# Package ‘frailtypack’

May 8, 2026

**Title** Shared, Joint (Generalized) Frailty Models; Surrogate Endpoints

**Version** 3.8.0

**Date** 2025-11-20

**Author** Virginie Rondeau [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7109-4831>),
Juan R. Gonzalez [aut],
Yassin Mazroui [aut],
Audrey Mauguen [aut],
Amadou Diakite [aut],
Alexandre Laurent [aut],
Myriam Lopez [aut],
Agnieszka Krol [aut],
Casimir L. Sofeu [aut],
Julien Dumerc [aut],
Denis Rustand [aut],
Jocelyn Chauvet [aut],
Quentin Le Coent [aut],
Romain Pierlot [aut],
Lacey Etzkorn [aut],
Derek Dinart [aut],
Adrien Orué [aut],
Ayoub Bifenzi [aut],
Viviane Philipps [aut],
David Hill [cph],
John Burkardt [cph],
Alan Genz [cph],
Ashwith J. Rego [cph]

**Maintainer** Virginie Rondeau <virginie.rondeau@u-bordeaux.fr>

**Description** The following several classes of frailty models using a penalized likelihood estimation on the hazard function but also a parametric estimation can be fit using this R package: 1) A shared frailty model (with gamma or log-normal frailty distribution) and Cox proportional hazard model. Clustered and recurrent survival times can be studied. 2) Additive frailty models for proportional hazard models with two correlated random effects (intercept random effect with random slope). 3) Nested frailty models for hierarchically clustered data (with 2 levels of clustering) by including two iid gamma random effects. 4) Joint frailty models in the context of the joint modelling for recurrent events with terminal event for clustered data or not. A joint frailty model for two semi-competing risks and clustered data is also proposed. 5) Joint general frailty models in the context of the joint modelling for recurrent events with terminal event data with two independent frailty terms. 6) Joint Nested frailty models in the context of the joint modelling for recurrent events with terminal event, for hierarchically clustered data (with two levels of clustering) by including two iid gamma random effects. 7) Multivariate joint frailty models for two types of recurrent events and a terminal event. 8) Joint models for longitudinal data and a terminal event. 9) Trivariate joint models for longitudinal data, recurrent events and a terminal event. 10) Joint frailty models for the validation of surrogate endpoints in multiple randomized clinical trials with failure-time and/or longitudinal endpoints with the possibility to use a mediation analysis model. 11) Conditional and Marginal two-part joint models for longitudinal semicontinuous data and a terminal event. 12) Joint frailty-copula models for the validation of surrogate endpoints in multiple randomized clinical trials with failure-time endpoints. 13) Generalized shared and joint frailty models for recurrent and terminal events. Proportional hazards (PH), additive hazard (AH), proportional odds (PO) and probit models are available in a fully parametric framework. For PH and AH models, it is possible to consider type-varying coefficients and flexible semiparametric hazard function. Prediction values are available (for a terminal event or for a new recurrent event). Left-truncated (not for Joint model), right-censored data, interval-censored data (only for Cox proportional hazard and shared frailty model) and strata are allowed. In each model, the random effects have the gamma or normal distribution. Now, you can also consider time-varying covariates effects in Cox, shared and joint frailty models (1-5). The package includes concordance measures for Cox proportional hazards models and for shared frailty models. 14) Competing Joint Frailty Model: A single type of recurrent event and two terminal events. 15) functions to compute power and sample size for four Gamma-frailty-based designs: Shared Frailty Models, Nested Frailty Models, Joint Frailty Models, and General Joint Frailty Models. Each design includes two primary functions: a power function, which computes power given a specified sample size; and a sample size function, which computes the required sample size to achieve a specified power. 16) Weibull Illness-Death model with or without shared frailty between transitions. Left-truncated and right-censored data are allowed. 17) Weibull Competing risks model with or without shared frailty between the transitions. Left-truncated and right-censored data are allowed. Moreover, the package can be used with its shiny application, in a local mode or by following the link below.

**License** GPL (>= 2.0)

**URL** https://virginie1rondeau.wixsite.com/virginierondeau/software-frailtypack
https://frailtypack-pkg.shinyapps.io/shiny_frailtypack/

**Depends** boot, doBy, MASS, R (>= 3.5.0), survC1, survival

**Imports** graphics, matrixcalc, nlme, rootSolve, shiny, splines, statmod, stats, utils, dplyr, marqLevAlg, tidyr,

**Suggests** knitr, timereg, rmarkdown

**VignetteBuilder** knitr

**Date/Publication** 2025-11-22 23:30:02 UTC

**Encoding** UTF-8

**LazyLoad** no

**NeedsCompilation** yes

**Repository** CRAN

**RoxygenNote** 7.3.3

# Contents

---

`frailtypack-package` *General Frailty models: shared, joint and nested frailty models with prediction; Evaluation of Failure-Time Surrogate Endpoints*

---

## Description

Frailtypack fits several classes of frailty models using a penalized likelihood estimation on the hazard function but also a parametric estimation.

**Details**

The following several classes of frailty models using a penalized likelihood estimation on the hazard function but also a parametric estimation can be fit using this R package:

1) A shared frailty model (with gamma or log-normal frailty distribution) and Cox proportional hazard model. Clustered and recurrent survival times can be studied.

2) Additive frailty models for proportional hazard models with two correlated random effects (intercept random effect with random slope).

3) Nested frailty models for hierarchically clustered data (with 2 levels of clustering) by including two iid gamma random effects.

4) Joint frailty models in the context of the joint modelling for recurrent events with terminal event for clustered data or not. A joint frailty model for two semi-competing risks and clustered data is also proposed.

5) Joint general frailty models in the context of the joint modelling for recurrent events with terminal event data with two independent frailty terms.

6) Joint Nested frailty models in the context of the joint modelling for recurrent events with terminal event, for hierarchically clustered data (with two levels of clustering) by including two iid gamma random effects.

7) Multivariate joint frailty models for two types of recurrent events and a terminal event.

8) Joint models for longitudinal data and a terminal event.

9) Trivariate joint models for longitudinal data, recurrent events and a terminal event.

10) Joint frailty models for the validation of surrogate endpoints in multiple randomized clinical trials with failure-time and/or longitudinal endpoints with the possibility to use a mediation analysis model.

11) Conditional and Marginal two-part joint models for longitudinal semicontinuous data and a terminal event.

12) Joint frailty-copula models for the validation of surrogate endpoints in multiple randomized clinical trials with failure-time endpoints.

13) Generalized shared and joint frailty models for recurrent and terminal events. Proportional hazards (PH), additive hazard (AH), proportional odds (PO) and probit models are available in a fully parametric framework.

14) Competing Joint Frailty Model: A single type of recurrent event and two terminal events.

15) Functions to compute power and sample size for four Gamma-frailty-based designs: Shared Frailty Models, Nested Frailty Models, Joint Frailty Models, and General Joint Frailty Models. Each design includes two primary functions: a power function, which computes power given a specified sample size; and a sample size function, which computes the required sample size to achieve a specified power.

The package includes concordance measures for Cox proportional hazards models and for shared frailty models. Now, you can also consider time-varying covariates effects in Cox, shared and joint frailty models (1-5). Some of the Fortran routines in the package can speed-up computation time by making use of parallelization through OpenMP. Moreover, the package can be used with its shiny application, in a local mode or by following the link below.

| | |
|---|---|
| Package: | frailtypack |
| Type: | Package |
| Version: | 3.7.0 |
| Date: | 2025-03-06 |
| License: | GPL (>= 2.0) |
| LazyLoad: | no |

### Author(s)

Virginie Rondeau, Juan R. Gonzalez, Yassin Mazroui, Audrey Mauguen, Amadou Diakite, Alexandre Laurent, Myriam Lopez, Agnieszka Krol, Casimir L. Sofeu, Denis Rustand, Quentin Le Coent, Lacey Etzkorn, Romain Pierlot, Derek Dinart and Adrien Orué

### References

Derek Dinart, Carine Bellera & Virginie Rondeau (2024). Sample size estimation for recurrent event data using multifrailty and multilevel survival models. *Journal of Biopharmaceutical Statistics*, DOI: 10.1080/10543406.2024.2310306.

V. Rondeau, Y. Mazroui and J. R. Gonzalez (2012). Frailtypack: An R package for the analysis of correlated survival data with frailty models using penalized likelihood estimation or parametric estimation. *Journal of Statistical Software* **47**, 1-28.

Y. Mazroui, S. Mathoulin-Pelissier,P. Soubeyranb and Virginie Rondeau (2012) General joint frailty model for recurrent event data with a dependent terminalevent: Application to follicular lymphoma data. *Statistics in Medecine*, **31**, 11-12, 1162-1176.

V. Rondeau and J. R. Gonzalez (2005). Frailtypack: A computer program for the analysis of correlated failure time data using penalized likelihood estimation. *Computer Methods and Programs in Biomedicine* **80**, 2, 154-164.

V. Rondeau, S. Michiels, B. Liquet, and J. P. Pignon (2008). Investigating trial and treatment heterogeneity in an individual patient data meta-analysis of survival data by mean of the maximum penalized likelihood approach. *Statistics in Medecine*, **27**, 1894-1910.

V. Rondeau, S. Mathoulin-Pellissier, H. Jacqmin-Gadda, V. Brouste, P. Soubeyran (2007). Joint frailty models for recurring events and death using maximum penalized likelihood estimation:application on cancer events. *Biostatistics*, **8**, 4, 708-721.

V. Rondeau, D. Commenges, and P. Joly (2003). Maximum penalized likelihood estimation in a gamma-frailty model. *Lifetime Data Analysis* **9**, 139-153.

D. Marquardt (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematics*, 431-441.

V. Rondeau, L. Filleul, P. Joly (2006). Nested frailty models using maximum penalized likelihood estimation. *Statistics in Medecine*, **25**, 4036-4052.

### Examples

```
## Not run:

###--- Additive model with 1 covariate ---###
```

```
data(dataAdditive)
modAdd <- additivePenal(Surv(t1,t2,event)~
cluster(group)+var1+slope(var1),
correlation=TRUE,data=dataAdditive,
n.knots=8,kappa=10000,hazard="Splines")

###--- Joint model (recurrent and terminal events) with 2 covariates ---###

data(readmission)
modJoint.gap <- frailtyPenal(Surv(time,event)~
cluster(id)+sex+dukes+charlson+terminal(death),
formula.terminalEvent=~sex+dukes+charlson,
data=readmission,n.knots=10,kappa=c(100,100),
recurrentAG=FALSE,hazard="Splines")

###--- General Joint model (recurrent and terminal events) with 2 covariates ---###
data(readmission)
modJoint.general <- frailtyPenal(Surv(time,event) ~ cluster(id) + dukes +
charlson + sex + chemo + terminal(death),
formula.terminalEvent = ~ dukes + charlson + sex + chemo,
data = readmission, jointGeneral = TRUE, n.knots = 8,
kappa = c(2.11e+08, 9.53e+11))

###--- Nested model (or hierarchical model) with 2 covariates ---###

data(dataNested)
modClu <- frailtyPenal(Surv(t1,t2,event)~
cluster(group)+subcluster(subgroup)+cov1+cov2,
data=dataNested,n.knots=8,kappa=50000,hazard="Splines")

###--- Joint Nested Frailty model ---###

#-- here is generated cluster (30 clusters)
readmissionNested <- transform(readmission,group=id%%30+1)

modJointNested_Splines <- frailtyPenal(formula = Surv(t.start, t.stop, event)
~ subcluster(id) + cluster(group) + dukes + terminal(death),
formula.terminalEvent = ~dukes, data = readmissionNested, recurrentAG = TRUE,
n.knots = 8, kappa = c(9.55e+9, 1.41e+12), initialize = TRUE)

modJointNested_Weib <- frailtyPenal(Surv(t.start,t.stop,event)~subcluster(id)
+cluster(group)+dukes+ terminal(death),formula.terminalEvent=~dukes,
hazard = ('Weibull'), data=readmissionNested,recurrentAG=TRUE, initialize = FALSE)

JoiNes-GapSpline <- frailtyPenal(formula = Surv(time, event)
~ subcluster(id) + cluster(group) + dukes + terminal(death),
formula.terminalEvent = ~dukes, data = readmissionNested, recurrentAG = FALSE,
n.knots = 8, kappa = c(9.55e+9, 1.41e+12), initialize = TRUE,
init.Alpha = 1.091, Ksi = "None")

###--- Semiparametric Shared model ---###
```

```
data(readmission)
sha.sp <- frailtyPenal(Surv(t.start,t.stop,event)~
sex+dukes+charlson+cluster(id),data=readmission,
n.knots=6,kappa=5000,recurrentAG=TRUE,
cross.validation=TRUE,hazard="Splines")

###--- Parametric Shared model ---###

data(readmission)
sha.p <- frailtyPenal(Surv(t.start,t.stop,event)~
cluster(id)+sex+dukes+charlson,
data=readmission,recurrentAG=TRUE,
hazard="Piecewise-per",nb.int=6)

###--- Joint model for longitudinal ---###
###--- data and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

model.weib.RE <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
colorectalSurv,data.Longi = colorectalLongi,
random = c("1", "year"), id = "id", link = "Random-effects",
left.censoring = -3.33, hazard = "Weibull")

###--- Trivariate joint model for longitudinal ---###
###--- data, recurrent and terminal events ---###

data(colorectal)
data(colorectalLongi)

# (computation takes around 40 minutes)

model.spli.RE.cal <-trivPenal(Surv(time0, time1, new.lesions) ~ cluster(id)
+ age + treatment + who.PS +  terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = TRUE,
n.knots = 6, kappa=c(0.01, 2), method.GH="Pseudo-adaptive",
n.nodes=7, init.B = c(-0.07, -0.13, -0.16, -0.17, 0.42, #recurrent events covariates
-0.23, -0.1, -0.09, -0.12, 0.8, -0.23, #terminal event covariates
3.02, -0.30, 0.05, -0.63, -0.02, -0.29, 0.11, 0.74)) #biomarker covariates


##---Surrogacy evaluation based on ganerated data with a combination
##of Monte Carlo and classical Gaussian Hermite integration.
## (Computation takes around 5 minutes)
```

```
# Generation of data to use
data.sim <- jointSurrSimul(n.obs=600, n.trial = 30,cens.adm=549.24,
         alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1, sigma.s = 0.7,
         sigma.t = 0.7, rsqrt = 0.8, betas = -1.25, betat = -1.25,
         full.data = 0, random.generator = 1, seed = 0, nb.reject.data = 0)

# Joint surrogate model estimation
joint.surro.sim.MCGH <- jointSurroPenal(data = data.sim, int.method = 2,
                   nb.mc = 300, nb.gh = 20)


## End(Not run)
```

---

`additivePenal` *Fit an Additive Frailty model using a semiparametric penalized likelihood estimation or a parametric estimation*

---

## Description

Fit an additive frailty model using a semiparametric penalized likelihood estimation or a parametric estimation. The main issue in a meta-analysis study is how to take into account the heterogeneity between trials and between the treatment effects across trials. Additive models are proportional hazard model with two correlated random trial effects that act either multiplicatively on the hazard function or in interaction with the treatment, which allows studying for instance meta-analysis or multicentric datasets. Right-censored data are allowed, but not the left-truncated data. A stratified analysis is possible (maximum number of strata = 2). This approach is different from the shared frailty models.

In an additive model, the hazard function for the $j^{th}$ subject in the $i^{th}$ trial with random trial effect $u_i$ as well as the random treatment-by-trial interaction $v_i$ is:

$$\begin{cases} \lambda_{ij}(t|u_i,v_i) = \lambda_0(t)exp(u_i + v_iX_{ij1} + \sum_{k=1}^{p}\beta_kX_{ijk}) \\ \mathbf{cov}(u_i,v_i) = \boldsymbol{\rho\sigma\tau} \\ u_i \sim \boldsymbol{\mathcal{N}}(0,\boldsymbol{\sigma^2}), v_i \sim \boldsymbol{\mathcal{N}}(0,\boldsymbol{\tau^2}) \end{cases}$$

where $\lambda_0(t)$ is the baseline hazard function, $\beta_k$ the fixed effect associated to the covariate $X_{ijk}$ (k=1,..,p), $\beta_1$ is the treatment effect and $X_{ij1}$ the treatment variable. $\rho$ is the corresponding correlation coefficient for the two frailty terms.

## Usage

```
additivePenal(formula, data, correlation = FALSE, recurrentAG =
FALSE, cross.validation = FALSE, n.knots, kappa, maxit = 350, hazard =
"Splines", nb.int, LIMparam = 1e-4, LIMlogl = 1e-4, LIMderiv = 1e-3,
print.times = TRUE,init.hazard.weib)
```

## Arguments

`formula` a formula object, with the response on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. The `slope()` function is required. Interactions are possible using * or :.

`data` a 'data.frame' with the variables used in 'formula'.

`correlation` Logical value. Are the random effects correlated? If so, the correlation coefficient is estimated. The default is FALSE.

`recurrentAG` Always FALSE for additive models (left-truncated data are not allowed).

`cross.validation`
Logical value. Is cross validation procedure used for estimating smoothing parameter in the penalized likelihood estimation? If so a search of the smoothing parameter using cross validation is done, with kappa as the seed. The cross validation is not implemented for two strata. The default is FALSE.

`n.knots` integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. Number of knots must be between 4 and 20. (See Note)

`kappa` positive smoothing parameter in the penalized likelihood estimation. In a stratified additive model, this argument must be a vector with kappas for both strata. The coefficient kappa of the integral of the squared second derivative of hazard function in the fit. To obtain an initial value for `kappa`, a solution is to fit the corresponding shared frailty model using cross validation (see cross.validation). We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them. Value required. (See Note)

`maxit` maximum number of iterations for the Marquardtt algorithm. Default is 350

`hazard` Type of hazard functions: "Splines" for semiparametric hazard functions with the penalized likelihood estimation, "Piecewise-per" for piecewise constant hazards functions using percentile, "Piecewise-equi" for piecewise constant hazard functions using equidistant intervals, "Weibull" for parametric Weibull functions. Default is "Splines".

`nb.int` Number of intervals (between 1 and 20) for the parametric hazard functions ("Piecewise-per", "Piecewise-equi").

`LIMparam` Convergence threshold of the Marquardt algorithm for the parameters (see Details), $10^{-4}$ by default.

`LIMlogl` Convergence threshold of the Marquardt algorithm for the log-likelihood (see Details), $10^{-4}$ by default.

`LIMderiv` Convergence threshold of the Marquardt algorithm for the gradient (see Details), $10^{-3}$ by default.

`print.times` a logical parameter to print iteration process. Default is TRUE.

`init.hazard.weib`
If a weibull model is used, initialization values for hazard parameters.

## Details

The estimated parameter are obtained by maximizing the penalized log-likelihood or by a simple log-likelihood (in the parametric case) using the robust Marquardtt algorithm (Marquardtt,1963). The parameters are initialized with values obtained with Cox proportional hazard model. The iterations are stopped when the difference between two consecutive loglikelhoods was small ($< 10^{-4}$), the estimated coefficients were stable (consecutive values ($< 10^{-4}$), and the gradient small enough ($< 10^{-3}$). To be sure of having a positive function at all stages of the algorithm, the spline coefficients were reparametrized to be positive at each stage. The variance space of the two random effects is reduced, so the variances are positive, and the correlation coefficient values are constrained to be between -1 and 1. The marginal log-likelihood depends on integrations that are approximated by using the Laplace integration technique with a first order approximation. The smoothing parameter can be fixed or estimated by maximizing likelihood cross-validation criterion. The usual squared Wald statistic was modified to a mixture of two $\chi^2$ distribution to get significance test for the variance of the random effects.

**INITIAL VALUES**

The splines and the regression coefficients are initialized to 0.1. An adjusted Cox model is fitted, it provides new initial values for the splines coefficients and the regression coefficients. The variances of the frailties are initialized to 0.1. Then an additive frailty model with independent frailties is fitted. At last, an additive frailty model with correlated frailties is fitted.

## Value

An additive model or more generally an object of class 'additivePenal'. Methods defined for 'additivePenal' objects are provided for print, plot and summary.

| | |
|---|---|
| `b` | sequence of the corresponding estimation of the splines coefficients, the random effects variances and the regression coefficients. |
| `call` | The code used for fitting the model. |
| `coef` | the regression coefficients. |
| `cov` | covariance between the two frailty terms ($\boldsymbol{cov}(u_i, v_i)$) |
| `cross.Val` | Logical value. Is cross validation procedure used for estimating the smoothing parameters in the penalized likelihood estimation? |
| `correlation` | Logical value. Are the random effects correlated? |
| `DoF` | degrees of freedom associated with the "kappa". |
| `formula` | the formula part of the code used for the model. |
| `groups` | the maximum number of groups used in the fit. |
| `kappa` | A vector with the smoothing parameters in the penalized likelihood estimation corresponding to each baseline function as components. |
| `loglikPenal` | the complete marginal penalized log-likelihood in the semiparametric case. |
| `loglik` | the marginal log-likelihood in the parametric case. |
| `n` | the number of observations used in the fit. |
| `n.events` | the number of events observed in the fit. |
| `n.iter` | number of iterations needed to converge. |

n.knots
: number of knots for estimating the baseline functions.

n.strat
: number of stratum.

rho
: the corresponding correlation coefficient for the two frailty terms.

sigma2
: Variance for the random intercept (the random effect associated to the baseline hazard functions).

tau2
: Variance for the random slope (the random effect associated to the treatment effect across trials).

varH
: the variance matrix of all parameters before positivity constraint transformation (Sigma2, Tau2, the regression coefficients and the spline coefficients). Then after, the delta method is needed to obtain the estimated variance parameters.

varHIH
: the robust estimation of the variance matrix of all parameters (Sigma2, Tau2, the regression coefficients and the spline coefficients).

varSigma2
: The variance of the estimates of "sigma2".

varTau2
: The variance of the estimates of "tau2".

varcov
: Variance of the estimates of "cov".

x
: matrix of times where both survival and hazard functions are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times.

lam
: array (dim=3) of hazard estimates and confidence bands.

surv
: array (dim=3) of baseline survival estimates and confidence bands.

median
: The value of the median survival and its confidence bands. If there are two stratas or more, the first value corresponds to the value for the first strata, etc.

type.of.hazard
: Type of hazard functions (0:"Splines", "1:Piecewise", "2:Weibull").

type.of.Piecewise
: Type of Piecewise hazard functions (1:"percentile", 0:"equidistant").

nbintervR
: Number of intervals (between 1 and 20) for the parametric hazard functions ("Piecewise-per", "Piecewise-equi").

npar
: number of parameters.

nvar
: number of explanatory variables.

noVar
: indicator of explanatory variable.

LCV
: the approximated likelihood cross-validation criterion in the semiparametric case (with H minus the converged Hessian matrix, and l(.) the full log-likelihood).

$$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$

AIC
: the Akaike information Criterion for the parametric case.

$$AIC = \frac{1}{n}(np - l(.))$$

n.knots.temp
: initial value for the number of knots.

shape.weib
: shape parameter for the Weibull hazard function.

scale.weib
: scale parameter for the Weibull hazard function.

| | |
|---|---|
| martingale.res | martingale residuals for each cluster. |
| frailty.pred | empirical Bayes prediction of the first frailty term. |
| frailty.pred2 | empirical Bayes prediction of the second frailty term. |
| linear.pred | linear predictor: uses simply "Beta'X + u_i + v_i * X_1" in the additive Frailty models. |
| global_chisq | a vector with the values of each multivariate Wald test. |
| dof_chisq | a vector with the degree of freedom for each multivariate Wald test. |
| global_chisq.test | a binary variable equals to 0 when no multivariate Wald is given, 1 otherwise. |
| p.global_chisq | a vector with the p_values for each global multivariate Wald test. |
| names.factor | Names of the "as.factor" variables. |
| Xlevels | vector of the values that factor might have taken. |
| contrasts | type of contrast for factor variable. |
| beta_p.value | p-values of the Wald test for the estimated regression coefficients. |

## Note

"kappa" and "n.knots" are the arguments that the user have to change if the fitted model does not converge. "n.knots" takes integer values between 4 and 20. But with n.knots=20, the model would take a long time to converge. So, usually, begin first with n.knots=7, and increase it step by step until it converges. "kappa" only takes positive values. So, choose a value for kappa (for instance 10000), and if it does not converge, multiply or divide this value by 10 or 5 until it converges.

## References

V. Rondeau, Y. Mazroui and J. R. Gonzalez (2012). Frailtypack: An R package for the analysis of correlated survival data with frailty models using penalized likelihood estimation or parametric estimation. *Journal of Statistical Software* **47**, 1-28.

V. Rondeau, S. Michiels, B. Liquet, and J. P. Pignon (2008). Investigating trial and treatment heterogeneity in an individual patient data meta-analysis of survival data by mean of the maximum penalized likelihood approach. *Statistics in Medecine*, **27**, 1894-1910.

## See Also

slope

## Examples

```
###--- Additive model with 1 covariate ---###

data(dataAdditive)

modAdd <- additivePenal(Surv(t1,t2,event)~cluster(group)+
```

```
var1+slope(var1),correlation=TRUE,data=dataAdditive,
n.knots=8,kappa=10000)

#-- Var1 is boolean as a treatment variable
```

---

bcos *Breast Cosmesis Data*

---

## Description

The often used data set for interval-censored data, described and given in full in Finkelstein and Wolfe (1985). It involves 94 breast cancer patients who were randomized to either radiation therapy with chemotherapy or radiation therapy alone. The outcome is time until the onset of breast retraction which is interval-censored between the last clinic visit before the event was observed and the first visit when the event was observed. Patients without breast retraction were right-censored.

## Usage

```
data(bcos)
```

## Format

A data frame with 94 observations and 3 variables:

**left** left end point of the breast retraction interval

**right** right end point of the breast retraction interval

**treatment** type of treatment received

## Source

Finkelstein, D.M. and Wolfe, R.A. (1985). A semiparametric model for regression analysis of interval-censored failure time data. *Biometrics* **41**, 731-740.

cluster *Identify clusters*

## Description

This is a special function used in the context of the models for grouped data. It identifies correlated groups of observations defined by using 'cluster' function, and is used of 'frailtyPenal' formula for fitting univariate and joint models.

## Usage

```
cluster(x)
```

## Arguments

| | |
|---|---|
| x | A character, factor, or numeric variable which is supposed to indicate the variable group |

## Value

| | |
|---|---|
| x | A variable identified as a cluster |

## See Also

frailtyPenal

## Examples

```
data(readmission)
modSha <- frailtyPenal(Surv(time,event)~as.factor(dukes)+cluster(id),
n.knots=10,kappa=10000,data=readmission,hazard="Splines")

print(modSha)
```

`Cmeasures` *Concordance measures in shared frailty and Cox proportional hazard models*

## Description

Compute concordance probability estimation for Cox proportional hazard or shared frailty models in case of grouped data (Mauguen et al. 2012). Concordance is given at different levels of comparison, taking into account the cluster membership: between-groups, within-groups and an overall measure, being a weighted average of the previous two. Can also compute the c-index (Harrell et al. 1996) at these three levels. It is possible to exclude tied pairs from concordance estimation (otherwise, account for 1/2).

## Usage

```
Cmeasures(fitc, ties = 1, marginal = 0, cindex = 0, Nboot = 0, tau = 0,
data.val)
```

## Arguments

| | |
|---|---|
| `fitc` | A frailtyPenal object, for a shared frailty model. If the fit is a Cox model, no clustering membership is taken into account and only marginal concordance probability estimation is provided. Only an overall measure is given, where all patients are compared two by two. If a counting process formulation is used to performed the fit, with 't.start' and 't.stop', the gap-times (t.stop-t.start) are used in the concordance estimation. |
| `ties` | Indicates if the tied pairs on prediction value must be included (ties=1) or excluded (ties=0) from the concordance estimation. Default is ties=1. When included, tied pairs account for 1/2 in the concordance. |
| `marginal` | Indicates if the concordance based on marginal predictions must be given (marginal=1) in addition to conditional ones or not (marginal=0). Marginal predictions do not include the frailty estimation in the linear predictor computation: uses "'Beta'X"' instead of "Beta'X + log z_i". Default is marginal=0. |
| `cindex` | Indicates if the c-index (Harrell et al. 1996) must be computed (cindex=1) in addition to the concordance probability estimation or not (cindex=0). C-index is also given at the three comparison levels (between, within and overall). Default is cindex=0. |
| `Nboot` | Number of bootstrap resamplings to compute standard-error of the concordances measures, as well as a percentile 95% confidence interval. Nboot=0 indicates no bootstrap procedure. Maximum admitted is 1000. Minimum admitted is 2. Default is 0. Resampling is done at the group level. If Cox model is used, resampling is done at individual level. |
| `tau` | Time used to limit the interval on which the concordance is estimated. Note that the survival function for the underlying censoring time distribution needs to be positive at tau. If tau=0, the maximum of the observed event times is used. Default is tau=0. |

`data.val` A dataframe. It is possible to specify a different dataset than the one used in the model input in the argument 'fitc'. This new dataset will be a validation population and the function will compute new concordance measures from the parameters estimated on the development population. In this case for conditional measures, the frailties are a posteriori predicted. The two datasets must have the same covariates with the same coding without missing data.

## Value

`call` The shared frailty model evaluated.

`Frailty` Logical value. Was model with frailties fitted.

`frequencies` Numbers of patients, events and groups used to fit the model.

`Npairs` Number of pairs of subjects, between-groups, within-groups and over all the population. If cindex=1, number of comparable (useable) pairs also available.

`Nboot` Number of bootstrap resamplings required.

`ties` A binary, indicating if the tied pairs on prediction were used to compute the concordance.

`CPEcond` Values of Gonen & Heller's measure (conditional). If Nboot>0, give SE, the standard-error of the parameters evaluated by bootstrap, IC.low and IC.high, the lower and upper bounds of the percentile confidence interval evaluated by bootstrap (2.5% and 97.5% percentiles).

`Cunocond` Values of Uno's measure (conditional). If Nboot>0, give SE, the standard-error of the parameters evaluated by bootstrap, IC.low and IC.high, the lower and upper bounds of the percentile confidence interval evaluated by bootstrap (2.5% and 97.5% percentiles).

`marginal` A binary, indicating if the marginal values were computed.

`CPEmarg` Values of Gonen & Heller's measure (marginal), if marginal=1. If Nboot>0, give SE, the standard-error of the parameters evaluated by bootstrap, IC.low and IC.high, the lower and upper bounds of the percentile confidence interval evaluated by bootstrap (2.5% and 97.5% percentiles).

`Cunomarg` Values of Uno's measure (marginal), if marginal=1. If Nboot>0, give SE, the standard-error of the parameters evaluated by bootstrap, IC.low and IC.high, the lower and upper bounds of the percentile confidence interval evaluated by bootstrap (2.5% and 97.5% percentiles).

`cindex` A binary, indicating if the c-indexes were computed.

`cindexcond` Values of the C-index of Harrell (conditional). If Nboot>0, give SE, the standard-error of the parameters evaluated by bootstrap, IC.low and IC.high, the lower and upper bounds of the percentile confidence interval evaluated by bootstrap (2.5% and 97.5% percentiles).

`cindexmarg` Values of the C-index of Harrell (marginal), if marginal=1. If Nboot>0, give SE, the standard-error of the parameters evaluated by bootstrap, IC.low and IC.high, the lower and upper bounds of the percentile confidence interval evaluated by bootstrap (2.5% and 97.5% percentiles).

### References

Mauguen, A., Collette, S., Pignon, J. P. and Rondeau, V. (2013). Concordance measures in shared frailty models: application to clustered data in cancer prognosis. *Statistics in Medicine* **32**, 27, 4803-4820

Harrell, F.E. et al. (1996). Tutorial in biostatistics: multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine* **15**, 361-387.

Gonen, M., Heller, G. (2005). Concordance probability and discriminatory power in proportional hazards regression. *Biometrika* **92**, 965-970.

### See Also

`print.Cmeasures`,`frailtyPenal`

### Examples

```
#-- load data
data(readmission)

#-- a frailtypenal fit
fit <- frailtyPenal(Surv(time,event)~cluster(id)+dukes+
charlson+chemo,data=readmission,cross.validation=FALSE,
n.knots=10,kappa=1,hazard="Splines")

#-- a Cmeasures call
fit.Cmeasures <- Cmeasures(fit)
fit.Cmeasures.noties <- Cmeasures(fit, ties=0)
fit.Cmeasures.marginal <- Cmeasures(fit, marginal=1)
fit.Cmeasures.cindex <- Cmeasures(fit, cindex=1)

#-- a short summary
fit.Cmeasures
fit.Cmeasures.noties
fit.Cmeasures.marginal
fit.Cmeasures.cindex
```

---

`colorectal` *Follow-up of metastatic colorectal cancer patients: times of new lesions appearance and death*

---

## Description

Randomly chosen 150 patients from the follow-up of the FFCD 2000-05 multicenter phase III clinical trial originally including 410 patients with metastatic colorectal cancer randomized into two therapeutic strategies: combination and sequential. The dataset contains times of observed appearances of new lesions censored by a terminal event (death or right-censoring) with baseline characteristics (treatment arm, age, WHO performance status and previous resection).

## Usage

```
data(colorectal)
```

## Format

This data frame contains the following columns:

**id** identification of each subject. Repeated for each recurrence

**time0** start of interval (0 or previous recurrence time)

**time1** recurrence or censoring time

**new.lesions** Appearance of new lesions status. 0: censsored or no event, 1: new lesions

**treatment** To which treatment arm a patient was allocated? 1: sequential (S); 2: combination (C)

**age** Age at baseline: 1: <50 years, 2: 50-69 years, 3: >69 years

**who.PS** WHO performance status at baseline: 1: status 0, 2: status 1, 3: status 2

**prev.resection** Previous resection of the primate tumor? 0: No, 1: Yes

**state** death indicator. 0: alive, 1: dead

**gap.time** interocurrence time or censoring time

## Note

We thank the Federation Francophone de Cancerologie Digestive and Gustave Roussy for sharing the data of the FFCD 2000-05 trial supported by an unrestricted Grant from Sanofi.

## References

M. Ducreux, D. Malka, J. Mendiboure, P.-L. Etienne, P. Texereau, D. Auby, P. Rougier, M. Gasmi, M. Castaing, M. Abbas, P. Michel, D. Gargot, A. Azzedine, C. Lombard- Bohas, P. Geoffroy, B. Denis, J.-P., Pignon, L.,Bedenne, and O. Bouche (2011). Sequential versus combination chemotherapy for the treatment of advanced colorectal cancer (FFCD 2000-05): an open-label, randomised, phase 3 trial. *The Lancet Oncology* **12**, 1032-44.

`colorectalLongi` *Follow-up of metastatic colorectal cancer patients : longitudinal measurements of tumor size*

## Description

Randomly chosen 150 patients from the follow-up of the FFCD 2000-05 multicenter phase III clinical trial originally including 410 patients with metastatic colorectal cancer randomized into two therapeutic strategies: combination and sequential. The dataset contains measurements of tumor size (left-censored sums of the longest diameters of target lesions; transformed using Box-Cox) with baseline characteristics(treatment arm, age, WHO performance status and previous resection).

## Usage

```
data(colorectalLongi)
```

## Format

This data frame contains the following columns:

**id** identification of each subject. Repeated for each recurrence

**year** time of visit counted in years from baseline

**tumor.size** Individual longitudinal measurement of transformed (Box-Cox with parameter 0.3) sums of the longest diameters, left-censored due to a detection limit (threshold $s = -3.33$).

**treatment** To which treatment arm a patient was allocated? 1: sequential (S); 2: combination (C)

**age** Age at baseline: 1: <50 years, 2: 50-69 years, 3: >69 years

**who.PS** WHO performance status at baseline: 1: status 0, 2: status 1, 3: status 2

**prev.resection** Previous resection of the primate tumor? 0: No, 1: Yes

## Note

We thank the Federation Francophone de Cancerologie Digestive and Gustave Roussy for sharing the data of the FFCD 2000-05 trial supported by an unrestricted Grant from Sanofi.

## References

Ducreux, M., Malka, D., Mendiboure, J., Etienne, P.-L., Texereau, P., Auby, D., Rougier, P., Gasmi, M., Castaing, M., Abbas, M., Michel, P., Gargot, D., Azzedine, A., Lombard- Bohas, C., Geoffroy, P., Denis, B., Pignon, J.-P., Bedenne, L., and Bouche, O. (2011). Sequential versus combination chemotherapy for the treatment of advanced colorectal cancer (FFCD 2000-05): an open-label, randomised, phase 3 trial. *The Lancet Oncology* **12**, 1032-44.

`CPRSKbmtcrr` *Transformed Bone Marrow Transplant Data for Competing Risks*

## Description

A dataset derived from the `bmtcrr` data (originally from the `casebase` package). This version adds unique subject IDs, an example grouping variable, and calculates an 'observed_time' based on age at transplant plus follow-up time in years, setting age 0 as the origin. It's prepared for competing risks analyses, potentially with frailties or left truncation.

## Usage

```
data(CPRSKbmtcrr)
```

## Format

A data frame with 177 observations and the following columns:

**id** Unique subject identification number.

**Sex** Gender of the individual (Factor: Male, Female).

**D** Disease type: ALL or AML (Factor: ALL, AML).

**Phase** Phase at transplant (Factor: Relapse, CR1, CR2, CR3).

**Age** Age in years at transplant (start of follow-up).

**Status** Status indicator: 0 = censored, 1 = relapse, 2 = competing event.

**Source** Source of stem cells (Factor: BM+PB, PB).

**ftime** Original failure time in months since transplant.

**group** Example grouping variable (numeric, derived from id mod 10 + 1).

**observed_time** Time in years since birth (Age + ftime/12) representing the time of event or censoring relative to birth as origin.

## Details

This dataset was created by taking the original `bmtcrr` data from the `casebase` package and applying the following transformations:

1. Added a unique subject identifier `id`.
2. Added an example grouping variable `group` based on `id`.
3. Calculated `observed_time = Age + ftime/12` to represent the subject's age at event or censoring, potentially for use with left truncation at `Age`.

The primary event is typically relapse (Status=1), with death without relapse (Status=2) as a competing event. Censoring is Status=0. Note that the time scale for `observed_time` is years since birth.

### Source

Derived from the `bmtcrr` dataset available in the `casebase` package.

### References

Scrucca L, Santucci A, Aversa F. Competing risk analysis using R: an easy guide for clinicians. *Bone Marrow Transplant*. 2007 Aug;40(4):381-7. doi:10.1038/sj.bmt.1705727.

### See Also

`bmtcrr`. The `casebase` package.

---

## `dataAdditive` *Simulated data as a gathering of clinical trials databases*

---

### Description

This contains simulated samples of 100 clusters with 100 subjects in each cluster, like a gathering of clinical trials databases. Two correlated centred gaussian random effects are generated with the same variance fixed at 0.3 and the covariance at -0.2. The regression coefficient $\beta$ is fixed at -0.11. The percentage of right-censored data is around 30 percent which are generated from a uniform distribution on [1,150]. The baseline hazard function is considered as a simple Weibull.

### Usage

```
data(dataAdditive)
```

### Format

This data frame contains the following columns:

**group** identification variable

**t1** start of interval (=0, because left-truncated data are not allowed)

**t2** end of interval (death or censoring time)

**event** censoring status (0:alive, 1:death, as acensoring indicator

**var1** dichotomous covariate (=0 or 1,as a treatment variable)

**var2** dichotomous covariate (=0 or 1,as a treatment variable)

### Source

V. Rondeau, S. Michiels, B.Liquet, and J.P. Pignon (2008). Investigating trial and treatment heterogeneity in an individual patient data meta-analysis of survival data by mean of the maximum penalized likelihood approach. *Statistics in Medecine*, **27**, 1894-1910.

dataMultiv *Simulated data for two types of recurrent events and a terminal event*

## Description

This contains a simulated sample of of 800 subjects and 1652 observations. This dataset can be used to illustrate how to fit a joint multivariate frailty model. Two gaussian correlated random effects were generated with mean 0, variances 0.5 and a correlation coefficient equals to 0.5. The coefficients $\alpha_1$ and $\alpha_2$ were fixed to 1. The three baseline hazard functions followed a Weibull distribution and right censoring was fixed at 5.

## Usage

```
data(dataMultiv)
```

## Format

This data frame contains the following columns:

**PATIENT** identification of patient

**obs** number of observation for a patient

**TIME0** start of interval

**TIME1** end of interval (death or censoring time)

**INDICREC** recurrent of type 1 status (0:no, 1:yes)

**INDICMETA** recurrent of type 2 status (0:no, 1:yes)

**INDICDEATH** censoring status (0:alive, 1:death)

**v1** dichotomous covariate (0,1)

**v2** dichotomous covariate (0,1)

**v3** dichotomous covariate (0,1)

**TIMEGAP** time to event

dataNCC *Simulated data for recurrent events and a terminal event with weigths using nested case-control design*

## Description

This contains a simulated sample of of 819 subjects and 1510 observations. This dataset can be used to illustrate how to fit a joint frailty model for data from nested case-control studies.

## Usage

```
data(dataNCC)
```

### Format

This data frame contains the following columns:

**id** identification of patient

**cov1** dichotomous covariate (0,1)

**cov2** dichotomous covariate (0,1)

**t.start** start of interval

**t.stop** end of interval (death or censoring time)

**gaptime** time to event

**event** recurrent event status (0:no, 1:yes)

**deathdays** time of terminal event (death or right-censoring)

**death** censoring status (0:alive, 1:death)

**ncc.wts** weights for NCC design

---

## `dataNested` *Simulated data with two levels of grouping*

---

### Description

This contains a simulated sample of 400 observations which allow establishing 20 clusters with 4 subgroups and 5 subjects in each subgroup, in order to obtain two levels of grouping. This data set is useful to illustrate how to fit a nested model. Two independent gamma frailty parameters with a variance fixed at 0.1 for the cluster effect and at 0.5 for the subcluster effect were generated. Independent survival times were generated from a simple Weibull baseline risk function. The percentage of censoring data was around 30 per cent. The right-censoring variables were generated from a uniform distribution on [1,36] and a left-truncating variable was generated with a uniform distribution on [0,10]. Observations were included only if the survival time is greater than the truncated time.

### Usage

```
data(dataNested)
```

### Format

This data frame contains the following columns:

**group** group identification variable

**subgroup** subgroup identification variable

**t1** start of interval (0 or truncated time)

**t2** end of interval (death or censoring time)

**event** censoring status (0: alive, 1: death)

**cov1** dichotomous covariate (0,1)

**cov2** dichotomous covariate (0,1)

## Source

V. Rondeau, L. Filleul, P. Joly (2006). Nested frailty models using maximum penalized likelihood estimation. *Statistics in Medecine*, **25**, 4036-4052.

---

`dataOvarian` *Advanced Ovarian Cancer dataset*

---

## Description

This dataset combines the data that were collected in four double-blind randomized clinical trials in advanced ovarian cancer. In these trials, the objective was to examine the efficacy of cyclophosphamide plus cisplatin (CP) versus cyclophosphamide plus adriamycin plus cisplatin (CAP) to treat advanced ovarian cancer. The candidate surrogate endpoint **S** is progression-free survival time, defined as the time (in years) from randomization to clinical progression of the disease or death. The true endpoint **T** is survival time, defined as the time (in years) from randomization to death of any cause

## Usage

```
data(dataOvarian)
```

## Format

This data frame contains the following columns:

**patientID** The identification number of a patient

**trialID** The center in which a patient was treated

**trt** The treatment indicator, coded as 0 = cyclophosphamide plus cisplatin (CP) and 1 = cyclophosphamide plus adriamycin plus cisplatin(CAP)

**timeS** The candidate surrogate (progression-free survival)

**statusS** Censoring indicator for for Progression-free survival

**timeT** The true endpoint (survival time)

**statusT** Censoring indicator for survival time

## Source

Ovarian cancer Meta-Analysis Project (1991). Cyclophosphamide plus cisplatin plus adriamycin versus Cyclophosphamide, doxorubicin, and cisplatin chemotherapy of ovarian carcinoma: A meta-analysis. *Classic Papers and Current Comments*, **3**, 237-234.

`Diffepoce` *Difference of Expected Prognostic Observed Cross-Entropy (EPOCE) estimators and its 95% tracking interval between two joint models.*

## Description

This function computes the difference of two EPOCE estimates (CVPOL and MPOL) and its 95% tracking interval between two joint models estimated using `frailtyPenal`, `longiPenal` or `trivPenal`. Difference in CVPOL is computed when the EPOCE was previously estimated on the same dataset as used for estimation (using an approximated cross-validation), and difference in MPOL is computed when the EPOCE was previously estimated on an external dataset.

## Usage

```
Diffepoce(epoce1, epoce2)
```

## Arguments

| | |
|---|---|
| `epoce1` | a first object inheriting from class epoce. |
| `epoce2` | a second object inheriting from class epoce. |

## Details

From the EPOCE estimates and the individual contributions to the prognostic observed log-likelihood obtained with `epoce` function on the same dataset from two different estimated joint models, the difference of CVPOL (or MPOL) and its 95% tracking interval is computed. The 95% tracking interval is : Delta(MPOL) +/- qnorm(0.975)*sqrt(VARIANCE) for an external dataset Delta(CVPOL) +/- qnorm(0.975)*sqrt(VARIANCE) for the dataset used in `frailtyPenal`, `longiPenal` or `trivPenal` where Delta(CVPOL) (or Delta(MPOL)) is the difference of CVPOL (or MPOL) of the two joint models, and VARIANCE is the empirical variance of the difference of individuals contributions to the prognostic observed log-likelihoods of the two joint models.

The estimators of EPOCE from arguments epoce1 and epoce2 must have been computed on the same dataset and with the pred.times.

## Value

| | |
|---|---|
| `new.data` | a boolean which is FALSE if computation is done on the same data as for estimation, and TRUE otherwise |
| `pred.times` | time or vector of times used in the function |
| `DEPOCE` | the difference between the two MPOL or CVPOL for each time |
| `TIinf` | lower confidence band for the difference |
| `TIsup` | upper confidence band for the difference |

## References

D. Commenges, B. Liquet, C. Proust-Lima (2012). Choice of prognostic estimators in joint models by estimating differences of expected conditional Kullback-Leibler risks. *Biometrics*, **68(2)**, 380-387.

## Examples

```
## Not run:

#Example for joint frailty models
data(readmission)

# first joint frailty model
joint1 <- frailtyPenal(Surv(t.start,t.stop,event)~ cluster(id) +
  dukes + charlson + sex + chemo + terminal(death),
  formula.terminalEvent = ~ dukes + charlson + sex + chemo ,
  data = readmission, n.knots = 8, kappa = c(2.11e+08,9.53e+11),
  recurrentAG=TRUE)

# second joint frailty model without dukes nor charlson as covariates
joint2 <- frailtyPenal(Surv(t.start,t.stop,event)~ cluster(id) +
  sex + chemo + terminal(death),
  formula.terminalEvent = ~ sex + chemo ,
  data = readmission, n.knots = 8, kappa = c(2.11e+08,9.53e+11),
  recurrentAG=TRUE)

temps <- c(200,500,800,1100)

# computation of estimators of EPOCE for the two models
epoce1 <- epoce(joint1,temps)
epoce2 <- epoce(joint2,temps)

# computation of the difference
diff <- Diffepoce(epoce1,epoce2)

print(diff)
plot(diff)


#Example for joint models with a biomarker
data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

# first joint model for a biomarker and a terminal event
modLongi <- longiPenal(Surv(time0, time1, state) ~ age +
treatment + who.PS, tumor.size ~  year*treatment + age +
who.PS, colorectalSurv, data.Longi =colorectalLongi,
random = c("1", "year"),  id = "id", link = "Random-effects",
```

```
left.censoring = -3.33, hazard = "Weibull",
method.GH = "Pseudo-adaptive")

# second joint model for a biomarker, recurrent events and a terminal event
# (computation takes around 30 minutes)
modTriv <- model.weib.RE.gap <-trivPenal(Surv(gap.time, new.lesions)
~ cluster(id) + age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive", n.nodes=7)

time <- c(1, 1.5, 2, 2.5)

# computation of estimators of EPOCE for the two models
epoce1 <- epoce(modLongi, time)
# (computation takes around 10 minutes)
epoce2 <- epoce(modTriv, time)


# computation of the difference
diff <- Diffepoce(epoce1, epoce2)

print(diff)
plot(diff)

## End(Not run)
```

---

`epoce` *Estimators of the Expected Prognostic Observed Cross-Entropy (EPOCE) for evaluating predictive accuracy of joint models.*

---

## Description

This function computes estimators of the Expected Prognostic Observed Cross-Entropy (EPOCE) for evaluating the predictive accuracy of joint models using `frailtyPenal`, `longiPenal`, `trivPenal` or `trivPenalNL`. On the same data as used for estimation of the joint model, this function computes both the Mean Prognosis Observed Loss (MPOL) and the Cross-Validated Prognosis Observed Loss (CVPOL), two estimators of EPOCE. The latter corrects the MPOL estimate for over-optimism by approximated cross-validation. On external, this function only computes MPOL.

## Usage

```
epoce(fit, pred.times, newdata = NULL, newdata.Longi = NULL)
```

### Arguments

fit
: A jointPenal, longiPenal, trivPenal or trivPenalNL object.

pred.times
: Time or vector of times to compute epoce.

newdata
: Optional. In case of joint models obtained with `frailtyPenal`, `trivPenal` or `trivPenalNL`. For models inheriting from `trivPenal` or `trivPenalNL` class, if `newdata` is given, `newdata.Longi` must be given as well. When missing, the data used for estimating the fit are used, and CVPOL and MPOL are computed (internal validation). When `newdata` is specified, only MPOL is computed on this new dataset (external validation). The new dataset and the dataset used in the estimation must have the same covariates with the same coding without missing data.

newdata.Longi
: Optional. In case of joint models obtained with `longiPenal`, `trivPenal` or `trivPenalNL`. For models inheriting from `longiPenal`, if the `newdata.Longi` is given, `newdata` must be `NULL`, but for models from `trivPenal` or `trivPenalNL` class, if `newdata.Longi` is given, `newdata` must be provided as well. The two datasets newdata and newdata.Longi must include the information concerning the same patients with the same characteristics and the appropriate data on follow up (recurrences for `newdata` and longitudinal measurements for `newdata.Longi`).

### Value

data
: name of the data used to compute epoce

new.data
: a boolean which is FALSE if computation is done on the same data as for estimation, and TRUE otherwise

pred.times
: time or vector of times used in the function

mpol
: values of MPOL for each pred.times

cvpol
: values of CVPOL for each pred.times

IndivContrib
: all the contributions to the log-likelihood for each pred.times

AtRisk
: number of subject still at risk for each pred.times

### References

D. Commenges, B. Liquet, C. Proust-Lima (2012). Choice of prognostic estimators in joint models by estimating differences of expected conditional Kullback-Leibler risks. *Biometrics*, **68(2)**, 380-387.

### Examples

```
## Not run:

########################################
#### EPOCE on a joint frailty model ####
########################################

data(readmission)
```

```
modJoint.gap <- frailtyPenal(Surv(t.start,t.stop,event)~ cluster(id) +
  dukes + charlson + sex + chemo + terminal(death),
  formula.terminalEvent = ~ dukes + charlson + sex + chemo ,
  data = readmission, n.knots = 8, kappa =c(2.11e+08,9.53e+11),
  recurrentAG=TRUE)

# computation on the same dataset
temps <- c(200,500,800,1100)
epoce <- epoce(modJoint.gap,temps)

print(epoce)
plot(epoce,type = "cvpol")

# computation on a new dataset
# here a sample of readmission with the first 50 subjects
s <- readmission[1:100,]
epoce <- epoce(modJoint.gap,temps,newdata=s)

print(epoce)
plot(epoce,type = "mpol")

#################################################
#### EPOCE on a joint  model for a biomarker ####
#########   and a terminal event  ###############
#################################################

data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

modLongi <- longiPenal(Surv(time0, time1, state) ~ age +
treatment + who.PS, tumor.size ~  year*treatment + age +
who.PS, colorectalSurv, data.Longi =colorectalLongi,
random = c("1", "year"),  id = "id", link = "Random-effects",
left.censoring = -3.33, hazard = "Weibull",
method.GH = "Pseudo-adaptive")

# computation on the same dataset
time <- c(1, 1.5, 2, 2.5)
epoce <- epoce(modLongi,time)

print(epoce)
plot(epoce, type = "cvpol")

# computation on a new dataset
# here a sample of colorectal data with the first 50 subjects
s <-  subset(colorectal, new.lesions == 0 & id%in%1:50)
s.Longi <- subset(colorectalLongi, id%in%1:50)
epoce <- epoce(modLongi, time, newdata = s, newdata.Longi = s.Longi)

print(epoce)
```

```
plot(epoce, type = "mpol")


####################################################
#### EPOCE on a joint model for a biomarker, ######
#### recurrent events and a terminal event   ######
####################################################

data(colorectal)
data(colorectalLongi)

# Linear model for the biomarker
# (computation takes around 30 minutes)
model.trivPenalNL <-trivPenal(Surv(gap.time, new.lesions) ~ cluster(id)
+ age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive", n.nodes=7)

# computation on the same dataset
time <- c(1, 1.5, 2, 2.5)

# (computation takes around 10 minutes)
epoce <- epoce(model.trivPenalNL,time)
print(epoce)
plot(epoce, type = "cvpol")

# computation on a new dataset
# here a sample of colorectal data with the first 100 subjects
s <-  subset(colorectal,  id%in%1:100)
s.Longi <- subset(colorectalLongi, id%in%1:100)
# (computation takes around 10 minutes)
epoce <- epoce(model.trivPenalNL, time, newdata = s, newdata.Longi = s.Longi)

print(epoce)
plot(epoce, type = "mpol")



# Non-linear model for the biomarker

# No information on dose - creation of a dummy variable
colorectalLongi$dose <- 1

# (computation can take around 40 minutes)
model.trivPenalNL <- trivPenalNL(Surv(time0, time1, new.lesions) ~ cluster(id) + age + treatment
 + terminal(state), formula.terminalEvent =~ age + treatment, biomarker = "tumor.size",
 formula.KG ~ 1, formula.KD ~ treatment, dose = "dose", time.biomarker = "year",
 data = colorectal, data.Longi =colorectalLongi, random = c("y0", "KG"), id = "id",
 init.B = c(-0.22, -0.16, -0.35, -0.19, 0.04, -0.41, 0.23), init.Alpha = 1.86,
 init.Eta = c(0.5, 0.57, 0.5, 2.34), init.Biomarker = c(1.24, 0.81, 1.07, -1.53),
```

```
 recurrentAG = TRUE, n.knots = 5, kappa = c(0.01, 2), method.GH = "Pseudo-adaptive")

# computation on the same dataset
time <- c(1, 1.5, 2, 2.5)

epoce <- epoce(model.trivPenalNL, time)



## End(Not run)
```

---

`event2` *Identify event2 indicator*

---

## Description

This is a special function used in the context of multivariate frailty model with two types of recurrent events and a terminal event (e.g., censoring variable related to both recurrent events). It contains the indicator of the recurrent event of type 2, normally 0=no event, 1=event, and is used on the right hand side of a formula of a 'multivPenal' object. Using `event2()` in a formula implies that a multivariate frailty model for two types of recurrent events and a terminal event is fitted.

## Usage

```
event2(x)
```

## Arguments

| | |
|---|---|
| `x` | A numeric variable but should be a boolean which equals 1 if the subject has experienced an event of type 2 and 0 if not. |

## Value

| | |
|---|---|
| `x` | an indicator for an event of type 2 |

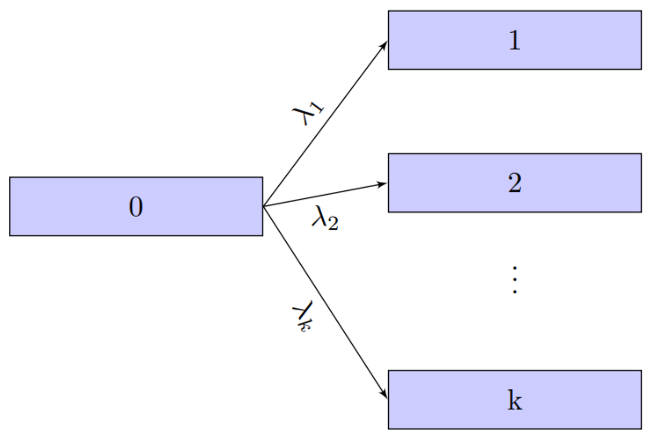
## See Also

multivPenal

frailtyCmprsk *Fit a Weibull Competing Risks Model with Optional Shared Frailty*

## Description

Fit a Weibull competing risks model with shared gamma frailty between all transitions (0->1,0->2,...,0->k). Handles left-truncated and right-censored data. The model considers transitions from an initial state (0) to (k) competing absorbing states.




## Usage

```
frailtyCmprsk(formulas, data, maxit = 300,
init.B, init.Theta, init.hazard.weib,
LIMparam = 1e-3, LIMlogl = 1e-3, LIMderiv = 1e-3,
x0, print.info = FALSE, print.result = TRUE,
partialH, blinding = TRUE)
```

## Arguments

`formulas` A list of formula objects. The first formula must include a response on the left-hand side of a ~ operator. The response must be a survival object as returned by the `Surv` function (e.g., `survival::Surv`). The argument `type = "mstate"` must be specified within the `Surv` function. The status indicator in the `Surv` object should be: 0 for right-censoring, 1 for the first competing event, 2 for the second, ..., and `k` for the `k`th competing event. Covariates for the transition from 0 to 1 are specified on the right-hand side (RHS) of the first formula. The remaining elements of the list should be one-sided formulas (e.g., `~ var1 + var2`), used only to specify the covariates. The second formula corresponds to the transition 0 -> 2, the third to 0 -> 3, and so on, up to the `k`th formula for transition 0 -> `k`. Left-truncation is supported and should be specified using the three-argument `Surv(time1, time2, status)` notation. Shared frailty can be specified via `cluster(group_variable)` on the RHS of the first formula only. It should not be included in the other formulas.

| | |
|---|---|
| `data` | A `data.frame` containing the variables named in `formulas`. |
| `maxit` | Maximum number of iterations for the Marquardt optimization algorithm. Default is 300. |
| `init.B` | Optional. A vector of initial values for the regression coefficients. It should contain the coefficients for each transition from 0->1, ..., up to 0->k (i.e., $\beta_1$, $\beta_2$, ..., $\beta_k$), concatenated in order. The total length must match the total number of covariates specified across all formulas. If omitted, the regression coefficients are initialized from default values. These defaults are obtained by fitting `k` independent Weibull proportional hazards models (without frailty), one for each transition. |
| `init.Theta` | Optional. Initial value for the frailty variance $\theta$. Default is 0.1. This parameter is only used if `cluster()` is present in `formulas`. |
| `init.hazard.weib` | Optional. A vector of initial values for the Weibull baseline hazard parameters. It must be of length `2 * k`, with the values ordered as: scale(0->1), shape(0->1), scale(0->2), shape(0->2), ..., scale(0->k), shape(0->k). If omitted, the baseline hazard parameters are initialized from default values. These defaults are obtained by fitting `k` independent Weibull proportional hazards models (without frailty), one for each transition. |
| `LIMparam` | Convergence threshold for the parameters based on the maximum absolute difference between successive iterations ($10^{-3}$ by default). |
| `LIMlogl` | Convergence threshold for the log-likelihood based on the absolute difference between successive iterations ($10^{-3}$ by default). |
| `LIMderiv` | Convergence threshold based on the relative distance to the optimum (related to gradient and Hessian) ($10^{-3}$ by default). See Details. |
| `x0` | Optional. A list of numeric vectors, where each vector specifies the time points at which to compute the baseline hazard and survival functions for a given transition. The order must follow the transitions: 0->1, 0->2, ..., 0->k. If not provided, defaults to a list where each element is a sequence of 99 time points from 0 to the maximum observed time for the corresponding transition. |
| `print.info` | Logical. If `TRUE`, prints information at each iteration of the optimization algorithm. Default is `FALSE`. |
| `print.result` | Logical. If `TRUE`, prints a formatted summary of the results. Default is `TRUE`. |
| `partialH` | Optional. Integer vector specifying the indices of parameters to exclude from the Hessian matrix when calculating the relative distance convergence criterion (`LIMderiv`). This is only considered if the first two criteria (`LIMparam`, `LIMlogl`) are met and the full Hessian is problematic (e.g., not invertible). Default is `NULL`. |
| `blinding` | Logical. If `TRUE`, the algorithm attempts to continue even if the log-likelihood calculation produces non-finite values (e.g., `Inf`, `NaN`) at some iteration. Setting to `FALSE` may cause the algorithm to stop earlier in such cases. Default is `TRUE`. |

## Details

Let $T$ be the time to event and $L \in \{1, 2, ..., k\}$ the indicator of the cause of the event.

The cause-specific hazard rate for cause $l \in \{1, 2, ..., k\}$ is:

$$\lambda_l(t) = \lim_{\Delta t \to 0^+} \frac{\mathbb{P}(t \leq T \leq t + \Delta t, L = l \mid T \geq t)}{\Delta t}$$

A proportional hazards model with a shared frailty term $\omega_i$ is assumed for each transition within group $i$. For the $j^{th}$ subject ($j = 1, ..., n_i$) in the $i^{th}$ group ($i = 1, ..., G$), the $l^{th}$ ($l = 1, ..., k$) transition intensity is defined as follows:

$$\lambda_l^{ij}(t|\omega_i, X_l^{ij}) = \lambda_{0l}(t)\omega_i \exp(\beta_l^T X_l^{ij})$$

where $\omega_i \sim \Gamma(\frac{1}{\theta}, \frac{1}{\theta})$ with $\boldsymbol{E}(\omega_i) = 1$ and $\boldsymbol{Var}(\omega_i) = \theta$.

$\omega_i$ is the frailty term for the $i^{th}$ group. For subject-specific frailties, use `cluster(id)` where id is unique ($n_i = 1$). $\beta_l$ ($l = 1, ..., k$) is the vector of time fixed regression coefficients for the transition 0->l. $X_l^{ij}$ ($l = 1, ..., k$) is the vector of time fixed covariates for the $j^{th}$ subject in the $i^{th}$ group for the transition 0->l. $\lambda_{0l}(.)$ ($l = 1, ..., k$) is the baseline hazard function for the transition 0->l.

The Weibull baseline hazard parameterization is:

$$\lambda(t) = \frac{\gamma}{\lambda^\gamma} \cdot t^{\gamma - 1}$$

where $\lambda$ is the scale parameter and $\gamma$ the shape parameter

## Value

An object of class 'frailtyCmprsk' containing:

**b** Vector of the estimated parameters. Order is: (scale(0->1), shape(0->1), scale(0->2), shape(0->2), ..., scale(0->k), shape(0->k),$\hat{\theta}$ (if frailty), $\hat{\beta}_1$, $\hat{\beta}_2$,...,$\hat{\beta}_k$).

**call** The matched function call.

**coef** Vector of estimated regression coefficients.

**loglik** The marginal log-likelihood value at the final parameter estimates.

**grad** Gradient vector of the log-likelihood at the final parameter estimates.

**n** The number of observations used in the fit.

**n.events** Vector containing the number of observed events: count for 0->1, count for 0->2,..., count for 0->k, count for censoring.

**n.iter** Number of iterations.

**vcov** Variance-covariance matrix for the parameters listed in `b`.

**npar** Total number of estimated parameters.

**nvar** Total number of regression coefficients.

**shape.weib** Vector of estimated Weibull baseline shape parameters (shape(0->1),shape(0->2),...,shape(0->k)).

**scale.weib** Vector of estimated Weibull baseline scale parameters (scale(0->1),scale(0->2),...,scale(0->k)).

**crit** Convergence status code: 1=converged, 2=maximum iterations reached, 3=converged using partial Hessian, 4=the algorithm encountered a problem in the loglikelihood computation.

**Frailty** Logical. `TRUE` if a model with shared frailty (`cluster(.)`) was fitted.

**beta_p.value** Vector of p-values from Wald tests for the regression coefficients in `coef`.

**AIC** Akaike Information Criterion, calculated as $AIC = \frac{1}{n}(np - l(.))$, where np is the number of parameters and l is the log-likelihood.

**x0** List of numeric vectors, where each vector contains the time points used for calculating the baseline functions for each transition.

**lam0** List of matrices, where each matrix contains baseline hazard estimates and 95% confidence intervals at the corresponding time points in `x0`, for each transition.

**surv0** List of matrices, where each matrix contains baseline survival estimates and 95% confidence intervals at the corresponding time points in `x0`, for each transition.

**medians** List of matrices, where each matrix contains the estimated median baseline survival time and its 95% confidence interval for each transition.

**linear.pred** List of numeric vectors, where each vector contains to the linear predictors for transition 0->l ($l = 1, ..., k$). For non-frailty models, each element is of the form $\hat{\beta}_l^t X_l$. For frailty models, it includes the estimated log-frailty: $\hat{\beta}_l^t X_l + \log(\hat{\omega}_i)$.

**names.factor** List of character vectors, where each vector contains the factor covariates included in the model for the corresponding transition.

**global_chisq** List of numeric vectors, where each vector contains the chi-squared statistics from global Wald tests for factor variables for the corresponding transition.

**dof_chisq** List of integer vectors, where each vector contains the degrees of freedom for the global Wald tests for the corresponding transition.

**p.global_chisq** List of numeric vectors, where each vector contains the p-values for the global Wald tests for the corresponding transition.

**global_chisq.test** Binary vector of length k indicating whether any global factor tests were performed for each transition.

If `Frailty` is `TRUE`, the following components related to frailty are also included:

**groups** The number of unique groups specified by `cluster(.)`.

**theta** The estimated variance ($\hat{\theta}$) of the Gamma frailty distribution.

**theta_p.value** The p-value from a Wald test for the null hypothesis $H_0 : \theta = 0$.

**VarTheta** The estimated variance of the frailty variance estimator: $\hat{Var}(\hat{\theta})$.

**frailty.pred** Vector containing the empirical Bayes predictions of the frailty term for each group.

**frailty.var** Vector containing the variances of the empirical Bayes frailty predictions.

**frailty.sd** Vector containing the standard errors of the empirical Bayes frailty predictions.

## Note

The optimization uses the robust Marquardt algorithm (Marquardt, 1963), combining Newton-Raphson and steepest descent steps. Iterations stop when criteria `LIMparam`, `LIMlogl`, and `LIMderiv` are all met. Confidence bands for the baseline hazard and baseline survival functions were computed using a Monte Carlo simulation approach based on the estimated Weibull parameters. A sample of size 1000 was drawn from the joint distribution of the shape and scale parameters. For each sampled parameter set, the baseline functions were evaluated over the time grid defined by the vector `x0`. Pointwise 95 confidence bands were then obtained by computing the 2.5th and 97.5th percentiles of the simulated values at each time point for each baseline function.

### References

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2), 431-441.

Liquet, B., Timsit, J. F., & Rondeau, V. (2012). Investigating hospital heterogeneity with a multi-state frailty model: application to nosocomial pneumonia disease in intensive care units. *BMC Medical Research Methodology*, 12(1), 1-14.

### Examples

```
data(CPRSKbmtcrr)

##-- Weibull competing risks model with
##   group frailty shared between transitions --##

ModCmprsk_Group <- frailtyCmprsk(
  formulas = list(
    Surv(observed_time, Status, type = "mstate") ~ cluster(group) + Sex,
    ~ Sex
  ),
  data        = CPRSKbmtcrr,
  print.info  = FALSE,
  maxit       = 100
)

##-- Weibull competing risks model with subject-specific
##   frailty shared between transitions --##

ModCmprsk_Subject <- frailtyCmprsk(
  formulas = list(
    Surv(observed_time, Status, type = "mstate") ~ cluster(id) + Sex,
    ~ Sex
  ),
  data        = CPRSKbmtcrr,
  print.info  = FALSE,
  maxit       = 100
)

##--- Simple Weibull competing risks model with left truncation ---##

ModCmprsk_LeftTrunc <- frailtyCmprsk(
  formulas = list(
    Surv(Age, observed_time, Status, type = "mstate") ~ Source,
    ~ Sex
  ),
  data       = CPRSKbmtcrr,
  print.info = FALSE
)

##--- Simple Weibull competing risks model with a factor and
```

```
##    no covariates for the first competing event (left truncation) ---##

ModCmprsk_Factor_LeftTrunc <- frailtyCmprsk(
  formulas = list(
    Surv(Age, observed_time, Status, type = "mstate") ~ Source,
    ~ factor(Phase)
  ),
  data       = CPRSKbmtcrr,
  print.info = FALSE
)
```

---

frailtyDesign *Sample Size calculation and Power Analysis using Gamma-Frailty Models*

---

## Description

A collection of functions to calculate statistical power and required sample sizes for survival analysis using frailty models, specifically the Shared Frailty Model (SFM), Nested Frailty Model (NFM), Joint Frailty Model (JFM), and General Joint Frailty Model (GJFM).

For each frailty model type (denoted by `*`, where `*` corresponds to SFM, NFM, JFM or GJFM), the package provides two distinct functions:

- `*.power`: Computes the statistical power under given study settings.
- `*.ssize`: Determines the required sample size needed to achieve a specified target power under given study settings.

## Usage

```
#######################################################
## 1. SHARED FRAILTY MODEL (SFM)
#######################################################

# Compute power for a given sample size in a SFM
# ---------------------------------------------------
SFM.power(
  Groups = 80, ni = 8, ni.type = "max", Acc.Dur = 0,FUP = 12,
  FUP.type = "UpToEnd", median.H0 = 1, beta.H0 = 0, beta.HA = log(0.75),
  shape.W = 1, theta = 0.25, ratio = 1, samples.mc = 1e4, seed = 42,
  timescale = "gap", data.type = "grouped",
  cens.par = 5, cens.type = "Expo", statistic = "Wald",
  typeIerror = 0.05, test.type = "2-sided"
)

# Compute sample size for a given power in a SFM
# ---------------------------------------------------
SFM.ssize(
```

```
  power = 0.8, ni = 8, ni.type = "max", Acc.Dur = 0,FUP = 12,
  FUP.type = "UpToEnd", median.H0 = 1, beta.H0 = 0, beta.HA = log(0.75),
  shape.W = 1, theta = 0.25, ratio = 1, samples.mc = 1e4, seed = 42,
  timescale = "gap", data.type = "grouped",
  cens.par = 5, cens.type = "Expo", statistic = "Wald",
  typeIerror = 0.05, test.type = "2-sided"
)

######################################################
## 2. NESTED FRAILTY MODEL (NFM)
######################################################

# Compute power for a given sample size in a NFM
# ---------------------------------------------------
NFM.power(
  Groups = 80, ni = 8, ni.type = "max", kij = 15, kij.type = "max",
  Acc.Dur = 0, FUP = 12, FUP.type = "UpToEnd", median.H0 = 1,
  beta.H0 = 0, beta.HA = log(0.75), shape.W = 1, theta = 0.25, eta = 0.5,
  ratio = 1, samples.mc = 1e4, seed = 42,
  timescale = "gap", data.type = "grouped", cens.par = 5, cens.type = "Expo",
  statistic = "Wald", typeIerror = 0.05, test.type = "2-sided"
)

# Compute sample size for a given power in a NFM
# ---------------------------------------------------
NFM.ssize(
  power = 0.8, ni = 8, ni.type = "max", kij = 15, kij.type = "max",
  Acc.Dur = 0, FUP = 12, FUP.type = "UpToEnd", median.H0 = 1,
  beta.H0 = 0, beta.HA = log(0.75), shape.W = 1, theta = 0.25, eta = 0.5,
  ratio = 1, samples.mc = 1e4, seed = 42,
  timescale = "gap", data.type = "grouped", cens.par = 5, cens.type = "Expo",
  statistic = "Wald", typeIerror = 0.05, test.type = "2-sided"
)

######################################################
## 3. JOINT FRAILTY MODEL (JFM)
######################################################

# Compute power for a given sample size in a JFM
# ---------------------------------------------------
JFM.power(
  Npts = 400, ni = 8, ni.type = "max", Acc.Dur = 0, FUP = 12,
 FUP.type = "UpToEnd", medianR.H0 = 3, medianD.H0 = 10, betaTest.type = "joint",
  betaR.H0 = 0, betaR.HA = log(0.75), betaD.H0 = 0, betaD.HA = log(0.85),
  shapeR.W = 1, shapeD.W = 1, theta = 0.25, alpha = 1, ratio = 1,
  samples.mc = 1e4, seed = 42, timescale = "gap",
  statistic = "Wald", typeIerror = 0.05, test.type = "2-sided"
)
```

```
# Compute sample size for a given power in a JFM
# ---------------------------------------------------
JFM.ssize(
  power = 0.8, ni = 8, ni.type = "max", Acc.Dur = 0, FUP = 12,
  FUP.type = "UpToEnd", medianR.H0 = 3, medianD.H0 = 10, betaTest.type = "joint",
  betaR.H0 = 0, betaR.HA = log(0.75), betaD.H0 = 0, betaD.HA = log(0.85),
  shapeR.W = 1, shapeD.W = 1, theta = 0.25, alpha = 1, ratio = 1,
  samples.mc = 1e4, seed = 42, timescale = "gap",
  statistic = "Wald", typeIerror = 0.05, test.type = "2-sided"
)

#####################################################
## 4. GENERAL JOINT FRAILTY MODEL (GJFM)
#####################################################

# Compute power for a given sample size in a GJFM
# ---------------------------------------------------
GJFM.power(
  Npts = 400, ni = 8, ni.type = "max", Acc.Dur = 0, FUP = 12,
  FUP.type = "UpToEnd", medianR.H0 = 3, medianD.H0 = 10,
  betaTest.type = "joint", betaR.H0 = 0, betaR.HA = log(0.75),
  betaD.H0 = 0, betaD.HA = log(0.85), shapeR.W = 1, shapeD.W = 1,
  theta = 0.25, eta = 0.5, ratio = 1, samples.mc = 1e4,
  seed = 42, timescale = "gap",
  statistic = "Wald", typeIerror = 0.05, test.type = "2-sided"
)

# Compute sample size for a given power in a GJFM
# ---------------------------------------------------
GJFM.ssize(
  power = 0.8, ni = 8, ni.type = "max", Acc.Dur = 0, FUP = 12,
  FUP.type = "UpToEnd", medianR.H0 = 3, medianD.H0 = 10,
  betaTest.type = "joint", betaR.H0 = 0, betaR.HA = log(0.75),
  betaD.H0 = 0, betaD.HA = log(0.85), shapeR.W = 1, shapeD.W = 1,
  theta = 0.25, eta = 0.5, ratio = 1, samples.mc = 1e4,
  seed = 42, timescale = "gap",
  statistic = "Wald", typeIerror = 0.05, test.type = "2-sided"
)
```

### Arguments

`Groups` Only in SFM and NFM: A numeric value, where interpretation depends on the `data.type` parameter and on the model:

- For SFM, it corresponds to either the number of groups (grouped data) or the number of subjects (recurrent events data).
- For NFM, it corresponds to either the number of groups (grouped and recurrent event data) or the number of subjects (multi-type recurrent events

data).

Default is 80.

ni
: A numeric value or an array (dim = 2), representing expected values or distribution parameters. Interpretation depends on the `data.type` parameter and on the model:

  - For SFM, it corresponds to either the expected number of subjects per group (grouped data) or the expected number of recurrent events per subject (recurrent events data).
  - For NFM, it corresponds to the expected number of subgroups within each group (grouped data), the expected number of recurrent events per group (recurrent event data), or the number of distinct recurrent event type (multi-type recurrent event data).
  - For JFM/GJFM, it corresponds to the expected number of recurrent events per subject.

  The default value is 8.

ni.type
: Character value, specifying `ni`. Valid options:

  - `"max"`: `ni` is a fixed number.
  - `"pois"`: `ni` is a mean (parameter of a Poisson distribution).
  - `"unif"`: `ni` is the lower and upper bound parameters of a uniform distribution.

  Options `"pois"` and `"unif"` can only be selected for recurrent event data; see Note for details. Default is `"max"`.

Acc.Dur
: Non-negative numeric value. Parameter for a uniform accrual from time 0 to time `Acc.Dur`. Default is 0.

FUP
: A positive numeric value of follow-up duration as defined in the study protocol (i.e. administrative censoring). Default is 12.

FUP.type
: Character value, indicating the type of follow-up. Valid options:

  - `"Fixed"`: each subject is followed exactly for `FUP` time units after enrollment.
  - `"UptoEnd"`: global study cutoff at time `FUP`; individual follow-up for at most `FUP`.

  Default is `"Fixed"`.

median.H0
: Only in SFM and NFM: A positive numeric value, used for the scale parameter (Weibull) calculation. If recurrent event data, it is the median gap time to event under the null (excluding censoring times). If grouped data, it is the median time to an event under the null (excluding censoring times). Default is 1.

beta.H0
: Only in SFM and NFM: log-hazard ratio parameter under the null hypothesis (H0). Default is 0.

beta.HA
: Only in SFM and NFM: log-hazard ratio parameter under the alternative hypothesis (HA). Default is log(0.75).

shape.W
: Only in SFM and NFM: A positive numeric value, corresponding to the shape parameter (Weibull) of the baseline hazard of the recurrent event. Default is 1.

`theta` A positive numeric value, corresponding to the Gamma-frailty variance for the main random effect. Default is 0.25.

`ratio` A positive numeric value, corresponding to the allocation ratio (*experimental : control*). Default is 1.

`samples.mc` A positive numeric value, corresponding to the number of Monte Carlo samples used to approximate the Fisher information matrix. Default is 1e4.

`seed` Integer number for random-number generation seed. Ensures reproducibility of the Monte-Carlo simulations. Default is 42.

`timescale` Character value indicating the timescale when recurrent event data type is considered. Can be either 'gap' or 'calendar'. See note for more detail. Default is 'gap'.

`data.type` Only in SFM and NFM: Character value indicating what kind of data we want to consider for the current frailty model. Valid options differ depending on the model:

- For SFM, can be either `"grouped"` (corresponding to subjects included in a group) or `"rec_event"` (corresponding to subjects experiencing recurrent events).
- For NFM, the hierarchical structure of the data can be either `"grouped"` (where subjects are included into subgroup and subgroups into groups), `"rec_event1"` (where the group level corresponds to a group (e.g., hospitals) and subgroup level to a subject) or `"rec_event2"` (where the group level corresponds to a subject and subgroup level to a type of recurrent event).

Default is "grouped".

`cens.par` Only in SFM and NFM: A numeric value corresponding to the parameter of the distribution for non-administrative censoring. Default is 10000.

`cens.type` Only in SFM and NFM: Character value, specifying the distribution for non-administrative censoring. Valid options:

- `"Expo"`: in this case, `cens.par` is the median from an exponential distribution.
- `"Unif"`: in this case, `cens.par` is the lower and upper bound parameters of a uniform distribution.

Default is "Expo".

`statistic` Type of test statistic used. Currently, only `"Wald"` is available.

`typeIerror` A numeric value corresponding to the type I error level. Default is 0.05.

`test.type` Character value indicating whether It is a one-tailed or two-tailed test. Valid options are either `"1-sided"` or `"2-sided"`. Default is "2-sided".

`power` Numeric in (0,0.99]. The target power $1 - \beta$. Default is 0.8.

`kij` Only in NFM: A numeric value or an array (dim = 2), representing expected values or distribution parameters. Interpretation depends on the `data.type` parameter:

- For grouped data: It is the number of observations per subgroup.
- For recurrent events data: It is the number of observation per subjects.

- For multi-type recurrent events data: It is the number of recurrences for each distinct type of event.

Default is 15.

`kij.type` Character value, specifying `kij`. Valid options:

- `"max"`: `kij` is a fixed number.
- `"pois"`: `kij` is a mean (parameter of a Poisson distribution).
- `"unif"`: `kij` is the lower and upper bound parameters of a uniform distribution.

Options `"pois"` and `"unif"` can only be selected for recurrent event data; see Note for details. Default is `"max"`.

`eta` Only in NFM and GJFM: positive numeric value, corresponding to an additional Gamma-frailty variance parameter for second-level nesting (NFM) or inter-recurrence dependence (GJFM). Default is 0.5.

`Npts` Only in JFM and GJFM: positive numeric value, corresponding to the total number of subjects. Default is 400.

`medianR.H0` Only in JFM and GJFM: positive numeric value, corresponding to the expected median time between two recurrent events under the null (H0), for the scale parameter (Weibull) calculation. Default is 3.

`medianD.H0` Only in JFM and GJFM: positive numeric value, corresponding to the expected median time to the terminal event under the null (H0), for the scale parameter (Weibull) calculation. Default is 10.

`betaTest.type` Only in JFM and GJFM: character value indicating which hypothesis is tested when computing power. Our implementation allows either power calculation or sample-size estimation, testing recurrent events alone, terminal event alone or both. Valid options: `"joint"` (for testing both $\beta_R$ and $\beta_D$), `"betaRtest"` (for testing only $\beta_R$) or `"betaDtest"` (for testing only $\beta_D$). Default is `"joint"`.

`betaR.H0` Only in JFM and GJFM: numeric value, corresponding to the log-hazard ratios for recurrent events under the null hypothesis (H0). Default is 0.

`betaR.HA` Only in JFM and GJFM: numeric value, corresponding to the log-hazard ratios for recurrent events under the alternative hypothesis (HA). Default is log(0.75).

`betaD.H0` Only in JFM and GJFM: numeric value, corresponding to the log-hazard ratios for terminal events under the null hypothesis (H0). Default is 0.

`betaD.HA` Only in JFM and GJFM: numeric value, corresponding to the log-hazard ratios for terminal events under the alternative hypothesis (HA). Default is log(0.85).

`shapeR.W` Only in JFM and GJFM: positive numeric value, corresponding to the shape parameter (Weibull) of the recurrent-event baseline hazard function. Default is 1.

`shapeD.W` Only in JFM and GJFM: positive numeric value, corresponding to the shape parameter (Weibull) of the terminal-event baseline hazard function. Default is 1.

`alpha` Only in JFM: numeric value, corresponding to the parameter $\alpha$ that modulates the association between recurrent and terminal events. Default is 1.

**Details**

See Dinart et al. (2024) for the original article. We present here the case where we want to assess the treatment effect only. Our null hypothesis is that there is no treatment effect (i.e. zero log-hazard ratio).

This approach relies on the squared Wald test to assess the presence of a treatment effect using an estimator $\hat{\Theta}$. Specifically, under our null hypothesis, the test statistic $X_w = Z^2 = (\hat{\Theta} - \Theta)^2/\mathcal{I}^{-1}(\hat{\Theta})$ follows a central $\chi_1^2$ distribution (i.e., non-centrality parameter $\mu = 0$), whereas under the alternative hypothesis, it follows a non-central $\chi_1^2(\mu)$ distribution with $\mu > 0$.

The parameter $\mu$ is estimated algorithmically, and the Fisher information $\mathcal{I}_1(\hat{\Theta})$ is obtained by simulation, leveraging the law of large numbers. Concretely, for an $M$-sample generated by simulation, the matrix $\mathcal{I}(\hat{\Theta})$ is approximated via the empirical mean of the products $\partial_{\Theta_k} l(\Theta(i)) \times \partial_{\Theta_l} l(\Theta(i))^\top$, $i \in [\![1, \ldots, M]\!]$. The algorithmic estimation for $\mu$ follows the three-step procedure described by Dinart et al. (2024):

1. Compute the $\alpha$-quantile (denoted $q_{1,\alpha}$) of a central chi-square distribution with 1 degree of freedom ($\chi_1^2$), given a specified type I error rate $\alpha$.
2. Determine a non-centrality parameter $\vartheta$ such that $1 - P\big(\chi_1^2(\vartheta) < q_{1,\alpha}\big) > 1 - \beta$, where $1 - \beta$ represents the desired statistical power and $1 - P\big(\chi_1^2(\vartheta) < q_{1,\alpha}\big)$ the computed power.
3. Optimize $\mu$ to find the smallest value satisfying that condition for all $x \in [0, \vartheta]$, i.e.,

$$\mu = \min_{x \in [0,\vartheta]} \Big\{ 1 - P\big(\chi_1^2(x) < q_{1,\alpha}\big) - (1 - \beta) \Big\}.$$

Once $\mu$ is estimated, the sample size $n$ is derived from

$$n \geq \mu \times \Big( \Theta_A^2 \times \mathcal{I}_1(\hat{\Theta}) \Big)^{-1},$$

where $\Theta_A$ denotes the parameter value under $H_A$. If we are interested in the evaluation of the power, we estimate the non-centrality parameter under $H_A$ for a given sample size $N$, then compute the power as $P\big(\chi_1^2(\vartheta) > q_{1,\alpha} \big| H_A)$.

For both the joint frailty model and general joint frailty model, by following the same methodology as in the univariate case, we can derive an expression for the sample size from the generalized Wald statistic. Let $H_0 : (\beta_R = 0)$ and $(\beta_D = 0)$ vs. $H_A : (\beta_R = \beta_R^A)$ or $(\beta_D = \beta_D^A)$, be our null and alternative hypotheses respectively. This multivariate test then follows a $\chi_Q^2$ distribution, where $Q$ is the rank of the matrix $C$, corresponding to the number of constraints applied on the parameters under the null hypothesis. The test statistic is:

$$X_W \ = \ n \, (C\,\Omega)^\top \Big( C\,\mathcal{I}_1^{-1}(\Omega)\, C^\top \Big)^{-1} (C\,\Omega) \ \sim \ \chi_q^2(\mu),$$

where $\Omega^\top$ is the vector parameter from the corresponding model. From this, we derive a sample size formula:

$$n \geq \mu \Big( (C\,\Omega)^\top \big( C\,\mathcal{I}_1^{-1}(\Omega)\, C^\top \big)^{-1} (C\,\Omega) \Big)^{-1}.$$

For instance, for the JFM, we have $\Omega^\top = (\beta_R, \beta_D, r_0(.), h_0(.), \theta, \alpha)^\top)$. If we want to test a treatment effect on both the recurrent and the terminal event (i.e. $H_0 : \beta_R = 0$ and $\beta_D = 0$), hence: $C \times \Omega = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \times (\beta_R, \beta_D, r_0(.), h_0(.), \theta, \alpha)^\top$

## Value

For SFM and NFM, the *.power() function returns a list with:

- `estimated.power`: the estimated power.
- `number.events`: an array (dim = 2) containing the number of recurrent events under the null and alternative hypotheses, respectively.

The *.ssize() function returns a list with:

- `Groups`: the total number of subjects (recurrent event data) or number of groups (clustered data).
- `number.events`: an array (dim = 2) containing the number of events under the null and alternative hypotheses, respectively.

For JFM and GJFM, the *.power() function returns a list with:

- `estimated.power`: the estimated power.
- `events.rec`: an array (dim = 2) containing the number of recurrent events under the null and alternative hypotheses, respectively.
- `events.D`: an array (dim = 2) containing the number of terminal events under the null and alternative hypotheses, respectively.

The *.ssize() function returns a list with:

- `Npts`: the computed total number of subjects.
- `events.rec`: an array (dim = 2) containing the number of recurrent events under the null and alternative hypotheses, respectively.
- `events.D`: an array (dim = 2) containing the number of terminal events under the null and alternative hypotheses, respectively.

All returned lists additionally include several input parameters: `target.power` or `Groups/Npts` (depending on the called function), `ni`, `FUP`, `FUP.type`, `Acc.Dur`, `ratio`, `data.type`, the test type `testType`, `alpha`, `theta`, `eta` (for NFM and GJFM) and `samplesMC`. For SFM and NFM, we have the corresponding hazard ratios (`HR.H0`, `HR.HA`) from the given betas, `median.H0`. For JGM and GJFM, we have the corresponding hazard ratios (`HR.R0`, `HR.RA`, `HR.D0`, `HR.DA`) from the given betas, `medianR.H0`, `medianD.H0`, and the testing structure `tested.structure`. Along with that, `model` ("SFM", "NFM", "JFM" or "GJFM"), `method` ("power" or "ssize") and `timescale` ("gap" or "calendar") are also included.

All these parameters are utilized by the S3 methods `print.frailtyDesign` and `summary.frailtyDesign` for further detailed output.

## Note

Internally, these functions rely on extensive numerical integration using Gaussian-Laguerre quadrature to approximate the Fisher information. As such, computations may become resource-intensive. You may need to adjust the parameter `samples.mc` or other integration parameters to enhance computational performance or precision.

In this implementation, users must provide both the median time and the shape parameter explicitly; the scale parameter is then computed automatically. Under the Weibull distribution, the median time

$t_{1/2}$ relates to the scale and shape parameters via: $t_{1/2} = \text{scale} \times \log(2)^{1/\text{shape}}$. Consequently, the scale parameter is calculated as: $\text{scale} = \frac{t_{1/2}}{\log(2)^{1/\text{shape}}}$.

For both SFM and NFM, when analyzing grouped data, the arguments `ni.type` and `kij.type` are restricted to the value `"max"` to define an exact sample size. For instance, specifying `ni.type = "pois"` would represent a mean number of subgroups per group, thereby precluding the determination of a precise sample size.

In NFM, the parameter `"rec_event2"` might initially appear difficult to interpret. As clarified by Derek et al. (2024), multitype recurrent events include situations such as transient ischemic attacks classified by anatomical location in cardiovascular studies, or migraines differentiated according to severity in neurological research.

In survival analysis involving recurrent events, the interpretation of regression coefficients ($\beta$) is contingent upon the chosen timescale. This distinction is crucial, as the timescale directly influences the risk assessment and the corresponding interpretation of model parameters.

- When employing a gap timescale, the timescale resets after each event, measuring the duration until the next occurrence. Consequently, the regression coefficients represent the modification of the inter-event risk, reflecting how the treatment influence the hazard of experiencing a subsequent event after the previous one. This approach focuses on the conditional risk between events.
- In contrast, utilizing a calendar timescale measures the time from a fixed origin, such as study entry, without resetting after each event. Here, the regression coefficients pertain to the modification of the risk since the initiation of the study, indicating how the treatment affect the hazard of experiencing events over the entire follow-up period. This approach focuses on the cumulative risk from the study entry.

## Author(s)

Original code by Dinart Derek. Implementation by Adrien Orué.

## References

Derek Dinart, Carine Bellera & Virginie Rondeau (09 Feb 2024). Sample size estimation for recurrent event data using multifrailty and multilevel survival models, *Journal of Biopharmaceutical Statistics*, DOI: 10.1080/10543406.2024.2310306.

## See Also

`frailtyPenal`, `print.frailtyDesign`, `summary.frailtyDesign`

## Examples

```
# Example 1 (SFM): a total of 400 patients (1:1 randomization scheme),
# with a fixed number of 3 recurrent events per patient. Gamma-frailty
# variance of 0.5. Expected hazard ratio of 0.7, time-to-death are uniformly
# distributed, with a mean time to death of (3+10)/2=6.5 years. Each subject is
# followed-up for a maximum of 6 years, with a median time-to-event of 1.5 years.
# Patients are recruited over a 0.5-year period.
SFM.power(
  Groups = 400, ni = 3, ni.type = "max",
```

```
  FUP = 6, Acc.Dur = 0.5, median.H0 = 1.5,
  beta.HA = log(0.7), theta = 0.5,
  cens.par = c(3, 10), cens.type = "Unif",
  data.type = "rec_event"
) # power ~ 90%


# Example 2 (NFM): same parameters as above, but we now assume that we have
# 40 hospitals, 10 subjects per hospital (10 × 40 = 400 subjects in total)
# and 3 recurrent events per subject.
NFM.power(
  Groups = 40, ni = 10, ni.type = "max", kij = 3, kij.type = "max",
  FUP = 6, Acc.Dur = 0.5, median.H0 = 1.5,
  beta.HA = log(0.7), theta = 0.5,
  cens.par = c(3, 10), cens.type = "Unif",
  data.type = "rec_event1"
) # power ~ 83%


# Example 3 (NFM): we aim to compute the required sample size to achieve
# 80% power for detecting a hazard ratio of 0.75 in a neurological study,
# where migraine episodes experienced by subjects are classified into three
# severity subtypes (mild, moderate, severe). For each subject, we anticipate
# a mean number of 2 migraine episodes per severity subtype, with a median
# time-to-event of 6 months. The study duration includes a 1-year accrual period
# followed by a 5-year total follow-up. All subjects will be followed until
# the end of the study.
NFM.ssize(
  power = 0.80, ni = 3, ni.type = "max", kij = 2, kij.type = "pois",
  FUP = 5, Acc.Dur = 1, FUP.type = "uptoend", median.H0 = 0.5,
  beta.HA = log(0.75), data.type = "rec_event2"
) # sample size ~ 363 patients

# Example 4 (JFM): power estimation, testing a treatment effect on recurrent
# events only. We assume a uniformly distributed number of recurrent events,
# ranging from 1 to 6 recurrent events per subject. The allocation ratio
# experimental:control is 2:1, and the follow-up is 10 weeks.
# The expected hazard ratio is 0.70 for recurrent event and 0.90 for the
# terminal event. We have chosen 0.5 as the variance of the frailties.
JFM.power(
  Npts = 400, ni = c(1, 6), ni.type = "unif",
  FUP = 10, FUP.type = "fixed", ratio = 2,
  betaTest.type = "betaRtest", betaR.HA = log(.70), betaD.HA = log(.90),
  theta = .5
) # power ~ 76%


# Example 5 (JFM): sample size calculation, to assess the treatment effect on
# both recurrent and terminal events. We want to achieve an 80% power.
# We anticipate a maximum of 5 recurrent events, over a 6-year period and a
# 0.5-year accrual period. We assume that the gamma-frailty variance is 0.5.
# For the control group, we expect a 2-year and a 5-year median time-to-event
# for recurrent events and terminal events, respectively. We consider a 30%
```

```
# and 20% risk reduction for recurrent events and terminal event, respectively.
JFM.ssize(
  power = 0.80, ni = 9,
  FUP = 6, Acc.Dur = 1.5, medianR.H0 = 2, medianD.H0 = 5,
  betaTest.type = "joint", betaR.HA = log(.70), betaD.HA = log(.80), theta = .5
) # sample size ~ 445 patients / ~ approx


# Example 6: Sample size calculation for GJFM
# Same as above, but with two random effects (with two Gamma-frailty variances
# theta and eta). To ensure sample size estimation stability, we use 10000
# Monte-Carlo samples.
GJFM.ssize(
  power = 0.80, ni = 5,
  FUP = 6, Acc.Dur = 0.5, medianR.H0 = 2, medianD.H0 = 5,
  betaR.HA = log(0.70), betaD.HA = log(0.80), theta = 0.5, eta = 0.75,
  samples.mc = 1e5
) # sample size ~ 705 patients / ~ approx 4 min.


# Example 7:
# --------------------------------------------------
# Post-hoc power analysis for a Joint Frailty Model
# --------------------------------------------------
# See original article by Gonzalez et al. (2005)
data(readmission)
modJFM <- frailtyPenal(
  Surv(time, event) ~ cluster(id) + as.factor(chemo) + terminal(death),
  formula.terminalEvent = ~ as.factor(chemo), data = readmission,
  hazard = "Weibull"
)

# Test both recurrent and death events
# # - Let us assume an underlying Poisson distribution for ni.type. The
# # empirical mean of the number of recurrent events per patients is: ni = 1.136476.
# # - For the null hypothesis, let us consider betaR.H0 = betaD.H0 = 0. For the
# # alternative hypothesis, we use the estimated parameters for betaD.HA and
# # betaR.HA.
# # - "Patients were actively followed up until June 2002" -> the follow-up
# # type is "UpToEnd".
# # - "The study took place in the Hospital de Bellvitge, [...] between
# # January 1996 and December 1998" -> the accrual time is approximately 3 years
# # - We can assume that the study duration is approximately 6 years

ni <- 1.136476
ni.type <- "Pois"
Acc.Dur <- 3 * 365.25 # time unit = days
FUP <- 6 * 365.25     # same as above
betaR.HA <- as.numeric(modJFM$coef[1]) # else "Named numeric"
betaD.HA <- as.numeric(modJFM$coef[2]) # same as above
med <- modJFM$scale.weib * log(2)^(1 / modJFM$shape.weib)
medianR.H0 <- med[1]
medianD.H0 <- med[2]
```

```
shapeR.W <- modJFM$shape[1]
shapeD.W <- modJFM$shape[2]
theta <- modJFM$theta
alpha <- modJFM$alpha
Npts <- length(unique(readmission[, "id"])) # 403 patients
nTreated <- length(unique(readmission[readmission$chemo == "Treated", "id"])) #217 treated patients
ratio <- nTreated / (Npts - nTreated)

JFM.power(
  Npts = Npts, ni = ni, ni.type = ni.type,
  Acc.Dur = Acc.Dur, FUP = FUP, medianR.H0 = medianR.H0, medianD.H0 = medianD.H0,
  betaTest.type = "joint", betaR.HA = betaR.HA, betaD.HA = betaD.HA,
  shapeR.W = shapeR.W, shapeD.W = shapeD.W, theta = theta, alpha = alpha,
  ratio = ratio
) # power ~ 92%

# ----------------------------------------------------
# Required sample size under the same setting
# ----------------------------------------------------
# Here, let us consider that readmission is a “pilot study” with 403 patients,
# from which we estimate parameters. Under this scenario, let us compute the
# needed sample size, but to achieve an 80% power.

JFM.ssize(
  power = 0.80, ni = ni, ni.type = ni.type,
  Acc.Dur = Acc.Dur, FUP = FUP, medianR.H0 = medianR.H0, medianD.H0 = medianD.H0,
  betaTest.type = "joint", betaR.HA = betaR.HA, betaD.HA = betaD.HA,
  shapeR.W = shapeR.W, shapeD.W = shapeD.W, theta = theta, alpha = alpha,
  ratio = ratio
) # 289 patients needed under the same settings vs. 403
```

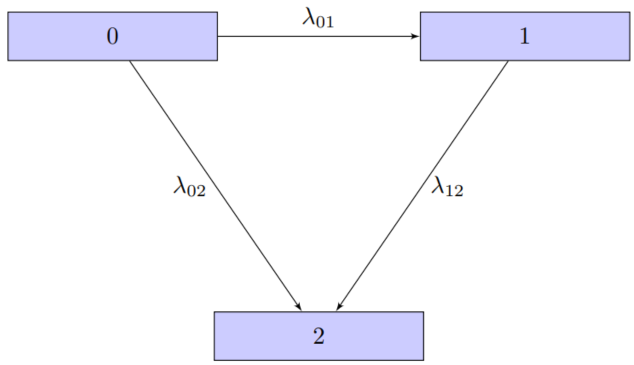
---

`frailtyIllnessDeath` *Fit a Weibull Illness-Death Model with Optional Shared Frailty*

---

## Description

Fit a three-state illness-death model (states: 0=Healthy, 1=Illness, 2=Death) using Weibull baseline hazards for all transitions (0->1, 0->2, 1->2). Allows for shared gamma frailty between the three transitions, acting multiplicatively on the hazards within specified groups (clusters). The model accommodates right-censored and left-truncated data. The transition from the illness state to death (1->2) can be modeled using either a Markov or a Semi-Markov assumption for the baseline hazards time scale.




## Usage

```
frailtyIllnessDeath (formula, formula.terminalEvent, data, model = "Semi-Markov",
maxit = 300, init.B, init.Theta, init.hazard.weib,
LIMparam = 1e-3, LIMlogl = 1e-3, LIMderiv = 1e-3,
partialH, x01, x02, x12, print.info = FALSE,
print.result = TRUE, blinding = TRUE)
```

## Arguments

`formula` A formula object with the response on the left of a ~ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function. Status should be 1 if event 0->1 occurred, 0 otherwise. Covariates for transition 0->1 are specified here. Shared frailty is specified using `cluster(group_variable)`. Left truncation can be included via `Surv(time1, time2, status)`.

`formula.terminalEvent` A formula object. Response must be a `survival::Surv` object representing the time and status for the terminal event (death). Status should be 1 if transition to state 2 occurred (either from state 0 or state 1), 0 otherwise (censored). Covariates for transitions 0->2 and 1->2 are specified on the RHS (currently assumes the same covariates apply to both 0->2 and 1->2).

**Note on Left Truncation:** The illness-death model implemented assumes a **single entry time** per subject. This entry time, specified in `formula`, indicates that the subject must be in the initial state (state 0, having experienced neither transition 0->1 nor 0->2) at that `entry_time` to be included in the analysis. The entry time should **not** be specified again here in `formula.terminalEvent`.

`data` A 'data.frame' with the variables used in the formulas.

`model` Character string specifying the model for the 1->2 transition baseline hazard. Allowed values: "Semi-Markov" (default) or "Markov".

`maxit` Maximum number of iterations for the Marquardt algorithm. Default is 300.

`init.B` Optional. A vector of initial values for regression coefficients. Order is ($\beta_1$,$\beta_2$,$\beta_3$). The total length must match the total number of regression coefficients. If omitted, the regression coefficients are initialized from default values. These defaults are obtained by fitting three independent Weibull proportional hazards models (without frailty), one for each transition.

`init.Theta` Optional. Initial value for the frailty variance $\theta$. Default is 0.1. This parameter is only used if `cluster()` is present in `formula`.

`init.hazard.weib` Optional. A vector of initial values for the Weibull baseline hazard parameters. Must be of size 6, in the order: scale(0->1), shape(0->1), scale(0->2), shape(0->2), scale(1->2), shape(1->2). If omitted, the baseline hazard parameters are initialized from default values. These defaults are obtained by fitting three independent Weibull proportional hazards models (without frailty), one for each transition.

`LIMparam` Convergence threshold for the parameters based on the maximum absolute difference between successive iterations ($10^{-3}$ by default).

`LIMlogl` Convergence threshold for the log-likelihood based on the absolute difference between successive iterations ($10^{-3}$ by default).

`LIMderiv` Convergence threshold based on the relative distance to the optimum (related to gradient and Hessian) ($10^{-3}$ by default). See Details.

`partialH` Optional. Integer vector specifying the indices of parameters to exclude from the Hessian matrix when calculating the relative distance convergence criterion (`LIMderiv`). This is only considered if the first two criteria (`LIMparam`, `LIMlogl`) are met and the full Hessian is problematic (e.g., not invertible). Default is `NULL`.

`x01` Optional. Numeric vector of time points at which to calculate baseline hazard and survival functions for transition 0->1. Defaults to a sequence of 99 points from 0 to the maximum observed time for transition 0->1.

`x02` Optional. Numeric vector of time points at which to calculate baseline hazard and survival functions for transition 0->2. Defaults to a sequence of 99 points from 0 to the maximum observed time for transition 0->2.

`x12` Optional. Numeric vector of time points at which to calculate baseline hazard and survival functions for transition 1->2. Defaults to a sequence of 99 points from 0 to the maximum observed time for transition 1->2.

`print.info` Logical. If `TRUE`, prints information at each iteration of the optimization algorithm. Default is `FALSE`.

`print.result` Logical. If `TRUE`, prints a formatted summary of the results. Default is `TRUE`.

`blinding` Logical. If `TRUE`, the algorithm attempts to continue even if the log-likelihood calculation produces non-finite values (e.g., `Inf`, `NaN`) at some iteration. Setting to `FALSE` may cause the algorithm to stop earlier in such cases. Default is `TRUE`.

## Details

Let $T_1$ be the time to the non-terminal event (illness, 0->1) and $T_2$ be the time to the terminal event (death, 0->2 or 1->2).

The transition intensities are defined as:

$$\lambda_{01}(t) = \lim_{\Delta t \to 0^+} \frac{\mathbb{P}(t \leq T_1 \leq t + \Delta t \mid T_1 \geq t, T_2 \geq t)}{\Delta t}$$

$$\lambda_{02}(t) = \lim_{\Delta t \to 0^+} \frac{\mathbb{P}(t \le T_2 \le t + \Delta t \mid T_1 \ge t, T_2 \ge t)}{\Delta t}$$

$$\lambda_{12}(t \mid T_1 = s) = \lim_{\Delta t \to 0^+} \frac{\mathbb{P}(t \le T_2 \le t + \Delta t \mid T_1 = s, T_2 \ge t)}{\Delta t} \quad (0 < s < t)$$

A proportional hazards model with a shared frailty term $\omega_i$ is assumed for each transition within group $i$. For the $j^{th}$ subject ($j = 1, ..., n_i$) in the $i^{th}$ group ($i = 1, ..., G$) the transition intensities are defined as follows:

$$\lambda_{01}^{ij}(t|\omega_i, X_{01}^{ij}) = \lambda_{0,01}(t)\omega_i \exp(\beta_1^T X_{01}^{ij})$$

$$\lambda_{02}^{ij}(t|\omega_i, X_{02}^{ij}) = \lambda_{0,02}(t)\omega_i \exp(\beta_2^T X_{02}^{ij})$$

$$\lambda_{12}^{ij}(t|T_1 = s, \omega_i, X_{12}^{ij}) = \lambda_{0,12}(t|T_1 = s)\omega_i \exp(\beta_3^T X_{12}^{ij}) \quad (0 < s < t)$$

$\omega_i$ is the frailty term for the $i^{th}$ group. For subject-specific frailties, use `cluster(id)` where id is unique ($n_i = 1$).

$\beta_1$, $\beta_2$ and $\beta_3$ are respectively the vectors of time fixed regression coefficients for the transitions 0->1, 0->2 and 1->2. $X_{01}^{ij}$, $X_{02}^{ij}$ and $X_{12}^{ij}$ are respectively the vectors of time fixed covariates for the $j^{th}$ subject in the $i^{th}$ group for the transitions 0->1, 0->2 and 1->2. $\lambda_{0,01}(.)$, $\lambda_{0,02}(.)$ and $\lambda_{0,12}(.)$ are respectively the baseline hazard functions for the transitions 0->1, 0->2 and 1->2.

The baseline hazard $\lambda_{0,12}(t|T_1 = s)$ depends on the 'model' argument:

- **Markov model**: $\lambda_{0,12}(t|T_1 = s) = \lambda_{0,12}(t)$. The risk depends on the time since origin.This model is suitable when the risk for the transition 1 -> 2 is primarily influenced by absolute time rather than the duration spent in state 1.
- **Semi-Markov model**: $\lambda_{0,12}(t|T_1 = s) = \lambda_{0,12}(t - s)$. The risk depends on the time since entering state 1 (sojourn time). This model is appropriate when the risk for the transition 1 -> 2 is more influenced by the time spent in state 1 rather than the time elapsed since the initial starting point.

The Weibull baseline hazard parameterization is:

$$\lambda(t) = \frac{\gamma}{\lambda^\gamma} \cdot t^{\gamma - 1}$$

where $\lambda$ is the scale parameter and $\gamma$ the shape parameter

## Value

An object of class 'frailtyIllnessDeath ' containing:

**b** Vector of the estimated parameters. Order is: (scale(0->1), shape(0->1), scale(0->2), shape(0->2), scale(1->2), shape(1->2), $\hat{\theta}$ (if frailty), $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\beta}_3$).

**call** The matched function call.

**coef** Vector of estimated regression coefficients.

**loglik** The marginal log-likelihood value at the final parameter estimates.

**grad** Gradient vector of the log-likelihood at the final parameter estimates.

**n** The number of subjects (observations) used in the fit.

**n.events** Vector containing the number of observed events: count for 0->1, count for 0->2, count for 1->2 and count for censoring.

**n.iter** Number of iterations.

**vcov** Variance-covariance matrix for the parameters listed in `b`.

**npar** Total number of estimated parameters.

**nvar** Total number of regression coefficients.

**shape.weib** Vector of estimated Weibull baseline shape parameters (shape(0->1),shape(0->2),shape(1->2)).

**scale.weib** Vector of estimated Weibull baseline scale parameters (scale(0->1),scale(0->2),scale(1->2)).

**crit** Convergence status code: 1=converged, 2=maximum iterations reached, 3=converged using partial Hessian, 4=the algorithm encountered a problem in the loglikelihood computation.

**Frailty** Logical. `TRUE` if a model with shared frailty (`cluster(.)`) was fitted.

**beta_p.value** Vector of p-values from Wald tests for the regression coefficients in `coef`.

**AIC** Akaike Information Criterion, calculated as $AIC = \frac{1}{n}(np - l(.))$, where np is the number of parameters and l is the log-likelihood.

**x01** Vector of time points used for calculating baseline functions for transition 0->1.

**x02** Vector of time points used for calculating baseline functions for transition 0->2.

**x12** Vector of time points (or sojourn times if Semi-Markov) used for calculating baseline functions for transition 1->2.

**lam01** Matrix containing baseline hazard estimates and 95% confidence intervals for transition 0->1 calculated at `x01`.

**lam02** Matrix containing baseline hazard estimates and 95% confidence intervals for transition 0->2 calculated at `x02`.

**lam12** Matrix containing baseline hazard estimates and 95% confidence intervals for transition 1->2 calculated at `x12`.

**surv01** Matrix containing baseline survival estimates and 95% confidence intervals for transition 0->1 calculated at `x01`.

**surv02** Matrix containing baseline survival estimates and 95% confidence intervals for transition 0->2 calculated at `x01`.

**surv12** Matrix containing baseline survival estimates and 95% confidence intervals for transition 0->2 calculated at `x12`.

**median.01** Matrix containing the estimated median baseline survival time and its 95% confidence interval for transition 0->1.

**median.02** Matrix containing the estimated median baseline survival time and its 95% confidence interval for transition 0->2.

**median.12** Matrix containing the estimated median baseline survival time and its 95% confidence interval for transition 1->2.

**linear.pred01** Vector of linear predictors calculated for transition 0->1. For non-frailty models, this is $\hat{\beta}_1^t X_{01}$. For frailty models, it includes the estimated log-frailty: $\hat{\beta}_1^t X_{01} + \log(\hat{\omega}_i)$.

**linear.pred02** Vector of linear predictors calculated for transition 0->2. For non-frailty models, this is $\hat{\beta}_2^t X_{02}$. For frailty models, it includes the estimated log-frailty: $\hat{\beta}_2^t X_{02} + \log(\hat{\omega}_i)$.

**linear.pred12** Vector of linear predictors calculated for transition 1->2. For non-frailty models, this is $\hat{\beta}_3^t X_{12}$. For frailty models, it includes the estimated log-frailty: $\hat{\beta}_3^t X_{12} + \log(\hat{\omega}_i)$.

**names.factor.01** Character vector identifying factor covariates included for transition 0->1.

**names.factor.02** Character vector identifying factor covariates included for transition 0->2.

**names.factor.12** Character vector identifying factor covariates included for transition 1->2.

**global_chisq.01** Vector containing the chi-squared statistics for global Wald tests of factor variables for transition 0->1.

**global_chisq.02** Vector containing the chi-squared statistics for global Wald tests of factor variables for transition 0->2.

**global_chisq.12** Vector containing the chi-squared statistics for global Wald tests of factor variables for transition 1->2.

**dof_chisq.01** Vector containing the degrees of freedom for the global Wald tests for transition 0->1.

**dof_chisq.02** Vector containing the degrees of freedom for the global Wald tests for transition 0->2.

**dof_chisq.12** Vector containing the degrees of freedom for the global Wald tests for transition 1->2.

**p.global_chisq.01** Vector containing the p-values for the global Wald tests for transition 0->1.

**p.global_chisq.02** Vector containing the p-values for the global Wald tests for transition 0->2.

**p.global_chisq.12** Vector containing the p-values for the global Wald tests for transition 1->2.

**global_chisq.test.01** Indicator (0/1) whether any global factor tests were performed for transition 0->1.

**global_chisq.test.02** Indicator (0/1) whether any global factor tests were performed for transition 0->2.

**global_chisq.test.12** Indicator (0/1) whether any global factor tests were performed for transition 1->2.

If `Frailty` is `TRUE`, the following components related to frailty are also included:

**groups** The number of unique groups specified by `cluster(.)`.

**theta** The estimated variance ($\hat{\theta}$) of the Gamma frailty distribution.

**theta_p.value** The p-value from a Wald test for the null hypothesis $H_0 : \theta = 0$.

**VarTheta** The estimated variance of the frailty variance estimator: $\hat{Var}(\hat{\theta})$.

**frailty.pred** Vector containing the empirical Bayes predictions of the frailty term for each group.

**frailty.var** Vector containing the variances of the empirical Bayes frailty predictions.

**frailty.sd** Vector containing the standard errors of the empirical Bayes frailty predictions.

## Note

The optimization uses the robust Marquardt algorithm (Marquardt, 1963), combining Newton-Raphson and steepest descent steps. Iterations stop when criteria `LIMparam`, `LIMlogl`, and `LIMderiv` are all met. Confidence bands for the baseline hazard and baseline survival functions were computed using a Monte Carlo simulation approach based on the estimated Weibull parameters. A sample of size 1000 was drawn from the joint distribution of the shape and scale parameters. For each sampled parameter set, the baseline functions were evaluated over the time grid defined by the vector `x0`. Pointwise 95 confidence bands were then obtained by computing the 2.5th and 97.5th percentiles of the simulated values at each time point for each baseline function.

## References

Lee, C., Gilsanz, P., & Haneuse, S. (2021). Fitting a shared frailty illness-death model to left-truncated semi-competing risks data to examine the impact of education level on incident dementia. *BMC Medical Research Methodology*, 21(1), 1-13.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2), 431-441.

Liquet, B., Timsit, J. F., & Rondeau, V. (2012). Investigating hospital heterogeneity with a multi-state frailty model: application to nosocomial pneumonia disease in intensive care units. *BMC Medical Research Methodology*, 12(1), 1-14.

## Examples

```
data(readmission2)

##-- Fitting models --##

#-- Semi-Markovian Weibull Illness-Death model with
#   group frailty shared between transitions --#

ModIllnessDeath_SemiMarkov_Group <- frailtyIllnessDeath(
  formula = Surv(observed_disease_time, disease_status) ~ cluster(group) + sex,
  formula.terminalEvent = Surv(observed_death_time, death_status) ~ sex,
  data        = readmission2,
  model       = "Semi-Markov",
  print.info  = FALSE,
  maxit       = 100
)

#-- Markovian Weibull Illness-Death model with subject-specific
#   frailty shared between transitions --#

ModIllnessDeath_Markov_Subject <- frailtyIllnessDeath(
  formula = Surv(observed_disease_time, disease_status) ~ cluster(id) + sex,
  formula.terminalEvent = Surv(observed_death_time, death_status) ~ sex,
  data        = readmission2,
  model       = "Markov",
  print.info  = FALSE,
  maxit       = 100
)

#--- Semi-Markovian Weibull Illness-Death model with a factor and
#    no covariates for the non-terminal event ---#

ModIllnessDeath_SemiMarkov_NoCov_Factor <- frailtyIllnessDeath(
  formula = Surv(observed_disease_time, disease_status) ~ 1,
  formula.terminalEvent = Surv(observed_death_time, death_status) ~ factor(dukes),
  data        = readmission2,
  model       = "Semi-Markov",
  print.info  = FALSE,
  maxit       = 100
```

```
)

#--- Semi-Markovian Weibull Illness-Death model with left truncation ---#

data(Paq810)

ModIllnessDeath_SemiMarkov_LeftTrunc <- frailtyIllnessDeath(
  formula = Surv(e, r, dementia) ~ gender + certif,
  formula.terminalEvent = Surv(t, death) ~ certif,
  data       = Paq810,
  model      = "Semi-Markov",
  print.info  = FALSE,
  maxit      = 100
)

#--- Markovian Weibull Illness-Death model with left truncation ---#

ModIllnessDeath_Markov_LeftTrunc <- frailtyIllnessDeath(
  formula = Surv(e, r, dementia) ~ gender + certif,
  formula.terminalEvent = Surv(t, death) ~ certif,
  data       = Paq810,
  model      = "Markov",
  print.info  = FALSE,
  maxit      = 100
)
```

---

frailtyPenal *Fit a Shared, Joint or Nested Frailty model*

---

## Description

### Joint Nested Frailty model

***Data should be ordered according to cluster and subcluster***

Fit a joint model for recurrent and terminal events using a penalized likelihood on the hazard functions or a parametric estimation. Right-censored data are allowed but left-truncated data and stratified analysis are not allowed.

Joint nested frailty models allow studying, jointly, survival processes of recurrent and terminal events for hierarchically clustered data, by considering the terminal event as an informative censoring and by including two iid gamma random effects.

The joint nested frailty model includes two shared frailty terms, one for the subgroup ($u_{fi}$) and one for the group ($\omega_f$) into the hazard functions. This random effects account the heterogeneity in the data, associated with unobserved covariates. The frailty terms act differently for the two rates ($u_{fi}, \omega_f^{\xi}$ for the recurrent rate and $u_{fi}$, $\omega_i$ for the terminal event rate). The covariates could be different for the recurrent rate and death rate.

For the $j^{th}$ recurrence $(j = 1, ..., n_i)$ of the $j^{th}$ individual $(i = 1, ..., m_f)$ of the $f^{th}$ group $(f = 1, ..., n)$, the joint nested gamma frailty model for recurrent event hazard function $r_{fij}(.)$ and for terminal event hazard function $\lambda_{fi}$ is:

$$\begin{cases} r_{fij}(t|\omega_f, u_{fi}, \boldsymbol{X_{fij}}) = r_0(t)u_{fi}\omega_f^{\xi}\exp(\boldsymbol{\beta}'\boldsymbol{X_{fij}(t)}) & \text{(Recurrent)} \\ \lambda_{fi}(t|\omega_f, u_{fi}, \boldsymbol{X_{fij}}) = \lambda_0(t)u_{fi}^{\alpha}\omega_f\exp(\boldsymbol{\gamma}'\boldsymbol{X_{fi}(t)}) & \text{(Death)} \end{cases}$$

where $r_0$(resp. $\lambda_0$) is the recurrent (resp. terminal) event baseline hazard function, $\beta$(resp. $\gamma$) the regression coefficient vector, $\boldsymbol{X_{fij}(t)}$(resp. $\boldsymbol{X_{fi}(t)}$) the covariates vectors. The random effects $\boldsymbol{\omega_f \sim \Gamma(\frac{1}{\eta}, \frac{1}{\eta})}$ and $\boldsymbol{u_{fi} \sim \Gamma(\frac{1}{\theta}, \frac{1}{\theta})}$.

**Shared Frailty model**

Fit a shared gamma or log-normal frailty model using a semiparametric Penalized Likelihood estimation or parametric estimation on the hazard function. Left-truncated, right-censored data, interval-censored data and strata (up to 6 levels) are allowed. It allows to obtain a non-parametric smooth hazard of survival function. This approach is different from the partial penalized likelihood approach of Therneau et al.

The hazard function, conditional on the frailty term $\omega_i$, of a shared gamma frailty model for the $j^{th}$ subject in the $i^{th}$ group:

$$\lambda_{ij}(t|\omega_i) = \lambda_0(t)\omega_i\exp(\boldsymbol{\beta}'\boldsymbol{Z_{ij}})$$

$$\omega_i \sim \Gamma\left(\frac{1}{\theta}, \frac{1}{\theta}\right), \qquad \boldsymbol{E}(\omega_i) = 1, \qquad \boldsymbol{Var}(\omega_i) = \theta$$

where $\lambda_0(t)$ is the baseline hazard function, $\boldsymbol{\beta}$ the vector of the regression coefficient associated to the covariate vector $\boldsymbol{Z_{ij}}$ for the $j^{th}$ individual in the $i^{th}$ group.

Otherwise, in case of a shared log-normal frailty model, we have for the $j^{th}$ subject in the $i^{th}$ group:

$$\lambda_{ij}(t|\eta_i) = \lambda_0(t)\exp(\eta_i + \boldsymbol{\beta}'\boldsymbol{Z_{ij}})$$

$$\eta_i \sim \mathcal{N}(0, \sigma^2)$$

From now on, you can also consider time-varying effects covariates in your model, see `timedep` function for more details.

**Joint Frailty model**

Fit a joint either with gamma or log-normal frailty model for recurrent and terminal events using a penalized likelihood estimation on the hazard function or a parametric estimation. Right-censored data and strata (up to 6 levels) for the recurrent event part are allowed. Left-truncated data is not possible. Joint frailty models allow studying, jointly, survival processes of recurrent and terminal events, by considering the terminal event as an informative censoring.

There is two kinds of joint frailty models that can be fitted with `frailtyPenal`:

- The first one (Rondeau et al. 2007) includes a common frailty term to the individuals $(\omega_i)$ for the two rates which will take into account the heterogeneity in the data, associated with unobserved covariates. The frailty term acts differently for the two rates ($\omega_i$ for the recurrent rate and $\omega_i^{\alpha}$ for the death rate). The covariates could be different for the recurrent rate and death rate.

For the $j^{th}$ recurrence $(j = 1, ..., n_i)$ and the $i^{th}$ subject $(i = 1, ..., G)$, the joint gamma frailty model for recurrent event hazard function $r_{ij}(.)$ and death rate $\lambda_i(.)$ is:

$$\begin{cases} r_{ij}(t|\omega_i) = \omega_i r_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_i(t)}) & \text{(Recurrent)} \\ \lambda_i(t|\omega_i) = \omega_i^\alpha \lambda_0(t) \exp(\boldsymbol{\beta_2'} \boldsymbol{Z_i(t)}) & \text{(Death)} \end{cases}$$

where $r_0(t)$ (resp. $\lambda_0(t)$) is the recurrent (resp. terminal) event baseline hazard function, $\boldsymbol{\beta_1}$ (resp. $\boldsymbol{\beta_2}$) the regression coefficient vector, $\boldsymbol{Z_i(t)}$ the covariate vector. The random effects of frailties $\omega_i \sim \boldsymbol{\Gamma}(\frac{1}{\theta}, \frac{1}{\theta})$ and are iid.

The joint log-normal frailty model will be:

$$\begin{cases} r_{ij}(t|\eta_i) = r_0(t) \exp(\eta_i + \boldsymbol{\beta_1'} \boldsymbol{Z_i(t)}) & \text{(Recurrent)} \\ \lambda_i(t|\eta_i) = \lambda_0(t) \exp(\alpha\eta_i + \boldsymbol{\beta_2'} \boldsymbol{Z_i(t)}) & \text{(Death)} \end{cases}$$

where $\eta_i \sim \mathcal{N}(0, \sigma^2)$

- The second one (Rondeau et al. 2011) is quite similar but the frailty term is common to the individuals from a same group. This model is useful for the joint modelling two clustered survival outcomes. This joint models have been developed for clustered semi-competing events. The follow-up of each of the two competing outcomes stops when the event occurs. In this case, j is for the subject and i for the cluster.

$$\begin{cases} r_{ij}(t|u_i) = u_i r_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_{ij}(t)}) & \text{(Time to event)} \\ \lambda_{ij}(t|u_i) = u_i^\alpha \lambda_0(t) \exp(\boldsymbol{\beta_2'} \boldsymbol{Z_{ij}(t)}) & \text{(Death)} \end{cases}$$

It should be noted that in these models it is not recommended to include $\alpha$ parameter as there is not enough information to estimate it and thus there might be convergence problems.

In case of a log-normal distribution of the frailties, we will have:

$$\begin{cases} r_{ij}(t|v_i) = r_0(t) \exp(v_i + \boldsymbol{\beta_1'} \boldsymbol{Z_{ij}(t)}) & \text{(Time to event)} \\ \lambda_{ij}(t|v_i) = \lambda_0(t) \exp(\alpha v_i + \boldsymbol{\beta_2'} \boldsymbol{Z_{ij}(t)}) & \text{(Death)} \end{cases}$$

where $v_i \sim \mathcal{N}(0, \sigma^2)$

This joint frailty model can also be applied to clustered recurrent events and a terminal event (example on "readmission" data below).

From now on, you can also consider time-varying effects covariates in your model, see `timedep` function for more details.

There is a possibility to use a weighted penalized maximum likelihood approach for nested case-control design, in which risk set sampling is performed based on a single outcome (Jazic et al., *Submitted*).

**General Joint Frailty model** Fit a general joint frailty model for recurrent and terminal events considering two independent frailty terms. The frailty term $u_i$ represents the unobserved association between recurrences and death. The frailty term $v_i$ is specific to the recurrent event rate. Thus, the general joint frailty model is:

$$\begin{cases} r_{ij}(t|u_i, v_i) = u_i v_i r_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_{ij}(t)}) = u_i v_i r_{ij}(t) & \text{(Recurrent)} \\ \lambda_i(t|u_i) = u_i \lambda_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_i(t)}) = u_i \lambda_i(t) & \text{(Death)} \end{cases}$$

where the $iid$ random effects $\boldsymbol{u_i} \sim \Gamma(\frac{1}{\theta}, \frac{1}{\theta})$ and the $iid$ random effects $\boldsymbol{v_i} \sim \Gamma(\frac{1}{\eta}, \frac{1}{\eta})$ are independent from each other. The joint model is fitted using a penalized likelihood estimation on the hazard. Right-censored data and time-varying covariates $\boldsymbol{Z}_i(t)$ are allowed.

**Nested Frailty model**

***Data should be ordered according to cluster and subcluster***

Fit a nested frailty model using a Penalized Likelihood on the hazard function or using a parametric estimation. Nested frailty models allow survival studies for hierarchically clustered data by including two iid gamma random effects. Left-truncated and right-censored data are allowed. Stratification analysis is allowed (maximum of strata = 2).

The hazard function conditional on the two frailties $v_i$ and $\omega_{ij}$ for the $k^{th}$ individual of the $j^{th}$ subgroup of the $i^{th}$ group is:

$$\left\{ \begin{array}{l} \lambda_{ijk}(t|v_i, \omega_{ij}) = v_i \omega_{ij} \lambda_0(t) \exp(\boldsymbol{\beta}' \boldsymbol{X_{ijk}}) \\ v_i \sim \Gamma\left(\frac{1}{\alpha}, \frac{1}{\alpha}\right) \; i.i.d. \;\; \boldsymbol{E}(v_i) = 1 \;\; \boldsymbol{Var}(v_i) = \alpha \\ \omega_{ij} \sim \Gamma\left(\frac{1}{\eta}, \frac{1}{\eta}\right) \; i.i.d. \;\; \boldsymbol{E}(\omega_{ij}) = 1 \;\; \boldsymbol{Var}(\omega_{ij}) = \eta \end{array} \right.$$

where $\lambda_0(t)$ is the baseline hazard function, $X_{ijk}$ denotes the covariate vector and $\beta$ the corresponding vector of regression parameters.

**Joint Nested Frailty Model**

Fit a joint model for recurrent and terminal events using a penalized likelihood on the hazard functions or a parametric estimation. Right-censored data are allowed but left-truncated data and stratified analysis are not allowed.

Joint nested frailty models allow studying, jointly, survival processes of recurrent and terminal events for hierarchically clustered data, by considering the terminal event as an informative censoring and by including two iid gamma random effects.

The joint nested frailty model includes two shared frailty terms, one for the subgroup ($u_{fi}$) and one for the group ($\omega_f$) into the hazard functions. This random effects account the heterogeneity in the data, associated with unobserved covariates. The frailty terms act differently for the two rates ($u_{fi}$, $\omega_f^{\xi}$ for the recurrent rate and $u_{fi}^{\alpha}, \omega_i$ for the terminal event rate). The covariates could be different for the recurrent rate and death rate.

For the $j^{th}$ recurrence (j = 1, ..., $n_i$) of the $i^{th}$ individual (i = 1, ..., $m_f$) of the $f^{th}$ group (f = 1, ..., n), the joint nested gamma frailty model for recurrent event hazard function $r_{fij}$(.) and for terminal event hazard function $\lambda_{fi}$ is:

$$\left\{ \begin{array}{ll} r_{fij}(t|\omega_f, u_{fi}, \boldsymbol{X_{fij}}) = r_0(t) u_{fi} \omega_f^{\xi} \exp(\boldsymbol{\beta' X_{fij}}) & \text{(Recurrent)} \\ \lambda_{fi}(t|\omega_f, u_{fi}, \boldsymbol{X_{fi}}) = \lambda_0(t) u_{fi}^{\alpha} \omega_f \exp(\boldsymbol{\gamma' X_{fi}}) & \text{(Death)} \end{array} \right.$$

where $r_0(t)$(resp. $\lambda_0(t)$) is the recurrent (resp. terminal) event baseline hazard function, $\beta$ (resp. $\gamma$) the regression coefficient vector, $\boldsymbol{X_{fij}}(t)$ the covariates vector. The random effects are $\omega_f \sim \Gamma\left(\frac{1}{\eta}, \frac{1}{\eta}\right)$ and $u_{fi} \sim \Gamma\left(\frac{1}{\theta}, \frac{1}{\theta}\right)$

## Usage

```
frailtyPenal(formula, formula.terminalEvent, data, recurrentAG = FALSE,
cross.validation = FALSE, jointGeneral,n.knots, kappa, maxit = 300, hazard =
```

```
"Splines", nb.int, RandDist = "Gamma", nb.gh, nb.gl, betaknots = 1, betaorder = 3,
initialize = TRUE, init.B, init.Theta, init.Alpha, Alpha, init.hazard.weib, init.Ksi,
Ksi, init.Eta, LIMparam = 1e-3, LIMlogl = 1e-3, LIMderiv = 1e-3, print.times =
TRUE)
```

## Arguments

`formula` a formula object, with the response on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. In case of interval-censored data, the response must be an object as returned by the 'SurvIC' function from this package. Interactions are possible using * or :.

`formula.terminalEvent` only for joint and joint nested frailty models : a formula object, only requires terms on the right to indicate which variables are modelling the terminal event. Interactions are possible using * or :.

`data` a 'data.frame' with the variables used in 'formula'.

`recurrentAG` Logical value. Is Andersen-Gill model fitted? If so indicates that recurrent event times with the counting process approach of Andersen and Gill is used. This formulation can be used for dealing with time-dependent covariates. The default is FALSE.

`cross.validation` Logical value. Is cross validation procedure used for estimating smoothing parameter in the penalized likelihood estimation? If so a search of the smoothing parameter using cross validation is done, with kappa as the seed. The cross validation is not implemented for several strata, neither for interval-censored data. The cross validation has been implemented for a Cox proportional hazard model, with no covariates. The default is FALSE.

`jointGeneral` Logical value. Does the model include two independent random effects? If so, this will fit a general joint frailty model with an association between the recurrent events and a terminal event (explained by the variance $\theta$) and an association amongst the recurrent events (explained by the variance $\eta$).

`n.knots` integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See Note)

`kappa` positive smoothing parameter in the penalized likelihood estimation. In a stratified shared model, this argument must be a vector with kappas for both strata. In a stratified joint model, this argument must be a vector with kappas for both strata for recurrent events plus one kappa for terminal event. The coefficient kappa of the integral of the squared second derivative of hazard function in the fit (penalized log likelihood). To obtain an initial value for `kappa`, a solution is to fit the corresponding shared frailty model using cross validation (See cross.validation). We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them. Value required. (See Note).

`maxit` maximum number of iterations for the Marquardt algorithm. Default is 300

`hazard` Type of hazard functions: "Splines" for semiparametric hazard functions using equidistant intervals or "Splines-per" using percentile with the penalized likelihood estimation, "Piecewise-per" for piecewise constant hazard function using percentile (not available for interval-censored data), "Piecewise-equi" for piecewise constant hazard function using equidistant intervals, "Weibull" for parametric Weibull functions. Default is "Splines". In case of `jointGeneral = TRUE` or if a joint nested frailty model is fitted, only `hazard = "Splines"` can be chosen.

`nb.int` Number of time intervals (between 1 and 20) for the parametric hazard functions ("Piecewise-per", "Piecewise-equi"). In a joint model, you need to specify a number of time interval for both recurrent hazard function and the death hazard function (vector of length 2).

`RandDist` Type of random effect distribution: "Gamma" for a gamma distribution, "LogN" for a log-normal distribution. Default is "Gamma". Not implemented for nested model. If `jointGeneral = TRUE` or if a joint nested frailty model is fitted, the log-normal distribution cannot be chosen.

`nb.gh` Number of nodes for the Gaussian-Hermite quadrature. It can be chosen among 5, 7, 9, 12, 15, 20, 32 and 50. The default is 20 if hazard = "Splines", 32 otherwise.

`nb.gl` Number of nodes for the Gaussian-Laguerre quadrature. It can be chosen among 20, 32 and 50. The default is 20 if hazard = "Splines", 32 otherwise.

`betaknots` Number of inner knots used for the estimation of B-splines. Default is 1. See 'timedep' function for more details. Not implemented for nested and joint nested frailty models.

`betaorder` Order of the B-splines. Default is cubic B-splines (order = 3). See 'timedep' function for more details. Not implemented for nested and joint nested frailty models.

`initialize` Logical value, only for joint nested frailty models. Option `TRUE` indicates fitting an appropriate standard joint frailty model (without group effect, only the subgroup effect) to provide initial values for the joint nested model. Default is `TRUE`.

`init.B` A vector of initial values for regression coefficients. This vector should be of the same size as the whole vector of covariates with the first elements for the covariates related to the recurrent events and then to the terminal event (interactions in the end of each component). Default is 0.1 for each (for Cox and shared model) or 0.5 (for joint and joint nested frailty models).

`init.Theta` Initial value for variance of the frailties.

`init.Alpha` Only for joint and joint nested frailty models : initial value for parameter alpha.

`Alpha` Only for joint and joint nested frailty model : input "None" so as to fit a joint model without the parameter alpha.

`init.hazard.weib`
Only if a weibull model is used. A vector of initial and positive values for the hazard function parameters. This vector is of size 2 for a shared frailty model, is of size 2 * n.strat for a stratified shared frailty model, is of size 2 for nested frailty model, is of size 4 for a stratified nested frailty model (the order

is: shapeStrat1, scaleStrat1, shapeStrat2, scaleStrat2...), is of size 4 for Joint frailty model (the order is shapeR, scaleR, shapeD, scaleD), is of size 6 in case of a joint stratified frailty model as stratification in this case is only possible for recurrent event (order in case of a joint stratified frailty model is (shapeRStrat1, scaleRStrat1,shapeRStrat2, scaleRStrat2, shapeD, scaleD) . Default is 0.1 for each for shared frailty model and stratified shared frailty model, 0.8 for each for nested frailty model and stratified nested frailty model, 0.5 for each for Joint frailty model and joint stratified frailty model.

`init.Ksi` Only for joint nested frailty model : initial value for parameter $\xi$.

`Ksi` Only for joint nested frailty model : input `"None"` indicates a joint nested frailty model without the parameter $\xi$.

`init.Eta` Only for general joint and joint nested frailty models : initial value for the variance $\eta$ of the frailty $v_i$ (general joint model) and of the frailty $\omega_i$ (joint nested frailty model).

`LIMparam` Convergence threshold of the Marquardt algorithm for the parameters (see Details), $10^{-3}$ by default.

`LIMlogl` Convergence threshold of the Marquardt algorithm for the log-likelihood (see Details), $10^{-3}$ by default.

`LIMderiv` Convergence threshold of the Marquardt algorithm for the gradient (see Details), $10^{-3}$ by default.

`print.times` a logical parameter to print iteration process. Default is TRUE.

## Details

Typical usages are for a Cox model

```
frailtyPenal(Surv(time,event)~var1+var2, data, \dots)
```

for a shared model

```
frailtyPenal(Surv(time,event)~cluster(group)+var1+var2, data,
\dots)
```

for a joint model

```
frailtyPenal(Surv(time,event)~cluster(group)+var1+var2+
var3+terminal(death), formula.terminalEvent=~ var1+var4, data, \dots)
```

for a joint model for clustered data

```
frailtyPenal(Surv(time,event)~cluster(group)+num.id(group2)+
var1+var2+var3+terminal(death), formula.terminalEvent=~var1+var4, data,
\dots)
```

for a joint model for data from nested case-control studies

```
frailtyPenal(Surv(time,event)~cluster(group)+num.id(group2)+
var1+var2+var3+terminal(death)+wts(wts.ncc),
formula.terminalEvent=~var1+var4, data, \dots)
```

for a nested model

```
frailtyPenal(Surv(time,event)~cluster(group)+subcluster(sbgroup)+
var1+var2, data, \dots)
```

for a joint nested frailty model

```
frailtyPenal(Surv(time,event)~cluster(group)+subcluster(sbgroup)+
var1+var2++terminal(death), formula.terminalEvent=~var1+var4, data, \dots)
```

The estimated parameter are obtained using the robust Marquardt algorithm (Marquardt, 1963) which is a combination between a Newton-Raphson algorithm and a steepest descent algorithm. The iterations are stopped when the difference between two consecutive log-likelihoods was small ($< 10^{-3}$), the estimated coefficients were stable (consecutive values ($< 10^{-3}$), and the gradient small enough ($< 10^{-3}$). When frailty parameter is small, numerical problems may arise. To solve this problem, an alternative formula of the penalized log-likelihood is used (see Rondeau, 2003 for further details). Cubic M-splines of order 4 are used for the hazard function, and I-splines (integrated M-splines) are used for the cumulative hazard function.

The inverse of the Hessian matrix is the variance estimator and to deal with the positivity constraint of the variance component and the spline coefficients, a squared transformation is used and the standard errors are computed by the $\Delta$-method (Knight & Xekalaki, 2000).

Two variance matrices are provided in the output: `varH` is the variance matrix of all parameters and `varHIH` is the robust estimation of the variance matrix, that takes into account the penalization. When fitting a joint frailty model, `varHIH` is used to compute the p-values of all the parameters.

The smooth parameter can be chosen by maximizing a likelihood cross validation criterion (Joly and other, 1998). The integrations in the full log likelihood were evaluated using Gaussian quadrature. Laguerre polynomials with 20 points were used to treat the integrations on $[0, \infty[$

**INITIAL VALUES**

The splines and the regression coefficients are initialized to 0.1. In case of shared model, the program fits, firstly, an adjusted Cox model to give new initial values for the splines and the regression coefficients. The variance of the frailty term $\theta$ is initialized to 0.1. Then, a shared frailty model is fitted.

In case of a joint frailty model, the splines and the regression coefficients are initialized to 0.5. The program fits an adjusted Cox model to have new initial values for the regression and the splines coefficients. The variance of the frailty term $\theta$ and the coefficient $\alpha$ associated in the death hazard function are initialized to 1. Then, it fits a joint frailty model.

In case of a general joint frailty model we need to initialize the `jointGeneral` logical value to `TRUE`.

In case of a nested model, the program fits an adjusted Cox model to provide new initial values for the regression and the splines coefficients. The variances of the frailties are initialized to 0.1. Then, a shared frailty model with covariates with only subgroup frailty is fitted to give a new initial value for the variance of the subgroup frailty term. Then, a shared frailty model with covariates and only group frailty terms is fitted to give a new initial value for the variance of the group frailties. In a last step, a nested frailty model is fitted.

In case of a joint nested model, the splines and the regression coefficients are initialized to 0.5 and the variances of the frailty terms $\eta$ and $\xi$ are initialized to 1. If the option `'initialize'` is `TRUE`, the program fits a joint frailty model to provide initial values for the splines, covariates coefficients, variance $\theta$ of the frailty terms and $\alpha$. The variances of the second frailty term ($\eta$) and the second coefficient $\xi$ are initialized to 1. Then, a joint nested frailty model is fitted.

**NCC DESIGN**

It is possible to fit a joint frailty model for data from nested case-control studies using the approach of weighted penalized maximum likelihood. For this model, only splines can be used for baseline hazards and no time-varying effects of covariates can be included. To accommodate the nested case-control design, the formula for the recurrent events should simply include the special term wts(wts.ncc), where wts.ncc refers to a column of prespecified weights in the data set for every observation. For details, see Jazic et al., *Submitted* (available on request from the package authors).

## Value

The following components are included in a 'frailtyPenal' object for each model.

| | |
|---|---|
| `b` | sequence of the corresponding estimation of the coefficients for the hazard functions (parametric or semiparametric), the random effects variances and the regression coefficients. |
| `call` | The code used for the model. |
| `formula` | the formula part of the code used for the model. |
| `coef` | the regression coefficients. |
| `cross.Val` | Logical value. Is cross validation procedure used for estimating the smoothing parameters in the penalized likelihood estimation? |
| `DoF` | Degrees of freedom associated with the "kappa". |
| `groups` | the maximum number of groups used in the fit. |
| `kappa` | A vector with the smoothing parameters in the penalized likelihood estimation corresponding to each baseline function as components. |
| `loglikPenal` | the complete marginal penalized log-likelihood in the semiparametric case. |
| `loglik` | the marginal log-likelihood in the parametric case. |
| `n` | the number of observations used in the fit. |
| `n.events` | the number of events observed in the fit. |
| `n.iter` | number of iterations needed to converge. |
| `n.knots` | number of knots for estimating the baseline functions in the penalized likelihood estimation. |
| `n.strat` | number of stratum. |
| `varH` | the variance matrix of all parameters before positivity constraint transformation. Then, the delta method is needed to obtain the estimated variance parameters. That is why some variances don't match with the printed values at the end of the model. |
| `varHIH` | the robust estimation of the variance matrix of all parameters. |

x
: matrix of times where both survival and hazard function are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times.

lam
: Matrix (dim=3) of hazard estimates and confidence bands.

surv
: Matrix (dim=3) of baseline survival estimates and confidence bands.

median
: The value of the median survival and its confidence bands. If there are two stratas or more, the first value corresponds to the value for the first strata, etc.

nbintervR
: Number of intervals (between 1 and 20) for the parametric hazard functions ("Piecewise-per", "Piecewise-equi").

npar
: number of parameters.

nvar
: number of explanatory variables.

LCV
: the approximated likelihood cross-validation criterion in the semiparametric case (with H minus the converged Hessian matrix, and l(.) the full log-likelihood).

$$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$

AIC
: the Akaike information Criterion for the parametric case.

$$AIC = \frac{1}{n}(np - l(.))$$

n.knots.temp
: initial value for the number of knots.

shape.weib
: shape parameter for the Weibull hazard function.

scale.weib
: scale parameter for the Weibull hazard function.

martingale.res
: martingale residuals for each cluster.

martingaleCox
: martingale residuals for observation in the Cox model.

Frailty
: Logical value. Was model with frailties fitted ?

frailty.pred
: empirical Bayes prediction of the frailty term (ie, using conditional posterior distributions).

frailty.var
: variance of the empirical Bayes prediction of the frailty term (only for gamma frailty models).

frailty.sd
: standard error of the frailty empirical Bayes prediction (only for gamma frailty models).

global_chisq
: a vector with the values of each multivariate Wald test.

dof_chisq
: a vector with the degree of freedom for each multivariate Wald test.

global_chisq.test
: a binary variable equals to 0 when no multivariate Wald is given, 1 otherwise.

p.global_chisq
: a vector with the p_values for each global multivariate Wald test.

names.factor
: Names of the "as.factor" variables.

Xlevels
: vector of the values that factor might have taken.

contrasts
: type of contrast for factor variable.

beta_p.value
: p-values of the Wald test for the estimated regression coefficients.

The following components are specific to **shared** models.

- `equidistant` Indicator for the intervals used the estimation of baseline hazard functions (for splines or pieceiwse-constaant functions) : 1 for equidistant intervals ; 0 for intervals using percentile (note: `equidistant` = 2 in case of parametric estimation using Weibull distribution).
- `intcens` Logical value. Indicator if a joint frailty model with interval-censored data was fitted)
- `theta` variance of the gamma frailty parameter $(\boldsymbol{Var}(\omega_i))$
- `sigma2` variance of the log-normal frailty parameter $(\boldsymbol{Var}(\eta_i))$
- `linear.pred` linear predictor: uses simply "Beta'X" in the cox proportional hazard model or "Beta'X + log w_i" in the shared gamma frailty models, otherwise uses "Beta'X + w_i" for log-normal frailty distribution.
- `BetaTpsMat` matrix of time varying-effects and confidence bands (the first column used for abscissa of times)
- `theta_p.value` p-value of the Wald test for the estimated variance of the gamma frailty.
- `sigma2_p.value` p-value of the Wald test for the estimated variance of the log-normal frailty.

The following components are specific to **joint** models.

- `intcens` Logical value. Indicator if a joint frailty model with interval-censored data was fitted)
- `theta` variance of the gamma frailty parameter $(\boldsymbol{Var}(\omega_i))$ or $(\boldsymbol{Var}(u_i))$
- `sigma2` variance of the log-normal frailty parameter $(\boldsymbol{Var}(\eta_i))$ or $(\boldsymbol{Var}(v_i))$
- `eta` variance of the second gamma frailty parameter in general joint frailty models $(\boldsymbol{Var}(v_i))$
- `indic_alpha` indicator if a joint frailty model with $\alpha$ parameter was fitted
- `alpha` the coefficient $\alpha$ associated with the frailty parameter in the terminal hazard function.
- `nbintervR` Number of intervals (between 1 and 20) for the recurrent parametric hazard functions ("Piecewise-per", "Piecewise-equi").
- `nbintervDC` Number of intervals (between 1 and 20) for the death parametric hazard functions ("Piecewise-per", "Piecewise-equi").
- `nvar` A vector with the number of covariates of each type of hazard function as components.
- `nvarRec` number of recurrent explanatory variables.
- `nvarEnd` number of death explanatory variables.
- `noVar1` indicator of recurrent explanatory variables.
- `noVar2` indicator of death explanatory variables.
- `xR` matrix of times where both survival and hazard function are estimated for the recurrent event. By default seq(0,max(time),length=99), where time is the vector of survival times.
- `xD` matrix of times for the terminal event.

`lamR` Matrix (dim=3) of hazard estimates and confidence bands for recurrent event.

`lamD` the same value as lamR for the terminal event.

`survR` Matrix (dim=3) of baseline survival estimates and confidence bands for recurrent event.

`survD` the same value as survR for the terminal event.

`martingale.res` martingale residuals for each cluster (recurrent).

`martingaledeath.res` martingale residuals for each cluster (death).

`linear.pred` linear predictor: uses "Beta'X + log w_i" in the gamma frailty model, otherwise uses "Beta'X + eta_i" for log-normal frailty distribution

`lineardeath.pred` linear predictor for the terminal part : "Beta'X + alpha.log w_i" for gamma, "Beta'X + alpha.eta_i" for log-normal frailty distribution

`Xlevels` vector of the values that factor might have taken for the recurrent part.

`contrasts` type of contrast for factor variable for the recurrent part.

`Xlevels2` vector of the values that factor might have taken for the death part.

`contrasts2` type of contrast for factor variable for the death part.

`BetaTpsMat` matrix of time varying-effects and confidence bands for recurrent event (the first column used for abscissa of times of recurrence)

`BetaTpsMatDc` matrix of time varying-effects and confidence bands for terminal event (the first column used for abscissa of times of death)

`alpha_p.value` p-value of the Wald test for the estimated $\alpha$.

`ncc` Logical value whether nested case-control design with weights was used for the joint model.

The following components are specific to **nested** models.

`alpha` variance of the cluster effect $(\boldsymbol{Var}(v_i))$

`eta` variance of the subcluster effect $(\boldsymbol{Var}(\omega_{ij}))$

`subgroups` the maximum number of subgroups used in the fit.

`frailty.pred.group` empirical Bayes prediction of the frailty term by group.

`frailty.pred.subgroup` empirical Bayes prediction of the frailty term by subgroup.

`linear.pred` linear predictor: uses "Beta'X + log v_i.w_ij".

`subgbyg` subgroup by group.

`n.strat` A vector with the number of covariates of each type of hazard function as components.

`alpha_p.value` p-value of the Wald test for the estimated variance of the cluster effect.

`eta_p.value` p-value of the Wald test for the estimated variance of the subcluster effect.

The following components are specific to **joint nested frailty** models.

| | |
|---|---|
| theta | variance of the subcluster effect $(\boldsymbol{Var}(u_{fi}))$ |
| eta | variance of the cluster effect $(\boldsymbol{Var}(\omega_f))$ |
| alpha | the power coefficient $\alpha$ associated with the frailty parameter $(u_{fi})$ in the terminal event hazard function. |
| ksi | the power coefficient $\xi$ associated with the frailty parameter $(\omega_f)$ in the recurrent event hazard function. |
| indic_alpha | indicator if a joint frailty model with $\alpha$ parameter was fitted or not. |
| indic_ksi | indicator if a joint frailty model with $\xi$ parameter was fitted or not. |
| frailty.fam.pred | empirical Bayes prediction of the frailty term by family. |
| eta_p.value | p-value of the Wald test for the estimated variance of the cluster effect. |
| alpha_p.value | p-value of the Wald test for the estimated power coefficient $\alpha$. |
| ksi_p.value | p-value of the Wald test for the estimated power coefficient $\xi$. |

## Note

From a prediction aim, we recommend you to input a data sorted by the group variable with numerical numbers from 1 to n (number of groups). In case of a nested model, we recommend you to input a data sorted by the group variable then sorted by the subgroup variable both with numerical numbers from 1 to n (number of groups) and from 1 to m (number or subgroups). "kappa" and "n.knots" are the arguments that the user have to change if the fitted model does not converge. "n.knots" takes integer values between 4 and 20. But with n.knots=20, the model would take a long time to converge. So, usually, begin first with n.knots=7, and increase it step by step until it converges. "kappa" only takes positive values. So, choose a value for kappa (for instance 10000), and if it does not converge, multiply or divide this value by 10 or 5 until it converges.

## References


I. Jazic, S. Haneuse, B. French, G. MacGrogan, and V. Rondeau. Design and analysis of nested case-control studies for recurrent events subject to a terminal event. *Submitted*.

A. Krol, A. Mauguen, Y. Mazroui, A. Laurent, S. Michiels and V. Rondeau (2017). Tutorial in Joint Modeling and Prediction: A Statistical Software for Correlated Longitudinal Outcomes, Recurrent Events and a Terminal Event. *Journal of Statistical Software* **81**(3), 1-52.

V. Rondeau, Y. Mazroui and J. R. Gonzalez (2012). Frailtypack: An R package for the analysis of correlated survival data with frailty models using penalized likelihood estimation or parametric estimation. *Journal of Statistical Software* **47**, 1-28.

Y. Mazroui, S. Mathoulin-Pelissier, P. Soubeyranb and V. Rondeau (2012) General joint frailty model for recurrent event data with a dependent terminalevent: Application to follicular lymphoma data. *Statistics in Medecine*, **31**, 11-12, 1162-1176.

V. Rondeau, J.P. Pignon, S. Michiels (2011). A joint model for the dependance between clustered times to tumour progression and deaths: A meta-analysis of chemotherapy in head and neck cancer. *Statistical methods in medical research* **897**, 1-19.

V. Rondeau, S. Mathoulin-Pellissier, H. Jacqmin-Gadda, V. Brouste, P. Soubeyran (2007). Joint frailty models for recurring events and death using maximum penalized likelihood estimation:application on cancer events. *Biostatistics* **8**,4, 708-721.

V. Rondeau, L. Filleul, P. Joly (2006). Nested frailty models using maximum penalized likelihood estimation. *Statistics in Medicine*, **25**, 4036-4052.

V. Rondeau, D. Commenges, and P. Joly (2003). Maximum penalized likelihood estimation in a gamma-frailty model. *Lifetime Data Analysis* **9**, 139-153.

C.A. McGilchrist, and C.W. Aisbett (1991). Regression with frailty in survival analysis. *Biometrics* **47**, 461-466.

D. Marquardt (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematics*, 431-441.

## See Also

SurvIC, cluster, subcluster, terminal, num.id, timedep

## Examples

```
### ---  COX proportional hazard model (SHARED without frailties) ---###
### ---  estimated with penalized likelihood ---###

data(kidney)
frailtyPenal(Surv(time, status) ~ sex + age,
  n.knots = 12, kappa = 10000, data = kidney
)

### ---  Shared Frailty model  ---###

frailtyPenal(Surv(time, status) ~ cluster(id) + sex + age,
  n.knots = 12, kappa = 10000, data = kidney
)

#-- with an initialisation of regression coefficients

frailtyPenal(Surv(time, status) ~ cluster(id) + sex + age,
  n.knots = 12, kappa = 10000, data = kidney, init.B = c(-1.44, 0)
)

#-- with truncated data

data(dataNested)

frailtyPenal(Surv(t1, t2, event) ~ cluster(group),
  data = dataNested, n.knots = 10, kappa = 10000,
  cross.validation = TRUE, recurrentAG = FALSE
)

#-- stratified analysis

data(readmission)
frailtyPenal(Surv(time, event) ~ cluster(id) + dukes + strata(sex),
  n.knots = 10, kappa = c(10000, 10000), data = readmission
)
```

```
#-- recurrentAG=TRUE

frailtyPenal(Surv(t.start, t.stop, event) ~ cluster(id) + sex + dukes +
  charlson, data = readmission, n.knots = 6, kappa = 1e5, recurrentAG = TRUE)

#-- cross.validation=TRUE

frailtyPenal(
  Surv(t.start, t.stop, event) ~ cluster(id) + sex + dukes +
    charlson,
  data = readmission, n.knots = 6, kappa = 5000, recurrentAG = TRUE,
  cross.validation = TRUE
)

#-- log-normal distribution

frailtyPenal(
  Surv(t.start, t.stop, event) ~ cluster(id) + sex + dukes +
    charlson,
  data = readmission, n.knots = 6, kappa = 5000, recurrentAG = TRUE,
  RandDist = "LogN"
)

### --- Joint Frailty model (recurrent and terminal events) ---###

data(readmission)
#-- Gap-time
modJoint.gap <- frailtyPenal(
  Surv(time, event) ~ cluster(id) + sex + dukes + charlson +
    terminal(death),
  formula.terminalEvent = ~ sex + dukes + charlson,
  data = readmission, n.knots = 14, kappa = c(9.55e+9, 1.41e+12),
  recurrentAG = FALSE
)

#-- Calendar time
modJoint.calendar <- frailtyPenal(
  Surv(t.start, t.stop, event) ~ cluster(id) +
    sex + dukes + charlson + terminal(death),
  formula.terminalEvent = ~ sex
  + dukes + charlson, data = readmission, n.knots = 10, kappa = c(9.55e9, 1.41e12),
  recurrentAG = TRUE
)

#-- without alpha parameter
modJoint.gap <- frailtyPenal(
  Surv(time, event) ~ cluster(id) + sex + dukes + charlson +
    terminal(death),
  formula.terminalEvent = ~ sex + dukes + charlson,
  data = readmission, n.knots = 10, kappa = c(9.55e9, 1.41e12),
  recurrentAG = FALSE, Alpha = "None"
)
```

```
#-- log-normal distribution

modJoint.log <- frailtyPenal(
  Surv(t.start, t.stop, event) ~ cluster(id) + sex
    + dukes + charlson + terminal(death),
  formula.terminalEvent = ~ sex
  + dukes + charlson, data = readmission, n.knots = 10, kappa = c(9.55e9, 1.41e12),
  recurrentAG = TRUE, RandDist = "LogN"
)

### --- Joint frailty model for NCC data ---###
data(dataNCC)
modJoint.ncc <- frailtyPenal(
  Surv(t.start, t.stop, event) ~ cluster(id) + cov1
    + cov2 + terminal(death) + wts(ncc.wts),
  formula.terminalEvent = ~ cov1 + cov2,
 data = dataNCC, n.knots = 8, kappa = c(1.6e+10, 5.0e+03), recurrentAG = TRUE, RandDist = "LogN"
)


### --- Joint Frailty model for clustered data ---###

#-- here is generated cluster (5 clusters)
readmission <- transform(readmission, group = id %% 5 + 1)

#-- exclusion all recurrent events --#
#--  to obtain framework of semi-competing risks --#
readmission2 <- subset(readmission, (t.start == 0 & event == 1) | event == 0)

joi.clus.gap <- frailtyPenal(
  Surv(time, event) ~ cluster(group) +
    num.id(id) + dukes + charlson + sex + chemo + terminal(death),
  formula.terminalEvent = ~ dukes + charlson + sex + chemo,
  data = readmission2, recurrentAG = FALSE, n.knots = 8,
  kappa = c(1.e+10, 1.e+10), Alpha = "None"
)


### --- General Joint model (recurrent and terminal events)
### --- with 2 covariates ---###

data(readmission)
modJoint.general <- frailtyPenal(
  Surv(time, event) ~ cluster(id) + dukes +
    charlson + sex + chemo + terminal(death),
  formula.terminalEvent = ~ dukes + charlson + sex + chemo,
  data = readmission, jointGeneral = TRUE, n.knots = 8,
  kappa = c(2.11e+08, 9.53e+11)
)


### --- Nested Frailty model ---###
```

```
##***** WARNING *****##
# Data should be ordered according to cluster and subcluster

data(dataNested)
modClu <- frailtyPenal(
  Surv(t1, t2, event) ~ cluster(group) +
    subcluster(subgroup) + cov1 + cov2,
  data = dataNested,
  n.knots = 8, kappa = 50000
)

modClu.str <- frailtyPenal(
  Surv(t1, t2, event) ~ cluster(group) +
    subcluster(subgroup) + cov1 + strata(cov2),
  data = dataNested,
  n.knots = 8, kappa = c(50000, 50000)
)

## Not run:
### --- Joint Nested Frailty model ---###

#-- here is generated cluster (30 clusters)
readmissionNested <- transform(readmission, group = id %% 30 + 1)

modJointNested_Splines <- frailtyPenal(
  formula = Surv(t.start, t.stop, event)
  ~ subcluster(id) + cluster(group) + dukes + terminal(death),
  formula.terminalEvent = ~dukes, data = readmissionNested, recurrentAG = TRUE,
  n.knots = 8, kappa = c(9.55e+9, 1.41e+12), initialize = TRUE
)

modJointNested_Weib <- frailtyPenal(
  Surv(t.start, t.stop, event) ~ subcluster(id)
    + cluster(group) + dukes + terminal(death),
  formula.terminalEvent = ~dukes,
  hazard = ("Weibull"), data = readmissionNested, recurrentAG = TRUE, initialize = FALSE
)

JoiNesGapSpline <- frailtyPenal(
  formula = Surv(time, event)
  ~ subcluster(id) + cluster(group) + dukes + terminal(death),
  formula.terminalEvent = ~dukes, data = readmissionNested,
  recurrentAG = FALSE, n.knots = 8, kappa = c(9.55e+9, 1.41e+12),
  initialize = TRUE, init.Alpha = 1.091, Ksi = "None"
)

## End(Not run)
```

---

gastadj *Advanced Gastric Cancer dataset*

---

## Description

This meta-analysis was carried out by the GASTRIC (Global Advanced/Adjuvant Stomach Tumor Research international Collaboration) group, using individual data on patients with curatively resected gastric cancer. Data from all published randomized trials, with a patient recruitment end date before 2004, and comparing adjuvant chemotherapy with surgery alone for resectable gastric cancers, were searched electronically. The candidate surrogate endpoint **S** was Desease-free survival time, defined as the time (in days) to relapse, second cancer or dead from any cause. The true endpoint **T** was the overall survival time, defined as the time (in days) from randomization to death of any cause or to the last follow-up.

## Usage

```
data(gastadj)
```

## Format

This data frame contains the following columns:

**trialID** The trial in which the patient was treated

**patientID** The identification number of a patient

**trt** The treatment indicator, coded as 0 = Control and 1 = Experimental

**timeS** The candidate surrogate (progression-free survival in days)

**statusS** Censoring indicator for for Progression-free survival (0 = alive and progression-free, 1 = with progression or dead)

**timeT** The true endpoint (overall survival time in days)

**statusT** Censoring indicator for survival time (0 = alive, 1 = dead)

## Source

Oba K, Paoletti X, Alberts S, Bang YJ, Benedetti J, Bleiberg H, Catalona P, Lordick F, Michiels S, Morita A, Okashi Y, Pignon JP, Rougier P, Sasako M, Sakamoto J, Sargent D, Shitara K, Van Cutsem E, Buyse M, Burzykowski T on behalf of the GASTRIC group (2013). Disease-Free Survival as a Surrogate for Overall Survival in Adjuvant Trials of Gastric Cancer: A Meta-Analysis. *JNCI: Journal of the National Cancer Institute*;**105(21)**:1600-1607

---

`GenfrailtyPenal` *Fit a Shared or a Joint Frailty Generalized Survival Model*

---

## Description

### Shared Frailty model

Fit a shared gamma or log-normal frailty model using a semiparametric Penalized Likelihood estimation or parametric estimation on the hazard function. Left-truncated, right-censored data, interval-censored data and strata (up to 6 levels) are allowed. It allows to obtain a non-parametric smooth hazard of survival function. This approach is different from the partial penalized likelihood approach of Therneau et al.

The hazard function, conditional on the frailty term $\omega_i$, of a shared gamma frailty model for the $j^{th}$ subject in the $i^{th}$ group:

$$\lambda_{ij}(t|\omega_i) = \lambda_0(t)\omega_i \exp(\boldsymbol{\beta}^{'} \boldsymbol{Z_{ij}})$$

$$\omega_i \sim \Gamma\left(\frac{1}{\theta}, \frac{1}{\theta}\right) \qquad \boldsymbol{E}(\omega_i) = 1 \qquad \boldsymbol{Var}(\omega_i) = \theta$$

where $\lambda_0(t)$ is the baseline hazard function, $\boldsymbol{\beta}$ the vector of the regression coefficient associated to the covariate vector $\boldsymbol{Z_{ij}}$ for the $j^{th}$ individual in the $i^{th}$ group.

Otherwise, in case of a shared log-normal frailty model, we have for the $j^{th}$ subject in the $i^{th}$ group:

$$\lambda_{ij}(t|\eta_i) = \lambda_0(t) \exp(\eta_i + \boldsymbol{\beta}^{'} \boldsymbol{Z_{ij}})$$

$$\eta_i \sim N(0, \sigma^2)$$

From now on, you can also consider time-varying effects covariates in your model, see `timedep` function for more details.

**Joint Frailty model**

Fit a joint either with gamma or log-normal frailty model for recurrent and terminal events using a penalized likelihood estimation on the hazard function or a parametric estimation. Right-censored data and strata (up to 6 levels) for the recurrent event part are allowed. Left-truncated data is not possible. Joint frailty models allow studying, jointly, survival processes of recurrent and terminal events, by considering the terminal event as an informative censoring.

There is two kinds of joint frailty models that can be fitted with `GenfrailtyPenal` :

- The first one (Rondeau et al. 2007) includes a common frailty term to the individuals $(\omega_i)$ for the two rates which will take into account the heterogeneity in the data, associated with unobserved covariates. The frailty term acts differently for the two rates ( $\omega_i$ for the recurrent rate and $\omega_i^{\alpha}$ for the death rate). The covariates could be different for the recurrent rate and death rate.

For the $j^{th}$ recurrence $(j = 1, ..., n_i)$ and the $i^{th}$ subject $(i = 1, ..., G)$, the joint gamma frailty model for recurrent event hazard function $r_{ij}(.)$ and death rate $\lambda_i(.)$ is :

$$\left\{ \begin{array}{ll} r_{ij}(t|\omega_i) = \omega_i r_0(t) \exp(\boldsymbol{\beta_1^{'}} \boldsymbol{Z_i(t)}) & \text{(Recurrent)} \\ \lambda_i(t|\omega_i) = \omega_i^{\alpha} \lambda_0(t) \exp(\boldsymbol{\beta_2^{'}} \boldsymbol{Z_i(t)}) & \text{(Death)} \end{array} \right.$$

where $r_0(t)$ (resp. $\lambda_0(t)$) is the recurrent (resp. terminal) event baseline hazard function, $\boldsymbol{\beta_1}$ (resp. $\boldsymbol{\beta_2}$) the regression coefficient vector, $\boldsymbol{Z_i(t)}$ the covariate vector. The random effects of frailties $\omega_i \sim \boldsymbol{\Gamma}(\frac{1}{\theta}, \frac{1}{\theta})$ and are iid.

The joint log-normal frailty model will be :

$$\left\{ \begin{array}{ll} r_{ij}(t|\eta_i) = r_0(t) \exp(\eta_i + \boldsymbol{\beta_1^{'}} \boldsymbol{Z_i(t)}) & \text{(Recurrent)} \\ \lambda_i(t|\eta_i) = \lambda_0(t) \exp(\alpha\eta_i + \boldsymbol{\beta_2^{'}} \boldsymbol{Z_i(t)}) & \text{(Death)} \end{array} \right.$$

where

$$\eta_i \sim N(0, \sigma^2)$$

- The second one (Rondeau et al. 2011) is quite similar but the frailty term is common to the individuals from a same group. This model is useful for the joint modelling two clustered survival outcomes. This joint models have been developed for clustered semi-competing events. The follow-up of each of the two competing outcomes stops when the event occurs. In this case, j is for the subject and i for the cluster.

$$\begin{cases} r_{ij}(t|u_i) = u_i r_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_{ij}(t)}) & \text{(Time to event)} \\ \lambda_{ij}(t|u_i) = u_i^{\alpha} \lambda_0(t) \exp(\boldsymbol{\beta_2'} \boldsymbol{Z_{ij}(t)}) & \text{(Death)} \end{cases}$$

It should be noted that in these models it is not recommended to include $\alpha$ parameter as there is not enough information to estimate it and thus there might be convergence problems.

In case of a log-normal distribution of the frailties, we will have :

$$\begin{cases} r_{ij}(t|v_i) = r_0(t) \exp(v_i + \boldsymbol{\beta_1'} \boldsymbol{Z_{ij}(t)}) & \text{(Time to event)} \\ \lambda_{ij}(t|v_i) = \lambda_0(t) \exp(\alpha v_i + \boldsymbol{\beta_2'} \boldsymbol{Z_{ij}(t)}) & \text{(Death)} \end{cases}$$

where

$$v_i \sim N(0, \sigma^2)$$

This joint frailty model can also be applied to clustered recurrent events and a terminal event (example on "readmission" data below).

From now on, you can also consider time-varying effects covariates in your model, see `timedep` function for more details.

There is a possibility to use a weighted penalized maximum likelihood approach for nested case-control design, in which risk set sampling is performed based on a single outcome (Jazic et al., *Submitted*).

General Joint Frailty model Fit a general joint frailty model for recurrent and terminal events considering two independent frailty terms. The frailty term $u_i$ represents the unobserved association between recurrences and death. The frailty term $v_i$ is specific to the recurrent event rate. Thus, the general joint frailty model is:

$$\begin{cases} r_{ij}(t|u_i, v_i) = u_i v_i r_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_{ij}(t)}) = u_i v_i r_{ij}(t) & \text{(Recurrent)} \\ \lambda_i(t|u_i) = u_i \lambda_0(t) \exp(\boldsymbol{\beta_1'} \boldsymbol{Z_i(t)}) = u_i \lambda_i(t) & \text{(Death)} \end{cases}$$

where the $iid$ random effects $\boldsymbol{u_i} \sim \Gamma(\frac{1}{\theta}, \frac{1}{\theta})$ and the $iid$ random effects $\boldsymbol{v_i} \sim \Gamma(\frac{1}{\eta}, \frac{1}{\eta})$ are independent from each other. The joint model is fitted using a penalized likelihood estimation on the hazard. Right-censored data and time-varying covariates $\boldsymbol{Z}_i(t)$ are allowed.

**Nested Frailty model**

***Data should be ordered according to cluster and subcluster***

Fit a nested frailty model using a Penalized Likelihood on the hazard function or using a parametric estimation. Nested frailty models allow survival studies for hierarchically clustered data by including two iid gamma random effects. Left-truncated and right-censored data are allowed. Stratification analysis is allowed (maximum of strata = 2).

The hazard function conditional on the two frailties $v_i$ and $w_{ij}$ for the $k^{th}$ individual of the $j^{th}$ subgroup of the $i^{th}$ group is :

$$\begin{cases} \lambda_{ijk}(t|v_i, w_{ij}) = v_i w_{ij} \lambda_0(t) exp(\boldsymbol{\beta'} \boldsymbol{X_{ijk}}) \\ v_i \sim \Gamma\left(\frac{1}{\alpha}, \frac{1}{\alpha}\right) \ i.i.d. \ \boldsymbol{E}(v_i) = 1 \ \ \boldsymbol{Var}(v_i) = \alpha \\ w_{ij} \sim \Gamma\left(\frac{1}{\eta}, \frac{1}{\eta}\right) \ i.i.d. \ \ \boldsymbol{E}(w_{ij}) = 1 \ \ \boldsymbol{Var}(w_{ij}) = \eta \end{cases}$$

where $\lambda_0(t)$ is the baseline hazard function, $X_{ijk}$ denotes the covariate vector and $\beta$ the corresponding vector of regression parameters.

**Joint Nested Frailty Model**

Fit a joint model for recurrent and terminal events using a penalized likelihood on the hazard functions or a parametric estimation. Right-censored data are allowed but left-truncated data and stratified analysis are not allowed.

Joint nested frailty models allow studying, jointly, survival processes of recurrent and terminal events for hierarchically clustered data, by considering the terminal event as an informative censoring and by including two iid gamma random effects.

The joint nested frailty model includes two shared frailty terms, one for the subgroup ($u_{fi}$) and one for the group ($w_f$) into the hazard functions. This random effects account the heterogeneity in the data, associated with unobserved covariates. The frailty terms act differently for the two rates ($u_{fi}$, $w_f^{\xi}$ for the recurrent rate and $u_{fi}^{\alpha}, w_i$ for the terminal event rate). The covariates could be different for the recurrent rate and death rate.

For the $j^{th}$ recurrence (j = 1, ..., $n_i$) of the $i^{th}$ individual (i = 1, ..., $m_f$) of the $f^{th}$ group (f = 1, ..., n), the joint nested gamma frailty model for recurrent event hazard function $r_{fij}$(.) and for terminal event hazard function $\lambda_{fi}$ is :

$$\begin{cases} r_{fij}(t|\omega_f, u_{fi}, \boldsymbol{X_{fij}}) = r_0(t) u_{fi} \omega_f^{\xi} \exp(\boldsymbol{\beta'} \boldsymbol{X_{fij}}) & \text{(Recurrent)} \\ \lambda_{fi}(t|\omega_f, u_{fi}, \boldsymbol{X_{fij}}) = \lambda_0(t) u_{fi}^{\alpha} \omega_f \exp(\boldsymbol{\gamma'} \boldsymbol{X_{fi}}) & \text{(Death)} \end{cases}$$

where $r_0(t)$(resp. $\lambda_0(t)$) is the recurrent (resp. terminal) event baseline hazard function, $\beta$ (resp. $\gamma$) the regression coefficient vector, $\boldsymbol{X_{fij}}(t)$ the covariates vector. The random effects are

$$\omega_f \sim \Gamma\left(\frac{1}{\eta}, \frac{1}{\eta}\right)$$

and

$$u_{fi} \sim \Gamma\left(\frac{1}{\theta}, \frac{1}{\theta}\right)$$

## Usage

```
GenfrailtyPenal(formula, formula.terminalEvent, data, recurrentAG = FALSE,
family, hazard = "Splines", n.knots, kappa, betaknots = 1, betaorder = 3,
RandDist = "Gamma", init.B, init.Theta, init.Alpha, Alpha, maxit = 300,
nb.gh, nb.gl, LIMparam = 1e-3, LIMlogl = 1e-3, LIMderiv = 1e-3, print.times = TRUE,
cross.validation, jointGeneral, nb.int, initialize, init.Ksi, Ksi, init.Eta)
```

## Arguments

`formula` A formula object, with the response on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. Interactions are possible using ' * ' or ' : '.

`formula.terminalEvent` Only for joint frailty models: a formula object, only requires terms on the right to indicate which variables are used for the terminal event. Interactions are possible using ' * ' or ' : '.

`data` A 'data.frame' with the variables used in 'formula'.

`recurrentAG` Logical value. Is Andersen-Gill model fitted? If so indicates that recurrent event times with the counting process approach of Andersen and Gill is used. This formulation can be used for dealing with time-dependent covariates. The default is FALSE.

`family` Type of Generalized Survival Model to fit. "PH" for a proportional hazards model, "AH" for an additive hazards model, "PO" for a proportional odds model and "probit" for a probit model. A vector of length 2 is expected for joint models (e.g., `family=c("PH","PH")`).

`hazard` Type of hazard functions: "Splines" for semi-parametric hazard functions using equidistant intervals, or "parametric" for parametric distribution functions. In case of `family="PH"` or `family="AH"`, the "parametric" option corresponds to a Weibull distribution. In case of `family="PO"` and `family="probit"`, the "parametric" option corresponds to a log-logistic and a log-normal distribution, respectively. So far, the "Splines" option is only available for PH and AH submodels. Default is "Splines".

`n.knots` Integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the `n.knots+2` splines functions for the approximation of the hazard or the survival functions. We estimate I- or M-splines of order 4. When the user set a number of knots equals to `k` (i.e. `n.knots=k`), then the number of interior knots is `k-2` and the number of splines is `(k-2)+order`. Number of knots must be between 4 and 20. (See Note)

`kappa` Positive smoothing parameter in the penalized likelihood estimation. The coefficient kappa tunes the intensity of the penalization (the integral of the squared second derivative of hazard function). In a stratified shared model, this argument must be a vector with kappas for both strata. In a stratified joint model, this argument must be a vector with kappas for both strata for recurrent events plus one kappa for terminal event. We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them. Value required. (See Note).

`betaknots` Number of inner knots used for the B-splines time-varying coefficient estimation. Default is 1. See 'timedep' function for more details.

`betaorder` Order of the B-splines used for the time-varying coefficient estimation. Default is cubic B-splines (`order=3`). See 'timedep' function for more details. Not implemented for Proportional Odds and Probit submodels.

`RandDist` Type of random effect distribution: "Gamma" for a gamma distribution, and "LogN" for a log-normal distribution (not implemented yet). Default is "Gamma".

`init.B` A vector of initial values for regression coefficients. This vector should be of the same size as the whole vector of covariates with the first elements for the covariates related to the recurrent events and then to the terminal event (interactions in the end of each component). Default is 0.1 for each (for Generalized Survival and Shared Frailty Models) or 0.5 (for Generalized Joint Frailty Models).

`init.Theta` Initial value for frailty variance.

`init.Alpha` Only for Generalized Joint Frailty Models: initial value for parameter alpha.

`Alpha` Only for Generalized Joint Frailty Models: input "None" so as to fit a joint model without the parameter alpha.

`maxit` Maximum number of iterations for the Marquardt algorithm. Default is 300

`nb.gh` Number of nodes for the Gaussian-Hermite quadrature. It can be chosen among 5, 7, 9, 12, 15, 20 and 32. The default is 20 if `hazard="Splines"`, 32 otherwise.

`nb.gl` Number of nodes for the Gaussian-Laguerre quadrature. It can be chosen between 20 and 32. The default is 20 if `hazard="Splines"`, 32 otherwise.

`LIMparam` Convergence threshold of the Marquardt algorithm for the parameters (see Details), 10<sup>-3</sup> by default.

`LIMlogl` Convergence threshold of the Marquardt algorithm for the log-likelihood (see Details), 10<sup>-3</sup> by default.

`LIMderiv` Convergence threshold of the Marquardt algorithm for the gradient (see Details), 10<sup>-3</sup> by default.

`print.times` A logical parameter to print iteration process. Default is TRUE.

`cross.validation`
Not implemented yet for the generalized settings.

`jointGeneral` Not implemented yet for the generalized settings.

`nb.int` Not implemented yet for the generalized settings.

`initialize` Not implemented yet for the generalized settings.

`init.Ksi` Not implemented yet for the generalized settings.

`Ksi` Not implemented yet for the generalized settings.

`init.Eta` Not implemented yet for the generalized settings.

## Details

**TYPICAL USES**

For a Generalized Survival Model:

```
GenfrailtyPenal(
formula=Surv(time,event)~var1+var2,
data, family, \dots)
```

For a Shared Frailty Generalized Survival Model:

```
GenfrailtyPenal(
formula=Surv(time,event)~cluster(group)+var1+var2,
data, family, \dots)
```

For a Joint Frailty Generalized Survival Model:

```
GenfrailtyPenal(
formula=Surv(time,event)~cluster(group)+var1+var2+var3+terminal(death),
formula.terminalEvent= ~var1+var4,
data, family, \dots)
```

**OPTIMIZATION ALGORITHM**

The estimated parameters are obtained using the robust Marquardt algorithm (Marquardt, 1963) which is a combination between a Newton-Raphson algorithm and a steepest descent algorithm. The iterations are stopped when the difference between two consecutive log-likelihoods is small ($<$ 10<sup>-3</sup>), the estimated coefficients are stable (consecutive values $<$ 10<sup>-3</sup>, and the gradient small enough ($<$ 10<sup>-3</sup>). When the frailty variance is small, numerical problems may arise. To solve this problem, an alternative formula of the penalized log-likelihood is used (see Rondeau, 2003 for further details). For Proportional Hazards and Additive Hazards submodels, cubic M-splines of order 4 can be used to estimate the hazard function. In this case, I-splines (integrated M-splines) are used to compute the cumulative hazard function.

The inverse of the Hessian matrix is the variance estimator. To deal with the positivity constraint of the variance component and the spline coefficients, a squared transformation is used and the standard errors are computed by the $\Delta$-method (Knight & Xekalaki, 2000). The integrations in the full log likelihood are evaluated using Gaussian quadrature. Laguerre polynomials with 20 points are used to treat the integrations on $[0, \infty[$.

**INITIAL VALUES**

In case of a shared frailty model, the splines and the regression coefficients are initialized to 0.1. The program fits, firstly, an adjusted Cox model to give new initial values for the splines and the regression coefficients. The variance of the frailty term $\theta$ is initialized to 0.1. Then, a shared frailty model is fitted.

In case of a joint frailty model, the splines and the regression coefficients are initialized to 0.5. The program fits firstly, an adjusted Cox model to have new initial values for the splines and the regression coefficients. The variance of the frailty term $\theta$ and the association parameter $\alpha$ are initialized to 1. Then, a joint frailty model is fitted.

## Value

The following components are included in a 'frailtyPenal' object for each model.

| | |
|---|---|
| `b` | Sequence of the corresponding estimation of the coefficients for the hazard functions (parametric or semiparametric), the random effects variances and the regression coefficients. |
| `call` | The code used for the model. |
| `formula` | The formula part of the code used for the model. |
| `n` | The number of observations used in the fit. |
| `groups` | The maximum number of groups used in the fit. |
| `n.events` | The number of events observed in the fit. |
| `n.eventsbygrp` | A vector of length the number of groups giving the number of observed events in each group. |

`loglik` The marginal log-likelihood in the parametric case.

`loglikPenal` The marginal penalized log-likelihood in the semiparametric case.

`coef` The regression coefficients.

`varH` The variance matrix of the regression coefficients before positivity constraint transformation. Then, the delta method is needed to obtain the estimated variance parameters. That is why some variances don't match with the printed values at the end of the model.

`varHtotal` The variance matrix of all the parameters before positivity constraint transformation. Then, the delta method is needed to obtain the estimated variance parameters. That is why some variances don't match with the printed values at the end of the model.

`varHIH` The robust estimation of the variance matrix of the regression coefficients

`varHIHtotal` The robust estimation of the variance matrix of all parameters.

`x` Matrix of times where the hazard functions are estimated.

`xSu` Matrix of times where the survival functions are estimated.

`lam` Array (dim=3) of baseline hazard estimates and confidence bands.

`surv` Array (dim=3) of baseline survival estimates and confidence bands.

`type` Character string specifying the type of censoring, see the `Surv` function for more details.

`n.strat` Number of strata.

`n.iter` Number of iterations needed to converge.

`median` The value of the median survival and its confidence bands. If there are two strata or more, the first value corresponds to the value for the first strata, etc.

`LCV` The approximated likelihood cross-validation criterion in the semiparametric case. With H (resp. H<sub>pen</sub>) the hessian matrix of log-likelihood (resp. penalized log-likelihood), EDF = H<sub>pen</sub><sup>-1</sup> H the effective degrees of freedom, L($\xi$,$\theta$) the log-likelihood and n the number of observations,

$$LCV = 1/nx(trace(EDF) - L(\xi, \theta)).$$

`AIC` The Akaike information Criterion for the parametric case. With p the number of parameters, n the number of observations and L($\xi$,$\theta$) the log-likelihood,

$$AIC = 1/nx(p - L(\xi, \theta)).$$

`npar` Number of parameters.

`nvar` Number of explanatory variables.

`typeof` Indicator of the type of hazard functions computed : 0 for "Splines", 2 for "parametric".

`istop` Convergence indicator: 1 if convergence is reached, 2 if convergence is not reached, 3 if the hessian matrix is not positive definite, 4 if a numerical problem has occurred in the likelihood calculation `shape.param` Shape parameter for the parametric hazard function (a Weibull distribution is used for proportional and additive hazards models, a log-logistic distribution is used for proportional odds models, a log-normal distribution is used for probit models).

`scale.param` Scale parameter for the parametric hazard function.

`Names.data` Name of the dataset.

`Frailty` Logical value. Was model with frailties fitted ?

`linear.pred` Linear predictor: $\beta$'$X$ in the generalized survival models or $\beta$'$X$ + log($u$<sub>i</sub>) in the shared frailty generalized survival models.

`BetaTpsMat` Matrix of time varying-effects and confidence bands (the first column used for abscissa of times).

`nvartimedep` Number of covariates with time-varying effects.

`Names.vardep` Name of the covariates with time-varying effects.

`EPS` Convergence criteria concerning the parameters, the likelihood and the gradient.

`family` Type of Generalized Survival Model fitted (0 for PH, 1 for PO, 2 for probit, 3 for AH).

`global_chisq.test` A binary variable equals to 0 when no multivariate Wald is given, 1 otherwise.

`beta_p.value` p-values of the Wald test for the estimated regression coefficients.

`cross.Val` Logical value. Is cross validation procedure used for estimating the smoothing parameters in the penalized likelihood estimation?

`DoF` Degrees of freedom associated with the smoothing parameter `kappa`.

`kappa` A vector with the smoothing parameters in the penalized likelihood estimation corresponding to each baseline function as components.

`n.knots` Number of knots for estimating the baseline functions in the penalized likelihood estimation.

`n.knots.temp` Initial value for the number of knots.

`global_chisq` A vector with the values of each multivariate Wald test.

`dof_chisq` A vector with the degree of freedom for each multivariate Wald test.

`p.global_chisq` A vector with the p-values for each global multivariate Wald test.

`names.factor` Names of the "as.factor" variables.

`Xlevels` Vector of the values that factor might have taken.

The following components are specific to **shared** models.

`equidistant` Indicator for the intervals used in the spline estimation of baseline hazard functions : 1 for equidistant intervals ; 0 for intervals using percentile (note: `equidistant` = 2 in case of parametric estimation).

`Names.cluster` Cluster names.

`theta` Variance of the gamma frailty parameter, i.e. Var($u$<sub>i</sub>).

`varTheta` Variance of parameter `theta`.

`theta_p.value` p-value of the Wald test for the estimated variance of the gamma frailty.

The following components are specific to **joint** models.

`formula` The formula part of the code used for the recurrent events.

`formula.terminalEvent` The formula part of the code used for the terminal model.

`n.deaths` Number of observed deaths.

`n.censored` Number of censored individuals.

`theta` Variance of the gamma frailty parameter, i.e. Var($u$<sub>i</sub>).

`indic_alpha` Indicator if a joint frailty model with $\alpha$ parameter was fitted.

`alpha` The coefficient $\alpha$ associated with the frailty parameter in the terminal hazard function.

`nvar` A vector with the number of covariates of each type of hazard function as components.

`nvarnotdep` A vector with the number of constant effect covariates of each type of hazard function as components.

`nvarRec` Number of recurrent explanatory variables.

`nvarEnd` Number of death explanatory variables.

`noVar1` Indicator of recurrent explanatory variables.

`noVar2` Indicator of death explanatory variables.

`Names.vardep` Name of the covariates with time-varying effects for the recurrent events.

`Names.vardepdc` Name of the covariates with time-varying effects for the terminal event.

`xR` Matrix of times where both survival and hazard function are estimated for the recurrent event.

`xD` Matrix of times for the terminal event.

`lamR` Array (dim=3) of hazard estimates and confidence bands for recurrent event.

`lamD` The same value as `lamR` for the terminal event.

`survR` Array (dim=3) of baseline survival estimates and confidence bands for recurrent event.

`survD` The same value as `survR` for the terminal event.

`nb.gh` Number of nodes for the Gaussian-Hermite quadrature.

`nb.gl` Number of nodes for the Gaussian-Laguerre quadrature.

`medianR` The value of the median survival for the recurrent events and its confidence bands.

`medianD` The value of the median survival for the terminal event and its confidence bands.

`names.factor` Names of the "as.factor" variables for the recurrent events.

`names.factordc` Names of the "as.factor" variables for the terminal event.

`Xlevels` Vector of the values that factor might have taken for the recurrent events.

`Xlevels2` Vector of the values that factor might have taken for the terminal event.

`linear.pred` Linear predictor for the recurrent part: $\beta$'$X$ + log($u$<sub>i</sub>).

`leardeath.pred`
Linear predictor for the terminal part: $\beta$'$X$ + $\alpha$ x log($u$<sub>i</sub>).

`Xlevels` Vector of the values that factor might have taken for the recurrent part.

`Xlevels2` vector of the values that factor might have taken for the death part.

`BetaTpsMat` Matrix of time varying-effects and confidence bands for recurrent event (the first column used for abscissa of times of recurrence).

`BetaTpsMatDc` Matrix of time varying-effects and confidence bands for terminal event (the first column used for abscissa of times of death).

`alpha_p.value` p-value of the Wald test for the estimated $\alpha$.

## Note

In the flexible semiparametric case, smoothing parameters `kappa` and number of knots `n.knots` are the arguments that the user have to change if the fitted model does not converge. `n.knots` takes integer values between 4 and 20. But with `n.knots=20`, the model would take a long time to converge. So, usually, begin first with `n.knots=7`, and increase it step by step until it converges. `kappa` only takes positive values. So, choose a value for kappa (for instance 10000), and if it does not converge, multiply or divide this value by 10 or 5 until it converges.

## References


J. Chauvet and V. Rondeau (2021). A flexible class of generalized joint frailty models for the analysis of survival endpoints. In revision.

Liu XR, Pawitan Y, Clements M. (2018) Parametric and penalized generalized survival models. *Statistical Methods in Medical Research* **27**(5), 1531-1546.

Liu XR, Pawitan Y, Clements MS. (2017) Generalized survival models for correlated time-to-event data. *Statistics in Medicine* **36**(29), 4743-4762.

A. Krol, A. Mauguen, Y. Mazroui, A. Laurent, S. Michiels and V. Rondeau (2017). Tutorial in Joint Modeling and Prediction: A Statistical Software for Correlated Longitudinal Outcomes, Recurrent Events and a Terminal Event. *Journal of Statistical Software* **81**(3), 1-52.

V. Rondeau, Y. Mazroui and J. R. Gonzalez (2012). Frailtypack: An R package for the analysis of correlated survival data with frailty models using penalized likelihood estimation or parametric estimation. *Journal of Statistical Software* **47**, 1-28.

V. Rondeau, J.P. Pignon, S. Michiels (2011). A joint model for the dependance between clustered times to tumour progression and deaths: A meta-analysis of chemotherapy in head and neck cancer. *Statistical methods in medical research* **897**, 1-19.

V. Rondeau, S. Mathoulin-Pellissier, H. Jacqmin-Gadda, V. Brouste, P. Soubeyran (2007). Joint frailty models for recurring events and death using maximum penalized likelihood estimation:application on cancer events. *Biostatistics* **8**,4, 708-721.

V. Rondeau, D. Commenges, and P. Joly (2003). Maximum penalized likelihood estimation in a gamma-frailty model. *Lifetime Data Analysis* **9**, 139-153.

C.A. McGilchrist, and C.W. Aisbett (1991). Regression with frailty in survival analysis. *Biometrics* **47**, 461-466.

D. Marquardt (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematics*, 431-441.

## See Also

Surv, terminal, timedep

## Examples

```
###############################################################################
# -----       GENERALIZED SURVIVAL MODELS (without frailties)       -----  #
###############################################################################

adult.retino = retinopathy[retinopathy$type == "adult", ]
adult.retino[adult.retino$futime >= 50, "status"] = 0
adult.retino[adult.retino$futime >= 50, "futime"] = 50

### ---  Parametric PH, AH, PO and probit models  --- ###

GenfrailtyPenal(formula=Surv(futime,status)~trt, data=adult.retino,
hazard="parametric", family="PH")
GenfrailtyPenal(formula=Surv(futime,status)~trt, data=adult.retino,
hazard="parametric", family="AH")
GenfrailtyPenal(formula=Surv(futime,status)~trt, data=adult.retino,
hazard="parametric", family="PO")
GenfrailtyPenal(formula=Surv(futime,status)~trt, data=adult.retino,
hazard="parametric", family="probit")

### ---  Semi-parametric PH and AH models  --- ###

GenfrailtyPenal(formula=Surv(futime,status)~timedep(trt), data=adult.retino,
family="PH", hazard="Splines", n.knots=8, kappa=10^6, betaknots=1, betaorder=2)
GenfrailtyPenal(formula=Surv(futime,status)~timedep(trt), data=adult.retino,
family="AH", hazard="Splines", n.knots=8, kappa=10^10, betaknots=1, betaorder=2)



###############################################################################
# -----         SHARED FRAILTY GENERALIZED SURVIVAL MODELS          -----  #
###############################################################################

adult.retino = retinopathy[retinopathy$type == "adult", ]
adult.retino[adult.retino$futime >= 50, "status"] = 0
adult.retino[adult.retino$futime >= 50, "futime"] = 50

### ---  Parametric PH, AH, PO and probit models  --- ###

GenfrailtyPenal(formula=Surv(futime,status)~trt+cluster(id), data=adult.retino,
hazard="parametric", family="PH")
GenfrailtyPenal(formula=Surv(futime,status)~trt+cluster(id), data=adult.retino,
hazard="parametric", family="AH")
GenfrailtyPenal(formula=Surv(futime,status)~trt+cluster(id), data=adult.retino,
hazard="parametric", family="PO")
GenfrailtyPenal(formula=Surv(futime,status)~trt+cluster(id), data=adult.retino,
hazard="parametric", family="probit")
```

```
### ---  Semi-parametric PH and AH models  --- ###

GenfrailtyPenal(formula=Surv(futime,status)~cluster(id)+timedep(trt),
data=adult.retino, family="PH", hazard="Splines",
n.knots=8, kappa=10^6, betaknots=1, betaorder=2)
GenfrailtyPenal(formula=Surv(futime,status)~cluster(id)+timedep(trt),
data=adult.retino, family="AH", hazard="Splines",
n.knots=8, kappa=10^10, betaknots=1, betaorder=2)



##############################################################################
# -----         JOINT FRAILTY GENERALIZED SURVIVAL MODELS            -----  #
##############################################################################

data("readmission")
readmission[, 3:5] = readmission[, 3:5]/365.25

### ---  Parametric dual-PH, AH, PO and probit models  --- ###

GenfrailtyPenal(
formula=Surv(t.start,t.stop,event)~cluster(id)+terminal(death)+sex+dukes+chemo,
formula.terminalEvent=~sex+dukes+chemo, data=readmission, recurrentAG=TRUE,
hazard="parametric", family=c("PH","PH"))
GenfrailtyPenal(
formula=Surv(t.start,t.stop,event)~cluster(id)+terminal(death)+sex+dukes+chemo,
formula.terminalEvent=~sex+dukes+chemo, data=readmission, recurrentAG=TRUE,
hazard="parametric", family=c("AH","AH"))
GenfrailtyPenal(
formula=Surv(t.start,t.stop,event)~cluster(id)+terminal(death)+sex+dukes+chemo,
formula.terminalEvent=~sex+dukes+chemo, data=readmission, recurrentAG=TRUE,
hazard="parametric", family=c("PO","PO"))
GenfrailtyPenal(
formula=Surv(t.start,t.stop,event)~cluster(id)+terminal(death)+sex+dukes+chemo,
formula.terminalEvent=~sex+dukes+chemo, data=readmission, recurrentAG=TRUE,
hazard="parametric", family=c("probit","probit"))

### ---  Semi-parametric dual-PH and AH models  --- ###

GenfrailtyPenal(
formula=Surv(t.start,t.stop,event)~cluster(id)+terminal(death)+sex+dukes+timedep(chemo),
formula.terminalEvent=~sex+dukes+timedep(chemo), data=readmission, recurrentAG=TRUE,
hazard="Splines", family=c("PH","PH"),
n.knots=5, kappa=c(100,100), betaknots=1, betaorder=3)
GenfrailtyPenal(
formula=Surv(t.start,t.stop,event)~cluster(id)+terminal(death)+sex+dukes+timedep(chemo),
formula.terminalEvent=~sex+dukes+timedep(chemo), data=readmission, recurrentAG=TRUE,
hazard="Splines", family=c("AH","AH"),
n.knots=5, kappa=c(600,600), betaknots=1, betaorder=3)
```

| hazard | *Hazard function.* |
|---|---|

## Description

Let t be a continuous variable, we determine the value of the hazard function to t after run fit.

## Usage

```
hazard(t, ObjFrailty)
```

## Arguments

| | |
|---|---|
| `t` | time for hazard function. |
| `ObjFrailty` | an object from the frailtypack fit. |

## Value

return the value of hazard function in t.

## Examples

```
## Not run:

#-- a fit Shared
data(readmission)
fit.shared <- frailtyPenal(Surv(time,event)~dukes+cluster(id)+
strata(sex),n.knots=10,kappa=c(10000,10000),data=readmission)

#-- calling survival
hazard(20,fit.shared)


## End(Not run)
```

jointRecCompet *Competing Joint Frailty Model: A single type of recurrent event and two terminal events.*

## Description

Fit a joint competing frailty model for a single recurrent event and two terminal events defined as,

$$\begin{aligned}\text{Recurrent event:}\quad & r_{ij}\left(t \mid w_i, \boldsymbol{X}_{r,ij}\right) = r_0(t) \exp\left(\boldsymbol{X}_{r,ij}\boldsymbol{\beta}_r + w_i\right)\\ \text{First terminal event:}\quad & \lambda_{1,i}\left(t \mid w_i, \boldsymbol{X}_{1i}\right) = \lambda_{1,0}(t) \exp\left(\boldsymbol{X}_{1,i}\boldsymbol{\beta}_1 + \alpha_1 w_i\right)\\ \text{Second terminal event:}\quad & \lambda_{2,i}\left(t \mid w_i, \boldsymbol{X}_{2,i}\right) = \lambda_{2,0}(t) \exp\left(\boldsymbol{X}_{2,i}\boldsymbol{\beta}_2 + \alpha_2 w_i\right).\end{aligned}$$

where $\omega_i \sim \mathcal{N}(0, \theta)$ is the frailty term and $\boldsymbol{X}_{r,ij}, \boldsymbol{X}_{1,i}$ and $\boldsymbol{X}_{2,i}$ are vectors of baseline covariates (possibly the same). The parameters $\alpha_1$ and $\alpha_2$ are power parameters.

## Usage

```
jointRecCompet(formula,
          formula.terminalEvent = NULL,
          formula.terminalEvent2 = NULL,
          data,
          initialize = TRUE,
          recurrentAG = FALSE,
          maxit = 350,
          hazard = "Weibull",
          n.knots=7,
          kappa = rep(10, 3),
          crossVal=FALSE,
          constraint.frailty = "squared",
          GHpoints = 32,
          tolerance = rep(10^-3, 3),
          init.hazard = NULL,
          init.Sigma = 0.5,
          init.Alpha1 = 0.1,
          init.Alpha2 = -0.1,
          init.B = NULL)
```

## Arguments

`formula` a formula object, with the response for the first recurrent event on the left of a $\sim$ operator, and the terms on the right. The response must be in the format Surv(t0, t1, recurrentevent) for a calendar-time specification where `t0` is the start time for an at-risk period for the recurrent event, `t1` is the end time for an at-risk period for the recurrent event, and `recurrentevent` is a numeric indicator for whether an event was observed (1) or was censored (2). In a gap-time setting, an object of the format Surv(t, recurrentevent) should be used instead. Note that to not be confused with a left-truncation setting, when using a calendar-time specification argument `recurrentAG` should be set to `TRUE`.

`formula.terminalEvent`
: a formula object, empty on the left of a $\sim$ operator, and the terms on the right. Leave the formula at the default value (NULL) for a model with no variables.

`formula.terminalEvent2`
: a formula object, empty on the left of a $\sim$ operator, and the terms on the right.Leave the formula at the default value (NULL) for a model with no variables.

`data`
: a 'data.frame' with the variables used in 'formula', 'formula.terminalEvent', and 'formula.terminalEvent2'.

`initialize`
: Logical value to internally initialize regression coefficients and baseline hazard functions parameters using simpler models from frailtypack. When initialization is requested, the program first fits two joint frailty models for the recurrent events and each terminal event. When FALSE, parameters are initialized via the arguments init.hazard, init.Sigma, init.Alpha1, init.Alpha2, init.B.

`recurrentAG`
: Logical value. Is Andersen-Gill model fitted? If so indicates that recurrent event times with the counting process approach of Andersen and Gill is used. This formulation can be used for dealing with time-dependent covariates. The default is FALSE.

`maxit`
: maximum number of iterations for the Marquardt algorithm. Default is 350.

`hazard`
: Type of hazard functions. Available options are `"Weibull"` for parametric Weibull function, `"Splines"` for semiparametric hazard functions using equidistant intervals or `"Splines-per"` for percentile intervals. Default is `"Weibull"`.

`n.knots`
: In the case of splines hazard functions, number of knots to be used in the splines basis. This number should be between 4 and 20. Default is 7.

`kappa`
: In the case of splines hazard functions, a vector of size 3 containing the values of the smoothing parameters to be used for each baseline hazard function. Default value is 10 for each function.

`crossVal`
: In the case of splines hazard functions, indicates how the smoothing parameters are chosen. If set to "TRUE" then those parameters are chosen automatically using cross-validation on reduced models for each baseline hazard function. If set to "FALSE" then the parameters are those provided by the argument `kappa`. If set to "TRUE" then the argument `kappa` is ignored. Default is "TRUE".

`constraint.frailty`
: Type of positivity constraint used for the variance of of the random effect in the likelihood. Possible values are 'squared' or 'exponential'. Default is 'squared'. See Details.

`GHpoints`
: Integer. Number of nodes for Gauss-Hermite integration to marginalize random effects/frailties. Default is 32.

`tolerance`
: Numeric, length 3. Optimizer's tolerance for (1) successive change in parameter values, (2) log likelihood, and (3) score, respectively.

`init.hazard`
: Numeric. Initialization values for hazard parameters. If a weibull model is used, the order is: shapeR, scaleR, shapeTerminal1, scaleTerminal1, shapeTerminal2, scaleTerminal2.

`init.Sigma` Numeric,. Initialization value for the standard deviation of the normally-distributed random effects.

`init.Alpha1` Numeric. Initialization value for the parameter alpha that links the hazard function of the recurrent event to the first terminal event.

`init.Alpha2` Numeric. Initialization value for the parameter alpha that links the hazard function of the recurrent event to the second terminal event.

`init.B` Numeric vector of the same length and order as the three covariate vectors for the recurrent, terminal1, and terminal2 events (in that order).

## Details

Right-censored data are allowed. Left-truncated data and stratified analysis are not possible. Prediction options are not yet available. The `constraint.frailty` argument defines the positivity constraint used for the frailty variance in the likelihood. By default it uses the square so that the absolute value of the parameter is the standard deviation of the frailty (i.e $\theta^2 = \beta^2$). The other parametrization uses the square of the exponential for the variance so that the parameter is the logarithm of the standard deviation ($\theta^2 = (\exp \beta)^2$). For others parameters in the model needing a positivity constraint (parameters related to the baseline hazard functions), the parametrization used is the exponential squared.

## Value

Parameters estimates of a competing joint frailty model, more generally a 'jointRecCompet' object. Methods defined for 'jointRecCompet' objects are provided for print, plot and summary. The following components are included in a 'jointRecCompet' object.

`summary.table` A table describing the estimate, standard error, confidence interval, and pvalues for each of the parameters in the model.

`controls` A vector of named control parameters

`k0` For splines baseline hazard functions, vector of penalization terms.

`noVarEvent` A vector containing for each event type if there is no covariate used in the model.

`np` Total number of parameters

`b` Vector containing the estimated coefficients of the model before any positivity constraint. The values are in order: the coefficients associated with the baseline hazard functions (either the splines or the shape and scale parameters for Weibull hazard), the random effect variance, the coefficients of the frailty ($\alpha_1$ and $\alpha_2$) and the regression coefficients.

`H_hessOut` Covariance matrix of the estimated parameters

`HIHOut` Covariance matrix of the estimated parameters for the penalized likelihood in the case of Splines baseline hazard functions.

`LCV` The approximated likelihood cross-validation criterion in the spline case

`critCV` Convergence criteria

`x1` Vector of times for which the hazard function of the recurrent event is estimated. By default seq(0,max(time),length=99), where time is the vector of survival times.

| | |
|---|---|
| lam1 | Matrix of hazard estimates and confidence bands for the recurrent event. |
| xSu1 | Vector of times for the survival function of the recurrent event. |
| surv1 | Matrix of baseline survival estimates and confidence bands for recurrent event. |
| x2 | Vector of times for the first terminal event (see x1 value). |
| lam2 | Matrix of hazard estimates and confidence bands for the first terminal event. |
| xSu2 | Vector of times for the survival function of the first terminal event. |
| surv2 | Vector of the survival function of the first terminal event evaluated at xSu2. |
| x3 | Vector of times for the second terminal event (see x1 value). |
| lam3 | Matrix of hazard estimates and confidence bands for the second terminal event. |
| xSu3 | Vector of times for the survival function of the second terminal event. |
| surv3 | Vector of the survival function of the second terminal event evaluated at xSu3. |
| ni | Number of iterations needed to converge. |
| constraintfrailty | Positivity constraint used for the variance of the random effect |
| ziOut1 | In the spline case, vector of knots used in the spline basis for the recurrent event |
| ziOutdc | In the spline case, vector of knots used in the spline basis for the terminal events |
| ghnodes | Nodes used for the Gauss-Hermite quadrature. |
| ghweights | Weights used for the Gauss-Hermite quadrature. |
| tolerance | Numeric, length 3. Optimizer's tolerance for (1) successive change in parameter values, (2) log likelihood, and (3) score, respectively. |
| call | Call of the function. |
| loglikPenal | Estimated penalized log-likelihood in the spline case |
| logLik | Estimated log-likelihood in the Weibull case |
| AIC | For the Weibull case, Akaike Information criterion |
| n | Total number of subjects |
| nevts | Number of events for each event type. |

## See Also

terminal

## Examples

```
set.seed(1)
data=simulatejointRecCompet(n=500,
par0=c(shapeR = 1.5, scaleR = 10,
shapeM = 1.75, scaleM = 16, shapeD = 1.75, scaleD = 16, sigma = 0.5,
alphaM = 1, alphaD = 1, betaR = -0.5, betaM = -0.5, betaD = 0) )
mod <-jointRecCompet(formula = Surv(tstart, tstop, event)~cluster(id)+treatment+
                    terminal(terminal1)+terminal2(terminal2),
                    formula.terminalEvent = ~treatment,
                    formula.terminalEvent2 = ~treatment,
                    data = data,
```

```
                    recurrentAG = TRUE,
                    initialize = TRUE,
   n.knots=7,
   crossVal=TRUE,
                    hazard = "Splines",
                    maxit = 350)

#This example uses an extract of 500 patients of the REDUCE trial
data(reduce)
mod_reduce <-jointRecCompet(formula = Surv(t.start,t.stop, del)~cluster(id)+
   treatment+terminal(death)+terminal2(discharge),
                    formula.terminalEvent = ~treatment,
                    formula.terminalEvent2 = ~treatment,
                    data = reduce,
                    initialize = TRUE,
                    recurrentAG = TRUE,
                    hazard = "Weibull",
                    constraint.frailty = "exponential",
                    maxit = 350)
print(mod_reduce)
```

---

`jointSurrCopSimul` *Generate survival times for two endpoints using the joint frailty-copula model for surrogacy*

---

## Description

Date are generated from the one-step joint frailty-copula model, under the Claton copula function (see `jointSurroCopPenal` for more details)

## Usage

```
jointSurrCopSimul(
  n.obs = 600,
  n.trial = 30,
  prop.cens = 0,
  cens.adm = 549,
  alpha = 1.5,
  gamma = 2.5,
  sigma.s = 0.7,
  sigma.t = 0.7,
  cor = 0.9,
  betas = c(-1.25, 0.5),
  betat = c(-1.25, 0.5),
  frailt.base = 1,
  lambda.S = 1.3,
  nu.S = 0.0025,
  lambda.T = 1.1,
```

```
  nu.T = 0.0025,
  ver = 2,
  typeOf = 1,
  equi.subj.trial = 1,
  equi.subj.trt = 1,
  prop.subj.trial = NULL,
  prop.subj.trt = NULL,
  full.data = 0,
  random.generator = 1,
  random = 0,
  random.nb.sim = 0,
  seed = 0,
  nb.reject.data = 0,
  thetacopule = 6,
  filter.surr = c(1, 1),
  filter.true = c(1, 1),
  covar.names = "trt",
  pfs = 0
)
```

## Arguments

| | |
|---|---|
| n.obs | Number of considered subjects. The default is 600. |
| n.trial | Number of considered trials. The default is 30. |
| prop.cens | A value between 0 and 1, 1-prop.cens is the minimum proportion of people who are randomly censored. Represents the quantile to use for generating the random censorship time. In this case, the censorship time follows a uniform distribution in 1 and (prop.cens)ieme percentile of the generated death times. If this argument is set to 0, the fix censorship is considered. The default is 0. |
| cens.adm | Censorship time. If argument prop.cens is set to 0, it represents the administrative censorship time, else it represents the fix censoring time. The default is 549, for about 40% of fix censored subjects. |
| alpha | Fixed value for $\alpha$. The default is 1.5. |
| gamma | Fixed value for $\gamma$. The default is 2.5. |
| sigma.s | Fixed value for $\sigma^2_{v_S}$ . The default is 0.7. |
| sigma.t | Fixed value for $\sigma^2_{v_T}$ . The default is 0.7. |
| cor | Desired level of correlation between $v_{S_i}$ and $v_{T_i}$. $R^2_{trial} = cor^2$ . The default is 0.8. |
| betas | Vector of the fixed effects for $\beta_S$ . The size must be equal to ver The default is c(-1.25,0.5). |
| betat | Vector of the fixed effects for $\beta_T$ . The size must be equal to ver The default is c(-1.25,0.5). |
| frailt.base | Considered heterogeneity on the baseline risk (1) or not (0). The default is 1. |
| lambda.S | Desired scale parameter for the Weibull distribution associated with the Surrogate endpoint. The default is 1.8. |

nu.S Desired shape parameter for the Weibull distribution associated with the Surrogate endpoint. The default is 0.0045.

lambda.T Desired scale parameter for the Weibull distribution associated with the True endpoint. The default is 3.

nu.T Desired shape parameter for the Weibull distribution associated with the True endpoint. The default is 0.0025.

ver Number of covariates. The mandatory covariate is the treatment arm. The default is 2.

typeOf Type of joint model used for data generation: 0 = classical joint model with a shared individual frailty effect (Rondeau, 2007), 1 = joint frailty-copula model with shared frailty effects $u_i$ and two correlated random effects treatment-by-trial interaction ($v_{S_i}$, $v_{T_i}$), see jointSurroCopPenal.

equi.subj.trial A binary variable that indicates if the same proportion of subjects should be included per trial (1) or not (0). If 0, the proportions of subject per trial are required with parameter prop.subj.trial.

equi.subj.trt A binary variable that indicates if the same proportion of subjects is randomized per trial (1) or not (0). If 0, the proportions of subject per trial are required with parameter prop.subj.trt.

prop.subj.trial The proportions of subjects per trial. Requires if equi.subj.trial = 0.

prop.subj.trt The proportions of randomized subject per trial. Requires if equi.subj.trt = 0.

full.data Specified if you want the function to return the full dataset (1), including the random effects, or the restictive dataset (0) with at least 7 columns as required for the function jointSurroCopPenal.

random.generator The random number generator used by the Fortran compiler, 1 for the intrinsec subroutine Random_number and 2 for the subroutine uniran(). The default is 1.

random A binary that says if we reset the random number generation with a different environment at each call (1) or not (0). If it is set to 1, we use the computer clock as seed. In the last case, it is not possible to reproduce the generated datasets. The default is 0. Required if random.generator is set to 1.

random.nb.sim required if random.generator is set to 1, and if random is set to 1.

seed The seed to use for data (or samples) generation. Required if the argument random.generator is set to 1. Must be a positive value. If negative, the program do not account for seed. The default is 0.

nb.reject.data Number of generation to reject before the considered dataset. This parameter is required when data generation is for simulation. With a fixed parameter and random.generator set to 1, all ganerated data are the same. By varying this parameter, different datasets are obtained during data generations. The default value is 0, in the event of one dataset.

thetacopule The desired value for the copula parameter. The default is 6.

`filter.surr` Vector of size the number of covariates, with the i-th element that indicates if the hazard for surrogate is adjusted on the i-th covariate (code 1) or not (code 0). By default, 2 covariates are considered.

`filter.true` Vector defines as `filter.surr`, for the true endpoint. `filter.true` and `filter.surr` should have the same size

`covar.names` Vector of the names of covariables. By default it contains "trt" for the tratment arm. Should contains the names of all covarites wished in the generated dataset.

`pfs` Is used to specify if the time to progression should be censored by the death time (0) or not (1). The default is 0. In the event with pfs set to 1, death is included in the surrogate endpoint as in the definition of PFS or DFS.

## Details

We just considered in this generation, the Gaussian random effects. If the parameter `full.data` is set to 1, this function return a list containning severals parameters, including the generated random effects. The desired individual level correlation (Kendall's $\tau$) depend on the values of the copula parameter $\theta$, given that $\tau = \theta/(\theta + 2)$ under the clayton copula model.

## Value

This function returns if the parameter `full.data` is set to 0, a `data.frame` with columns :

`patientID` A numeric, that represents the patient's identifier, must be unique;

`trialID` A numeric, that represents the trial in which each patient was randomized;

`trt` The treatment indicator for each patient, with 1 = treated, 0 = untreated;

`timeS` The follow up time associated with the surrogate endpoint;

`statusS` The event indicator associated with the surrogate endpoint. Normally 0 = no event, 1 = event;

`timeT` The follow up time associated with the true endpoint;

`statusT` The event indicator associated with the true endpoint. Normally 0 = no event, 1 = event;

and other covariates named `Var2, var3, ..., var[ver-1]` if `ver > 1`. If the argument `full.data` is set to 1, additionnal colums corresponding to random effects $u_i$ , $v_{S_i}$ and $v_{T_i}$ are returned.

## Author(s)

Casimir Ledoux Sofeu `<casimir.sofeu@u-bordeaux.fr>`, `<scl.ledoux@gmail.com>` and Virginie Rondeau `<virginie.rondeau@inserm.fr>`

## References

Rondeau V., Mathoulin-Pelissier S., Jacqmin-Gadda H., Brouste V. and Soubeyran P. (2007). Joint frailty models for recurring events and death using maximum penalized likelihood estimation: application on cancer events. Biostatistics 8(4), 708-721.

Sofeu, C. L., Emura, T., and Rondeau, V. (2020). A joint frailty-copula model for meta-analytic validation of failure time surrogate endpoints in clinical trials. `Under review`

## See Also

`jointSurrSimul`, `jointSurroCopPenal`

## Examples

```
# dataset with 2 covariates and fixed censorship
data.sim <- jointSurrCopSimul(n.obs=600, n.trial = 30, prop.cens = 0, cens.adm=549,
            alpha = 1.5, gamma = 2.5, sigma.s = 0.7, sigma.t = 0.7,
            cor = 0.8, betas = c(-1.25, 0.5), betat = c(-1.25, 0.5),
            full.data = 0, random.generator = 1,ver = 2, covar.names = "trt",
            nb.reject.data = 0, thetacopule = 6, filter.surr = c(1,1),
            filter.true = c(1,1), seed = 0)

#dataset with 2 covariates and random censorship

data.sim2 <- jointSurrCopSimul(n.obs=600, n.trial = 30, prop.cens = 0.75,
            cens.adm = 549, alpha = 1.5, gamma = 2.5, sigma.s = 0.7,
            sigma.t = 0.7, cor = 0.8, betas = c(-1.25, 0.5),
            betat = c(-1.25, 0.5), full.data = 0, random.generator = 1,
            ver = 2, covar.names = "trt", nb.reject.data = 0, thetacopule = 6,
            filter.surr = c(1,1), filter.true = c(1,1), seed = 0)
```

---

`jointSurroCopPenal` *Fit the one-step Joint frailty-copula model for evaluating a canditate surrogate endpoint*

---

## Description

**Joint Frailty-Copula model for Surrogacy definition**

Fit the one-step Joint surrogate model for the evaluation of a canditate surrogate endpoint, with different integration methods on the random effects, using a semiparametric penalized likelihood estimation. This approach extends that of Burzykowski `et al.` (2001) by including in the bivariate copula model the random effects treatment-by-trial interaction.

Assume $S_{ij}$ and $T_{ij}$ the failure times associated respectively with the surrogate and the true endpoints, for subject `j` ($j = 1, \ldots, n_i$) belonging to the trial $i (i = 1, \ldots, G)$.

Let $\mathbf{v}_i = (u_i, v_{Si}, v_{Ti})$ be the vector of trial level random effects; $\mathbf{Z}_{S,ij} = (Z_{Sij1}, \cdots, Z_{Sijp})'$ and $\mathbf{Z}_{T,ij} = (Z_{Tij1}, \cdots, Z_{Tijp})'$ be covariates associated with $S_{ij}$ and $T_{ij}$ The joint frailty-copula model is defined as follows:

$$\begin{aligned}\bar{F}(s_{ij}, t_{ij}|\mathbf{Z}_{S,ij}, \mathbf{Z}_{T,ij}, \mathbf{v}_i) &= P(S_{ij} > s_{ij}, T_{ij} > t_{ij}|\mathbf{Z}_{S,ij}, \mathbf{Z}_{T,ij}, \mathbf{v}_i) \\ &= \varphi_\theta[\varphi_\theta^{-1}(\bar{F}(s_{ij}|\mathbf{Z}_{S,ij}, u_i, v_{Si})) + \varphi_\theta^{-1}(\bar{F}(t_{ij}|\mathbf{Z}_{T,ij}, u_i, v_{Ti}))]\end{aligned}$$

where,

$\varphi_\theta : [0,\infty) \to [0,1]$ the generator of a parametric Archimedean copula family

and the conditional survival functions are given by

$$\bar{F}_{S,ij}(s_{ij}|\mathbf{Z}_{S,ij},u_i,v_{Si}) = \exp\left\{ -\int_0^{s_{ij}} \lambda_{0S}(x)\exp\left(u_i + v_{Si}Z_{ij1} + \boldsymbol{\beta}_S\mathbf{Z}_{S,ij}\right)dx\right\}$$

$$\bar{F}_{T,ij}(t_{ij}|\mathbf{Z}_{T,ij},u_i,v_{Ti}) = \exp\left\{ -\int_0^{t_{ij}} \lambda_{0T}(x)\exp\left(\alpha u_i + v_{Ti}Z_{ij1} + \boldsymbol{\beta}_T\mathbf{Z}_{T,ij}\right)dx\right\}$$

in which

$$u_i \sim N(0,\gamma), u_i \perp v_{S_i}, u_i \perp v_{T_i}; (v_{S_i}, v_{T_i})^T \sim \mathcal{N}\left(0,\Sigma_v\right)$$

with

$$\Sigma_v = \begin{pmatrix} \sigma^2_{v_S} & \sigma_{v_{ST}} \\ \sigma_{v_{ST}} & \sigma^2_{v_T} \end{pmatrix}$$

In this model, $\lambda_{0S}(t)$ is the baseline hazard function associated with the surrogate endpoint and $\beta_S$ the fixed effects (or log-hazard ratio) corresponding to the covariates $\mathbf{Z}_{S,ij}$; $\lambda_{0T}(t)$ is the baseline hazard function associated with the true endpoint and $\beta_T$ the fixed effects corresponding to the covariates $\mathbf{Z}_{T,ij}$. The copula model serves to consider dependence between the surrogate and true endpoints at the individual level. In the copula model, $\theta$ is the copula parameter used to quantify the strength of association. $u_i$ is a shared frailty effect associated with the baseline hazard function that serve to take into account the heterogeneity between trials of the baseline hazard function, associated with the fact that we have several trials in this meta-analytical design. The power parameter $\alpha$ distinguishes trial-level heterogeneity between the surrogate and the true endpoint. $v_{S_i}$ and $v_{T_i}$ are two correlated random effects treatment-by-trial interactions. $Z_{Sij1}$ or $Z_{Tij1}$ represents the treatment arm to which the patient has been randomized.

For simplicity, we focus on the Clayton and Gumbel-Hougaard copula functions. In Clayton's model, the copula function has the form

$$\varphi_\theta(s) = (1+\theta s)^{-1/\theta}, \quad \theta > 0$$

and in Gumbel's model, the copula function has the form

$$\varphi_\theta(s) = \exp\left[-s^{1/(1+\theta)}\right], \quad \theta \geq 0$$

**Surrogacy evaluation**

We proposed to base validation of a candidate surrogate endpoint on Kendall's $\tau$ at the individual level and coefficient of determination at the trial level, as in the classical approach (Burzykowski `et al.`, 2001). The formulations are given below.

**Individual-level surrogacy**

From the proposed model, according to the copula function, it can be shown that Kendall's $\tau$ is defined as:

$\tau = \frac{\theta}{\theta+2}$ for Clayton copula and $\tau = \frac{\theta}{\theta+1}$ for Gumbel copula.

where $\theta$ is the copula parameter. Kendall's $\tau$ is the difference between the probability of concordance and the probability of discordance of two realizations of $S_{ij}$ and $T_{ij}$. It belongs to the interval [-1,1] and assumes a zero value when $S_{ij}$ and $T_{ij}$ are independent.

**Trial-level surrogacy**

The key motivation for validating a surrogate endpoint is to be able to predict the effect of treatment on the true endpoint, based on the observed effect of treatment on the surrogate endpoint. As shown by Buyse *et al.* (2000), the coefficenient of determination obtains from the covariance matrix $\Sigma_v$ of the random effects treatment-by-trial interaction can be used to evaluate underlined prediction, and therefore as surrogacy evaluation measurement at trial-level. It is defined by:

$$R^2_{trial} = \frac{\sigma^2_{v_{ST}}}{\sigma^2_{v_S}\sigma^2_{v_T}}$$

The SEs of $R^2_{trial}$ and $\tau$ are calculated using the Delta-method. We also propose $R^2_{trial}$ and 95% CI computed using the parametric bootstrap. The use of delta-method can lead to confidence limits violating the [0,1], as noted by (Burzykowski *et al.*, 2001). However, using other methods would not significantly alter the findings of the surrogacy assessment

## Usage

```
jointSurroCopPenal(data, maxit = 40, indicator.alpha = 1,
   frail.base = 1, n.knots = 6, LIMparam = 0.001, LIMlogl = 0.001,
   LIMderiv = 0.001, nb.mc = 1000, nb.gh = 20, nb.gh2 = 32,
   adaptatif = 0, int.method = 0, nb.iterPGH = 5, true.init.val = 0,
   thetacopula.init = 1, sigma.ss.init = 0.5, sigma.tt.init = 0.5,
   sigma.st.init = 0.48, gamma.init = 0.5, alpha.init = 1,
   betas.init = 0.5, betat.init = 0.5, scale = 1,
   random.generator = 1, kappa.use = 4, random = 0,
   random.nb.sim = 0, seed = 0, init.kappa = NULL, ckappa = c(0,0),
   typecopula = 1, nb.decimal = 4, print.times = TRUE, print.iter = FALSE)
```

## Arguments

`data` A `data.frame` containing at least seven variables entitled:

- `patientID`: A numeric, that represents the patient's identifier and must be unique;
- `trialID`: A numeric, that represents the trial in which each patient was randomized;
- `timeS`: The follow-up time associated with the surrogate endpoint;
- `statusS`: The event indicator associated with the surrogate endpoint. Normally 0 = no event, 1 = event;
- `timeT`: The follow-up time associated with the true endpoint;
- `statusT`: The event indicator associated with the true endpoint. Normally 0 = no event, 1 = event;
- `trt`: The treatment indicator for each patient, with 1 = treated, 0 = untreated.

maxit
: maximum number of iterations for the Marquardt algorithm. The default being `40`.

indicator.alpha
: A binary, indicating whether the power's parameter $\alpha$ should be estimated (1) or not (0). If `0`, $\alpha$ will be set to `1` during estimation. The default is 1.

frail.base
: A binary, indicating whether the heterogeneity between trial on the baseline risk is considered (`1`) or not (`0`), using the shared cluster specific frailties ($u_i$). The default is `1`.

n.knots
: integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See `frailtyPenal` for more details).

LIMparam
: Convergence threshold of the Marquardt algorithm for the parameters, $10^{-3}$ by default (See `frailtyPenal` for more details).

LIMlogl
: Convergence threshold of the Marquardt algorithm for the log-likelihood, $10^{-3}$ by default (See `frailtyPenal` for more details).

LIMderiv
: Convergence threshold of the Marquardt algorithm for the gradient, $10^{-3}$ by default (See `frailtyPenal` for more details).

nb.mc
: Number of samples considered in the Monte-Carlo integration. Required in the event `int.method` is equals to `0`, `2` or `4`. A value between 500 and 1000 most often gives good results. The default is `1000`.

nb.gh
: Number of nodes for the Gaussian-Hermite quadrature. It can be chosen among 5, 7, 9, 12, 15, 20 and 32. The default is `20`. A value greater than or equals to `15` allowed good results in simulation studies

nb.gh2
: Number of nodes for the Gauss-Hermite quadrature used to re-estimate the model, in the event of non-convergence, defined as previously. The default is `32`.

adaptatif
: A binary, indicates whether the pseudo adaptive Gaussian-Hermite quadrature `(1)` or the classical Gaussian-Hermite quadrature `(0)` is used. The default is `0`.

int.method
: A numeric, indicates the integration method: `0` for Monte carlo, `1` for Gaussian-Hermite quadrature, `3` for Laplace approximation. The default is `0`.

nb.iterPGH
: Number of iterations before the re-estimation of the posterior random effects, in the event of the two-steps pseudo-adaptive Gaussian-hermite quadrature. If set to `0` there is no re-estimation". The default is `5`.

true.init.val
: Numerical value. Indicates if the given initial values to parameters `(0)` should be considered. If set to `2`, $\alpha$ and $\gamma$ are initialised using two separed shared frailty model (see `frailtyPenal` for more details);

  $\sigma^2_{v_S}$, $\sigma^2_{v_T}$ and $\sigma_{v_{ST}}$ are fixed by the user or the default values; $\theta$, $\beta_S$ and $\beta_T$ are initialized using a classical joint frailty model, considering individual level random effects, with $\theta$ the variance of individual level random effects. If the joint frailty model is faced to convergence issues, $\beta_S$ and $\beta_T$ are initialized using two shared frailty models. In all other scenarios, if the simplified model does not converge, default given parameters values are used. Initial values for spline's associated parameters are fixed to 0.5. The default for this argument is 0.

thetacopula.init
: Initial values for the copula parameter ($\theta$), required if true.init.val is set to 0 or 2. The default is 1.

sigma.ss.init
: Initial values for $\sigma^2_{v_S}$ , required if true.init.val is set to 0 or 2. The default is 0.5.

sigma.tt.init
: Initial values for $\sigma^2_{v_T}$ , required if true.init.val is set to 0 or 2. The default is 0.5.

sigma.st.init
: Initial values for $\sigma_{v_{ST}}$ , required if true.init.val is set to 0 or 2. The default is 0.48.

gamma.init
: Initial values for $\gamma$, required if true.init.val is set to 0 or 2. The default is 0.5.

alpha.init
: Initial values for $\alpha$, required if true.init.val is set to 0 or 2. The default is 1.

betas.init
: Initial values for $\beta_S$ , required if true.init.val is set to 0 or 2. The default is 0.5.

betat.init
: Initial values for $\beta_T$ , required if true.init.val is set to 0 or 2. The default is 0.5.

scale
: A numeric that allows to rescale (by multiplication) the survival times, to avoid numerical problems in the event of some convergence issues. If no change is needed the argument is set to 1, the default value. eg: 1/365 aims to convert days to years ".

random.generator
: Random number generator used by the Fortran compiler, 1 for the intrinsec subroutine Random_number and 2 for the subroutine uniran(). The default is 1. In the event of convergence problem with int.method set to 0, 2 or 4, that requires integration by Monte-Carlo, user could change the random numbers generator.

kappa.use
: A numeric, that indicates how to manage the smoothing parameters k_1 and k_2 in the event of convergence issues. If it is set to 1, the given smoothing parameters or those obtained by cross-validation are used. If it is set to 3, the associated smoothing parameters are successively divided by 10, in the event of convergence issues until 5 times. If it is set to 4, the management of the smoothing parameter is as in the event 1, follows by the successive division as described in the event 3 and preceded by the changing of the number of nodes for the Gauss-Hermite quadrature. The default is 4.

random
: A binary that says if we reset the random number generation with a different environment at each call (1) or not (0). If it is set to 1, we use the computer clock as seed. In the last case, it is not possible to reproduce the generated datasets". The default is 0. Required if random.generator is set to 1.

random.nb.sim
: If random is set to 1, a binary that indicates the number of generations that will be made.

seed
: The seed to use for data (or samples) generation. required if random is set to 0. The default is 0.

init.kappa
: smoothing parameter used to penalized the log-likelihood. By default (init.kappa = NULL) the values used are obtain by cross-validation.

`ckappa` Vector of two fixed values to add to the smoothing parameters. By default it is set to (0,0). this argument allows to well manage the smoothing parameters in the event of convergence issues.

`typecopula` The copula function used, can be 1 for clayton or 2 for Gumbel-Hougaard. The default is `1`

`nb.decimal` Number of decimal required for results presentation.

`print.times` a logical parameter to print estimation time. Default is TRUE.

`print.iter` a logical parameter to print iteration process. Default is FALSE.

## Details

The estimated parameter are obtained using the robust Marquardt algorithm (Marquardt, 1963) which is a combination between a Newton-Raphson algorithm and a steepest descent algorithm. The iterations are stopped when the difference between two consecutive log-likelihoods was small ($< 10^{-3}$), the estimated coefficients were stable (consecutive values ($< 10^{-3}$)), and the gradient small enough ($< 10^{-3}$), by default. Cubic M-splines of order 4 are used for the hazard function, and I-splines (integrated M-splines) are used for the cumulative hazard function.

The inverse of the Hessian matrix is the variance estimator and to deal with the positivity constraint of the variance component and the spline coefficients, a squared transformation is used and the standard errors are computed by the $\Delta$-method (Knight & Xekalaki, 2000). The smooth parameter can be chosen by maximizing a likelihood cross validation criterion (Joly and other, 1998).

We proposed based on the joint surrogate model a new definition of the Kendall's $\tau$. Moreover, distinct numerical integration methods are available to approximate the integrals in the marginal log-likelihood.

**Non-convergence case management procedure**

Special attention must be given to initializing model parameters, the choice of the number of spline knots, the smoothing parameters and the number of quadrature points to solve convergence issues. We first initialized parameters using the user's desired strategy, as specified by the option `true.init.val`. When numerical or convergence problems are encountered, with `kappa.use` set to `4`, the model is fitted again using a combination of the following strategies: vary the number of quadrature point (`nb.gh` to `nb.gh2` or `nb.gh2` to `nb.gh`) in the event of the use of the Gaussian Hermite quadrature integration (see `int.method`); divided or multiplied the smoothing parameters (`k_1` , `k_2` ) by 10 or 100 according to their preceding values, or used parameter vectors obtained during the last iteration (with a modification of the number of quadrature points and smoothing parameters). Using this strategy, we usually obtained during simulation the rejection rate less than $3\%$. A sensitivity analysis was conducted without this strategy, and similar results were obtained on the converged samples, with about a $23\%$ rejection rate.

## Value

This function return an object of class jointSurroPenal with elements :

`EPS` A vector containing the obtained convergence thresholds with the Marquardt algorithm, for the parameters, the log-likelihood and for the gradient;

`b` A vector containing estimates for the splines parameter's; elements of the lower triangular matrix (L) from the Cholesky decomposition such that $\Sigma = LL^T$, with $\Sigma$ the covariance of the random effects $(v_{S_i}, v_{T_i})$; the coefficient $\alpha$ (if `indicator.alpha` is set to 1); the satandard error of the random effect $u_i$; the logarithm of the copula parameter ($\theta$) if the Clayton copula function is considered, or the squared root of $\theta$ if the Gumbel copula is considered. The last two parameters represent the regression coefficients $\beta_S$ and $\beta_T$;

`varH` The variance matrix of all parameters in `b` (before positivity constraint transformation for the variance of the measurement error, for which the delta method is used);

`varHIH` The robust estimation of the variance matrix of all parameters in `b`;

`loglikPenal` The complete marginal penalized log-likelihood;

`LCV` the approximated likelihood cross-validation criterion in the semiparametric case (with `H` minus the converged Hessian matrix, and `l(.)` the full log-likelihood).

$$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$

;

`xS` vector of times for surrogate endpoint where both survival and hazard function are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times;

`lamS` array (dim = 3) of hazard estimates and confidence bands, for surrogate endpoint;

`survS` array (dim = 3) of baseline survival estimates and confidence bands, for surrogate endpoint;

`xT` vector of times for true endpoint where both survival and hazard function are estimated. By default seq(0, max(time), length = 99), where time is the vector of survival times;

`lamT` array (dim = 3) of hazard estimates and confidence bands, for true endpoint;

`survT` array (dim = 3) of baseline survival estimates and confidence bands, for true endpoint;

`n.iter` number of iterations needed to converge;

`theta` Estimate for $\theta$;

`gamma` Estimate for $\gamma$;

`alpha` Estimate for $\alpha$;

`zeta` A value equals to `1`, no really use in this function;

`sigma.s` Estimate for $\sigma^2_{v_S}$;

`sigma.t` Estimate for $\sigma^2_{v_T}$;

`sigma.st` Estimate for $\sigma_{v_{ST}}$ ;

`beta.s` Estimate for $\beta_S$ ;

`beta.t` Estimate for $\beta_T$ ;

`ui` A binary, that indicates if the heterogeneity between trial on the baseline risk has been Considered (`1`), using the shared cluster specific frailties $(u_i)$ , or not (`0`);

| | |
|---|---|
| ktau | The Kendall's $\tau$ with the correspondant 95 $\%$ CI obtained from the delta-method; |
| R2.boot | The $R^2_{trial}$ with the correspondant 95 $\%$ CI obtained from the parametric bootstrap; |
| Coefficients | The estimates with the corresponding standard errors and the 95 $\%$ CI |
| kappa | Positive smoothing parameters used for convergence. These values could be different to initial values if `kappa.use` is set to 3 or 4; |
| scale | The value used to rescale the survival times |
| data | The dataset used in the model |
| varcov.Sigma | Covariance matrix of the estimates of ($\sigma^2_{v_S}$,$\sigma^2_{v_T}$,$\sigma_{v_{ST}}$) obtained from the delta-method |
| parameter | List of all arguments used in the model |
| type.joint | A code 3 that represents the joint frailty-copula model. This output is used in other functions |

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

## References

Burzykowski, T., Molenberghs, G., Buyse, M., Geys, H., and Renard, D. (2001). Validation of surrogate end points in multiple randomized clinical trials with failure time end points. Journal of the Royal Statistical Society: Series C (Applied Statistics) 50, 405-422.

Buyse, M., Molenberghs, G., Burzykowski, T., Renard, D., and Geys, H. (2000). The validation of surrogate endpoints in meta-analyses of randomized experiments. Biostatistics 1, 49-67

Sofeu, C. L., Emura, T., and Rondeau, V. (2019). One-step validation method for surrogate endpoints using data from multiple randomized cancer clinical trials with failure-time endpoints. Statistics in Medicine 38, 2928-2942.

R. B. Nelsen. An introduction to Copulas. Springer, 2006

Prenen, L., Braekers, R., and Duchateau, L. (2017). Extending the archimedean copula methodology to model multivariate survival data grouped in clusters of variable size. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 79, 483-505.

Sofeu, C. L., Emura, T., and Rondeau, V. (2020). A joint frailty-copula model for meta-analytic validation of failure time surrogate endpoints in clinical trials. `Under review`

## See Also

jointSurrCopSimul, summary.jointSurroPenal, jointSurroPenal, jointSurroPenalSimul

## Examples

```
## Not run:
# Data from the advanced ovarian cancer randomized clinical trials.
data(dataOvarian)
```

```
joint.surro.Gumbel <- jointSurroCopPenal(data = dataOvarian, int.method = 0,
      n.knots = 8, maxit = 50, kappa.use = 4, nb.mc = 1000, typecopula = 2,
      print.iter = FALSE, scale = 1/365)

print(joint.surro.Gumbel)

joint.surro.Clayton <- jointSurroCopPenal(data = dataOvarian, int.method = 0,
      n.knots = 8, maxit = 50, kappa.use = 4, nb.mc = 1000, typecopula = 1,
      print.iter = FALSE, scale = 1/365)

print(joint.surro.Clayton)

## End(Not run)
```

---

`jointSurroPenal` *Fit the one-step Joint surrogate model for evaluating a canditate surrogate endpoint*

---

## Description

### Joint Frailty Surrogate model definition

Fit the one-step Joint surrogate model for the evaluation of a canditate surrogate endpoint, with different integration methods on the random effects, using a semiparametric penalized likelihood estimation. This approach extends that of Burzykowski `et al.` (2001) by including in the same joint frailty model the individual-level and the trial-level random effects. This function can also be used for mediation analysis where a direct effect of the surrogate time $S$ on the final endpoint $T$ is allowed through a function $g(S)$. For the $j^{th}$ subject (`j=1,...,`$n_i$) of the $i^{th}$ trial `i` (`i=1,...,`$G$), the joint surrogate model is defined as follows:

$$\left\{ \begin{array}{ll} \lambda_{S,ij}(t|\omega_{ij}, u_i, v_{S_i}, Z_{ij1}) & = \lambda_{0S}(t)\exp(\omega_{ij} + u_i + v_{S_i} Z_{ij1} + \beta_S Z_{ij1}) \\ \lambda_{T,ij}(t|\omega_{ij}, u_i, v_{T_i}, Z_{ij1}) & = \lambda_{0T}(t)\exp(\zeta\omega_{ij} + \alpha u_i + v_{T_i} Z_{ij1} + \beta_T Z_{ij1}) \end{array} \right.$$

where,

$$\omega_{ij} \sim N(0,\theta), u_i \sim N(0,\gamma), \omega_{ij} \perp u_i, u_i \perp v_{S_i}, u_i \perp v_{T_i}$$

and $(v_{S_i}, v_{T_i})^T \sim \mathcal{N}(0, \Sigma_v)$, with

$$\Sigma_v = \left( \begin{array}{cc} \sigma^2_{v_S} & \sigma_{v_{ST}} \\ \sigma_{v_{ST}} & \sigma^2_{v_T} \end{array} \right)$$

In this model, $\lambda_{0S}(t)$ is the baseline hazard function associated with the surrogate endpoint and $\beta_S$ the fixed treatment effect (or log-hazard ratio); $\lambda_{0T}(t)$ is the baseline hazard function associated with the true endpoint and $\beta_T$ the fixed treatment effect. $\omega_{ij}$ is a shared individual-level frailty that serve to take into account the heterogeneity in the data at the individual level; u<sub>i</sub> is a shared frailty effect associated with the baseline hazard function that serve to take into account the heterogeneity between trials of the baseline hazard function, associated with the fact that we have several trials in this meta-analytical design. The power parameters $\zeta$ and $\alpha$ distinguish both individual and trial-level heterogeneities between the surrogate and the true endpoint. $v_{S_i}$ and $v_{T_i}$ are two correlated random effects treatment-by-trial interactions. $Z_{ij1}$ represents the treatment arm to which the patient has been randomized. In the mediation analysis setting, the hazard function for the true endpoint becomes:

$$\lambda_{T,ij}(t|\omega_{ij}, u_i, v_{T_i}, Z_{ij1}, S_{ij}) = \lambda_{0T}(t)\exp(\zeta\omega_{ij} + \alpha u_i + v_{T_i}Z_{ij1} + \beta_T Z_{ij1} + I(S_{ij} \leq t)g(S_{ij}))$$

where the term $I(S_{ij} \leq t)g(S_{ij})$ allows for a direct effect of the surrogate time $S$ on the risk of occurrence of the final endpoint $T$.

**Surrogacy evaluation**

We proposed new definitions of Kendall's $\tau$ and coefficient of determination as individual-level and trial-level association measurements, to evaluate a candidate surrogate endpoint (Sofeu *et al.*, 2018). For the surrogacy in the mediation analysis setting see the "Surrogacy through mediation" Section.

**Individual-level surrogacy**

To measure the strength of association between $S_{ij}$ and $T_{ij}$ after adjusting the marginal distributions for the trial and the treatment effects, as show in Sofeu *et al.*(2018), we use the Kendall's $\tau$ define by :

$$\begin{aligned}\tau \; &= 2\int_{u_i}\int_{\omega_{ij}}\int_{u_{i'}}\int_{\omega_{i'j'}}\{ \\ &\frac{\exp(\omega_{ij}+u_i+\zeta\omega_{ij}+\alpha u_i)+\exp(\omega_{i'j'}+u_{i'}+\zeta\omega_{i'j'}+\alpha u_{i'})}{(\exp(\omega_{i'j'}+u_{i'})+\exp(\omega_{ij}+u_i))(\exp(\zeta\omega_{i'j'}+\alpha u_{i'})+\exp(\zeta\omega_{ij}+\alpha u_i))} \\ &\frac{1}{\sqrt{2\pi\theta}}\exp\left[-\frac{1}{2}\frac{\omega_{i'j'}^2}{\theta}\right]\frac{1}{\sqrt{2\pi\gamma}}\exp\left[-\frac{1}{2}\frac{u_{i'}^2}{\gamma}\right]d\omega_{i'j'}du_{i'} \\ &\frac{1}{\sqrt{2\pi\theta}}\exp\left[-\frac{1}{2}\frac{\omega_{ij}^2}{\theta}\right]\frac{1}{\sqrt{2\pi\gamma}}\exp\left[-\frac{1}{2}\frac{u_i^2}{\gamma}\right]d\omega_{ij}du_i\} - 1\end{aligned}$$

where $\theta, \zeta, \alpha$ and $\gamma$ are estimated using the joint surrogate model defined previously. Kendall's $\tau$ is the difference between the probability of concordance and the probability of discordance of two realizations of $S_{ij}$ and $T_{ij}$. It belongs to the interval [-1,1] and assumes a zero value when $S_{ij}$ and $T_{ij}$ are independent. We estimate Kendall's $\tau$ using Monte-Carlo or Gaussian Hermite quadrature integration methods. Its confidence interval is estimated using parametric bootstrap

**Trial-level surrogacy**

The key motivation for validating a surrogate endpoint is to be able to predict the effect of treatment on the true endpoint, based on the observed effect of treatment on the surrogate endpoint. As shown by Buyse *et al.* (2000), the coefficenient of determination obtains from the covariance matrix $\Sigma_v$ of the random effects treatment-by-trial interaction can be used to evaluate underlined prediction, and therefore as surrogacy evaluation measurement at trial-level. It is defined by:

$$R^2_{trial} = \frac{\sigma^2_{v_{ST}}}{\sigma^2_{v_S}\sigma^2_{v_T}}$$

The SEs of $R^2_{trial}$ is calculated using the Delta-method. We also propose $R^2_{trial}$ and 95% CI computed using the parametric bootstrap. The use of delta-method can lead to confidence limits violating the [0,1], as noted by (Burzykowski *et al.*, 2001). However, using other methods would not significantly alter the findings of the surrogacy assessment

**Surrogacy through mediation**

In the mediation analysis setting, the surrogacy measure is the proportion of treatment effect on the final endpoint $T$ that goes through its effect on the surrogate $S$. This measure is a time-dependent function $PTE(t)$ defined as:

$$PTE(t) = \frac{S_{11}(t) - S_{10}(t)}{S_{11}(t) - S_{00}(t)} = \frac{NIE(t)}{TE(t)}$$

where $NIE$ and $TE$ stand for "natual indirect effect" and "total effect" respectively. The numerator is the difference of the survival function of $T$ for a subject whose treatment has been set to $1$ (experimental arm) for both $S$ and $T$ versus a subject for which the treatment for $T$ is still $1$ but is set $0$ for $S$. This corresponds to the indirect effect (in ther of survival probability) of the treatment on $T$ through $S$. The denominator is the total effect of the treatment on $T$.

### Usage

```
jointSurroPenal(data, maxit=50, indicator.zeta = 1,
   indicator.alpha = 1, frail.base = 1, n.knots = 6,
   LIMparam = 0.001, LIMlogl = 0.001, LIMderiv = 0.001,
   nb.mc = 300, nb.gh = 32, nb.gh2 = 20, adaptatif = 0,
   int.method = 2, nb.iterPGH = 5, nb.MC.kendall = 10000,
   nboot.kendall = 1000, true.init.val = 0,
   theta.init = 1, sigma.ss.init = 0.5, sigma.tt.init = 0.5,
   sigma.st.init = 0.48, gamma.init = 0.5, alpha.init = 1,
   zeta.init = 1, betas.init = 0.5, betat.init = 0.5, scale = 1,
   random.generator = 1, kappa.use = 4, random = 0,
   random.nb.sim = 0, seed = 0, init.kappa = NULL, ckappa = c(0,0),
   nb.decimal = 4, print.times = TRUE, print.iter=FALSE,mediation=FALSE,
   g.nknots=1,pte.times=NULL,pte.ntimes=NULL,pte.nmc=500,pte.boot=FALSE,
   pte.nboot=2000,pte.boot.nmc=500,pte.integ.type=2)
```

### Arguments

`data` A `data.frame` containing at least seven variables entitled:

- `patientID`: A numeric, that represents the patient's identifier and must be unique;
- `trialID`: A numeric, that represents the trial in which each patient was randomized;
- `timeS`: The follow-up time associated with the surrogate endpoint;
- `statusS`: The event indicator associated with the surrogate endpoint. Normally 0 = no event, 1 = event;
- `timeT`: The follow-up time associated with the true endpoint;
- `statusT`: The event indicator associated with the true endpoint. Normally 0 = no event, 1 = event;
- `trt`: The treatment indicator for each patient, with 1 = treated, 0 = untreated.

`maxit` maximum number of iterations for the Marquardt algorithm. The default being `40`.

`indicator.zeta` A binary, indicates whether the power's parameter $\zeta$ should be estimated (1) or not (0). If `0`, $\zeta$ will be set to `1` during estimation. The default is `1`. This parameter can be set to `0` in the event of convergence and identification issues.

`indicator.alpha` A binary, indicating whether the power's parameter $\alpha$ should be estimated (1) or not (0). If `0`, $\alpha$ will be set to `1` during estimation. The default is 1.

`frail.base` A binary, indicating whether the heterogeneity between trial on the baseline risk is considered (`1`) or not (`0`), using the shared cluster specific frailties ($u_i$). The default is `1`.

`n.knots` integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See `frailtyPenal` for more details).

`LIMparam` Convergence threshold of the Marquardt algorithm for the parameters, $10^{-3}$ by default (See `frailtyPenal` for more details).

`LIMlogl` Convergence threshold of the Marquardt algorithm for the log-likelihood, $10^{-3}$ by default (See `frailtyPenal` for more details).

`LIMderiv` Convergence threshold of the Marquardt algorithm for the gradient, $10^{-3}$ by default (See `frailtyPenal` for more details).

`nb.mc` Number of samples considered in the Monte-Carlo integration. Required in the event `int.method` is equals to `0`, `2` or `4`. A value between 100 and 300 most often gives good results. However, beyond 300, the program takes a lot of time to estimate the parameters. The default is `300`.

`nb.gh` Number of nodes for the Gaussian-Hermite quadrature. It can be chosen among 5, 7, 9, 12, 15, 20 and 32. The default is 32.

`nb.gh2` Number of nodes for the Gauss-Hermite quadrature used to re-estimate the model, in the event of non-convergence, defined as previously. The default is `20`.

`adaptatif` A binary, indicates whether the pseudo adaptive Gaussian-Hermite quadrature `(1)` or the classical Gaussian-Hermite quadrature `(0)` is used. The default is `0`.

`int.method` A numeric, indicates the integration method: `0` for Monte carlo, `1` for Gaussian-Hermite quadrature, `2` for a combination of both Gaussian-Hermite quadrature to integrate over the individual-level random effects and Monte carlo to integrate over the trial-level random effects, `4` for a combination of both Monte carlo to integrate over the individual-level random effects and Gaussian-Hermite quadrature to integrate over the trial-level random effects. The default is `2`.

`nb.iterPGH` Number of iterations before the re-estimation of the posterior random effects, in the event of the two-steps pseudo-adaptive Gaussian-hermite quadrature. If set to `0` there is no re-estimation". The default is `5`.

`nb.MC.kendall` Number of generated points used with the Monte-Carlo to estimate integrals in the Kendall's $\tau$ formulation. Beter to use at least 4000 points for stable reseults. The default is `10000`.

nboot.kendall
: Number of samples considered in the parametric bootstrap to estimate the confidence interval of the Kendall's $\tau$. The default is 1000.

true.init.val
: Numerical value. Indicates if the given initial values to parameters (0) should be considered. If set to 2, $\alpha$ and $\gamma$ are initialised using two separed shared frailty model (see frailtyPenal for more details); $\sigma^2_{v_S}$, $\sigma^2_{v_T}$ and $\sigma_{v_{ST}}$ are fixed by the user or the default values; $\zeta$, $\theta$, $\beta_S$ and $\beta_T$ are initialized using a classical joint frailty model, considering individual level random effects. If the joint frailty model is faced to convergence issues, $\beta_S$ and $\beta_T$ are initialized using two shared frailty models. In all other scenarios, if the simplified model does not converge, default given parameters values are used. Initial values for spline's associated parameters are fixed to 0.5. The default for this argument is 0.

theta.init
: Initial values for $\theta$, required if true.init.val is set to 0 or 2. The default is 1.

sigma.ss.init
: Initial values for $\sigma^2_{v_S}$ , required if true.init.val is set to 0 or 2. The default is 0.5.

sigma.tt.init
: Initial values for $\sigma^2_{v_T}$ , required if true.init.val is set to 0 or 2. The default is 0.5.

sigma.st.init
: Initial values for$\sigma_{v_{ST}}$ , required if true.init.val is set to 0 or 2. The default is 0.48.

gamma.init
: Initial values for $\gamma$, required if true.init.val is set to 0 or 2. The default is 0.5.

alpha.init
: Initial values for $\alpha$, required if true.init.val is set to 0 or 2. The default is 1.

zeta.init
: Initial values for $\zeta$, required if true.init.val is set to 0 or 2. The default is 1.

betas.init
: Initial values for $\beta_S$ , required if true.init.val is set to 0 or 2. The default is 0.5.

betat.init
: Initial values for $\beta_T$ , required if true.init.val is set to 0 or 2. The default is 0.5.

scale
: A numeric that allows to rescale (multiplication) the survival times, to avoid numerical problems in the event of some convergence issues. If no change is needed the argument is set to 1, the default value. eg: 1/365 aims to convert days to years ".

random.generator
: Random number generator used by the Fortran compiler, 1 for the intrinsec subroutine Random_number and 2 for the subroutine uniran(). The default is 1. in the event of convergence problem with int.method set to 0, 2 or 4, that requires integration by Monte-Carlo, user could change the random numbers generator.

kappa.use
: A numeric, that indicates how to manage the smoothing parameters k_1 and k_2 in the event of convergence issues. If it is set to 1, the given smoothing parameters or those obtained by cross-validation are used. If it is set to 3, the associated smoothing parameters are successively divided by 10, in the event of convergence issues until 5 times. If it is set to 4, the management of the smoothing parameter is as in the event 1, follows by the successive division as described in the event 3 and preceded by the changing of the number of nodes for the Gauss-Hermite quadrature. The default is 4.

| | |
|---|---|
| `random` | A binary that says if we reset the random number generation with a different environment at each call `(1)` or not `(0)`. If it is set to `1`, we use the computer clock as seed. In the last case, it is not possible to reproduce the generated datasets. The default is `0`. Required if `random.generator` is set to 1. |
| `random.nb.sim` | If `random` is set to `1`, a binary that indicates the number of generations that will be made. |
| `seed` | The seed to use for data (or samples) generation. required if `random` is set to `0`. The default is `0`. |
| `init.kappa` | smoothing parameter used to penalized the log-likelihood. By default (init.kappa = NULL) the values used are obtain by cross-validation. |
| `ckappa` | Vector of two fixed values to add to the smoothing parameters. By default it is set to (0,0). this argument allows to well manage the smoothing parameters in the event of convergence issues. |
| `nb.decimal` | Number of decimal required for results presentation. |
| `print.times` | a logical parameter to print estimation time. Default is TRUE. |
| `print.iter` | a logical parameter to print iteration process. Default is FALSE. |
| `mediation` | a logical value indicating if the mediation analysis method is used. Default is FALSE. |
| `g.nknots` | In the case of a mediation analysis, indicates how many inner knots are used in the splines basis for estimating the function $g(s)$. The value of `g.nknots` should be between 1 and 5. Default is 1. |
| `pte.times` | In the mediation analysis setting, a vector of times for which the funtion $PTE(t)$ is evaluated. Specified time points must be in the range of the observed event times. The length of the vector should be less than 200. |
| `pte.ntimes` | In the mediation setting, if the argument `pte.times` is not specified the argument `pte.ntimes` allows the user to only specify a number of time points for which the function $PTE(t)$ has to be computed. This argument is only to be used if `pte.times` is not specified. In that case the default value for `pte.ntimes` is 10. The value of `pte.ntimes` should be less than 200. |
| `pte.nmc` | An integer indicating how many Monte Carlo simulations are used to integrate over the random effects in the computation of the function $PTE(t)$. in the mediation analysis setting. Default is 500. |
| `pte.boot` | A logical value indicating if bootstrapped confidence bands needs to be computed for the function $PTE(t)$ in the mediation analysis setting. Default is FALSE. |
| `pte.nboot` | An integer indicating how many bootstrapped replicates of $PTE(t)$ needs to be computed to derive confidence bands for $PTE(t)$. Default is 2000. |
| `pte.boot.nmc` | If `pte.boot` is TRUE, indicates how many Monte Carlo simulations are used to integrate over the random effects in the bootstrapped functions $PTE(t)$ in the mediation analysis setting. Default is 500 |
| `pte.integ.type` | An integer indicating which type of integration over the distribution of $S$ should be used in the computation of the function $PTE(t)$. If set to `1`, a simple trapezoidal rule is used with 300 integration points. If set to `2` a Gauss-Laguerre quadrature is used with 30 knots. Default is `2`. |

## Details

The estimated parameter are obtained using the robust Marquardt algorithm (Marquardt, 1963) which is a combination between a Newton-Raphson algorithm and a steepest descent algorithm. The iterations are stopped when the difference between two consecutive log-likelihoods was small ($< 10^{-3}$), the estimated coefficients were stable (consecutive values ($< 10^{-3}$)), and the gradient small enough ($< 10^{-3}$), by default. Cubic M-splines of order 4 are used for the hazard function, and I-splines (integrated M-splines) are used for the cumulative hazard function.

The inverse of the Hessian matrix is the variance estimator and to deal with the positivity constraint of the variance component and the spline coefficients, a squared transformation is used and the standard errors are computed by the $\Delta$-method (Knight & Xekalaki, 2000). The smooth parameter can be chosen by maximizing a likelihood cross validation criterion (Joly and other, 1998).

We proposed based on the joint surrogate model a new definition of the Kendall's $\tau$. Moreover, distinct numerical integration methods are available to approximate the integrals in the marginal log-likelihood.

**Non-convergence case management procedure**

Special attention must be given to initializing model parameters, the choice of the number of spline knots, the smoothing parameters and the number of quadrature points to solve convergence issues. We first initialized parameters using the user's desired strategy, as specified by the option `true.init.val`. When numerical or convergence problems are encountered, with `kappa.use` set to `4`, the model is fitted again using a combination of the following strategies: vary the number of quadrature point (`nb.gh` to `nb.gh2` or `nb.gh2` to `nb.gh`) in the event of the use of the Gaussian Hermite quadrature integration (see `int.method`); divided or multiplied the smoothing parameters (`k_1` , `k_2` ) by 10 or 100 according to their preceding values, or used parameter vectors obtained during the last iteration (with a modification of the number of quadrature points and smoothing parameters). Using this strategy, we usually obtained during simulation the rejection rate less than $3\%$. A sensitivity analysis was conducted without this strategy, and similar results were obtained on the converged samples, with about a 23% rejection rate.

## Value

This function return an object of class jointSurroPenal or jointSurroMed in the mediation analysis setting with elements:

`EPS` A vector containing the obtained convergence thresholds with the Marquardt algorithm, for the parameters, the log-likelihood and for the gradient;

`b` A vector containing estimates for the splines parameter's; the power's parameter $\zeta$ (if `indicator.zeta` is set to `1`), the standard error of the shared individual-level frailty $\omega_{ij}$ ($\theta$),elements of the lower triangular matrix (L) from the Cholesky decomposition such that $\Sigma = LL^T$, with $\Sigma$ the covariance of the random effects $(v_{S_i}, v_{T_i})$; the coefficient $\alpha$ (if `indicator.alpha` is set to `1`); the satandard error of the random effect $u_i$; and the regression coefficients $\beta_S$ and $\beta_T$;

`varH` The variance matrix of all parameters in `b` (before positivity constraint transformation for the variance of the measurement error, for which the delta method is used);

`varHIH` The robust estimation of the variance matrix of all parameters in `b`;

`loglikPenal` The complete marginal penalized log-likelihood;

`LCV` the approximated likelihood cross-validation criterion in the semiparametric case (with `H` minus the converged Hessian matrix, and `l(.)` the full log-likelihood).

$$LCV = \frac{1}{n}(trace(H^{-1}_{pl}H) - l(.))$$

;

`xS` vector of times for surrogate endpoint where both survipte.nmc.bootval and hazard function are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times;

`lamS` array (dim = 3) of hazard estimates and confidence bands, for surrogate endpoint;

`survS` array (dim = 3) of baseline survival estimates and confidence bands, for surrogate endpoint;

`xT` vector of times for true endpoint where both survival and hazard function are estimated. By default seq(0, max(time), length = 99), where time is the vector of survival times;

`lamT` array (dim = 3) of hazard estimates and confidence bands, for true endpoint;

`survT` array (dim = 3) of baseline survival estimates and confidence bands, for true endpoint;

`n.iter` number of iterations needed to converge;

`theta` Estimate for $\theta$;

`gamma` Estimate for $\gamma$;

`alpha` Estimate for $\alpha$;

`zeta` Estimate for $\zeta$;

`sigma.s` Estimate for $\sigma^2_{v_S}$;

`sigma.t` Estimate for $\sigma^2_{v_T}$;

`sigma.st` Estimate for $\sigma_{v_{ST}}$ ;

`beta.s` Estimate for $\beta_S$ ;

`beta.t` Estimate for $\beta_T$ ;

`ui` A binary, that indicates if the heterogeneity between trial on the baseline risk has been Considered (`1`), using the shared cluster specific frailties $(u_i)$ , or not (`0`);

`ktau` The Kendall's $\tau$ with the correspondant 95 $\%$ CI computed using the parametric bootstrap;

`R2.boot` The $R^2_{trial}$ with the correspondant 95 $\%$ CI computed using the parametric bootstrap;

`Coefficients` The estimates with the corresponding standard errors and the 95 $\%$ CI

`kappa` Positive smoothing parameters used for convergence. These values could be different to initial values if `kappa.use` is set to `3` or `4`;

`scale` The value used to rescale the survival times

`data` The dataset used in the model

`varcov.Sigma` covariance matrix of ($\sigma^2_{v_S}$,$\sigma^2_{v_T}$,$\sigma_{v_{ST}}$)
obtained from the delta-method

`parameter` list of all arguments used in the model

`mediation` List returned in the case where the option `mediation` is set to `TRUE` which contains:

- data.pte: A dataframe containing estimated values for the funtion $PTE(t)$ and the natural effects for differents time points
- g.knots: The vector of knots used in the spline basis for the function g.
- g.order: The order of the spline basis used to estimate the function g.
- g.coefficients: A vector containing the estimated coefficients associated with the splines in the estimation of the function g.
- data.g: A dataframe containing the values of the estimated function g computed at several time points and the associated 95
- pte.ci: A dataframe containing the 95
- TE.ci: A dataframe containing the 95
- NDE.ci: A dataframe containing the 95
- NIE.ci: A dataframe containing the 95

### Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com>, Quentin Le Coent <quentin.le-coent@u-bordeaux.fr> and Virginie Rondeau <virginie.rondeau@inserm.fr>,

### References

Burzykowski, T., Molenberghs, G., Buyse, M., Geys, H., and Renard, D. (2001). Validation of surrogate end points in multiple randomized clinical trials with failure time end points. Journal of the Royal Statistical Society: Series C (Applied Statistics) 50, 405-422.

Buyse, M., Molenberghs, G., Burzykowski, T., Renard, D., and Geys, H. (2000). The validation of surrogate endpoints in meta-analyses of randomized experiments. Biostatistics 1, 49-67

Sofeu, C. L., Emura, T., and Rondeau, V. (2019). One-step validation method for surrogate endpoints using data from multiple randomized cancer clinical trials with failure-time endpoints. Statistics in Medicine 38, 2928-2942.

Le Coent, Q., Legrand, C., Rondeau, V. (2021). Time-to-event surrogate endpoint validation using mediation and meta-analytic data. Article submitted.

### See Also

`jointSurrSimul`, `summary.jointSurroPenal`, `jointSurroPenalSimul`

### Examples

```
## Not run:
# Generation of data to use
data.sim <- jointSurrSimul(n.obs=600, n.trial = 30,cens.adm=549.24,
         alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1, sigma.s = 0.7,
```

```
         sigma.t = 0.7, cor = 0.8, betas = -1.25, betat = -1.25,
         full.data = 0, random.generator = 1, seed = 0, nb.reject.data = 0)


#Surrogacy evaluation based on generated data with a combination of Monte Carlo
#and classical Gaussian Hermite integration.*
# (Computation takes around 5 minutes)

joint.surro.sim.MCGH <- jointSurroPenal(data = data.sim, int.method = 2,
                   nb.mc = 300, nb.gh = 20)

#Surrogacy evaluation based on generated data with a combination of Monte Carlo
# and Pseudo-adaptive Gaussian Hermite integration.
# (Computation takes around 4 minutes)

joint.surro.sim.MCPGH <- jointSurroPenal(data = data.sim, int.method = 2,
                   nb.mc = 300, nb.gh = 20, adaptatif = 1)

# Results
summary(joint.surro.sim.MCGH)
summary(joint.surro.sim.MCPGH)

# Data from the advanced ovarian cancer randomized clinical trials.
# Joint surrogate model with \eqn{\zeta} fixed to 1, 8 nodes spline
# and the rescaled survival time.

data(dataOvarian)
# (Computation takes around 20 minutes)

joint.surro.ovar <- jointSurroPenal(data = dataOvarian, n.knots = 8,
                init.kappa = c(2000,1000), indicator.alpha = 0, nb.mc = 200,
                scale = 1/365)
# results
summary(joint.surro.ovar)

print(joint.surro.ovar)


# Mediation analysis on the adjuvant chemotherapy
# dataset where the surrogate is a time-to-relapse and the final endpoint is death.
# 4 knots are used to estimate the two baseline hazard functions.
# The function g(s) is estimated using cubic b-splines with 1 interior
# knot ('g.nkots=1'). The function \eqn{PTE(t)} is computed at 100 time points
# using 10.000 Monte Carlo simulations for integration over the random effects.
# To reduce computation time in the provided example only one fifth of the
# the original dataset is used and the confidence bands for the function
# \eqn{PTE(t)} are not computed as well as the power parameters associated with
# the random effects. Full example is commented thereafter.

# We first need to change the variable "statusS" which in the dataset
# encodes the indicator of a disease free survival event to an indicator
# of a time to relapse event (i.e., resurgence of cancer or
# onset of a second cancer) that excludes death as a composite event.
```

```
# Thus, the patients whose variables "timeS" and "timeT" are equal
# and whose variable "statusS" is equal to 1 will have
# "statuS" be set to 0. We do this because composite endpoint may not
# be appropriate in the setting of mediation analysis.

data(gastadj)
gastadj$timeS<-gastadj$timeS/365
gastadj$timeT<-gastadj$timeT/365
#here changing "statusS" to corresponds to a time to relapse event
gastadj[gastadj$timeS==gastadj$timeT & gastadj$statusS==1,c("statusS")]<-0

# select 20% of the original dataset
set.seed(1)
n<-nrow(gastadj)
subset<-gastadj[sort(sample(1:nrow(gastadj),round(n*0.2),replace = FALSE)),]

# Mediation model ('mediation=TRUE'). Computation takes around 17 minutes
mod.gast<-jointSurroPenal(subset,n.knots = 4,indicator.zeta = 0,
                         indicator.alpha = 0,mediation=TRUE,g.nknots=1,
                         pte.ntimes=30,pte.nmc=10000,pte.boot=FALSE)

summary(mod.gast)
plot(mod.gast)

# Example on the full dataset, including estimation of the power parameters
# mod.gast2<-jointSurroPenal(gastadj,n.knots = 4,mediation=TRUE,g.nknots=1,
#                             pte.ntimes=30,pte.nmc=10000,pte.boot=TRUE,
#                             pte.nboot=2000,pte.boot.nmc=10000)

# results
# plot(mod.gast2)
# summary(mod.gast2)

## End(Not run)
```

---

jointSurroPenalSimul *Simulation studies based on the one-step Joint surrogate models for the evaluation of a canditate surrogate endpoint*

---

## Description

This function aims to allow simulation studies, based on the joint frailty surrogate model, described in jointSurroPenal. Simulation can also be based on the joint frailty-copula model described in jointSurroCopPenal

## Usage

```
jointSurroPenalSimul(maxit=40, indicator.zeta = 1,
   indicator.alpha = 1, frail.base = 1, n.knots = 6, nb.dataset = 1,
   nbSubSimul=1000, ntrialSimul=30, LIMparam = 0.001,
```

```
LIMlogl = 0.001, LIMderiv = 0.001, nb.mc = 300, nb.gh = 32,
nb.gh2 = 20, adaptatif = 0, int.method = 2, nb.iterPGH = 5,
nb.MC.kendall = 10000, nboot.kendall = 1000, true.init.val = 0,
theta.init = 1, sigma.ss.init = 0.5, sigma.tt.init = 0.5,
sigma.st.init = 0.48, gamma.init = 0.5, alpha.init = 1,
zeta.init = 1, betas.init = 0.5, betat.init = 0.5,
random.generator = 1, equi.subj.trial = 1, prop.subj.trial = NULL,
equi.subj.trt = 1, prop.subj.trt = NULL,
theta2 = 3.5, zeta = 1, gamma.ui = 2.5, alpha.ui = 1,
betas = -1.25, betat = -1.25, lambdas = 1.8, nus = 0.0045,
lambdat = 3, nut = 0.0025, prop.cens = 0, time.cens = 549, R2 = 0.81,
sigma.s = 0.7, sigma.t = 0.7, kappa.use = 4, random = 0,
random.nb.sim = 0, seed = 0, nb.reject.data = 0, init.kappa = NULL,
ckappa = c(0,0), type.joint.estim = 1, type.joint.simul = 1,
mbetast =NULL, mbetast.init = NULL, typecopula =1, theta.copula = 6,
thetacopula.init = 3, filter.surr = c(1), filter.true = c(1),
nb.decimal = 4, pfs = 0, print.times = TRUE, print.iter=FALSE)
```

## Arguments

| | |
|---|---|
| maxit | maximum number of iterations for the Marquardt algorithm. Default is 40. |
| indicator.zeta | A binary, indicates whether the power's parameter $\zeta$ should be estimated (1) or not (0). It is required if type.joint.estim = 1. If 0, $\zeta$ will be set to 1 during estimation. The default is 1. This parameter can be seted to 0 in the event of identification issues. |
| indicator.alpha | A binary, indicates whether the power's parameter $\alpha$ should be estimated (1) or not (0). If 0, $\alpha$ will be set to 1 during estimation. The default is 1. This parameter can be seted to 0 in the event of identification issues. |
| frail.base | Considered the heterogeneity between trial on the baseline risk (1), using the shared cluster specific frailties ($u_i$), or not (0). The default is 1. |
| n.knots | integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See frailtyPenal for more details). |
| nb.dataset | Number of dataset to analyze. The default is 1. |
| nbSubSimul | Number of subjects. |
| ntrialSimul | Number of trials. |
| LIMparam | Convergence threshold of the Marquardt algorithm for the parameters, $10^{-3}$ by default (See frailtyPenal for more details). |
| LIMlogl | Convergence threshold of the Marquardt algorithm for the log-likelihood, $10^{-3}$ by default (See frailtyPenal for more details). |
| LIMderiv | Convergence threshold of the Marquardt algorithm for the gradient, $10^{-3}$ by default (See frailtyPenal for more details). |

`nb.mc` Number of samples considered in the Monte-Carlo integration. Required in the event `int.method` is equals to `0`, `2` or `4`. A value between 100 and 300 most often gives good results. However, beyond 300, the program takes a lot of time to estimate the parameters. The default is `300`.

`nb.gh` Number of nodes for the Gaussian-Hermite quadrature. It can be chosen among 5, 7, 9, 12, 15, 20 and 32. The default is 32.

`nb.gh2` Number of nodes for the Gauss-Hermite quadrature used to re-estimate the model, in the event of non-convergence, defined as previously. The default is `20`.

`adaptatif` A binary, indicates whether the pseudo adaptive Gaussian-Hermite quadrature `(1)` or the classical Gaussian-Hermite quadrature `(0)` is used. The default is `0`.

`int.method` A numeric, indicates the integration method: `0` for Monte carlo, `1` for Gaussian-Hermite quadrature. If `type.joint.estim = 1` this parameter can be set to `2` for a combination of both Gaussian-Hermite quadrature to integrate over the individual-level random effects and Monte carlo to integrate over the trial-level random effects, `4` for a combination of both Monte carlo to integrate over the individual-level random effects and Gaussian-Hermite quadrature to integrate over the trial-level random effects. If `type.joint.estim = 3`, value `3` indicates integration using Laplace approximation . The default is `2`.

`nb.iterPGH` Number of iterations before the re-estimation of the posterior random effects, in the event of the two-steps pseudo-adaptive Gaussian-hermite quadrature. If set to `0` there is no re-estimation". The default is `5`.

`nb.MC.kendall` Number of generated points used with the Monte-Carlo to estimate integrals in the Kendall's $\tau$ formulation. Beter to use at least 4000 points for stable results. Required if `type.joint.estim = 1`, the default is `10000`.

`nboot.kendall` Number of samples considered in the parametric bootstrap to estimate the confidence interval of the Kendall's $\tau$, or R<sup>2</sup><sub>trial</sub>. The default is `1000`.

`true.init.val` Numerical value. Indicates if the real parameter values `(1)`, or the given initial values to parameters `(0)` should be considered. If set to `2`, $\alpha$ and $\gamma$ are initialised using two separed shared frailty model (see `frailtyPenal` for more details); $\sigma^2_{v_S}$, $\sigma^2_{v_T}$ and $\sigma_{v_{ST}}$ are fixed using the default initial values given by the user; $\zeta$, $\theta$, $\beta_S$ and $\beta_T$ are initialized using a classical joint frailty model, considering individual level random effects. If the joint frailty model is faced to convergence issues, $\beta_S$ and $\beta_T$ are initialized using two shared frailty models. In all others scenarios, if the simplified model does not converge, default given parameters values are used. Initial values for spline's associated parameters are fixed to `0.5`. The default for this argument is `0`.

`theta.init` Initial values for $\theta$, required if `true.init.val` is set to `0` or `2`, and `type.joint.estim = 1`. The default is `1`.

`sigma.ss.init` Initial values for $\sigma^2_{v_S}$ , required if `true.init.val` is set to `0` or `2`. The default is `0.5`.

`sigma.tt.init` Initial values for $\sigma^2_{v_T}$ , required if `true.init.val` is set to `0` or `2`. The default is `0.5`.

`sigma.st.init` Initial values for $\sigma_{v_{ST}}$ , required if `true.init.val` is set to `0` or `2`. The default is `0.48`.

gamma.init — Initial values for $\gamma$, required if `true.init.val` is set to `0` or `2`. The default is `0.5`.

alpha.init — Initial values for $\alpha$, required if `true.init.val` is set to `0` or `2`. The default is `1`.

zeta.init — Initial values for $\zeta$, required if `true.init.val` is set to `0` or `2` and `type.joint.estim = 1`. The default is `1`.

betas.init — Initial values for $\beta_S$ , required if `true.init.val` is set to `0` or `2`. The default is `0.5`.

betat.init — Initial values for $\beta_T$ , required if `true.init.val` is set to `0` or `2`. The default is `0.5`.

random.generator — Random number generator used by the Fortran compiler, `1` for the intrinsec subroutine `Random_number` and `2` for the subroutine `uniran()`. The default is `1`.

equi.subj.trial — A binary, that indicates if the same proportion of subjects per trial should be considered in the procces of data generation (1) or not (0). In the event of different trial sizes, fill in `prop.subj.trial` the proportions of subjects to be considered per trial. The default is `1`.

prop.subj.trial — Vector of the proportions of subjects to consider per trial. Requires if the argument `equi.subj.trial` is different to `1`. The size of this vector is equal to the number of trials.

equi.subj.trt — Indicates if the same proportion of treated subjects per trial should be considered `(1)` or not `(0)`. If `0`, fill in `prop.subj.trt` the proportions of treated subjects to be considered per trial. The default is `1`.

prop.subj.trt — Vector of the proportions of treated subjects to consider per trial. Requires if the argument `equi.subj.trt` is different to `0.5`. The size of this vector is equal to the number of trials.

theta2 — True value for $\theta$. Require if `type.joint.simul = 1`, the default is `3.5`.

zeta — True value for $\zeta$ in the event of simulation. The default is `1`.

gamma.ui — True value for $\gamma$ in the event of simulation. The default is `2.5`.

alpha.ui — True value for $\alpha$ in the event of simulation. The default is `1`.

betas — True value for $\beta_S$ in the event of simulation. The default is `-1.25`.

betat — True value for $\beta_T$ in the event of simulation. The default is `-1.25`.

lambdas — Desired scale parameter for the `Weibull` distribution associated with the Surrogate endpoint. The default is `1.8`.

nus — Desired shape parameter for the `Weibull` distribution associated with the Surrogate endpoint. The default is `0.0045`.

lambdat — Desired scale parameter for the `Weibull` distribution associated with the True endpoint.The default is `3`.

nut — Desired shape parameter for the `Weibull` distribution associated with the True endpoint. The default is `0.0025`.

| | |
|---|---|
| prop.cens | A value between 0 and 1, 1-prop.cens is the minimum proportion of people who are randomly censored. Represents the quantile to use for generating the random censorship time. In this case, the censorship time follows a uniform distribution in 1 and (prop.cens)ieme percentile of the generated death times. If this argument is set to 0, the fix censorship is considered. The default is 0. Required if type.joint.simul = 3. |
| time.cens | Censorship time. If argument prop.cens is set to 0, it represents the administrative censorship time, else it represents the fix censoring time. The default is 549, for about 40% of fix censored subjects. |
| R2 | Desired $R^2_{trial}$ . The default is 0.81. |
| sigma.s | True value for $\sigma^2_{v_S}$ The default is 0.7. |
| sigma.t | True value for $\sigma^2_{v_T}$. The default is 0.7. |
| kappa.use | A numeric, that indicates how to manage the smoothing parameters k_1 and k_2 in the event of convergence issues. If it is set to 0, the first smoothing parameters that allowed convergence on the first dataset is used for all simulations. if it is set to 1, a smoothing parameter is estimated by cross-validation for each dataset generated. If it is set to 2, the same process for chosing kappas as in the event 1 is used, but in the event of convergence issue, the first smoothing parameters that allowed convergence among the three previous that have worked is used. If it is set to 3, the associated smoothing parameters are successively divided by 10, in the event of convergence issues until 5 times. If it is set to 4, the management of the smoothing parameters is as in the event 2, preceded by the successive division described in the event 3 and by the changing of the number of nodes for the Gauss-Hermite quadrature. The default is 4. |
| random | A binary that says if we reset the random number generation with a different environment at each call (1) or not (0). If it is set to 1, we use the computer clock as seed. In the last case, it is not possible to reproduce the generated datasets. The default is 0. Required if random.generator is set to 1. |
| random.nb.sim | If random is set to 1, a binary that indicates the number of generations that will be made, equal to nb.dataset in this case. |
| seed | The seed to use for data generation. Required if random is set to 0. The default is 0. |
| nb.reject.data | When the simulations have been split into several packets, this argument indicates the number of generated datasets to reject before starting the simulations studies. This prevents to reproduce the same datasets for all simulation packages. It must be set to 0 if just one packet is considered, the default. Otherwise for each packet of simulation run, this value must be updated. e.g. If 10 packets are considered for a total of 100 datasets, one can assigned 0 for the first packet run, 10 for the second, 20 for the 3rd, ... , 90 for the 10th. If this argument is different to 0, the argument nb.dataset must be set to the number of dataset to consider in the packet. |
| init.kappa | smoothing parameter used to penalized the log-likelihood. By default (init.kappa = NULL) the values used are obtain by cross-validation. |
| ckappa | Vector of two constantes to add to the smoothing parameters. By default it is set to (0,0). this argument allows to well manage the smoothing parameters in the event of convergence issues. |

`type.joint.estim`
: Model to considered for the estimation. If this argument is set to `1`, the joint surrogate model is used, the default (see jointSurroPenal). If set to `3`, parameters are estimated under the joint frailty-copula model for surrogacy (see jointSurroCopPenal).

`type.joint.simul`
: Model to considered for data generation. If this argument is set to `1`, the joint surrogate model is used, the default (see jointSurroPenal). If set to `3`, data are generated following the joint frailty-copula model for surrogacy (see jointSurroCopPenal).

`mbetast`
: Matrix or dataframe containing the true fixed traitment effects associated with the covariates. This matrix includes two columns (first one for surrogate endpoint and second one for true endpoint) and the number of row corresponding to the number of covariate. Require if `type.joint.simul = 3` with more than one covariate. The default is NULL and assume only the treatment effect

`mbetast.init`
: Matrix or dataframe containing the initial values for the fixed effects associated with the covariates. This matrix include two columns (first one for surrogate endpoint and second one for true endpoint) and the number of row corresponding to the number of covariate. Require if `type.joint.simul = 3` with more than one covariate. The default is NULL and assume only the treatment effect

`typecopula`
: The copula function used for estimation: 1 = clayton, 2 = Gumbel. Require if `type.joint.simul = 3`, the default is 1

`theta.copula`
: The copula parameter. Require if `type.joint.simul = 3`. The default is `6`, for an individual-level association (kendall's $\tau$) of 0.75 in the event of Clayton copula

`thetacopula.init`
: Initial value for the copula parameter. Require if `type.joint.estim = 3`, the default is 3

`filter.surr`
: Vector of size the number of covariates, with the i-th element that indicates if the hazard for surrogate is adjusted on the i-th covariate (code 1) or not (code 0). By default, only the treatment effect is considered.

`filter.true`
: Vector defines as `filter.surr`, for true endpoint. `filter.true` and `filter.surr` should have the same size

`nb.decimal`
: Number of decimal required for results presentation.

`pfs`
: Is used to specified if the time to progression should be censored by the death time (0) or not (1). The default is 0. In the event with pfs set to 1, death is included in the surrogate endpoint as in the definition of PFS or DFS.

`print.times`
: a logical parameter to print estimation time. Default is TRUE.

`print.iter`
: a logical parameter to print iteration process. Default is FALSE.

## Details

The estimated parameter are obtained using the robust Marquardt algorithm (Marquardt, 1963) which is a combination between a Newton-Raphson algorithm and a steepest descent algorithm. The iterations are stopped when the difference between two consecutive log-likelihoods was small

(< $10^{-3}$), the estimated coefficients were stable (consecutive values (< $10^{-3}$), and the gradient small enough (< $10^{-3}$), by default. Cubic M-splines of order 4 are used for the hazard function, and I-splines (integrated M-splines) are used for the cumulative hazard function.

The inverse of the Hessian matrix is the variance estimator and to deal with the positivity constraint of the variance component and the spline coefficients, a squared transformation is used and the standard errors are computed by the $\Delta$-method (Knight & Xekalaki, 2000). The smooth parameter can be chosen by maximizing a likelihood cross validation criterion (Joly and other, 1998).

We proposed based on the joint surrogate model a new definition of the Kendall's $\tau$. By cons, for the joint frailty-copula model, we based the individual-level association on a definition of $\tau$ clause to that of the classical two-step approch (Burzykowski et al, 2001), but conditional on the random effects. Moreover, distinct numerical integration methods are available to approximate the integrals in the marginal log-likelihood.

**Non-convergence case management procedure**

Special attention must be given to initializing model parameters, the choice of the number of spline knots, the smoothing parameters and the number of quadrature points to solve convergence issues. We first initialized parameters using the user's desired strategy, as specified by the option `true.init.val`. When numerical or convergence problems are encountered, with `kappa.use` set to `4`, the model is fitted again using a combination of the following strategies: vary the number of quadrature point (`nb.gh` to `nb.gh2` or `nb.gh2` to `nb.gh`) in the event of the use of the Gaussian Hermite quadrature integration (see `int.method`); divided or multiplied the smoothing parameters (`k_1` , `k_2` ) by 10 or 100 according to their preceding values, or used parameter vectors obtained during the last iteration (with a modification of the number of quadrature points and smoothing parameters). Using this strategy, we usually obtained during simulation the rejection rate less than 3%. A sensitivity analysis was conducted without this strategy, and similar results were obtained on the converged samples, with about a 23% rejection rate.

## Value

This function returns an object of class jointSurroPenalSimul with elements :

| | |
|---|---|
| `theta2` | True value for $\theta$, if `type.joint.estim = 1`; |
| `theta.copula` | Copula parameter, if `type.joint.estim = 3`; |
| `zeta` | true value for $\zeta$, if `type.joint.estim = 1`; |
| `gamma.ui` | true value for $\gamma$; |
| `alpha.ui` | true value for $\alpha$; |
| `sigma.s` | true value for $\sigma^2_{v_S}$ ; |
| `sigma.t` | true value for $\sigma^2_{v_T}$ ; |
| `sigma.st` | true value for $\sigma_{v_{ST}}$ ; |
| `betas` | true value for $\beta_S$ ; |
| `betat` | true value for $\beta_T$ ; |
| `R2` | true value for $R^2_{trial}$ ; |
| `nb.subject` | total number of subjects used; |
| `nb.trials` | total number of trials used; |
| `nb.simul` | number of simulated datasets; |

| | |
|---|---|
| nb.gh | number of nodes for the Gaussian-Hermite quadrature; |
| nb.gh2 | number of nodes for the Gauss-Hermite quadrature used to re-estimate the model, in the event of non-convergence; |
| nb.mc | number of samples considered in the Monte-Carlo integration; |
| kappa.use | a numeric, that indicates how to manage the smoothing parameters k_1 and k_2 in the event of convergence issues; |
| n.knots | number of knots used for splines; |
| int.method | integration method used; |
| n.iter | mean number of iterations needed to converge; |
| dataTkendall | a matrix with nb.dataset line(s) and three columns, of the estimates of Kendall's $\tau$ and theirs confidence intervals (obtained using parametric bootstrap if type.joint.estim = 1 or Delta method if type.joint.estim = 3). All non-convergence cases are represented by a line of 0; |
| dataR2boot | a matrix with nb.dataset line(s) and three columns, of the estimates of $R^2_{trial}$ and theirs confidence intervals using the parametric bootstrap. All non-convergence cases are represented by a line of 0. |
| dataParamEstim | a dataframe including all estimates with the associated standard errors, for all simulation. All non-convergence cases are represented by a line of 0; |
| dataHessian | Dataframe of the variance-Covariance matrices of the estimates for all simulations |
| dataHessianIH | Dataframe of the robust estimation of the variance matrices of the estimates for all simulations |
| datab | Dataframe of the estimates for all simulations which rich convergence |
| type.joint | the estimation model; 1 for the joint surrogate and 3 for joint frailty-copula model |
| type.joint.simul | The model used for data generation; 1 for joint surrogate and 3 for joint frailty-copula |
| true.init.val | Indicates if the real parameter values have been used as initial values for the model (1), or the given initial values (0) |

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

## References

Burzykowski, T., Molenberghs, G., Buyse, M., Geys, H., and Renard, D. (2001). Validation of surrogate end points in multiple randomized clinical trials with failure time end points. Journal of the Royal Statistical Society: Series C (Applied Statistics) 50, 405-422.

Sofeu, C. L., Emura, T., and Rondeau, V. (2019). One-step validation method for surrogate endpoints using data from multiple randomized cancer clinical trials with failure-time endpoints. Statistics in Medicine 38, 2928-2942.

## See Also

jointSurroPenal, jointSurroCopPenal, summary.jointSurroPenalSimul , jointSurrSimul, jointSurrCopSimul

## Examples

```
## Not run:
# Surrogacy model evaluation performance study based on 10 generated data
# (Computation takes around 20 minutes using a processor including 40
# cores and a read only memory of 378 Go)
# To realize a simulation study on 100 samples or more (as required), use
# nb.dataset = 100

### joint frailty model
joint.simul <- jointSurroPenalSimul(nb.dataset = 10, nbSubSimul= 600,
                   ntrialSimul = 30, LIMparam = 0.001, LIMlogl = 0.001,
                   LIMderiv = 0.001, nb.mc = 200, nb.gh = 20,
                   nb.gh2 = 32, true.init.val = 1, print.iter = FALSE, pfs = 0)

# results
summary(joint.simul, d = 3, R2boot = 1) # bootstrap
summary(joint.simul, d = 3, R2boot = 0) # Delta-method

### joint frailty copula model

joint.simul.cop.clay <- jointSurroPenalSimul(nb.dataset = 10, nbSubSimul= 600,
                   ntrialSimul = 30, nb.mc = 1000, type.joint.estim = 3,
                   typecopula = 1, type.joint.simul = 3, theta.copula = 3,
                   time.cens = 349, true.init.val = 1, R2 = 0.81, maxit = 40,
                   print.iter = FALSE)

summary(joint.simul.cop.clay)


## End(Not run)
```

---

`jointSurroTKendall` *Kendall's $\tau$ estimation using numerical integration methods*

---

## Description

This function estimate the Kendall's $\tau$ based on the joint surrogate model described in jointSurroPenal (Sofeu *et al.*, 2018), for the evaluation of a candidate surrogate endpoints, at the individual-level . We used the Monte-carlo and the gaussian Hermite quadrature methods for numerical integration. in the event of Gaussian Hermite quadrature, it is better to choose at least `20` quadature nodes for better results. The actual value of nodes used is the maximum between `20` and `nb.gh`

## Usage

```
jointSurroTKendall(object = NULL, theta, gamma, alpha = 1, zeta = 1,
                   sigma.v = matrix(rep(0,4),2,2), int.method = 0,
                   nb.MC.kendall = 10000, nb.gh = 32,
                   random.generator = 1, random = 0,
                   random.nb.sim = 0, seed = 0, ui = 1)
```

## Arguments

| | |
|---|---|
| `object` | An object inheriting from `jointSurroPenal` class. The default is `NULL` |
| `theta` | Variance of the individual-level random effect, $\omega_{ij}$ . Required if `object` is set to `NULL` |
| `gamma` | Variance of the trial-level random effect associated with the baseline risk, $u_i$ . Required if `object` is set to `NULL`. The default is `3.5`. |
| `alpha` | Power parameter associated with $u_i$ . Required if `object` is set to `NULL`. The default is `1`. |
| `zeta` | Power parameter associated with $\omega_{ij}$ . Required if `object` is set to `NULL` The default is `1`. |
| `sigma.v` | Covariance matrix of the random effects treatment-by-trial interaction $(v_{S_i}, v_{T_i})$ |
| `int.method` | A numeric, indicates the integration method: `0` for Monte carlo and `1` for Gaussian-Hermite quadrature. The default is `0` |
| `nb.MC.kendall` | Number of generated points used with the Monte-Carlo to estimate integrals in the Kendall's $\tau$ formulation. Beter to use at least 4000 points for stable results. The default is `10000`. |
| `nb.gh` | Number of nodes for the Gaussian-Hermite quadrature. The default is `32`. |
| `random.generator` | Random number generator to use by the Fortran compiler, `1` for the intrinsec subroutine `Random_number` and `2` for the subroutine `uniran()`. The default is `1`. |
| `random` | A binary that says if we reset the random number generation with a different environment at each call `(1)` or not `(0)`. If it is set to `1`, we use the computer clock as a seed. In the last case, it is not possible to reproduce the generated datesets". The default is `0`. |
| `random.nb.sim` | If `random` is set to `1`, a binary that indicates the number of generations that will be made. |
| `seed` | The seed to use for data (or samples) generation. required if `random` is set to `0`. The default is `0`. |
| `ui` | A binary, indicates whether one considered trial random effect associated with the baseline risk (`1`) or not (`0`). The default is `1`. |

## Value

This function return the estimated Kendall's $\tau$

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

## References

Sofeu C.L., Emura T. and Rondeau V. (2018). One-step validation method for surrogate endpoints in multiple randomized cancer clinical trials with failure-time endpoints. Under review

## See Also

jointSurrSimul, summary.jointSurroPenal

## Examples

```
Ktau1 <- jointSurroTKendall(theta = 3.5, gamma = 2.5, nb.gh = 32)
Ktau2 <- jointSurroTKendall(theta = 1, gamma = 0.8, alpha = 1, zeta = 1,
         nb.gh = 32)

###---Kendall's \eqn{\tau} from a joint surrogate model ---###

## Not run:
data.sim <-jointSurrSimul(n.obs=400, n.trial = 20,cens.adm=549,
          alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1,
          sigma.s = 0.7, sigma.t = 0.7,cor = 0.8, betas = -1.25,
          betat = -1.25, full.data = 0, random.generator = 1,
          seed = 0, nb.reject.data = 0)


###---Estimation---###
joint.surrogate <- jointSurroPenal(data = data.sim, nb.mc = 300,
                   nb.gh = 20, indicator.alpha = 1, n.knots = 6)

 Ktau3 <- jointSurroTKendall(joint.surrogate)
 Ktau4 <- jointSurroTKendall(joint.surrogate,nb.MC.kendall = 4000,
          seed = 1)

## End(Not run)
```

---

# jointSurrSimul *Generate survival times for two endpoints using the joint frailty surrogate model*

---

## Description

Date are generated from the one-step joint surrogate model (see jointSurroPenal for more details)

### Usage

```
jointSurrSimul(
  n.obs = 600,
  n.trial = 30,
  cens.adm = 549.24,
  alpha = 1.5,
  theta = 3.5,
  gamma = 2.5,
  zeta = 1,
  sigma.s = 0.7,
  sigma.t = 0.7,
  cor = 0.8,
  betas = -1.25,
  betat = -1.25,
  frailt.base = 1,
  lambda.S = 1.8,
  nu.S = 0.0045,
  lambda.T = 3,
  nu.T = 0.0025,
  ver = 1,
  typeOf = 1,
  equi.subj.trial = 1,
  equi.subj.trt = 1,
  prop.subj.trial = NULL,
  prop.subj.trt = NULL,
  full.data = 0,
  random.generator = 1,
  random = 0,
  random.nb.sim = 0,
  seed = 0,
  nb.reject.data = 0,
  pfs = 0
)
```

### Arguments

| | |
|---|---|
| `n.obs` | Number of considered subjects. The default is `600`. |
| `n.trial` | Number of considered trials. The default is `30`. |
| `cens.adm` | censorship time. The default is `549`, for about `40%` of censored subjects. |
| `alpha` | Fixed value for $\alpha$. The default is `1.5`. |
| `theta` | Fixed value for $\theta$. The default is `3.5`. |
| `gamma` | Fixed value for $\gamma$. The default is `2.5`. |
| `zeta` | Fixed value for $\zeta$. The default is `1`. |
| `sigma.s` | Fixed value for $\sigma^2_{v_S}$ . The default is `0.7`. |
| `sigma.t` | Fixed value for $\sigma^2_{v_T}$ . The default is `0.7`. |

cor
: Desired level of correlation between $v_{S_i}$ and $v_{T_i}$. $R^2_{trial} = cor^2$ . The default is `0.8`.

betas
: Fixed value for $\beta_S$ . The default is `-1.25`.

betat
: Fixed value for $\beta_T$ . The default is `-1.25`.

frailt.base
: considered the heterogeneity on the baseline risk `(1)` or not `(0)`. The default is `1`.

lambda.S
: Desired scale parameter for the `Weibull` distribution associated with the Surrogate endpoint. The default is 1.8.

nu.S
: Desired shape parameter for the `Weibull` distribution associated with the Surrogate endpoint. The default is 0.0045.

lambda.T
: Desired scale parameter for the `Weibull` distribution associated with the True endpoint. The default is 3.

nu.T
: Desired shape parameter for the `Weibull` distribution associated with the True endpoint. The default is 0.0025.

ver
: Number of covariates. For surrogte evaluation, we just considered one covatiate, the treatment arm

typeOf
: Type of joint model used for data generation: 0 = classical joint model with a shared individual frailty effect (Rondeau, 2007), 1 = joint surrogate model with shared frailty effects $u_i$ and $\omega_{ij}$ , and two correlated random effects treatment-by-trial interaction ($v_{S_i}$, $v_{T_i}$) as described in Sofeu et al. (2018).

equi.subj.trial
: A binary variable that indicates if the same proportion of subjects should be included per trial (1) or not (0). If 0, the proportions of subject per trial are required in parameter `prop.subj.trial`.

equi.subj.trt
: A binary variable that indicates if the same proportion of subjects is randomized per trial (1) or not (0). If 0, the proportions of subject per trial are required in parameter `prop.subj.trt`.

prop.subj.trial
: The proportions of subjects per trial. Requires if `equi.subj.trial=0`.

prop.subj.trt
: The proportions of randomized subject per trial. Requires if `equi.subj.trt=0`.

full.data
: Specified if you want the function to return the full dataset (1), including the random effects, or the restictive dataset (0) with `7` columns required for the function `jointSurroPenal`.

random.generator
: Random number generator used by the Fortran compiler, `1` for the intrinsec subroutine `Random_number` and `2` for the subroutine `uniran()`. The default is `1`.

random
: A binary that says if we reset the random number generation with a different environment at each call `(1)` or not `(0)`. If it is set to `1`, we use the computer clock as seed. In the last case, it is not possible to reproduce the generated datasets. The default is `0`. Required if `random.generator` is set to 1.

random.nb.sim
: required if `random.generator` is set to 1, and if `random` is set to 1.

seed
: The seed to use for data (or samples) generation. Required if the argument `random.generator` is set to 1. Must be a positive value. If negative, the program do not account for seed. The default is `0`.

`nb.reject.data` Number of generation to reject before the considered dataset. This parameter is required when data generation is for simulation. With a fixed parameter and `random.generator` set to 1, all ganerated data are the same. By varying this parameter, different datasets are obtained during data genarations. The default value is 0, in the event of one dataset.

`pfs` Is used to specify if the time to progression should be censored by the death time (0) or not (1). The default is 0. In the event with pfs set to 1, death is included in the surrogate endpoint as in the definition of PFS or DFS.

## Details

We just considered in this generation, the Gaussian random effects. If the parameter `full.data` is set to 1, this function return a list containning severals parameters, including the generated random effects. the desired individual level correlation (Kendall's $\tau$) depend on the values of $\alpha$, $\theta$, $\gamma$ and $\zeta$.

## Value

This function return if the parameter `full.data` is set to 0, a `data.frame` with columns :

`patientID` A numeric, that represents the patient's identifier, must be unique;

`trialID` A numeric, that represents the trial in which each patient was randomized;

`trt` The treatment indicator for each patient, with 1 = treated, 0 = untreated;

`timeS` The follow up time associated with the surrogate endpoint;

`statusS` The event indicator associated with the surrogate endpoint. Normally 0 = no event, 1 = event;

`timeT` The follow up time associated with the true endpoint;

`statusT` The event indicator associated with the true endpoint. Normally 0 = no event, 1 = event;

If the argument `full.data` is set to 1, additionnal colums corresponding to random effects $\omega_{ij}$ , $u_i$ , $v_{S_i}$ and $v_{T_i}$ are returned. Note that $u_i$ , $v_{S_i}$ and $v_{T_i}$ are returned if `typeOf` is set to 1

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

## References

Rondeau V., Mathoulin-Pelissier S., Jacqmin-Gadda H., Brouste V. and Soubeyran P. (2007). Joint frailty models for recurring events and death using maximum penalized likelihood estimation: application on cancer events. Biostatistics 8(4), 708-721.

Sofeu, C. L., Emura, T., and Rondeau, V. (2019). One-step validation method for surrogate endpoints using data from multiple randomized cancer clinical trials with failure-time endpoints. Statistics in Medicine 38, 2928-2942.

## See Also

`jointSurrSimul`

## Examples

```
data.sim <- jointSurrSimul(n.obs=600, n.trial = 30,cens.adm=549.24,
            alpha = 1.5, theta = 3.5, gamma = 2.5, sigma.s = 0.7,
            zeta = 1, sigma.t = 0.7, cor = 0.8, betas = -1.25,
            betat = -1.25, full.data = 0, random.generator = 1,
            seed = 0, nb.reject.data = 0, pfs = 0)
```

---

`longDat` *Longitudinal semicontinuous biomarker dataset (TPJM)*

---

## Description

This is a simulated dataset used to illustrate the two-part joint model included in the longiPenal function.

## Usage

```
data(longDat)
```

## Format

This data frame contains the following columns:

**id** The identification number of a patient

**timej** The measurement times of the biomarker

**trtY** Treatment covariate

**Y** Biomarker value

---

`longiPenal` *Fit a Joint Model for Longitudinal Data and a Terminal Event*

---

## Description

Fit a joint model for longitudinal data and a terminal event using a semiparametric penalized likelihood estimation or a parametric estimation on the hazard function.

The longitudinal outcomes $y_i(t_{ik})$ ($k = 1, \ldots, n_i$, $i = 1, \ldots, N$) for $N$ subjects are described by a linear mixed model and the risk of the terminal event is represented by a proportional hazard risk model. The joint model is constructed assuming that the processes are linked via a latent structure (Wulfsohn and Tsiatsis 1997):

$$\begin{cases} y_i(t_{ik}) = \boldsymbol{X}_{Li}(t_{ik})^\top \boldsymbol{\beta}_L + \boldsymbol{Z}_i(t_{ik})^\top \boldsymbol{b}_i + \epsilon_i(t_{ik}) & \text{(Longitudinal)} \\ \lambda_i(t|\boldsymbol{b}_i) = \lambda_0(t) \exp(\boldsymbol{X}_{Ti}(t)\boldsymbol{\beta}_T + h(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))^\top \boldsymbol{\eta}_T) & \text{(Terminal)} \end{cases}$$

where $\boldsymbol{X}_{Li}(t)$ and $\boldsymbol{X}_{Ti}$ are vectors of fixed effects covariates and $\boldsymbol{\beta}_L$ and $\boldsymbol{\beta}_T$ are the associated coefficients. Measurements errors $\epsilon_i(t_{ik})$ are iid normally distributed with mean 0 and variance $\sigma_\epsilon^2$. The random effects $\boldsymbol{b}_i = (b_{0i}, \ldots, b_{qi})^\top \sim \mathcal{N}(0, \boldsymbol{B}_1)$ are associated to covariates $\boldsymbol{Z}_i(t)$ and independent from the measurement error. The relationship between the two processes is explained via $h(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))$ with coefficients $\boldsymbol{\eta}_T$. Two forms of the function $h(\cdot)$ are available: the random effects $\boldsymbol{b}_i$ and the current biomarker level $m_i(t) = \boldsymbol{X}_{Li}(t_{ik})^\top \boldsymbol{\beta}_L + \boldsymbol{Z}_i(t_{ik})^\top \boldsymbol{b}_i$.

We consider that the longitudinal outcome can be a subject to a quantification limit, i.e. some observations, below a level of detection $s$ cannot be quantified (left-censoring).

## Usage

```
longiPenal(formula, formula.LongitudinalData, data, data.Longi,
 formula.Binary = FALSE, random, random.Binary = FALSE,
 fixed.Binary = FALSE, GLMlog = FALSE, MTP = FALSE, id,
 intercept = TRUE, link = "Random-effects", timevar = FALSE,
 left.censoring = FALSE, n.knots, kappa, maxit = 350, hazard = "Splines",
 mediation = FALSE, med.center = NULL, med.trt = NULL,
 init.longi = NULL, init.surv = NULL, init.Eta,
 method.GH = "Standard",seed.MC = 1, n.nodes, LIMparam = 1e-3,
 LIMlogl = 1e-3, LIMderiv = 1e-3, print.times = TRUE, med.nmc = 500,
 pte.times = NULL, pte.ntimes = NULL, pte.nmc = 500,
 pte.boot = FALSE, pte.nboot = 2000)
```

## Arguments

`formula` a formula object, with the response on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. Interactions are possible using * or :.

`formula.LongitudinalData` a formula object, with the longitudinal outcome on the left and terms indicating the modelling variables on the right. It must follow the standard form used for linear mixed-effects models. Interactions are possible using * or :.

`data` a 'data.frame' with the variables used in `formula`.

`data.Longi` a 'data.frame' with the variables used in `formula.LongitudinalData`.

`formula.Binary` a formula object, with terms on the right to indicate which variables are modelling the binary part of the two-part model fitting the longitudinal semicontinuous outcome. It must follow the standard form used for linear mixed-effects models. Interactions are possible using * or :.

`random` Names of variables for the random effects of the longitudinal outcome. Maximum 3 random effects are possible at the moment. The random intercept is chosen using `"1"`.

`random.Binary` Names of variables for the random effects of the binary part of the two-part model fitting the longitudinal semicontinuous outcome. The random intercept is chosen using `"1"`.

`fixed.Binary` Fix the value of the intercept in the binary part of a two-part model.

`GLMlog` Logical value. Use a lognormal distribution for the biomarker (instead of the default normal distribution).

| | |
|---|---|
| `MTP` | Logical value. Marginal two-part joint model instead of conditional two-part joint model (only with two-part models). |
| `id` | Name of the variable representing the individuals. |
| `intercept` | Logical value. Is the fixed intercept of the biomarker included in the mixed-effects model? The default is `TRUE`. |
| `link` | Type of link function for the dependence between the biomarker and death: `"Random-effects"` for the association directly via the random effects of the biomarker, `"Current-level"` for the association via the true current level of the biomarker. The option `"Current-level"` can be chosen only if the biomarker random effects are associated with the intercept and time (following this order). `"Two-part"`, this structure is only applicable with conditional two-part models, the effect of the current probability of positive value and the effect of the expected value among positive values on the risk of event is evaluated separately. The default is `"Random-effects"`. |
| `timevar` | Indicates the time varying variables to take into account this evolution over time in the link with the survival model (useful with 'Current-level' and 'Two-part' links) |
| `left.censoring` | Is the biomarker left-censored below a threshold $s$? The default is `FALSE`, ie. no left-censoring. In case of a left-censored biomarker, this argument must be equal to the threshold $s$. |
| `n.knots` | Integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See Note in `frailtyPenal` function) |
| `kappa` | Positive smoothing parameter in the penalized likelihood estimation. The coefficient kappa of the integral of the squared second derivative of hazard function in the fit (penalized log likelihood). To obtain an initial value for `kappa`, a solution is to fit the corresponding Cox model using cross validation (See `cross.validation` in function `frailtyPenal`). We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them. |
| `maxit` | Maximum number of iterations for the Marquardt algorithm. The default is 350. |
| `hazard` | Type of hazard functions: `"Splines"` for semiparametric hazard functions using equidistant intervals or `"Splines-per"` using percentile with the penalized likelihood estimation, `"Weibull"` for the parametric Weibull functions. The default is `"Splines"`. |
| `mediation` | a logical value indicating if the mediation analysis method is used. Default is FALSE. |
| `med.center` | For mediation analysis, a vector containing the center indicator for each subject If no center then this argument should be `NULL`. Default is `NULL`. |
| `med.trt` | For mediation analysis, a vector containing the treatment indicator for each subject. |

init.longi
: A model estimated with the lme function from the nlme package used to set initial values for the regression coefficients, the random effects and the measurement error in the longitudinal outcome submodel. The lme model should have the same number of parameters as the longitudinal submodel from longiPenal model to be estimated. Alternatively, init.longi can be a list with elements "fixed", "random" and "sigma" representing respectively the regression coefficients, the random effects variance matrix and the standard deviation of the measurement error. Default is 0.5 for each.

init.surv
: A model estimated with the frailtyPenal function used to set initial values for the baseline risk function and the regression coefficients in the survival outcome submodel. The frailtyPenal model should have the same number of parameters (baseline risk and regression coeffcients) than the survival submodel of the longiPenal model. Alternatively, init.surv can be a list with elements "hazard" and "fixed" representing the baseline risk function parameters (with splines baseline hazard, it corresponds to the coefficients of the splins functions, with Weibull hazard, it corresponds to two parameters) and the regression coeffcients. Default is 0.5 for each element.

init.Eta
: Initial values for regression coefficients for the link function. Default is 0.5 for each.

method.GH
: Method for the Gauss-Hermite quadrature: "Standard" for the standard non-adaptive Gaussian quadrature, "Pseudo-adaptive" for the pseudo-adaptive Gaussian quadrature, "Monte-carlo" for the Monte-carlo method and "HRMSYM" for the algorithm for the multivariate non-adaptive Gaussian quadrature (see Details). The default is "Standard".

seed.MC
: Monte-carlo integration points selection (1=fixed, 0=random)

n.nodes
: Number of nodes for the Gauss-Hermite quadrature or the Monte-carlo method. They can be chosen among 5, 7, 9, 12, 15, 20 and 32 for the GH quadrature and any number for the Monte-carlo method. The default is 9.

LIMparam
: Convergence threshold of the Marquardt algorithm for the parameters (see Details of frailtyPenal function), $10^{-3}$ by default.

LIMlogl
: Convergence threshold of the Marquardt algorithm for the log-likelihood (see Details of frailtyPenal function), $10^{-3}$ by default.

LIMderiv
: Convergence threshold of the Marquardt algorithm for the gradient (see Details of frailtyPenal function), $10^{-3}$ by default.

print.times
: a logical parameter to print iteration process. The default is TRUE.

med.nmc
: For mediation analysis, the number of Monte Carlo points used for computing the integral over the random effects in the likelihood computation. Default is 500.

pte.times
: For mediation analysis, a vector of times for which the funtion $PTE(t)$ is evaluated. Specified time points must be in the range of the observed event times. The length of the vector should be less than 200.

pte.ntimes
: For mediation analysis, if the argument pte.times is not specified the argument pte.ntimes allows the user to only specify a number of time points for which the function $PTE(t)$ has to be computed. This argument is only to be used if pte.times is not specified. In that case the default value for pte.ntimes is 10. Should be less than 200.

`pte.nmc` For mediation analysis, nn integer indicating how many Monte Carlo simulations are used to integrate over the random effects in the computation of the function $PTE(t)$. Should be less than 10000. Default is 500.

`pte.boot` For mediation analysis, a logical value indicating if bootstrapped confidence bands needs to be computed for the function $PTE(t)$ in the mediation analysis setting. Default is FALSE.

`pte.nboot` For mediation analysis, an integer indicating how many bootstrapped replicates of PTE(t) needs to be computed to derive confidence bands for PTE(t). Should be less than 10000. Default is 2000.

## Details

Typical usage for the joint model

```
longiPenal(Surv(time,event)~var1+var2, biomarker ~ var1+var2,
data, data.Longi, ...)
```

The method of the Gauss-Hermite quadrature for approximations of the multidimensional integrals, i.e. length of `random` is 2, can be chosen among the standard, non-adaptive, pseudo-adaptive in which the quadrature points are transformed using the information from the fitted mixed-effects model for the biomarker (Rizopoulos 2012) or multivariate non-adaptive procedure proposed by Genz et al. 1996 and implemented in FORTRAN subroutine HRMSYM. The choice of the method is important for estimations. The standard non-adaptive Gauss-Hermite quadrature (`"Standard"`) with a specific number of points gives accurate results but can be time consuming. The non-adaptive procedure (`"HRMSYM"`) offers advantageous computational time but in case of datasets in which some individuals have few repeated observations (biomarker measures or recurrent events), this method may be moderately unstable. The pseudo-adaptive quadrature uses transformed quadrature points to center and scale the integrand by utilizing estimates of the random effects from an appropriate linear mixed-effects model. This method enables using less quadrature points while preserving the estimation accuracy and thus lead to a better computational time.The Monte-Carlo method is also proposed for approximations of the multidimensional integrals.

NOTE. Data frames `data` and `data.Longi` must be consistent. Names and types of corresponding covariates must be the same, as well as the number and identification of individuals.

## Value

The following components are included in a 'longiPenal' object for each model:

`b` The sequence of the corresponding estimation of the coefficients for the hazard functions (parametric or semiparametric), the random effects variances and the regression coefficients.

`call` The code used for the model.

`formula` The formula part of the code used for the terminal event part of the model.

`formula.LongitudinalData`
The formula part of the code used for the longitudinal part of the model.

`formula.Binary` The formula part of the code used for the binary part of the two-part model.

`coef` The regression coefficients (first for the terminal event and then for the biomarker.

| | |
|---|---|
| `groups` | The number of groups used in the fit. |
| `kappa` | The value of the smoothing parameter in the penalized likelihood estimation corresponding to the baseline hazard function for the terminal event. |
| `logLikPenal` | The complete marginal penalized log-likelihood in the semiparametric case. |
| `logLik` | The marginal log-likelihood in the parametric case. |
| `n.measurements` | The number of biomarker observations used in the fit. |
| `max_rep` | The maximal number of repeated measurements per individual. |
| `n.deaths` | The number of events observed in the fit. |
| `n.iter` | The number of iterations needed to converge. |
| `n.knots` | The number of knots for estimating the baseline hazard function in the penalized likelihood estimation. |
| `n.strat` | The number of stratum. |
| `varH` | The variance matrix of all parameters (before positivity constraint transformation for the variance of the measurement error, for which the delta method is used). |
| `varHIH` | The robust estimation of the variance matrix of all parameters. |
| `xD` | The vector of times where both survival and hazard function of the terminal event are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times. |
| `lamD` | The array (dim=3) of baseline hazard estimates and confidence bands (terminal event). |
| `survD` | The array (dim=3) of baseline survival estimates and confidence bands (terminal event). |
| `median` | The value of the median survival and its confidence bands. |
| `typeof` | The type of the baseline hazard functions (0:"Splines", "2:Weibull"). |
| `npar` | The number of parameters. |
| `nvar` | The vector of number of explanatory variables for the terminal event and biomarker. |
| `nvarEnd` | The number of explanatory variables for the terminal event. |
| `nvarY` | The number of explanatory variables for the biomarker. |
| `noVarEnd` | The indicator of absence of the explanatory variables for the terminal event. |
| `noVarY` | The indicator of absence of the explanatory variables for the biomarker. |
| `LCV` | The approximated likelihood cross-validation criterion in the semiparametric case (with H minus the converged Hessian matrix, and l(.) the full log-likelihood). $$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$ |
| `AIC` | The Akaike information Criterion for the parametric case. $$AIC = \frac{1}{n}(np - l(.))$$ |
| `n.knots.temp` | The initial value for the number of knots. |

`shape.weib` The shape parameter for the Weibull hazard function.

`scale.weib` The scale parameter for the Weibull hazard function.

`martingaledeath.res` The martingale residuals for each individual.

`conditional.res` The conditional residuals for the biomarker (subject-specific): $\boldsymbol{R}_i^{(m)} = \boldsymbol{y}_i - \boldsymbol{X}_{Li}^\top \widehat{\boldsymbol{\beta}}_L - \boldsymbol{Z}_i^\top \widehat{\boldsymbol{b}}_i$.

`marginal.res` The marginal residuals for the biomarker (population averaged): $\boldsymbol{R}_i^{(c)} = \boldsymbol{y}_i - \boldsymbol{X}_{Li}^\top \widehat{\boldsymbol{\beta}}_L$.

`marginal_chol.res` The Cholesky marginal residuals for the biomarker: $\boldsymbol{R}_i^{(m)} = \widehat{\boldsymbol{U}_i^{(m)}} \boldsymbol{R}_i^{(m)}$, where $\widehat{\boldsymbol{U}_i^{(m)}}$ is an upper-triangular matrix obtained by the Cholesky decomposition of the variance matrix $\boldsymbol{V}_{\boldsymbol{R}_i^{(m)}} = \widehat{\boldsymbol{V}_i} - \boldsymbol{X}_{Li}(\sum_{i=1}^N \boldsymbol{X}_{Li} \widehat{\boldsymbol{V}_i}^{-1} \boldsymbol{X}_{Li})^{-1} \boldsymbol{X}_{Li}^\top$.

`conditional_st.res` The standardized conditional residuals for the biomarker.

`marginal_st.res` The standardized marginal residuals for the biomarker.

`random.effects.pred` The empirical Bayes predictions of the random effects (ie. using conditional posterior distributions).

`pred.y.marg` The marginal predictions of the longitudinal outcome.

`pred.y.cond` The conditional (given the random effects) predictions of the longitudinal outcome.

`lineardeath.pred` The linear predictor for the terminal part.

`global_chisq_d` The vector with values of each multivariate Wald test for the terminal part.

`dof_chisq_d` The vector with degrees of freedom for each multivariate Wald test for the terminal part.

`global_chisq.test_d` The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the terminal part).

`p.global_chisq_d` The vector with the p_values for each global multivariate Wald test for the terminal part.

`global_chisq` The vector with values of each multivariate Wald test for the longitudinal part.

`dof_chisq` The vector with degrees of freedom for each multivariate Wald test for the longitudinal part.

`global_chisq.test` The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the longitudinal part).

`p.global_chisq` The vector with the p_values for each global multivariate Wald test for the longitudinal part.

`names.factordc` The names of the "as.factor" variables for the terminal part.

`names.factor` The names of the "as.factor" variables for the longitudinal part.

`intercept` The logical value. Is the fixed intercept included in the linear mixed-effects model?

`B1` The variance matrix of the random effects for the longitudinal outcome.

`ResidualSE` The standard deviation of the measurement error.

`eta` The regression coefficients for the link function.

`ne_re` The number of random effects used in the fit.

`names.re` The names of variables for the random effects.

`link` The name of the type of the link function.

`eta_p.value` p-values of the Wald test for the estimated regression coefficients for the link function.

`beta_p.value` p-values of the Wald test for the estimated regression coefficients.

`leftCensoring` The logical value. Is the longitudinal outcome left-censored?

`leftCensoring.threshold`
For the left-censored biomarker, the value of the left-censoring threshold used for the fit.

`prop.censored` The fraction of observations subjected to the left-censoring.

`methodGH` The method used for approximations of the multidimensional integrals.

`n.nodes` The number of integration points.

## References

A. Krol, A. Mauguen, Y. Mazroui, A. Laurent, S. Michiels and V. Rondeau (2017). Tutorial in Joint Modeling and Prediction: A Statistical Software for Correlated Longitudinal Outcomes, Recurrent Events and a Terminal Event. *Journal of Statistical Software* **81**(3), 1-52.

A. Krol, L. Ferrer, JP. Pignon, C. Proust-Lima, M. Ducreux, O. Bouche, S. Michiels, V. Rondeau (2016). Joint Model for Left-Censored Longitudinal Data, Recurrent Events and Terminal Event: Predictive Abilities of Tumor Burden for Cancer Evolution with Application to the FFCD 2000-05 Trial. *Biometrics* **72**(3) 907-16.

D. Rizopoulos (2012). Fast fitting of joint models for longitudinal and event time data using a pseudo-adaptive Gaussian quadrature rule. *Computational Statistics and Data Analysis* **56**, 491-501.

M.S. Wulfsohn, A.A. and Tsiatis, A. A. (1997). A joint model for survival and longitudinal data measured with error. *Biometrics* **53**, 330-9.

A. Genz and B. Keister (1996). Fully symmetric interpolatory rules for multiple integrals over infinite regions with Gaussian weight. *Journal of Computational and Applied Mathematics* **71**, 299-309.

D. Rustand, L. Briollais, C. Tournigand and V. Rondeau (2020). Two-part joint model for a longitudinal semicontinuous marker and a terminal event with application to metastatic colorectal cancer data. *Biostatistics*.

## See Also

plot.longiPenal,print.longiPenal,summary.longiPenal

## Examples

```
## Not run:

###--- Joint model for longitudinal data and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

# We recommend fitting first a mixed model on the longitudinal data and
# a survival model on the terminal event and use these estimations as
# initial values for the joint model.
# Mixed model with lme:
mlongi <- nlme::lme(tumor.size ~ year * treatment + age + who.PS,
                    data = colorectalLongi, random = ~1 + year | id)
# Survival model with frailtyPenal:
msurv <- frailtyPenal(Surv(time1, state) ~ age + treatment + who.PS + prev.resection,
                      data = colorectalSurv, n.knots = 7, kappa = 2)
msurv.weib <- frailtyPenal(Surv(time1, state) ~ age + treatment + who.PS + prev.resection,
                           data = colorectalSurv, hazard = "Weibull")


# Joint model with
# Baseline hazard function approximated with splines
# Random effects as the link function
model.spli.RE <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
data=colorectalSurv,data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Random-effects", left.censoring = -3.33,
n.knots = 7, kappa = 2, init.longi = mlongi, init.surv = msurv, init.Eta = c(0, 0))

# Joint model with
# Weibull baseline hazard function
# Current level of the biomarker as the link function
model.weib.CL <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS , timevar="year",
data=colorectalSurv, data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Current-level", left.censoring = -3.33, hazard = "Weibull",
init.longi = mlongi, init.surv = msurv.weib, init.Eta = 0)

# Joint model with user specified initial values
initL <- list(fixed = c(3, -0.3, -0.1, 0.2, -0.4, 0.2, 1, -0.6),
              random = matrix(c(2, -0.4, -0.4, 0.8), 2, 2),
              sigma = 1)
initS <- list(hazard = c(1.2, 1.5),
```

```
             fixed = c(-0.2, 0, 0.1, -0.2, 0.5, -0.3))

model.weib.CL.2 <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS , timevar="year",
data=colorectalSurv, data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Current-level", left.censoring = -3.33, hazard = "Weibull",
init.longi = initL, init.surv = initS, init.Eta = 0.3)

###--- Two-part Joint model for semicontinuous
#      longitudinal data and a terminal event ---###

data(colorectal)
data(colorectalLongi)
colorectalSurv <- subset(colorectal, new.lesions == 0)

# Box-cox back transformation (lambda=0.3) and apply logarithm (with a 1 unit shift)
colorectalLongi$Yo <- (colorectalLongi$tumor.size*0.3+1)^(1/0.3)
colorectalLongi$Y <- log(colorectalLongi$Y+1) # log transformation with shift=1

# Conditional two-part joint model - random-effects association structure (~15min)

CTPJM_re <-longiPenal(Surv(time1, state)~age + treatment +
who.PS+ prev.resection, Y~year*treatment, formula.Binary=Y~year*treatment,
data = colorectalSurv, data.Longi = colorectalLongi, random = c("1"),
random.Binary=c("1"), id = "id", link ="Random-effects", left.censoring = F,
n.knots = 7, kappa = 2, hazard="Splines-per")

print(CTPJM_re)

# Conditional two-part joint model - current-level association structure (~15min)
# Simulated dataset (github.com/DenisRustand/TPJM_sim)
data(longDat)
data(survDat)
tte <- frailtyPenal(Surv(deathTimes, d)~trt,n.knots=5,kappa=0, data=survDat,
cross.validation = T)
kap <- round(tte$kappa,2);kap # smoothing parameter
  CTPJM_cl <- longiPenal(Surv(deathTimes, d)~trt, Y~timej*trtY,
  data=survDat, data.Longi = longDat,
  random = c("1","timej"), formula.Binary=Y~timej*trtY,
  random.Binary=c("1"), timevar="timej", id = "id",
  link = "Current-level", n.knots = 5, kappa = kap,
  hazard="Splines-per", method.GH="Monte-carlo",
  n.nodes=500)

  print(CTPJM_cl)


# Marginal two-part joint model - random-effects association structure (~10min)
longDat$Yex <- exp(longDat$Y)-1
MTPJM_re <- longiPenal(Surv(deathTimes, d)~trt, Yex~timej*trtY,
                  data=survDat, data.Longi = longDat,MTP=T,GLMlog = T,
                  random = c("1"), formula.Binary=Y~timej*trtY,
                  random.Binary=c("1"), timevar="timej", id = "id",
```

```
                  link = "Random-effects", n.knots = 5, kappa = kap,
                  hazard="Splines-per", method.GH="Monte-carlo",
                  n.nodes=500)

print(MTPJM_re)

# Marginal two-part joint model - current-level association structure (~45min)
 MTPJM_cl <- longiPenal(Surv(deathTimes, d)~trt, Yex~timej*trtY,
                  data=survDat, data.Longi = longDat,MTP=T,GLMlog = T,
                  random = c("1","timej"), formula.Binary=Y~timej*trtY,
                  random.Binary=c("1"), timevar="timej", id = "id",
                  link = "Current-level", n.knots = 5, kappa = kap,
                  hazard="Splines-per", method.GH="Monte-carlo",
                  n.nodes=500)

print(MTPJM_cl)

###--- Mediation analysis
#Takes ~ 10 minutes to run
data(colorectal)
data(colorectalLongi)
colorectalSurv <- subset(colorectal, new.lesions == 0)

colorectalSurv$treatment<-sapply(colorectalSurv$treatment,function(t) ifelse(t=="S",1,0))
colorectalLongi$treatment<-sapply(colorectalLongi$treatment,function(t) ifelse(t=="S",1,0))

longi.col <- nlme::lme(tumor.size ~ age+year*treatment,
                       random = ~1 + year | id,
                       data = colorectalLongi)
surv.col <- frailtyPenal(Surv(time1, state) ~ age + treatment ,
                         n.knots = 7, kappa = 2,
                         data = colorectalSurv)

mod.col <- longiPenal(Surv(time1, state) ~ age+treatment,
            tumor.size ~ age+year*treatment,
            data=colorectalSurv, data.Longi = colorectalLongi, random = c("1", "year"),
          id = "id", link = "Current-level",timevar="year",method.GH = "Pseudo-adaptive",
            mediation = TRUE,med.trt = colorectalSurv$treatment,
            med.center = NULL,med.nmc = 50,n.knots = 7, kappa = 2,
            pte.ntimes = 30,pte.boot = T,pte.nmc = 1000,pte.nboot = 1000,
            init.longi = longi.col, init.surv = surv.col)

print(mod.col)
plot(mod.col,plot.mediation='All')

## End(Not run)
```

---

| | |
|---|---|
| loocv | *The trials leave-one-out crossvalidation for the one-step Joint surrogate model for evaluating a canditate surrogate endpoint.* |

---

## Description

The trials leave-one-out crossvalidation for evaluating the joint surrogate model

## Usage

```
loocv(object, unusedtrial, var.used = "error.estim", alpha. = 0.05,
dec = 3, print.times = TRUE)
```

## Arguments

| | |
|---|---|
| `object` | An object inheriting from `jointSurroPenal` class (output from calling the function `jointSurroPenal` or `jointSurroCopPenal`). |
| `unusedtrial` | A list of trial not to be taken into account in the cross-validation. This parameter is useful when after excluding some trials, the model is facing convergence problem. |
| `var.used` | This argument takes two values. The first one is `"error.estim"` and indicates if the prediction variance takes into account the estimation errors from the estimates of the parameters. If estimates are supposed to be known or if the dataset includes a high number of trials with a high number of subject per trial, value `"No.error"` can be used. The default is `error.estim`. |
| `alpha.` | The confidence level for the prediction interval. The default is `0.05` |
| `dec` | The desired number of digits after the decimal point for parameters and confidence intervals. Default of 3 digits is used. |
| `print.times` | a logical parameter to print estimation time. Default is TRUE. |

## Value

This function returns an object of class `jointSurroPenalloocv` containing:

| | |
|---|---|
| `result` | A dataframe including for each trial the number of included subjects, the observed treatment effect on the surrogate endpoint, the observed treatment effect on the true endpoint and the predicted treatment effect on the true enpoint with the associated prediction intervals. If the observed treatment effect on the true endpoint is included into the prediction interval, the last columns contains "*". |
| `ntrial` | The number of trials in the meta-analysis |
| `notconvtrial` | The vector of trials that have not converged |
| `pred.error` | The prediction error, corresponding to the number of cases where the prediction interval does not included the observed treatment effect on T |
| `different.models` | The list of the `G` models obtained after excuded for the `i-th` trial |
| `loocv.summary` | A dataframe of the estimates for the `G` models; each raw including the results without the subjects of the given trial |

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

### References

Burzykowski T, Buyse M (2006). "Surrogate threshold effect: an alternative measure for meta-analytic surrogate endpoint validation." Pharmaceutical Statistics, 5(3), 173-186.ISSN 1539-1612.

### See Also

`jointSurroPenal`, `jointSurroCopPenal`

### Examples

```
## Not run:
# Generation of data to use
 data.sim <- jointSurrSimul(n.obs=300, n.trial = 10,cens.adm=549.24,
             alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1, sigma.s = 0.7,
             sigma.t = 0.7, cor = 0.8, betas = -1.25, betat = -1.25,
             full.data = 0, random.generator = 1, seed = 0,
             nb.reject.data = 0)

###--- Joint surrogate model ---###

joint.surro.sim.MCGH <- jointSurroPenal(data = data.sim, int.method = 2,
                        nb.mc = 300, nb.gh = 20, print.iter = F)

# Example of loocv taking into accountn ony trial 2 trials (1 and 3)
dloocv <- loocv(joint.surro.sim.MCGH, unusedtrial = c(2,4:10))
dloocv$result
dloocv$loocv.summary

# In order to summarize all the estimated models during the loocv proccess:
dloocv$different.models


## End(Not run)
```

---

## multivPenal — *Fit a multivariate frailty model for two types of recurrent events and a terminal event.*

---

### Description

Fit a multivariate frailty model for two types of recurrent events with a terminal event using a penalized likelihood estimation on the hazard function or a parametric estimation. Right-censored data are allowed. Left-truncated data and stratified analysis are not possible. Multivariate frailty models allow studying, with a joint model, three survival dependent processes for two types of recurrent events and a terminal event. Multivariate joint frailty models are applicable in mainly two settings. First, when focus is on the terminal event and we wish to account for the effect of previous endogenous recurrent event. Second, when focus is on a recurrent event and we wish to correct for informative censoring.

The multivariate frailty model for two types of recurrent events with a terminal event is (in the calendar or time-to-event timescale):

$$\begin{cases} r_i^{(1)}(t|u_i,v_i) & = r_0^{(1)}(t)\exp(\beta_1' Z_i(t)+u_i) & \text{(rec. of type 1)} \\ r_i^{(2)}(t|u_i,v_i) & = r_0^{(2)}(t)\exp(\beta_2' Z_i(t)+v_i) & \text{(rec. of type 2)} \\ \lambda_i(t|u_i,v_i) & = \lambda_0(t)\exp(\beta_3' Z_i(t)+\alpha_1 u_i+\alpha_2 v_i) & \text{(death)} \end{cases}$$

where $r_0^{(l)}(t)$, $l \in \{1,2\}$ and $\lambda_0(t)$ are respectively the recurrent and terminal event baseline hazard functions, and $\beta_1, \beta_2, \beta_3$ the regression coefficient vectors associated with $Z_i(t)$ the covariate vector. The covariates could be different for the different event hazard functions and may be time-dependent. We consider that death stops new occurrences of recurrent events of any type, hence given $t > D$, $dN^{R(l)*}(t), l \in \{1,3\}$ takes the value 0. Thus, the terminal and the two recurrent event processes are not independent or even conditional upon frailties and covariates. We consider the hazard functions of recurrent events among individuals still alive. components in the above multivariate frailty model are linked together by two Gaussian and correlated random effects $u_i, v_i$: $(u_i, v_i)^T \sim \mathcal{N}(0, \Sigma_{uv})$, with

$$\Sigma_{uv} = \begin{pmatrix} \theta_1 & \rho\sqrt{\theta_1\theta_2} \\ \rho\sqrt{\theta_1\theta_2} & \theta_2 \end{pmatrix}$$

Dependencies between these three types of event are taken into account by two correlated random effects and parameters $\theta_1, \theta_2$ the variance of the random effects and $\alpha_1, \alpha_2$ the coefficients for these random effects into the terminal event part. If $\alpha_1$ and $\theta_1$ are both significantly different from 0, then the recurrent events of type 1 and death are significantly associated (the sign of the association is the sign of $\alpha_1$). If $\alpha_2$ and $\theta_2$ are both significantly different from 0, then the recurrent events of type 2 and death are significantly associated (the sign of the association is the sign of $\alpha_2$). If $\rho$, the correlation between the two random effects, is significantly different from 0, then the recurrent events of type 1 and the recurrent events of type 2 are significantly associated (the sign of the association is the sign of $\rho$).

## Usage

```
multivPenal(formula, formula.Event2, formula.terminalEvent, data, initialize
= TRUE, recurrentAG = FALSE, n.knots, kappa, maxit = 350, hazard =
"Splines", nb.int, print.times = TRUE)
```

## Arguments

`formula` a formula object, with the response for the first recurrent event on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. Interactions are possible using * or :.

`formula.Event2` a formula object, with the response for the second recurrent event on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. Interactions are possible using * or :.

`formula.terminalEvent`
: a formula object, with the response for the terminal event on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package.

`data`
: a 'data.frame' with the variables used in 'formula', 'formula.Event2' and 'formula.terminalEvent'.

`initialize`
: Logical value to initialize regression coefficients and baseline hazard functions parameters. When the estimation is semi-parametric with splines, this initialization produces also values for smoothing parameters (by cross validation). When initialization is requested, the program first fit two shared frailty models (for the two types of recurrent events) and a Cox proportional hazards model (for the terminal event). Default is TRUE.

`recurrentAG`
: Logical value. Is Andersen-Gill model fitted? If so indicates that recurrent event times with the counting process approach of Andersen and Gill is used. This formulation can be used for dealing with time-dependent covariates. The default is FALSE.

`n.knots`
: integer vector of length 3 (for the three outcomes) giving the number of knots to use. First is for the recurrent of type 1, second is for the recurrent of type 2 and third is for the terminal event hazard function. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. Number of knots must be between 4 and 20. (See Note)

`kappa`
: vector of length 3 (for the three outcomes) for positive smoothing parameters in the penalized likelihood estimation. First is for the recurrent of type 1, second is for the recurrent of type 2 and third is for the terminal event hazard function. The coefficient kappa of the integral of the squared second derivative of hazard function in the fit (penalized log likelihood). Initial values for the kappas can be obtained with the option "initialize=TRUE". We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them. Value required.(See Note)

`maxit`
: maximum number of iterations for the Marquardt algorithm. Default is 350.

`hazard`
: Type of hazard functions: "Splines" for semi-parametric hazard functions with the penalized likelihood estimation, "Piecewise-per" for piecewise constant hazard function using percentile, "Piecewise-equi" for piecewise constant hazard function using equidistant intervals, "Weibull" for parametric Weibull function. Default is "Splines".

`nb.int`
: An integer vector of length 3 (for the three outcomes). First is the Number of intervals (between 1 and 20) for the recurrent of type 1 parametric hazard functions ("Piecewise-per", "Piecewise-equi"). Second is the Number of intervals (between 1 and 20) for the recurrent of type 2 parametric hazard functions ("Piecewise-per", "Piecewise-equi"). Third is Number of intervals (between 1 and 20) for the death parametric hazard functions ("Piecewise-per", "Piecewise-equi")

`print.times`
: a logical parameter to print iteration process. Default is TRUE.

### Value

Parameters estimates of a multivariate joint frailty model, more generally a 'multivPenal' object. Methods defined for 'multivPenal' objects are provided for print, plot and summary. The following components are included in a 'multivPenal' object for multivariate Joint frailty models.

- `b` sequence of the corresponding estimation of the splines coefficients, the random effects variances, the coefficients of the frailties and the regression coefficients.
- `call` The code used for fitting the model.
- `n` the number of observations used in the fit.
- `groups` the number of subjects used in the fit.
- `n.events` the number of recurrent events of type 1 observed in the fit.
- `n.events2` the number of the recurrent events of type 2 observed in the fit.
- `n.deaths` the number of deaths observed in the fit.
- `loglikPenal` the complete marginal penalized log-likelihood in the semi-parametric case.
- `loglik` the marginal log-likelihood in the parametric case.
- `LCV` the approximated likelihood cross-validation criterion in the semi parametric case (with H minus the converged Hessian matrix, and l(.) the full log-likelihood.

$$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$

)
- `AIC` the Akaike information Criterion for the parametric case.

$$AIC = \frac{1}{n}(np - l(.))$$

- `theta1` variance of the frailty parameter for recurrences of type 1 $(\boldsymbol{Var}(u_i))$
- `theta2` variance of the frailty parameter for recurrences of type 2 $(\boldsymbol{Var}(v_i))$
- `alpha1` the coefficient associated with the frailty parameter $u_i$ in the terminal hazard function.
- `alpha2` the coefficient associated with the frailty parameter $v_i$ in the terminal hazard function.
- `rho` the correlation coefficient between $u_i$ and $v_i$
- `npar` number of parameters.
- `coef` the regression coefficients.
- `nvar` A vector with the number of covariates of each type of hazard function as components.
- `varH` the variance matrix of all parameters before positivity constraint transformation (theta, the regression coefficients and the spline coefficients). Then, the delta method is needed to obtain the estimated variance parameters.
- `varHIH` the robust estimation of the variance matrix of all parameters (theta, the regression coefficients and the spline coefficients).

formula
: the formula part of the code used for the model for the recurrent event.

formula.Event2
: the formula part of the code used for the model for the second recurrent event.

formula.terminalEvent
: the formula part of the code used for the model for the terminal event.

x1
: vector of times for hazard functions of the recurrent events of type 1 are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times.

lam1
: matrix of hazard estimates and confidence bands for recurrent events of type 1.

xSu1
: vector of times for the survival function of the recurrent event of type 1.

surv1
: matrix of baseline survival estimates and confidence bands for recurrent events of type 1.

x2
: vector of times for the recurrent event of type 2 (see x1 value).

lam2
: the same value as lam1 for the recurrent event of type 2.

xSu2
: vector of times for the survival function of the recurrent event of type 2

surv2
: the same value as surv1 for the recurrent event of type 2.

xEnd
: vector of times for the terminal event (see x1 value).

lamEnd
: the same value as lam1 for the terminal event.

xSuEnd
: vector of times for the survival function of the terminal event

survEnd
: the same value as surv1 for the terminal event.

median1
: The value of the median survival and its confidence bands for the recurrent event of type 1.

median2
: The value of the median survival and its confidence bands for the recurrent event of type 2.

medianEnd
: The value of the median survival and its confidence bands for the terminal event.

type.of.Piecewise
: Type of Piecewise hazard functions (1:"percentile", 0:"equidistant").

n.iter
: number of iterations needed to converge.

type.of.hazard
: Type of hazard functions (0:"Splines", "1:Piecewise", "2:Weibull").

n.knots
: a vector with number of knots for estimating the baseline functions.

kappa
: a vector with the smoothing parameters in the penalized likelihood estimation corresponding to each baseline function as components.

n.knots.temp
: initial value for the number of knots.

zi
: splines knots.

time
: knots for Piecewise hazard function for the recurrent event of type 1.

timedc
: knots for Piecewise hazard function for the terminal event.

time2
: knots for Piecewise hazard function for the recurrent event of type 2.

noVar
: indicator vector for recurrent, death and recurrent 2 explanatory variables.

nvarRec
: number of the recurrent of type 1 explanatory variables.

nvarEnd
: number of death explanatory variables.

nvarRec2 number of the recurrent of type 2 explanatory variables.

nbintervR Number of intervals (between 1 and 20) for the the recurrent of type 1 parametric hazard functions ("Piecewise-per", "Piecewise-equi").

nbintervDC Number of intervals (between 1 and 20) for the death parametric hazard functions ("Piecewise-per", "Piecewise-equi").

nbintervR2 Number of intervals (between 1 and 20) for the the recurrent of type 2 parametric hazard functions ("Piecewise-per", "Piecewise-equi").

istop Vector of the convergence criteria.

shape.weib shape parameters for the Weibull hazard function.

scale.weib scale parameters for the Weibull hazard function.

martingale.res martingale residuals for each cluster (recurrent of type 1).

martingale2.res martingale residuals for each cluster (recurrent of type 2).

martingaledeath.res martingale residuals for each cluster (death).

frailty.pred empirical Bayes prediction of the first frailty term.

frailty2.pred empirical Bayes prediction of the second frailty term.

frailty.var variance of the empirical Bayes prediction of the first frailty term.

frailty2.var variance of the empirical Bayes prediction of the second frailty term.

frailty.corr Correlation between the empirical Bayes prediction of the two frailty.

linear.pred linear predictor: uses Beta'X + ui in the multivariate frailty models.

linear2.pred linear predictor: uses Beta'X + vi in the multivariate frailty models.

lineardeath.pred linear predictor for the terminal part form the multivariate frailty models: Beta'X + alpha1 ui + alpha2 vi

global_chisq Recurrent event of type 1: a vector with the values of each multivariate Wald test.

dof_chisq Recurrent event of type 1: a vector with the degree of freedom for each multivariate Wald test.

global_chisq.test Recurrent event of type 1: a binary variable equals to 0 when no multivariate Wald is given, 1 otherwise.

p.global_chisq Recurrent event of type 1: a vector with the p-values for each global multivariate Wald test.

names.factor Recurrent event of type 1: Names of the "as.factor" variables.

global_chisq2 Recurrent event of type 2: a vector with the values of each multivariate Wald test.

dof_chisq2 Recurrent event of type 2: a vector with the degree of freedom for each multivariate Wald test.

global_chisq.test2 Recurrent event of type 2: a binary variable equals to 0 when no multivariate Wald is given, 1 otherwise.

p.global_chisq2
: Recurrent event of type 2: a vector with the p_values for each global multivariate Wald test.

names.factor2
: Recurrent event of type 2: Names of the "as.factor" variables.

global_chisq_d
: Terminal event: a vector with the values of each multivariate Wald test.

dof_chisq_d
: Terminal event: a vector with the degree of freedom for each multivariate Wald test.

global_chisq.test_d
: Terminal event: a binary variable equals to 0 when no multivariate Wald is given, 1 otherwise.

p.global_chisq_d
: Terminal event: a vector with the p-values for each global multivariate Wald test.

names.factordc
: Terminal event: Names of the "as.factor" variables.

## Note

"kappa" (kappa[1], kappa[2] and kappa[3]) and "n.knots" (n.knots[1], n.knots[2] and n.knots[3]) are the arguments that the user has to change if the fitted model does not converge. "n.knots" takes integer values between 4 and 20. But with n.knots=20, the model will take a long time to converge. So, usually, begin first with n.knots=7, and increase it step by step until it converges. "kappa" only takes positive values. So, choose a value for kappa (for instance 10000), and if it does not converge, multiply or divide this value by 10 or 5 until it converges. Moreover, it may be useful to change the value of the initialize argument.

## References

Mazroui Y., Mathoulin-Pellissier S., MacGrogan G., Brouste V., Rondeau V. (2013). Multivariate frailty models for two types of recurrent events with an informative terminal event : Application to breast cancer data. *Biometrical journal*, **55(6)**, 866-884.

## See Also

terminal,event2, print.multivPenal,summary.multivPenal,plot.multivPenal

## Examples

```
###--- Multivariate Frailty model ---###

data(dataMultiv)

# (computation takes around 60 minutes)
modMultiv.spli <- multivPenal(Surv(TIMEGAP,INDICREC)~cluster(PATIENT)+v1+v2+
             event2(INDICMETA)+terminal(INDICDEATH),formula.Event2=~v1+v2+v3,
             formula.terminalEvent=~v1,data=dataMultiv,n.knots=c(8,8,8),
             kappa=c(1,1,1),initialize=FALSE)
```

```
print(modMultiv.spli)

modMultiv.weib <- multivPenal(Surv(TIMEGAP,INDICREC)~cluster(PATIENT)+v1+v2+
             event2(INDICMETA)+terminal(INDICDEATH),formula.Event2=~v1+v2+v3,
             formula.terminalEvent=~v1,data=dataMultiv,hazard="Weibull")

print(modMultiv.weib)

modMultiv.cpm <- multivPenal(Surv(TIMEGAP,INDICREC)~cluster(PATIENT)+v1+v2+
             event2(INDICMETA)+terminal(INDICDEATH),formula.Event2=~v1+v2+v3,
             formula.terminalEvent=~v1,data=dataMultiv,hazard="Piecewise-per",
             nb.int=c(6,6,6))

print(modMultiv.cpm)
```

---

num.id *Identify individuals in Joint model for clustered data*

---

## Description

This is a special function used in addition to the `cluster()` function in the context of survival joint models for clustered data. This function identifies subject index. It is used on the right hand side of a 'frailtyPenal' formula. Using `num.id()` in a formula implies that a joint frailty model for clustered data is fitted (Rondeau et al. 2011).

## Usage

```
num.id(x)
```

## Arguments

| | |
|---|---|
| x | A character or numeric variable which is supposed to indicate the variable identifying individuals |

## Value

No return value

## References

V. Rondeau, J.P. Pignon, S. Michiels (2011). A joint model for the dependence between clustered times to tumour progression and deaths: A meta-analysis of chemotherapy in head and neck cancer. *Statistical methods in medical research* **897**, 1-19.

### See Also

frailtyPenal

### Examples

```
data(readmission)
#-- here is generated cluster (5 clusters)
readmission <- transform(readmission,group=id%%5+1)

#-- exclusion all recurrent events --#
#--  to obtain framework of semi-competing risks --#
readmission2 <- subset(readmission, (t.start == 0 & event == 1) | event == 0)

joi.clus.gap <- frailtyPenal(Surv(time,event)~cluster(group)+
num.id(id)+dukes+charlson+sex+chemo+terminal(death),
formula.terminalEvent=~dukes+charlson+sex+chemo,
data=readmission2,recurrentAG=FALSE, n.knots=8,
kappa=c(1.e+10,1.e+10) ,Alpha="None")
```

---

## Paq810 *Filtered Paquid Sample Data Set for Illness-Death Modeling*

---

### Description

A dataset derived from Paq1000 (originally a sample from the Paquid study available via the SmoothHazard package). This version excludes subjects where the age of dementia diagnosis or censoring ('r') was exactly equal to the age at study entry ('e'), ensuring valid observation intervals when using left truncation.

### Usage

```
data(Paq810)
```

### Format

A data frame with approximately 810 rows (original 1000 minus exclusions) and the following 8 columns:

**dementia** Dementia status indicator: 0 = non-demented, 1 = demented.

**death** Death status indicator: 0 = alive, 1 = dead.

**e** Age at entry into the study (left-truncation time).

**l** For demented subjects: age at the visit *before* the diagnostic visit. For non-demented subjects: age at the last visit.

**r** Age at dementia diagnosis or censoring for dementia (event/censoring time for 0->1 transition). Guaranteed to be > ‘e‘.

**t** Overall exit age. For dead subjects: age at death. For alive subjects: age at the latest news (censoring time for death).

**certif** Primary school certificate indicator: 0 = without certificate, 1 = with certificate.

**gender** Gender indicator: 0 = female, 1 = male.

## Details

This dataset was created by filtering the `Paq1000` data: `Paq810 <- Paq1000[Paq1000$r > Paq1000$e, ]`. This step is necessary to prevent issues with `survival::Surv(e, r, dementia)` which requires the stop time (’r’) to be strictly greater than the start/entry time (’e’).

The time variables ‘e‘, ‘l‘, ‘r‘, and ‘t‘ are all ages in years. This dataset is suitable for fitting illness-death models with left truncation using functions like `IllnessDeath`.

## Source

Derived from the `Paq1000` dataset, which originates from the Paquid study and is included in the `SmoothHazard` package.

## See Also

`Paq1000`, The `SmoothHazard` package.

---

`plot.additivePenal` *Plot Method for an Additive frailty model.*

---

## Description

Plots estimated baseline survival and hazard functions (output from an object of class’additivePenal’ object for additive frailty model ). Confidence bands are allowed.

## Usage

```
## S3 method for class 'additivePenal'
plot(x, type.plot="Hazard", conf.bands=TRUE,
pos.legend="topright", cex.legend=0.7, main, color=2, median=TRUE, Xlab = "Time", Ylab =
"Hazard function", ...)
```

## Arguments

| | |
|---|---|
| `x` | An object of a fitted additive frailty model (output from calling `additivePenal`). |
| `type.plot` | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first words are required, e.g "Haz", "Su" |
| `conf.bands` | logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| `pos.legend` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| `cex.legend` | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| `main` | plot title |
| `color` | curve color (integer) |
| `median` | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"Hazard function"' |
| `...` | Other graphical parameters like those in `plot.frailtyPenal` |

## Value

Print a plot of the baseline survival or hazard functions with the confidence bands or not (conf.bands argument)

## See Also

`additivePenal`

## Examples

```
data(dataAdditive)

modAdd <- additivePenal(Surv(t1,t2,event)~cluster(group)+var1+slope(var1),
correlation=TRUE,data=dataAdditive,n.knots=8,kappa=862,hazard="Splines")

#-- 'var1' is boolean as a treatment variable

plot(modAdd)
```

---

`plot.Diffepoce` *Plot difference of EPOCE estimators between two joint frailty models.*

---

## Description

Plots values of the difference of two Cross-Validated Prognosis Observed Loss (CVPOL) computed with two joint frailty models. Confidence intervals are allowed.

## Usage

```
## S3 method for class 'Diffepoce'
plot(x, conf.bands=TRUE, Xlab = "Time", Ylab =
"EPOCE difference" , ...)
```

## Arguments

| | |
|---|---|
| `x` | An object inheriting from `Diffepoce` class. |
| `conf.bands` | Logical value. Determines whether confidence intervals will be plotted. The default is FALSE. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"EPOCE difference"' |
| `...` | Other unused arguments. |

## Value

Print one plot with one curve and its confidence interval.

## See Also

`Diffepoce`

---

`plot.epoce` *Plot values of estimators of the Expected Prognostic Observed Cross-Entropy (EPOCE).*

---

## Description

Plots values of estimators MPOL and CVPOL for evaluating EPOCE. No confidence interval.

## Usage

```
## S3 method for class 'epoce'
plot(x, type, pos.legend="topright", cex.legend=0.7,
Xlab="Time",Ylab="Epoce", ...)
```

### Arguments

| | |
|---|---|
| `x` | An object inheriting from epoce class |
| `type` | Type of estimator to plot. If new dataset was used only mpol can be plotted (`"mpol"`), otherwise mpol and cvpol can be plotted (`"mpol"` and `"cvpol"`, default is `"cvpol"`). |
| `pos.legend` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"'. |
| `cex.legend` | size of the legend. Default is 0.7. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"Epoce"' |
| `...` | Other unused arguments. |

### Value

Print a curve of the estimator of EPOCE using time points defined in `epoce`.

### See Also

`epoce`

---

`plot.frailtyCmprsk` *Plot Method for a Weibull competing risks model with optional shared frailty between transitions.*

---

### Description

Plots estimated baseline survival and hazard functions from an object of class 'frailtyCmprsk'. Confidence bands are allowed.

### Usage

```
## S3 method for class 'frailtyCmprsk'
plot(x, type.plot="Baseline hazard", events, conf.bands=TRUE,
pos.legend="topright", cex.legend=0.7, lwd=c(1,1,1), color=2, median=TRUE,
Xlab = "Time", Ylab = "Baseline hazard function",...)
```

### Arguments

| | |
|---|---|
| `x` | A Weibull competing risks model, i.e. an `frailtyCmprsk` class object (output from calling `frailtyCmprsk` function). |
| `type.plot` | a character string specifying the type of curve. Possible value are "Baseline hazard", or "Baseline survival". The default is "Baseline hazard". Only the first letters are required, e.g "Haz", "Su". |

| | |
|---|---|
| events | Integer vector specifying which competing events to display in the plots. Use 1 for the first event, 2 for the second, and so on. If not specified, plots for all events are shown. |
| conf.bands | Logical value. Determines whether confidence bands will be plotted. The default is TRUE. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.5 |
| lwd | A vector of length 3 of positive values to specify the line width of the plots and their confidence bands. If no confidence bands are plotted, the last two elements of the vector are ignored. Default is (1,1,1). |
| color | color of the curve (integer) |
| median | Logical value. Determines whether median survival time will be plotted. Default is TRUE. |
| Xlab | Label of x-axis. Default is '"Time"' |
| Ylab | Label of y-axis. Default is '"Baseline hazard function"' |
| ... | other unused arguments |

## Details

Plot Method for a Weibull competing risks model with optional shared frailty between transitions.

## Value

Print a plot of a Weibull competing risks model with optional shared frailty between transitions.

## See Also

frailtyCmprsk

## Examples

```
###--- Simple Weibull competing risks model ---###

data(CPRSKbmtcrr)

ModCmprsk_NoCov_Factor <- frailtyCmprsk(
  formulas = list(
    Surv(observed_time, Status, type = "mstate") ~ D,
    ~ 1
  ),
  data = CPRSKbmtcrr,
  print.info = FALSE,
  maxit = 100
)
```

```
plot(ModCmprsk_NoCov_Factor)

#-- No confidence bands
plot(ModCmprsk_NoCov_Factor, conf.bands = FALSE)
```

---

plot.frailtyIllnessDeath

*Plot Method for a Weibull Illness-Death model with optional shared frailty between transitions.*

---

## Description

Plots estimated baseline survival and hazard functions from an object of class 'frailtyIllnessDeath'. Confidence bands are allowed.

## Usage

```
## S3 method for class 'frailtyIllnessDeath'
plot(x, type.plot = "Baseline hazard", transition,
   conf.bands=TRUE, pos.legend = "topright", cex.legend=0.7, lwd=c(1,1,1),
   color=2, median=TRUE, Xlab = "Time", Ylab= "Baseline hazard function",...)
```

## Arguments

| | |
|---|---|
| x | A Weibull Illness-Death model, i.e. an `IllnessDeath` class object (output from calling `IllnessDeath` function). |
| type.plot | a character string specifying the type of curve. Possible value are "Baseline hazard", or "Baseline survival". The default is "Baseline hazard". Only the first letters are required, e.g "Haz", "Su". |
| transition | Argument to specify if only the plot of one of the three transitions is wanted. If not specified, the plots for all transitions are displayed. Possible values are "01", "02" or "12". |
| conf.bands | Logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"'. |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.7. |
| lwd | A vector of length 3 of positive values to specify the line width of the plots and their confidence bands. If no confidence bands are plotted, the last two elements of the vector are ignored. Default is (1,1,1). |
| color | color of the curve (integer). |

| | |
|---|---|
| `median` | Logical value. Determines whether median survival time will be plotted. Default is TRUE. |
| `Xlab` | Label of x-axis. Default is '"Time"'. |
| `Ylab` | Label of y-axis. Default is '"Baseline hazard function"'. |
| `...` | other unused arguments |

## Value

Print a plot of a Weibull Illness-Death model with optional shared frailty between transitions.

## See Also

`frailtyIllnessDeath`

## Examples

```
###--- Semi-Markovian Weibull Illness-Death model with left truncation ---###

data(Paq810)

ModIllnessDeath_SemiMarkov_LeftTrunc <- frailtyIllnessDeath(
  formula = Surv(e, r, dementia) ~ gender + certif,
  formula.terminalEvent = Surv(t, death) ~ gender + certif,
  data = Paq810,
  print.info = FALSE,
  maxit = 100
)

plot(ModIllnessDeath_SemiMarkov_LeftTrunc)

#-- No confidence bands
plot(ModIllnessDeath_SemiMarkov_LeftTrunc, conf.bands = FALSE)
```

---

`plot.frailtyPenal` *Plot Method for a Shared frailty model.*

---

## Description

Plots estimated baseline survival and hazard functions from an object of class 'frailtyPenal'. Confidence bands are allowed.

### Usage

```
## S3 method for class 'frailtyPenal'
plot(x, type.plot = "Hazard", conf.bands=TRUE,
pos.legend = "topright", cex.legend=0.7, main, color=2, median=TRUE, Xlab = "Time", Ylab
= "Hazard function", ...)
```

### Arguments

| | |
|---|---|
| `x` | A shared frailty model, i.e. a `frailtyPenal` class object (output from calling `frailtyPenal` function). |
| `type.plot` | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first letters are required, e.g "Haz", "Su" |
| `conf.bands` | Logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| `pos.legend` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| `cex.legend` | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| `main` | title of plot |
| `color` | color of the curve (integer) |
| `median` | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"Hazard function"' |
| `...` | other unused arguments |

### Value

Print a plot of a shared frailty model.

### See Also

`frailtyPenal`

### Examples

```
## Not run:

data(readmission)

###--- Shared frailty model ---###

modSha <- frailtyPenal(Surv(time,event)~as.factor(dukes)+cluster(id),
n.knots=10,kappa=10000,data=readmission,hazard="Splines")
```

```
plot(modSha,type="Survival",conf=FALSE)

###--- Cox proportional hazard model ---###

modCox <- frailtyPenal(Surv(time,event)~as.factor(dukes),n.knots=10,
kappa=10000,data=readmission,hazard="Splines")

plot(modCox)

#-- no confidence bands
plot(modSha,conf.bands=FALSE)
plot(modCox,conf.bands=FALSE)


## End(Not run)
```

---

`plot.jointNestedPenal` *Plot method for a joint nested frailty model.*

---

## Description

Plots estimated baseline survival and hazard functions of a joint nested frailty model (output from an object of class 'jointNestedPenal' for joint nested frailty models) for each type of event (terminal or recurrent). Confidence bands are allowed.

## Usage

```
## S3 method for class 'jointNestedPenal'
plot(x, event = "Both", type.plot = "Hazard",
conf.bands = FALSE, pos.legend="topright", cex.legend = 0.7, ylim, main,
color = 2, median=TRUE, Xlab = "Time", Ylab = "Hazard function", ...)
```

## Arguments

| | |
|---|---|
| `x` | A joint nested model, i.e. an object of class `jointNestedPenal` for joint nested frailty model (output from calling `frailtyPenal` function). |
| `event` | a character string specifying the type of curve. Possible value are "Terminal", "Recurrent", or "Both". The default is "Both". |
| `type.plot` | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first letters are required, e.g "Haz", "Su" |
| `conf.bands` | logical value. Determines whether confidence bands will be plotted. The default is to do so. |

| | |
|---|---|
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| ylim | y-axis limits |
| main | plot title |
| color | curve color (integer) |
| median | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| Xlab | Label of x-axis. Default is '"Time"' |
| Ylab | Label of y-axis. Default is '"Hazard function"' |
| ... | other unused arguments |

## Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence bands or not (conf.bands argument)

## See Also

frailtyPenal

## Examples

```
## Not run:

#-- here is generated cluster (30 clusters)
readmissionNested <- transform(readmission,group=id%%30+1)

# Baseline hazard function approximated with splines with calendar-timescale

model.spli.AG <- frailtyPenal(formula = Surv(t.start, t.stop, event)
~ subcluster(id) + cluster(group) + dukes + terminal(death),
formula.terminalEvent = ~dukes, data = readmissionNested, recurrentAG = TRUE,
 n.knots = 8, kappa = c(9.55e+9, 1.41e+12),initialize = TRUE)

# Plot the estimated baseline hazard function with the confidence intervals
plot(model.spli.AG)

# Plot the estimated baseline hazard function with the confidence intervals
plot(model.spli.RE, type = "Survival")


## End(Not run)
```

plot.jointPenal *Plot Method for a Joint frailty model.*

## Description

Plots estimated baseline survival and hazard functions of a joint frailty model (output from an object of class 'JointPenal' for joint frailty models ) for each type of event (terminal or recurrent). Confidence bands are allowed.

## Usage

```
## S3 method for class 'jointPenal'
plot(x, event = "Both", type.plot = "Hazard", conf.bands
= FALSE, pos.legend="topright", cex.legend = 0.7, ylim, main, color = 2, median=TRUE,
Xlab = "Time", Ylab = "Hazard function", ...)
```

## Arguments

| | |
|---|---|
| x | A joint model, i.e. an object of class `frailtyPenal` for Joint frailty model (output from calling `frailtyPenal` function). |
| event | a character string specifying the type of curve. Possible value are "Terminal", "Recurrent", or "Both". The default is "Both". |
| type.plot | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first letters are required, e.g "Haz", "Su" |
| conf.bands | logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| ylim | y-axis limits |
| main | plot title |
| color | curve color (integer) |
| median | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| Xlab | Label of x-axis. Default is '"Time"' |
| Ylab | Label of y-axis. Default is '"Hazard function"' |
| ... | other unused arguments |

## Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence bands or not (conf.bands argument)

## See Also

frailtyPenal

## Examples

```
## Not run:

data(readmission)

#-- Gap-time
modJoint.gap <- frailtyPenal(Surv(time,event)~cluster(id)+sex+dukes+
charlson+terminal(death),formula.terminalEvent=~sex+dukes+charlson,
data=readmission,n.knots=14,kappa=c(100,100))

#-- It takes around 1 minute to converge --#

plot(modJoint.gap,type.plot="Haz",event="recurrent",conf.bands=TRUE)
plot(modJoint.gap,type.plot="Haz",event="terminal",conf.bands=TRUE)
plot(modJoint.gap,type.plot="Haz",event="both",conf.bands=TRUE)

plot(modJoint.gap,type.plot="Su",event="recurrent",conf.bands=TRUE)
plot(modJoint.gap,type.plot="Su",event="terminal",conf.bands=TRUE)
plot(modJoint.gap,type.plot="Su",event="both",conf.bands=TRUE)



## End(Not run)
```

---

plot.jointRecCompet *Plot Method for a joint competing risk model with one recurrent event and two terminal events.*

---

## Description

Plots of estimated baseline survival and hazard functions of joint competing recurrent model (output from an object of class 'jointRecCompet') for each type of event (recurrent and the two terminal events). Confidence intervals are allowed.

## Usage

```
## S3 method for class 'jointRecCompet'
plot(x, event = "All", type.plot = "Hazard",
conf.bands = FALSE, pos.legend = "topright", cex.legend = 0.7, ylim, main,
color1="red", color2="blue", colorEnd="green", median=TRUE, Xlab = "Time",
Ylab = "Hazard function", ...)
```

## Arguments

| | |
|---|---|
| x | A joint competing risk model, i.e. an object of class jointRecCompet (output from calling jointRecCompet function). |
| event | a character string specifying the type of outcome. Possible value are "Recurrent", "Terminal1", "Terminal2", or "All". The default is "All". |
| type.plot | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first words are required, e.g "Haz", "Su" |
| conf.bands | logical value. Determines whether confidence intervals will be plotted. The default is to do so. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| ylim | y-axis limits |
| main | plot title |
| color1 | curve color for recurrent event of type 1 (integer or color name in quotation marks) |
| color2 | curve color for recurrent event of type 2 (integer or color name in quotation marks) |
| colorEnd | curve color for terminal event (integer or color name in quotation marks) |
| median | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| Xlab | Label of x-axis. Default is '"Time"' |
| Ylab | Label of y-axis. Default is '"Hazard function"' |
| ... | Other graphical parameters |

## Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence intervals or not (conf.bands argument)

## See Also

jointRecCompet

## plot.jointSurroMed *Plot Method for a joint surrogate mediation analysis model.*

### Description

Plots the estimated functions associated with the mediation analysis, i.e. $g(s)$, $PTE(t)$ as well as the natural direct, indirect and total effects. An option to plot the confidence bands of the function $g(s)$ is available. This option is also implemented for the confidence bands of the functions $PTE(t)$ and of the natural effects if these confidence bands are available.

### Usage

```
## S3 method for class 'jointSurroMed'
plot(x,plot.mediation="All",type.plot="Hazard",
conf.bands=TRUE,endpoint=2,
legend.pos = "topleft",...)
```

### Arguments

| | |
|---|---|
| `x` | An object of class `jointSurroMed` from a joint surrogate model with a mediation analysis for longitudinal outcome and a terminal event, i.e., an output from calling `jointSurroPenal` function with the option 'mediation' set to TRUE. |
| `plot.mediation` | A character string specifying the desired plot. Possible values are "All", "g","PTE" or "Effects". The default is "All" which displays all three plots. |
| `type.plot` | A character string specifying the type of curve for the baseline hazards functions. Possible value are "Hazard", or "Survival". |
| `conf.bands` | Logical value. Determines whether confidence bands should be plotted. The default is to do so if the confidence bands are available. |
| `endpoint` | An integer specifying for which endpoint should the baseline curves be plotted. Possible values are 0 for the surrogate endpoint only and 1 for the final endpoint or 2 for both. Default is 2. |
| `legend.pos` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topleft"' |
| `...` | other unused arguments. |

### Value

Print one or several plots for the mediation analysis of a joint surrogate model

### See Also

`jointSurroPenal`

`plot.jointSurroPenal` *Plot Method for the one-step Joint surrogate model for the evaluation of a canditate surrogate endpoint.*

## Description

Plots estimated baseline survival and hazard functions for the surrogate endpoint and the true endpoint from an object of class 'jointSurroPenal'. Confidence bands are allowed.

## Usage

```
## S3 method for class 'jointSurroPenal'
plot(x, type.plot = "Hazard", conf.bands=TRUE,
pos.legend = "topright", cex.legend=0.7, main, Xlab = "Time",
Ylab = "Baseline hazard function", median = TRUE, xmin = 0, xmax = NULL,
ylim = c(0,1), endpoint = 2, scale = 1, ...)
```

## Arguments

| | |
|---|---|
| `x` | An object inheriting from `jointSurroPenal` class (output from calling the function `jointSurroPenal` ). |
| `type.plot` | A character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first letters are required, e.g "Haz", "Su". |
| `conf.bands` | Logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| `pos.legend` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"'. |
| `cex.legend` | Character expansion factor *relative* to current 'par("cex")'. Default is 0.7. |
| `main` | Title of plot. |
| `Xlab` | Label of x-axis. Default is '"Time"'. |
| `Ylab` | Label of y-axis. Default is '"Baseline hazard function"'. |
| `median` | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| `xmin` | Minimum value for x-axis, the default is `0`. |
| `xmax` | Maximum value for x-axis, the default is `NULL`. |
| `ylim` | Range of y-axis. Default is from 0 to 1. |
| `endpoint` | A binary that indicates the endpoint to represent. `0` for the surrogate endpoint, `1` for the true endpoint, and `2` for both surrogate endpoint and true endpoint. The default is `2`. |
| `scale` | A numeric that allows to rescale (by multiplication) the survival times. If no change is need the argument is set to 1, the default value. eg: 1/365 aims to convert days to years . |
| `...` | other unused arguments. |

## Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence bands or not (conf.bands argument)

## See Also

jointSurroPenal, jointSurroCopPenal

## Examples

```
## Not run:


###--- Joint surrogate model ---###
###---evaluation of surrogate endpoints---###

data(dataOvarian)
joint.surro.ovar <- jointSurroPenal(data = dataOvarian, n.knots = 8,
                init.kappa = c(2000,1000), indicator.alpha = 0,
                nb.mc = 200, scale = 1/365)

# Baseline Hazards fonctions for both the surrogate endpoint
# and the true endpoint
plot(joint.surro.ovar,endpoint = 2,type.plot = "Haz", conf.bands = T)

# Baseline survival fonctions for both the surrogate endpoint
# and the true endpoint
plot(joint.surro.ovar,endpoint = 2,type.plot = "Su", conf.bands = T)




## End(Not run)
```

---

# plot.jointSurroPenalloocv

*Plot of trials leave-one-out crossvalidation Outputs from the one-step Joint surrogate model for evaluating a canditate surrogate endpoint.*

---

## Description

Plot of trials leave-one-out crossvalidation Outputs for evaluating the joint surrogate model

### Usage

```
## S3 method for class 'jointSurroPenalloocv'
plot(x, unusedtrial = NULL, xleg = "bottomleft",
yleg = NULL, main = NULL, xlab = "Trials",
ylab = "Log Hazard ratio of the true endpoint",
legend = c("Beta observed", "Beta predict"), ...)
```

### Arguments

| | |
|---|---|
| `x` | An object inherent from the `jointSurroPenalloocv` Class |
| `unusedtrial` | Vector of unconsidered trials, may be due to the fact that the predicted treatment effects on true endpoint have an outlier. In this case, one can drop from the data the trials with very hight absolute predicted value |
| `xleg` | X-coordinate for the location of the legend. |
| `yleg` | Y-coordinate for the location of the legend, the default is `NULL` |
| `main` | An overall title for the plot: see title. |
| `xlab` | A title for the x axis: see title. |
| `ylab` | A title for the y axis: see title. |
| `legend` | A vector of characters string of length >= 1 to appear in the legend |
| `...` | other unused arguments. |

### Value

This function displays the boxplots corresponding to the number of trials in the dataset. Each boxplot includes 3 elements corresponding to the predicted treatment effect on true endpoint with the prediction interval. The circles inside or outside the boxplot represent the observed treatment effects on true endpoint. For each trial with convergence issues or outliers, the boxplot is replaced by a dash. In this case, we display in the title of the figure a vector of these trials, if argument `main` is set to `NULL`. The function returns the list of unused trials.

### Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

### References

Burzykowski T, Buyse M (2006). "Surrogate threshold effect: an alternative measure for meta-analytic surrogate endpoint validation." Pharmaceutical Statistics, 5(3), 173-186.ISSN 1539-1612.

### See Also

`loocv`

## Examples

```
## Not run:
# Generation of data to use
 data.sim <- jointSurrSimul(n.obs=300, n.trial = 10,cens.adm=549.24,
             alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1, sigma.s = 0.7,
             sigma.t = 0.7, cor = 0.8, betas = -1.25, betat = -1.25,
             full.data = 0, random.generator = 1, seed = 0,
             nb.reject.data = 0)

###--- Joint surrogate model ---###

joint.surro.sim.MCGH <- jointSurroPenal(data = data.sim, int.method = 2,
                        nb.mc = 300, nb.gh = 20, print.iter = T)

# Example of loocv taking into accountn ony trial 2 trials (1 and 3)
dloocv <- loocv(joint.surro.sim.MCGH, unusedtrial = c(2,4:10))

plot(x = dloocv, xleg = "topright", bty = "n")


## End(Not run)
```

---

`plot.longiPenal` *Plot Method for a joint model for longitudinal data and a terminal event.*

---

## Description

Plots estimated baseline survival and hazard functions for a terminal outcome from an object of class 'longiPenal'. If available, plot the estimated quantities related to a mediation analysis. Confidence bands are allowed.

## Usage

```
## S3 method for class 'longiPenal'
plot(x, type.plot = "Hazard",plot.mediation="All",
conf.bands=TRUE,pos.legend= "topright", cex.legend=0.7, main, color,
median=TRUE, Xlab = "Time", Ylab = "Hazard function", ...)
```

## Arguments

| | |
|---|---|
| `x` | A joint model for longitudinal outcome and a terminal event, i.e. a `longiPenal` class object (output from calling `longiPenal` function). |
| `type.plot` | a character string specifying the type of curve for the terminal event. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first words are required, e.g "Haz", "Su" |

| | |
|---|---|
| `plot.mediation` | A character string specifying the desired plot. Possible values are "All", "PTE" or "Effects". The default is "All" which displays both plots. |
| `conf.bands` | Logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| `pos.legend` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| `cex.legend` | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| `main` | title of plot |
| `color` | color of the curve (integer) |
| `median` | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"Hazard function"' |
| `...` | other unused arguments |

## Value

Print a plot for the terminal event of the joint model for a longitudinal and survival data.

## See Also

longiPenal

## Examples

```
## Not run:
###--- Joint model for longitudinal data and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

# Baseline hazard function approximated with splines
# Random effects as the link function

model.spli.RE <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
colorectalSurv,data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Random-effects", left.censoring = -3.33,
n.knots = 7, kappa = 2)
pdf(file = "/home/agareb1/etudiants/al10/newpack/test/plot_longi.pdf")

# Plot the estimated baseline hazard function with the confidence intervals
plot(model.spli.RE)
```

```
# Plot the estimated baseline hazard function with the confidence intervals
plot(model.spli.RE, type = "Survival")

## End(Not run)
```

---

plot.multivPenal *Plot Method for a multivariate frailty model.*

---

## Description

Plots of estimated baseline survival and hazard functions of a multivariate frailty model (output from an object of class 'multivPenal' for multivariate frailty models ) for each type of event (recurrent, terminal and second recurrent). Confidence intervals are allowed.

## Usage

```
## S3 method for class 'multivPenal'
plot(x, event = "Both", type.plot = "Hazard",
conf.bands = FALSE, pos.legend = "topright", cex.legend = 0.7, ylim, main,
color1="red", color2="blue", colorEnd="green", median=TRUE, Xlab = "Time",
Ylab = "Hazard function", ...)
```

## Arguments

| | |
|---|---|
| x | A joint multivariate model, i.e. an object of class `multivPenal` (output from calling `multivPenal` function). |
| event | a character string specifying the type of outcome. Possible value are "Terminal", "Recurrent", "Recurrent2", or "Both". The default is "Both". |
| type.plot | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first words are required, e.g "Haz", "Su" |
| conf.bands | logical value. Determines whether confidence intervals will be plotted. The default is to do so. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| ylim | y-axis limits |
| main | plot title |
| color1 | curve color for recurrent event of type 1 (integer or color name in quotation marks) |

| | |
|---|---|
| color2 | curve color for recurrent event of type 2 (integer or color name in quotation marks) |
| colorEnd | curve color for terminal event (integer or color name in quotation marks) |
| median | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| Xlab | Label of x-axis. Default is '"Time"' |
| Ylab | Label of y-axis. Default is '"Hazard function"' |
| ... | Other graphical parameters |

## Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence intervals or not (conf.bands argument)

## See Also

multivPenal

---

plot.nestedPenal *Plot Method for a Nested frailty model.*

---

## Description

Plots estimated baseline survival and hazard functions (output from an object of class 'NestedPenal' for nested frailty models). Confidence bands are allowed.

## Usage

```
## S3 method for class 'nestedPenal'
plot(x, type.plot="Hazard", conf.bands=TRUE,
pos.legend="topright", cex.legend=0.7, main, color=2, median=TRUE, Xlab = "Time", Ylab =
"Hazard function", ...)
```

## Arguments

| | |
|---|---|
| x | A nested model, i.e. an object of class `frailtyPenal` for Nested frailty models (output from calling `frailtyPenal` function). |
| type.plot | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first words are required, e.g "Haz", "Su" |
| conf.bands | logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |

| | |
|---|---|
| `cex.legend` | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| `main` | plot title |
| `color` | curve color (integer) |
| `median` | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"Hazard function"' |
| `...` | Other graphical parameters like those in `plot.frailtyPenal` |

## Value

Print a plot of the baseline survival or hazard functions with the confidence bands or not (conf.bands argument)

## See Also

`frailtyPenal`

## Examples

```
## Not run:

data(dataNested)
modNested <- frailtyPenal(Surv(t1,t2,event)~cluster(group)+
subcluster(subgroup)+cov1+cov2,data=dataNested,n.knots=8,
kappa=50000,hazard="Splines")

plot(modNested,conf.bands=FALSE)


## End(Not run)
```

---

`plot.predFrailty` *Plot predictions using a Cox or a shared frailty model.*

---

## Description

Plots predicted probabilities of event. Confidence intervals are allowed.

## Usage

```
## S3 method for class 'predFrailty'
plot(x, conf.bands=FALSE, pos.legend="topright",
cex.legend=0.7, ylim=c(0,1), Xlab = "Time t", Ylab, ...)
```

### Arguments

| | |
|---|---|
| x | An object from the 'prediction' function, i.e. a `predFrailty` class object. |
| conf.bands | Logical value. Determines whether confidence intervals will be plotted. The default is FALSE. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"'. |
| cex.legend | size of the legend. Default is 0.7. |
| ylim | range of y-axis. Default is from 0 to 1. |
| Xlab | Label of x-axis. Default is '"Time t"' |
| Ylab | Label of y-axis. |
| ... | Other unused arguments. |

### Value

Print one plot with as many curves as the number of profiles.

---

## plot.predJoint — *Plot predictions using a joint frailty model.*

---

### Description

Plots predicted probabilities of terminal event. Confidence intervals are allowed.

### Usage

```
## S3 method for class 'predJoint'
plot(x, conf.bands=FALSE,
relapses=TRUE,pos.legend="topright", cex.legend=0.7, ylim=c(0,1), Xlab =
"Time t", Ylab = "Prediction probability of event", ...)
```

### Arguments

| | |
|---|---|
| x | An object from the 'prediction' function, more generaly a `predJoint` class object. |
| conf.bands | Logical value. Determines whether confidence intervals will be plotted. The default is FALSE. |
| relapses | Logical value. Determines whether observed recurrent events will be plotted. The default is TRUE. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | size of the legend. Default is 0.7 |

| | |
|---|---|
| ylim | range of y-axis. Default is from 0 to 1 |
| Xlab | Label of x-axis. Default is '"Time t"' |
| Ylab | Label of y-axis. Default is '"Prediction probability of event"' |
| ... | Other unused arguments |

## Value

Print as many plots as the number of subjects.

---

plot.predLongi — *Plot predictions using a joint model for longitudinal data and a terminal event or a trivariate joint model for longitudinal data, recurrent events and a terminal event.*

---

## Description

Plots predicted probabilities of the event. Confidence intervals are allowed.

## Usage

```
## S3 method for class 'predLongi'
plot(x, conf.bands=FALSE, pos.legend="topright",
cex.legend=0.7, ylim=c(0,1), Xlab = "Time t", Ylab, ...)
```

## Arguments

| | |
|---|---|
| x | An object inheriting from `predLongi`. |
| conf.bands | Logical value. Determines whether confidence intervals will be plotted. The default is FALSE. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"'. |
| cex.legend | size of the legend. Default is 0.7. |
| ylim | range of y-axis. Default is from 0 to 1. |
| Xlab | Label of x-axis. Default is '"Time t"' |
| Ylab | Label of y-axis. |
| ... | Other unused arguments. |

## Value

Print one plot with as many curves as the number of profiles.

`plot.trivPenal` *Plot Method for a trivariate joint model for longitudinal data, recurrent events and a terminal event.*

## Description

Plots estimated baseline survival and hazard functions of a joint model (output from an object of class 'trivPenal') for each type of event (terminal or recurrent). Confidence bands are allowed.

## Usage

```
## S3 method for class 'trivPenal'
plot(x, event = "Both", type.plot = "Hazard", conf.bands =
FALSE, pos.legend="topright", cex.legend = 0.7, ylim, main, color = 2, median=TRUE, Xlab
= "Time", Ylab = "Hazard function", ...)
```

## Arguments

| | |
|---|---|
| `x` | A joint model, an object of class `trivPenal`. |
| `event` | a character string specifying the type of curve. Possible value are "Terminal", "Recurrent", or "Both". The default is "Both". |
| `type.plot` | a character string specifying the type of curve. Possible value are "Hazard", or "Survival". The default is "Hazard". Only the first words are required, e.g "Haz", "Su" |
| `conf.bands` | logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| `pos.legend` | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| `cex.legend` | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| `ylim` | y-axis limits |
| `main` | plot title |
| `color` | curve color (integer) |
| `median` | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| `Xlab` | Label of x-axis. Default is '"Time"' |
| `Ylab` | Label of y-axis. Default is '"Hazard function"' |
| `...` | other unused arguments |

## Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence bands or not (conf.bands argument)

## See Also

trivPenal

## Examples

```
## Not run:
###--- Trivariate joint model for longitudinal data, ---###
###--- recurrent events and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Weibull baseline hazard function
# Random effects as the link function, Gap timescale
# (computation takes around 30 minutes)
model.weib.RE.gap <-trivPenal(Surv(gap.time, new.lesions) ~ cluster(id)
+ age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive", n.nodes = 7)

plot(model.weib.RE.gap)
plot(model.weib.RE.gap, type = "survival")

## End(Not run)
```

---

plot.trivPenalNL *Plot Method for a Non-Linear Trivariate Joint Model for Recurrent Events and a Terminal Event with a Biomarker Described with an ODE.*

---

## Description

Plots estimated baseline survival and hazard functions of a joint model (output from an object of class 'trivPenalNL') for each type of event (terminal or recurrent). Confidence bands are allowed.

## Usage

```
## S3 method for class 'trivPenalNL'
plot(x, event = "Both", type.plot = "Hazard", conf.bands
= FALSE, pos.legend="topright", cex.legend = 0.7, ylim, main, color = 2, median=TRUE,
Xlab = "Time", Ylab = "Hazard function", ...)
```

### Arguments

| | |
|---|---|
| x | A joint model, an object of class trivPenalNL. |
| event | a character string specifying the type of curve. Possible value are "terminal", "recurrent", or "both". The default is "both". |
| type.plot | a character string specifying the type of curve. Possible value are "Hazard", or "survival". The default is "hazard". Only the first words are required, e.g "haz", "su" |
| conf.bands | logical value. Determines whether confidence bands will be plotted. The default is to do so. |
| pos.legend | The location of the legend can be specified by setting this argument to a single keyword from the list '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"', '"top"', '"topright"', '"right"' and '"center"'. The default is '"topright"' |
| cex.legend | character expansion factor *relative* to current 'par("cex")'. Default is 0.7 |
| ylim | y-axis limits |
| main | plot title |
| color | curve color (integer) |
| median | Logical value. Determines whether survival median will be plotted. Default is TRUE. |
| Xlab | Label of x-axis. Default is '"Time"' |
| Ylab | Label of y-axis. Default is '"Hazard function"' |
| ... | other unused arguments |

### Value

Print a plot of the baseline survival or hazard functions for each type of event or both with the confidence bands or not (conf.bands argument)

### See Also

trivPenalNL

### Examples

```
## Not run:
###--- Trivariate joint model for longitudinal data, ---###
###--- recurrent events and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Weibull baseline hazard function
# Random effects as the link function, Gap timescale
# (computation takes around 30 minutes)
model.weib.RE.gap <-trivPenal(Surv(gap.time, new.lesions) ~ cluster(id)
+ age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
```

```
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive", n.nodes = 7)

plot(model.weib.RE.gap)
plot(model.weib.RE.gap, type = "survival")

## End(Not run)
```

---

plotTreatPredJointSurro    *Plot of the prediction of the treatment effect on the true endpoint and the STE*

---

## Description

Plot the prediction of the treatment effect on the true endpoint based on the observed treatment effect on the surrogate endpoint, with the prediction interval: results from the one-step Joint surrogate model for evaluating a canditate surrogate endpoint. The graphic also includes vertical lines that cut the x axis to the values of ste. A hatched rectagle/zone indicates the values of $\beta_S$ that predict a non zeto $\beta_T$ , according to the number of value for STE and the shape of the upper confidence limit for the prediction model.

## Usage

```
plotTreatPredJointSurro(
  object,
  from = -3,
  to = 2,
  type = "Coef",
  var.used = "error.estim",
  alpha. = 0.05,
  n = 1000,
  lty = 2,
  d = 3,
  colCI = "blue",
  xlab = "beta.S",
  ylab = "beta.T.predict",
  pred.int.use = "up",
  main = NULL,
  add.accept.area.betaS = TRUE,
  ybottom = -0.05,
  ytop = 0.05,
  density = 20,
  angle = 45,
  legend.show = TRUE,
  leg.x = NULL,
```

```
  leg.y = 2,
  legend = c("Prediction model", "95% prediction Interval", "Beta.S for nonzero beta.T",
    "STE"),
  leg.text.col = "black",
  leg.lty = c(1, 2, 4, NA),
  leg.pch = c(NA, NA, 7, 1),
  leg.bg = "white",
  leg.bty = "n",
  leg.cex = 0.85,
  ...
)
```

## Arguments

| | |
|---|---|
| `object` | An object inheriting from `jointSurroPenal` class (output from calling the function `jointSurroPenal` ). |
| `from` | The range (with `to`) over which the function will be plotted. The default is `from -2 to 2` |
| `to` | The range (with `from`) over which the function will be plotted. The default is `from -2 to 2` |
| `type` | The type of graphic, `"Coef"` for the `log HR` or `"HR"` for hazard ratio. If set to `HR`, the arguments `from` and `to` must take positive values. The default is `"Coef"`. |
| `var.used` | This argument can take two values. The first one is `"error.estim"` and indicates if the prediction error take into account the estimation error of the estimates of the parameters. If the estimates are supposed to be known or if the dataset includes a high number of trials with a high number of subject per trial, value `No.error` can be used. The default is `error.estim` (highly recommended). |
| `alpha.` | The confidence level for the prediction interval. The default is `0.05` |
| `n` | An integer that indicates the number of values for $\beta_S$ . The default is `1000`. |
| `lty` | The line type. Line types can either be specified as an integer (0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings `"blank"`, `"solid"`, `"dashed"`, `"dotted"`, `"dotdash"`, `"longdash"`, or `"twodash"`, where `"blank"` uses "invisible lines" (i.e., does not draw them). The default is `2`. |
| `d` | The desired number of digits after the decimal point for parameters and confidence intervals. Default of 3 digits is used. |
| `colCI` | The color used to display the confidence interval. |
| `xlab` | A title for the x axis. |
| `ylab` | A title for the y axis. |
| `pred.int.use` | A character string that indicates the bound of the prediction interval to use to compute the STE. Possible values are `up` for the upper bound (the default) or `lw` for the lower bound. `up` when we have a protective treatment effect and `lw` when we have a deleterious treatment effect. |
| `main` | Title of the graphics |

`add.accept.area.betaS`
: A boolean that indicates if the plot should add acceptance area for $\beta_S$ that predict a nonzero $\beta_T$ . The default is TRUE

`ybottom`
: A scalar for the left y bottom position of the rectangle on the x-axis associated with acceptable value for $\beta_S$ to predict a non zero $\beta_T$ . The default is `-0.05`.

`ytop`
: A scalar for the top right y position of the rectangle on the x-axis associated with acceptable value for $\beta_S$ to predict a non zero $\beta_T$ . The default is `0.05`.

`density`
: The density of shading lines, in lines per inch. The default value of 'NULL' means that no shading lines are drawn. A zero value of 'density' means no shading lines whereas negative values (and 'NA') suppress shading (and so allow color filling). The default is `20`

`angle`
: Angle (in degrees) of the shading lines. The default is `45`

`legend.show`
: A boolean that indicates if the legend should be displayed

`leg.x`
: The x co-ordinate to be used to position the legend.

`leg.y`
: The y co-ordinate to be used to position the legend. The default is `4`

`legend`
: A character or expression vector of length >= 1 to appear in the legend

`leg.text.col`
: The color used for the legend text. The default is `black`.

`leg.lty`
: The line type, width and color for the legend box (if bty = "o").

`leg.pch`
: = The plotting symbols appearing in the legend, as numeric vector or a vector of 1-character strings (see points). Unlike `points`, this can all be specified as a single multi-character string. Must be specified for symbol drawing.

`leg.bg`
: The background color for the legend box. (Note that this is only used if bty `!=` `"n"`.)

`leg.bty`
: The type of box to be drawn around the legend. The allowed values are `"o"` (the default) and `"n"`.

`leg.cex`
: Character expansion factor relative to current par(`"cex"`). Used for text as defined in legend.

`...`
: other unused arguments

## Value

For a considered treatment effects on the surrogate enpoint, plot the associated treatment effects on the true endpoint predicted from the joint surrogate model with the prediction interval.

## Author(s)

Casimir Ledoux Sofeu `<casimir.sofeu@u-bordeaux.fr>`, `<scl.ledoux@gmail.com>` and Virginie Rondeau `<virginie.rondeau@inserm.fr>`

## References

Burzykowski T, Buyse M (2006). "Surrogate threshold effect: an alternative measure for meta-analytic surrogate endpoint validation." Pharmaceutical Statistics, 5(3), 173-186.ISSN 1539-1612.

Sofeu, C. L. and Rondeau, V. (2020). How to use frailtypack for validating failure-time surrogate endpoints using individual patient data from meta-analyses of randomized controlled trials. PLOS ONE; 15, 1-25.

## See Also

jointSurroPenal, jointSurroCopPenal, predict.jointSurroPenal

## Examples

```
## Not run:


###--- Joint surrogate model ---###
###---evaluation of surrogate endpoints---###

data(dataOvarian)
joint.surro.ovar <- jointSurroPenal(data = dataOvarian, n.knots = 8,
                init.kappa = c(2000,1000), indicator.alpha = 0,
                nb.mc = 200, scale = 1/365)

## "HR"
plotTreatPredJointSurro(joint.surro.ovar, from = 0, to = 4,
                type = "HR", lty = 2, leg.y = 13)

## or without acceptance area for betaS:
plotTreatPredJointSurro(joint.surro.ovar, from = 0, to = 4,
                type = "HR", lty = 2, leg.y = 13,
                add.accept.area.betaS = FALSE)

## "log HR"
plotTreatPredJointSurro(joint.surro.ovar, from = -2, to = 2,
                type = "Coef", lty = 2, leg.y = 3.5)

### For a value of ste greater than 0 (HR > 1), which induces deleterious
### treatment effet, argument "pred.int.use" can be set to "lw"

plotTreatPredJointSurro(joint.surro.ovar, from = 0, to = 2,
                type = "HR", lty = 2, leg.y = 4,
                pred.int.use = "lw")


## End(Not run)
```

---

predict.jointSurroPenal

*S3method predict for the one-step Joint surrogate models for the evaluation of a canditate surrogate endpoint.*

---

## Description

Predict the treatment effect on the true endpoint ($\beta_T$ ), based on the treatment effect observed on the surrogate endpoint ($\beta_S$ ).

## Usage

```
## S3 method for class 'jointSurroPenal'
predict(object, datapred = NULL, betaS.obs = NULL,
betaT.obs = NULL, ntrial0 = NULL, var.used = "error.estim", alpha. = 0.05,
dec = 3, colCI = "red", from = -2, to = 2, type = "Coef", ...)
```

## Arguments

| | |
|---|---|
| `object` | An object inheriting from `jointSurroPenal` class (output from calling the function `jointSurroPenal` or `jointSurroCopPenal`). |
| `datapred` | Dataset to use for the prediction. If this argument is specified, the data structure must be the same as the parameter `data` in the function jointSurroPenal or jointSurroCopPenal. However, if observation on the true endpoint are not available, columns timeT and `statusT` can be absent. In this case, the $\beta_S$ are calculated using Cox proportional hazards models. |
| `betaS.obs` | Observed treatment effect on the surrogate endpoint, to use for the prediction of the treatment effect on the true endpoint. If not null, this value is used for prediction instead of `datapred`. The default is `NULL`. |
| `betaT.obs` | Observed treatment effect on the true endpoint. Used to assess the prediction if not null. The defaut is `NULL`. |
| `ntrial0` | Number of subjects include in the new trial. Required if `betaS.obs` is not null. The default is `NULL`. |
| `var.used` | This argument can take two values. The first one is `"error.estim"` and indicates if the prediction error take into account the estimation error of the estimates of the parameters. If the estimates are supposed to be known or if the dataset includes a high number of trials with a high number of subject per trial, value `No.error` can be used. The default is `error.estim` (highly recommended). |
| `alpha.` | The confidence level for the prediction interval. The default is `0.05` |
| `dec` | The desired number of digits after the decimal point for parameters and confidence intervals. Default of 3 digits is used. |
| `colCI` | The color used to display the confidence interval. |
| `from` | The range (with `to`) over which the function will be plotted. The default is `from -2 to 2` |
| `to` | The range (with `from`) over which the function will be plotted. The default is `from -2 to 2` |
| `type` | The type of graphic, `"Coef"` for the `log HR` or `"HR"` for hazard ratio. If set to `HR`, the arguments `from` and `to` must take positive values. The default is `"Coef"`. |
| `...` | other unused arguments. See the function (plotTreatPredJointSurro) |

## Details

Prediction is based on the formulas described in (Burzikwosky *et al.*, 2006). We do not consider the case in which the prediction take into account estimation error on the estimate of the treatment effect on the surrogate endpoint in the new trial.

## Value

Returns and display a dataframe including for each trial the number of included subjects (if available), the observed treatment effect on surrogate endpoint, the observed treatment effect on true endpoint (if available) and the predicted treatment effect on true enpoint with the associated prediction intervals. If the observe treatment effect on true endpoint (if available) is included into the prediction interval, the last columns contains "*". This function also produces a plot of predicted treatment effects on the true endpoint according to the given values of the treatment effects on the surrogate endpoint, with the prediction intervals.

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

## References

Burzykowski T, Buyse M (2006). "Surrogate threshold effect: an alternative measure for meta-analytic surrogate endpoint validation." Pharmaceutical Statistics, 5(3), 173-186.ISSN 1539-1612.

Sofeu, C. L. and Rondeau, V. (2020). How to use frailtypack for validating failure-time surrogate endpoints using individual patient data from meta-analyses of randomized controlled trials. PLOS ONE; 15, 1-25.

## See Also

jointSurroPenal, jointSurroCopPenal

## Examples

```
## Not run:


###--- Joint surrogate model ---###
###---evaluation of surrogate endpoints---###

data(dataOvarian)
joint.surro.ovar <- jointSurroPenal(data = dataOvarian, n.knots = 8,
                init.kappa = c(2000,1000), indicator.alpha = 0,
                nb.mc = 200, scale = 1/365)

# prediction of the treatment effects on the true endpoint in each trial of
# the dataOvarian dataset
predict(joint.surro.ovar)

# prediction of the treatment effect on the true endpoint from an observed
# treatment effect on the surrogate endpoint in a given trial

# in log HR
predict(joint.surro.ovar, betaS.obs = -0.797, betaT.obs = -1.018)
predict(joint.surro.ovar, type = "Coef", betaS.obs = -1, leg.y = 0, leg.x = 0.3, to = 2.3)
```

```
predict(joint.surro.ovar, type = "Coef", leg.y = 3.5, add.accept.area.betaS = F, to = 2.3)

# in HR
predict(joint.surro.ovar, betaS.obs = exp(-0.797), betaT.obs = exp(-1.018))
predict(joint.surro.ovar, type = "HR", betaS.obs = log(0.65), leg.y = 5, to = 2.3)
predict(joint.surro.ovar, type = "HR", leg.y = 5, add.accept.area.betaS = F, to = 2.3)

## End(Not run)
```

---

`prediction` *Prediction probabilities for Cox proportional hazard, Shared, Joint frailty models, Joint models for longitudinal data and a terminal event and Trivariate joint model for longitudinal data, recurrent events and a terminal event (linear and non-linear).*

---

## Description

**For Cox proportional hazard model**

A predictive probability of event between t and horizon time t+w, with w the window of prediction.

$$P(t,t+w) = \frac{S_i(t) - S_i(t+w)}{S_i(t)} = 1 - \left(\frac{S_0(t+w)}{S_0(t)}\right)^{\exp(\beta' Z_i)}$$

**For Gamma Shared Frailty model for clustered (not recurrent) events**

Two kinds of predictive probabilities can be calculated:

- a conditional predictive probability of event between t and horizon time t+w, i.e. given a specific group

$$P^{cond}(t,t+w) = \frac{S_{ij}(t|u_i) - S_{ij}(t+w|u_i)}{S_{ij}(t|u_i)} = 1 - \left(\frac{S_0(t+w)}{S_0(t)}\right)^{u_i \exp(\beta' Z_{ij})}$$

- a marginal predictive probability of event between t and horizon time t+w, i.e. averaged over the population

$$P^{marg}(t,t+w) = 1 - \left(\frac{1+\theta H_0(t)\exp(\beta' Z_{ij})}{1+\theta H_0(t+w)\exp(\beta' Z_{ij})}\right)^{1/\theta}$$

**For Gaussian Shared Frailty model for clustered (not recurrent) events**

Two kinds of predictive probabilities can be calculated:

- a conditional predictive probability of event between t and horizon time t+w, i.e. given a specific group and given a specific Gaussian random effect $\eta$

$$P^{cond}(t,t+w) = \frac{S_{ij}(t|\eta_i) - S_{ij}(t+w|\eta_i)}{S_{ij}(t|\eta_i)} = 1 - \left(\frac{S_0(t+w)}{S_0(t)}\right)^{\exp(\eta_i + \beta' Z_{ij})}$$

- a marginal predictive probability of event between t and horizon time t+w, i.e. averaged over the population

$$P^{marg}(t, t+w) = \frac{\int_{-\infty}^{+\infty} (S_{ij}(t|\eta_i) - S_{ij}(t+w|\eta_i))g(\eta)d\eta}{\int_{-\infty}^{+\infty} S_{ij}(t)g(\eta)d\eta}$$

**For Gamma Shared Frailty model for recurrent events**

Two kinds of predictive probabilities can be calculated:

- A marginal predictive probability of event between t and horizon time t+w, i.e. averaged over the population.

$$P^{marg}(t, t+w) = \frac{\int_0^{+\infty} (S_{i(J+1)}(t|u_i) - S_{ij}(t+w|u_i)) \cdot (u_i)^J S_{ij}(X_{iJ}|u_i)g(u)du}{\int_0^{+\infty} S_{i(J+1)}(t|u_i)(u_i)^J S_{i(J+1)}(X_{iJ}|u_i))g(u)du}$$

- a conditional predictive probability of event between t and horizon time t+w, i.e. given a specific individual.

This prediction method is the same as the conditional gamma prediction method applied for clustered events (see formula

$$P^{cond}$$

before).

**For Gaussian Shared Frailty model for recurrent events**

Two kinds of predictive probabilities can be calculated:

- A marginal predictive probability of event between t and horizon time t+w, i.e. averaged over the population.

$$P^{marg}(t, t+w) = \frac{\int_0^{+\infty} (S_{i(J+1)}(t|\eta_i) - S_{ij}(t+w|\eta_i)) \cdot \exp(J\eta_i)S_{ij}(X_{iJ}|\eta_i)g(\eta)d\eta}{\int_0^{+\infty} S_{i(J+1)}(t|\eta_i)\exp(J\eta_i)S_{i(J+1)}(X_{iJ}|\eta_i))g(\eta)d\eta}$$

- a conditional predictive probability of event between t and horizon time t+w, i.e. given a specific individual.

This prediction method is the same as the conditional Gaussian prediction method applied for clustered events (see formula

$$P^{cond}$$

before).

It is possible to compute all these predictions in two ways on a scale of times : - either you want a cumulative probability of developing the event between t and t+w (with t fixed, but with a varying window of prediction w); - either you want at a specific time the probability to develop the event in the next w (ie, for a varying prediction time t, but for a fixed window of prediction). See Details.

**For Joint Frailty model**

Prediction for two types of event can be calculated : for a terminal event or for a new recurrent event, knowing patient's characteristics.

**- Prediction of death knowing patients' characteristics :**

It is to predict the probability of death in a specific time window given the history of patient i before the time of prediction t. The history $H_i^{J,l}$, $(l = 1, 2)$ is the information on covariates before time t, but also the number of recurrences and the time of occurences. Three types of marginal probabilities are computed:

- a prediction of death between t and t+w given that the patient had exactly J recurrences ($H_i^{J,1}$) before t

$$P^1(t, t+w) = P(D_i \leq t+w | D_i > t, H_i^{J,1}) = \frac{\int_0^\infty [S_i^D(t) - S_i^D(t+w)](u_i)^J S_{i(J+1)}^R(t) g(u) du_i}{\int_0^\infty S_i^D(t)(u_i)^J S_{i(J+1)}^R(t) g(u) du_i}$$

- a prediction of death between t and t+w given that the patient had at least J recurrences ($H_i^{J,2}$) before t

$$P^2(t, t+w) = P(D_i \leq t+w | D_i > t, H_i^{J,2}) = \frac{\int_0^\infty [S_i^D(t) - S_i^D(t+w)](u_i)^J S_{iJ}^R(X_{iJ}) g(u) du_i}{\int_0^\infty S_i^D(t)(u_i)^J S_{iJ}^R(X_{iJ}) g(u) du_i}$$

- a prediction of death between t and t+w considering the recurrence history only in the parameters estimation. It corresponds to the average probability of death between t and t+w for a patient with these given characteristics.

$$P^3(t, t+w) = P(D_i \leq t+w | D_i > t) = \frac{\int_0^\infty [S_i^D(t) - S_i^D(t+w)] g(u) du_i}{\int_0^\infty S_i^D(t) g(u) du_i}$$

**- Prediction of risk of a new recurrent event knowing patients' characteristics :**

It is to predict the probability of a new recurrent event in a specific time window given the history of patient i before the time of prediction t. The history $H_i^J$ is the information on covariates before time t, but also the number of recurrences and the time of occurences. The marginal probability computed is a prediction of a new recurrent event between t and t+w given that the patient had exactly J recurrences ($H_i^J$) before t:

$$P^R(t, t+w) = P(X_{i(j+1)} \leq t+w | X_{i(j+1)} > t, D_i > t, H_i^J) =$$

$$\frac{\int_0^\infty [S_{i(J+1)}^R(t) - S_{i(J+1)}^R(t+w)] S_i^D(t)(u_i)^J S_{i(J+1)}^R(X_{ij}) g(u) du_i}{\int_0^\infty S_{i(J+1)}^R(t) S_i^D(t)(u_i)^J S_{i(J+1)}^R(X_{ij}) g(u) du_i}$$

It is possible to compute all these predictions in two ways : - either you want a cumulative probability of developing the event between t and t+w (with t fixed, but with a varying window of prediction w); - either you want at a specific time the probability to develop the event in the next w (ie, for a varying prediction time t, but for a fixed window of prediction). See Details.

With Gaussian frailties ($\eta$), the same expressions are used but with ${u_i}^J$ replaced by $\exp(J\eta_i)$ and $g(\eta)$ corresponds to the Gaussian distribution.

**For Joint Nested Frailty models**

Prediction of the probability of developing a terminal event between t and t+w for subject i who survived by time t based on the visiting and disease histories of their own and other family members observed by time t.

Let $(Y_{fi}^R(t))$ be the history of subject $i$ in family $f$, before time $t$, which includes all the recurrent events and covariate information. For disease history, let $T_{fi}^D(t) = min(T_{fi}, t)$ be the observed time to an event before t ; $\delta_{fi}^D(t)$ the disease indicator by time $t$ and $X_{fi}^D(t)$ the covariate information observed up to time t. We define the family history of subject $i$ in family $f$ by

$$H_{f(-i)}(t) = \{Y_{fl}^R(t), T_{fl}^D(t), \delta_{fl}^D(t), X_{fl}^D(t), \forall l \in \{1, ..., i-1, i+1, ..., m_f\}\}$$

which includes the visiting and disease history of all subjects except for subject $i$ in family $f$ as well as their covariate information by time $t$.

The prediction probability can be written as :

$$P(T_{fi}^D < t+s | T_{fi}^D > t, Y_i(t), H_{(f-i)}(t)) =$$

$$\frac{\int\int P(t < T_{fi}^D < t+s | X_{fi}^D, \omega_{fi}) P(Y_i(t) | X_{fi}^R(t), \omega_i) P(H_{f(-i)}(t) | X_{f(-i)}(t), \omega_{fi}) g_{ui} g_{\omega f}}{\int\int P(T_{fi}^D > t | X_{fi}^D, \omega_{fi}) P(Y_i(t) | X_{fi}^R(t), \omega_i) P(H_{f(-i)}(t) | X_{f(-i)}(t), \omega_{fi}) g_{ui} g_{\omega f}}$$

**For Joint models for longitudinal data and a terminal event**

The predicted probabilities are calculated in a specific time window given the history of biomarker measurements before the time of prediction t ($\mathcal{Y}_i(t)$). The probabilities are conditional also on covariates before time t and that the subject was at risk at t. The marginal predicted probability of the terminal event is

$$P(t, t{+}w) = P(D_i \leq t{+}w | D_i > t, \mathcal{Y}_i(t)) = \frac{\int_0^\infty [S_i^D(t) - S_i^D(t+w)] f(\mathcal{Y}_i(t) | \boldsymbol{X}_{Li}, \boldsymbol{b}_i) f(\boldsymbol{b}_i) d\boldsymbol{b}_i}{\int_0^\infty S_i^D(t) f(\mathcal{Y}_i(t) | \boldsymbol{X}_{Li}, \boldsymbol{b}_i) f(\boldsymbol{b}_i) d\boldsymbol{b}_i}$$

These probabilities can be calculated in several time points with fixed time of prediction t and varying window w or with fixed window w and varying time of prediction t. See Details for an example of how to construct time windows.

**For Trivariate joint models for longitudinal data, recurrent events and a terminal event**

The predicted probabilities are calculated in a specific time window given the history of biomarker measurements $\mathcal{Y}_i(t)$ and recurrences $H_i^{J,1}$ (complete history of recurrences with known $J$ number of observed events) before the time of prediction t. The probabilities are conditional also on covariates before time t and that the subject was at risk at t. The marginal predicted probability of the terminal event is

$$\begin{aligned} P(t, t+w) &= P(D_i \leq t+w | D_i > t, H_i^{J,1}, \mathcal{Y}_i(t)) \\ &= \frac{\int_0^\infty [S_i^D(t) - S_i^D(t+w)])] \exp(J(v_i + g(t)^\top \boldsymbol{\eta}_R)) S_{i(J+1)}^R(t) f(\mathcal{Y}_i(t) | \boldsymbol{X}_{Li}, \boldsymbol{b}_i) f(\boldsymbol{u}_i) d\boldsymbol{u}_i}{\int_0^\infty S_i^D(t) \exp(J(v_i + g(t)^\top \boldsymbol{\eta}_R)) S_{i(J+1)}^R(t) f(\mathcal{Y}_i(t) | \boldsymbol{X}_{Li}, \boldsymbol{b}_i) f(\boldsymbol{u}_i) d\boldsymbol{u}_i} \end{aligned}$$

The biomarker history can be represented using a linear (`trivPenal`) or non-linear mixed-effects model (`trivPenalNL`).

These probabilities can be calculated in several time points with fixed time of prediction t and varying window w or with fixed window w and varying time of prediction t. See Details for an example of how to construct time windows.

## Usage

```
prediction(fit, data, data.Longi, t, window, event="Both", conditional =
FALSE, MC.sample=0, individual)
```

## Arguments

| | |
|---|---|
| `fit` | A frailtyPenal, jointPenal, longiPenal, trivPenal or trivPenalNL object. |
| `data` | Data frame for the prediction. See Details. |
| `data.Longi` | Data frame for the prediction used for joint models with longitudinal data. See Details. |
| `t` | Time or vector of times for prediction. |
| `window` | Window or vector of windows for prediction. |
| `event` | Only for joint and shared models. The type of event you want to predict : "Terminal" for a terminal event, "Recurrent" for a recurrent event or "Both". Default value is "Both". For joint nested model, only 'Terminal' is allowed. In a shared model, if you want to predict a new recurrent event then the argument "Recurrent" should be use. If you want to predict a new event from clustered data, do not use this option. |
| `conditional` | Only for prediction method applied on shared models. Provides distinction between the conditional and marginal prediction methods. Default is FALSE. |
| `MC.sample` | Number of samples used to calculate confidence bands with a Monte-Carlo method (with a maximum of 1000 samples). If MC.sample=0 (default value), no confidence intervals are calculated. |
| `individual` | Only for joint nested model. Vector of individuals (of the same family) you want to make prediction. |

## Details

To compute predictions with a prediction time t fixed and a variable window:

```
prediction(fit, datapred, t=10, window=seq(1,10,by=1))
```

Otherwise, you can have a variable prediction time and a fixed window.

```
prediction(fit, datapred, t=seq(10,20,by=1), window=5)
```

Or fix both prediction time t and window.

```
prediction(fit, datapred,
t=10, window=5)
```

The data frame building is an important step. It will contain profiles of patient on which you want to do predictions. To make predictions on a Cox proportional hazard or a shared frailty model, only covariates need to be included. You have to distinguish between numerical and categorical variables (factors). If we fit a shared frailty model with two covariates sex (factor) and age (numeric), here is the associated data frame for three profiles of prediction.

```
 datapred <- data.frame(sex=0,age=0) datapred$sex <-
as.factor(datapred$sex) levels(datapred$sex)<- c(1,2) datapred[1,] <-
c(1,40) # man, 40 years old datapred[2,] <- c(2,45) # woman, 45 years old
datapred[3,] <- c(1,60) # man, 60 years old
```

**Time-dependent covariates:** In the context of time-dependent covariate, the last previous value of the covariate is used before the time t of prediction.

It should be noted, that in a data frame for both marginal and conditional prediction on a shared frailty model for clustered data, the group must be specified. In the case of marginal predictions this can be any number as it does not influence predictions. However, for conditional predictions, the group must be also included in the data set used for the model fitting. The conditional predictions apply the empirical Bayes estimate of the frailty from the specified cluster. Here, three individuals belong to group 5.

```
 datapred <- data.frame(group=0, sex=0,age=0) datapred$sex <-
as.factor(datapred$sex) levels(datapred$sex)<- c(1,2) datapred[1,] <-
c(5,1,40) # man, 40 years old (cluster 5) datapred[2,] <- c(5,2,45) # woman,
45 years old (cluster 5) datapred[3,] <- c(5,1,60) # man, 60 years old
(cluster 5)
```

To use the prediction function on joint frailty models and trivariate joint models, the construction will be a little bit different. In these cases, the prediction for the terminal event takes into account covariates but also history of recurrent event times for a patient. You have to create a data frame with the relapse times, the indicator of event, the cluster variable and the covariates. Relapses occurring after the prediction time may be included but will be ignored for the prediction. A joint model with calendar-timescale need to be fitted with Surv(start,stop,event), relapse times correspond to the "stop" variable and indicators of event correspond to the "event" variable (if event=0, the relapse will not be taken into account). For patients without relapses, all the values of "event" variable should be set to 0. Finally, the same cluster variable name needs to be in the joint model and in the data frame for predictions ("id" in the following example). For instance, we observe relapses of a disease and fit a joint model adjusted for two covariates sex (1:male 2:female) and chemo (treatment by chemotherapy 1:no 2:yes). We describe 3 different profiles of prediction all treated by chemotherapy: 1) a man with four relapses at 100, 200, 300 and 400 days, 2) a man with only one relapse at 1000 days, 3) a woman without relapse.

```
 datapred <- data.frame(time=0,event=0,id=0,sex=0,chemo=0)
datapred$sex <- as.factor(datapred$sex) levels(datapred$sex) <- c(1,2)
datapred$chemo <- as.factor(datapred$chemo) levels(datapred$chemo) <- c(1,2)
datapred[1,] <- c(100,1,1,1,2) # first relapse of the patient 1 datapred[2,]
<- c(200,1,1,1,2) # second relapse of the patient 1 datapred[3,] <-
c(300,1,1,1,2) # third relapse of the patient 1 datapred[4,] <-
c(400,1,1,1,2) # fourth relapse of the patient 1 datapred[5,] <-
```

```
c(1000,1,2,1,2) # one relapse at 1000 days for patient 2 datapred[6,] <-
c(100,0,3,2,2) # patient 3 did not relapse
```

The data can also be the dataset used to fit the joint model. In this case, you will obtain as many prediction rows as patients.

Finally, for the predictions using joint models for longitudinal data and a terminal event and trivariate joint models, a data frame with the history of the biomarker measurements must be provided. It must include data on measurements (values and time points), cluster variable and covariates. Measurements taken after the prediction time may be included but will be ignored for the prediction. The same cluster variable name must be in the data frame, in the data frame used for the joint model and in the data frame with the recurrent event and terminal event times. For instance, we observe two patients and each one had 5 tumor size measurements (patient 1 had an increasing tumor size and patient 2, decreasing). The joint model used for the predictions was adjusted on sex (1: male, 2: female), treatment (1: sequential arm, 2: combined arm), WHO baseline performance status (1: 0 status, 2: 1 status, 3: 2 status) and previous resection of the primate tumor (0: no, 1: yes). The data frame for the biomarker measurements can be:

```
 datapredj_longi <- data.frame(id = 0, year = 0, tumor.size =
0, treatment = 0, age = 0, who.PS = 0, prev.resection = 0)
datapredj_longi$treatment <- as.factor(datapredj_longi$treatment)
levels(datapredj_longi$treatment) <- 1:2 datapredj_longi$age <-
as.factor(datapredj_longi$age) levels(datapredj_longi$age) <- 1:3
datapredj_longi$who.PS <- as.factor(datapredj_longi$who.PS)
levels(datapredj_longi$who.PS) <- 1:3 datapredj_longi$prev.resection <-
as.factor (datapredj_longi$prev.resection)
levels(datapredj_longi$prev.resection) <- 1:2 # patient 1: increasing tumor
size datapredj_longi[1,] <- c(1, 0,1.2 ,2,1,1,1) datapredj_longi[2,] <-
c(1,0.3,1.4,2,1,1,1) datapredj_longi[3,] <- c(1,0.6,1.9,2,1,1,1)
datapredj_longi[4,] <- c(1,0.9,2.5,2,1,1,1) datapredj_longi[5,] <-
c(1,1.5,3.9,2,1,1,1)

# patient 2: decreasing tumor size datapredj_longi[6,] <- c(2, 0,1.2
,2,1,1,1) datapredj_longi[7,] <- c(2,0.3,0.7,2,1,1,1) datapredj_longi[8,] <-
c(2,0.5,0.3,2,1,1,1) datapredj_longi[9,] <- c(2,0.7,0.1,2,1,1,1)
datapredj_longi[10,] <- c(2,0.9,0.1,2,1,1,1)
```

## Value

The following components are included in a 'predFrailty' object obtained by using prediction function for Cox proportional hazard and shared frailty model.

| | |
|---|---|
| `npred` | Number of individual predictions |
| `x.time` | A vector of prediction times of interest (used for plotting predictions): vector of prediction times t if fixed window. Otherwise vector of prediction times t+w |
| `window` | Prediction window or vector of prediction windows |
| `pred` | Predictions estimated for each profile |
| `icproba` | Logical value. Were confidence intervals estimated ? |

| | |
|---|---|
| predLow | Lower limit of Monte-Carlo confidence interval for each prediction |
| predHigh | Upper limit of Monte-Carlo confidence interval for each prediction |
| type | Type of prediction probability (marginal or conditional) |
| group | For conditional probability, the list of group on which you make predictions |

The following components are included in a 'predJoint' object obtained by using prediction function for joint frailty model.

| | |
|---|---|
| npred | Number of individual predictions |
| x.time | A vector of prediction times of interest (used for plotting predictions): vector of prediction times t if fixed window. Otherwise vector of prediction times t+w |
| window | Prediction window or vector of prediction windows |
| group | Id of each patient |
| pred1 | Estimation of probability of type 1: exactly j recurrences |
| pred2 | Estimation of probability of type 2: at least j recurrences |
| pred3 | Estimation of probability of type 3 |
| pred1_rec | Estimation of prediction of relapse |
| icproba | Logical value. Were confidence intervals estimated ? |
| predlow1 | Lower limit of Monte-Carlo confidence interval for probability of type 1 |
| predhigh1 | Upper limit of Monte-Carlo confidence interval for probability of type 1 |
| predlow2 | Lower limit of Monte-Carlo confidence interval for probability of type 2 |
| predhigh2 | Upper limit of Monte-Carlo confidence interval for probability of type 2 |
| predlow3 | Lower limit of Monte-Carlo confidence interval for probability of type 3 |
| predhigh3 | Upper limit of Monte-Carlo confidence interval for probability of type 3 |
| predhigh1_rec | Upper limit of Monte-Carlo confidence interval for prediction of relapse |
| predlow1_rec | Lower limit of Monte-Carlo confidence interval for prediction of relapse |

The following components are included in a 'predLongi' object obtained by using prediction function for joint models with longitudinal data.

| | |
|---|---|
| npred | Number of individual predictions |
| x.time | A vector of prediction times of interest (used for plotting predictions): vector of prediction times t if fixed window. Otherwise vector of prediction times t+w |
| window | Prediction window or vector of prediction windows |
| group | Id of each patient |
| pred | Estimation of probability |
| icproba | Logical value. Were confidence intervals estimated? |
| predLow | Lower limit of Monte-Carlo confidence intervals |
| predHigh | Upper limit of Monte-Carlo confidence intervals |
| trivariate | Logical value. Are the prediction calculated from the trivariate model? |

## References

A. Krol, L. Ferrer, JP. Pignon, C. Proust-Lima, M. Ducreux, O. Bouche, S. Michiels, V. Rondeau (2016). Joint Model for Left-Censored Longitudinal Data, Recurrent Events and Terminal Event: Predictive Abilities of Tumor Burden for Cancer Evolution with Application to the FFCD 2000-05 Trial. *Biometrics* **72**(3) 907-16.

A. Mauguen, B. Rachet, S. Mathoulin-Pelissier, G. MacGrogan, A. Laurent, V. Rondeau (2013). Dynamic prediction of risk of death using history of cancer recurrences in joint frailty models. *Statistics in Medicine*, **32(30)**, 5366-80.

V. Rondeau, A. Laurent, A. Mauguen, P. Joly, C. Helmer (2015). Dynamic prediction models for clustered and interval-censored outcomes: investigating the intra-couple correlation in the risk of dementia. *Statistical Methods in Medical Research*

## Examples

```
## Not run:

#####################################################
#### prediction on a COX or SHARED frailty model ####
#####################################################

data(readmission)
#-- here is a generated cluster (31 clusters of 13 subjects)
readmission <- transform(readmission,group=id%%31+1)

#-- we compute predictions of death
#-- we extract last row of each subject for the time of death
readmission <- aggregate(readmission,by=list(readmission$id),
                         FUN=function(x){x[length(x)]})[,-1]

##-- predictions on a Cox proportional hazard model --##
cox <- frailtyPenal(Surv(t.stop,death)~sex+dukes,
n.knots=10,kappa=10000,data=readmission)

#-- construction of the data frame for predictions
datapred <- data.frame(sex=0,dukes=0)
datapred$sex <- as.factor(datapred$sex)
levels(datapred$sex)<- c(1,2)
datapred$dukes <- as.factor(datapred$dukes)
levels(datapred$dukes)<- c(1,2,3)
datapred[1,] <- c(1,2) # man, dukes 2
datapred[2,] <- c(2,3) # woman, dukes 3

#-- prediction of death for two patients between 100 and 100+w,
#-- with w in (50,100,...,1900)
pred.cox <- prediction(cox,datapred,t=100,window=seq(50,1900,50))
plot(pred.cox)

#-- prediction of death for two patients between t and t+400,
#-- with t in (100,150,...,1500)
pred.cox2 <- prediction(cox,datapred,t=seq(100,1500,50),window=400)
```

```
plot(pred.cox2)

##-- predictions on a shared frailty model for clustered data --##
sha <- frailtyPenal(Surv(t.stop,death)~cluster(group)+sex+dukes,
n.knots=10,kappa=10000,data=readmission)

#-- marginal prediction
# a group must be specified but it does not influence the results
# in the marginal predictions setting
datapred$group[1:2] <- 1
pred.sha.marg <- prediction(sha,datapred,t=100,window=seq(50,1900,50))
plot(pred.sha.marg)

#-- conditional prediction, given a specific cluster (group=5)
datapred$group[1:2] <- 5
pred.sha.cond <- prediction(sha,datapred,t=100,window=seq(50,1900,50),
                            conditional = TRUE)
plot(pred.sha.cond)

##-- marginal prediction of a recurrent event, on a shared frailty model
data(readmission)

datapred <- data.frame(t.stop=0,event=0,id=0,sex=0,dukes=0)
datapred$sex <- as.factor(datapred$sex)
levels(datapred$sex)<- c(1,2)
datapred$dukes <- as.factor(datapred$dukes)
levels(datapred$dukes)<- c(1,2,3)

datapred[1,] <- c(100,1,1,1,2) #man, dukes 2, 3 recurrent events
datapred[2,] <- c(200,1,1,1,2)
datapred[3,] <- c(300,1,1,1,2)
datapred[4,] <- c(350,0,2,1,2) #man, dukes 2  0 recurrent event

#-- Shared frailty model with gamma distribution
sha <- frailtyPenal(Surv(t.stop,event)~cluster(id)+sex+dukes,n.knots=10,
kappa=10000,data=readmission)
pred.sha.rec.marg <- prediction(sha,datapred,t=200,window=seq(50,1900,50),
event='Recurrent',MC.sample=100)

plot(pred.sha.rec.marg,conf.bands=TRUE)

##-- conditional prediction of a recurrent event, on a shared frailty model
pred.sha.rec.cond <- prediction(sha,datapred,t=200,window=seq(50,1900,50),
event='Recurrent',conditional = TRUE,MC.sample=100)

plot(pred.sha.rec.cond,conf.bands=TRUE)
#####################################################
######## prediction on a JOINT frailty model ########
#####################################################

data(readmission)

##-- predictions of death on a joint model --##
```

```
joi <- frailtyPenal(Surv(t.start,t.stop,event)~cluster(id)
+sex+dukes+terminal(death),formula.terminalEvent=~sex
+dukes,data=readmission,n.knots=10,kappa=c(100,100),recurrentAG=TRUE)

#-- construction of the data frame for predictions
datapredj <- data.frame(t.stop=0,event=0,id=0,sex=0,dukes=0)
datapredj$sex <- as.factor(datapredj$sex)
levels(datapredj$sex) <- c(1,2)
datapredj$dukes <- as.factor(datapredj$dukes)
levels(datapredj$dukes) <- c(1,2,3)
datapredj[1,] <- c(100,1,1,1,2)
datapredj[2,] <- c(200,1,1,1,2)
datapredj[3,] <- c(300,1,1,1,2)
datapredj[4,] <- c(400,1,1,1,2)
datapredj[5,] <- c(380,1,2,1,2)

#-- prediction of death between 100 and 100+500 given relapses
pred.joint0 <- prediction(joi,datapredj,t=100,window=500,event = "Terminal")
print(pred.joint0)

#-- prediction of death between 100 and 100+w given relapses
# (with confidence intervals)
pred.joint <- prediction(joi,datapredj,t=100,window=seq(50,1500,50),
event = "Terminal",MC.sample=100)
plot(pred.joint,conf.bands=TRUE)
# each y-value of the plot corresponds to the prediction between [100,x]

#-- prediction of death between t and t+500 given relapses
pred.joint2 <- prediction(joi,datapredj,t=seq(100,1000,50),
window=500,event = "Terminal")
plot(pred.joint2)
# each y-value of the plot corresponds to the prediction between [x,x+500],
#or in the next 500

#-- prediction of relapse between 100 and 100+w given relapses
# (with confidence intervals)
pred.joint <- prediction(joi,datapredj,t=100,window=seq(50,1500,50),
event = "Recurrent",MC.sample=100)
plot(pred.joint,conf.bands=TRUE)
# each y-value of the plot corresponds to the prediction between [100,x]

#-- prediction of relapse and death between 100 and 100+w given relapses
# (with confidence intervals)
pred.joint <- prediction(joi,datapredj,t=100,window=seq(50,1500,50),
event = "Both",MC.sample=100)
plot(pred.joint,conf.bands=TRUE)
# each y-value of the plot corresponds to the prediction between [100,x]

#############################################################################
### prediction on a JOINT model for longitudinal data and a terminal event ####
#############################################################################
```

```
data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

#-- construction of the data-frame for predictions
#-- biomarker observations
datapredj_longi <- data.frame(id = 0, year = 0, tumor.size = 0, treatment = 0,
 age = 0, who.PS = 0, prev.resection = 0)
datapredj_longi$treatment <- as.factor(datapredj_longi$treatment)
levels(datapredj_longi$treatment) <- 1:2
datapredj_longi$age <- as.factor(datapredj_longi$age)
levels(datapredj_longi$age) <- 1:3
datapredj_longi$who.PS <- as.factor(datapredj_longi$who.PS)
levels(datapredj_longi$who.PS) <- 1:3
datapredj_longi$prev.resection <- as.factor(datapredj_longi$prev.resection)
levels(datapredj_longi$prev.resection) <- 1:2

# patient 1: increasing tumor size
datapredj_longi[1,] <- c(1, 0,1.2 ,2,1,1,1)
datapredj_longi[2,] <- c(1,0.3,1.4,2,1,1,1)
datapredj_longi[3,] <- c(1,0.6,1.9,2,1,1,1)
datapredj_longi[4,] <- c(1,0.9,2.5,2,1,1,1)
datapredj_longi[5,] <- c(1,1.5,3.9,2,1,1,1)

# patient 2: decreasing tumor size
datapredj_longi[6,] <- c(2, 0,1.2 ,2,1,1,1)
datapredj_longi[7,] <- c(2,0.3,0.7,2,1,1,1)
datapredj_longi[8,] <- c(2,0.5,0.3,2,1,1,1)
datapredj_longi[9,] <- c(2,0.7,0.1,2,1,1,1)
datapredj_longi[10,] <- c(2,0.9,0.1,2,1,1,1)

#-- terminal event
datapredj <- data.frame(id = 0, treatment = 0, age = 0, who.PS = 0,
prev.resection = 0)
datapredj$treatment <- as.factor(datapredj$treatment)
levels(datapredj$treatment) <- 1:2
datapredj$age <- as.factor(datapredj$age)
levels(datapredj$age) <- 1:3
datapredj$who.PS <- as.factor(datapredj$who.PS)
datapredj$prev.resection <- as.factor(datapredj$prev.resection)
levels(datapredj$prev.resection) <- 1:2
levels(datapredj$who.PS) <- 1:3
datapredj[1,] <- c(1,2,1,1,1)
datapredj[2,] <- c(2,2,1,1,1)

model.spli.CL <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
colorectalSurv, data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Current-level", left.censoring = -3.33, n.knots = 6,
kappa = 1)
```

```
#-- prediction of death between 1 year and 1+2 given history of the biomarker
pred.jointLongi0 <- prediction(model.spli.CL, datapredj, datapredj_longi,
t = 1, window = 2)
print(pred.jointLongi0)

#-- prediction of death between 1 year and 1+w given history of the biomarker
pred.jointLongi <- prediction(model.spli.CL, datapredj, datapredj_longi,
t = 1, window = seq(0.5, 2.5, 0.2), MC.sample = 100)
plot(pred.jointLongi, conf.bands = TRUE)
# each y-value of the plot corresponds to the prediction between [1,x]

#-- prediction of death between t and t+0.5 given history of the biomarker
pred.jointLongi2 <- prediction(model.spli.CL, datapredj, datapredj_longi,
t = seq(1, 2.5, 0.5), window = 0.5, MC.sample = 100)
plot(pred.jointLongi2, conf.bands = TRUE)
# each y-value of the plot corresponds to the prediction between [x,x+0.5],
#or in the next 0.5

#############################################################################
##### marginal prediction on a JOINT NESTED model for a terminal event ######
#############################################################################
#*--Warning! You can compute this prediction method with ONLY ONE family
#*--by dataset of prediction.
#*--Please make sure your data frame contains a column for individuals AND a
#*--column for the reference number of the family chosen.

data(readmission)
readmissionNested <- transform(readmission,group=id%%30+1)

#-- construction of the data frame for predictions :
#-- family 5 was selected for the prediction

DataPred <- readmissionNested[which(readmissionNested$group==5),]

#-- Fitting the model
modJointNested_Splines <-
frailtyPenal(formula = Surv(t.start, t.stop, event)~subcluster(id)+
cluster(group) + dukes + terminal(death),formula.terminalEvent
=~dukes, data = readmissionNested, recurrentAG = TRUE,n.knots = 8,
kappa = c(9.55e+9, 1.41e+12), initialize = TRUE)

#-- Compute prediction over the individuals 274 and 4 of the family 5
predRead <- prediction(modJointNested_Splines, data=DataPred,t=500,
window=seq(100,1500,200), conditional=FALSE, individual = c(274, 4))


#########################################################################
##### prediction on TRIVARIATE JOINT model (linear and non-linear) ######
#########################################################################

data(colorectal)
data(colorectalLongi)
```

```
#-- construction of the data frame for predictions
#-- history of recurrences and terminal event
datapredj <- data.frame(time0 = 0, time1 = 0, new.lesions = 0, id = 0,
treatment = 0, age = 0, who.PS = 0, prev.resection =0)
datapredj$treatment <- as.factor(datapredj$treatment)
levels(datapredj$treatment) <- 1:2
datapredj$age <- as.factor(datapredj$age)
levels(datapredj$age) <- 1:3
datapredj$who.PS <- as.factor(datapredj$who.PS)
levels(datapredj$who.PS) <- 1:3
datapredj$prev.resection <- as.factor(datapredj$prev.resection)
levels(datapredj$prev.resection) <- 1:2

datapredj[1,] <- c(0,0.4,1,1,2,1,1,1)
datapredj[2,] <- c(0.4,1.2,1,1,2,1,1,1)
datapredj[3,] <- c(0,0.5,1,2,2,1,1,1)

# Linear trivariate joint model
# (computation takes around 40 minutes)
model.trivPenal <-trivPenal(Surv(time0, time1, new.lesions) ~ cluster(id)
+ age + treatment + who.PS +  terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = TRUE,
n.knots = 6, kappa=c(0.01, 2), method.GH="Pseudo-adaptive",
n.nodes=7, init.B = c(-0.07, -0.13, -0.16, -0.17, 0.42, #recurrent events covarates
-0.23, -0.1, -0.09, -0.12, 0.8, -0.23, #terminal event covariates
3.02, -0.30, 0.05, -0.63, -0.02, -0.29, 0.11, 0.74)) #biomarker covariates

#-- prediction of death between 1 year and 1+2
pred.jointTri0 <- prediction(model.trivPenal, datapredj,
datapredj_longi, t = 1, window = 2)
print(pred.jointTri0)

#-- prediction of death between 1 year and 1+w
pred.jointTri <- prediction(model.trivPenal, datapredj,
datapredj_longi, t = 1, window = seq(0.5, 2.5, 0.2), MC.sample = 100)
plot(pred.jointTri, conf.bands = TRUE)

#-- prediction of death between t and t+0.5
pred.jointTri2 <- prediction(model.trivPenal, datapredj,
datapredj_longi, t = seq(1, 2.5, 0.5), window = 0.5, MC.sample = 100)
plot(pred.jointTri2, conf.bands = TRUE)


###############################

# No information on dose - creation of a dummy variable
colorectalLongi$dose <- 1

# (computation can take around 40 minutes)
model.trivPenalNL <- trivPenalNL(Surv(time0, time1, new.lesions) ~ cluster(id) + age + treatment
```

```
 + terminal(state), formula.terminalEvent =~ age + treatment, biomarker = "tumor.size",
 formula.KG ~ 1, formula.KD ~ treatment, dose = "dose", time.biomarker = "year",
 data = colorectal, data.Longi =colorectalLongi, random = c("y0", "KG"), id = "id",
 init.B = c(-0.22, -0.16, -0.35, -0.19, 0.04, -0.41, 0.23), init.Alpha = 1.86,
 init.Eta = c(0.5, 0.57, 0.5, 2.34), init.Biomarker = c(1.24, 0.81, 1.07, -1.53),
 recurrentAG = TRUE, n.knots = 5, kappa = c(0.01, 2), method.GH = "Pseudo-adaptive")

#-- prediction of death between 1 year and 1+2
pred.jointTriNL0 <- prediction(model.trivPenalNL, datapredj,
datapredj_longi, t = 1, window = 2)
print(pred.jointTriNL0)

#-- prediction of death between 1 year and 1+w
pred.jointTriNL <- prediction(model.trivPenalNL, datapredj,
datapredj_longi, t = 1, window = seq(0.5, 2.5, 0.2), MC.sample = 100)
plot(pred.jointTriNL, conf.bands = TRUE)

#-- prediction of death between t and t+0.5
pred.jointTriNL2 <- prediction(model.trivPenalNL, datapredj,
datapredj_longi, t = seq(2, 3, 0.2), window = 0.5, MC.sample = 100)
plot(pred.jointTriNL2, conf.bands = TRUE)


## End(Not run)
```

---

print.additivePenal *Print a Short Summary of parameter estimates of an additive frailty model*

---

## Description

Prints a short summary of the parameter estimates of an additive frailty model or more generally of an 'additivePenal' object

## Usage

```
## S3 method for class 'additivePenal'
print(x, digits = max(options()$digits - 4, 6),
...)
```

## Arguments

| | |
|---|---|
| x | the result of a call to the additivePenal function |
| digits | number of digits to print |
| ... | other unused arguments |

### Value

Print the parameter estimates of the survival or hazard functions.

### See Also

additivePenal

---

`print.Cmeasures` *Print a short summary of results of Cmeasure function.*

---

### Description

Print a short summary of results of the concordance measure estimated by the Cmeasure function.

### Usage

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

### Arguments

| | |
|---|---|
| `x` | a Cmeasures object. |
| `...` | Other unused arguments |

### Value

Print concordance measures estimated.

### See Also

Cmeasures

---

`print.frailtyCmprsk` *Print a Short Summary of parameter estimates of a Weibull competing risks model with (or without) shared frailty between transitions.*

---

### Description

Prints a short summary of parameter estimates of a 'frailtyCmprsk' object

### Usage

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

### Arguments

| | |
|---|---|
| x | the result of a call to the frailtyCmprsk function. |
| ... | other unused arguments. |

### Value

Print the parameter estimates of the survival or hazard functions.

### See Also

frailtyCmprsk

---

## print.frailtyDesign *Print a short table of a 'frailtyDesign' result.*

---

### Description

Print a short table of a 'frailtyDesign' result.

### Usage

```
## S3 method for class 'frailtyDesign'
print(x, digits = 2, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class 'frailtyDesign' (output from one of the *.power or *.ssize functions). |
| digits | number of decimals to print for numeric fields. Default is 2. |
| ... | other unused arguments. |

### See Also

frailtyDesign

### Examples

```
est.ex <- SFM.power(
  Groups = 400, ni = 3, ni.type = "max", FUP = 6, Acc.Dur = 0.5, median.H0 = 1.5,
 beta.HA = log(0.7), theta = 0.5, cens.par = c(3, 10), cens.type = "Unif", data.type = "rec_event"
)

print(est.ex)
```

`print.frailtyIllnessDeath` *Print a Short Summary of parameter estimates of a Weibull Illness-Death model with (or without) shared frailty between transitions.*

### Description

Prints a short summary of parameter estimates of a 'frailtyIllnessDeath' object

### Usage

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

### Arguments

| | |
|---|---|
| `x` | the result of a call to the frailtyIllnessDeath function. |
| `...` | other unused arguments. |

### Value

Print the parameter estimates of the survival or hazard functions.

### See Also

`frailtyIllnessDeath`

`print.frailtyPenal` *Print a Short Summary of parameter estimates of a shared frailty model*

### Description

Prints a short summary of parameter estimates of a 'frailtyPenal' object

### Usage

```
## S3 method for class 'frailtyPenal'
print(x, digits = max(options()$digits - 4, 6),
...)
```

### Arguments

| | |
|---|---|
| `x` | the result of a call to the frailtyPenal function. |
| `digits` | number of digits to print. |
| `...` | other unused arguments. |

### Value

Print the parameter estimates of the survival or hazard functions.

### See Also

`frailtyPenal`

---

`print.jointNestedPenal` *Print a Short Summary of parameter estimates of a joint nested frailty model*

---

### Description

Prints a short summary of parameter estimates of a joint nested frailty model, or more generally an object of class 'jointNestedPenal' for joint nested frailty models.

### Usage

```
## S3 method for class 'jointNestedPenal'
print(x, digits = max(options()$digits - 4,
6), ...)
```

### Arguments

| | |
|---|---|
| `x` | the result of a call to the jointNestedPenal function |
| `digits` | number of digits to print |
| `...` | other unused arguments |

### Value

Print, separately for each type of event (recurrent and terminal), the parameter estimates of the survival or hazard functions.

### See Also

`frailtyPenal`

## print.jointPenal — *Print a Short Summary of parameter estimates of a joint frailty model*

### Description

Prints a short summary of parameter estimates of a joint frailty model, or more generally an object of class 'frailtyPenal' for joint frailty models.

### Usage

```
## S3 method for class 'jointPenal'
print(x, digits = max(options()$digits - 4, 6), ...)
```

### Arguments

| | |
|---|---|
| x | the result of a call to the jointPenal function |
| digits | number of digits to print |
| ... | other unused arguments |

### Value

Print, separately for each type of event (recurrent and terminal), the parameter estimates of the survival or hazard functions.

### See Also

frailtyPenal

## print.jointRecCompet — *Print a Short Summary of parameter estimates of a joint competing risks midel*

### Description

Prints a short summary of parameter estimates of a joint competing risks model or more generally an object of class 'jointRecCompet'.

### Usage

```
## S3 method for class 'jointRecCompet'
print(x, digits = max(options()$digits - 4, 6),
...)
```

### Arguments

| | |
|---|---|
| x | the result of a call to the jointRecCompet function |
| digits | number of digits to print |
| ... | other unused arguments |

### Value

Print, separately for each type of event (Recurrent, Terminal1 and Terminal2), the parameter estimates of the survival or hazard functions.

### See Also

jointRecCompet

---

## print.jointSurroPenal — *Summary of the random effects parameters, the fixed treatment effects, and the surrogacy evaluation criteria estimated from a joint surrogate model*

---

### Description

This function returns the estimate of the coefficients and their standard error with p-values of the Wald test for the joint surrogate model, also hazard ratios (HR) and their confidence intervals for the fixed treatment effects, and finaly an estimate of the surrogacy evaluation criterian (Kendall's $\tau$ and $R^2_{trial}$ )

### Usage

```
## S3 method for class 'jointSurroPenal'
print(x, d = 4, len = 3, nb.gh = 32, ...)
```

### Arguments

| | |
|---|---|
| x | An object inheriting from `jointSurroPenal` class. |
| d | The desired number of digits after the decimal point for parameters. The maximum of 4 digits is required for the estimates. Default of 3 digits is used. |
| len | The desired number of digits after the decimal point for p-value and convergence criteria. Default of 4 digits is used. |
| nb.gh | Number of nodes for the Gaussian-Hermite quadrature. The default is `32 1` for Gaussian-Hermite quadrature. |
| ... | other unused arguments. |

### Value

For the variances parameters of the random effects, it prints the estimate of the coefficients with their standard error, Z-statistics and p-values of the Wald test. For the fixed treatment effects, it also prints HR and its confidence intervals for each covariate. For the surrogacy evaluation criteria, its prints the estimated Kendall's $\tau$ with its 95% Confidence interval obtained by the parametric bootstrap or Delta-method, the estimated $R^2_{trial}$ (R2trial) with standard error and the 95% Confidence interval obtained by Delta-method (Dowd *et al.*, 2014), $R^2_{trial}$ (R2.boot) and its 95% Confidence interval obtained by the parametric bootstrap. We notice that, using bootstrap, the standard error of the point estimate is not available. We propose a classification of $R^2_{trial}$ according to the suggestion of the Institute of Quality and Efficiency in Health Care (Prasad *et al.*, 2015). We also display the surrogate threshold effect (ste) with the associated hazard risk. The rest of parameters concerns the convergence characteristics and included: the penalized marginal log-likelihood, the number of iterations, the LCV and the Convergence criteria.

### Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

### References

Dowd BE, Greene WH, Norton EC (2014). "Computation of Standard Errors." Health Services Research, 49(2), 731-750.

Prasad V, Kim C, Burotto M, Vandross A (2015). "The strength of association between surrogate end points and survival in oncology: A systematic review of trial-level meta- alyses." JAMA Internal Medicine, 175(8), 1389-1398.

### See Also

jointSurroPenal, jointSurroCopPenal, jointSurroTKendall

### Examples

```
## Not run:

###---Data generation---###
data.sim <-jointSurrSimul(n.obs=400, n.trial = 20,cens.adm=549,
          alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1,
          sigma.s = 0.7, sigma.t = 0.7, cor = 0.8, betas = -1.25,
          betat = -1.25, full.data = 0, random.generator = 1,
          seed = 0, nb.reject.data = 0)

###---Estimation---###
joint.surrogate <- jointSurroPenal(data = data.sim, nb.mc = 300,
                   nb.gh = 20, indicator.alpha = 1, n.knots = 6)

print(joint.surrogate)

# or
```

```
joint.surrogate

## End(Not run)
```

print.longiPenal *Print a Summary of parameter estimates of a joint model for longitudinal data and a terminal event*

## Description

Prints a short summary of parameter estimates of a joint model for longitudinal data and a terminal event, an object inheriting from class 'longiPenal'. If a mediation analysis was performed (option mediation set to TRUE in longiPenal) this function displays estimations of the related quantities.

## Usage

```
## S3 method for class 'longiPenal'
print(x, digits = max(options()$digits - 4, 6), ...)
```

## Arguments

| | |
|---|---|
| x | an object inheriting from longiPenal class |
| digits | number of digits to print |
| ... | other unused arguments |

## Value

Print, separately for each part of the model (longitudinal and terminal) the parameter estimates and details on the estimation. Also print in a separate part the results of the mediation analysis if one was performed

## See Also

longiPenal

print.multivPenal *Print a Short Summary of parameter estimates of a multivariate frailty model*

## Description

Prints a short summary of parameter estimates of a multivariate frailty model, or more generally an object of class 'multivPenal'.

## Usage

```
## S3 method for class 'multivPenal'
print(x, digits = max(options()$digits - 4, 6),
...)
```

## Arguments

| | |
|---|---|
| x | the result of a call to the multivPenal function |
| digits | number of digits to print |
| ... | other unused arguments |

## Value

Print, separately for each type of event (recurrent1, recurrent2 and terminal), the parameter estimates of the survival or hazard functions.

## See Also

multivPenal

print.nestedPenal *Print a Short Summary of parameter estimates of a nested frailty model*

## Description

Prints a short summary of parameter estimates of a nested frailty model

## Usage

```
## S3 method for class 'nestedPenal'
print(x, digits = max(options()$digits - 4, 6),
...)
```

### Arguments

| | |
|---|---|
| `x` | the result of a call to the frailtyPenal function for nested frailty models |
| `digits` | number of digits to print |
| `...` | other unused arguments |

### Value

| | |
|---|---|
| `n` | the number of observations used in the fit. |
| `n.groups` | the maximum number of groups used in the fit |
| `n.events` | the number of events observed in the fit |
| `eta` | variance of the subcluster effect $(Var(w_{ij}))$ |
| `theta` | variance of the cluster effect $(Var(v_i))$ |
| `coef` | the coefficients of the linear predictor, which multiply the columns of the model matrix. |
| `SE(H)` | the standard error of the estimates deduced from the variance matrix of theta and of the coefficients. |
| `SE(HIH)` | the standard error of the estimates deduced from the robust estimation of the variance matrix of theta and of the coefficients. |
| `p` | p-value |

### See Also

`frailtyPenal`

---

## `print.prediction` *Print a short summary of results of prediction function.*

---

### Description

Print a short summary of results of prediction function.

### Usage

```
## S3 method for class 'predFrailty'
print(x, digits = 3, ...)
## S3 method for class 'predJoint'
print(x, digits = 3, ...)
## S3 method for class 'predLongi'
print(x, digits = 3, ...)
```

### Arguments

| | |
|---|---|
| `x` | An object from the 'prediction' function, objects inheriting from `predFrailty`, `predJoint` and `predLongi` classes. |
| `digits` | Number of digits to print |
| `...` | Other unused arguments |

## Value

Print the probabilities estimated.

## See Also

prediction

---

`print.trivPenal` *Print a Summary of parameter estimates of a joint model for longitudinal data, recurrent events and a terminal event*

---

## Description

Prints a short summary of parameter estimates of a joint model for longitudinal data, recurrent events and a terminal event, an object inheriting from class 'trivPenal'.

## Usage

```
## S3 method for class 'trivPenal'
print(x, digits = max(options()$digits - 4, 6), ...)
```

## Arguments

| | |
|---|---|
| `x` | an object inheriting from `trivPenal` class |
| `digits` | number of digits to print |
| `...` | other unused arguments |

## Value

Print, separately for each part of the model (longitudinal, recurrent and terminal) the parameter estimates and details on the estimation.

## See Also

trivPenal

## print.trivPenalNL *Print a Summary of parameter estimates of a non-linear trivariate joint model for longitudinal data, recurrent events and a terminal event*

### Description

Prints a short summary of parameter estimates of a non-linear trivariate joint model for longitudinal data, recurrent events and a terminal event, an object inheriting from class 'trivPenalNL'.

### Usage

```
## S3 method for class 'trivPenalNL'
print(x, digits = max(options()$digits - 4, 6), ...)
```

### Arguments

| | |
|---|---|
| x | an object inheriting from `trivPenaNLl` class |
| digits | number of digits to print |
| ... | other unused arguments |

### Value

Print, separately for each part of the model (biomarker growth, biomarker decline, recurrent events and terminal event) the parameter estimates and details on the estimation.

### See Also

`trivPenalNL`

## readmission *Rehospitalization colorectal cancer*

### Description

This contains rehospitalization times after surgery in patients diagnosed with colorectal cancer

### Usage

```
data(readmission)
```

## Format

This data frame contains the following columns:

**id** identification of each subject. Repeated for each recurrence

**enum** which readmission

**t.start** start of interval (0 or previous recurrence time)

**t.stop** recurrence or censoring time

**time** interocurrence or censoring time

**event** rehospitalization status. All event are 1 for each subject excepting last one that it is 0

**chemo** Did patient receive chemotherapy? 1: No; 2:Yes

**sex** gender: 1:Males 2:Females

**dukes** Dukes' tumoral stage: 1:A-B; 2:C 3:D

**charlson** Comorbidity Charlson's index. Time-dependent covariate. 0: Index 0; 1: Index 1-2; 3: Index >=3

**death** death indicator. 1:dead and 0:alive

## Source

Gonzalez, JR., Fernandez, E., Moreno, V., Ribes, J., Peris, M., Navarro, M., Cambray, M. and Borras, JM (2005). Sex differences in hospital readmission among colorectal cancer patients. *Journal of Epidemiology and Community Health*, **59**, 6, 506-511.

---

`readmission2` *Transformed Readmission Data for Illness-Death Modeling*

---

## Description

A dataset derived from the `readmission` data (originally from the `frailtypack` package, related to rehospitalization times after surgery in colorectal cancer patients). This transformed version reshapes the data to fit a standard illness-death model framework, focusing on the first event (rehospitalization) and the terminal event (death). Recurrent rehospitalization events beyond the first one are excluded. Time is scaled to years.

## Usage

```
data(readmission2)
```

## Format

A data frame with one row per subject, containing columns suitable for use with the `IllnessDeath` function:

**id** Unique subject identification number.

**observed_disease_time** Time (in years since surgery) to either the first rehospitalization (illness), death, or administrative censoring, whichever occurred first.

**observed_death_time** Time (in years since surgery) to either death or administrative censoring.

**disease_status** Indicator for the non-terminal event (first rehospitalization). 1 if the subject experienced a first rehospitalization before death/censoring, 0 otherwise.

**death_status** Indicator for the terminal event (death). 1 if the subject died, 0 if censored.

**dukes** Dukes' tumoral stage at baseline (Factor or numeric: 1:A-B; 2:C; 3:D).

**sex** Gender (Factor or numeric: 1:Male; 2:Female).

**charlson** Comorbidity Charlson's index at baseline (Factor or numeric: 0: Index 0; 1: Index 1-2; 3: Index >=3). Note: Original data had this as time-dependent, this version likely uses the baseline value.

**chemo** Indicator whether patient received chemotherapy (Factor or numeric: 1:No; 2:Yes).

**group** An example grouping variable (numeric, derived from id mod 10 + 1), useful for fitting grouped frailty models.

## Details

The transformation process involved:

1. Starting with the original `readmission` data.
2. Excluding recurrent rehospitalization events, keeping only the interval from surgery (t.start=0) to the first event (event=1) or censoring (event=0).
3. Reshaping the data so each row represents one subject.
4. Defining `observed_disease_time` and `disease_status` based on the first event interval (t.stop when t.start=0).
5. Defining `observed_death_time` and `death_status` based on the overall follow-up time and final death status for the subject. If a subject had a first event and then further follow-up, the death time comes from the second interval if available.
6. Scaling time variables (`t.stop`) from days (assumed) to years by dividing by 365.
7. Copying baseline covariates (dukes, sex, charlson, chemo) from the subject's first record.

This dataset is intended primarily for demonstrating the `IllnessDeath` function.

## Source

Derived from the `readmission` dataset, originally described in: Gonzalez, JR., Fernandez, E., Moreno, V., Ribes, J., Peris, M., Navarro, M., Cambray, M. and Borras, JM (2005). Sex differences in hospital readmission among colorectal cancer patients. *Journal of Epidemiology and Community Health*, **59**, 6, 506-511.

| reduce | *Delirium in critically ill ICU patients dataset: the REDUCE clinical trial* |
|---|---|

## Description

This dataset contains an extract of 500 randomly selected patients from the randomized, double-blind, placebo-controlled REDUCE trial for critically ill patient admited to ICU. This trial investigated whether Haloperidol (1 or 2 mg) administered as a prophylactic improved 28-day survival compared to placebo. Recurrent episodes of delirium are recorded and patients and patients can be censored by death or discharge from the ICU.

## Usage

```
data(reduce)
```

## Format

This data frame contains the following columns:

**id** Identification number of a patient

**t.start** Start time of the interval (0 or time of last recurrence)

**t.stop** Stop time of the interval, either delirium recurrence time or censoring time.

**del** Delirium status

**death** Death status

**discharge** Discharge status

**treatment** Treatment indicator, 1 if patient was randomized to receive 2mg of Haloperidol, 0 for control

## Source

Van Den Boogaard, M., Slooter, A. J., Bruggemann, R. J., Schoonhoven, L., Beishuizen, A., Vermeijden, J. W., et al. (2018). Effect of haloperidol on survival among critically ill adults with a high risk of delirium: the REDUCE randomized clinical trial. *Jama*, **319(7)**, 680-690.

| runShiny | *Shiny application for modelisation and prediction of frailty models* |
|---|---|

## Description

This function loads the shiny package and runs the application for modelisation and prediction of several frailty models using package frailtypack.

## Usage

```
runShiny()
```

## Value

No value returned.

## References

Rizopoulos D. (2016)

## Examples

```
## Not run:

runShiny()


## End(Not run)
```

---

simulatejointRecCompet

*Generating from a joint competing Joint frailty model with a recurrent event and two terminal events.*

---

## Description

Generates data under a joint frailty model for a single recurrent event and two terminal events in a calendar-time format. Only a single covariate representing the treatment is allowed. Event times are generated under Weibull distributions.

## Usage

```
simulatejointRecCompet(n, censoring = 28, maxrecurrent = 50,
           par0 = c(shapeR = 1.5, scaleR = 10,
           shapeM = 1.75, scaleM = 16,
           shapeD = 1.75, scaleD = 16,
           sigma = 0.5, alphaM = 1, alphaD = 0,
           betaR = -0.5, betaM = -0.5, betaD = 0))
```

## Arguments

| | |
|---|---|
| n | Number of subjects. Default is 1500. |
| censoring | A number indicating a fixed right censoring time for all subjects (as an administrative censoring). If NULL, no censoring is applied. Default is 28. |
| maxrecurrent | Maximum number of recurrent events per subject. If NULL, no upper bound is set for the number of of recurrent events Default is 50. |

`par0` A vector of arguments controlling the parameters of the generating model.

- `shapeR`: `shape` parameter of the Weibull distribution for the recurrent event
- `scaleR`: `scale` parameter of the Weibull distribution for the recurrent event
- `shapeT1`: `shape` parameter of the Weibull distribution for the first terminal event
- `scaleT1`: `scale` parameter of the Weibull distribution for the first terminal event
- `shapeT2`: `shape` parameter of the Weibull distribution for the second terminal event
- `scaleT2`: `scale` parameter of the Weibull distribution for the second terminal event
- `sigma`: Standard deviation of the frailty
- `alphaT1`: Power parameter (link) of the frailty for the first terminal event
- `alphaT2`: Power parameter (link) of the frailty for the second terminal event
- `betaR`: Association parameter for the treatment effect on the recurrent event
- `betaT1`: Association parameter for the treatment effect on the first terminal event
- `betaT2`: Association parameter for the treatment effect on the second terminal event

## Value

Returns a `data.frame` object with the following columns:

- `id` Id number for each subject
- `treatment` Binary treatment indicator
- `tstart` Start time of the observation period
- `tstop` Stop time of the observation period
- `recurrent` Censoring indicator for the recurrent event
- `terminal1` Censoring indicator for the first terminal event
- `terminal2` Censoring indicator for the second terminal event

## See Also

jointRecCompet

slope *Identify variable associated with the random slope*

## Description

This is a special function used in the context of survival additive models. It identifies the variable which is in interaction with the random slope ($v_i$). Generally, this variable is the treatment variable. Using `interaction()` in a formula implies that an additive frailty model is fitted.

## Usage

```
slope(x)
```

## Arguments

| | |
|---|---|
| x | A factor, a character or a numerical variable |

## Value

| | |
|---|---|
| x | The variable in interaction with the random slope |

## Note

It is necessary to specify which variable is in interaction with the random slope, even if only one explanatory variable is included in the model.

## See Also

additivePenal

## Examples

```
data(dataAdditive)

##-- Additive with one covariate --##

modAdd1cov <- additivePenal(Surv(t1,t2,event)~cluster(group)+var1+
slope(var1),data=dataAdditive,n.knots=8,kappa=10000,hazard="Splines")

##-- Additive with two covariates --##

set.seed(1234)
dataAdditive$var2 <- rbinom(nrow(dataAdditive),1,0.5)

modAdd2cov <- additivePenal(Surv(t1,t2,event)~cluster(group)+var1+
var2+slope(var1),data=dataAdditive,n.knots=8,kappa=10000,
```

```
hazard="Splines")

##-- Additive with 2 covariates and stratification --##

dataAdditive$var2 <- rbinom(nrow(dataAdditive),1,0.5)

modAddstrat <- additivePenal(Surv(t1,t2,event)~cluster(group)+
strata(var2)+var1+slope(var1),data=dataAdditive,n.knots=8,
kappa=c(10000,10000),hazard="Splines")
```

---

`ste` *Surrogate threshold effect for the one-step Joint surrogate model for the evaluation of a canditate surrogate endpoint.*

---

## Description

This function compute the surrogate threshold effect (STE) from the one-step joint frailty model or joint frailty-copula model. The STE is defined as the minimum treament effect on surrogate endpoint, necessary to predict a non-zero effect on true endpoint (Burzykowski *et al.*, 2006).

## Usage

```
ste(object, var.used = "error.estim", alpha. = 0.05,
    pred.int.use = "up")
```

## Arguments

`object` An object inheriting from `jointSurroPenal` class (output from calling the `jointSurroPenal` or `jointSurroCopPenal` function ).

`var.used` This argument takes two values. The first one is `"error.estim"` and indicates if the prediction error takes into account the estimation error of the estimates of the parameters. If the estimates are supposed to be known or if the dataset includes a high number of trials with a high number of subject per trial, value `No.error` can be used. The default is `error.estim`, which is highly recommended in practice.

`alpha.` The confidence level for the prediction interval. The default is `0.05`

`pred.int.use` A character string that indicates the bound of the prediction interval to use to compute the STE. Possible values are `up` for the upper bound (the default) or `lw` for the lower bound. `up` when we have a protective treatment effect and `lw` when we have a deleterious treatment effect (see details).

## Details

The STE is obtained by solving the equation $l(\alpha_0)$ = 0 (resp. $u(\alpha_0)$ = 0), where $\alpha_0$ represents the corresponding STE, and $l(\alpha_0)$ (resp. $u(\alpha_0)$ ) is the lower (resp. upper) bound of the prediction interval of the treatment effect on the true endpoint ($\beta$ + $b_0$). Thereby,

$l(\alpha_0) \equiv E(\beta + b_0|\alpha_0, \vartheta) - Z_{1-(\gamma/2)}\sqrt(Var(\beta + b_0|\alpha_0, \vartheta))$

and

$u(\alpha_0) \equiv E(\beta + b_0|\alpha_0, \vartheta) + Z_{1-(\gamma/2)}\sqrt(Var(\beta + b_0|\alpha_0, \vartheta))$

where $\vartheta$ represents the set of estimates for the fixed-effects and the variance-covariance parameters of the random effects obtained from the joint surrogate `model` (Sofeu *et al.*, 2019).

If the previous equations gives two solutions, STE can be the minimum (resp. the maximum) value or both of them, according to the shape of the function. If the concavity of the function is turned upwards, STE is the first value and the second value represents the maximum (res. the minimum) treament value observable on the surrogate that can predict a nonzero treatment effect on true endpoint. If the concavity of the function is turned down, both of the solutions represent the STE and the interpretation is such that accepted values of the treatment effects on `S` predict a nonzero treatment effects on `T`

Given that negative values of treatment effect indicate a reduction of the risk of failure and are considered beneficial, STE is recommended to be computed from the upper prediction limit $u(\alpha_0)$ .

The details on the computation of STE are described in Burzykowski *et al.* (2006).

## Value

Returns and displays the STE.

## Author(s)

Casimir Ledoux Sofeu `<casimir.sofeu@u-bordeaux.fr>`, `<scl.ledoux@gmail.com>` and Virginie Rondeau `<virginie.rondeau@inserm.fr>`

## References

Burzykowski T, Buyse M (2006). "Surrogate threshold effect: an alternative measure for meta-analytic surrogate endpoint validation." Pharmaceutical Statistics, 5(3), 173-186.ISSN 1539-1612.

Sofeu, C. L., Emura, T., and Rondeau, V. (2019). One-step validation method for surrogate endpoints using data from multiple randomized cancer clinical trials with failure-time endpoints. Statistics in Medicine 38, 2928-2942.

Sofeu, C. L. and Rondeau, V. (2020). How to use frailtypack for validating failure-time surrogate endpoints using individual patient data from meta-analyses of randomized controlled trials. PLOS ONE; 15, 1-25.

## See Also

`jointSurroPenal`, `jointSurroCopPenal`, `predict`

## Examples

```
###--- Joint surrogate model ---###
###---evaluation of surrogate endpoints---###

data(dataOvarian)
joint.surro.ovar <- jointSurroPenal(data = dataOvarian, n.knots = 8,
                init.kappa = c(2000,1000), indicator.alpha = 0,
                nb.mc = 200, scale = 1/365)

# ======STE=====
# Assuming errors on the estimates
ste(joint.surro.ovar, var.used = "error.estim")
# Assuming no errors on the estimates
ste(joint.surro.ovar, var.used = "No.error", pred.int.use = "up")
```

---

subcluster *Identify subclusters*

---

## Description

This is a special function used in the context of survival nested or joint nested models. It identifies correlated groups of observations within other groups defined by using 'cluster' function from 'survival' package, and is used on the right hand side of 'frailtyPenal' formula for fitting a nested or joint nested model. Using `subcluster()` in a formula implies that a nested or a joint nested frailty model is estimated.

## Usage

```
subcluster(x)
```

## Arguments

| | |
|---|---|
| x | A character, factor, or numeric variable which is supposed to indicate the variable subgroup |

## Value

| | |
|---|---|
| x | A variable identified as a subcluster |

## See Also

frailtyPenal

## Examples

```
## Not run:

data(dataNested)
modClu <- frailtyPenal(Surv(t1,t2,event)~cluster(group)+
subcluster(subgroup)+cov1+cov2,data=dataNested,
n.knots=8,kappa=c(50000,50000),hazard="Splines")

print(modClu)

#-- here is generated cluster (30 clusters)
readmissionNested <- transform(readmission,group=id%%30+1)

modJointNested_Splines <- frailtyPenal(formula = Surv(t.start, t.stop, event)
~ subcluster(id) + cluster(group) + dukes +
terminal(death), formula.terminalEvent = ~dukes,
data = readmissionNested, recurrentAG = TRUE, n.knots = 8,
kappa = c(9.55e+9, 1.41e+12), initialize = TRUE)


## End(Not run)
```

---

# summary.additivePenal *summary of parameter estimates of an additive frailty model*

---

## Description

This function returns hazard ratios (HR) and its confidence intervals

## Usage

```
## S3 method for class 'additivePenal'
summary(object, level = 0.95, len = 6, d = 2,
lab="hr", ...)
```

## Arguments

| | |
|---|---|
| `object` | output from a call to additivePenal. |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

## Value

Prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument)

## See Also

additivePenal

## Examples

```
## Not run:

data(dataAdditive)

modAdd <- additivePenal(Surv(t1,t2,event)~cluster(group)+var1+slope(var1),
correlation=TRUE,data=dataAdditive,n.knots=8,kappa=862,hazard="Splines")

#- 'var1' is boolean as a treatment variable.

summary(modAdd)


## End(Not run)
```

---

summary.frailtyCmprsk *summary of parameter estimates of a Weibull competing risks model with (or without) shared frailty between transitions.*

---

## Description

This function returns hazard rations (HR) and its confidence intervals

## Usage

```
## S3 method for class 'frailtyCmprsk'
summary(object, level = 0.95, len = 6, d = 2,
                                    lab="hr",...)
```

## Arguments

| | |
|---|---|
| object | output from a call to frailtyCmprsk. |
| level | significance level of confidence interval. Default is 95%. |
| len | the total field width. Default is 6. |
| d | the desired number of digits after the decimal point. Default of 6 digits is used. |
| lab | label of printed results. |
| ... | other unused arguments. |

## Value

Prints HR and its confidence intervals. Confidence level is allowed (level argument).

## See Also

frailtyCmprsk

## Examples

```
###--- Simple Weibull competing risks model ---###

###--- Weibull competing risks model with shared frailty between transitions ---###

data(CPRSKbmtcrr)

##--- Simple Weibull competing risks model with left truncation ---##

modCmprskFrailty <- frailtyCmprsk(
  formulas = list(
    Surv(Age, observed_time, Status, type = "mstate") ~ Sex,
    ~ Sex
  ),
  data = CPRSKbmtcrr,
  print.info = FALSE,
  maxit = 100
)

#-- Confidence interval at 95% level (default)

summary(modCmprskFrailty)

#-- Confidence interval at 99% level

summary(modCmprskFrailty, level = 0.99)
```

---

summary.frailtyDesign    *Summarize a 'frailtyDesign' object.*

---

## Description

Summarize a 'frailtyDesign' object.

## Usage

```
## S3 method for class 'frailtyDesign'
summary(object, digits = 2, ...)
```

### Arguments

| | |
|---|---|
| `object` | an object of class 'frailtyDesign' (output from one of the *.power or *.ssize functions). |
| `digits` | number of decimals to print for numeric fields. Default is 2. |
| `...` | other unused arguments. |

### See Also

frailtyDesign

### Examples

```
est.ex <- SFM.power(
  Groups = 400, ni = 3, ni.type = "max", FUP = 6, Acc.Dur = 0.5, median.H0 = 1.5,
 beta.HA = log(0.7), theta = 0.5, cens.par = c(3, 10), cens.type = "Unif", data.type = "rec_event"
)

summary(est.ex)
```

---

## summary.frailtyIllnessDeath

*Summary of parameter estimates of a Weibull Illness-Death model with (or without) shared frailty between transitions.*

---

### Description

This function returns hazard rations (HR) and its confidence intervals

### Usage

```
## S3 method for class 'frailtyIllnessDeath'
summary(object, level = 0.95, len = 6, d = 2,
                                    lab="hr",...)
```

### Arguments

| | |
|---|---|
| `object` | output from a call to frailtyIllnessDeath. |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

## Value

Prints HR and its confidence intervals. Confidence level is allowed (level argument).

## See Also

frailtyIllnessDeath

## Examples

```
###--- Semi-Markovian Weibull Illness-Death model with left truncation ---###

data(Paq810)

ModIllnessDeath_LeftTrunc <- frailtyIllnessDeath(formula = Surv(e,r,dementia) ~ gender+certif,
formula.terminalEvent = Surv(t,death) ~ gender+certif  ,
data=Paq810, print.info = FALSE, maxit=100)

#-- confidence interval at 95\% level (default)

summary(ModIllnessDeath_LeftTrunc)

#-- confidence interval at 99\% level

summary(ModIllnessDeath_LeftTrunc,level=0.99)
```

---

summary.frailtyPenal *summary of parameter estimates of a shared frailty model*

---

## Description

This function returns hazard rations (HR) and its confidence intervals

## Usage

```
## S3 method for class 'frailtyPenal'
summary(object, level = 0.95, len = 6, d = 2,
lab="hr", ...)
```

### Arguments

| | |
|---|---|
| `object` | output from a call to frailtyPenal. |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

### Value

Prints HR and its confidence intervals. Confidence level is allowed (level argument).

### See Also

`frailtyPenal`

### Examples

```
## Not run:

data(kidney)

##-- Shared frailty model --##

modSha <- frailtyPenal(Surv(time,status)~age+sex+cluster(id),
n.knots=8,kappa=10000,data=kidney,hazard="Splines")

##-- Cox proportional hazard model --##

modCox <- frailtyPenal(Surv(time,status)~age+sex,
n.knots=8,kappa=10000,data=kidney,hazard="Splines")

#-- confidence interval at 95% level (default)

summary(modSha)
summary(modCox)

#-- confidence interval at 99% level

summary(modSha,level=0.99)
summary(modCox,level=0.99)


## End(Not run)
```

`summary.jointNestedPenal`
*summary of parameter estimates of a joint nested frailty model*

## Description

This function returns hazard rations (HR) and its confidence intervals.

## Usage

```
## S3 method for class 'jointNestedPenal'
summary(object, level = 0.95, len = 6, d =
2, lab="hr", ...)
```

## Arguments

| | |
|---|---|
| `object` | output from a call to frailtyPenal for joint nested models |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

## Value

Prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument).

## See Also

`frailtyPenal`

## Examples

```

## Not run:

#-- here is generated cluster (30 clusters)
readmissionNested <- transform(readmission,group=id%%30+1)

# Baseline hazard function approximated with splines with calendar-timescale

model.spli.AG <- frailtyPenal(formula = Surv(t.start, t.stop, event)
 ~ subcluster(id) + cluster(group) + dukes + terminal(death),
 formula.terminalEvent = ~dukes, data = readmissionNested,
 recurrentAG = TRUE, n.knots = 8, kappa = c(9.55e+9, 1.41e+12),
 initialize = TRUE)
```

```
summary(model.spli.AG)

## End(Not run)
```

---

`summary.jointPenal` *summary of parameter estimates of a joint frailty model*

---

## Description

This function returns hazard rations (HR) and its confidence intervals.

## Usage

```
## S3 method for class 'jointPenal'
summary(object, level = 0.95, len = 6, d = 2,
lab="HR", ...)
```

## Arguments

| | |
|---|---|
| `object` | output from a call to frailtyPenal for joint models |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

## Value

Prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument).

## See Also

`frailtyPenal`

## Examples

```
## Not run:

data(readmission)

#-- gap-time
modJoint.gap <- frailtyPenal(
  Surv(time, event) ~ cluster(id) + sex + dukes +
```

```
    charlson + terminal(death),
  formula.terminalEvent = ~ sex + dukes + charlson,
  data = readmission, n.knots = 14, kappa = c(9.55e+9, 1.41e+12)
)

#-- calendar time
modJoint.calendar <- frailtyPenal(
  Surv(t.start, t.stop, event) ~ cluster(id) +
    sex + dukes + charlson + terminal(death),
  formula.terminalEvent = ~ sex + dukes + charlson,
  data = readmission, n.knots = 10, kappa = c(9.55e+9, 1.41e+12), recurrentAG = TRUE
)

#-- It takes around 1 minute to converge

summary(modJoint.gap)
summary(modJoint.calendar)

## End(Not run)
```

---

summary.jointRecCompet    *Summary method for a joint competing risks midel*

---

## Description

Prints a short summary of parameter estimates of a joint competing risks model or more generally an object of class 'jointRecCompet'.

## Usage

```
## S3 method for class 'jointRecCompet'
summary(object, digits = max(options()$digits - 4, 6),
...)
```

## Arguments

| | |
|---|---|
| object | the result of a call to the jointRecCompet function |
| digits | number of digits to print |
| ... | other unused arguments |

## Value

Print, separately for each type of event (Recurrent, Terminal1 and Terminal2), the parameter estimates of the survival or hazard functions.

## See Also

jointRecCompet

`summary.jointSurroMed` *Short summary of the random effects parameters, the fixed treatment effects, and the surrogacy evaluation criteria estimated from a joint surrogate mediation model*

## Description

This function returns the estimate of the coefficients of the model, their standard error and the associated p-values of the Wald test for the joint surrogate model, also hazard ratios (HR) and their confidence intervals for the fixed treatment effects. It also displays summary of the surrogacy measure $PTE(t)$ and of the natural direct, indirect and total effect.

## Usage

```
## S3 method for class 'jointSurroMed'
summary(object,d=4,len=3,n=3,...)
```

## Arguments

| | |
|---|---|
| `object` | An object inheriting from `jointSurroMed` class. |
| `d` | The desired number of digits after the decimal point for parameters. The maximum of 4 digits is required for the estimates. Default of 3 digits is used. |
| `len` | The desired number of digits after the decimal point for p-value and convergence criteria. Default of 4 digits is used. |
| `n` | The number of time points to be used in the results of the differents function related to the mediation analysis: $g(s)$, $PTE(t)$ and the direct, indirect and total effect. The provided value should be between 1 and 20. Default is 3. |
| `...` | other unused arguments. |

## Value

For the variances parameters of the random effects, it prints the estimate of the coefficients with their standard error, Z-statistics and p-values of the Wald test. For the fixed treatment effects, it also prints HR and its confidence intervals for each covariate. For the surrogacy assessment, prints `n` value of the estimation function $g(s)$ and $PTE(t)$. Also prints the values of the estimated direct, indirect and total effects. The remaining displayed information concern the convergence characteristics and include the penalized marginal log-likelihood, the number of iterations, the LCV and the convergence criteria.

## See Also

`jointSurroPenal`

---

summary.jointSurroPenal

*Short summary of the surrogacy evaluation criteria estimated from a joint surrogate model*

---

## Description

This function returns the estimate of the coefficient, the hazard ratios (HR) and their confidence intervals for the fixed treatment effects. Also, an estimate of the surrogacy evaluation criteria (Kendall's $\tau$, $R^2_{trial}$ and STE)

## Usage

```
## S3 method for class 'jointSurroPenal'
summary(object, d = 4, len = 3, nb.gh = 32, ...)
```

## Arguments

| | |
|---|---|
| `object` | An object inheriting from `jointSurroPenal` class. |
| `d` | The desired number of digits after the decimal point for parameters. The maximum of 4 digits is required for the estimates. Default of 3 digits is used. |
| `len` | The desired number of digits after the decimal point for p-value and convergence criteria. Default of 4 digits is used. |
| `nb.gh` | Number of nodes for the Gaussian-Hermite quadrature. The default is `32 1` for Gaussian-Hermite quadrature. |
| `...` | other unused arguments. |

## Value

For the fixed treatment effects, it also prints HR and its confidence intervals for each covariate. For the surrogacy evaluation criteria, its prints the estimated Kendall's $\tau$ with its 95% Confidence interval obtained by the parametric bootstrap or Delta-method, the estimated $R^2_{trial}$ (R2trial) with standard error and the 95% Confidence interval obtained by Delta-method (Dowd *et al.*, 2014), $R^2_{trial}$ (R2.boot) and its 95% Confidence interval obtained by the parametric bootstrap. We notice that, using bootstrap, the standard error of the point estimate is not available. We propose a classification of $R^2_{trial}$ according to the suggestion of the Institute of Quality and Efficiency in Health Care (Prasad *et al.*, 2015). We also display the surrogate threshold effect (`ste`) with the associated hazard risk.

## Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

### References

Dowd BE, Greene WH, Norton EC (2014). "Computation of Standard Errors." Health Services Research, 49(2), 731-750.

Prasad V, Kim C, Burotto M, Vandross A (2015). "The strength of association between surrogate end points and survival in oncology: A systematic review of trial-level meta- alyses." JAMA Internal Medicine, 175(8), 1389-1398.

### See Also

jointSurroPenal, jointSurroCopPenal, jointSurroTKendall, print.jointSurroPenal

### Examples

```
## Not run:

###---Data generation---###
data.sim <-jointSurrSimul(n.obs=400, n.trial = 20,cens.adm=549,
          alpha = 1.5, theta = 3.5, gamma = 2.5, zeta = 1,
          sigma.s = 0.7, sigma.t = 0.7, cor = 0.8, betas = -1.25,
          betat = -1.25, full.data = 0, random.generator = 1,
          seed = 0, nb.reject.data = 0)

###---Estimation---###
joint.surrogate <- jointSurroPenal(data = data.sim, nb.mc = 300,
                   nb.gh = 20, indicator.alpha = 1, n.knots = 6)

summary(joint.surrogate)

## End(Not run)
```

---

## summary.jointSurroPenalSimul

*Short summary of the simulation studies based on a joint surrogate model*

---

### Description

This function returns the true value, the mean of the estimates, the empirical standard error, the mean of the estimated standard errors (Mean SE), and the coverage probability for model parameters

### Usage

```
## S3 method for class 'jointSurroPenalSimul'
summary(object, d = 3, R2boot = 0, displayMSE = 0, printResult = 1, CP = 0,  ...)
```

### Arguments

| | |
|---|---|
| `object` | an object inheriting from `jointSurroPenalSimul` class. |
| `d` | The desired number of digits after the decimal point f. Default of 3 |
| `R2boot` | A binary that specifies whether the confidence interval of $R^2_{trial}$ should be computed using parametric bootstrap (1) or Delta-method (`0`). The default is `0` |
| `displayMSE` | A binary that indicates if the results include bias and mean square errors (MSE), case 1, or the standard errors with the coverage percentage, case 0. By default this argument is set to 0. In the event of 1 the results just include the individual level and the trial level association measurements. |
| `printResult` | A binary that indicates if the summary of the results should be displayed `(1)` or not `(0)`. If this argument is set to 0, resuls are just returned to the user |
| `CP` | A binary that indicate in the event of `displayMSE = 1` if the percentage of coverage should be display (1) or not (0). The default is 0 |
| `...` | other unused arguments. |

### Value

For each parameter of the joint surrogate model , we print the true simulation value, the empirical standard error (empirical SE), the mean of the estimated standard errors (Mean SE), and the coverate probability (CP). For Kendall's $\tau$, the 95% Confidence interval is obtained by parametric bootstrap (for joint frailty model) or Delta-method (for joint frailty-copula model). For $R^2_{trial}$ (R2trial), the standard error is obtained by Delta-method and the 95% Confidence interval could be obtained directly or by parametric bootstrap. We also display the total number of non convergence case with the associated percentage (R : n(%)), the mean number of iterations to reach convergence, and other estimation and simulation parameters. We also return a dataframe of the simulations results .

### Author(s)

Casimir Ledoux Sofeu <casimir.sofeu@u-bordeaux.fr>, <scl.ledoux@gmail.com> and Virginie Rondeau <virginie.rondeau@inserm.fr>

### See Also

jointSurroPenalSimul

### Examples

```
# Studies simulation
## Not run:
# (Computation takes around 45 minutes using a processor including 40
# cores and a read only memory of 378 Go)
joint.simul <- jointSurroPenalSimul(nb.dataset = 10, nbSubSimul=600,
                   ntrialSimul=30, LIMparam = 0.001, LIMlogl = 0.001,
                   LIMderiv = 0.001, nb.mc = 200, nb.gh = 20,
                   nb.gh2 = 32, true.init.val = 1, print.iter=F)

# results
```

```
summary(joint.simul, d = 3, R2boot = 1) # bootstrap
summary(joint.simul, d = 3, R2boot = 0) # Delta-method


## End(Not run)
```

---

`summary.longiPenal` *Short summary of fixed covariates estimates of a joint model for longitudinal data and a terminal event.*

---

## Description

This function returns coefficients estimates and their standard error with p-values of the Wald test for the longitudinal outcome and hazard ratios (HR) and their confidence intervals for the terminal event. If a mediation analysis was performed (option `mediation` set to `TRUE` in `longiPenal`) this function displays estimations of the related quantities.

## Usage

```
## S3 method for class 'longiPenal'
summary(object, level = 0.95, len = 6, d = 2,
lab=c("coef","hr"), ...)
```

## Arguments

| | |
|---|---|
| `object` | an object inheriting from `longiPenal` class |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width for the terminal part. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | labels of printed results for the longitudinal outcome and the terminal event respectively. |
| `...` | other unused arguments. |

## Value

For the longitudinal outcome it prints the estimates of coefficients of the fixed covariates with their standard error and p-values of the Wald test. For the terminal event it prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument).

## See Also

`longiPenal`

## Examples

```
## Not run:
###--- Joint model for longitudinal data and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

# Baseline hazard function approximated with splines
# Random effects as the link function

model.spli.RE <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
colorectalSurv,data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Random-effects", left.censoring = -3.33,
n.knots = 7, kappa = 2)

# Weibull baseline hazard function
# Current level of the biomarker as the link function

model.weib.CL <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
colorectalSurv, data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Current-level", left.censoring = -3.33, hazard = "Weibull")

summary(model.spli.RE)
summary(model.weib.CL)

## End(Not run)
```

---

`summary.multivPenal` *summary of parameter estimates of a multivariate frailty model.*

---

## Description

This function returns hazard ratio (HR) and its confidence intervals.

## Usage

```
## S3 method for class 'multivPenal'
summary(object, level = 0.95, len = 6, d = 2, lab
= "hr", ...)
```

### Arguments

| | |
|---|---|
| `object` | output from a call to multivPenal for joint multivariate models |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

### Value

Prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument)

### See Also

multivPenal

---

## `summary.nestedPenal` *summary of regression coefficient estimates of a nested frailty model*

---

### Description

This function returns hazard rations (HR) and its confidence intervals for each regression coefficient.

### Usage

```
## S3 method for class 'nestedPenal'
summary(object, level = 0.95, len = 6, d = 2,
lab="hr", ...)
```

### Arguments

| | |
|---|---|
| `object` | output from a call to nestedPenal. |
| `level` | significance level of confidence interval. Default is 95%. |
| `len` | the total field width. Default is 6. |
| `d` | the desired number of digits after the decimal point. Default of 6 digits is used. |
| `lab` | label of printed results. |
| `...` | other unused arguments. |

### Value

Prints HR and its confidence intervals for each regression coefficient. Confidence level is allowed (level argument).

## See Also

frailtyPenal

## Examples

```
## Not run:

data(dataNested)

modNested <- frailtyPenal(Surv(t1,t2,event)~cluster(group)+
subcluster(subgroup)+cov1+cov2,data=dataNested,
n.knots=8,kappa=c(50000,50000),hazard="Splines")

#- It takes 90 minutes to converge (depends on processor)

summary(modNested)


## End(Not run)
```

---

summary.trivPenal — *Short summary of fixed covariates estimates of a joint model for longitudinal data, recurrent events and a terminal event*

---

## Description

This function returns coefficients estimates and their standard error with p-values of the Wald test for the longitudinal outcome and hazard ratios (HR) and their confidence intervals for the terminal event.

## Usage

```
## S3 method for class 'trivPenal'
summary(object, level = 0.95, len = 6, d = 2,
lab=c("coef","hr"), ...)
```

## Arguments

| | |
|---|---|
| object | an object inheriting from `trivPenal` class |
| level | significance level of confidence interval. Default is 95%. |
| len | the total field width for the terminal part. Default is 6. |
| d | the desired number of digits after the decimal point. Default of 6 digits is used. |
| lab | labels of printed results for the longitudinal outcome and the terminal event respectively. |
| ... | other unused arguments. |

## Value

For the longitudinal outcome it prints the estimates of coefficients of the fixed covariates with their standard error and p-values of the Wald test. For the terminal event it prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument).

## See Also

trivPenal

## Examples

```
## Not run:

###--- Trivariate joint model for longitudinal data, ---###
###--- recurrent events and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Weibull baseline hazard function
# Random effects as the link function, Gap timescale
# (computation takes around 30 minutes)
model.weib.RE.gap <-trivPenal(Surv(gap.time, new.lesions) ~ cluster(id)
+ age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive", n.nodes = 7)

summary(model.weib.RE.gap)

## End(Not run)
```

---

| summary.trivPenalNL | *Short summary of fixed covariates estimates of a non-linear trivariate joint model for longitudinal data, recurrent events and a terminal event* |
|---|---|

## Description

This function returns coefficients estimates and their standard error with p-values of the Wald test for the biomarker growth (KG) and decline (KD) and hazard ratios and their confidence intervals for the terminal event.

### Usage

```
## S3 method for class 'trivPenalNL'
summary(object, level = 0.95, len = 6, d = 2,
lab=c("coef","hr"), ...)
```

### Arguments

| | |
|---|---|
| object | an object inheriting from trivPenal class |
| level | significance level of confidence interval. Default is 95%. |
| len | the total field width for the terminal part. Default is 6. |
| d | the desired number of digits after the decimal point. Default of 6 digits is used. |
| lab | labels of printed results for the longitudinal outcome and the terminal event respectively. |
| ... | other unused arguments. |

### Value

For the longitudinal outcome it prints the estimates of coefficients of the fixed covariates with their standard error and p-values of the Wald test (separetely for the biomarker growth and decline). For the terminal event it prints HR and its confidence intervals for each covariate. Confidence level is allowed (level argument).

### See Also

trivPenalNL

### Examples

```
## Not run:

###--- Trivariate joint model for longitudinal data, ---###
###--- recurrent events and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Weibull baseline hazard function
# Random effects as the link function, Gap timescale
# (computation takes around 30 minutes)
model.weib.RE.gap <-trivPenal(Surv(gap.time, new.lesions) ~ cluster(id)
+ age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive", n.nodes = 7)

summary(model.weib.RE.gap)
```

```
## End(Not run)
```

---

`survDat` *Survival dataset (TPJM)*

---

## Description

This is a simulated dataset used to illustrate the two-part joint model included in the longiPenal function.

## Usage

```
data(survDat)
```

## Format

This data frame contains the following columns:

**id** The identification number of a patient

**deathTimes** The event times (death or censoring)

**d** Censoring indicator

**trt** Treatment covariate

---

`SurvIC` *Create a survival object for interval censoring and possibly left truncated data*

---

## Description

This is a function used in case of interval-censoring as a response variable in a model formula only for Cox proportional hazard or shared frailty model. Sometimes, an unobserved event might occur in a time interval [L,U]. RecurrentAG argument gets invalid with the use of SurvIC. Note that this function used a Kronecker product which can suffer from computation issue when the number of subjects in each cluster is high. Time dependent variables are not allowed.

## Usage

```
SurvIC(t0, lower, upper, event)
```

### Arguments

| | |
|---|---|
| t0 | Truncation time for left truncated data only. To be ignored otherwise. |
| lower | Starting time of the interval for interval-censored data. Time of right-censoring instead. |
| upper | Ending time of the interval for interval-censored data. For right-censored data, lower and upper time must be equal (for numerical reason). |
| event | Status indicator 0=right-censored, 1=interval-censored |

### Details

Typical usages are `SurvIC(lower,upper,event)` or `SurvIC(t0,lower,upper,event)`

### Value

No return value

### Examples

```
data(bcos)
bcos$event <- ifelse(bcos$left!=bcos$right,1,0)

###---  Cox proportional hazard model with interval censoring ---###

cox.ic <- frailtyPenal(SurvIC(left,right,event)~treatment,
data=bcos,n.knots=8,kappa=10000)

###---  Shared model with interval censoring ---###

bcos$group <- c(rep(1:20,4),1:14)

sha.ic <- frailtyPenal(SurvIC(left,right,event)~cluster(group)+
treatment,data=bcos,n.knots=8,kappa=10000)
```

---

## `survival` *Survival function*

---

### Description

Let t be a continuous variable, we determine the value of the survival function to t after run fit.

## Usage

```
survival(t, ObjFrailty)
```

## Arguments

| | |
|---|---|
| t | time for survival function. |
| ObjFrailty | an object from the frailtypack fit. |

## Value

return the value of survival function in t.

## Examples

```
## Not run:

#-- a fit Shared
data(readmission)

fit.shared <- frailtyPenal(Surv(time,event)~dukes+cluster(id)+
strata(sex),n.knots=10,kappa=c(10000,10000),data=readmission)

#-- calling survival
survival(20,fit.shared)


## End(Not run)
```

---

# terminal *Identify terminal indicator*

---

## Description

This is a special function used in the context of recurrent event models with terminal event (e.g., censoring variable related to recurrent events). It contains the status indicator, normally 0=alive, 1=dead, and is used on the right hand side of a formula of a 'frailtyPenal', 'longiPenal' and 'trivPenal' functions. Using `terminal()` in a formula implies that a joint frailty model for recurrent events and terminal events is fitted.

## Usage

```
terminal(x)
```

### Arguments

| | |
|---|---|
| `x` | A numeric variable but should be a Boolean which equals 1 if the subject is dead and 0 if he is alive or censored, as a death indicator. |

### Value

| | |
|---|---|
| `x` | a death indicator |

### See Also

`frailtyPenal`

---

`timedep` *Identify time-varying effects*

---

### Description

This is a special function used in the context of Cox models and shared and joint frailty models. It identifies time-varying effects of covariates in the model. It is used in 'frailtyPenal' on the right hand side of formula or of formula.terminalEvent.

When considering time-varying effects in a survival model, regression coefficients can be modeled with a linear combination of B-splines $B(t)$ with coefficients $\zeta$ of order $q$ with $m$ interior knots :

$$\beta(t) = \sum_{j=-q+1}^{m} \zeta_j B_{j,q}(t)$$

You can notice that a linear combination of B-splines of order 1 without any interior knots (0 interior knot) is the same as a model without time-varying effect (or with constant effect over time).

Statistical tests (likelihood ratio tests) can be done in order to know whether the time-dependent coefficients are significantly different from zero or to test whether a covariate has a time-dependent effect significantly different from zero or not. These tests are correct only with a parametric approach yet.

- Proportional Hazard assumption ?

Time-dependency of a covariate effect can be tested. We need to estimate $m + q$ parameters $\zeta_j$ for $j = -q + 1, ..., m$ for a time-varying coefficient. Only one ($q = 1$,$m = 0$) parameter is estimated for a constant effect. A global test is done.

$$H_0 : \beta(t) = \beta$$

The corresponding LR statistic has a $\chi^2$ distribution of degree $m + q - 1$.

- Significant association ?

We can also use a LR test to test whether a covariate has a significant effect on the hazard function. The null hypothesis is :

$$H_0 : \beta(t) = 0$$

For that we fit a model considering the covariate with a regression coefficent modeled using B-splines and a model without the covariate. Hence, the LR statistic has a $\chi^2$ distribution of degree $m + q$.

## Usage

```
timedep(x)
```

## Arguments

| | |
|---|---|
| x | A numerical or a factor variable that would have a time-varying effect on the event |

## Value

| | |
|---|---|
| x | A variable identified with a time-varying effect |

## References

Y. Mazroui, A. Mauguen, S. Mathoulin-Pelissier, G. MacGrogan, V. Brouste, V. Rondeau (2013). Time-varying coefficients in a multivariate frailty model: Application to breast cancer recurrences of several types and death. To appear.

## Examples

```
data(readmission)

###--- Shared Frailty model with time-varying effect ---###

sha.time <- frailtyPenal(Surv(time,event)~cluster(id)+dukes+charlson+
timedep(sex)+chemo,data=readmission,n.knots=8,kappa=1,
betaknots=3,betaorder=3)

#-- print results of the fit and the associated curves for the
#-- time-dependent effects
print(sha.time)

###--- Joint Frailty model with time-varying effect ---###

joi.time <- frailtyPenal(Surv(time,event)~cluster(id)+timedep(sex)+
chemo+terminal(death),formula.terminalEvent=~timedep(sex)+chemo,
data=readmission,n.knots=8,kappa=c(1,1),betaknots=3,betaorder=3)

print(joi.time)
```

trivPenal *Fit a Trivariate Joint Model for Longitudinal Data, Recurrent Events and a Terminal Event*

## Description

Fit a trivariate joint model for longitudinal data, recurrent events and a terminal event using a semi-parametric penalized likelihood estimation or a parametric estimation on the hazard functions.

The longitudinal outcomes $y_i(t_{ik})$ ($k = 1, \ldots, n_i$, $i = 1, \ldots, N$) for $N$ subjects are described by a linear mixed model and the risks of the recurrent and terminal events are represented by proportional hazard risk models. The joint model is constructed assuming that the processes are linked via a latent structure (Krol et al. 2015):

$$\left\{ \begin{array}{ll} y_i(t_{ik}) = \boldsymbol{X}_{Li}(t_{ik})^\top \boldsymbol{\beta}_L + \boldsymbol{Z}_i(t_{ik})^\top \boldsymbol{b}_i + \epsilon_i(t_{ik}) & \text{(Longitudinal)} \\ r_{ij}(t|\boldsymbol{b}_i) = r_0(t)\exp(v_i + \boldsymbol{X}_{Rij}(t)\boldsymbol{\beta}_R + g(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))^\top \boldsymbol{\eta}_R) & \text{(Recurrent)} \\ \lambda_i(t|\boldsymbol{b}_i) = \lambda_0(t)\exp(\alpha v_i + \boldsymbol{X}_{Ti}(t)\boldsymbol{\beta}_T + h(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))^\top \boldsymbol{\eta}_T) & \text{(Terminal)} \end{array} \right.$$

where $\boldsymbol{X}_{Li}(t)$, $\boldsymbol{X}_{Rij}(t)$ and $\boldsymbol{X}_{Ti}$ are vectors of fixed effects covariates and $\boldsymbol{\beta}_L$, $\boldsymbol{\beta}_R$ and $\boldsymbol{\beta}_T$ are the associated coefficients. Measurements errors $\epsilon_i(t_{ik})$ are iid normally distributed with mean 0 and variance $\sigma_\epsilon^2$. The random effects $\boldsymbol{b}_i = (b_{0i}, \ldots, b_{qi})^\top \sim \mathcal{N}(0, \boldsymbol{B}_1)$ are associated to covariates $\boldsymbol{Z}_i(t)$ and independent from the measurement error. The relationship between the biomarker and recurrent events is explained via $g(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))$ with coefficients $\boldsymbol{\eta}_R$ and between the biomarker and terminal event is explained via $h(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))$ with coefficients $\boldsymbol{\eta}_T$. Two forms of the functions $g(\cdot)$ and $h(\cdot)$ are available: the random effects $\boldsymbol{b}_i$ and the current biomarker level $m_i(t) = \boldsymbol{X}_{Li}(t_{ik})^\top \boldsymbol{\beta}_L + \boldsymbol{Z}_i(t_{ik})^\top \boldsymbol{b}_i$. The frailty term $v_i$ is gaussian with mean 0 and variance $\sigma_v$. Together with $\boldsymbol{b}_i$ constitutes the random effects of the model:

$$\boldsymbol{u}_i = \left( \begin{array}{c} \mathbf{b}_i \\ v_i \end{array} \right) \sim \mathcal{N}\left( \mathbf{0}, \left( \begin{array}{cc} \mathbf{B}_1 & \mathbf{0} \\ \mathbf{0} & \sigma_v^2 \end{array} \right) \right),$$

We consider that the longitudinal outcome can be a subject to a quantification limit, i.e. some observations, below a level of detection $s$ cannot be quantified (left-censoring).

## Usage

```
trivPenal(formula, formula.terminalEvent, formula.LongitudinalData, data,
data.Longi, random, id, intercept = TRUE, link = "Random-effects",
left.censoring = FALSE, recurrentAG = FALSE, n.knots, kappa, maxit = 300,
hazard = "Splines", init.B, init.Random, init.Eta, init.Alpha, method.GH =
"Standard", n.nodes, LIMparam = 1e-3, LIMlogl = 1e-3, LIMderiv = 1e-3,
print.times = TRUE)
```

## Arguments

formula
: a formula object, with the response on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. Interactions are possible using * or :.

formula.terminalEvent
: A formula object, only requires terms on the right to indicate which variables are modelling the terminal event. Interactions are possible using * or :.

formula.LongitudinalData
: A formula object, only requires terms on the right to indicate which variables are modelling the longitudinal outcome. It must follow the standard form used for linear mixed-effects models. Interactions are possible using * or :.

data
: A 'data.frame' with the variables used in `formula`.

data.Longi
: A 'data.frame' with the variables used in `formula.LongitudinalData`.

random
: Names of variables for the random effects of the longitudinal outcome. Maximum 3 random effects are possible at the moment. The random intercept is chosen using `"1"`.

id
: Name of the variable representing the individuals.

intercept
: Logical value. Is the fixed intercept of the biomarker included in the mixed-effects model? The default is `TRUE`.

link
: Type of link functions for the dependence between the biomarker and death and between the biomarker and the recurrent events: `"Random-effects"` for the association directly via the random effects of the biomarker, `"Current-level"` for the association via the true current level of the biomarker. The option `"Current-level"` can be chosen only if the biomarker random effects are associated with the intercept and time (following this order). The default is `"Random-effects"`.

left.censoring
: Is the biomarker left-censored below a threshold $s$? If there is no left-censoring, the argument must be equal to `FALSE`, otherwise the value of the threshold must be given.

recurrentAG
: Logical value. Is Andersen-Gill model fitted? If so indicates that recurrent event times with the counting process approach of Andersen and Gill is used. This formulation can be used for dealing with time-dependent covariates. The default is FALSE.

n.knots
: Integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See Note in `frailtyPenal` function)

kappa
: Positive smoothing parameters in the penalized likelihood estimation. The coefficient kappa of the integral of the squared second derivative of hazard function in the fit (penalized log likelihood). To obtain an initial value for `kappa`, a solution is to fit the corresponding Cox model using cross validation (See `cross.validation` in function `frailtyPenal`). We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them.

| | |
|---|---|
| maxit | Maximum number of iterations for the Marquardt algorithm. Default is 300 |
| hazard | Type of hazard functions: "Splines" for semiparametric hazard functions using equidistant intervals or "Splines-per" using percentile with the penalized likelihood estimation, "Weibull" for the parametric Weibull functions. The default is "Splines". |
| init.B | Vector of initial values for regression coefficients. This vector should be of the same size as the whole vector of covariates with the first elements for the covariates related to the recurrent events, then to the terminal event and then to the biomarker (interactions in the end of each component). Default is 0.5 for each. |
| init.Random | Initial value for variance of the elements of the matrix of the distribution of the random effects. |
| init.Eta | Initial values for regression coefficients for the link functions, first for the recurrent events ($\boldsymbol{\eta}_R$) and for the terminal event ($\boldsymbol{\eta}_T$). |
| init.Alpha | Initial value for parameter alpha |
| method.GH | Method for the Gauss-Hermite quadrature: "Standard" for the standard non-adaptive Gaussian quadrature, "Pseudo-adaptive" for the pseudo-adaptive Gaussian quadrature and "HRMSYM" for the algorithm for the multivariate non-adaptive Gaussian quadrature (see Details). The default is "Standard". |
| n.nodes | Number of nodes for the Gauss-Hermite quadrature. They can be chosen amon 5, 7, 9, 12, 15, 20 and 32. The default is 9. |
| LIMparam | Convergence threshold of the Marquardt algorithm for the parameters (see Details), $10^{-3}$ by default. |
| LIMlogl | Convergence threshold of the Marquardt algorithm for the log-likelihood (see Details), $10^{-3}$ by default. |
| LIMderiv | Convergence threshold of the Marquardt algorithm for the gradient (see Details), $10^{-3}$ by default. |
| print.times | a logical parameter to print iteration process. Default is TRUE. |

## Details

Typical usage for the joint model

```
trivPenal(Surv(time,event)~cluster(id) + var1 + var2 +
terminal(death), formula.terminalEvent =~ var1 + var3, biomarker ~
var1+var2, data, data.Longi, ...)
```

The method of the Gauss-Hermite quadrature for approximations of the multidimensional integrals, i.e. length of random is 2, can be chosen among the standard, non-adaptive, pseudo-adaptive in which the quadrature points are transformed using the information from the fitted mixed-effects model for the biomarker (Rizopoulos 2012) or multivariate non-adaptive procedure proposed by Genz et al. 1996 and implemented in FORTRAN subroutine HRMSYM. The choice of the method is important for estimations. The standard non-adaptive Gauss-Hermite quadrature ("Standard") with a specific number of points gives accurate results but can be time consuming. The non-adaptive procedure ("HRMSYM") offers advantageous computational time but in case of datasets in which some individuals have few repeated observations (biomarker measures or recurrent events), this method may be moderately unstable. The pseudo-adaptive quadrature uses transformed quadrature points to center and scale the integrand by utilizing estimates of the random effects from an appropriate linear mixed-effects model (this transformation does not include the frailty in the trivariate model, for which the standard method is used). This method enables using less quadrature points while preserving the estimation accuracy and thus lead to a better computational time.

NOTE. Data frames `data` and `data.Longi` must be consistent. Names and types of corresponding covariates must be the same, as well as the number and identification of individuals.

## Value

The following components are included in a 'trivPenal' object for each model:

- `b` The sequence of the corresponding estimation of the coefficients for the hazard functions (parametric or semiparametric), the random effects variances and the regression coefficients.
- `call` The code used for the model.
- `formula` The formula part of the code used for the recurrent event part of the model.
- `formula.terminalEvent` The formula part of the code used for the terminal event part of the model.
- `formula.LongitudinalData` The formula part of the code used for the longitudinal part of the model.
- `coef` The regression coefficients (first for the recurrent events, then for the terminal event and then for the biomarker.
- `groups` The number of groups used in the fit.
- `kappa` The values of the smoothing parameters in the penalized likelihood estimation corresponding to the baseline hazard functions for the recurrent and terminal events.
- `logLikPenal` The complete marginal penalized log-likelihood in the semiparametric case.
- `logLik` The marginal log-likelihood in the parametric case.
- `n.measurements` The number of biomarker observations used in the fit.
- `max_rep` The maximal number of repeated measurements per individual.
- `n` The number of observations in 'data' (recurrent and terminal events) used in the fit.
- `n.events` The number of recurrent events observed in the fit.
- `n.deaths` The number of terminal events observed in the fit.
- `n.iter` The number of iterations needed to converge.
- `n.knots` The number of knots for estimating the baseline hazard function in the penalized likelihood estimation.
- `n.strat` The number of stratum.
- `varH` The variance matrix of all parameters (before positivity constraint transformation for the variance of the measurement error, for which the delta method is used).

varHIH
: The robust estimation of the variance matrix of all parameters.

xR
: The vector of times where both survival and hazard function of the recurrent events are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times.

lamR
: The array (dim=3) of baseline hazard estimates and confidence bands (recurrent events).

survR
: The array (dim=3) of baseline survival estimates and confidence bands (recurrent events).

xD
: The vector of times where both survival and hazard function of the terminal event are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times.

lamD
: The array (dim=3) of baseline hazard estimates and confidence bands.

survD
: The array (dim=3) of baseline survival estimates and confidence bands.

medianR
: The value of the median survival and its confidence bands for the recurrent event.

medianD
: The value of the median survival and its confidence bands for the terminal event.

typeof
: The type of the baseline hazard function (0:"Splines", "2:Weibull").

npar
: The number of parameters.

nvar
: The vector of number of explanatory variables for the recurrent events, terminal event and biomarker.

nvarRec
: The number of explanatory variables for the recurrent events.

nvarEnd
: The number of explanatory variables for the terminal event.

nvarY
: The number of explanatory variables for the biomarker.

noVarRec
: The indicator of absence of the explanatory variables for the recurrent events.

noVarEnd
: The indicator of absence of the explanatory variables for the terminal event.

noVarY
: The indicator of absence of the explanatory variables for the biomarker.

LCV
: The approximated likelihood cross-validation criterion in the semiparametric case (with H minus the converged Hessian matrix, and l(.) the full log-likelihood).

$$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$

AIC
: The Akaike information Criterion for the parametric case.

$$AIC = \frac{1}{n}(np - l(.))$$

n.knots.temp
: The initial value for the number of knots.

shape.weib
: The shape parameter for the Weibull hazard functions (the first element for the recurrences and the second one for the terminal event).

scale.weib
: The scale parameter for the Weibull hazard functions (the first element for the recurrences and the second one for the terminal event).

martingale.res
: The martingale residuals related to the recurrences for each individual.

martingaledeath.res
: The martingale residuals related to the terminal event for each individual.

`conditional.res`
: The conditional residuals for the biomarker (subject-specific): $\boldsymbol{R}_i^{(m)} = \boldsymbol{y}_i - \boldsymbol{X}_{Li}^\top \widehat{\boldsymbol{\beta}}_L - \boldsymbol{Z}_i^\top \widehat{\boldsymbol{b}}_i$.

`marginal.res`
: The marginal residuals for the biomarker (population averaged): $\boldsymbol{R}_i^{(c)} = \boldsymbol{y}_i - \boldsymbol{X}_{Li}^\top \widehat{\boldsymbol{\beta}}_L$.

`marginal_chol.res`
: The Cholesky marginal residuals for the biomarker: $\boldsymbol{R}_i^{(m)} = \widehat{\boldsymbol{U}_i^{(m)}} \boldsymbol{R}_i^{(m)}$, where $\widehat{\boldsymbol{U}_i^{(m)}}$ is an upper-triangular matrix obtained by the Cholesky decomposition of the variance matrix $\boldsymbol{V}_{\boldsymbol{R}_i^{(m)}} = \widehat{\boldsymbol{V}_i} - \boldsymbol{X}_{Li}(\sum_{i=1}^N \boldsymbol{X}_{Li} \widehat{\boldsymbol{V}_i}^{-1} \boldsymbol{X}_{Li})^{-1} \boldsymbol{X}_{Li}^\top$.

`conditional_st.res`
: The standardized conditional residuals for the biomarker.

`marginal_st.res`
: The standardized marginal residuals for the biomarker.

`random.effects.pred`
: The empirical Bayes predictions of the random effects (ie. using conditional posterior distributions).

`frailty.pred`
: The empirical Bayes predictions of the frailty term (ie. using conditional posterior distributions).

`pred.y.marg`
: The marginal predictions of the longitudinal outcome.

`pred.y.cond`
: The conditional (given the random effects) predictions of the longitudinal outcome.

`linear.pred`
: The linear predictor for the recurrent events part.

`lineardeath.pred`
: The linear predictor for the terminal event part.

`global_chisqR`
: The vector with values of each multivariate Wald test for the recurrent part.

`dof_chisqR`
: The vector with degrees of freedom for each multivariate Wald test for the recurrent part.

`global_chisq.testR`
: The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the recurrent part).

`p.global_chisqR`
: The vector with the p_values for each global multivariate Wald test for the recurrent part.

`global_chisqT`
: The vector with values of each multivariate Wald test for the terminal part.

`dof_chisqT`
: The vector with degrees of freedom for each multivariate Wald test for the terminal part.

`global_chisq.testT`
: The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the terminal part).

`p.global_chisqT`
: The vector with the p_values for each global multivariate Wald test for the terminal part.

global_chisqY The vector with values of each multivariate Wald test for the longitudinal part.

dof_chisqY The vector with degrees of freedom for each multivariate Wald test for the longitudinal part.

global_chisq.testY
The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the longitudinal part).

p.global_chisqY
The vector with the p_values for each global multivariate Wald test for the longitudinal part.

names.factorR The names of the "as.factor" variables for the recurrent part.

names.factorT The names of the "as.factor" variables for the terminal part.

names.factorY The names of the "as.factor" variables for the longitudinal part.

AG The logical value. Is Andersen-Gill model fitted?

intercept The logical value. Is the fixed intercept included in the linear mixed-effects model?

B1 The variance matrix of the random effects for the longitudinal outcome.

sigma2 The variance of the frailty term ($\sigma_v$).

alpha The coefficient $\alpha$ associated with the frailty parameter in the terminal hazard function.

ResidualSE The variance of the measurement error.

etaR The regression coefficients for the link function $g(\cdot)$.

etaT The regression coefficients for the link function $h(\cdot)$.

ne_re The number of random effects b used in the fit.

names.re The names of variables for the random effects $\boldsymbol{b}_i$.

link The name of the type of the link functions.

leftCensoring The logical value. Is the longitudinal outcome left-censored?

leftCensoring.threshold
For the left-censored biomarker, the value of the left-censoring threshold used for the fit.

prop.censored The fraction of observations subjected to the left-censoring.

methodGH The Gaussian quadrature method used in the fit.

n.nodes The number of nodes used for the Gaussian quadrature in the fit.

alpha_p.value p-value of the Wald test for the estimated coefficient $\alpha$.

sigma2_p.value p-value of the Wald test for the estimated variance of the frailty term ($\sigma_v$).

etaR_p.value p-values of the Wald test for the estimated regression coefficients for the link function $g(\cdot)$.

etaT_p.value p-values of the Wald test for the estimated regression coefficients for the link function $h(\cdot)$.

beta_p.value p-values of the Wald test for the estimated regression coefficients.

## Note

It is recommended to initialize the parameter values using the results from the reduced models (for example, `longiPenal` for the longitudinal and terminal part and `frailtyPenal` for the recurrent part. See example.

## References

A. Krol, A. Mauguen, Y. Mazroui, A. Laurent, S. Michiels and V. Rondeau (2017). Tutorial in Joint Modeling and Prediction: A Statistical Software for Correlated Longitudinal Outcomes, Recurrent Events and a Terminal Event. *Journal of Statistical Software* **81**(3), 1-52.

A. Krol, L. Ferrer, JP. Pignon, C. Proust-Lima, M. Ducreux, O. Bouche, S. Michiels, V. Rondeau (2016). Joint Model for Left-Censored Longitudinal Data, Recurrent Events and Terminal Event: Predictive Abilities of Tumor Burden for Cancer Evolution with Application to the FFCD 2000-05 Trial. *Biometrics* **72**(3) 907-16.

D. Rizopoulos (2012). Fast fitting of joint models for longitudinal and event time data using a pseudo-adaptive Gaussian quadrature rule. *Computational Statistics and Data Analysis* **56**, 491-501.

A. Genz and B. Keister (1996). Fully symmetric interpolatory rules for multiple integrals over infinite regions with Gaussian weight. *Journal of Computational and Applied Mathematics* **71**, 299-309.

## See Also

plot.trivPenal,print.trivPenal,summary.trivPenal

## Examples

```
## Not run:

###--- Trivariate joint model for longitudinal data, ---###
###--- recurrent events and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# Parameter initialisation for covariates - longitudinal and terminal part

# Survival data preparation - only terminal events
colorectalSurv <- subset(colorectal, new.lesions == 0)

initial.longi <- longiPenal(Surv(time1, state) ~ age + treatment + who.PS
+ prev.resection, tumor.size ~  year * treatment + age + who.PS ,
colorectalSurv,data.Longi = colorectalLongi, random = c("1", "year"),
id = "id", link = "Random-effects", left.censoring = -3.33,
n.knots = 6, kappa = 2, method.GH="Pseudo-adaptive",
 maxit=40, n.nodes=7)


# Parameter initialisation for covariates - recurrent part
```

```
initial.frailty <- frailtyPenal(Surv(time0, time1, new.lesions) ~ cluster(id)
+ age + treatment + who.PS, data = colorectal,
recurrentAG = TRUE, RandDist = "LogN", n.knots = 6, kappa =2)


# Baseline hazard function approximated with splines
# Random effects as the link function, Calendar timescale
# (computation takes around 40 minutes)

model.spli.RE.cal <-trivPenal(Surv(time0, time1, new.lesions) ~ cluster(id)
+ age + treatment + who.PS +  terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = TRUE,
n.knots = 6, kappa=c(0.01, 2), method.GH="Standard", n.nodes = 7,
init.B = c(-0.07, -0.13, -0.16, -0.17, 0.42, #recurrent events covariates
-0.16, -0.14, -0.14, 0.08, 0.86, -0.24, #terminal event covariates
2.93, -0.28, -0.13, 0.17, -0.41, 0.23, 0.97, -0.61)) #biomarker covariates



# Weibull baseline hazard function
# Random effects as the link function, Gap timescale
# (computation takes around 30 minutes)
model.weib.RE.gap <-trivPenal(Surv(gap.time, new.lesions) ~ cluster(id)
+ age + treatment + who.PS + prev.resection + terminal(state),
formula.terminalEvent =~ age + treatment + who.PS + prev.resection,
tumor.size ~ year * treatment + age + who.PS, data = colorectal,
data.Longi = colorectalLongi, random = c("1", "year"), id = "id",
link = "Random-effects", left.censoring = -3.33, recurrentAG = FALSE,
hazard = "Weibull", method.GH="Pseudo-adaptive",n.nodes=7)



## End(Not run)
```

---

`trivPenalNL` *Fit a Non-Linear Trivariate Joint Model for Recurrent Events and a Terminal Event with a Biomarker Described with an ODE Population Model*

---

## Description

Fit a non-linear trivariate joint model for a longitudinal biomarker, recurrent events and a terminal event using a semiparametric penalized likelihood estimation or a parametric estimation on the hazard functions.

The values $y_i(t)$ ($i = 1, \ldots, N$) for $N$ subjects represent the individual evolution of the biomarker e.g. tumor size expressed as the sum of the longest diameters (SLD) of target lesions. The dynamics of the biomarker are described by an ordinary differential equation (ODE) that includes the effect of the natural net growth and the treatment effect:

$$\begin{cases} \frac{dy_i(t)}{dt} & = \exp(K_{G,0} + b_{G,i} + \boldsymbol{X}_{G,i}(t)^\top \boldsymbol{\beta}_G) y_i(t) \\ & -d_i \exp(K_{D,0} + b_{D,i} - t \times \exp(\lambda + b_{\lambda,i}) + \boldsymbol{X}_{D,i}(t)^\top \boldsymbol{\beta}_D) y_i(t) \\ y_i(0) & = \exp(y_0 + b_{y_0,i}) \end{cases},$$

The model includes the following parameters (using the interpretation of tumor dynamics): $\exp(K_{G,0})$ the constant tumor growth rate, $\exp(K_{D,0})$ the drug-induced tumor decline rate, $\lambda$ resistance effect to drug (exponential tumor decay change with time), $\exp(y_0)$ the initial level of the biomarker and $d_i$ is the treatment concentration (e.g dose). The random effects $\boldsymbol{b}_i^\top = (b_{y_0,i}, b_{G,i}, b_{D,i}, b_{\lambda,i})^\top$ are gaussian variables with a diagonal covariance matrix $\boldsymbol{B}_1$. In the trivariate model we use the analytical solution of the equation with the population-based approach of the non-linear mixed effects model. We can also assume a transformation for the observations of the biomarker (one parameter Box-Cox transformation) and we include a gaussian measurement error, for individual $i$ and observation $k$ ($k = 1, \ldots, n_i$), $\epsilon_{ik} \sim N(0, \sigma_\epsilon^2)$.

The risks of the recurrent ($r_{ij}(\cdot)$ the risk of the $j$-th event of the individual $i$) and terminal events ($\lambda_i$ the risk of the event of the individual $i$) are represented by proportional hazard risk models. The joint model is constructed assuming that the processes are linked via a latent structure and includes the non-linear mixed effects model for the longitudinal data:

$$\begin{cases} y(t_{ik}) & = \exp[y_0 + b_{y_0,i} + t_{ik} \times \exp(K_{G,0} + b_{G,i} + \boldsymbol{X}_{G,i}(t)^\top \boldsymbol{\beta}_G) \\ & +d_i \times \exp(K_{D,0} + b_{D,i} - \lambda - b_{\lambda,i} + \boldsymbol{X}_{D,i}(t)^\top \boldsymbol{\beta}_D) \\ & \times(\exp(-\exp(\lambda + b_{\lambda,i}) t_{ik}) - 1)] + \epsilon_{ik} \\ r_{ij}(t|\boldsymbol{b}_i) & = r_0(t) \exp(v_i + \boldsymbol{X}_{Rij}(t)^\top \boldsymbol{\beta}_R + g(y_i(t))^\top \boldsymbol{\eta}_R) \\ \lambda_i(t|\boldsymbol{b}_i) & = \lambda_0(t) \exp(\alpha v_i + \boldsymbol{X}_{Ti}(t)^\top \boldsymbol{\beta}_T + h(y_i(t))^\top \boldsymbol{\eta}_T) \end{cases}$$

where $\boldsymbol{X}_{G,i}(t)$, $\boldsymbol{X}_{D,i}(t)$, $\boldsymbol{X}_{R,ij}(t)$ and $\boldsymbol{X}_{T,i}(t)$ are vectors of possible time-varying fixed effects covariates and $\boldsymbol{\beta}_G$, $\boldsymbol{\beta}_D$, $\boldsymbol{\beta}_R$ and $\boldsymbol{\beta}_T$ are the associated coefficients. The random effects $\boldsymbol{b}_i$ are independent from the measurement error. The relationship between the biomarker and recurrent events is explained via $g(y_i(t))$ with coefficients $\boldsymbol{\eta}_R$ and between the biomarker and terminal event is explained via $h(y_i(t))$ with coefficients $\boldsymbol{\eta}_T$. Currently, only one form of the functions $g(\cdot)$ and $h(\cdot)$ is available: the random effects $\boldsymbol{b}_i$. The frailty term $v_i$ is gaussian with mean 0 and variance $\sigma_v$. Together with $\boldsymbol{b}_i$ constitutes the random effects of the model:

$$\boldsymbol{u}_i = \left( \begin{array}{c} \mathbf{b}_i \\ v_i \end{array} \right) \sim \mathcal{N} \left( \mathbf{0}, \left( \begin{array}{cc} \mathbf{B}_1 & \mathbf{0} \\ \mathbf{0} & \sigma_v^2 \end{array} \right) \right),$$

Any combination of the random effects $\boldsymbol{b}_i$, e.g. $\boldsymbol{b}_i = b_{y_0,i}$ or $\boldsymbol{b}_i = \{b_{G,i}, b_{D,i}, b_{\lambda,i}\}$ can be chosen for the model.

We consider that the longitudinal outcome can be a subject to a quantification limit, i.e. some observations, below a level of detection $s$ cannot be quantified (left-censoring).

## Usage

```
trivPenalNL(formula, formula.terminalEvent, biomarker, formula.KG,
```

```
formula.KD, dose, time.biomarker, data, data.Longi, random, id, link =
"Random-effects", BoxCox = FALSE, left.censoring = FALSE, recurrentAG =
FALSE, n.knots, kappa, maxit = 300, hazard = "Splines", init.B, init.Random,
init.Eta, init.Alpha, init.Biomarker, method.GH = "Standard", init.GH =
FALSE, n.nodes, LIMparam = 1e-3, LIMlogl = 1e-3, LIMderiv = 1e-3,
print.times = TRUE)
```

### Arguments

- `formula` a formula object, with the response on the left of a $\sim$ operator, and the terms on the right. The response must be a survival object as returned by the 'Surv' function like in survival package. Interactions are possible using * or :.
- `formula.terminalEvent` A formula object, only requires terms on the right to indicate which variables are modelling the terminal event. Interactions are possible using * or :.
- `biomarker` Name of the variable representing the longitudinal biomarker.
- `formula.KG` A formula object, only requires terms on the right to indicate which covariates related to the biomarker growth are included in the longitudinal sub-model. It must follow the standard form used for linear mixed-effects models. Interactions are possible using * or :.
- `formula.KD` A formula object, only requires terms on the right to indicate which covariates related to the biomarker drug-induced decline are included in the longitudinal sub-model. It must follow the standard form used for linear mixed-effects models. Interactions are possible using * or :.
- `dose` Name of the variable representing the drug concentration indicator.
- `time.biomarker` Name of the variable of times of biomarker measurements.
- `data` A 'data.frame' with the variables used in `formula`.
- `data.Longi` A 'data.frame' with the variables used in `formula.KG`, `formula.KD`, `biomarker`, `dose`, `time.biomarker` and `id`.
- `random` Names of parameters for which the random effects are included in the mixed model. The names must be chosen among `"y0"`, `"KG"`, `"KD"` and `"lambda"`. Any combination of the mentioned names is allowed.
- `id` Name of the variable representing the individuals.
- `link` Type of link functions for the dependence between the biomarker and death and between the biomarker and the recurrent events: only `"Random-effects"` for the association directly via the random effects of the biomarker is allowed for the moment (option for a future extension).
- `BoxCox` Should the Box-Cox transformation be used for the longitudinal biomarker? If there is no transformation, the argument must be equal to `FALSE`, otherwise the of the transformation parameter must be given, then the transformed values are $y^* = (y^\xi - 1)/\xi$, where $\xi$ is the Box-Cox parameter.
- `left.censoring` Is the biomarker left-censored below a threshold $s$? If there is no left-censoring, the argument must be equal to `FALSE`, otherwise the value of the threshold must be given.

recurrentAG
: Logical value. Is Andersen-Gill model fitted? If so indicates that recurrent event times with the counting process approach of Andersen and Gill is used. This formulation can be used for dealing with time-dependent covariates. The default is FALSE.

n.knots
: Integer giving the number of knots to use. Value required in the penalized likelihood estimation. It corresponds to the (n.knots+2) splines functions for the approximation of the hazard or the survival functions. We estimate I or M-splines of order 4. When the user set a number of knots equals to k (n.knots=k) then the number of interior knots is (k-2) and the number of splines is (k-2)+order. Number of knots must be between 4 and 20. (See Note in `frailtyPenal` function)

kappa
: Positive smoothing parameters in the penalized likelihood estimation. The coefficient kappa of the integral of the squared second derivative of hazard function in the fit (penalized log likelihood). To obtain an initial value for `kappa`, a solution is to fit the corresponding Cox model using cross validation (See `cross.validation` in function `frailtyPenal`). We advise the user to identify several possible tuning parameters, note their defaults and look at the sensitivity of the results to varying them.

maxit
: Maximum number of iterations for the Marquardt algorithm. Default is 300

hazard
: Type of hazard functions: `"Splines"` for semiparametric hazard functions using equidistant intervals or `"Splines-per"` using percentile with the penalized likelihood estimation, `"Weibull"` for the parametric Weibull functions. The default is `"Splines"`.

init.B
: Vector of initial values for regression coefficients. This vector should be of the same size as the whole vector of covariates with the first elements for the covariates related to the recurrent events, then to the terminal event and then to the biomarker (interactions in the end of each component). Default is 0.5 for each.

init.Random
: Initial value for variance of the elements of the matrix of the distribution of the random effects.

init.Eta
: Initial values for regression coefficients for the link functions, first for the recurrent events ($\boldsymbol{\eta}_R$) and for the terminal event ($\boldsymbol{\eta}_T$).

init.Alpha
: Initial value for parameter alpha

init.Biomarker
: Initial values for biomarker parameters: $y_0$, $K_{G,0}$, $K_{D,0}$ and $\lambda$ (using this order).

method.GH
: Method for the Gauss-Hermite quadrature: `"Standard"` for the standard non-adaptive Gaussian quadrature and `"Pseudo-adaptive"` for the pseudo-adaptive Gaussian quadrature (see Details). The default is `"Standard"`. When the option `"Pseudo-adaptive"` is chosen, then a univariate model (non-linear mixed model for the biomarker) is fitted in order to obtain the estimations of the random effects $\boldsymbol{b}_i$.

init.GH
: Only when the opiton `"Pseudo-adaptive"` of the argument `method.GH` is chosen. If `TRUE`, the estimations of the biomarker parameters ($y_0$, $K_{G,0}$, $K_{D,0}$ and $\lambda$), $\sigma_\epsilon$, $\boldsymbol{\beta}_G$ and $\boldsymbol{\beta}_D$ from the univariate mixed model are used as the initial values of the parameters related to the biomarker.

n.nodes
: Number of nodes for the Gauss-Hermite quadrature (from 5 to 32). The default is 9.

LIMparam Convergence threshold of the Marquardt algorithm for the parameters (see Details), $10^{-3}$ by default.

LIMlogl Convergence threshold of the Marquardt algorithm for the log-likelihood (see Details), $10^{-3}$ by default.

LIMderiv Convergence threshold of the Marquardt algorithm for the gradient (see Details), $10^{-3}$ by default.

print.times a logical parameter to print iteration process. Default is TRUE.

## Details

Typical usage for the joint model

```
trivPenalNL(Surv(time,event)~cluster(id) + var1 + var2 +
terminal(death), formula.terminalEvent =~ var1 + var3, biomarker =
"biomarker.name", dose = "dose.name", time.biomarker = "time", formula.KG ~
var1, formula.KD ~ var2, data, data.Longi, ...)
```

The method of the Gauss-Hermite quadrature for approximations of the multidimensional integrals, i.e. length of random more than 2, can be chosen among the standard (non-adaptive) and pseudo-adaptive in which the quadrature points are transformed using the information from the fitted mixed-effects model for the biomarker (Rizopoulos 2012) or multivariate non-adaptive procedure proposed by Genz et al. 1996 and implemented in FORTRAN subroutine HRMSYM. The choice of the method is important for estimations. The standard non-adaptive Gauss-Hermite quadrature ("Standard") with a specific number of points gives accurate results but can be time consuming. The pseudo-adaptive quadrature uses transformed quadrature points to center and scale the integrand by utilizing estimates of the random effects from an appropriate non-linear mixed-effects model (this transformation does not include the frailty in the trivariate model, for which the standard method, with 20 quadrature points, is used). This method enables using less quadrature points while preserving the estimation accuracy and thus lead to a better computational time.

NOTE. Data frames data and data.Longi must be consistent. Names and types of corresponding covariates must be the same, as well as the number and identification of individuals.

## Value

The following components are included in a 'trivPenalNL' object for each model:

b The sequence of the corresponding estimation of the coefficients for the hazard functions (parametric or semiparametric), the random effects variances and the regression coefficients.

call The code used for the model.

formula The formula part of the code used for the recurrent event part of the model.

formula.terminalEvent
The formula part of the code used for the terminal event part of the model.

formula.KG The formula part of the code used for the longitudinal part of the model, for the biomarker growth dynamics.

formula.KD The formula part of the code used for the longitudinal part of the model, for the biomarker decline dynamics.

| | |
|---|---|
| coef | The regression coefficients (first for the recurrent events, then for the terminal event, then for the biomarker growth and then for the biomarker decline. |
| groups | The number of groups used in the fit. |
| kappa | The values of the smoothing parameters in the penalized likelihood estimation corresponding to the baseline hazard functions for the recurrent and terminal events. |
| logLikPenal | The complete marginal penalized log-likelihood in the semiparametric case. |
| logLik | The marginal log-likelihood in the parametric case. |
| n.measurements | The number of biomarker observations used in the fit. |
| max_rep | The maximal number of repeated measurements per individual. |
| n | The number of observations in 'data' (recurrent and terminal events) used in the fit. |
| n.events | The number of recurrent events observed in the fit. |
| n.deaths | The number of terminal events observed in the fit. |
| n.iter | The number of iterations needed to converge. |
| n.knots | The number of knots for estimating the baseline hazard function in the penalized likelihood estimation. |
| n.strat | The number of stratum. |
| varH | The variance matrix of all parameters (before positivity constraint transformation for the variance of the measurement error, for which the delta method is used). |
| varHIH | The robust estimation of the variance matrix of all parameters. |
| xR | The vector of times where both survival and hazard function of the recurrent events are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times. |
| lamR | The array (dim=3) of baseline hazard estimates and confidence bands (recurrent events). |
| survR | The array (dim=3) of baseline survival estimates and confidence bands (recurrent events). |
| xD | The vector of times where both survival and hazard function of the terminal event are estimated. By default seq(0,max(time),length=99), where time is the vector of survival times. |
| lamD | The array (dim=3) of baseline hazard estimates and confidence bands. |
| survD | The array (dim=3) of baseline survival estimates and confidence bands. |
| medianR | The value of the median survival and its confidence bands for the recurrent event. |
| medianD | The value of the median survival and its confidence bands for the terminal event. |
| typeof | The type of the baseline hazard function (0:"Splines", "2:Weibull"). |
| npar | The number of parameters. |
| nvar | The vector of number of explanatory variables for the recurrent events, terminal event, biomarker growth and biomarker decline. |

nvarRec
: The number of explanatory variables for the recurrent events.

nvarEnd
: The number of explanatory variables for the terminal event.

nvarKG
: The number of explanatory variables for the biomarker growth.

nvarKD
: The number of explanatory variables for the biomarker decline.

noVarRec
: The indicator of absence of the explanatory variables for the recurrent events.

noVarEnd
: The indicator of absence of the explanatory variables for the terminal event.

noVarKG
: The indicator of absence of the explanatory variables for the biomarker growth.

noVarKD
: The indicator of absence of the explanatory variables for the biomarker decline.

LCV
: The approximated likelihood cross-validation criterion in the semiparametric case (with H minus the converged Hessian matrix, and l(.) the full log-likelihood).

$$LCV = \frac{1}{n}(trace(H_{pl}^{-1}H) - l(.))$$

AIC
: The Akaike information Criterion for the parametric case.

$$AIC = \frac{1}{n}(np - l(.))$$

n.knots.temp
: The initial value for the number of knots.

shape.weib
: The shape parameter for the Weibull hazard functions (the first element for the recurrences and the second one for the terminal event).

scale.weib
: The scale parameter for the Weibull hazard functions (the first element for the recurrences and the second one for the terminal event).

random.effects.pred
: The empirical Bayes predictions of the random effects (ie. using conditional posterior distributions).

global_chisq.testR
: The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the recurrent part).

global_chisq.testT
: The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the terminal part).

global_chisq.testKG
: The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the biomarker growth).

global_chisq.testKD
: The binary variable equals to 0 when no multivariate Wald is given, 1 otherwise (for the biomarker decline).

AG
: The logical value. Is Andersen-Gill model fitted?

B1
: The variance matrix of the random effects for the longitudinal outcome.

sigma2
: The variance of the frailty term ($\sigma_v$).

alpha
: The coefficient $\alpha$ associated with the frailty parameter in the terminal hazard function.

- `ResidualSE` The variance of the measurement error.
- `etaR` The regression coefficients for the link function $g(\cdot)$.
- `etaT` The regression coefficients for the link function $h(\cdot)$.
- `ne_re` The number of random effects b used in the fit.
- `names.re` The names of variables for the random effects $\boldsymbol{b}_i$.
- `link` The name of the type of the link functions.
- `leftCensoring` The logical value. Is the longitudinal outcome left-censored?
- `leftCensoring.threshold` For the left-censored biomarker, the value of the left-censoring threshold used for the fit.
- `prop.censored` The fraction of observations subjected to the left-censoring.
- `methodGH` The Gaussian quadrature method used in the fit.
- `n.nodes` The number of nodes used for the Gaussian quadrature in the fit.
- `K_G0` Value of the estimate of the biomarker growth parameter.
- `K_D0` Value of the estimate of the biomarker decay parameter.
- `lambda` Value of the estimate of the biomarker resistance to drug.
- `y_0` Value of the estimate of the biomarker intial level.
- `biomarker` Name of the variable associated with the biomarker in the data.
- `time.biomarker` Name of the variable associated with the time of measurements of the biomarker in the data.
- `dose` Name of the variable associated with the drug concentration in the data.
- `BoxCox` The logical value. Is the BoxCox transformation applied for the biomarker?
- `BoxCox_parameter` The value of the BoxCox transformation parameter.
- `alpha_p.value` p-value of the Wald test for the estimated coefficient $\alpha$.
- `sigma2_p.value` p-value of the Wald test for the estimated variance of the frailty term ($\sigma_v$).
- `etaR_p.value` p-values of the Wald test for the estimated regression coefficients for the link function $g(\cdot)$.
- `etaT_p.value` p-values of the Wald test for the estimated regression coefficients for the link function $h(\cdot)$.
- `y_0_p.value` p-value of the Wald test for the estimated biomarker intial level.
- `K_G0_p.value` p-value of the Wald test for the estimated biomarker growth parameter.
- `K_D0_p.value` p-value of the Wald test for the estimated biomarker decay parameter.
- `lambda_p.value` p-value of the Wald test for the estimated biomarker resistance to drug.
- `beta_p.value` p-values of the Wald test for the estimated regression coefficients.

## Note

It is recommended to initialize the parameter values using the results from a corresponding reduced model (`frailtyPenal` for the recurrent and terminal part). See example.

Estimations of models with more than three random effects can be very long.

## References

A. Krol, C. Tournigand, S. Michiels and V. RondeauS (2018). Multivariate joint frailty model for the analysis of nonlinear tumor kinetics and dynamic predictions of death. *Statistics in Medicine*.

A. Krol, L. Ferrer, JP. Pignon, C. Proust-Lima, M. Ducreux, O. Bouche, S. Michiels, V. Rondeau (2016). Joint Model for Left-Censored Longitudinal Data, Recurrent Events and Terminal Event: Predictive Abilities of Tumor Burden for Cancer Evolution with Application to the FFCD 2000-05 Trial. *Biometrics* **72**(3) 907-16.

D. Rizopoulos (2012). Fast fitting of joint models for longitudinal and event time data using a pseudo-adaptive Gaussian quadrature rule. *Computational Statistics and Data Analysis* **56**, 491-501.

L. Claret, P. Girard, P.M. Hoff, E. Van Cutsem, K.P. Zuideveld, K. Jorga, J. Fagerberg, R Bruno (2009). Model-based prediction of phase III overall survival in colorectal cancer on the basis of phase II tumor dynamics. *Journal of Clinical Oncology* **27**(25), 4103-8.

## See Also

plot.trivPenalNL,print.trivPenalNL,summary.trivPenalNL

## Examples

```
## Not run:

###--- Non-linear trivariate joint model for longitudinal data, ---###
###--- recurrent events and a terminal event ---###

data(colorectal)
data(colorectalLongi)

# No information on dose - creation of a dummy variable
colorectalLongi$dose <- 1


# Parameters initialisation - estimation of a simplified model
# with two random effects (a frailty term and a random effect
# related to biomarker growth (KG))
initial.model <- trivPenalNL(Surv(time0, time1, new.lesions) ~ cluster(id)
 + age + treatment + terminal(state), formula.terminalEvent =~ age + treatment,
 biomarker = "tumor.size", formula.KG ~ 1, formula.KD ~ treatment, dose = "dose",
 time.biomarker = "year", data = colorectal, data.Longi =colorectalLongi,
 random = "KG", id = "id", recurrentAG = TRUE, n.knots = 5, kappa = c(0.01, 2),
 method.GH = "Pseudo-adaptive")


# Trivariate joint model with initial values for parameters
# (computation takes around 40 minutes)

model <- trivPenalNL(Surv(time0, time1, new.lesions) ~ cluster(id) + age + treatment
 + terminal(state), formula.terminalEvent =~ age + treatment, biomarker = "tumor.size",
 formula.KG ~ 1, formula.KD ~ treatment, dose = "dose", time.biomarker = "year",
```

```
 data = colorectal, data.Longi =colorectalLongi, random = c("y0", "KG"), id = "id",
 init.B = c(-0.22, -0.16, -0.35, -0.19, 0.04, -0.41, 0.23), init.Alpha = 1.86,
 init.Eta = c(0.5, 0.57, 0.5, 2.34), init.Biomarker = c(1.24, 0.81, 1.07, -1.53),
 recurrentAG = TRUE, n.knots = 5, kappa = c(0.01, 2), method.GH = "Pseudo-adaptive")


## End(Not run)
```

---

`wts` *Identify weights*

---

## Description

This is a special function used in the context of the joint frailty models for data from nested case-control studies. It specifies weights defined by using 'wts' function, and is used of 'frailtyPenal' formula for fitting joint models.

## Usage

```
wts(x)
```

## Arguments

| | |
|---|---|
| `x` | A numeric variable which is supposed to indicate the weights |

## Value

| | |
|---|---|
| `x` | A variable identified as weights |

## See Also

`frailtyPenal`

## Examples

```
data(dataNCC)
modJoint.ncc <- frailtyPenal(Surv(t.start,t.stop,event)~cluster(id)+cov1
+cov2+terminal(death)+wts(ncc.wts), formula.terminalEvent=~cov1+cov2,
data=dataNCC,n.knots=8,kappa=c(1.6e+10, 5.0e+03),recurrentAG=TRUE, RandDist="LogN")


print(modJoint.ncc)
```

# Index