Package: JM (via r-universe)

September 27, 2024

Title Joint Modeling of Longitudinal and Survival Data

Version 1.5-2

Date 2022-08-08

Author Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Maintainer Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Description Shared parameter models for the joint modeling of longitudinal and time-to-event data.

Depends R (>= 3.0.0), MASS, nlme, splines, survival

Enhances xtable

LazyLoad yes

LazyData yes

URL http://jmr.r-forge.r-project.org/

NeedsCompilation no

Date/Publication 2022-08-08 13:40:29 UTC

Repository https://drizopoulos.r-universe.dev

RemoteUrl https://github.com/cran/JM

RemoteRef HEAD

RemoteSha f00e607e886bc7a3fdd95cf95277c6ae6807349e

Contents

`aids` *Didanosine versus Zalcitabine in HIV Patients*

Description

A randomized clinical trial in which both longitudinal and survival data were collected to compare the efficacy and safety of two antiretroviral drugs in treating patients who had failed or were intolerant of zidovudine (AZT) therapy.

Format

A data frame with 1408 observations on the following 9 variables.

`patient` patients identifier; in total there are 467 patients.

`Time` the time to death or censoring.

`death` a numeric vector with 0 denoting censoring and 1 death.

`CD4` the CD4 cells count.

`obstime` the time points at which the CD4 cells count was recorded.

`drug` a factor with levels `ddC` denoting zalcitabine and `ddI` denoting didanosine.

`gender` a factor with levels `female` and `male`.

`prevOI` a factor with levels `AIDS` denoting previous opportunistic infection (AIDS diagnosis) at study entry, and `noAIDS` denoting no previous infection.

`AZT` a factor with levels `intolerance` and `failure` denoting AZT intolerance and AZT failure, respectively.

Note

The data frame `aids.id` contains the first CD4 cell count measurement for each patient. This data frame is used to fit the survival model.

References

Goldman, A., Carlin, B., Crane, L., Launer, C., Korvick, J., Deyton, L. and Abrams, D. (1996) Response of CD4+ and clinical consequences to treatment using ddI or ddC in patients with advanced HIV infection. *Journal of Acquired Immune Deficiency Syndromes and Human Retrovirology* **11**, 161–169.

Guo, X. and Carlin, B. (2004) Separate and joint modeling of longitudinal and event time data using standard computer packages. *The American Statistician* **58**, 16–24.

Examples

```
summary(aids.id)
```

`anova` *Anova Method for Fitted Joint Models*

Description

Produces marginal Wald tests or Performs a likelihood ratio test between two nested joint models.

Usage

```
## S3 method for class 'jointModel'
anova(object, object2, test = TRUE,
   process = c("both", "Longitudinal", "Event"), L = NULL, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`, nested in `object2`.
`object2`	an object inheriting from class `jointModel`.
`test`	logical; if `TRUE` the likelihood ratio test is performed.
`process`	for which of the two submodels to produce the marginal Wald tests table.
`L`	a numeric matrix of appropriate dimensions defining the contrasts of interest.
`...`	additional arguments; currently none is used.

Value

An object of class `aov.jointModel` with components,

`nam0`	the name of `object`.
`L0`	the log-likelihood under the null hypothesis (`object`).
`aic0`	the AIC value for the model given by `object`.
`bic0`	the BIC value for the model given by `object`.
`nam1`	the name of `object2`.
`L1`	the log-likelihood under the alternative hypothesis (`object2`).
`aic1`	the AIC value for the model given by `object2`.
`bic1`	the BIC value for the model given by `object2`.
`df`	the degrees of freedom for the test (i.e., the difference in the number of parameters).
`LRT`	the value of the Likelihood Ratio Test statistic (returned if `test = TRUE`).
`p.value`	the p-value of the test (returned if `test = TRUE`).
`aovTab.Y`	a data.frame with the marginal Wald tests for the longitudinal process; produced only when `object2` is missing.
`aovTab.T`	a data.frame with the marginal Wald tests for the event process; produced only when `object2` is missing.
`aovTab.L`	a data.frame with the marginal Wald tests for the user-defined contrasts matrix; produced only when `object2` is missing and `L` is not `NULL`.

Warning

The code minimally checks whether the models are nested! The user is responsible to supply nested models in order the LRT to be valid.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2010) JM: An R Package for the Joint Modelling of Longitudinal and Time-to-Event Data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

See Also

`jointModel`

Examples

```
## Not run:
# linear mixed model fit without treatment effect
fitLME.null <- lme(sqrt(CD4) ~ obstime,
    random = ~ 1 | patient, data = aids)
# cox model fit without treatment effect
fitCOX.null <- coxph(Surv(Time, death) ~ 1,
    data = aids.id, x = TRUE)
# joint model fit without treatment effect
fitJOINT.null <- jointModel(fitLME.null, fitCOX.null,
    timeVar = "obstime", method = "weibull-PH-aGH")

# linear mixed model fit with treatment effect
fitLME.alt <- lme(sqrt(CD4) ~ obstime * drug - drug,
    random = ~ 1 | patient, data = aids)
# cox model fit with treatment effect
fitCOX.alt <- coxph(Surv(Time, death) ~ drug,
    data = aids.id, x = TRUE)
# joint model fit with treatment effect
fitJOINT.alt <- jointModel(fitLME.alt, fitCOX.alt, timeVar = "obstime",
    method = "weibull-PH-aGH")

# likelihood ratio test for treatment effect
anova(fitJOINT.null, fitJOINT.alt)

## End(Not run)
```

---

aucJM *Time-Dependent AUCs for Joint Models*

Description

Using the available longitudinal information up to a starting time point, this function computes an estimate of the prediction error of survival at a horizon time point based on joint models.

Usage

```
aucJM(object, newdata, Tstart, ...)

## S3 method for class 'jointModel'
aucJM(object, newdata, Tstart, Thoriz = NULL,
    Dt = NULL, idVar = "id", simulate = FALSE, M = 100, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.

newdata	a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required. The names of the variables in this data frame must be the same as in the data frames that were used to fit the linear mixed effects model (using `lme()`) and the survival model (using `coxph()`) that were supplied as the two first argument of `jointModel`. In addition, this data frame should contain a variable that identifies the different subjects (see also argument `idVar`).
Tstart	numeric scalar denoting the time point up to which longitudinal information is to be used to derive predictions.
Thoriz	numeric scalar denoting the time point for which a prediction of the survival status is of interest; `Thoriz` mast be later than `Tstart` and either `Dt` or `Thoriz` must be specified. If `Thoriz` is `NULL` is set equal to `Tstart + Dt`.
Dt	numeric scalar denoting the length of the time interval of prediction; either `Dt` or `Thoriz` must be specified.
idVar	the name of the variable in `newdata` that identifies the different subjects.
simulate	logical; if `TRUE`, a Monte Carlo approach is used to estimate survival probabilities. If `FALSE`, a first order estimator is used instead. See `survfitJM` for mote details.
M	a numeric scalar denoting the number of Monte Carlo samples; see `survfitJM` for mote details.
...	additional arguments; currently none is used.

Details

Based on a fitted joint model (represented by `object`) and using the data supplied in argument `newdata`, this function computes the following estimate of the AUC:

$$\mathrm{AUC}(t, \Delta t) = \Pr\big[\pi_i(t + \Delta t \mid t) < \pi_j(t + \Delta t \mid t) \mid \{T_i^* \in (t, t + \Delta t]\} \cap \{T_j^* > t + \Delta t\}\big],$$

with i and j denote a randomly selected pair of subjects, and $\pi_i(t + \Delta t \mid t)$ and $\pi_j(t + \Delta t \mid t)$ denote the conditional survival probabilities calculated by `survfitJM` for these two subjects, for different time windows Δt specified by argument `Dt` using the longitudinal information recorded up to time `t = Tstart`.

The estimate of $\mathrm{AUC}(t, \Delta t)$ provided by `aucJM()` is in the spirit of Harrell's c-index, that is for the comparable subjects (i.e., the ones whose observed event times can be ordered), we compare their dynamic survival probabilities calculated by `survfitJM`. As with Harrell's c-index, $\mathrm{AUC}(t, \Delta t)$ does not fully take into account censoring, and therefore is expected to mask the true discriminative capability of the joint model under heavy censoring.

Value

A list of class `aucJM` with components:

auc	a numeric scalar denoting the estimated prediction error.
Tstart	a copy of the `Tstart` argument.
Thoriz	a copy of the `Thoriz` argument.
nr	a numeric scalar denoting the number of subjects at risk at time `Tstart`.

`classObject`	the class of `object`.
`nameObject`	the name of `object`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Antolini, L., Boracchi, P., and Biganzoli, E. (2005). A time-dependent discrimination index for survival data. *Statistics in Medicine* **24**, 3927–3944.

Harrell, F., Kerry, L. and Mark, D. (1996). Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine* **15**, 361–387.

Heagerty, P. and Zheng, Y. (2005). Survival model predictive accuracy and ROC curves. *Biometrics* **61**, 92–105.

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D., Murawska, M., Andrinopoulou, E.-R., Lesaffre, E. and Takkenberg, J. (2013). Dynamic predictions with time-dependent covariates in survival analysis: A comparison between joint modeling and landmarking. *under preparation*.

See Also

`survfitJM`, `dynCJM`, `jointModel`

Examples

```
## Not run:
# we construct the composite event indicator (transplantation or death)
pbc2$status2 <- as.numeric(pbc2$status != "alive")
pbc2.id$status2 <- as.numeric(pbc2.id$status != "alive")

# we fit the joint model using splines for the subject-specific
# longitudinal trajectories and a spline-approximated baseline
# risk function
lmeFit <- lme(log(serBilir) ~ ns(year, 3),
    random = list(id = pdDiag(form = ~ ns(year, 3))), data = pbc2)
survFit <- coxph(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)
jointFit <- jointModel(lmeFit, survFit, timeVar = "year",
    method = "piecewise-PH-aGH")

# AUC using data up to year 5 with horizon at year 8
aucJM(jointFit, pbc2, Tstart = 5, Thoriz = 8)

## End(Not run)
```

coef *Estimated Coefficients for Joint Models*

Description

Extracts estimated coefficients from fitted joint models.

Usage

```
## S3 method for class 'jointModel'
coef(object, process = c("Longitudinal", "Event"),
    include.splineCoefs = FALSE, ...)
## S3 method for class 'jointModel'
fixef(object, process = c("Longitudinal", "Event"),
    include.splineCoefs = FALSE, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`process`	for which model (i.e., linear mixed model or survival model) to extract the estimated coefficients.
`include.splineCoefs`	logical; if `TRUE` and the method argument in `jointModel()` is `"ch-Laplace"`, the estimated B-spline coefficients are included as well.
`...`	additional arguments; currently none is used.

Details

When `process = "Event"` both methods return the same output. However, for `process = "Longitudinal"`, the `coef()` method returns the subject-specific coefficients, whereas `fixef()` only the fixed effects.

Value

A numeric vector or a matrix of the estimated parameters for the fitted model.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

`ranef.jointModel`

Examples

```
## Not run:
# linear mixed model fit
fitLME <- lme(sqrt(CD4) ~ obstime * drug - drug,
    random = ~ 1 | patient, data = aids)
# cox model fit
fitCOX <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)

# joint model fit
fitJOINT <- jointModel(fitLME, fitCOX,
    timeVar = "obstime")

# fixed effects for the longitudinal process
fixef(fitJOINT)

# fixed effects + random effects estimates for the longitudinal
# process
coef(fitJOINT)

# fixed effects for the event process
fixef(fitJOINT, process = "Event")
coef(fitJOINT, process = "Event")

## End(Not run)
```

`crLong` *Transform Competing Risks Data in Long Format*

Description

In a competing risks setting this function expands the data frame with a single row per subject to the a data frame in long format in which each subject has as many rows as the number of competing events.

Usage

```
crLong(data, statusVar, censLevel,
    nameStrata = "strata", nameStatus = "status2")
```

Arguments

`data`	the data frame containing the competing risk data with a single row per subject.
`statusVar`	a character string denoting the name of the variable in `data` that identifies the status variable which equals 1 if the subject had any of the competing events and 0 otherwise.
`censLevel`	a character string or a scalar denoting the censoring level in the `statusVar` variable of `data`.

`nameStrata` a character string denoting the variable that will be added in the long version of `data` denoting the various causes of event.

`nameStatus` a character string denoting the variable that will be added in the long version of `data` denoting if the subject experience any of the competing events.

Value

A data frame in the long format with multiple rows per subject.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Putter, H., Fiocco, M., and Geskus, R. (2007). Tutorial in biostatistics: Competing risks and multi-state models. *Statistics in Medicine* **26**, 2389–2430.

Examples

```
head(crLong(pbc2.id, "status", "alive"))
```

`DerivSplines` *Derivatives and Integrals of B-splines and Natural Cubic splines*

Description

Numerical derivatives and integrals of functions `bs()` and `ns()` at their first argument.

Usage

```
dns(x, df = NULL, knots = NULL, intercept = FALSE,
    Boundary.knots = range(x), eps = 1e-03)

dbs(x, df = NULL, knots = NULL, intercept = FALSE,
    Boundary.knots = range(x), eps = 1e-03)

ins(x, df = NULL, knots = NULL, intercept = FALSE,
    Boundary.knots = range(x), from = 0, weight.fun = NULL, ...)

ibs(x, df = NULL, knots = NULL, intercept = FALSE,
    Boundary.knots = range(x), from = 0, weight.fun = NULL, ...)
```

Arguments

`x`, `df`, `knots`, `intercept`, `Boundary.knots`
: see the help pages of functions `ns()` and `bs()`.

`eps`
: a numeric scalar denoting the step length for the central difference approximation, which calculates the derivative.

`from`
: a numeric scalar denoting the lower limit of the integral.

`weight.fun`
: a function to applied as weights.

`...`
: extra arguments passed to `weight.fun`.

Value

an object of class `dns`, `dbs`, `ins` or `ibs`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Examples

```
x <- rnorm(10)
dns(x, df = 4)
ins(x, df = 4)
```

---

dynCJM — *A Dynamic Discrimination Index for Joint Models*

Description

This function computes a dynamic discrimination index for joint models based on a weighted average of time-dependent AUCs.

Usage

```
dynCJM(object, newdata, Dt, ...)

## S3 method for class 'jointModel'
dynCJM(object, newdata, Dt, idVar = "id", t.max = NULL,
    simulate = FALSE, M = 100, weightFun = NULL, ...)
```

Arguments

`object`
: an object inheriting from class `jointModel`.

`newdata` a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required. The names of the variables in this data frame must be the same as in the data frames that were used to fit the linear mixed effects model (using `lme()`) and the survival model (using `coxph()`) that were supplied as the two first argument of `jointModel`. In addition, this data frame should contain a variable that identifies the different subjects (see also argument `idVar`).

`Dt` a numeric scalar denoting the time frame within which the occurrence of events is of interest.

`idVar` the name of the variable in `newdata` that identifies the different subjects.

`t.max` a numeric scalar denoting the time maximum follow-up time up to which the dynamic discrimination index is to be calculated. If `NULL`, it is set equal to `max(Time) + 1e-05` where `Time` denotes the observed event times.

`simulate` logical; if `TRUE`, a Monte Carlo approach is used to estimate survival probabilities. If `FALSE`, a first order estimator is used instead. See `survfitJM` for mote details.

`M` a numeric scalar denoting the number of Monte Carlo samples; see `survfitJM` for mote details.

`weightFun` a function of two arguments the first denoting time and the second the length of the time frame of interest, i.e., `Dt`.

`...` additional arguments; currently none is used.

Details

(**Note:** The following contain some math formulas, which are better viewed in the pdf version of the manual accessible at https://cran.r-project.org/package=JM.)

Function `dynC` computes the following discrimination index

$$\mathrm{C}_{dyn}^{\Delta t} = \int_0^{t_{max}} \mathrm{AUC}(t, \Delta t) \, \mathrm{Pr}\{\mathcal{E}(t, \Delta t)\} \; dt \Big/ \int_0^{t_{max}} \mathrm{Pr}\{\mathcal{E}(t, \Delta t)\} \; dt,$$

where

$$\mathrm{AUC}(t, \Delta t) = \mathrm{Pr}\big[\pi_i(t + \Delta t \mid t) < \pi_j(t + \Delta t \mid t) \mid \{T_i^* \in (t, t + \Delta t]\} \cap \{T_j^* > t + \Delta t\}\big],$$

and

$$\mathcal{E}(t, \Delta t) = \big[\{T_i^* \in (t, t + \Delta t]\} \cap \{T_j^* > t + \Delta t\}\big],$$

with i and j denote a randomly selected pair subjects, and $\pi_i(t+\Delta t \mid t)$ and $\pi_j(t+\Delta t \mid t)$ denote the conditional survival probabilities calculated by `survfitJM` for these two subjects, for different time windows Δt specified by argument `Dt`. The upper limit of integral in specified by argument `t.max`. The integrals in the numerator and denominator are approximated using a 15-point Gauss-Kronrod quadrature rule.

Index $\mathrm{C}_{dyn}^{\Delta t}$ is in the spirit of Harrell's c-index, that is for the comparable subjects (i.e., the ones whose observed event times can be ordered), we compare their dynamic survival probabilities calculated by `survfitJM`. As with Harrell's c-index, $\mathrm{C}_{dyn}^{\Delta t}$ does not take into account censoring, and therefore is expected to mask the true discriminative capability of the joint model under heavy censoring.

Value

A list of class dynCJM with components:

dynC	a numeric scalar denoting the dynamic discrimination index.
times	a numeric vector of time points at which the AUC was calculated.
AUCs	a numeric vector of the estimated AUCs at the aforementioned time points.
weights	a numeric vector of the estimated weights at the aforementioned time points.
t.max	a copy of the t.max argument.
Dt	a copy of the Dt argument.
classObject	the class of object.
nameObject	the name of object.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Antolini, L., Boracchi, P., and Biganzoli, E. (2005). A time-dependent discrimination index for survival data. *Statistics in Medicine* **24**, 3927–3944.

Harrell, F., Kerry, L. and Mark, D. (1996). Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine* **15**, 361–387.

Heagerty, P. and Zheng, Y. (2005). Survival model predictive accuracy and ROC curves. *Biometrics* **61**, 92–105.

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D., Murawska, M., Andrinopoulou, E.-R., Lesaffre, E. and Takkenberg, J. (2013). Dynamic predictions with time-dependent covariates in survival analysis: A comparison between joint modeling and landmarking. *under preparation*.

See Also

survfitJM, aucJM, jointModel

Examples

```
## Not run:
# we construct the composite event indicator (transplantation or death)
pbc2$status2 <- as.numeric(pbc2$status != "alive")
pbc2.id$status2 <- as.numeric(pbc2.id$status != "alive")

# we fit the joint model using splines for the subject-specific
# longitudinal trajectories and a spline-approximated baseline
```

```
# risk function
lmeFit <- lme(log(serBilir) ~ ns(year, 3),
    random = list(id = pdDiag(form = ~ ns(year, 3))), data = pbc2)
survFit <- coxph(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)
jointFit <- jointModel(lmeFit, survFit, timeVar = "year",
    method = "piecewise-PH-aGH")

# dynamic discrimination index up to year 10 using a two-year interval
dynCJM(jointFit, pbc2, Dt = 2, t.max = 10)

## End(Not run)
```

`fitted` *Fitted Values for Joint Models*

Description

Calculates fitted values for joint models.

Usage

```
## S3 method for class 'jointModel'
fitted(object, process = c("Longitudinal", "Event"),
type = c("Marginal", "Subject", "EventTime", "Slope"), scale = c("survival",
"cumulative-Hazard", "log-cumulative-Hazard"), M = 200, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`process`	for which model (i.e., linear mixed model or survival model) to calculate the fitted values.
`type`	what type of fitted values to calculate for the survival outcome. See **Details**.
`scale`	in which scale to calculate; relevant only when `process = "Event"`.
`M`	how many times to simulate random effects; see **Details** for more info.
`...`	additional arguments; currently none is used.

Details

For `process = "Longitudinal"`, let X denote the design matrix for the fixed effects β, and Z the design matrix for the random effects b. Then for `type = "Marginal"` the fitted values are $X\hat{\beta}$, whereas for `type = "Subject"` they are $X\hat{\beta} + Z\hat{b}$. For `type = "EventTime"` is the same as `type = "Subject"` but evaluated at the observed event times. Finally, `type == "Slope"` returns $Xs\hat{\beta} + Zs\hat{b}$ where Xs and Zs denote the fixed- and random-effects design matrices corresponding to the slope term which is specified in the `derivForm` argument of `jointModel`.

For `process = "Event"` and `type = "Subject"` the linear predictor conditional on the random effects estimates is calculated for each sample unit. Depending on the value of the `scale` argument the fitted survival function, cumulative hazard function or log cumulative hazard function is returned. For `type = "Marginal"`, random effects values for each sample unit are simulated `M` times from a normal distribution with zero mean and covariance matrix the estimated covariance matrix for the random effects. The marginal survival function for the ith sample unit is approximated by

$$S_i(t) = \int S_i(t|b_i)p(b_i)db_i \approx (1/M)\sum_{m=1}^{M} S_i(t|b_{im}),$$

where $p(b_i)$ denotes the normal probability density function, and b_{im} the mth simulated value for the random effect of the ith sample unit. The cumulative hazard and log cumulative hazard functions are calculated as $H_i(t) = -\log S_i(t)$ and $\log H_i(t) = \log\{-\log S_i(t)\}$, respectively.

Value

a numeric vector of fitted values.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2010) JM: An R Package for the Joint Modelling of Longitudinal and Time-to-Event Data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

See Also

`residuals.jointModel`

Examples

```
## Not run:
# linear mixed model fit
fitLME <- lme(log(serBilir) ~ drug * year,
    random = ~ 1 | id, data = pbc2)
# survival regression fit
fitSURV <- survreg(Surv(years, status2) ~ drug,
    data = pbc2.id, x = TRUE)
# joint model fit, under the (default) Weibull model
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year")

# fitted for the longitudinal process
head(cbind(
    "Marg" = fitted(fitJOINT),
    "Subj" = fitted(fitJOINT, type = "Subject")
))

# fitted for the event process - survival function
head(cbind(
```

```
    "Marg" = fitted(fitJOINT, process = "Ev"),
    "Subj" = fitted(fitJOINT, process = "Ev", type = "Subject")
))

# fitted for the event process - cumulative hazard function
head(cbind(
    "Marg" = fitted(fitJOINT, process = "Ev",
        scale = "cumulative-Hazard"),
    "Subj" = fitted(fitJOINT, process = "Ev", type = "Subject",
        scale = "cumulative-Hazard")
))

## End(Not run)
```

JM *Joint Modeling of Longitudinal and Time-to-Event Data in R*

Description

This package fits shared parameter models for the joint modeling of normal longitudinal responses and event times under a maximum likelihood approach. Various options for the survival model and optimization/integration algorithms are provided.

Details

Package:	JM
Type:	Package
Version:	1.5-2
Date:	2022-08-08
License:	GPL

The package has a single model-fitting function called `jointModel`, which accepts as main arguments a linear mixed effects object fit returned by function `lme()` of package **nlme**, and a survival object fit returned by either function `coxph()` or function `survreg()` of package **survival**. In addition, the `method` argument of `jointModel()` specifies the type of the survival submodel to be fitted and the type of the numerical integration technique; available options are:

`"Cox-PH-GH"` the time-dependent version of a proportional hazards model with unspecified baseline hazard function. The Gauss-Hermite integration rule is used to approximate the required integrals. (This option corresponds to the joint model proposed by Wulfsohn and Tsiatis, 1997)

`"weibull-PH-GH"` the Weibull model under the proportional hazards formulation. The Gauss-Hermite integration rule is used to approximate the required integrals.

`"weibull-AFT-GH"` the Weibull model under the accelerated failure time formulation. The Gauss-Hermite integration rule is used to approximate the required integrals.

`"piecewise-PH-GH"` a proportional hazards model with a piecewise constant baseline risk function. The Gauss-Hermite integration rule is used to approximate the required integrals.

`"spline-PH-GH"` a proportional hazards model, in which the log baseline hazard is approximated using B-splines. The Gauss-Hermite integration rule is used to approximate the required integrals.

`"ch-Laplace"` an additive log cumulative hazard model, in which the log cumulative baseline hazard is approximated using B-splines. A fully exponential Laplace approximation method is used to approximate the required integrals (Rizopoulos et al., 2009).

For all the above mentioned options (except the last one), a pseudo-adaptive Gauss-Hermite integration rule is also available (Rizopoulos, 2012b). This is much faster than the classical Gauss-Hermite rule, and in several simulations it has been shown to perform equally well (though its performance is still under investigation).

The package also offers several utility functions that can extract useful information from fitted joint models. The most important of those are included in the **See also** section below.

Author(s)

Dimitris Rizopoulos

Maintainer: Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Henderson, R., Diggle, P. and Dobson, A. (2000) Joint modelling of longitudinal measurements and event time data. *Biostatistics* **1**, 465–480.

Rizopoulos, D. (2012a) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2012b) 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.

Rizopoulos, D. (2011) Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D. (2010) JM: An R package for the joint modelling of longitudinal and time-to-event data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

Rizopoulos, D., Verbeke, G. and Lesaffre, E. (2009) Fully exponential Laplace approximation for the joint modelling of survival and longitudinal data. *Journal of the Royal Statistical Society, Series B* **71**, 637–654.

Rizopoulos, D., Verbeke, G. and Molenberghs, G. (2010) Multiple-imputation-based residuals and diagnostic plots for joint models of longitudinal and survival outcomes. *Biometrics* **66**, 20–29.

Tsiatis, A. and Davidian, M. (2004) Joint modeling of longitudinal and time-to-event data: an overview. *Statistica Sinica* **14**, 809–834.

Wulfsohn, M. and Tsiatis, A. (1997) A joint model for survival and longitudinal data measured with error. *Biometrics* **53**, 330–339.

See Also

jointModel, survfitJM, rocJM, aucJM, dynCJM, prederrJM, predict

---

jointModel *Joint Models for Longitudinal and Survival Data*

Description

This function fits shared parameter models for the joint modelling of normal longitudinal responses and time-to-event data under a maximum likelihood approach. Various options for the survival model are available.

Usage

```
jointModel(lmeObject, survObject, timeVar,
  parameterization = c("value", "slope", "both"),
  method = c("weibull-PH-aGH", "weibull-PH-GH", "weibull-AFT-aGH",
    "weibull-AFT-GH", "piecewise-PH-aGH", "piecewise-PH-GH",
    "Cox-PH-aGH", "Cox-PH-GH", "spline-PH-aGH", "spline-PH-GH",
    "ch-Laplace"),
  interFact = NULL, derivForm = NULL, lag = 0, scaleWB = NULL,
  CompRisk = FALSE, init = NULL, control = list(), ...)
```

Arguments

lmeObject	an object inheriting from class lme (see also **Note**).
survObject	an object inheriting from class coxph or class survreg. In the call to coxph() or survreg(), you need to specify the argument x = TRUE such that the design matrix is contained in the object fit. See **Examples**.
timeVar	a character string indicating the time variable in the linear mixed effects model.
parameterization	a character string indicating the type of parameterization. See **Details**
method	a character string specifying the type of joint model to fit. See **Details**.
interFact	a list with components value a formula for the interaction terms corresponding to the value parameterization, slope a formula for the interaction terms corresponding to the slope parameterization, data a data frame containing these variables (this should have the same number of rows and ordering of subjects, as the one in survObject).
derivForm	a list with components fixed a formula representing the derivative of the fixed-effects part of the liner mixed model with respect to time, indFixed a numeric vector indicating which fixed effects of lmeObject correspond to the derivative, random a formula representing the derivative of the random-effects part of the liner mixed model with respect to time, and indRamdom a numeric vector indicating which random effects of lmeObject correspond to the derivative. When a random intercepts linear mixed model is assumed, then random = ~ 1

and `indRandom = FALSE`. Required only when `parameterization == "slope"` or `parameterization == "both"`. See **Examples**.

`lag` a numeric scalar denoting a lag effect in the time-dependent covariate represented by the mixed model; default is 0.

`scaleWB` a numeric scalar denoting a fixed value for the scale parameter of the Weibull hazard; used only when `method = "weibull-AFT-GH"` or `method = "weibull-PH-GH"`. The default `NULL` means that the scale parameter is estimated.

`CompRisk` logical; should a competing risks joint model be fitted.

`init` a named list of user-specified initial values:

betas the vector of fixed effects for the linear mixed effects model.

sigma the measurement error standard deviation for the linear mixed effects model.

D the variance-covariance matrix of the random effects.

gammas the vector of baseline covariates for the survival model. For `method = "ch-Laplace"` this vector should first contain initial values for the sorted B-spline coefficients used to model the log cumulative baseline hazard.

alpha the association parameters.

Dalpha the association parameters for the true slopes parameterization

xi the vector of baseline risk function values within the intervals specified by the knots; specified only when `method = "piecewise-PH-GH"`.

gammas.bs the vector of spline coefficients; specified only when `method = "spline-PH-GH"`.

sigma.t the scale parameter for the Weibull baseline risk function; specified only when `method = "weibull-AFT-GH"` or `method = "weibull-PH-GH"`.

lambda0 a vector of the baseline hazard values at the sorted unique event times; specified only when `method = "Cox-PH-GH"`.

When this list of initial values does not contain some of these components or contains components not of the appropriate length, then the default initial values are used instead.

`control` a list of control values with components:

only.EM logical; if `TRUE` only the EM algorithm is used in the optimization, otherwise if convergence has not been achieved a quasi-Newton algorithm is initiated. Default is `FALSE` except for `method = "Cox-PH-GH"` for which only the EM algorithm is available.

iter.EM the number of EM iterations. Default is 50 except for `method = "Cox-PH-GH"` for which the default is 200.

iter.qN the number of quasi-Newton iterations. Default is 150.

optimizer a character string indicating which optimizer to use; options are "optim" (default) and "nlminb".

tol1 tolerance value for convergence in the parameters; see **Details**. Default is 1e-03.

tol2 tolerance value for convergence in the parameters; see **Details**. Default is 1e-04.

tol3 tolerance value for convergence in the log-likelihood; see **Details**. Default is `sqrt(.Machine$double.eps)`.

numeriDeriv a character string indicating which type of numerical derivative to use to compute the Hessian matrix; options are "fd" (default) denoting the forward difference approximation, and "cd" denoting the central difference approximation.

eps.Hes tolerance value used in the numerical derivative method. Default is 1e-06; if you choose `numeriDeriv = "cd"` a larger value (e.g., 1e-04) is suggested.

parscale the `parscale` control argument for `optim()`, or the `scale` argument for `nlminb()`. It should be a numeric vector of length equal to the number of parameters. Default is 0.01 for all parameters.

step.max tolerance value for the maximum step size in the Newton-Raphson algorithm used to update the parameters of the survival submodel for `method = "ch-Laplace"`. Default is 0.1.

backtrackSteps the number of backtrack steps to use when updating the parameters of the survival submodel under `method = "ch-Laplace"`.

knots a numeric vector of the knots positions for the piecewise constant baseline risk function of for the log times used in the B-splines approximation of the log cumulative baseline hazard; therefore, this argument is relevant only when `method = "piecewise-PH-GH"`, `method = "spline-PH-GH"` or `method = "ch-Laplace"`. The default is to place equally-spaced `lng.in.kn` knots in the quantiles of the observed event times. For stratified models fitted with `method = "spline-PH-GH"` this should be a list with elements numeric vectors of knots positions for each strata.

ObsTimes.knots logical; if `TRUE` (default) the positions of the knots are specified based in the observed event times, otherwise the positions of the knots are specified using only the true event times.

lng.in.kn the number of internal knots; relevant only when when `method = "piecewise-PH-GH"` where it denotes the number of internal knots for the piecewise constant baseline risk function or when `method = "spline-PH-GH"` or `method = "ch-Laplace"` where it denotes the number of internal knots for B-splines approximation of the log baseline hazard. Default is 6 when `method = "piecewise-PH-GH"` and 5 otherwise.

equal.strata.knots logical; if `TRUE` (the default), then the same knots are used in the approximation of the baseline risk function in different strata when `method = "spline-PH-GH"`.

ord a positive integer denoting the order of the B-splines used to approximate the log cumulative hazard (default is 4); relevant only when `method = "spline-PH-GH"` or `method = "ch-Laplace"`.

typeGH a character string indicating the type of Gauss-Hermite rule to be used. Options are "simple" and "adaptive". The default is "simple" but it is turned to adaptive when the user specifies in the `method` argument an option that contains `aGH`.

GHk the number of Gauss-Hermite quadrature points used to approximate the integrals over the random effects. The default is 15 for one- or two-dimensional integration and for $N < 2000$, and 9 otherwise for the simple Gauss-Hermite rule, and 5 for one-, two-dimensional or three-dimensional integration and for $N < 2000$, and 3 otherwise for the pseudo adaptive Gauss-Hermite rule, where N denotes the total number of longitudinal measurements.

GKk the number of Gauss-Kronrod points used to approximate the integral involved in the calculation of the survival function. Two options are available, namely 7 or 15. For `method = "weibull-PH-GH"`, `method = "weibull-AFT-GH"` and `method = "spline-PH-GH"` 15 are used, whereas for `method = "piecewise-PH-GH"` 7.

verbose logical; if `TRUE`, the parameter estimates and the log-likelihood value are printed during the optimization procedure. Default is `FALSE`.

`...` options passed to the `control` argument.

Details

Function `jointModel` fits joint models for longitudinal and survival data (more detailed information about the formulation of these models can be found in Rizopoulos (2010)). For the longitudinal responses the linear mixed effects model represented by the `lmeObject` is assumed. For the survival times let w_i denote the vector of baseline covariates in `survObject`, with associated parameter vector γ, $m_i(t)$ the value of the longitudinal outcome at time point t as approximated by the linear mixed model (i.e., $m_i(t)$ equals the fixed-effects part + random-effects part of the linear mixed effects model for sample unit i), α the association parameter for $m_i(t)$, $m_i'(t)$ the derivative of $m_i(t)$ with respect to t, and α_d the association parameter for $m_i'(t)$. Then, for `method = "weibull-AFT-GH"` a time-dependent Weibull model under the accelerated failure time formulation is assumed. For `method = "weibull-PH-GH"` a time-dependent relative risk model is postulated with a Weibull baseline risk function. For `method = "piecewise-PH-GH"` a time-dependent relative risk model is postulated with a piecewise constant baseline risk function. For `method = "spline-PH-GH"` a time-dependent relative risk model is assumed in which the log baseline risk function is approximated using B-splines. For `method = "ch-Laplace"` an additive model on the log cumulative hazard scale is assumed (see Rizopoulos et al., 2009 for more info). Finally, for `method = "Cox-PH-GH"` a time-dependent relative risk model is assumed where the baseline risk function is left unspecified (Wulfsohn and Tsiatis, 1997). For all these options the linear predictor for the survival submodel is written as

$$\eta = \gamma^\top w_i + \alpha m_i\{max(t-k, 0)\},$$

when `parameterization = "value"`,

$$\eta = \gamma^\top w_i + \alpha_s m_i'\{max(t-k, 0)\},$$

when `parameterization = "slope"`, and

$$\eta = \gamma^\top w_i + \alpha m_i\{max(t-k, 0)\} + \alpha_s m_i'\{max(t-k, 0)\},$$

when `parameterization = "both"`, where in all the above the value of k is specified by the `lag` argument and $m_i'(t) = dm_i(t)/dt$. If `interFact` is specified, then $m_i\{max(t-k, 0)\}$ and/or $m_i'\{max(t-k, 0)\}$ are multiplied with the design matrices derived from the formulas supplied as the first two arguments of `interFact`, respectively. In this case α and/or α_s become vectors of association parameters.

For `method = "spline-PH-GH"` it is also allowed to include stratification factors. These should be included in the specification of the `survObject` using function `strata()`. Note that in this case `survObject` must only be a 'coxph' object.

For all survival models except for the time-dependent proportional hazards model, the optimization algorithm starts with EM iterations, and if convergence is not achieved, it switches to quasi-Newton iterations (i.e., BFGS in `optim()` or `nlminb()`, depending on the value of the `optimizer` control argument). For `method = "Cox-PH-GH"` only the EM algorithm is used. During the EM iterations, convergence is declared if either of the following two conditions is satisfied: (i) $L(\theta^{it}) - L(\theta^{it-1}) < tol_3\{|L(\theta^{it-1})| + tol_3\}$, or (ii) $\max\{|\theta^{it} - \theta^{it-1}|/(|\theta^{it-1}| + tol_1)\} < tol_2$, where θ^{it} and θ^{it-1} is the vector of parameter values at the current and previous iterations, respectively, and $L(.)$ is the log-likelihood function. The values for tol_1, tol_2 and tol_3 are specified via the `control` argument. During the quasi-Newton iterations, the default convergence criteria of either `optim()` or `nlminb()` are used.

The required integrals are approximated using the standard Gauss-Hermite quadrature rule when the chosen option for the `method` argument contains the string "GH", and the (pseudo) adaptive Gauss-Hermite rule when the chosen option for the `method` argument contains the string "aGH". For `method = "ch-Laplace"` the fully exponential Laplace approximation described in Rizopoulos et al. (2009) is used. The (pseudo) adaptive Gauss-Hermite and the Laplace approximation are particularly useful when high-dimensional random effects vectors are considered (e.g., when modelling nonlinear subject-specific trajectories with splines or high-order polynomials).

Value

See `jointModelObject` for the components of the fit.

Note

1. The `lmeObject` argument should represent a linear mixed model object with a simple random-effects structure, i.e., only the `pdDiag()` class is currently allowed.

2. The `lmeObject` object should not contain any within-group correlation structure (i.e., `correlation` argument of `lme()`) or within-group heteroscedasticity structure (i.e., `weights` argument of `lme()`).

3. It is assumed that the linear mixed effects model `lmeObject` and the survival model `survObject` have been fitted to the same subjects. Moreover, it is assumed that the ordering of the subjects is the same for both `lmeObject` and `survObject`, i.e., that the first line in the data frame containing the event times corresponds to the first set of lines identified by the grouping variable in the data frame containing the repeated measurements, and so on.

4. In the `print` and `summary` generic functions for class `jointModel`, the estimated coefficients (and standard errors for the `summary` generic) for the event process are augmented with the element "Assoct" that corresponds to the association parameter α and the element "Assoct.s" that corresponds to the parameter α_s when `parameterization` is `"slope"` or `"both"` (see **Details**).

5. The standard errors returned by the `summary` generic function for class `jointModel` when `method = "Cox-PH-GH"` are based on the profile score vector (i.e., given the NPMLE for the unspecified baseline hazard). Hsieh et al. (2006) have noted that these standard errors are underestimated.

6. As it is the case for all types of mixed models that require numerical integration, it is advisable (especially in difficult datasets) to check the stability of the maximum likelihood estimates with an increasing number of Gauss-Hermite quadrature points.

7. It is assumed that the scale of the time variable (e.g., days, months years) is the same in both `lmeObject` and `survObject`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Henderson, R., Diggle, P. and Dobson, A. (2000) Joint modelling of longitudinal measurements and event time data. *Biostatistics* **1**, 465–480.

Hsieh, F., Tseng, Y.-K. and Wang, J.-L. (2006) Joint modeling of survival and longitudinal data: Likelihood approach revisited. *Biometrics* **62**, 1037–1043.

Rizopoulos, D. (2012a) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2012b) 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.

Rizopoulos, D. (2011) Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D. (2010) JM: An R package for the joint modelling of longitudinal and time-to-event data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

Rizopoulos, D., Verbeke, G. and Lesaffre, E. (2009) Fully exponential Laplace approximations for the joint modelling of survival and longitudinal data. *Journal of the Royal Statistical Society, Series B* **71**, 637–654.

Rizopoulos, D., Verbeke, G. and Molenberghs, G. (2010) Multiple-imputation-based residuals and diagnostic plots for joint models of longitudinal and survival outcomes. *Biometrics* **66**, 20–29.

Tsiatis, A. and Davidian, M. (2004) Joint modeling of longitudinal and time-to-event data: an overview. *Statistica Sinica* **14**, 809–834.

Wulfsohn, M. and Tsiatis, A. (1997) A joint model for survival and longitudinal data measured with error. *Biometrics* **53**, 330–339.

See Also

`jointModelObject`, `anova.jointModel`, `coef.jointModel`, `fixef.jointModel`, `ranef.jointModel`, `fitted.jointModel`, `residuals.jointModel`, `plot.jointModel`, `survfitJM`, `rocJM`, `dynCJM`, `aucJM`, `prederrJM`

Examples

```
## Not run:
# linear mixed model fit (random intercepts)
fitLME <- lme(log(serBilir) ~ drug * year, random = ~ 1 | id, data = pbc2)
# survival regression fit
fitSURV <- survreg(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)
# joint model fit, under the (default) Weibull model
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year")
```

```
fitJOINT
summary(fitJOINT)

# linear mixed model fit (random intercepts + random slopes)
fitLME <- lme(log(serBilir) ~ drug * year, random = ~ year | id, data = pbc2)
# survival regression fit
fitSURV <- survreg(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)
# joint model fit, under the (default) Weibull model
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year")
fitJOINT
summary(fitJOINT)

# we also include an interaction term of log(serBilir) with drug
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year",
    interFact = list(value = ~ drug, data = pbc2.id))
fitJOINT
summary(fitJOINT)


# a joint model in which the risk for and event depends both on the true value of
# marker and the true value of the slope of the longitudinal trajectory
lmeFit <- lme(sqrt(CD4) ~ obstime * drug, random = ~ obstime | patient, data = aids)
coxFit <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)

# to fit this model we need to specify the 'derivForm' argument, which is a list
# with first component the derivative of the fixed-effects formula of 'lmeFit' with
# respect to 'obstime', second component the indicator of which fixed-effects
# coefficients correspond to the previous defined formula, third component the
# derivative of the random-effects formula of 'lmeFit' with respect to 'obstime',
# and fourth component the indicator of which random-effects correspond to the
# previous defined formula
dForm <- list(fixed = ~ 1 + drug, indFixed = c(2, 4), random = ~ 1, indRandom = 2)
jointModel(lmeFit, coxFit, timeVar = "obstime", method = "spline-PH-aGH",
  parameterization = "both", derivForm = dForm)


# Competing Risks joint model
# we first expand the PBC dataset in the competing risks long format
# with two competing risks being death and transplantation
pbc2.idCR <- crLong(pbc2.id, "status", "alive")

# we fit the linear mixed model as before
lmeFit.pbc <- lme(log(serBilir) ~ drug * ns(year, 3),
    random = list(id = pdDiag(form = ~ ns(year, 3))), data = pbc2)

# however, for the survival model we need to use the data in the long
# format, and include the competing risks indicator as a stratification
# factor. We also take interactions of the baseline covariates with the
# stratification factor in order to allow the effect of these covariates
# to be different for each competing risk
coxCRFit.pbc <- coxph(Surv(years, status2) ~ (drug + sex)*strata + strata(strata),
    data = pbc2.idCR, x = TRUE)
```

```
# the corresponding joint model is fitted simply by including the above
# two submodels as main arguments, setting argument CompRisk to TRUE,
# and choosing as method = "spline-PH-aGH". Similarly as above, we also
# include strata as an interaction factor to allow serum bilirubin to
# have a different effect for each of the two competing risks
jmCRFit.pbc <- jointModel(lmeFit.pbc, coxCRFit.pbc, timeVar = "year",
    method = "spline-PH-aGH",
    interFact = list(value = ~ strata, data = pbc2.idCR),
    CompRisk = TRUE)
summary(jmCRFit.pbc)

# linear mixed model fit
fitLME <- lme(sqrt(CD4) ~ obstime * drug - drug,
    random = ~ 1 | patient, data = aids)
# cox model fit
fitCOX <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)
# joint model fit with a spline-approximated baseline hazard function
fitJOINT <- jointModel(fitLME, fitCOX,
    timeVar = "obstime", method = "spline-PH-aGH")
fitJOINT
summary(fitJOINT)

## End(Not run)
```

`jointModelObject` *Fitted jointModel Object*

Description

An object returned by the `jointModel` function, inheriting from class `jointModel` and representing a fitted joint model for longitudinal and time-to-event data. Objects of this class have methods for the generic functions `anova`, `coef`, `fitted`, `fixed.effects`, `logLik`, `plot`, `print`, `random.effects`, `residuals`, `summary`, and `vcov`.

Value

The following components must be included in a legitimate `jointModel` object.

`coefficients` a list with the estimated coefficients. The components of this list are:

- **betas** the vector of fixed effects for the linear mixed effects model.
- **sigma** the measurement error standard deviation for the linear mixed effects model.
- **gammas** the vector of baseline covariates for the survival model.
- **alpha** the association parameter(s).
- **Dalpha** the association parameter(s) corresponding to the slope of the true trajectory.
- **sigma.t** the scale parameter for the Weibull survival model; returned only when `method = "weibull-PH-GH"` or `method = "weibull-AFT-GH"`.

xi the parameter of the piecewise constant baseline hazard; returned only when `method = "piecewise-PH-GH"`.

gamma.bs the coefficients of the B-splines use to approximate the baseline hazard; returned only when `method = "spline-PH-GH"`.

lambda0 a two-column numeric matrix with the first column containing the estimated baseline hazard values, and the second the unique sorted event times; returned only when `method = "Cox-PH-GH"`.

D the variance-covariance matrix of the random effects.

`Hessian` the Hessian matrix evaluated at the estimated parameter values.

`logLik` the log-likelihood value.

`EB` a list with components:

post.b the estimated random effects values.

post.vb the estimated variance for the random effects estimates.

Zb the estimated random effects part of the linear predictor for the longitudinal outcome (i.e., Z is the design matrix for the random effects b).

Ztimeb the estimated random effects part of the linear predictor for the survival outcome (i.e., evaluated at the observed event times).

Ztime2b the estimated random effects part of the linear predictor for the survival outcome (i.e., for the ith sample unit is evaluated at all event times that are less or equal to the ith observed event time); returned only when `method = "Cox-PH-GH"`.

`knots` the numeric vector of the knots positions; returned only when `method = "spline-PH-GH"`, `method = "piecewise-PH-GH"` or `method = "ch-Laplace"`.

`iters` the number of iterations in the optimization algorithm.

`convergence` convergence identifier: 0 corresponds to successful convergence, whereas 1 to a problem (i.e., when 1, usually more iterations are required).

`n` the number of sample units.

`N` the total number of repeated measurements for the longitudinal outcome.

`ni` a vector with the number of repeated measurements for each sample unit.

`d` a numeric vector with 0 denoting censored observation and 1 events.

`id` the grouping vector for the longitudinal responses.

`x` a list with the design matrices for the longitudinal and event processes.

`y` a list with the response vectors for the longitudinal and event processes.

`data.id` a `data.frame` containing the variables for the linear mixed effects model at the time of the event.

`method` the value of the `method` argument.

`termsY` the `terms` component of the `lmeObject`.

`termsT` the `terms` component of the `survObject`.

`formYx` the formula for the fixed effects part of the longitudinal model.

`formYz` the formula for the random effects part of the longitudinal model.

`formT` the formula for the survival model.

`timeVar` the value of the `timeVar` argument

`control` the value of the `control` argument.

`parameterization` the value of the `parameterization` argument.

`interFact` the value of the `interFact` argument

`derivForm` the value of the `derivForm` argument.

`lag` the value of the `lag` argument.

`call` the matched call.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

`jointModel`

---

pbc2 *Mayo Clinic Primary Biliary Cirrhosis Data*

Description

Followup of 312 randomised patients with primary biliary cirrhosis, a rare autoimmune liver disease, at Mayo Clinic.

Format

A data frame with 1945 observations on the following 20 variables.

`id` patients identifier; in total there are 312 patients.

`years` number of years between registration and the earlier of death, transplantion, or study analysis time.

`status` a factor with levels `alive`, `transplanted` and `dead`.

`drug` a factor with levels `placebo` and `D-penicil`.

`age` at registration in years.

`sex` a factor with levels `male` and `female`.

`year` number of years between enrollment and this visit date, remaining values on the line of data refer to this visit.

`ascites` a factor with levels `No` and `Yes`.

`hepatomegaly` a factor with levels `No` and `Yes`.

`spiders` a factor with levels `No` and `Yes`.

`edema` a factor with levels `No edema` (i.e., no edema and no diuretic therapy for edema), `edema no diuretics` (i.e., edema present without diuretics, or edema resolved by diuretics), and `edema despite diuretics` (i.e., edema despite diuretic therapy).

`serBilir` serum bilirubin in mg/dl.

`serChol` serum cholesterol in mg/dl.

`albumin` albumin in gm/dl.

`alkaline` alkaline phosphatase in U/liter.

`SGOT` SGOT in U/ml.

`platelets` platelets per cubic ml / 1000.

`prothrombin` prothrombin time in seconds.

`histologic` histologic stage of disease.

`status2` a numeric vector with the value 1 denoting if the patient was dead, and 0 if the patient was alive or transplanted.

Note

The data frame `pbc2.id` contains the first measurement for each patient. This data frame is used to fit the survival model.

References

Fleming, T. and Harrington, D. (1991) *Counting Processes and Survival Analysis*. Wiley, New York.

Therneau, T. and Grambsch, P. (2000) *Modeling Survival Data: Extending the Cox Model*. Springer-Verlag, New York.

Examples

```
summary(pbc2.id)
```

`piecewiseExp.ph`	*Proportional Hazards Models with Piecewise Constant Baseline Hazard Function*

Description

Based on a fitted Cox model this function fits the corresponding relative risk model with a piecewise constant baseline hazard using the Poisson regression equivalence

Usage

```
piecewiseExp.ph(coxObject, knots = NULL, length.knots = 6)
```

Arguments

coxObject	an object of class coxph.
knots	A numeric vector denoting the internal knots (cut points) defining the intervals in which the baseline hazard is assumed constant.
length.knots	a numeric value denoting the number of internal knots to use in the fit. Used when knots = NULL.

Value

an object of class glm.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Examples

```
coxFit <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)
piecewiseExp.ph(coxFit)
```

---

plot *Plot Diagnostics for Joint Models*

Description

Produces a variety of plots for fitted joint models.

Usage

```
## S3 method for class 'jointModel'
plot(x, which = 1:4, caption = c("Residuals vs Fitted",
  "Normal Q-Q", "Marginal Survival", "Marginal Cumulative Hazard",
  "Marginal log Cumulative Hazard", "Baseline Hazard",
  "Cumulative Baseline Hazard", "Subject-specific Survival",
  "Subject-specific Cumulative Hazard",
  "Subject-specific log Cumulative Hazard"), survTimes = NULL,
  main = "",
  ask = prod(par("mfcol")) < length(which) && dev.interactive(),
  ..., ids = NULL, add.smooth = getOption("add.smooth"),
  add.qqline = TRUE, add.KM = FALSE, cex.caption = 1, return = FALSE)
```

Arguments

`x`	an object inheriting from class `jointModel`.
`which`	which types of plots to produce, specify a subset of the numbers 1:10.
`caption`	captions to appear above the plots defined by argument `which`.
`survTimes`	a vector of survival times for which the survival, cumulative hazard or log cumulative hazard will be computed. Default is `seq(minT, maxT, length = 15)`, where `minT` and `maxT` are the minimum and maximum observed survival times, respectively.
`main`	a character string specifying the title in the plot.
`ask`	logical; if `TRUE`, the user is asked before each plot, see `par(ask=.)`.
`...`	other parameters to be passed through to plotting functions.
`ids`	a numeric vector specifying which subjects, the subject-specific plots will include; default is all subjects.
`add.smooth`	logical; if `TRUE` a smooth line is superimposed in the "Residuals vs Fitted" plot.
`add.qqline`	logical; if `TRUE` a qq-line is superimposed in the "Normal Q-Q" plot.
`add.KM`	logical; if `TRUE` the Kaplan-Meier estimate of the survival function is superimposed in the "Marginal Survival" plot.
`cex.caption`	magnification of captions.
`return`	logical; if `TRUE` and `which` takes in values in `c(3:5, 8:10)`, then the values used to create the plot are returned.

Note

The plots of the baseline hazard and the cumulative baseline hazard are only produced when the joint model has been fitted using `method = "Cox-PH-GH"`.

Author(s)

Dimitris Rizopoulos `<d.rizopoulos@erasmusmc.nl>`

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2010) JM: An R package for the joint modelling of longitudinal and time-to-event data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

See Also

`jointModel`

Examples

```
## Not run:
# linear mixed model fit
fitLME <- lme(log(serBilir) ~ drug * year, random = ~ 1 | id, data = pbc2)
# survival regression fit
fitSURV <- survreg(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)
# joint model fit, under the (default) Weibull model
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year")

plot(fitJOINT, 3, add.KM = TRUE, col = "red", lwd = 2)

par(mfrow = c(2, 2))
plot(fitJOINT)

## End(Not run)
```

`plot.rocJM` *Plot Method for rocJM Objects*

Description

Produces plots of ROC curves and the corresponding areas under the curve.

Usage

```
## S3 method for class 'rocJM'
plot(x, which = NULL, type = c("ROC", "AUC"),
  ndt = "all", main = NULL, caption = NULL, xlab = NULL,
  ylab = NULL, ask = NULL, legend = FALSE, lx = NULL, ly = NULL,
  lty = NULL, col = NULL, cex.caption = 0.8, cex.axis = NULL,
  cex.lab = NULL, cex.main = NULL, ...)
```

Arguments

`x`	an object inheriting from class `rocJM`.
`which`	a numeric vector specifying for which generic subjects to produce the plots. This refers to the different cases identified by the `idVar` argument in `rocJM`.
`type`	a character string specifying which plot to produce the ROC curves or the areas under the ROC curves.
`ndt`	the character string `"all"` or a numeric scalar specifying for which time windows (`dt` argument of `rocJM`) to produce the plots.
`main`	a character string specifying the title in the plot.
`caption`	a character string specifying a caption in the plot.
`xlab`	a character string specifying the x-axis label in the plot.
`ylab`	a character string specifying the y-axis label in the plot.

`ask`	logical; if `TRUE`, the user is asked before each plot, see `par()`.
`legend`	logical; if `TRUE`, a legend is included in the plot.
`lx, ly`	the `x` and `y` arguments of `legend()`.
`lty`	what types of lines to use.
`col`	which colors to use.
`cex.caption`	font size for the caption.
`cex.axis, cex.lab, cex.main`	graphical parameters; see `par` for more info.
`...`	extra graphical parameters passed to `plot()`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

See Also

`rocJM`

Examples

```
## Not run:
fitLME <- lme(sqrt(CD4) ~ obstime + obstime:(drug + AZT + prevOI + gender),
    random = ~ obstime | patient, data = aids)
fitSURV <- coxph(Surv(Time, death) ~ drug + AZT + prevOI + gender,
    data = aids.id, x = TRUE)
fit.aids <- jointModel(fitLME, fitSURV, timeVar = "obstime",
    method = "piecewise-PH-aGH")

ND <- aids[aids$patient == "7", ]
roc <- rocJM(fit.aids, dt = c(2, 4, 8), ND, idVar = "patient")
plot(roc, lwd = 2, legend = TRUE)
plot(roc, type = "AUC")

## End(Not run)
```

`plot.survfitJM` *Plot Method for survfitJM Objects*

Description

Produces plots of conditional probabilities of survival.

Usage

```
## S3 method for class 'survfitJM'
plot(x, estimator = c("both", "mean", "median"),
    which = NULL, fun = NULL, conf.int = FALSE,
   fill.area = FALSE, col.area = "grey", col.abline = "black", col.points = "black",
    add.last.time.axis.tick = FALSE, include.y = FALSE, main = NULL,
    xlab = NULL, ylab = NULL, ylab2 = NULL, lty = NULL, col = NULL,
    lwd = NULL, pch = NULL, ask = NULL, legend = FALSE, ...,
    cex.axis.z = 1, cex.lab.z = 1)
```

Arguments

- `x` — an object inheriting from class `survfitJM`.
- `estimator` — character string specifying, whether to include in the plot the mean of the conditional probabilities of survival, the median or both. The mean and median are taken as estimates of these conditional probabilities over the M replications of the Monte Carlo scheme described in `survfitJM`.
- `which` — a numeric or character vector specifying for which subjects to produce the plot. If a character vector, then is should contain a subset of the values of the `idVar` variable of the `newdata` argument of `survfitJM`.
- `fun` — a vectorized function defining a transformation of the survival curve. For example with `fun=log` the log-survival curve is drawn.
- `conf.int` — logical; if `TRUE`, then a pointwise confidence interval is included in the plot.
- `fill.area` — logical; if `TRUE` the area defined by the confidence interval of the survival function is put in color.
- `col.area` — the color of the area defined by the confidence interval of the survival function.
- `col.abline`, `col.points` — the color for the vertical line and the points when `include.y` is `TRUE`.
- `add.last.time.axis.tick` — logical; if `TRUE`, a tick is added in the x-axis for the last available time point for which a longitudinal measurement was available.
- `include.y` — logical; if `TRUE`, two plots are produced per subject, i.e., the plot of conditional probabilities of survival and a scatterplot of his longitudinal measurements.
- `main` — a character string specifying the title in the plot.
- `xlab` — a character string specifying the x-axis label in the plot.

ylab	a character string specifying the y-axis label in the plot.
ylab2	a character string specifying the y-axis label in the plotm when include.y = TRUE.
lty	what types of lines to use.
col	which colors to use.
lwd	the thickness of the lines.
pch	the type of points to use.
ask	logical; if TRUE, the user is asked before each plot, see par().
legend	logical; if TRUE, a legend is included in the plot.
cex.axis.z, cex.lab.z	the par cex argument for the axis at side 4, when include.y = TRUE.
...	extra graphical parameters passed to plot().

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D. (2010) JM: An R Package for the Joint Modelling of Longitudinal and Time-to-Event Data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

See Also

survfitJM

Examples

```
# linear mixed model fit
fitLME <- lme(sqrt(CD4) ~ obstime + obstime:drug,
    random = ~ 1 | patient, data = aids)
# cox model fit
fitCOX <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)

# joint model fit
fitJOINT <- jointModel(fitLME, fitCOX,
    timeVar = "obstime", method = "weibull-PH-aGH")

# sample of the patients who are still alive
ND <- aids[aids$patient == "141", ]
ss <- survfitJM(fitJOINT, newdata = ND, idVar = "patient", M = 50)
plot(ss)
plot(ss, include.y = TRUE, add.last.time.axis.tick = TRUE, legend = TRUE)
```

---

prederrJM *Prediction Errors for Joint Models*

Description

Using the available longitudinal information up to a starting time point, this function computes an estimate of the prediction error of survival at a horizon time point based on joint models.

Usage

```
prederrJM(object, newdata, Tstart, Thoriz, ...)

## S3 method for class 'jointModel'
prederrJM(object, newdata, Tstart, Thoriz,
    lossFun = c("absolute", "square"), interval = FALSE, idVar = "id",
    simulate = FALSE, M = 100, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`newdata`	a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required. The names of the variables in this data frame must be the same as in the data frames that were used to fit the linear mixed effects model (using `lme()`) and the survival model (using `coxph()`) that were supplied as the two first argument of `jointModel`. In addition, this data frame should contain a variable that identifies the different subjects (see also argument `idVar`).
`Tstart`	numeric scalar denoting the time point up to which longitudinal information is to be used to derive predictions.
`Thoriz`	numeric scalar denoting the time point for which a prediction of the survival status is of interest; `Thoriz` mast be later than `Tstart`.
`lossFun`	either the options `"absolute"` (default) or `"square"`, or a user-specified loss function. As the names suggest, when `lossFun = "absolute"` the loss function is $L(x) = \|x\|$, whereas when `lossFun = "square"` the loss function is $L(x) = x^2$. If a user-specified function is supplied, this should have a single argument and be vectorized.
`interval`	logical; if `TRUE` the weighted prediction error in the interval `[Tstart, Thoriz]` is calculated, while if `FALSE` the prediction error at time `Thoriz` is calculated using the longitudinal information up to time `Tstart`.
`idVar`	the name of the variable in `newdata` that identifies the different subjects.
`simulate`	logical; if `TRUE`, a Monte Carlo approach is used to estimate survival probabilities. If `FALSE`, a first order estimator is used instead. See `survfitJM` for mote details.
`M`	a numeric scalar denoting the number of Monte Carlo samples; see `survfitJM` for mote details.
`...`	additional arguments; currently none is used.

Details

Based on a fitted joint model (represented by `object`) and using the data supplied in argument `newdata`, this function computes the following estimate of the prediction:

$$PE(u|t) = \{R(t)\}^{-1} \sum_{i:T_i \geq s} I(T_i \geq u) L\{1 - Pr(T_i > u | T_i > t, \tilde{y}_i(t), x_i)\}$$

$$+\delta_i I(T_i < u) L\{0 - Pr(T_i > u | T_i > t, \tilde{y}_i(t), x_i)\}$$

$$+(1 - \delta_i) I(T_i < u)[S_i(u \mid T_i, \tilde{y}_i(t)) L\{1 - Pr(T_i > u | T_i > t, \tilde{y}_i(t), x_i)\}$$

$$+\{1 - S_i(u \mid T_i, \tilde{y}_i(t))\} L\{0 - Pr(T_i > u | T_i > t, \tilde{y}_i(t), x_i)\}],$$

where $R(t)$ denotes the number of subjects at risk at time $t =$ `Tstart`, $\tilde{y}_i(t) = \{y_i(s), 0 \leq s \leq t\}$ denotes the available longitudinal measurements up to time t, T_i denotes the observed event time for subject i, δ_i is the event indicator, s is the starting time point `Tstart` up to which the longitudinal information is used, and $u > s$ is the horizon time point `Thoriz`. Function $L(.)$ is the loss function that can be the absolute value (i.e., $L(x) = |x|$), the squared value (i.e., $L(x) = x^2$), or a user-specified function. The probabilities $Pr(T_i > u | T_i > t, \tilde{y}_i(t), x_i)$ are calculated by `survfitJM`.

When `interval` is set to `TRUE`, then function `prederrJM` computes the integrated prediction error in the interval $(u, t) =$ `(Tstart, Thoriz)` defined as

$$IPE(u|t) = \sum_{i:t \leq T_i \leq u} w_i(T_i) PE(T_i|t),$$

where

$$w_i(T_i) = \frac{\delta_i G(T_i)/G(t)}{\sum_{i:t \leq T_i \leq u} \delta_i G(T_i)/G(t)},$$

with $G(.)$ denoting the Kaplan-Meier estimator of the censoring time distribution.

Value

A list of class `prederrJM` with components:

`prederr`	a numeric scalar denoting the estimated prediction error.
`nr`	a numeric scalar denoting the number of subjects at risk at time `Tstart`.
`Tstart`	a copy of the `Tstart` argument.
`Thoriz`	a copy of the `Thoriz` argument.
`interval`	a copy of the `interval` argument.
`classObject`	the class of `object`.
`nameObject`	the name of `object`.
`lossFun`	a copy of the `lossFun` argument.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Henderson, R., Diggle, P. and Dobson, A. (2002). Identification and efficacy of longitudinal markers for survival. *Biostatistics* **3**, 33–50.

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D., Murawska, M., Andrinopoulou, E.-R., Lesaffre, E. and Takkenberg, J. (2013). Dynamic predictions with time-dependent covariates in survival analysis: A comparison between joint modeling and landmarking. *under preparation*.

See Also

survfitJM, aucJM, dynCJM, jointModel

Examples

```
## Not run:
# we construct the composite event indicator (transplantation or death)
pbc2$status2 <- as.numeric(pbc2$status != "alive")
pbc2.id$status2 <- as.numeric(pbc2.id$status != "alive")

# we fit the joint model using splines for the subject-specific
# longitudinal trajectories and a spline-approximated baseline
# risk function
lmeFit <- lme(log(serBilir) ~ ns(year, 3),
    random = list(id = pdDiag(form = ~ ns(year, 3))), data = pbc2)
survFit <- coxph(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)
jointFit <- jointModel(lmeFit, survFit, timeVar = "year",
    method = "piecewise-PH-aGH")

# prediction error at year 10 using longitudinal data up to year 5
prederrJM(jointFit, pbc2, Tstart = 5, Thoriz = 10)
prederrJM(jointFit, pbc2, Tstart = 5, Thoriz = 6.5, interval = TRUE)

## End(Not run)
```

---

predict *Predictions for Joint Models*

Description

Calculates predicted values for the longitudinal part of a joint model.

Usage

```
## S3 method for class 'jointModel'
predict(object, newdata, type = c("Marginal", "Subject"),
    interval = c("none", "confidence", "prediction"), level = 0.95, idVar = "id",
    FtTimes = NULL, M = 300, returnData = FALSE, scale = 1.6, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`newdata`	a data frame in which to look for variables with which to predict.
`type`	a character string indicating the type of predictions to compute, marginal or subject-specific. See **Details**.
`interval`	a character string indicating what type of intervals should be computed.
`level`	a numeric scalar denoting the tolerance/confidence level.
`idVar`	a character string indicating the name of the variable in `newdata` that corresponds to the subject identifier; required when `type = "Subject"`.
`FtTimes`	a list with components numeric vectors denoting the time points for which we wish to compute subject-specific predictions after the last available measurement provided in `newdata`. For each subject in `newdata` the default is a sequence of 25 equally spaced time points from the last available measurement to the maximum follow-up time of all subjects (plus a small quantity). This argument is only used when `type = "Subject"`.
`M`	numeric scalar denoting the number of Monte Carlo samples. See **Details**.
`returnData`	logical; if `TRUE` the data frame supplied in `newdata` is returned augmented with the outputs of the function.
`scale`	a numeric value setting the scaling of the covariance matrix of the empirical Bayes estimates in the Metropolis step during the Monte Carlo sampling.
`...`	additional arguments; currently none is used.

Details

When `type = "Marginal"`, this function computes predicted values for the fixed-effects part of the longitudinal submodel. In particular, let X denote the fixed-effects design matrix calculated using `newdata`. The `predict()` calculates $\hat{y} = X\hat{\beta}$, and if `interval = "confidence"`, $var(\hat{y}) = XVX^t$, with V denoting the covariance matrix of $\hat{\beta}$. Confidence intervals are constructed under the normal approximation.

When `type = "Subject"`, this functions computes subject-specific predictions for the longitudinal outcome based on the joint model. This accomplished with a Monte Carlo simulation scheme, similar to the one described in `survfitJM`. The only difference is in Step 3, where for `interval = "confidence"` $y_i^* = X_i\beta^* + Z_i b_i^*$, whereas for `interval = "prediction"` y_i^* is a random vector from a normal distribution with mean $X_i\beta^* + Z_i b_i^*$ and standard deviation σ^*. Based on this Monte Carlo simulation scheme we take as estimate of $\hat{y}_i$ the average of the `M` estimates y_i^* from each Monte Carlo sample. Confidence intervals are constructed using the percentiles of y_i^* from the Monte Carlo samples.

Value

If `se.fit = FALSE` a numeric vector of predicted values, otherwise a list with components `pred` the predicted values, `se.fit` the standard error for the fitted values, and `low` and `upp` the lower and upper limits of the confidence interval. If `returnData = TRUE`, it returns the data frame `newdata` with the previously mentioned components added.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

See Also

jointModel

Examples

```
## Not run:
# linear mixed model fit
fitLME <- lme(log(serBilir) ~ drug * year,
    random = ~ year | id, data = pbc2)
# survival regression fit
fitSURV <- survreg(Surv(years, status2) ~ drug,
    data = pbc2.id, x = TRUE)
# joint model fit, under the (default) Weibull model
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year")

DF <- with(pbc2, expand.grid(drug = levels(drug),
    year = seq(min(year), max(year), len = 100)))
Ps <- predict(fitJOINT, DF, interval = "confidence", return = TRUE)
require(lattice)
xyplot(pred + low + upp ~ year | drug, data = Ps,
    type = "l", col = c(2,1,1), lty = c(1,2,2), lwd = 2,
    ylab = "Average log serum Bilirubin")


# Subject-specific predictions
ND <- pbc2[pbc2$id == 2, ]
Ps.ss <- predict(fitJOINT, ND, type = "Subject",
  interval = "confidence", return = TRUE)
require(lattice)
xyplot(pred + low + upp ~ year | id, data = Ps.ss,
    type = "l", col = c(2,1,1), lty = c(1,2,2), lwd = 2,
    ylab = "Average log serum Bilirubin")

## End(Not run)
```

prothro *Prednisone versus Placebo in Liver Cirrhosis Patients*

Description

A randomized trial on 488 liver cirrhosis patients

Format

Two data frames with the following variable.

- `id` patients identifier; in total there are 467 patients.
- `pro` prothrobin measurements.
- `time` for data frame `prothro` the time points at which the prothrobin measurements were taken; for data frame `prothros` the time to death or censoring.
- `death` a numeric vector with 0 denoting censoring and 1 death.
- `treat` randomized treatment; a factor with levels "placebo" and "prednisone".

Source

http://www.gllamm.org/books/readme.html#14.6,

References

Andersen, P. K., Borgan, O., Gill, R. D. and Keiding, N. (1993). *Statistical Models Based on Counting Processes*. New York: Springer.

Examples

```
summary(prothros)
```

ranef *Random Effects Estimates for Joint Models*

Description

Extracts the random effects estimates from a fitted joint model.

Usage

```
## S3 method for class 'jointModel'
ranef(object, type = c("mean", "mode"), postVar = FALSE, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`type`	what type of empirical Bayes estimates to compute, the mean of the posterior distribution or the mode of the posterior distribution.
`postVar`	logical; if `TRUE` the variance of the posterior distribution is also returned. When `type == "mode"`, then this equals $\{-\partial^2 \log p(b_i \mid T_i, \delta_i, y_i) / \partial b_i^\top \partial b_i\}^{-1}$.
`...`	additional arguments; currently none is used.

Value

a numeric matrix with rows denoting the individuals and columns the random effects (e.g., intercepts, slopes, etc.). If `postVar = TRUE`, the numeric matrix has an extra attribute "postVar".

Author(s)

Dimitris Rizopoulos `<d.rizopoulos@erasmusmc.nl>`

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

See Also

`coef.jointModel`, `fixef.jointModel`

Examples

```
## Not run:
# linear mixed model fit
fitLME <- lme(log(serBilir) ~ drug * year, random = ~ 1 | id, data = pbc2)
# survival regression fit
fitSURV <- survreg(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)

# joint model fit, under the (default) Weibull model
fitJOINT <- jointModel(fitLME, fitSURV, timeVar = "year")
ranef(fitJOINT)

## End(Not run)
```

`residuals` *Residuals for Joint Models*

Description

Calculates residuals for joint models.

Usage

```
## S3 method for class 'jointModel'
residuals(object, process = c("Longitudinal", "Event"),
  type = c("Marginal", "Subject", "stand-Marginal",
  "stand-Subject", "Martingale", "nullMartingale", "CoxSnell", "AFT"),
  MI = FALSE, M = 50, time.points = NULL, return.data = FALSE,
  ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`process`	for which model (i.e., linear mixed model or survival model) to calculate residuals.
`type`	what type of residuals to calculate. See **Details**.
`MI`	logical; if `TRUE` multiple-imputation-based residuals are calculated.
`M`	integer denoting how many imputations to use for the MI residuals.
`time.points`	for fixed visit times, this should be a numeric vector with the unique times points at which longitudinal measurements are supposed to be taken; if `NULL`, then the code attempts to extract these unique time points using the design matrix for the fixed effects of the longitudinal model and the value of the `timeVar` argument of `jointModel`. For random visit times, this should be an object of class `weibull.frailty` that represents the fit of Weibull model with Gamma frailties for the visiting process. The user may also augment the object `weibull.frailty` with the following two attributes: `"prev.y"` denoting the variable name for the previous longitudinal responses, and `"tmax"` denoting the end of the study.
`return.data`	logical; if `TRUE` and `MI = TRUE` and fixed visit times are considered, then the multiply imputed data sets are returned.
`...`	additional arguments; currently none is used.

Details

When `process = "Longitudinal"`, residuals are calculated for the longitudinal outcomes. In particular, if `type = "Marginal"` these are $e_{ij} = y_{ij} - x_{ij}^T \hat{\beta}$, whereas for `type = "Subject"`, $e_{ij} = y_{ij} - x_{ij}^T \hat{\beta} - z_{ij}^T b_i$, where i denotes the subject and j the measurement, y_{ij} the longitudinal responses, x_{ij}^T and z_{ij}^T the corresponding rows of the fixed and random effects design matrices, respectively, and β and b_i denote the fixed effects and random effects components. If `type = "stand-Marginal"` or `type = "stand-Subject"`, the above defined residuals are divided by the estimated standard deviation of the corresponding error term. If MI = TRUE, multiple-imputation-based residuals are calculated for the longitudinal process; for more information regarding these residuals, check Rizopoulos et al. (2009).

When process = "Event", residuals are calculated for the survival outcome. Martingale residuals are available for all options for the survival submodel (for the different options of survival submodel, check the method argument of jointModel). when option type = "nullMartingale" is invoked, the martingale residuals are calculated with the coefficient(s) that correspond to the marker set to zero. Cox-Snell residuals (Cox and Snell, 1968) are available for the Weibull model and the additive log cumulative hazard model. AFT residuals are only available for the Weibull model.

Value

If MI = FALSE, a numeric vector of residual values. Otherwise a list with components:

- `fitted.values` the fitted values for the observed data.
- `residuals` the residuals for the observed data.
- `fitted.valsM` the fitted values for the missing data.
- `resid.valsM` the multiply imputed residuals for the missing longitudinal responses.
- `mean.resid.valsM` the average of the multiply imputed residuals for the missing longitudinal responses; returned only if fixed visit times are considered.
- `dataM` if return.data = TRUE and fixed visit times are considered, then it returns the data set with the simulated response values for the longitudinal outcome, for each of the multiple imputations.

Note

The multiple-imputation-based residuals are not available for joint models with method = "Cox-PH-GH".

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Cox, D. and Snell, E. (1968) A general definition of residuals. *Journal of the Royal Statistical Society, Series B* **30**, 248–275.

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D., Verbeke, G. and Molenberghs, G. (2010) Multiple-imputation-based residuals and diagnostic plots for joint models of longitudinal and survival outcomes. *Biometrics* **66**, 20–29.

Rizopoulos, D. (2010) JM: An R Package for the Joint Modelling of Longitudinal and Time-to-Event Data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

See Also

fitted.jointModel

Examples

```
## Not run:
# linear mixed model fit
fitLME <- lme(sqrt(CD4) ~ obstime * drug - drug,
    random = ~ 1 | patient, data = aids)
# cox model fit
fitCOX <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)

# joint model fit, under the additive log cumulative hazard model
fitJOINT <- jointModel(fitLME, fitCOX,
    timeVar = "obstime")

# residuals for the longitudinal outcome
head(cbind(
    "Marginal" = residuals(fitJOINT),
    "std-Marginal" = residuals(fitJOINT, type = "stand-Marginal"),
    "Subject" = residuals(fitJOINT, type = "Subject"),
    "std-Subject" = residuals(fitJOINT, type = "stand-Subject")
))

# residuals for the survival outcome
head(cbind(
    "Martingale" = residuals(fitJOINT, process = "Event", type = "Martingale"),
    "CoxSnell" = residuals(fitJOINT, process = "Event", type = "CoxSnell")
))

## End(Not run)
```

---

rocJM — *Predictive Accuracy Measures for Longitudinal Markers under a Joint Modelling Framework*

Description

It computes sensitivity, specificity, ROC and AUC measures for joint models.

Usage

```
rocJM(object, dt, data, idVar = "id", directionSmaller = NULL, cc = NULL, min.cc = NULL,
  max.cc = NULL, optThr = c("sens*spec", "youden"),
  diffType = c("absolute", "relative"), abs.diff = 0, rel.diff = 1,
  M = 300, burn.in = 100, scale = 1.6)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`dt`	a numeric vector indicating the lengths of the time intervals of primary interest within which we want to distinguish between subjects who died within the intervals from subjects who survived longer than that.

`data`	a data frame that contains the baseline covariates for the longitudinal and survival submodels, including a case identifier (i.e., the variable denoted by the argument `idVar`), the time points on which longitudinal measurements are assumed to be taken (this should have the same name as in the argument `timeVar` of `jointModel`).
`idVar`	the name of the variable in `data` that identifies the different generic subjects to be considered.
`directionSmaller`	logical; if `TRUE`, then smaller values for the longitudinal outcome are associated with higher risk for an event.
`cc`	a numeric vector of threshold values for the longitudinal marker; if `NULL`, this is computed using a regular sequence based on percentiles of the observed marker values.
`min.cc`	the start of the regular sequence for the threshold values for the longitudinal marker; see argument `cc` above.
`max.cc`	the end of the regular sequence for the threshold values for the longitudinal marker; see argument `cc` above.
`optThr`	character string defining how the optimal threshold is to be computed. The default chooses the cut-point for the marker that maximizes the product of sensitivity and specificity. Option `"youden"` chooses the cut-point that maximizes Youden's index that equals sensitivity + specificity - 1.
`diffType`	character string defining the type of prediction rule. See **Details**.
`abs.diff`	a numeric vector of absolute differences in the definition of composite prediction rules.
`rel.diff`	a numeric vector of relative differences in the definition of composite prediction rules.
`M`	a numeric scalar denoting the number of Monte Carlo samples.
`burn.in`	a numeric scalar denoting the iterations to discard.
`scale`	a numeric scalar that controls the acceptance rate of the Metropolis-Hastings algorithm. See **Details**.

Details

(**Note:** the following contain some math formulas, which are better viewed in the pdf version of the manual accessible at https://cran.r-project.org/package=JM.)

Assume that we have collected longitudinal measurements $Y_i(t) = \{y_i(s); 0 \leq s \leq t\}$ up to time point t for subject i. We are interested in events occurring in the medically relevant time frame $(t, t + \Delta t]$ within which the physician can take an action to improve the survival chance of the patient. Using an appropriate function of the marker history $Y_i(t)$, we can define a prediction rule to discriminate between patients of high and low risk for an event. For instance, for in HIV infected patients, we could consider values of CD4 cell count smaller than a specific threshold as predictive for death. Since we are in a longitudinal context, we have the flexibility of determining which values of the longitudinal history of the patient will contribute to the specification of the prediction rule. That is, we could define a composite prediction rule that is not based only on the last available measurement but rather on the last two or last three measurements of a patient. Furthermore, it could be of relevance to consider different threshold values for each of these measurements, for instance, we could define as success the event that the pre-last CD4 cell count is c and the last one $0.5c$, indicating that a 50% decrease is strongly indicative for death. Under this setting we define sensitivity and specificity as,

$$Pr\big\{\mathcal{S}_i(t, k, c) \mid T_i^* > t, T_i^* \in (t, t + \Delta t]\big\},$$

and

$$Pr\big\{\mathcal{F}_i(t, k, c) \mid T_i^* > t, T_i^* > t + \Delta t\big\},$$

respectively, where we term $\mathcal{S}_i(t, k, c) = \{y_i(s) \leq c_s; k \leq s \leq t\}$ as success (i.e., occurrence of the event of interest), and $\mathcal{F}_i(t, k, c) = \{y_i(s) > c_s; k \leq s \leq t\}$ as a failure, T_i^* denotes the time-to-event, and Δt the length of the medically relevant time window (specified by argument `dt`). The cut values for the marker c are specified by the `cc`, `min.cc` and `max.cc` arguments. Two types of composite prediction rules can be defined depending on the value of the `diffType` argument. Absolute prediction rules in which, between successive measurements there is an absolute difference of between the cut values, and relative prediction rules in which there is a relative difference between successive measurements of the marker. The lag values for these differences are defined by the `abs.diff` and `rel.diff` arguments. Some illustrative examples:

Ex1: keeping the defaults we define a simple rule that is only based on the last available marker measurement.

Ex2: to define a prediction rule that is based on the last two available measurements using the same cut values (e.g., if a patient had two successive measurements below a medically relevant threshold), we need to set `abs.diff = c(0, 0)`.

Ex3: to define a prediction rule that is based on the last two available measurements using a drop of 5 units between the cut values (e.g., the pre-last measurement is c and the last $c - 5$), we need to set `abs.diff = c(0, -5)`.

Ex4: to define a prediction rule that is based on the last two available measurements using a drop of 20% units between the cut values (e.g., the pre-last measurement is c and the last $0.8c$), we need to set `diffType = "relative"` and `rel.diff = c(0, 0.8)`.

The estimation of the above defined probabilities is achieved with a Monte Carlo scheme similar to the one described in `survfitJM`. The number of Monte Carlo samples is defined by the `M` argument, and the burn-in iterations for the Metropolis-Hastings algorithm using the `burn.in` argument.

More details can be found in Rizopoulos (2011).

Value

An object of class `rocJM` is a list with components,

`MCresults`	a list of length the number of distinct cases in `data`. Each component of this list is again a list with four components the estimated Sensitivity `Sens` and its standard error `seSens`, and the estimated Specificity `Spec` and its standard error `seSpec`. All these four components are matrices with rows corresponding to the different `dt` values and columns corresponding to the different `cc` values.
`AUCs`	a numeric vector of estimated areas under the ROC curves for the different values of `dt`.

`optThr`	a numeric vector with the optimal threshold values for the markers for the different `dt` under the choice made in argument `optThr`.
`times`	a list of length the number of distinct cases in `data` with components numeric vectors of the time points at which longitudinal measurements are supposed to be taken.
`dt`	a copy of the `dt` argument.
`M`	a copy of the `M` argument.
`diffType`	a copy of the `diffType` argument.
`abs.diff`	a copy of the `abs.diff` argument.
`rel.diff`	a copy of the `rel.diff` argument.
`cc`	a copy of the `cc` argument.
`min.cc`	a copy of the `min.cc` argument.
`max.cc`	a copy of the `max.cc` argument.
`success.rate`	a numeric matrix with the success rates of the Metropolis-Hastings algorithm described above.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Heagerty, P. and Zheng, Y. (2005). Survival model predictive accuracy and ROC curves. *Biometrics* **61**, 92–105.

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D. (2010) JM: An R package for the joint modelling of longitudinal and time-to-event data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

Zheng, Y. and Heagerty, P. (2007). Prospective accuracy for longitudinal markers. *Biometrics* **63**, 332–341.

See Also

`plot.rocJM`, `survfitJM`, `dynCJM`, `aucJM`, `prederrJM`, `jointModel`

Examples

```
## Not run:
fitLME <- lme(sqrt(CD4) ~ obstime * (drug + AZT + prevOI + gender),
    random = ~ obstime | patient, data = aids)
fitSURV <- coxph(Surv(Time, death) ~ drug + AZT + prevOI + gender,
    data = aids.id, x = TRUE)
fit.aids <- jointModel(fitLME, fitSURV, timeVar = "obstime",
    method = "piecewise-PH-aGH")
```

```
# the following will take some time to execute...
ND <- aids[aids$patient == "7", ]
roc <- rocJM(fit.aids, dt = c(2, 4, 8), ND, idVar = "patient")
roc

## End(Not run)
```

---

simulate *Simulate from Joint Models.*

Description

simulate longitudinal responses and event times from joint models

Usage

```
simulateJM(nsim, nsub, thetas, times, formulas, Data = NULL,
    method = c("weibull-PH", "weibull-AFT", "piecewise-PH", "spline-PH"),
    lag = 0, censoring = "uniform", max.FUtime = NULL, seed = NULL,
    return.ranef = FALSE)
## S3 method for class 'jointModel'
simulate(object, nsim, seed = NULL, times = NULL,
    Data = NULL, ...)
```

Arguments

`nsim`	number of data sets to be simulated.
`nsub`	the number of subjects in each data set.
`thetas`	a list with the parameter values. This should be of the same structure as the `coefficients` component returned by `jointModel()`.
`times`	a numeric vector denoting the time points at which longitudinal measurements are planned to be taken.
`formulas`	a list with components: `Yfixed` a formula for the fixed-effects part of the linear mixed model, `Yrandom` a formula for the random-effects part of the linear mixed model, `Tfixed` a formula for the baseline covariates part of the survival submodel, `timeVar` a character string indicating the name of the time variable in the linear mixed model.
`Data`	a data frame containing any covariates used in the formulas defined in the `formulas` argument.
`method`	a character string indicating from what type of survival submodel to simulate. There are the same options as the ones provided by `jointModel`.
`lag`	a numeric value denoting a lagged effect; the same as the `lag` argument of `jointModel`.
`censoring`	a character string or a numeric vector.

max.FUtime	a numeric value denoting the maximum follow-up time for the study. The default is max(times) + 2 * IQR(times).
seed	an object specifying if and how the random number generator should be initialized ('seeded'). It could be either NULL or an integer that will be used in a call to set.seed() before simulating the response vectors. If set, the value is saved as the "seed" attribute of the returned value.
return.ranef	logical; if TRUE, each component of the returned list has the attributed "ranef" that contains the random-effects values used in the simulation.
object	an object inheriting from class jointModel.
...	additional arguments; currently none is used.

Value

A list of length nsim of data frames that contains the simulated responses for the longitudinal process "y", the simulated event times "Time", the event indicator "Event", and the subject identification number "id". If extra covariates were assumed, these are also included.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

jointModel

Examples

```
## Not run:
prothro$t0 <- as.numeric(prothro$time == 0)
lmeFit <- lme(pro ~ treat * (time + t0), random = ~ time | id, data = prothro)
survFit <- coxph(Surv(Time, death) ~ treat, data = prothros, x = TRUE)
jointFit <- jointModel(lmeFit, survFit, timeVar = "time",
    method = "weibull-PH-aGH")

newData <- simulate(jointFit, nsim = 1, times = seq(0, 11, len = 15))
newData

## End(Not run)
```

---

summary.weibull.frailty

Summary Method for weibull.frailty Objects

Description

Summarizes the fit of a Weibull model with Gamma frailties

Usage

```
## S3 method for class 'weibull.frailty'
summary(object, sand.se = FALSE, ...)
```

Arguments

`object`	an object inheriting from class `weibull.frailty`.
`sand.se`	logical; if `TRUE`, sandwich standard errors are also produced.
`...`	additional arguments; currently none is used.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

`weibull.frailty`

Examples

```
fit <- weibull.frailty(Surv(time, status) ~ age + sex, kidney)
summary(fit)
summary(fit, TRUE)
```

---

survfitJM *Prediction in Joint Models*

Description

This function computes the conditional probability of surviving later times than the last observed time for which a longitudinal measurement was available.

Usage

```
survfitJM(object, newdata, ...)

## S3 method for class 'jointModel'
survfitJM(object, newdata, idVar = "id", simulate = TRUE, survTimes = NULL,
 last.time = NULL, M = 200, CI.levels = c(0.025, 0.975), scale = 1.6, ...)
```

Arguments

`object`	an object inheriting from class `jointModel`.
`newdata`	a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required. The names of the variables in this data frame must be the same as in the data frames that were used to fit the linear mixed effects model (using `lme()`) and the survival model (using `coxph()` or `survreg()`) that were supplied as the two first argument of `jointModel`. In addition, this data frame should contain a variable that identifies the different subjects (see also argument `idVar`).
`idVar`	the name of the variable in `newdata` that identifies the different subjects.
`simulate`	logical; if `TRUE`, a Monte Carlo approach is used to estimate survival probabilities. If `FALSE`, a first order estimator is used instead. (see **Details**)
`survTimes`	a numeric vector of times for which prediction survival probabilities are to be computed.
`last.time`	a numeric vector or character string. This specifies the known time at which each of the subjects in `newdat` was known to be alive. If `NULL`, then this is automatically taken as the last time each subject provided a longitudinal measurement. If a numeric vector, then it is assumed to contain this last time point for each subject. If a character string, then it should be a variable in the data frame `newdata`.
`M`	integer denoting how many Monte Carlo samples to use – see **Details**.
`CI.levels`	a numeric vector of length two that specifies which quantiles to use for the calculation of confidence interval for the predicted probabilities – see **Details**.
`scale`	a numeric scalar that controls the acceptance rate of the Metropolis-Hastings algorithm – see **Details**.
`...`	additional arguments; currently none is used.

Details

Based on a fitted joint model (represented by `object`), and a history of longitudinal responses $\tilde{y}_i(t) = \{y_i(s), 0 \leq s \leq t\}$ and a covariates vector x_i (stored in `newdata`), this function provides estimates of $Pr(T_i > u|T_i > t, \tilde{y}_i(t), x_i)$, i.e., the conditional probability of surviving time u given that subject i, with covariate information x_i, has survived up to time t and has provided longitudinal the measurements $\tilde{y}_i(t)$.

To estimate $Pr(T_i > u|T_i > t, \tilde{y}_i(t), x_i)$ and if `simulate = TRUE`, a Monte Carlo procedure is followed with the following steps:

Step 1: Simulate new parameter values, say θ^*, from $N(\hat{\theta}, C(\hat{\theta}))$, where $\hat{\theta}$ are the MLEs and $C(\hat{\theta})$ their large sample covariance matrix, which are extracted from `object`.

Step 2: Simulate random effects values, say b_i^*, from their posterior distribution given survival up to time t, the vector of longitudinal responses $\tilde{y}_i(t)$ and θ^*. This is achieved using a Metropolis-Hastings algorithm with independent proposals from a properly centered and scaled multivariate t distribution. The `scale` argument controls the acceptance rate for this algorithm.

Step 3 Using θ^* and b_i^*, compute $Pr(T_i > u|T_i > t, b_i^*, x_i; \theta^*)$.

Step 4: Repeat Steps 1-3 `M` times.

Based on the `M` estimates of the conditional probabilities, we compute useful summary statistics, such as their mean, median, and quantiles (to produce a confidence interval).

If `simulate = FALSE`, then survival probabilities are estimated using the formula

$$Pr(T_i > u | T_i > t, \hat{b}_i, x_i; \hat{\theta}),$$

where $\hat{\theta}$ denotes the MLEs as above, and $\hat{b}_i$ denotes the empirical Bayes estimates.

Value

A list of class `survfitJM` with components:

`summaries`	a list with elements numeric matrices with numeric summaries of the predicted probabilities for each subject.
`survTimes`	a copy of the `survTimes` argument.
`last.time`	a numeric vector with the time of the last available longitudinal measurement of each subject.
`obs.times`	a list with elements numeric vectors denoting the timings of the longitudinal measurements for each subject.
`y`	a list with elements numeric vectors denoting the longitudinal responses for each subject.
`full.results`	a list with elements numeric matrices with predicted probabilities for each subject in each replication of the Monte Carlo scheme described above.
`success.rate`	a numeric vector with the success rates of the Metropolis-Hastings algorithm described above for each subject.
`scale`	a copy of the `scale` argument.

Note

Predicted probabilities are not computed for joint models with `method = "ch-Laplace"` and `method = "Cox-PH-GH"`.

Author(s)

Dimitris Rizopoulos `<d.rizopoulos@erasmusmc.nl>`

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Rizopoulos, D. (2011). Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics* **67**, 819–829.

Rizopoulos, D. (2010) JM: An R Package for the Joint Modelling of Longitudinal and Time-to-Event Data. *Journal of Statistical Software* **35** (9), 1–33. doi:10.18637/jss.v035.i09

See Also

`jointModel`, `plot.survfitJM`

Examples

```
# linear mixed model fit
fitLME <- lme(sqrt(CD4) ~ obstime + obstime:drug,
    random = ~ 1 | patient, data = aids)
# cox model fit
fitCOX <- coxph(Surv(Time, death) ~ drug, data = aids.id, x = TRUE)

# joint model fit
fitJOINT <- jointModel(fitLME, fitCOX,
    timeVar = "obstime", method = "weibull-PH-aGH")

# sample of the patients who are still alive
ND <- aids[aids$patient == "141", ]
ss <- survfitJM(fitJOINT, newdata = ND, idVar = "patient", M = 50)
ss
```

`wald.strata` *Wald Test for Stratification Factors*

Description

It performs a Wald test to test the hypothesis of equal spline coefficients among strata in the approximation of baseline risk function.

Usage

```
wald.strata(fit)
```

Arguments

`fit`	an object of class `jointModel` with `method = "spline-PH-GH"` and with a strata specification in the survival part.

Value

an object of class `wald.strata` with components:

`alternative`	a character string naming the alternative.
`Result`	a numeric matrix with the results of the Wald test.

Note

This test is valid when the same knots have been used across strata.

Author(s)

Dimitris Rizopoulos `<d.rizopoulos@erasmusmc.nl>`

References

Rizopoulos, D. (2012) *Joint Models for Longitudinal and Time-to-Event Data: with Applications in R*. Boca Raton: Chapman and Hall/CRC.

Examples

```
## Not run:
fitLME <- lme(log(serBilir) ~ drug * year - drug, random = ~ year | id,
    data = pbc2)
fitSURV <- coxph(Surv(years, status2) ~ drug + strata(hepatomegaly),
    data = pbc2.id, x = TRUE)
fit.pbc <- jointModel(fitLME, fitSURV, timeVar = "year", method = "spline-PH-aGH")
wald.strata(fit.pbc)

## End(Not run)
```

`weibull.frailty` *Weibull Model with Gamma Frailties*

Description

Fits a Weibull model with Gamma frailties for multivariate survival data under maximum likelihood

Usage

```
weibull.frailty(formula = formula(data), data = parent.frame(),
    id = "id", subset, na.action, init, control = list())
```

Arguments

`formula`	an object of class `formula`: a symbolic description of the model to be fitted. The response must be a survival object as returned by function `Surv()`.
`data`	an optional data frame containing the variables specified in the model.
`id`	either a character string denoting a variable name in `data` or a numeric vector specifying which event times belong to the same cluster (e.g., hospital, patient, etc.).
`subset`	an optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	what to do with missing values.
`init`	a numeric vector of length $p + 3$ of initial values. The first p elements should correspond to the regression coefficients for the covariates, and the last 3 to log-scale, log-shape, and log-frailty-variance, respectively. See **Details**.
`control`	a list of control values with components:
	optimizer a character string indicating which optimizer to use; options are "optim" (default) and "nlminb".

parscale the `parscale` control argument for `optim()`, or the `scale` argument for `nlminb()`. It should be a numeric vector of length equal to the number of parameters. Default is 0.01 for all parameters.

maxit the maximum number of iterations. Default is 500.

numeriDeriv a character string indicating which type of numerical derivative to use to compute the Hessian matrix; options are "fd" denoting the forward difference approximation, and "cd" (default) denoting the central difference approximation.

eps.Hes tolerance value used in the numerical derivative method. Default is 1e-03.

Details

The fitted model is defined as follows:

$$\lambda(t_i|\omega_i) = \lambda_0(t_i)\omega_i \exp(x_i^T\beta),$$

where i denotes the subject, $\lambda(\cdot)$ denotes the hazard function, conditionally on the frailty ω_i, x_i is a vector of covariates with corresponding regression coefficients β, and $\lambda_0(\cdot)$ is the Weibull baseline hazard defined as $\lambda_0(t) = shape * scale * t^{shape-1}$. Finally, for the frailties we assume $\omega_i \sim Gamma(\eta, \eta)$, with η^{-1} denoting the unknown variance of ω_i's.

Value

an object of class `weibull.frailty` with components:

`coefficients`	a list with the estimated coefficients values. The components of this list are: `betas`, `scale`, `shape`, and `var.frailty`, and correspond to the coefficients with the same name.
`hessian`	the hessian matrix at convergence. For the shape, scale, and var-frailty parameters the Hessian is computed on the log scale.
`logLik`	the log-likelihood value.
`control`	a copy of the `control` argument.
`y`	an object of class `Surv` containing the observed event times and the censoring indicator.
`x`	the design matrix of the model.
`id`	a numeric vector specifying which event times belong to the same cluster.
`nam.id`	the value of argument `id`, if that was a character string.
`terms`	the term component of the fitted model.
`data`	a copy of `data` or the created `model.frame`.
`call`	the matched call.

Note

`weibull.frailty()` currently supports only right-censored data.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Examples

```
weibull.frailty(Surv(time, status) ~ age + sex, kidney)
```

---

xtable *xtable Method from Joint Models.*

Description

produces a LaTeX table with the results of a joint model using package xtable.

Usage

```
## S3 method for class 'jointModel'
xtable(x, caption = NULL, label = NULL, align = NULL,
    digits = NULL, display = NULL, which = c("all", "Longitudinal", "Event"),
    varNames.Long = NULL, varNames.Event = NULL, p.values = TRUE,
    digits.pval = 4, ...)
```

Arguments

x	an object inheriting from class jointModel.
caption	the caption argument of xtable().
label	the label argument of xtable().
align	the align argument of xtable().
digits	the digits argument of xtable().
display	the display argument of xtable().
which	a character string indicating which results to include in the LaTeX table. Options are all results, the results of longitudinal submodel or the results of the survival submodel.
varNames.Long	a character vector of the variable names for the longitudinal submodel.
varNames.Event	a character vector of the variable names for the survival submodel.
p.values	logical; should p-values be included in the table.
digits.pval	a numeric scalare denoting the number of significance digits in the p-value.
...	additional arguments; currently none is used.

Value

A LaTeX code chunk with the results of the joint modeling analysis.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

jointModel

Examples

```
## Not run:
require(xtable)
prothro$t0 <- as.numeric(prothro$time == 0)
lmeFit <- lme(pro ~ treat * (time + t0), random = ~ time | id, data = prothro)
survFit <- coxph(Surv(Time, death) ~ treat, data = prothros, x = TRUE)
jointFit <- jointModel(lmeFit, survFit, timeVar = "time",
    method = "weibull-PH-aGH")

xtable(jointFit, math.style.negative = TRUE)

## End(Not run)
```

Index