if not failed), with dimensions (n_individuals, n_intervals).

`e_matrix` A matrix of dimensions (n_individuals, n_intervals) where each element contains the evaluation of the temporal integral performed by the function `time_int_eval`.

`n_points` An integer specifying the number of points at which to evaluate the log-likelihood function. A value that is neither too small nor too high is recommended; the default is 150.

`cex` A numeric value specifying the size of the points used for the graphical representation of the log-likelihood function. Default is 0.7.

`cex_max` A numeric value indicating the size of the optimal point (the one maximizing the log-likelihood function). Default is 0.8.

`color_bg` A string specifying the color for the points representing the log-likelihood trend. Default is `'black'`.

`color_max_bg` A string specifying the color for the optimal point provided as the first argument. Default is `'red'`.

`pch` A numeric or character symbol representing the shape of the plotted points. Default is a circle (`21`).

Value

A plot displaying the trend of the log-likelihood function with respect to a single parameter, including the maximum point.

`plot_post_frailty_est` *Plot the Posterior Frailty Estimates*

Description

This function plots the posterior frailty estimates for each group in each time interval (represented by its mid point). Each group's estimates are represented by a sequence of points connected by straight lines. The function can plot either the entire posterior frailty estimate or its time-independent and time-dependent components based on user-specified flags.

Usage

```
plot_post_frailty_est(
  result,
  data,
  flag_eps = FALSE,
  flag_alpha = FALSE,
  xlim = NULL,
  ylim = NULL,
  xlab = "Time",
  ylab = "Values",
  main = "Posterior frailty estimates",
  cex = 0.7,
  pch_type = rep(21, length(centre_codes)),
  color_bg = rep("black", length(centre_codes)),
  cex_legend = 0.7,
  pos_legend = "topright"
)
```

Arguments

result	S3 object of class 'AdPaik', returned by the method call 'AdPaikModel(...)'.
data	Dataset (dataframe) in which all variables of the formula object must be found and contained.
flag_eps	Logical flag indicating whether to plot only the time-dependent posterior frailty estimates. Default is FALSE.
flag_alpha	Logical flag indicating whether to plot only the time-independent posterior frailty estimates. Default is FALSE.
xlim	A numeric vector specifying the range for the x-axis (intervals). If NULL, default is set to the interval min-max of the time-domain, plus space for the legend. If flag_alpha = TRUE, the plot is produced around 1 (defaults to 0.8-1.4).
ylim	A numeric vector specifying the range for the y-axis (intervals). If NULL, default is min-max value of the posterior frailty estimate.
xlab, ylab	String giving the x and y axis name. Default values are 'Time' and 'Values'.
main	Title of the plot. Default title is 'Posterior frailty estimates'.
cex	Dimension of the points used for plotting the estimates.
pch_type	Numerical vector of length equal to the number of clusters in the data, giving the symbol to be used for plotting the estimates. Default symbol (circle, 21) is the same for all clusters.
color_bg	Numerical vector of length equal to the number of clusters in the data, giving the color to be used for plotting the symbols for the estimates. Default ('black') is the same for all faculties. On the other hand, the same color is used throughout the intervals for the same faculty.
cex_legend	Dimension of the symbol in the legend. Default is 0.7.
pos_legend	Either a numeric vector providing the x and y coordinates for the legend or a string specifying the legend's position (e.g., 'bottomright', 'bottom', 'bottomleft', 'left', 'topleft', 'top', 'topright', 'right', 'center').

Details

Recalling the frailty structure as $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$ and the posterior frailty estimate as $\hat{Z}_{jk} = \hat{\alpha}_j/\hat{\alpha}_{max} + \hat{\epsilon}_{jk}/\hat{\epsilon}_{max}$, this function allows plotting either the entire posterior frailty estimate $\hat{Z}_{jk}$ or its time-independent $\frac{\hat{\alpha}_j}{\hat{\alpha}_{\max}}$ or time-dependent $\frac{\hat{\epsilon}_{jk}}{\hat{\epsilon}_{\max}}$ components. The user can control which components to display using the flag_eps and flag_alpha parameters. Only one of these flags can be set to TRUE at a time.

Value

The plot of the posterior frailty estimates.

Examples

```
# Import data
data(data_dropout)

# Define the variables needed for the model execution
eps_paik <- 1e-10
categories_range_min <- c(-8, -2, eps_paik, eps_paik, eps_paik)
categories_range_max <- c(-eps_paik, 0.4, 1 - eps_paik, 1, 10)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
formula <- time_to_event ~ Gender + CFUP + cluster(group)

# Call the main model function


result <- AdPaikModel(formula, data_dropout, time_axis, categories_range_min, categories_range_max)

# Define variables for plotting the estimates
pch_type <- c(21, seq(21,25,1), seq(21,25,1), seq(21,25,1))
color_bg <- c("darkblue", rep("red", 5), rep("purple", 5), rep("green",5))

plot_post_frailty_est(result, data_dropout,
                      pch_type = pch_type, color_bg = color_bg)
```

---

plot_survival *Plot of Conditional Survival Function*

Description

Plots the conditional survival function based on the 'Adapted Paik et al.' model's estimated coefficients and frailty effects, for each unit in each time interval (represented by its mid point).

Usage

```
plot_survival(
  result,
  survival_df,
  lwd = 1,
  xlim = c(min(result$TimeDomain), max(result$TimeDomain)),
  ylim = c(0, 1),
  xlab = "Time",
  ylab = "Values",
  main = "Conditional Survival",
  cex = 0.2,
  cexlegend = 0.8
)
```

Arguments

result	S3 object of class 'AdPaik' containing model results.
survival_df	The dataframe returned by 'survival' function, where each row corresponds to the survival function values over the time intervals for each individual in the dataset. The first column represents the cluster to which the individual belongs.
lwd	The line width of the plot. Default is 1.
xlim	A numeric vector specifying the range for the x-axis (intervals). Default is min-max value of the time domain.
ylim	A numeric vector specifying the range for the y-axis (intervals). Default is the range 0-1.
xlab, ylab	String giving the x and y axis name. Default values are 'Time' and 'Values'.
main	Title of the plot. Default title is 'Survival'.
cex	Dimension of the points used for plotting the estimates. Defaults to 0.2.
cexlegend	Dimension of the text used for the legend. Defaults to 0.9.

Value

The plot of the conditional survival function.

Examples

```
# Import data
data(data_dropout)

# Define the variables needed for the model execution
eps_paik <- 1e-10
categories_range_min <- c(-8, -2, eps_paik, eps_paik, eps_paik)
categories_range_max <- c(-eps_paik, 0.4, 1 - eps_paik, 1, 10)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
formula <- time_to_event ~ Gender + CFUP + cluster(group)

# Call the main model function
```

```
result <- AdPaikModel(formula, data_dropout, time_axis, categories_range_min, categories_range_max)

survival_df = survival(result, data_dropout)
plot_survival(result, survival_df)
```

`post_frailty.AdPaik` *Posterior frailty estimates and variances for the 'Adapted Paik et al.'s Model'*

Description

Function for computing the posterior frailty estimates and variances of the time-dependent shared frailty Cox model. Recalling the structure of the frailty $Z_{jk} = \alpha_j + \epsilon_{jk}, \forall j, k$ as being composed by the sum of two independent gamma distributions:

- $\alpha_j \sim gamma(\mu_1/\nu, 1/\nu), \forall j$
- $\epsilon_{jk} \sin gamma(\mu_2/\gamma_k, 1/\gamma_k), \forall j, k$ the posterior distribution of both terms is still a gamma with different mean and variance and the posterior frailty estimate corresponds to the 'empirical Bayes estimate', that is the previous mentioned posterior mean.

Usage

```
post_frailty.AdPaik(optimal_params, dataset, time_to_event, centre, time_axis)
```

Arguments

`optimal_params`	Optimal parameters estimated by maximizing the log-likelihood function, through the constraint multi-dimensional optmization method.
`dataset`	Dataset containing all the covariates/regressors.
`time_to_event`	Time-instant, in the follow-up, in which an individual faces the event or fails. If an individual does not face the event in the follow-up, then the time-instant must assume a default value.
`centre`	Individual group/cluster membership.
`time_axis`	Temporal domain.

Value

S3 object of class 'PF.AdPaik' composed of two elements of different class:

- PosteriorFrailtyEst: S3 object of class 'PFE.AdPaik'.
- PosteriorFrailtyVar: S3 object of class 'PFV.AdPaik'.

`post_frailty_CI.AdPaik` *Confidence interval for posterior frailty estimates*

Description

Function for computing the confidence interval for each posterior frailty estimates $\hat{Z}_{jk}$.

Usage

```
post_frailty_CI.AdPaik(
  post_frailty_est,
  post_frailty_est_var,
  n_centres,
  n_intervals,
  level
)
```

Arguments

`post_frailty_est`	Posterior frailty estimates list.
`post_frailty_est_var`	Posterior frailty variance list.
`n_centres`	Number of clusters/centres.
`n_intervals`	Number of intervals of the time-domain. it is equal to the length of the tima_axis minus one.
`level`	A numeric value representing the confidence level.

Value

S3 object of class 'PFCI.AdPaik' composed of two matrices of dimension (number groups, number of intervals):

- PostFrailtyCI_left: left confidence interval for each posterior frailty estimates
- PostFrailtyCI_right: right confidence interval for each each posterior frailty estimates

summary *Summary for Time-Dependent Frailty Models*

Description

This function displays a summary of the model output based on the class of the result object. It delegates to the appropriate summary method according to the class of the result.

Usage

```
summary(result)
```

Arguments

result	An object containing the output of the model call. The class of this object determines which summary method is invoked.

Value

A summary of the model output printed to the console.

summary.AdPaik *Summary of the Adapted Paik et al.'s Time-Dependent Shared Frailty Model*

Description

This function provides a comprehensive summary of the results from the Adapted Paik et al.'s Time-Dependent Shared Frailty Model. It includes key information about the dataset (e.g., number of individuals, regressors, intervals, and clusters), model parameters, and output (log-likelihood, AIC). The summary also lists the estimated regressors along with their standard errors

Usage

```
## S3 method for class 'AdPaik'
summary(result)
```

Arguments

result	'S3' class object returned by the main model call, i.e. output of the 'Adapted Paik et al.'s Model'.

Details

The function reports the estimated regressors, their standard errors, and confidence intervals (if available).

Value

Model summary printed on output.

Examples

```
# Define the variables needed for the model execution

data(data_dropout)
eps_paik <- 1e-10
categories_range_min <- c(-8, -2, eps_paik, eps_paik, eps_paik)
categories_range_max <- c(-eps_paik, 0.4, 1 - eps_paik, 1, 10)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
formula <- time_to_event ~ Gender + CFUP + cluster(group)


# Call the main model function
result <- AdPaikModel(formula, data_dropout, time_axis, categories_range_min, categories_range_max)

# Call the summary
summary(result)
```

---

survival *Compute the Conditional Survival Function*

Description

Computes the conditional survival function based on the 'Adapted Paik et al.' model's given the estimated coefficients and frailty effects.

Usage

```
survival(result, data)
```

Arguments

result	S3 object of class 'AdPaik' containing model results.
data	Data frame containing covariates used in the model.

Value

A dataset where each row corresponds to an individual unit in the dataset, and the columns represent the survival function values over time interval, with the first column indicating the cluster to which the individual belongs.

Examples

```
# Import data
data(data_dropout)

# Define the variables needed for the model execution
eps_paik <- 1e-10
categories_range_min <- c(-8, -2, eps_paik, eps_paik, eps_paik)
categories_range_max <- c(-eps_paik, 0.4, 1 - eps_paik, 1, 10)
time_axis <- c(1.0, 1.4, 1.8, 2.3, 3.1, 3.8, 4.3, 5.0, 5.5, 5.8, 6.0)
formula <- time_to_event ~ Gender + CFUP + cluster(group)

# Call the main model function

result <- AdPaikModel(formula, data_dropout, time_axis, categories_range_min, categories_range_max)

survival_df = survival(result, data_dropout)
```

---

time_int_eval *Resolution of integral with respect to time*

Description

Function for the resolution of an integral with respect to time, in the evaluation of the log-likelihood function. It is implemented as defined in the paper of Wintrebert et al.'s (2004)

Usage

```
time_int_eval(time_t, k, time_axis)
```

Arguments

time_t	Event time instant
k	k-th interval of the time-axis
time_axis	Temporal domain (it may coincide with the follow-up)

Value

Evaluation of the temporal integral

Index