Package 'frailtypack'

February 3, 2017

Version 2.10.5

Date 2017-02-02

Title General Frailty Models: Shared, Joint and Nested Frailty Models with Prediction

Author Virginie Rondeau, Juan R. Gonzalez, Yassin Mazroui, Audrey Mauguen, Agnieszka Krol, Amadou Diakite, Alexandre Laurent and Myriam Lopez

Maintainer Virginie Rondeau <Virginie.Rondeau@isped.u-bordeaux2.fr>

Depends R (>= 2.10), survival, boot, MASS, survC1, nlme

LazyLoad no

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.
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.

License GPL (>= 2.0)

URL http://virginierondeau.com/BiostatisticalConsulting/Liste_of_examples.html

Suggests knitr, rmarkdown

VignetteBuilder knitr

Repository CRAN

Date/Publication 2017-02-03 08:13:28

NeedsCompilation yes

R topics documented:

`frailtypack-package` *General Frailty models: shared, joint and nested frailty models with prediction*

Description

Frailtypack fits several classes of frailty models using a penalized likelihood estimation on the hazard function but also a parametric estimation. 1) A shared frailty model 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 joint modelling for recurrent events with terminal event for clustered data or not. A joint frailty model for two semi-competing risks for clustered data is also proposed. 5) Joint General frailty models in the context of a joint modelling for recurrent events with terminal event data with two independent frailty terms. 6) Joint Nested frailty models in the context of 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. Prediction values are available. Left truncated (not for the joint models), 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 effect covariates in Cox, shared and joint frailty models. The package includes concordance measures for Cox proportional hazards models and for shared frailty models.

Details

Package:	frailtypack
Type:	Package
Version:	2.8.3
Date:	2016-01-12
License:	GPL (>= 2.0)
LazyLoad:	no

Author(s)

Virginie Rondeau, Juan R. Gonzalez, Yassin Mazroui, Audrey Mauguen, Agnieszka Krol, Amadou Diakite and Alexandre Laurent

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.

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
```

```
## 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:

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

where $\lambda_0(t)$ is the baseline hazard function, β_k the fixed effect associated to the covariate X_{ijk} (k=1,..,p), β_1 is the treatment effect and X_{ij1} the treatment variable. ρ 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)
```

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.

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 χ^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.

`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.

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

```
## 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)

#-- Var1 is boolean as a treatment variable


## End(Not run)
```

---

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

```
## Not run:

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

print(modSha)


## End(Not run)
```

`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

```
## Not run:

#-- 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


## End(Not run)
```

`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.

`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 β 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 α_1 and α_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

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.

`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` or `trivPenal`. 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 object.
`pred.times`	Time or vector of times to compute epoce.
`newdata`	Optional. In case of joint models obtained with `frailtyPenal` or `trivPenal`. For models inheriting from `trivPenal` 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` or `trivPenal`. For models inheriting from `longiPenal`, if the `newdata.Longi` is given, `newdata` must be `NULL`, but for models from `trivPenal` 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 = "cvpol")

################################################
#### 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 = "cvpol")


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

data(colorectal)
data(colorectalLongi)

# (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)

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

# (computation takes around 10 minutes)
epoce <- epoce(modTriv,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(modTriv, time, newdata = s, newdata.Longi = s.Longi)

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


dev.off()

## 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`

`frailtyPenal` *Fit a Shared, Joint or Nested Frailty 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 ω_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 `frailtyPenal` :

- The first one (Rondeau et al. 2007) includes a common frailty term to the individuals (ω_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 (ω_i for the recurrent rate and ω_i^{α} 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 :

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

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 $\omega_i \sim \mathbf{\Gamma}(\frac{1}{\theta}, \frac{1}{\theta})$ and are iid.

The joint log-normal frailty model will be :

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

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.

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

It should be noted that in these models it is not recommended to include α 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 :

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

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.

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:

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

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

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 β 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^{ξ} for the recurrent rate and u_{fi}^{α}, 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 λ_{fi} is :

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

where $r_0(t)$(resp. $\lambda_0(t)$) is the recurrent (resp. terminal) event baseline hazard function, β (resp. γ) 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", betaknots = 1,
betaorder = 3, initialize = TRUE, init.B, init.Theta, init.Alpha,
Alpha, 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 θ) and an association amongst the recurrent events (explained by the variance η).
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.

`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.

`init.Ksi` Only for joint nested frailty model : initial value for parameter ξ.

`init.Eta` Only for general joint and joint nested frailty models : initial value for the variance η of the frailty v_i (general joint model) and of the frailty ω_i (joint nested frailty model).

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

for a shared model

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

for a joint model

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

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, ...)
```

for a nested model

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

for a joint nested frailty model

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

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 Δ-method (Knight & Xekalaki, 2000). 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 θ 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 θ and the coefficient α 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 η and ξ 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 θ of the frailty terms and α. The variances of the second frailty term (η) and the second coefficient ξ are initialized to 1. Then, a joint nested 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.
`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	array (dim=3) of hazard estimates and confidence bands.
surv	array (dim=3) of baseline survival estimates and confidence bands.
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.

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)

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 α parameter was fitted
`alpha`	the coefficient α 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	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.
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)

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

alpha	variance of the cluster effect ($\boldsymbol{Var}(v_i)$)
eta	variance of the subcluster effect ($\boldsymbol{Var}(w_{ij})$)
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.

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 α associated with the frailty parameter (u_{fi}) in the terminal event hazard function.

`ksi` the power coefficient ξ associated with the frailty parameter (ω_f) in the recurrent event hazard function.

`indic_alpha` indicator if a joint frailty model with α parameter was fitted or not.

`indic_ksi` indicator if a joint frailty model with ξ parameter was fitted or not.

`frailty.fam.pred`
: empirical Bayes prediction of the frailty term by family.

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

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, 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 Medecine*, **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.

McGilchrist CA and Aisbett CW (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

```
## Not run:

###---  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 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 ---###

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))

###--- 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")
```

```
## End(Not run)
```

---

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)
```

`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 σ^2_ϵ. 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,
      random, id, intercept = TRUE, link = "Random-effects",
      left.censoring = FALSE, n.knots, kappa, maxit = 350,
      hazard = "Splines", init.B, init.Random, init.Eta,
      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.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 2 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 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 default is "Random-effects".
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".
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 terminal event and then for the covariates related 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. 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 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 among 5, 7, 9, 12, 15, 20 and 32. 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.

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.

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.
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).

`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.
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 Guassian quadrature method used in the fit.
n.nodes	The number of nodes used for the Gaussian quadrature in the fit.

References

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.

A. Krol, L. Ferrer, JP. Pignon, C. Proust-Lima, M. Ducreux, O. Bouche, S. Michiels, V. Rondeau (2015). 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. *Submitted*.

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.

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)
```

```
# 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")

## 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 $t > D$, $dN^{R(l)*}(t), l \in 1,2$ 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.

The three 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 θ_1, θ_2 the variance of the random effects and α_1, α_2 the coefficients for these random effects into the terminal event part. If α_1 and θ_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 α_1). If α_2 and θ_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 α_2). If ρ, 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 ρ).

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 'fraityPenal' object. Methods defined for 'frailtyPenal' 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.
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

```
## Not run:

###--- 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)


## End(Not run)
```

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

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

```
## Not run:

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")
```

```
## End(Not run)
```

`plot.additivePenal` *Plot Method for an Additive frailty model.*

Description

Plots estimated baseline survival and hazard functions of an additive frailty model, more generally of a class 'additivePenal' object. 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, Xlab = "Time",
Ylab = "Hazard function", ...)
```

Arguments

`x`	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)
`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 HR and survival function of a class `additivePenal` object

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

plot(modAdd)


## End(Not run)
```

`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.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, 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)
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="surv",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,
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)
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,
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)

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.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'. Confidence bands are allowed.

Usage

```
## S3 method for class 'longiPenal'
plot(x, type.plot = "Hazard", conf.bands=TRUE, pos.legend=
"topright", cex.legend=0.7, main, color, 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"
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)
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",
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)
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, 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)
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,
Xlab = "Time", Ylab = "Hazard function", ...)
```

Arguments

x	A joint model, an object of class `trivPenall`.
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)
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)
```

---

prediction *Prediction probabilities for Cox proportionnal 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.*

Description

For Cox proportionnal 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 η

$$P^{cond}(t, t+w) = \frac{S_{ij}(t|\eta_i) - S_{ij}(t+w|\eta_i)}{S_{ij}(t|\eta_i)} = 1 - \left(\frac{S_0(t+w)}{S_0(t)}\right)^{\exp(\eta_i + \beta' Z_{ij})}$$

- 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 developping 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:

$$P^R(t,t+w) = P(X_{i(j+1)} \leq t+w | X_{i(j+1)} > t, D_i > t, H_i^J) =$$

$$\frac{\int_0^\infty [S^R_{i(J+1)}(t) - S^R_{i(J+1)}(t+w)] S_i^D(t) (u_i)^J S^R_{i(J+1)}(X_{ij}) g(u) du_i}{\int_0^\infty S^R_{i(J+1)}(t) S_i^D(t) (u_i)^J S^R_{i(J+1)}(X_{ij}) g(u) du_i}$$

It is possible to compute all these predictions in two ways : - either you want a cumulative probability of developping 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 (η), 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 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

$$\begin{aligned} P(t,t+w) &= P(D_i \leq t+w | D_i > t, H_i^{J,1}, \mathcal{Y}_i(t)) \\ &= \frac{\int_0^\infty [S_i^D(t) - S_i^D(t+w)])] \exp(J(v_i + g(t)^\top \boldsymbol{\eta}_R)) S_{i(J+1)}^R(t) f(\mathcal{Y}_i(t) | \boldsymbol{X}_{Li}, \boldsymbol{b}_i) f(\boldsymbol{u}_i) d\boldsymbol{u}_i}{\int_0^\infty S_i^D(t) \exp(J(v_i + g(t)^\top \boldsymbol{\eta}_R)) S_{i(J+1)}^R(t) f(\mathcal{Y}_i(t) | \boldsymbol{X}_{Li}, \boldsymbol{b}_i) f(\boldsymbol{u}_i) d\boldsymbol{u}_i} \end{aligned}$$

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)
```

Arguments

fit	A frailtyPenal or jointPenal object.
data	Dataframe for the prediction. See Details.
data.Longi	Dataframe for the prediction used for joint models with longitudinal data. See Details.
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". 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.
t	Time or vector of times for prediction.
window	Window or vector of windows for prediction.
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.

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 dataframe 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 dataframe 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 the dataframe for a conditional prediction on a shared frailty model, you need to specify the group to which the individuals belong to in adding the same cluster covariate as that used for model fitted. Here, the three individual belongs to the 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 dataframe with the relapse times, the indicator of event, the cluster variable and the covariates. Relapses occuring 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 dataframe 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 dataframe 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 dataframe, in the dataframe used for the joint model and in the dataframe 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 dataframe 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 proportionnal 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*.

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 dataframe 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
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)
#-- construction of the dataframe for predictions
datapred <- data.frame(group=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(5,1,2) # man, dukes 2
datapred[2,] <- c(5,2,3) # woman, dukes 3

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.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 dataframe 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 dataframe 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


#####################################################
##### prediction on a TRIVARIATE JOINT model ######
#####################################################

#-- construction of the dataframe 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)

# (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 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 given history of the biomarker
and recurrences pred.jointTri0 <- prediction(model.spli.RE.cal, datapredj,
datapredj_longi, t = 1, window = 2)
print(pred.jointTri0)

#-- prediction of death between 1 year and 1+w given history of the biomarker
and recurrences pred.jointTri <- prediction(model.spli.RE.cal, datapredj,
datapredj_longi, t = 1, window = seq(0.5, 2.5, 0.2), MC.sample = 100)
plot(pred.jointTri, 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
and recurrences pred.jointTri2 <- prediction(model.spli.RE.cal, datapredj,
datapredj_longi, t = seq(1, 2.5, 0.5), window = 0.5, MC.sample = 100)
plot(pred.jointTri2, conf.bands = TRUE)
# each y-value of the plot corresponds to the prediction between [x,x+0.5],
or in the next 0.5

## 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.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.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'.

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.

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

---

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.

`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

```
## Not run:

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")


## End(Not run)
```

---

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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width. Default is 6.
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.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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width. Default is 6.
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

summary(modSha)
summary(modCox)

#-- confidence interval at 99

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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width. Default is 6.
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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width. Default is 6.
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.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.

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%.
`d`	the desired number of digits after the decimal point. Default of 6 digits is used.
`len`	the total field width for the terminal part. Default is 6.
`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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width. Default is 6.
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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width. Default is 6.
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%.
d	the desired number of digits after the decimal point. Default of 6 digits is used.
len	the total field width for the terminal part. Default is 6.
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)
```

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)`

Examples

```
## Not run:

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)


## End(Not run)
```

`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 ζ 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 ζ_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 χ^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 χ^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

```
## Not run:

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)


## End(Not run)
```

`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):

$$\begin{cases} y_i(t_{ik}) = \boldsymbol{X}_{Li}(t_{ik})^\top \boldsymbol{\beta}_L + \boldsymbol{Z}_i(t_{ik})^\top \boldsymbol{b}_i + \epsilon_i(t_{ik}) & \text{(Longitudinal)} \\ r_{ij}(t|\boldsymbol{b}_i) = r_0(t)\exp(v_i + \boldsymbol{X}_{Rij}(t)\boldsymbol{\beta}_R + g(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))^\top \boldsymbol{\eta}_R) & \text{(Recurrent)} \\ \lambda_i(t|\boldsymbol{b}_i) = \lambda_0(t)\exp(\alpha v_i + \boldsymbol{X}_{Ti}(t)\boldsymbol{\beta}_T + h(\boldsymbol{b}_i, \boldsymbol{\beta}_L, \boldsymbol{Z}_i(t), \boldsymbol{X}_{Li}(t))^\top \boldsymbol{\eta}_T) & \text{(Terminal)} \end{cases}$$

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 σ^2_ϵ. 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 σ_v. Together with $\boldsymbol{b}_i$ constitutes the random effects of the model:

$$\boldsymbol{u}_i = \begin{pmatrix} \mathbf{b}_i \\ v_i \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} \mathbf{B}_1 & \mathbf{0} \\ \mathbf{0} & \sigma^2_v \end{pmatrix}\right),$$

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 2 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 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 (η_R) and for the terminal event (η_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 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.

`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 standard deviation of the frailty term (σ_v).

`alpha` The coefficient α associated with the frailty parameter in the terminal hazard function.

`ResidualSE` The standard deviation 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 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 Guassian quadrature method used in the fit.

`n.nodes` The number of nodes used for the Gaussian quadrature in the fit.

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. 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.

A. Krol, L. Ferrer, JP. Pignon, C. Proust-Lima, M. Ducreux, O. Bouche, S. Michiels, V. Rondeau (2015). 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. *Submitted*.

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.

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)
```

Index