Package 'joineRmeta'

March 9, 2018

Type Package

Title Joint Modelling for Meta-Analytic (Multi-Study) Data

Version 0.1.1

Encoding UTF-8

Description Fits joint models of the type proposed by Henderson and colleagues (2000) <doi:10.1093/biostatistics/1.4.465>, but extends to the multi-study, meta-analytic case. Functions for meta-analysis of a single longitudinal and a single time-to-event outcome from multiple studies using joint models. Options to produce plots for multi study joint data, to pool joint model fits from 'JM' and 'joineR' packages in a two stage meta-analysis, and to model multi-study joint data in a one stage meta-analysis.

License GPL-3 | file LICENSE

URL https://github.com/mesudell/joineRmeta/

BugReports https://github.com/mesudell/joineRmeta/issues

LazyData true

Depends R (>= 3.3.0), lme4, survival, JM

Imports ggplot2, grid, gridExtra, gtools, joineR, MASS, meta, msm, nlme, statmod, stats, Matrix, utils

RoxygenNote 6.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Maria Sudell [cre, aut] (<https://orcid.org/0000-0002-7919-4981>), Ruwanthi Kolamunnage-Dona [aut] (<https://orcid.org/0000-0003-3886-6208>), Catrin Tudur Smith [aut]

Maintainer Maria Sudell <mesudell@liverpool.ac.uk>

Repository CRAN

Date/Publication 2018-03-09 12:41:26 UTC

R topics documented:

`confint.jointmeta1SE` *Extract confidence intervals*

Description

`confint` returns the bootstrapped confidence intervals from a `jointmeta1SE` object, which holds the results of bootstrapping the fit from a `jointmeta1` function fit.

Usage

```
## S3 method for class 'jointmeta1SE'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

`object`	A jointmeta1SE object - the result of running the bootstrapping function `jointmetaSE` on a jointmeta1 object
`parm`	A vector indicating what parameters to return confidence intervals for. This should either be a vector of character strings, where any parameters matching the supplied parameters have their estimates and confidence intervals returned, or a numeric or integer vector of values indicating what rows of the results of the bootstapping procedure to print out
`level`	A numerical value greater than 0 and less than 1 that indicates the requireed level of the confidence interval. The default is `level = 0.95` giving 95% confidence intervals.
`...`	additional arguments; currently none are used.

Value

Returns the name of variables, the part of the joint model they relate to (sub-model, variance parameter...), their estimate and their 95% confidence interval

See Also

`jointmeta1`

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 3
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=3)

 ## Not run:
     #calculate the SE
     onestagefitSE <- jointmetaSE(fitted = onestagefit, n.boot = 200)

     #extract confidence intervals
     confint(onestagefitSE)
```

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

`fixef.jointmeta1` *Extract fixed effects*

Description

Function to extract the estimated fixed effects from a jointmeta1 model fit

Usage

```
## S3 method for class 'jointmeta1'
fixef(object, type = c("Longitudinal", "Survival",
  "Latent"), ...)
```

Arguments

`object`	A joint model fit from the function jointmeta1
`type`	Type of fixed effects to extract. To extract fixed effects from longitudinal sub-model set `type = 'Longitudinal'`, from the time-to-event sub-model set `type = 'Survival'`, or extract the latent association parameters set to `type = 'Latent'`.
`...`	additional arguments; currently none are used.

Value

The function returns a vector of the fixed effects from the specified part of the supplied jointmeta1 model fit.

See Also

`jointmeta1`

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 2
```

```
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it = 2)

 #extract longitudinal fixed effects
 fixef(onestagefit, type = 'Longitudinal')
```

`formula.jointmeta1` *Extract formulae from joint model fit*

Description

Extract the formula of various parts of the joint model fit

Usage

```
## S3 method for class 'jointmeta1'
formula(x, type = c("Longitudinal", "Survival",
  "Rand_ind", "Rand_stud"), ...)
```

Arguments

`x`	A `jointmeta1` object, the result of applying the `jointmeta1` function to joint longitudinal and survival data.
`type`	A character string indicating what part of the joint model the formula should be returned for. Specifying `'Longitudinal'` will result in the function returning the formula for the fixed effect portion of the longitudinal sub-model, `'Survival'` will result in the formula for the fixed effect portion of the survival sub-model being returned. To return the formula for the individual level random effect, specify `type = 'Rand_ind'`, or to return the formula for the study level random effects (if included in the joint model), specify `type = 'Rand_stud'`.
`...`	additional arguments; currently none are used.

Value

This function returns a formula for the specified portion of the joint model fitted in the supplied `jointmeta1` object.

See Also

`jointmeta1`, `jointmeta1.object`

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 #return the formula for the longitudinal fixed effects
 formula(onestagefit, type = 'Longitudinal')

 #return the formula for the time-to-event fixed effects
 formula(onestagefit, type = 'Survival')

 #return the formula for the individual level random effects
 formula(onestagefit, type = 'Rand_ind')

 #return the formula for the study level random effects
 formula(onestagefit, type = 'Rand_stud')
```

`JMfits` *Study specific joint model fits using the JM package*

Description

A dataset containing a list of the model fits for joint models fitted to the data for each study in the `simdat` dataset using the JM package. Further details of model fits supplied below.

Usage

```
JMfits
```

Format

A list of 5 `jointModel` objects, the result of fitting a joint model using the JM package to the data from each study in the `simDat` dataset in turn.

Details

These are the results of fitting a joint model using the `JM` package separately to the data from each study present in the `simdat` dataset. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3). The joint models fitted to each study's data had the same format. The longitudinal sub-model contained a fixed intercept, time and treatment assignment term, and random intercept and slope. The survival sub-model contained a fixed treatment assignment term. A current value association structure was used to link the sub-models. More formally, the longitudinal sub-model had the following format:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + \epsilon_{kij}$$

Where Y represents the continuous longitudinal outcome, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(\beta_{21}treat + \alpha(\beta_{10} + \beta_{11}time + \beta_{12}treat + b^{(2)}_{0ki} + b^{(2)}_{1ki}time))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the baseline hazard, which was modelled using splines. The fixed effect coefficient is represented by β_{21}. The association parameter quantifying the link between the sub-models is represented by α. Again $treat$ represents the binary factor treatment assignment variable, and $b^{(2)}_{0ki}$ and $b^{(2)}_{1ki}$ are the zero mean random effects from the longitudinal sub-model.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

See Also

`jointModel`, `jointModelObject`

JMfits2 *Study specific joint model fits using the JM package*

Description

A dataset containing a list of the model fits for joint models fitted to the data for each study in the `simdat` dataset using the JM package. Further details of model fits supplied below.

Usage

```
JMfits2
```

Format

A list of 5 `jointModel` objects, the result of fitting a joint model using the JM package to the data from each study in the `simDat` dataset in turn.

Details

These are the results of fitting a joint model using the `JM` package separately to the data from each study present in the `simdat` dataset. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3). The joint models fitted to each study's data had the same format. The longitudinal sub-model contained a fixed intercept, time and treatment assignment term, as well as a fixed time by treatment assignment interaction term, and random intercept and slope. The survival sub-model contained a fixed treatment assignment term. The sub-models were linked by inserting both the current value of the longitudinal trajectory and its first derivative with respect to time into the survival sub-model. More formally, the longitudinal sub-model had the following format:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + \beta_{13}time * treat + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + \epsilon_{kij}$$

Where Y represents the continuous longitudinal outcome, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(\beta_{21}treat + \alpha_1(\beta_{10} + \beta_{11}time + \beta_{12}treat + + \beta_{13}time * treat b^{(2)}_{0ki} + b^{(2)}_{1ki}time) + \alpha_2(\beta_{11} + \beta_{13}treat + b^{(2)}_{1ki}))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the baseline hazard, which was modelled using splines. The fixed effect coefficient is represented by β_{21}. Association parameters representing the link between the sub-models are represented by α terms, where α_1 represents the effect of the current value of the longitudinal outcome on the risk of an event, whilst α_2 represents the effect of the slope, or rate of change of

the longitudinal trajectory (the value of the first derivative of the longitudinal trajectory with respect to time) on the risk of an event. Again $treat$ represents the binary facator treatment assignment variable, and $b_{0ki}^{(2)}$ and $b_{1ki}^{(2)}$ are the zero mean random effects from the longitudinal sub-model.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

See Also

jointModel, jointModelObject

---

joineRfits *Study specific joint model fits using the joineR package*

Description

A dataset containing a list of the model fits for joint models fitted to the data for each study in the simdat dataset using the joineR package. Further details of model fits supplied below.

Usage

```
joineRfits
```

Format

A list of 10 objects:

joineRfit1 an object of class joint, the result of using the joint function to fit a joint model to the data from the first study in the simdat dataset.

joineRfit1SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit1.

joineRfit2 an object of class joint, the result of using the joint function to fit a joint model to the data from the second study in the simdat dataset.

joineRfit2SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit2.

joineRfit3 an object of class joint, the result of using the joint function to fit a joint model to the data from the third study in the simdat dataset.

joineRfit3SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit3.

joineRfit4 an object of class joint, the result of using the joint function to fit a joint model to the data from the fourth study in the simdat dataset.

joineRfit4SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit4.

`joineRfit5` an object of class `joint`, the result of using the `joint` function to fit a joint model to the data from the fifth study in the `simdat` dataset.

`joineRfit5SE` an object of class `data.frame`, the result of applying the function `jointSE` to the joint model fit `joineRfit5`.

Details

These are the results of fitting a joint model using the `joineR` package separately to the data from each study present in the `simdat` dataset. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3). The joint models fitted to each study's data had the same format. The longitudinal sub-model contained a fixed intercept, time and treatment assignment term, and random intercept and slope. The survival sub-model contained a fixed treatment assignment term. A proportional association structure was used to link the sub-models. More formally, the longitudinal sub-model had the following format:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + \epsilon_{kij}$$

Where Y represents the continuous longitudinal outcome, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(\beta_{21}treat + \alpha(b^{(2)}_{0ki} + b^{(2)}_{1ki}time))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the baseline hazard. The fixed effect coefficient is represented by β_{21}. The association parameter quantifying the link between the sub-models is represented by α. Again $treat$ represents the binary factor treatment assignment variable, and $b^{(2)}_{0ki}$ and $b^{(2)}_{1ki}$ are the zero mean random effects shared from the longitudinal sub-model.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

See Also

`jointdata`, `joint`, `jointSE`

joineRfits2 *Study specific joint model fits using the joineR package*

Description

A dataset containing a list of the model fits for joint models fitted to the data for each study in the simdat dataset using the joineR package. Further details of model fits supplied below.

Usage

```
joineRfits2
```

Format

A list of 10 objects:

joineRfit6 an object of class joint, the result of using the joint function to fit a joint model to the data from the first study in the simdat dataset.

joineRfit6SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit6.

joineRfit7 an object of class joint, the result of using the joint function to fit a joint model to the data from the second study in the simdat dataset.

joineRfit7SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit7.

joineRfit8 an object of class joint, the result of using the joint function to fit a joint model to the data from the third study in the simdat dataset.

joineRfit8SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit8.

joineRfit9 an object of class joint, the result of using the joint function to fit a joint model to the data from the fourth study in the simdat dataset.

joineRfit9SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit9.

joineRfit10 an object of class joint, the result of using the joint function to fit a joint model to the data from the fifth study in the simdat dataset.

joineRfit10SE an object of class data.frame, the result of applying the function jointSE to the joint model fit joineRfit10.

Details

These are the results of fitting a joint model using the joineR package separately to the data from each study present in the simdat dataset. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3). The joint models fitted to each study's data had the same format. The longitudinal submodel contained a fixed intercept, time and treatment assignment term, and random intercept. The

survival sub-model contained a fixed treatment assignment term. A proportional association structure was used to link the sub-models. More formally, the longitudinal sub-model had the following format:

$$Y_{kij} = \beta_{10} + \beta_{11} time + \beta_{12} treat + b_{0ki}^{(2)} + \epsilon_{kij}$$

Where Y represents the continuous longitudinal outcome, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t) exp(\beta_{21} treat + \alpha(b_{0ki}^{(2)}))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the baseline hazard. The fixed effect coefficient is represented by β_{21}. The association parameter quantifying the link between the sub-models is represented by α. Again $treat$ represents the binary factor treatment assignment variable, and $b_{0ki}^{(2)}$ are the zero mean random effects shared from the longitudinal sub-model.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

See Also

jointdata, joint, jointSE

`jointmeta1` *One stage joint meta function*

Description

Function to allow a one stage joint model (data from all studies analysed in one model) to be fitted to data from multiple studies. The function allows one longitudinal and one time-to-event outcome, and can accommodate baseline hazard stratified or not stratified by study, as well as random effects at the individual level and the study level. Currently only zero mean random effects only proportional association supported - see Wulfsohn and Tsiatis 1997

Usage

```
jointmeta1(data, long.formula, long.rand.ind, long.rand.stud = NULL,
  sharingstrct = c("randprop", "randsep", "value", "slope", "valandslope"),
  surv.formula, gpt, lgpt, max.it, tol, study.name, strat = F, longsep = F,
  survsep = F, bootrun = F, print.detail = F)
```

Arguments

data	an object of class jointdata containing the variables named in the model formulae
long.formula	a formula object with the response varaible, and the covariates to include in the longitudinal sub-model
long.rand.ind	a vector of character strings to indicate what variables to assign individual level random effects to. A maximum of three individual level random effects can be assigned. To assign a random intercept include 'int' in the vector. To not include an individual level random intercept include 'noint' in the vector. For example to fit a model with individual level random intercept and random slope set `long.rand.ind = c('int', 'time')`, where `'time'` is the longitudinal time variable in the `data`.
long.rand.stud	a vector of character strings to indicate what variables to assign study level random effects to. If no study level random effects then this either not specified in function call or set to `NULL`. If a study level random intercept is required, include the name of the study membership variable for example `long.rand.stud = 'study'`.
sharingstrct	currently must be set to `'randprop'`. This gives a model that shares the zero mean random effects (at both individual and study level if specified) between the sub-models. Separate association parameters are calculated for the linear combination of random effects at each level. There are plans to expand to more sharing structures in the future.
surv.formula	a formula object with the survival time, censoring indicator and the covariates to include in the survival sub-model. The response must be a survival object as returned by the `Surv` function.
gpt	the number of quadrature points across which the integration with respect to the random effects will be performed. If random effects are specified at both the individual and the study level, the same number of quadrature points is used in both cases. Defaults to `gpt = 5`.
lgpt	the number of quadrature points which the log-likelihood is evaluated over following a model fit. This defaults to `lgpt = 7`.
max.it	the maximum number of iterations of the EM algorithm that the function will perform. Defaults to `max.it = 350` although more iterations could be required for large complex datasets.
tol	the tolerance level used to determine convergence in the EM algorithm. Defaults to `tol = 0.001`.
study.name	a character string denoting the name of the variable in the baseline dataset in `data` holding study membership, for example `study.name = 'study'`.
strat	logical value: if `TRUE` then the survival sub-model is calculated with a baseline stratified by study. Otherwise baseline is unstratified
longsep	logical value: if `TRUE` then parameter estimates, model fit and the log-likelihood from a separate linear mixed model analysis of the longitudinal data are returned (see the `lmer` function). The separate longitudinal model fit has the same specification as the longitudinal sub-model of the joint model.

survsep	logical value: if TRUE then parameter estimates, model fit and log-likelihood from a separate analysis of the survival data using the Cox Proportional Hazards model are returned (see coxph function for more details). This survival fit has the same specification (apart from the association structure) as the survival sub-model in the joint model.
bootrun	logical value: if TRUE then the log-likelihood for the model is not calculated. This option is available so that when bootstrapping to obtain standard errors, as the log-likelihood is not needed, it is not calculated, thus speeding up the bootstrapping process.
print.detail	logical value: if TRUE then details of the parameter estimates at each iteration of the EM algorithm are printed to the console.

Value

An object of class jointmeta1 See jointmeta1.object

Details

The jointmeta1 function fits a one stage joint model to survival and longitudinal data from multiple studies. This model is an extension of the model proposed by Wulfsohn and Tsiatis (1997). The model must contain at least one individual level random effect (specified using the long.rand.ind argument). The model can also contain study level random effects (specified using the long.rand.stud argument), which can differ from the individual level random effects. The maximum number of random effects that can be specified at each level is three. Note that the fitting and bootstrapping time increases as the number of included random effects increases. The model can also include a baseline hazard stratified by study, or can utilise a common baseline across the studies in the dataset. Interaction terms can be specified in either the longitudinal or the survival sub-model.

The longitudinal sub-model is a mixed effects model. If both individual level and study level random effects are included in the function call, then the sub-model has the following format:

$$Y_{kij} = X_{1kij}\beta_1 + Z^{(2)}_{1kij}b^{(2)}_{ki} + Z^{(3)}_{1kij}b^{(3)}_{k} + \epsilon_{kij}$$

Otherwise, if only individual level random effects are included in the function call, then the longitudinal sub-model has the following format:

$$Y_{kij} = X_{1kij}\beta_1 + Z^{(2)}_{1kij}b^{(2)}_{ki} + \epsilon_{kij}$$

In the above equation, Y represents the longitudinal outcome and X_1 represents the design matrix for the longitudinal fixed effects. The subscript 1 is used to distinguish between items from the longitudinal sub-model and items from the survival sub-model (which contain a subscript 2). The design matrices for random effects are represented using Z, fixed effect coefficients are represented by β, random effects by b and the measurement error by ϵ. Study membership is represented by the subscript k whilst individuals are identified by i and time points at which they are measured by j. The longitudinal outcome is assumed continuous.

Currently this function only supports one linking structure between the sub-models, namely a random effects only proportional sharing structure. In this structure, the zero mean random effects

from the longitudinal sub-model are inserted into the survival sub-model, with a common association parameter for each level of random effects. Therefore the survival sub-model (for a case without baseline stratified by study) takes the following format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(X_{2ki}\beta_2 + \alpha^{(2)}(Z_{1ki}^{(2)}b_{ki}^{(2)}) + \alpha^{(3)}(Z_{1ki}^{(3)}b_{k}^{(3)}))$$

Otherwise, if only individual level random effects are included in the function call, this reduces to:

$$\lambda_{ki}(t) = \lambda_0(t)exp(X_{2ki}\beta_2 + \alpha^{(2)}(Z_{1ki}^{(2)}b_{ki}^{(2)})$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the baseline hazard. If a stratified baseline hazard were specified this would be replaced by $\lambda_{0k}(t)$. The design matrix for the fixed effects in the survival sub-model is represented by X_{2ki}, with fixed effect coefficients represented by β_2. Association parameters quantifying the link between the sub-models are represented by α terms.

The model is fitted using an EM algorithm, starting values for which are extracted from initial separate longitudinal and survival fits. Pseudo adaptive Gauss - Hermite quadrature is used to evaluate functions of the random effects in the EM algorithm, see Rizopoulos 2012.

References

Wulfsohn, M.S. and A.A. Tsiatis, A Joint Model for Survival and Longitudinal Data Measured with Error. 1997, International Biometric Society. p. 330

Rizopoulos, D. (2012) Fast fitting of joint models for longitudinal and event time data using a pseudo-adaptive Gaussian quadrature rule. Computational Statistics & Data Analysis 56 (3) p.491-501

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)
```

`jointmeta1.object` *Fitted* `jointmeta1` *object*

Description

An object returned by the `jointmeta1` function, inheriting from class `jointmeta1` and representing a fitted joint model for a single longitudinal and a single time-to-event outcome for data from multiple studies. Objects of this class have methods for the generic functions `confint`, `fixef`, `formula` and `ranef`. Additionally `rancov` allows the user to extract the estimated covariance matrices for the zero mean random effects.

Usage

```
jointmeta1.object
```

Format

An object of class `NULL` of length 0.

Value

A list with the following components.

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

- `fixed` the list of fixed effects for sub-models contained in the joint model. The components of this list are:
  - `longitudinal` a data frame containing the estimated fixed effect coefficients from the longitudinal sub-model
  - `survival` a numeric vector containing the estimated fixed effect coefficients from the longitudinal sub-model
- `random` the list of estimates random effects estimated by the joint model. The components of this list are:
  - `random_ind` a list of matrices of length equal to the number of studies in the dataset. Each matrix has number of columns equal to the number of individual level random effects, and number of rows equal to the number of individuals in the study. As `jointmeta1` insists on the presence of random effects at the individual level, this item will always be present.
  - `random_stud` a matrix with number of columns equal to the number of study level random effects, number of rows equal to the number of studies in the dataset. This item is only present if study level random effects are specified in the model fit.
- `latent` a numeric containing the estimates of the latent association parameters for each level of the random effects. The association parameter for the individual level random effects is labelled `gamma_ind_0`, and for the study level random effects is labelled `gamma_stud_0`.

`sigma.e` a numeric holding the estimate of the variance of the measurement error variance

rand_cov a list containing the covariance matrices for the random effects included in the model. The covariance matrix for the individual level random effects is labelled D. If study level random effects are included in the model, the covariance matrix for the study level random effects is also included in the list, labelled A.

hazard if strat = FALSE in the function call for jointmeta1 then this is a numeric vector containing the common baseline across all studies. If strat = TRuE then this is a list of numeric vectors, each of which is the baseline hazard for each study in the dataset.

loglik a list containing the overall likelihood for the joint model (labelled jointlhood), and the portions of the likelihood attributable to each sub-model (jointy for the longitudinal component and jointn for the survival component).

numIter the number of EM algorithm iterations completed during the fitting of the joint model

convergence a logical value, takes a value of TRUE if convergence was achieved within the set maximum number of iterations, FALSE otherwise.

sharingstrct a character string denoting the specified sharing structure used in the joint model. Currently only 'randprop' is supported, denoting zero mean random effects sharing structure (see Wulfsohn and Tsiatis (1997)).

sepests A list containing estimates from the separate longitudinal and survival analyses. If separate results are not requested, the elements of the list are set to 'No separate results requested'. However, if separate analyses are requested in the jointmeta1 function call, the components of this list are:

longests a list containing estimates from the initial longitudinal fit. The components of this list are:

beta1 a data frame of the estimates of the fixed effects from the longitudinal sub-model

sigma.e the value of the variance of the measurement error from the longitudinal sub-model

D the estimate of the covariance matrix for the individual level random effects. Individual level random effects are always included in the joint model

A the estimate of the covariance matrix for the study level random effects. This is only present if study level random effects are specified in the jointmeta1 function call.

log.like.long the numeric value of the log likelihood for the initial longitudinal model.

randstart.ind a list of the conditional modes of the individual level random effects in each study given the data and the estimates of the separate longitudinal model parameters

randstart.ind.cov a list of the conditional covariance matrices for each individual for the individual level random effects given the data and the estimates of the separate longitudinal model parameters

randstart.stud a data frame containing the conditional modes of the study level random effects given the data and the estimates of the separate longitudinal model parameters. This is only present if study level random effects were specified in the jointmeta1 function call.

randstart.stud.cov a list of conditional covariance matrices for each study for the study level random effects given the data and the estimates of the separate longitudinal model parameters. This is only present if study level random effects were specified in the jointmeta1 function call.

`modelfit` the initial longitudinal model fit. The model has the same specification as the longitudinal sub-model for the joint model, fitted using the `lmer` function from package `lme4`

`survests` a list containing estimates from the initial survival fit. The components of this list are:

`beta2` vector of the estimates of the fixed effects included in the survival model.

`haz` if `strat = TRUE` then this is a list of numeric vectors of length equal to the number of studies in the dataset, giving the study specific baseline hazard. If `strat = FALSE` then the baseline is not stratified by study, and this is one numeric vector giving the common baseline across studies.

`rs` a counter to indicate the last how many unique event times had occured by the individual's survival time - this is for use during further calculation in the joint model EM algorithm. If a stratified baseline this is a list of numerical vectors, whereas if the baseline is not stratified this is a single numeric vector.

`sf` the unique event times observed in the dataset. If a stratified baseline this is a list of numerical vectors, whereas if the baseline is not stratified this is a single numeric vector.

`nev` a counter of the number of events that occur at each event time.If a stratified baseline this is a list of numerical vectors, whereas if the baseline is not stratified this is a single numeric vector.

`log.like.surv` a numeric containing two values, the log-likelihood with the initial values and the log-likelihood with the final values, see `coxph.object`

`modelfit` the initial survival model fit. The model has the same specification as the survival sub-model for the joint model, fitted using the `coxph` function from package `survival`

`sep.loglik` a list containing the log-likelihoods estimated from the separate analyses. It contains three elements, namely `seplhood` - the sum of the log-likelihoods from the separate longitudinal and the separate survival analyses, `sepy` - the log-likelihood from the separate longitudinal analysis, `sepn` - the log-likelihood from the separate survival analysis.

`data` the `jointdata` object containing the data the joint model was fitted to

`call` the function call supplied to the `jointmeta1` function.

`numstudies` an integer containing the number of studies present in the data used to fit the joint model

`n.bystudy` a numeric vector containing the number of individuals present in each study in the data used to fit the joint model. This will be less than the number of individuals in the supplied dataset, if missing data is present in variables included in the model.

`missingids` the ids of any individuals excluded from the analysis due to missing data

`nobs` a table containing the number of longitudinal measurements supplied by each study in the data used to fit the model. This will be less than the number of longitudinal measurements in the dataset supplied to the function call, if missing data is present in variables included in the model

Author(s)

Maria Sudell (<mesudell@liverpool.ac.uk>)

See Also

jointmeta1.

`jointmeta1SE.object` *A* `jointmeta1SE` *object*

Description

An object returned by the `jointmetaSE` function, inheriting from class `jointmeta1SE` representing the results of bootstrapping a fit from the `jointmeta1` function. Objects of this class have methods for the `print.jointmeta1SE` and `vcov.jointmeta1SE` functions.

Usage

```
jointmeta1SE.object
```

Format

An object of class `NULL` of length 0.

Value

A list with the following components.

`results` a data frame containing the estimates, standard errors and 95% confidence intervals for the parameters from the model and any overall effects requested.

`covmat` the covariance matrix for the model parameters

`bootstraps` a data frame containing the results of each bootstrap

Author(s)

Maria Sudell (<mesudell@liverpool.ac.uk>)

See Also

jointmeta1, jointmetaSE.

jointmeta2 *Function to pool joint model fits in two stage MA*

Description

This function takes joint model fits from either joint or jointModel and pools the information from the fits in the second stage of a two stage meta-analysis (MA).

Usage

```
jointmeta2(fits, SE = NULL, longpar = NULL, survpar = NULL,
  assoc = TRUE, studynames = NULL)
```

Arguments

fits	a list of joint modelling fits. These fits should all be of the same type (i.e. all fitted using joint or all fitted using jointModel), with the same model specification.
SE	a list of the results from jointSE. Only to be supplied if the model fits supplied in fits are all fitted using the joineR package.
longpar	a vector of character strings of parameters from the longitudinal sub-model for which meta-analyses should be performed
survpar	a vector of character strings of parameters from the survival sub-model for which meta-analyses should be performed
assoc	a logical indicating whether a meta-analysis should be performed for the association parameter(s)
studynames	a vector of character strings containing the names for the studies present in the dataset that the joint models were fitted to. These character strings if supplied are used to label the meta-analyses performed by the function

Details

This function is designed to take study specific joint model fits, and pool them in a meta-analysis. This is the second stage in a two stage IPD meta-analysis. The function can handle fits using the joint function or the jointModel, but not a mixture of the two. This is due to the differing types of association structures available in each package (for more details see joineR, and JM).

The fixed effects in the survival sub-models estimated in both packages have slightly different interpretations. The joint model fits modelled using the joineR package link the sub-models using shared zero mean random effects (see Henderson et al (2000)). However the joint model fits modelled using the JM package link the sub-models using sharing structures that involve both the fixed and random effects. If a parameter specified in survpar is also present in the fixed effects of the longitudinal sub-model, a direct effect of the parameter on the risk of an event can be extracted from the survival sub-model, as well as the overall effect resulting from the sum of fixed effect in the survival sub-model, and the presence of the parameter in the longitudinal sub-model, present in the sharing structure of the joint model. As such, if a parameter specified in survpar is also present as

a fixed effect in the longitudinal sub-model, and the fixed and random effects make up the sharing structure linking the sub-models, the overall parameter effect is found by $\beta_2 + (\alpha * \beta_1)$, where α is the association parameter, β_2 is the coefficient for the parameter in question from the survival sub-model, and β_1 is the coefficient for the parameter in question from the longitudinal sub-model. For more information about overall effects versus direct effects see Ibrahim et al (2010), Rizopoulos (2012) and Gould et al (2015). Because both a direct and an overall effect of the survival parameters can be extracted from the model, both are present in the results if the joint models supplied in the fits are fitted using the `JM` package.

Value

This function returns a list of results for the two stage MA. These results are split by the type of parameter being pooled. If the names of longitudinal parameters were supplied to `longpar` then an element named `longMA` will be present in the results. If the names of survival parameters were supplied to `survpar` then if the supplied joint model fits were fitted using the `joint` function from the `joineR` package, an element named `survMA.direct` will be present in the results. If the supplied joint model fits were fitted using the `jointModel` function from the `JM` package, two elements named `survMA.direct` and `survMA.overall` will be present. If `assoc = TRUE` then an element labelled `assocMA` will be present in the results.

Each element of each of these components of the results (`longMA`, `survMA.direct`, `assocMA`...) is of class `metagen`, and is the result of using the `metagen` function on the results of joint models fitted to multiple studies in the dataset. This method pools the supplied information in fixed and random MA using inverse variance weighting. Forest plots can be produced for these results simply by applying the function `forest` to the objects of class `metagen` / `meta` supplied in the results.

References

Ibrahim et al (2010) Basic Concepts and Methods for Joint Models of Longitudinal and Survival Data. JOURNAL OF CLINICAL ONCOLOGY 28 (10): 2796-2801

Rizopoulos (2012) Joint Models for Longitudinal and Time-to-Event Data With Applications in R. Chapman and Hall/CRC Biostatistics Series

Henderson et al (2000) Joint modelling of longitudinal measurements and event time data. Biostatistics, 1,4, pp. 465–480

Gould et al (2015) Joint modeling of survival and longitudinal non-survival data: current methods and issues. Report of the DIA Bayesian joint modeling working group. Statistics in Medicine 34(14): 2181–2195. doi:10.1002/sim.6141.

See Also

`joint`, `jointModel`, `jointSE`, `metagen`

Examples

```
## Not run:
library(joineR)

#change data to jointdata format
jointdat<-tojointdata(longitudinal = simdat$longitudinal,
                      survival = simdat$survival, id = 'id',
```

```
                      longoutcome = 'Y',
                      timevarying = c('time','ltime'),
                      survtime = 'survtime', cens = 'cens',
                      time = 'time')

#ensure variables are correctly formatted
jointdat$baseline$study <- as.factor(jointdat$baseline$study)
jointdat$baseline$treat <- as.factor(jointdat$baseline$treat)

#subset the data by study
for(i in 1:length(unique(jointdat$baseline$study))){
     idstemp<-jointdat$baseline$id[which(jointdat$baseline$study %in%
                                    unique(jointdat$baseline$study)[i])]
     temp<-subset(jointdat,idstemp)
     class(temp)<-'jointdata'
     assign(paste('jointdat',unique(jointdat$baseline$study)[i],
                  sep='.'),temp)
  }

############################
### Example using joineR fits

#use the joineR package to fit study specific joint models
joineRfit1<-joint(data = jointdat.1, long.formula = Y ~ 1 + time + treat,
              surv.formula = Surv(survtime, cens) ~ treat,
              model = 'intslope')

joineRfit2<-joint(data = jointdat.2, long.formula = Y ~ 1 + time + treat,
              surv.formula = Surv(survtime, cens) ~ treat,
              model = 'intslope')

joineRfit3<-joint(data = jointdat.3, long.formula = Y ~ 1 + time + treat,
              surv.formula = Surv(survtime, cens) ~ treat,
              model = 'intslope')

joineRfit4<-joint(data = jointdat.4, long.formula = Y ~ 1 + time + treat,
              surv.formula = Surv(survtime, cens) ~ treat,
              model = 'intslope')

joineRfit5<-joint(data = jointdat.5, long.formula = Y ~ 1 + time + treat,
              surv.formula = Surv(survtime, cens) ~ treat,
              model = 'intslope')

joineRfit1SE<-jointSE(fitted = joineRfit1, n.boot = 200)
joineRfit2SE<-jointSE(fitted = joineRfit2, n.boot = 200)
joineRfit3SE<-jointSE(fitted = joineRfit3, n.boot = 200)
joineRfit4SE<-jointSE(fitted = joineRfit4, n.boot = 200)
joineRfit5SE<-jointSE(fitted = joineRfit5, n.boot = 200)

joineRfits<-list(joineRfit1, joineRfit1SE,
joineRfit2, joineRfit2SE,
joineRfit3, joineRfit3SE,
joineRfit4, joineRfit4SE,
```

```
    joineRfit5, joineRfit5SE)

    names(joineRfits)<-c('joineRfit1', 'joineRfit1SE',
                         'joineRfit2', 'joineRfit2SE',
                         'joineRfit3', 'joineRfit3SE',
                         'joineRfit4', 'joineRfit4SE',
                         'joineRfit5', 'joineRfit5SE')

    #perform the second stage of the two stage MA
    MAjoineRfits<-jointmeta2(fits = joineRmodels, SE = joineRmodelsSE,
                  longpar = c('time', 'treat1'), survpar = 'treat1',
                  assoc = TRUE, studynames = c('Study 1','Study 2',
                  'Study 3', 'Study 4', 'Study 5'))

    #produce forest plots
    library(meta)
    forest(MAjoineRfits$longMA$treat1)

## End(Not run)
```

`jointmetaplot` *Produce plots of longitudinal and survival outcomes*

Description

This function can produce plots for each study in the dataset of the longitudinal trajectory pannelled by event type with or without a smoother, and kaplan-meier plots for each study which plot the survival probability against time.

Usage

```
jointmetaplot(dataset, study, longoutcome, longtime, survtime, cens, id,
  smoother = FALSE, studynames = NULL, type = c("Longitudinal", "Event",
  "Both"), eventby = NULL, eventconfint = FALSE)
```

Arguments

`dataset`	a `jointdata` object
`study`	the name of the variable holding study membership in the supplied dataset
`longoutcome`	the name of the variable holding the longitudinal outcome in the supplied dataset
`longtime`	the name of the variable holding the longitudinal time variable in the supplied dataset
`survtime`	the name of the variable holding the survival time variable in the supplied dataset
`cens`	the name of the variable holding the censoring variable in the supplied dataset
`id`	the name of the variable holding the id in the supplied dataset

smoother	a logical indicating whether or not a smoother should be displayed on the longitudinal plot
studynames	a vector of character strings giving the names to label the study plots by - the first element of this vector will be the label for the plots for the first study in the dataset for both the longitudinal and the survival plots
type	option to select what type of plots should be returned. If just plots of the longitudinal trajectories are required then `type = 'Longitudinal'`. Else if just plots of the survival probabilities are required then `type = 'Survival'`. Finally if both survival and longitudinal plots are required then this should be set to `type = 'Both'`
eventby	an optional character string giving a grouping variable that the graph of survival probability by time will be split by.
eventconfint	a logical value indicating whether the survival plot should contain confidence intervals or not. Defaults to FALSE.

Value

Returns an object of class `'jointplots'`. This contains an element labelled `'longplots'` if `type` in the function call is set to one of `'Longitudinal'` or `'Both'`, and an element labelled `'eventplots'` if `type` in the function call is set to one of `'Survival'` or `'Both'`. The element `'longplots'` is a list of ggplot2 objects plotting the longitudinal trajectories for each study, and is of length equal to the number of studies in the supplied dataset. The element `'eventplots'` is a list of ggplot2 objects plotting the survival probabilities for each study and is of length equal to the number of studies in the supplied dataset.

To plot a particular graph, it can be called by position from the relevent element of the returned `'jointplots'` in the same way that an element in a particular position is called from a list, or it can be called by name if `study.names` supplied to the function call.

This function supplies separate plots for each study in the dataset. To arrange these plots into one grid, use the function jointmetaplotall.

See Also

jointdata, ggplot, jointmetaplotall

Examples

```
#change data to jointdata format
jointdat<-tojointdata(longitudinal = simdat$longitudinal,
                      survival = simdat$survival, id = 'id',
                      longoutcome = 'Y', timevarying = c('time','ltime'),
                      survtime = 'survtime', cens = 'cens',
                      time = 'time')

#ensure variables are correctly formatted
jointdat$baseline$study <- as.factor(jointdat$baseline$study)
jointdat$baseline$treat <- as.factor(jointdat$baseline$treat)

#produce plots
```

```
sepplots<-jointmetaplot(dataset = jointdat, study = 'study',
                    longoutcome = 'Y', longtime = 'time',
                    survtime = 'survtime', cens = 'cens', id = 'id',
                    smoother = TRUE, studynames = c('Study 1', 'Study 2',
                    'Study 3', 'Study 4', 'Study 5'), type = 'Both')
```

`jointmetaplotall` *Arrange study plots into a grid*

Description

This function is designed to take the output from `jointmetaplot` and output the study plots of each type arranged into a grid.

Usage

```
jointmetaplotall(plotlist, ncol, nrow = NULL, top = NULL,
  type = c("Longitudinal", "Event", "Both"))
```

Arguments

`plotlist`	the output from running the `jointmetaplot` function.
`ncol`	the number of columns of the grid to arrange the plots in. This must be supplied to the function
`nrow`	the number of rows of the grid to arrange the plot in. This is an optional parameter, which if not supplied is calculated in the function based on the number of supplied plots and the specified value of `ncol`.
`top`	a character string to act as the title for the plots
`type`	option to select what type of plots should be returned. If just the grid of the longitudinal trajectories are required then `type = 'Longitudinal'`. Else if just the grid of the survival probabilities graphs are required then `type = 'Survival'`. Finally if grids of both survival and longitudinal plots are required then this should be set to `type = 'Both'`. If both, then the same title as supplied to `top` will be used, similarly for `ncol` and `nrow`.

Value

An object of class `'jointplotsall'` is returned. If in the function call `type = 'Longitudinal'` or `type = 'Both'` then the element in the returned object names `'longall'` is the arranged grid of longitudinal trajectory plots from each study in the dataset. If `type = 'Survival'` or `type = 'Both'` then the element in the returned object labelled `'eventsall'` is the arranged grid of the survival probability plots from each study in the dataset. The arranged grids can either be printed by name, or by extracting them as you would an element from a list.

See Also

`jointmetaplot`

Examples

```
## Not run:
#change data to jointdata format
jointdat<-tojointdata(longitudinal = simdat$longitudinal,
                      survival = simdat$survival, id = 'id',
                      longoutcome = 'Y', timevarying = c('time','ltime'),
                      survtime = 'survtime', cens = 'cens',
                      time = 'time')

#ensure variables are correctly formatted
jointdat$baseline$study <- as.factor(jointdat$baseline$study)
jointdat$baseline$treat <- as.factor(jointdat$baseline$treat)

#produce plots
#note that inclusion of a smoother sometime results in error messages
#see ggplot2 for error message interpretation
sepplots<-jointmetaplot(dataset = jointdat, study = 'study',
                    longoutcome = 'Y', longtime = 'time',
                    survtime = 'survtime', cens = 'cens', id = 'id',
                    smoother = TRUE, studynames = c('Study 1', 'Study 2',
                    'Study 3', 'Study 4', 'Study 5'), type = 'Both')

allplot2<-jointmetaplotall(plotlist = sepplots, ncol =2,
          top = 'All studies', type = 'Both')

## End(Not run)
```

`jointmetaSE` *Bootstrapping function to obtain standard errors for jointmeta1 fit*

Description

This function takes the results of a `jointmeta1` fit and bootstraps it to find the standard errors of the parameter estimates.

Usage

```
jointmetaSE(fitted, n.boot, gpt, max.it, tol, print.detail = FALSE,
  overalleffects = NULL)
```

Arguments

`fitted`	a `jointmeta1` object
`n.boot`	the number of bootstraps to conduct. Note that confidence intervals will only be calculated if `n.boot` is greater than 100.
`gpt`	the number of quadrature points over which the integration with respect to the random effects will be performed. Will default to `gpt = 5`.

`max.it` the maximum number of iterations that the EM algorithm will perform for each bootstrap fit. Will default to 350.

`tol` the tolerance level before convergence of the algorithm is considered to have occurred. Default value is tol = 0.001.

`print.detail` this argument determines the level of printing that is done during the bootstrapping. If `TRUE` then the parameter estimates from each bootstrap sample are output. Otherwise a progress bar is printed to indicated the proportion of bootstraps currently completed.

`overalleffects` this argument indicates what if any overall effects will have their standard errors and confidence intervals calculated during the bootstrap procedure. An example of an overall effect would be the combined value of a treatment effect, and a treatment by study membership interaction. The overall treatment effect (the sum of these two values) could be of interest in an investigation. This argument is a list containing two elements, `long` and `surv`. Each of these elements contains a list of vectors, each of which contains the names of the parameters that make up the required overall effects

Details

This function takes the results of a one stage joint model fit to data from multiple studies using the function `jointmeta1` and performs `n.boot` bootstraps to determine the standard errors of the parameter estimates, and their confidence intervals if `n.boot > 100`.

The parameter `overalleffects` is designed for use in cases where interaction terms are included in the model specification, for example a model fitted using `jointmeta1` which includes both `treat` and `treat:study` where `treat` is a binary treatment indicator variable and `study` is a study indicator. In this case it may be of interest to calculate the confidence interval for the value of `treat + treat:study` for a given study. This is done by calculating the value of the expression for each bootstrap, and calculating the standard errors for the expression in the same way as for the other parameters. Any overall effects to be calculated for the longitudinal sub-model are supplied as a list named `long` in the list `overalleffects`, with each element of this list containing a vector of the character names of the fixed effects to be summed to form an overall effect. Overall effects from the survival model are specified in a similar way to an element named `surv`

Value

a list containing three elements:

`results` a data frame containing the estimates, standard errors and 95 any overall effects requested.

`covmat` the covariance matrix for the model parameters

`bootstraps` a data frame containing the results of each bootstrap

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')
```

```
#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 ## Not run:
     #calculate the SE
     onestagefitSE <- jointmetaSE(fitted = onestagefit, n.boot = 200)

## End(Not run)
```

`onestage0` *One stage jointmeta1 fit and bootstrapped standard errors*

Description

A list of length two containing a one stage jointmeta1 fit and corresponding bootstrapped standard errors.

Usage

```
onestage0
```

Format

A list of 2 objects:

`onestagefit0` an object of class `jointmeta1`

`onestagefit0SE` an object of class `jointmeta1SE`

Details

These are the results of using the `jointmeta1` function to fit a one stage joint meta model for multi-study data, and also the bootstrap results of applying the `jointmetaSE` function to the resulting model fit. The data used is the `simdat` data available in the `joineRmeta` package. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3).

The format of this model is as follows. The structure of the longitudinal sub-model is:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + \epsilon_{kij}$$

Y_{kij} represents the continuous longitudinal outcome for the ith individual in the kth study at the jth time point, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(\beta_{21}treat + \alpha^{(2)}(b^{(2)}_{0ki} + b^{(2)}_{1ki}time))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the unspecified baseline hazard. This baseline was not stratified by study. The fixed effect coefficient is represented by β_{21}. A proportional random effects only association structure links the sub-models, with $\alpha^{(2)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation of the individual in question from the population mean longitudinal trajectory.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

This is a naive model as it analyses data from all the studies in the dataset but does not account for between study heterogeneity (differences between the studies included in the dataset) in any way.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

The code used to fit this one stage model was:

```
onestagefit0<-jointmeta1(data = jointdat, long.formula = Y ~ 1 +  time + treat, long.rand.ind = c('ir
F)
```

And the code used to bootstrap the model was:

```
onestagefit0SE<-jointmetaSE(fitted = onestagefit0, n.boot = 200)
```

See Also

jointmeta1, jointmetaSE

---

onestage1 *One stage jointmeta1 fit and bootstrapped standard errors*

Description

A list of length two containing a one stage jointmeta1 fit and corresponding bootstrapped standard errors.

Usage

```
onestage1
```

Format

A list of 2 objects:

`onestagefit1` an object of class `jointmeta1`

`onestagefit1SE` an object of class `jointmeta1SE`

Details

These are the results of using the `jointmeta1` function to fit a one stage joint meta model for multi-study data, and also the bootstrap results of applying the `jointmetaSE` function to the resulting model fit. The data used is the `simdat` data available in the `joineRmeta` package. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3).

The format of this model is as follows. The structure of the longitudinal sub-model is:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + \beta_{13}study + \beta_{14}treat * study + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + \epsilon_{kij}$$

Y_{kij} represents the continuous longitudinal outcome for the ith individual in the kth study at the jth time point, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$. The study membership variable, a factor variable, is represented by $study$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(\beta_{21}treat + \beta_{22}study + \beta_{23}treat * study + \alpha^{(2)}(b^{(2)}_{0ki} + b^{(2)}_{1ki}time))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the unspecified baseline hazard. This baseline was not stratified by study. The fixed effect coefficients are represented by β terms. A proportional random effects only association structure links the sub-models, with $\alpha^{(2)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation of the individual in question from the population mean longitudinal trajectory.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

This model accounts for between study heterogeneity by including study membership and interactions between the study membership and the treatment assignment in the fixed effects of both sub-models.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

The code used to fit this one stage model was:

```
onestagefit1<-jointmeta1(data = jointdat, long.formula = Y ~ 1 +  time + treat*study, long.rand.ind =
'randprop', surv.formula = Surv(survtime, cens) ~ treat*study, study.name =
'study', strat = F)
```

And the code used to bootstrap the model was:

```
onestagefit1SE<-jointmetaSE(fitted = onestagefit1, n.boot = 200,  overalleffects = list(long = list(c
```

See Also

jointmeta1, jointmetaSE

---

onestage2 *One stage jointmeta1 fit and bootstrapped standard errors*

Description

A list of length two containing a one stage jointmeta1 fit and corresponding bootstrapped standard errors.

Usage

```
onestage2
```

Format

A list of 2 objects:

`onestagefit2` an object of class `jointmeta1`

`onestagefit2SE` an object of class `jointmeta1SE`

Details

These are the results of using the `jointmeta1` function to fit a one stage joint meta model for multi-study data, and also the bootstrap results of applying the `jointmetaSE` function to the resulting model fit. The data used is the `simdat` data available in the `joineRmeta` package. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3).

The format of this model is as follows. The structure of the longitudinal sub-model is:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + b^{(3)}_{0k} + b^{(3)}_{1k}treat + \epsilon_{kij}$$

Where Y_{kij} represents the continuous longitudinal outcome for the ith individual in the kth study at the jth time point, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. A superscript of 3 indicates study level random effects, or level 3 random effects. This

means that they can take a unique value for each study in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_0(t)exp(\beta_{21}treat + \alpha^{(2)}(b^{(2)}_{0ki} + b^{(2)}_{1ki}time) + \alpha^{(3)}(b^{(2)}_{0k} + b^{(3)}_{1k}treat))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_0(t)$ represents the unspecified baseline hazard. This baseline was not stratified by study. The fixed effect coefficients are represented by β terms. A proportional random effects only association structure links the sub-models, with $\alpha^{(2)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation of the individual in question from the population mean longitudinal trajectory, and $\alpha^{(3)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

This model accounts for between study heterogeneity using study level random effects.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

The code used to fit this one stage model was:

```
onestagefit2<-jointmeta1(data = jointdat, long.formula = Y ~ 1 +  time + treat, long.rand.ind = c('ir
treat, study.name = 'study', strat = F)
```

And the code used to bootstrap the model was:

```
onestagefit2SE<-jointmetaSE(fitted = onestagefit2, n.boot = 200)
```

See Also

jointmeta1, jointmetaSE

`onestage3` *One stage jointmeta1 fit and bootstrapped standard errors*

Description

A list of length two containing a one stage jointmeta1 fit and corresponding bootstrapped standard errors.

Usage

```
onestage3
```

Format

A list of 2 objects:

`onestagefit3` an object of class `jointmeta1`

`onestagefit3SE` an object of class `jointmeta1SE`

Details

These are the results of using the `jointmeta1` function to fit a one stage joint meta model for multi-study data, and also the bootstrap results of applying the `jointmetaSE` function to the resulting model fit. The data used is the `simdat` data available in the `joineRmeta` package. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3).

The format of this model is as follows. The structure of the longitudinal sub-model is:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + \beta_{13}study + \beta_{14}treat * study + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + \epsilon_{kij}$$

Y_{kij} represents the continuous longitudinal outcome for the ith individual in the kth study at the jth time point, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates that these are individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$. The study membership variable, a factor variable, is represented by $study$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_{0k}(t)exp(\beta_{21}treat + \alpha^{(2)}(b^{(2)}_{0ki} + b^{(2)}_{1ki}time))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_{0k}(t)$ represents the unspecified baseline hazard which is stratified by study. The fixed effect coefficients are represented by β terms. A proportional random effects only association structure links the sub-models, with $\alpha^{(2)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation of the individual in question from the population mean longitudinal trajectory.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

This model accounts for between study heterogeneity by including study membership and interactions between the study membership and the treatment assignment in the fixed effects of the longitudinal sub-model, and by stratifying the baseline hazard by study in the survival sub-model.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

The code used to fit this one stage model was:

```
onestagefit3<-jointmeta1(data = jointdat, long.formula = Y ~ 1 +  time + treat*study, long.rand.ind =
'randprop', surv.formula = Surv(survtime, cens) ~ treat, study.name =  'study', strat = T)
```

And the code used to bootstrap the model was:

```
onestagefit3SE<-jointmetaSE(fitted = onestagefit3, n.boot = 200,  overalleffects = list(long = list(c
```

See Also

jointmeta1, jointmetaSE

---

onestage4 *One stage jointmeta1 fit and bootstrapped standard errors*

Description

A list of length two containing a one stage jointmeta1 fit and corresponding bootstrapped standard errors.

Usage

```
onestage4
```

Format

A list of 2 objects:

onestagefit4 an object of class jointmeta1

onestagefit4SE an object of class jointmeta1SE

Details

These are the results of using the jointmeta1 function to fit a one stage joint meta model for multi-study data, and also the bootstrap results of applying the jointmetaSE function to the resulting model fit. The data used is the simdat data available in the joineRmeta package. This data has three levels, namely the longitudinal measurements at level 1, nested within individuals (level 2) who are themselves nested within studies (level 3).

The format of this model is as follows. The structure of the longitudinal sub-model is:

$$Y_{kij} = \beta_{10} + \beta_{11}time + \beta_{12}treat + \beta_{13}study + b^{(2)}_{0ki} + b^{(2)}_{1ki}time + b^{(3)}_{1k}treat + \epsilon_{kij}$$

Where Y_{kij} represents the continuous longitudinal outcome for the ith individual in the kth study at the jth time point, fixed effect coefficients are represented by β, random effects coefficients by b and the measurement error by ϵ. For the random effects the superscript of 2 indicates individual level, or level 2 random effects. This means they take can take a unique value for each individual in the dataset. A superscript of 3 indicates study level random effects, or level 3 random effects. This means that they can take a unique value for each study in the dataset. The longitudinal time variable is represented by $time$, and the treatment assignment variable (a binary factor) is represented by $treat$. The study membership, a factor variable, is represented by $study$.

The survival sub-model had format:

$$\lambda_{ki}(t) = \lambda_{0k}(t)exp(\beta_{21}treat + \alpha^{(2)}(b^{(2)}_{0ki} + b^{(2)}_{1ki}time) + \alpha^{(3)}(b^{(2)}_{0k} + b^{(3)}_{1k}treat))$$

In the above equation, $\lambda_{ki}(t)$ represents the survival time of the individual i in study k, and $\lambda_{0k}(t)$ represents the unspecified baseline hazard which is stratified by study. The fixed effect coefficients are represented by β terms. A proportional random effects only association structure links the sub-models, with $\alpha^{(2)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation of the individual in question from the population mean longitudinal trajectory, and $\alpha^{(3)}$ representing the association between the longitudinal and survival outcomes attributable to the deviation.

We differentiate between the fixed effect coefficients in the longitudinal and the survival sub-models by varying the first number present in the subscript of the fixed effect, which takes a 1 for coefficients from the longitudinal sub-model and a 2 for coefficients from the survival sub-model.

This model accounts for between study heterogeneity using study level random effects.

These fits have been provided in this package for use with the package vignette, see the vignette for more information.

The code used to fit this one stage model was:

```
onestagefit4<-jointmeta1(data = jointdat, long.formula = Y ~ 1 +  time + treat + study, long.rand.inc
c('treat'), sharingstrct = 'randprop', surv.formula = Surv(survtime, cens)
~ treat, study.name = 'study', strat = T)
```

And the code used to bootstrap the model was:

```
onestagefit4SE<-jointmetaSE(fitted = onestagefit4, n.boot = 200)
```

See Also

jointmeta1, jointmetaSE

`print.jointmeta1` *Print function for* `jointmeta1` *objects*

Description

A function to print a `jointmeta1.object`.

Usage

```
## S3 method for class 'jointmeta1'
print(x, ...)
```

Arguments

x	a jointmeta1.object, the result of fitting a jointmeta1 model
...	additional arguments; currently none are used.

Value

An object inheriting from class `print.jointmeta1` with all components included in x (see `jointmeta1.object`).

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 #print the fitted multi-study joint model
 print(onestagefit)
```

`print.jointmeta1SE` *Print function for* `jointmeta1SE` *objects*

Description

This function extracts the results of the bootstrapping function `jointmetaSE` applied to the results of a one stage joint meta model fitted using `jointmeta1`, namely `jointmeta1SE` objects. The bootstrapping function returns not only results but also the covariance matrix for the estimated parameters and the results of each bootstrap. This print function allows just the results of the bootstraps to easily be displayed.

Usage

```
## S3 method for class 'jointmeta1SE'
print(x, ...)
```

Arguments

`x`	a `jointmeta1SE` object, the result of applying `jointmetaSE` to the joint model fit obtained by applying `jointmeta1` to a multi-study joint data set.
`...`	additional arguments; currently none are used.

Value

a data frame containing the estimates, standard errors and 95% confidence intervals for the parameters from the model and any overall effects requested in the `jointmetaSE` function call.

See Also

`jointmeta1`, `jointmetaSE`, `jointmeta1SE.object`, `jointmeta1.object`

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 ## Not run:
     #calculate the SE
     onestagefitSE <- jointmetaSE(fitted = onestagefit, n.boot = 200)

     #print the results of the bootstrap function
     print(onestagefitSE)

## End(Not run)
```

---

rancov *Function to extract the estimated covariance matrices for the random effects specified in the model*

Description

A function to allow the random effects covariance matrix for a particular level of random effects specified in the sub-model to be extracted from the `jointmeta1` model fit.

Usage

```
rancov(fitted, type = c("individual", "study"))
```

Arguments

fitted	a jointmeta1.object
type	a character string indicating what level the random effects covariance matrix should be returned for. If the individual level random effects covariance matrix is required then type = "individual". If the study level random effects covariance matrix is required then type = "study". Note that if study level random effects are not included in the model, then attempting to extract them will result in an error message.

Value

a matrix of dimensions equal to the number of random effects at the level specified by the type parameter.

See Also

jointmeta1, jointmeta1.object

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 #extract the individual level random effects covariance matrix
 rancov(onestagefit, type = "individual")

 #extract the study level random effects covariance matrix
 rancov(onestagefit, type = "study")
```

`ranef.jointmeta1` *Function to extract estimated random effects*

Description

This function extracts the estimated values of the random effects from a supplied `jointmeta1` fit.

Usage

```
## S3 method for class 'jointmeta1'
ranef(object, type = c("individual", "study"), ...)
```

Arguments

`object`	a `jointmeta1` object (the result of fitting a model using `jointmeta1`, see `jointmeta1.object`)
`type`	the type of random effects to return. Set `type = 'individual'` to return the estimates of the individual level random effects. Set `type = 'study'` to return the estimates of the level random effects if included in the model.
`...`	additional arguments; currently none are used.

Value

If `type = 'individual'` then a list of matrices containing the individual level random effects is returned. This list is of length equal to the number of studies in the dataset. Each matrix has number of rows equal to the number of individuals in the corresponding study, and number of columns equal to the number of individual level random effects.

If `type = 'study'` then if study level random effects are present in the supplied model fit, a matrix of the estimated study level random effects is returned, with number of rows equal to the number of studies in the dataset, and number of columns equal to the number of study level random effects. If study level random effects are requested but are not present in the supplied model fit, an error message is returned.

See Also

`jointmeta1`, `jointmeta1.object`, `fixef`

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)
```

```
#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 #extract the individual level random effects covariance matrix
 ranef(onestagefit, type = 'individual')

 #extract the study level random effects covariance matrix
 ranef(onestagefit, type = 'study')
```

`removeafter` *Code to remove longitudinal information recorded after survival outcome*

Description

This function is designed to remove any longitudinal information recorded after the survival time for each individual. If the survival event is not terminal, it is possible that longitudinal information is available in the data after the survival time, but it should not contribute to the joint analysis. This function takes and returns a `jointdata` object.

Usage

```
removeafter(data, longitudinal, survival, id, time)
```

Arguments

`data`	a jointdata object (see `jointdata`)
`longitudinal`	a character string denoting name of the variable holding the longitudinal outcome of interest.
`survival`	a character string denoting the name of the variable holding the survival time for the event of interest.
`id`	a character string denoting the name of the variable holding the id variable for the data.
`time`	a character string denoting the name of the variable holding the longitudinal time variable.

Details

This function removes any longitudinal information recorded for an individual after their survival time. A joint data object should have the id column as the first column in each of the `survival`, `longitudinal` and `baseline` datasets. In the `survival` dataset the second column should be the survival time and the third should be the censoring variable. In the `longitudinal` dataset, the second column should be longitudinal outcome, the third the longitudinal time variable and the remaining columns any other time varying covariates. The baseline dataset should have columns 2 and onwards containing time stationary covariates such as treatment group assignment or study membership.

This function does not need to be run on the results of the multi-study data simulation function `simjointmeta`, because the longitudinal data simulated under this function is already capped at the individual's survival time.

Value

a jointdata object, see `jointdata`

See Also

`jointdata`

Examples

```
## Not run:
#the dataset simdat3 in this package contains joint data where longitudinal
#data exists after individual's survival times.
str(simdat3)

#first this data needs to be changed to a jointdata object
jointdat3<-tojointdata(longitudinal = simdat3$longitudinal,
                 survival = simdat3$survival, id = 'id', longoutcome = 'Y',
                 timevarying = c('time','ltime'), survtime = 'survtime',
                 cens = 'cens',time = 'time')

#then additional data recorded after the survival time can be removed
jointdat3.1<-removeafter(data = jointdat3, longitudinal = 'Y',
                 survival = 'survtime', id = 'id', time = 'time')

#we can compare the two datasets to see the removed data
str(jointdat3)
str(jointdat3.1)

## End(Not run)
```

simdat *Simulated joint longitudinal and survival dataset containing 5 studies*

Description

A simulated dataset containing a single continuous longitudinal outcome and a single survival outcome, with data available from 5 studies.

Usage

```
simdat
```

Format

A list of three objects:

`longitudinal` A list of long format longitudinal datasets one for each of the 5 studies included in the dataset. Each of these datasets contains the following variables:

- `id` long version of the id variable for the data. Identical ids between the longitudinal and the survival datasets identify the same individual
- `Y` a continuous longitudinal outcome
- `time` the longitudinal time variable
- `study` a long version of the study membership indicator
- `intercept` a long version of the intercept, always takes a value of 1
- `treat` a long version of the binary treatment group indicator
- `ltime` a duplicate of the longitudinal time variable, duplicated as part of the longitudinal data simulation process

`survival` A list of survival datasets, one for each of the 5 studies included in the dataset. Each of these datasets contains the following variables:

- `id` the id variable for the data. Identical ids between the longitudinal and the survival datasets identify the same individual
- `survtime` the survival time for each individual at which they experienced the event or were censored. This is on the same scale as the longitudinal time measurements.
- `cens` censoring indicator for the survival data where 1 indicates an event and 0 indicates censoring
- `study` study membership indicator
- `treat` binary treatment group indicator

`percentevent` A list of the percentage of events experienced in each datasets. The first element contains the percentage of events observed for the first simulated study and so on.

Details

This is a simulated dataset generated using the simjointmeta function using the following function call:

```
simdat<-simjointmeta(k = 5, n = rep(500, 5), sepassoc = FALSE, ntms  = 5, longmeasuretimes = c(0, 1,
1, sigb_ind = matrix(c(1,0.5,0.5,1.5),nrow=2), sigb_stud =  matrix(c(1,0.5,0.5,1.5),nrow=2), vare = 0
```

Note that this will not give you identical data to that held in simdat due to the differences in starting seed.

See Also

simjointmeta

---

simdat2 *Simulated joint longitudinal and survival dataset containing 3 studies*

Description

A simulated dataset containing a single continuous longitudinal outcome and a single survival outcome, with data available from 3 studies.

Usage

```
simdat2
```

Format

A list of three objects:

`longitudinal` A list of long format longitudinal datasets one for each of the 3 studies included in the dataset. Each of these datasets contains the following variables:

- `id` long version of the id variable for the data. Identical ids between the longitudinal and the survival datasets identify the same individual
- `Y` a continuous longitudinal outcome
- `time` the longitudinal time variable
- `study` a long version of the study membership indicator
- `intercept` a long version of the intercept, always takes a value of 1
- `treat` a long version of the binary treatment group indicator
- `ltime` a duplicate of the longitudinal time variable, duplicated as part of the longitudinal data simulation process

`survival` A list of survival datasets, one for each of the 5 studies included in the dataset. Each of these datasets contains the following variables:

- `id` the id variable for the data. Identical ids between the longitudinal and the survival datasets identify the same individual
- `survtime` the survival time for each individual at which they experienced the event or were censored. This is on the same scale as the longitudinal time measurements.

`cens` censoring indicator for the survival data where 1 indicates an event and 0 indicates censoring

`study` study membership indicator

`treat` binary treatment group indicator

`percentevent` A list of the percentage of events experienced in each datasets. The first element contains the percentage of events observed for the first simulated study and so on.

Details

This is a simulated dataset generated by subsetting the `simdat` dataset to leave only three studies with 100 individuals in each study. This dataset is for demonstration purposes only within the package.

See Also

`simjointmeta`

`simdat3` *Simulated joint longitudinal and survival dataset containing 5 studies*

Description

A simulated dataset containing a single continuous longitudinal outcome and a single survival outcome, with data available from 5 studies. This dataset does not have longitudinal measurements capped at each individual's survival time.

Usage

```
simdat3
```

Format

A list of three objects:

`longitudinal` A list of long format longitudinal datasets one for each of the 5 studies included in the dataset. Each of these datasets contains the following variables:

`id` long version of the id variable for the data. Identical ids between the longitudinal and the survival datasets identify the same individual

`Y` a continuous longitudinal outcome

`time` the longitudinal time variable

`study` a long version of the study membership indicator

`intercept` a long version of the intercept, always takes a value of 1

`treat` a long version of the binary treatment group indicator

`ltime` a duplicate of the longitudinal time variable, duplicated as part of the longitudinal data simulation process

`survival` A list of survival datasets, one for each of the 5 studies included in the dataset. Each of these datasets contains the following variables:

- `id` the id variable for the data. Identical ids between the longitudinal and the survival datasets identify the same individual
- `survtime` the survival time for each individual at which they experienced the event or were censored. This is on the same scale as the longitudinal time measurements.
- `cens` censoring indicator for the survival data where 1 indicates an event and 0 indicates censoring
- `study` study membership indicator
- `treat` binary treatment group indicator

`percentevent` A list of the percentage of events experienced in each datasets. The first element contains the percentage of events observed for the first simulated study and so on.

Details

This is a simulated dataset generated in order to show how the function `removeafter` works, as it does not cap longitudinal outcomes at each individual's survival time. This data was generated by manually stepping through the code available in simjointmeta and retaining instead of discarding any simulated longitudinal measurements recorded after the individual in question's survival time.

See Also

`simjointmeta`, `removeafter`

---

simjointmeta *Simulation of multi-study joint data*

Description

Function to allow the simulation of a correlated single continuous longitudinal outcome and a single survival outcome for data from multiple studies. The longitudinal sub-model contains a fixed intercept, time (slope) term and a binary treatment assignment covariate, whilst the survival sub-model contains only a binary treatment assignment covariate.

Usage

```
simjointmeta(k = 5, n = rep(500, 5), sepassoc = FALSE, ntms = 5,
  longmeasuretimes = c(0, 1, 2, 3, 4), beta1 = c(1, 1, 1), beta2 = 1,
  rand_ind = c("intslope", "int"), rand_stud = c("int", "inttreat", "treat",
  NULL), gamma_ind = 1, gamma_stud = NULL, sigb_ind, sigb_stud = NULL,
  vare = 0.01, theta0 = -3, theta1 = 1, censoring = TRUE,
  censlam = exp(-3), truncation = FALSE,
  trunctime = max(longmeasuretimes))
```

Arguments

k	the number of studies to be simulated
n	a vector of length equal to k denoting the number of individuals to simulate per study
sepassoc	a logical taking value FALSE if proportional association is required, TRUE if a separate association parameter is required for each random effect shared between the sub-models
ntms	the maximum possible number of longitudinal measurements - should equal the length of the supplied longmeasuretimes
longmeasuretimes	a vector giving the exact times of the longitudinal measurement times. If this is not specified in the function call then the measurement times of the longitudinal outcome are set to start at 0 then take integer values up to and including ntms - 1.
beta1	a vector of the fixed effects for the longitudinal sub-model. Here the first element gives the coefficient for a fixed or population intercept, the second gives the coefficient for the binary treatment assignment covariate and the third element gives the covariate for the time (slope) covariate
beta2	the coefficient for the binary treatment assignment covariate
rand_ind	a character string specifying the individual level random effects structure. If rand_ind = 'intslope' then there is an individual specific random intercept and random time (slope) term included in the model. If rand_ind = 'int' then the model includes only a individual specific random intercept.
rand_stud	a character string specifying the study level random effects structure. If this is set to NULL or not specified in the function call then no study level random effects are included in the model that the data is simulated from. There are three options if data is to be simulated with random effects at the study level. If a study level random intercept only is to be included, then set rand_stud = 'int'. Else if a study level random treatment assignment term only is to be included then set rand_stud = 'treat'. Finally if both a study level random intercept and a study level random treatment effect is to be included, then set rand_stud = 'inttreat'.
gamma_ind	parameter specifying the level of association between the longitudinal and survival outcomes attributable to the individual deviation from the population longitudinal trajectory. If different association parameters are required for each study then a list of length equal to the number of studies should be supplied to gamma_ind. If sepassoc = TRUE then gamma_ind should be either a vector of values of length equal to the number of individual level random effects, or a list of vectors each of length equal to the number of individual level random effects. However if sepassoc = FALSE then gamma_ind should be supplied as a single value, or a list of single values.
gamma_stud	parameter specifying the level of association between the longitudinal and survival outcomes attributable to the study level deviation from the overall population longitudinal trajectory. If different association parameters are required for each study then a list of length equal to the number of studies should be supplied to gamma_stud. If sepassoc = TRUE then gamma_stud should be either

a vector of values of length equal to the number of study level random effects, or a list of vectors each of length equal to the number of study level random effects. However if `sepassoc = FALSE` then `gamma_stud` should be supplied as a single value, or a list of single values. This parameter should only be present if `rand_stud` is specified in the function call.

`sigb_ind` the covariance matrix for the individual level random effects. This should have number of rows and columns equal to the number of individual level random effects.

`sigb_stud` the covariance matrix for the study level random effects. This should have number of rows and columns equal to the number of study level random effects. This should only be specified if `rand_stud` is specified in the function call.

`vare` the variance of the measurement error term

`theta0` parameter defining the distribution of the survival times. A separate parameter can be defined per study or a common parameter across all studies. See Bender et al 2005 for advice on approximating appropriate values for `theta0` and `theta1` the using extreme value distribution.

`theta1` parameter defining the distribution of the survival times. A separate parameter can be defined per study or a common parameter across all studies. See Bender et al 2005 for advice on approximating appropriate values for `theta0` and `theta1` the using extreme value distribution.

`censoring` a logical indicating whether the simulated survival times should be censored or not

`censlam` the lambda parameter controlling the simulated exponentially distributed censoring times. This can either be supplied as one value for all studies simulated, or a vector of length equal to the number of studies in the dataset.

`truncation` a logical value to specify whether the simulated survival times should be truncated at a specified time or not.

`trunctime` if `truncation = TRUE` then the survival times will be truncated at the specified `trunctime`

Details

This function allows the simulation of a single continuous longitudinal and a single survival outcome which are potentially correlated. The model simulates data under a joint model with a zero mean random effects only sharing structure. The longitudinal sub-model is adjusted by a fixed or population intercept, time (slope) term and a binary treatment assignment covariate. The survival sub-model is adjusted by only the fixed or population binary treatment assignment covariate.

Random effects can be specified at either just the individual level, or at both the individual and study level. For the options for the random effects see the above parameter definitions.

The parameters controlling the distributions for the survival times and the censoring times can be identical across the studies, or separate values can be supplied for each study. Similarly the association parameters can be identical across studies, or unique to each study.

The simulated longitudinal information is capped at each individual's survival time. If `truncation= TRUE` then the survival times are truncated at the specified `trunctime`.

For description of the methodology of simulating this data see Bender et al 2005, and Austin 2012.

Note that this function does not return data in a `jointdata` format. Function `tojointdata` can help to reformat this data into a `jointdata` format.

Value

This function returns a list with three named elements. The first element is named `'longdat'`, the second `'survdat'`, the third `'percentevent'`. Each of these elements is a list of length equal to the number of studies specified to simulate in the function call.

The element `'longdat'` is a list of the simulated longitudinal data sets. Each longitudinal dataset contains the following variables:

`id` a numeric id variable

`Y` the continuous longitudinal outcome

`time` the numeric longitudinal time variable

`study` a study membership variable

`intercept` an intercept term

`treat` a treatment assignment variable to one of two treatment groups

`ltime` a duplicate of the longitudinal time variable

The element `'survdat'` is a list of the simulated survival data sets. Each survival dataset contains the following variables:

`id` a numeric id variable

`survtime` the numeric survival times

`cens` the censoring indicator

`study` a study membership variable

`treat` a treatment assignment variable to one of two treatment groups

The element `'percentevent'` is a list of the percentage of events over censorings seen in the simulated survival data.

References

Bender et al (2005) Generating survival times to simulate Cox proportional hazards models. Statistics in Medicine 24:1713–1723

Austin (2012) Generating survival times to simulate Cox proportional hazards models with time-varying covariates. Statistics in Medicine 31: 3946–3958

See Also

`tojointdata`

Examples

```
#simulated data without study level variation specified
exampledat1<-simjointmeta(k = 5, n = rep(500, 5), sepassoc = FALSE,
             ntms = 5, longmeasuretimes = c(0, 1, 2, 3, 4),
             beta1 = c(1, 2, 3), beta2 = 1, rand_ind = 'intslope',
             rand_stud = NULL, gamma_ind = 1,
             sigb_ind = matrix(c(1,0.5,0.5,1.5),nrow=2), vare = 0.01,
             theta0 = -3, theta1 = 1, censoring = TRUE, censlam = exp(-3),
             truncation = FALSE, trunctime = max(longmeasuretimes))

#simulated data with different parameters for each study for the
#association parameters, censoring distribution parameters and survival time
#parameters
gamma_ind_set<-list(c(0.5, 1), c(0.4, 0.9), c(0.6, 1.1), c(0.5, 0.9),
                    c(0.4, 1.1))
gamma_stud_set<-list(c(0.6, 1.1), c(0.5, 1), c(0.5, 0.9), c(0.4, 1.1),
                    c(0.4, 0.9))
censlamset<-c(exp(-3), exp(-2.9), exp(-3.1), exp(-3), exp(-3.05))
theta0set<-c(-3, -2.9, -3, -2.9, -3.1)
theta1set<-c(1, 0.9, 1.1, 1, 0.9)

exampledat2<-simjointmeta(k = 5, n = rep(500, 5), sepassoc = TRUE, ntms = 5,
                          longmeasuretimes = c(0, 1, 2, 3, 4),
                          beta1 = c(1, 2, 3), beta2 = 1,
                          rand_ind = 'intslope', rand_stud = 'inttreat',
                          gamma_ind = gamma_ind_set,
                          gamma_stud = gamma_stud_set,
                          sigb_ind = matrix(c(1, 0.5, 0.5, 1.5), nrow = 2),
                          sigb_stud = matrix(c(1, 0.5, 0.5, 1.5), nrow = 2),
                          vare = 0.01, theta0 = theta0set,
                          theta1 = theta1set, censoring = TRUE,
                          censlam = censlamset, truncation = FALSE,
                          trunctime = max(longmeasuretimes))
```

`summary.jointmeta1` *Summary function for jointmeta1*

Description

A function to provide a summary of a `jointmeta1.object`.

Usage

```
## S3 method for class 'jointmeta1'
summary(object, variance = TRUE, ...)
```

Arguments

object	a jointmeta1.object, the result of fitting a jointmeta1 model
variance	a logical if set to TRUE the variances of the measurement errors and the random effects are returned, else if FALSE then the standard errors are returned.
...	additional arguments; currently none are used.

Value

An object inheriting from class summary.jointmeta1 with all components included in object (see jointmeta1.object).

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 #request a summary of the fitted model, with variances printed
 summary(onestagefit, variance = TRUE)

 #request a summary of the fitted model, with standard errors printed
 summary(onestagefit, variance = FALSE)
```

---

tojointdata *Function to change multi-study data into jointdata format*

Description

This function is designed to take data in various formats from multiple studies and output the data in a jointdata format.

Usage

```
tojointdata(dataset = NULL, longitudinal = NULL, survival = NULL,
  baseline = NULL, id, longoutcome, timevarying = NULL, survtime, cens,
  time = NULL, longtimes = NULL)
```

Arguments

dataset	a dataset or list of datasets. The datasets can be of either long or wide format holding both the longitudinal and survival information (and any baseline information). Either the parameter dataset should be specified, or the parameters longitudinal and survival (and baseline if available) should be supplied.
longitudinal	a dataset or list of datasets in long format containing the longitudinal outcome and any time varying covariates. It can also contain baseline information. This should not be supplied if dataset is supplied to the function call.
survival	a dataset or list of datasets in wide format (one row per individual) containing the survival information (survival time and censoring variable). It can also contain baseline information. This should not be supplied if dataset is supplied to the function call.
baseline	a dataset or list of datasets in wide format (one row per individual) containing any baseline information. This variable does not have to be supplied if there is no baseline information, or if it is already contained in the longitudinal or survival datasets. This should not be supplied if dataset is supplied to the function call.
id	a character string holding the name of the id variable. This should be present in all datasets supplied to this function.
longoutcome	a character string holding the name of the longitudinal outcome.
timevarying	a vector of character strings indicating the names of the time varying covariates in the dataset
survtime	a character string denoting the name of the surivival time variable in the dataset
cens	a character string denoting the name of the censoring variable in the dataset
time	a character string to label the time variable if the data is transformed from wide to long format, or the name of the variable holding the time variable if the data is supplied in long format
longtimes	if wide data, the labels that denote the time varying variables to allow the longitudinal data to be returned in long data format.

Details

The data supplied to the jointmeta1 function has to be in a jointdata format. However it is conceivable that data supplied from multiple studies could come in a range of formats.

This function can handle a range of formats in which data from multiple studies may be supplied. This are discussed below. We refer to wide data as data that contains both time varying and time stationary data, with one line per individual with variables measured over time recorded in multiple columns. We refer to long data as data with multiple lines per individual, with time varying covariates differing between rows, but time stationary covariates identical between rows. Survival data are

considered data with survival outcome (survival time and censoring variable) with or without baseline data. Longitiudinal datasets are considered long format datasets containing time varying data potentially with baseline data. Baseline datasets are considered wide format datasets containing non-time varying data measured at baseline. This function can take data in the following formats and output a `jointdata` object.

`One wide dataset` One dataset in wide format (one row per individual) supplied to the parameter `dataset` in the function call with `longitudinal`, `survival` and `baseline` set to `NULL`, or left unspecified in the function call. This dataset would contain all the data from all studies, and the survival, longitudinal and any baseline information would all be present in the same dataset.

`One long dataset` One dataset in long format (multiple rows for each individual) supplied to the parameter `dataset` in the function call with `longitudinal`, `survival` and `baseline` set to `NULL`, or left unspecified in the function call. This dataset would contain all the data from all studies, and the survival, longitudinal and any baseline information would all be present in the same dataset.

`A list of study specific wide datasets` One dataset for each study, each in wide format (one row per individual), supplied to the parameter `dataset` in the function call with `longitudinal`, `survival` and `baseline` set to `NULL`, or left unspecified in the function call. The data from each study would contain the survival, longitudinal and any baseline information.

`A list of study specific long datasets` One dataset for each study, each in long format (multiple rows per individual), supplied to the parameter `dataset` in the function call with `longitudinal`, `survival` and `baseline` set to `NULL` or left unspecified in the function call. The data from each study would contain the survival, longitudinal and any baseline information.

`One longitudinal and one survival dataset with or without an additional baseline dataset` In this case all the longitudinal and time varying data for all studies is supplied in a single dataset in long format to the parameter `longitudinal`. All the survival data for all studies is supplied in a single dataset in wide format to the parameter `survival`. Baseline data can be present in these two datasets, or can also be supplied as a dataset to the parameter `baseline`. If `longitudinal` and `survival` are specified, then parameter `dataset` should be set to `NULL` or left unspecified in the function call. The parameter `baseline` is optional, but should only be specified if parameter `dataset` is `NULL` or unspecified.

`A list of longitudinal and a list of survival datasets with or without a list of baseline datasets` In this case the longitudinal and time varying data for each study is supplied as one element of a list of long format datasets to the parameter `longitudinal`. The survival data for each study is supplied as one element of a list of wide format datasets to the parameter `survival`. Baseline data can be present in these two sets of datasets, or can be supplied as an additional list of datasets one for each study to the parameter `baseline`. If `longitudinal` and `survival` are specified (`baseline` is optional), then parameter `dataset` should be set to `NULL`, or left unspecified in the function call.

The specified id variable should be present in all datasets supplied to the function. Variables containing the same information should be identically named in each supplied dataset, for example if a variable `'age'` is present in one dataset, denoting age of individual at baseline, corresponding variables in other datasets also supplying age at baseline should also be named `'age'`. Similarly, different variables should not share the same name across different datasets, for example there should not be a variable named `'age'` in the longitudinal dataset denoting individual's age at last

longitudinal measurement along with a variable 'age' in the baseline dataset that denotes age of the individual at baseline. Before supplying data to this function, names of variables in each dataset should be checked to confirm that common variables share the same name, and differing variables are appropriately distinguished from each other.

Value

A jointdata object, see jointdata.

See Also

jointdata, jointmeta1

Examples

```
#simdat is a simulated dataset available in the joineRmeta package
#it is supplied as a list of longitudinal and a list of survival datasets,
#each list is of length equal to the number of studies in the entire
#dataset.
jointdat<-tojointdata(longitudinal = simdat$longitudinal,
                      survival = simdat$survival, id = 'id',
                      longoutcome = 'Y', timevarying = c('time','ltime'),
                      survtime = 'survtime', cens = 'cens', time = 'time')
```

---

vcov.jointmeta1SE *Extract the variance covariance matrix from the bootstrapped results*

Description

Function applied to a jointmeta1SE object, the result of the jointmetaSE function to extract the variance covariance matrix for the estimated model parameters

Usage

```
## S3 method for class 'jointmeta1SE'
vcov(object, ...)
```

Arguments

object	an object of class jointmeta1SE
...	additional arguments; currently none are used.

Value

a variance covariance matrix for the fixed effects from the longitudinal sub-model, the time-to-event sub-model, the association parameters, the random effects and the error term.

See Also

jointmeta1, jointmetaSE, jointmeta1SE.object

Examples

```
#change example data to jointdata object
jointdat2<-tojointdata(longitudinal = simdat2$longitudinal,
survival = simdat2$survival, id = 'id',longoutcome = 'Y',
timevarying = c('time','ltime'),
survtime = 'survtime', cens = 'cens',time = 'time')

#set variables to factors
jointdat2$baseline$study <- as.factor(jointdat2$baseline$study)
jointdat2$baseline$treat <- as.factor(jointdat2$baseline$treat)

#fit multi-study joint model
#note: for demonstration purposes only - max.it restricted to 5
#model would need more iterations to truely converge
onestagefit<-jointmeta1(data = jointdat2, long.formula = Y ~ 1 + time +
                        + treat + study, long.rand.ind = c('int', 'time'),
                        long.rand.stud = c('treat'),
                        sharingstrct = 'randprop',
                        surv.formula = Surv(survtime, cens) ~ treat,
                        study.name = 'study', strat = TRUE, max.it=5)

 ## Not run:
     #calculate the SE
     onestagefitSE <- jointmetaSE(fitted = onestagefit, n.boot = 200)

     #extract the variance covariance matrix
     vcov(onestagefitSE)

## End(Not run)
```

Index