Package ‘FastJM’

May 7, 2026

Type Package

Title Semi-Parametric Joint Modeling of Longitudinal and Survival Data

Version 1.6.0

Date 2026-03-28

Maintainer Shanpeng Li <`lishanpeng0913@ucla.edu`>

Encoding UTF-8

Description Implements scalable joint models for large-scale competing risks time-to-event data with one or multiple longitudinal biomarkers using the efficient algorithms developed by Li et al. (2022) <doi:10.1155/2022/1362913> and <doi:10.48550/arXiv.2506.12741>. The time-to-event process is modeled using a cause-specific Cox proportional hazards model with time-fixed covariates, while longitudinal biomarkers are modeled using linear mixed-effects models. The association between the longitudinal and survival processes is captured through shared random effects. The package enables analysis of large-scale biomedical data to model biomarker trajectories, estimate their effects on event risks, and perform dynamic prediction of future events based on patients' longitudinal histories. Functions for simulating survival and longitudinal data for multiple biomarkers are included, along with built-in example datasets. The package also supports modeling a single biomarker with heterogeneous within-subject variability via functionality adapted from the 'JMH' package.

License GPL (>= 3)

NeedsCompilation yes

Imports Rcpp (>= 1.0.7), dplyr, nlme, caret, pec, future, future.apply, rlang (>= 0.4.11)

LinkingTo Rcpp, RcppEigen

Depends R (>= 3.5.0), survival, utils, MASS, statmod, magrittr, stats

RoxygenNote 7.3.2

LazyData true

VignetteBuilder knitr

Suggests testthat (>= 3.0.0), spelling, knitr, rmarkdown

Language en-US

Config/testthat/edition 3

Author Shanpeng Li [aut, cre],
Emily Ouyang [ctb],
Gang Li [ctb]

Repository CRAN

Date/Publication 2026-03-28 16:00:02 UTC

Contents

`anova` *Anova Method for Fitted Joint Models*

Description

Performs a likelihood ratio test between two nested joint models.

Usage

```
## S3 method for class 'jmcs'
anova(object, object2, digits = 4, ...)
```

Arguments

`object`	an object inheriting from class `jmcs`, nested in `object2`.
`object2`	an object inheriting from class `jmcs`.
`digits`	the number of significant digits to use when printing. Default is 4.
`...`	further arguments passed to or from other methods.

Value

A table to summarize the likelihood ratio test.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`jmcs`

Examples

```
# Fit a joint model
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + x1,
            surv.formula = Surv(surv, failure_type) ~ x1 + x2,
            random =  ~ time| ID)

fit2 <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)

anova(fit, fit2)
```

`AUCJMMLSM` *Time-dependent AUC for joint models*

Description

Time-dependent AUC for joint models

Usage

```
AUCJMMLSM(
  seed = 100,
  object,
  landmark.time = NULL,
  horizon.time = NULL,
  obs.time = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  maxiter = 1000,
  n.cv = 3,
  survinitial = TRUE,
  opt = "nlminb",
  initial.para = FALSE,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  metric = c("AUC", "Cindex"),
  ...
)
```

Arguments

`seed`	a numeric value of seed to be specified for cross validation.
`object`	object of class 'JMMLSM'.
`landmark.time`	a numeric value of time for which dynamic prediction starts..
`horizon.time`	a numeric vector of future times for which predicted probabilities are to be computed.
`obs.time`	a character string of specifying a longitudinal time variable.
`method`	estimation method for predicted probabilities. If `Laplace`, then the empirical empirical estimates of random effects is used. If `GH`, then the pseudo-adaptive Gauss-Hermite quadrature is used.
`quadpoint`	the number of pseudo-adaptive Gauss-Hermite quadrature points if `method = "GH"`.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`n.cv`	number of folds for cross validation. Default is 3.

`survinitial`	Fit a Cox model to obtain initial values of the parameter estimates. Default is TRUE.
`opt`	Optimization method to fit a linear mixed effects model, either nlminb (default) or optim.
`initial.para`	Initial guess of parameters for cross validation. Default is FALSE.
`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`metric`	a string to indicate which metric is used.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`JMMLSM`, `survfitJMMLSM`

`cdata` *Simulated competing risks data correlated with ydata*

Description

The `cdata` data frame has 1000 rows and 7 columns.

Usage

```
data(cdata)
```

Format

This data frame contains the following columns:

`ID` patient identifier.

`surv` event time.

`failure_type` event indicator. `0` denotes censoring, `1` risk 1, and `2` risk 2.

`x1` continuous variable.

`x2` treatment indicator. `0` denotes the placebo group and `1` the treatment group.

`gender` gender indicator.

`race` race indicator.

`cdatah` *Simulated competing risks data where event hazards depend on within-subject variance*

Description

The `cdatah` data frame has 200 rows and 6 columns.

Usage

```
data(cdatah)
```

Format

This data frame contains the following columns:

`ID` patient identifier.

`survtime` event time.

`cmprsk` event indicator. `0` denotes censoring, `1` risk 1, and `2` risk 2.

`var1` treatment indicator. `0` denotes the placebo group and `1` the treatment group.

`var2` continuous variable.

`var3` continuous variable.

`Concordancejmcs` *Concordance for joint models*

Description

Concordance for joint models

Usage

```
Concordancejmcs(
  seed = 100,
  object,
  opt = "nlminb",
  n.cv = 3,
  maxiter = 10000,
  initial.optimizer = "BFGS",
  initial.para = TRUE,
  ...
)
```

Arguments

`seed`	a numeric value of seed to be specified for cross validation.
`object`	object of class 'jmcs'.
`opt`	Optimization method to fit a linear mixed effects model, either nlminb (default) or optim.
`n.cv`	number of folds for cross validation. Default is 3.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`initial.optimizer`	Method for numerical optimization to be used. Default is `BFGS`.
`initial.para`	Initial guess of parameters for cross validation. Default is FALSE.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`jmcs`

`ConcordanceJMMLSM` *Concordance for joint models*

Description

Concordance for joint models

Usage

```
ConcordanceJMMLSM(
  seed = 100,
  object,
  opt = "nlminb",
  n.cv = 3,
  maxiter = 10000,
  initial.optimizer = "BFGS",
  initial.para = TRUE,
  ...
)
```

Arguments

`seed`	a numeric value of seed to be specified for cross validation.
`object`	object of class 'JMMLSM'.
`opt`	Optimization method to fit a linear mixed effects model, either nlminb (default) or optim.
`n.cv`	number of folds for cross validation. Default is 3.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`initial.optimizer`	Method for numerical optimization to be used. Default is `BFGS`.
`initial.para`	Initial guess of parameters for cross validation. Default is FALSE.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`JMMLSM`

DynPredAccjmcs *Calculating evaluation metrics for joint models*

Description

Calculating evaluation metrics for joint models

Usage

```
DynPredAccjmcs(
  seed = 100,
  object,
  landmark.time = NULL,
  horizon.time = NULL,
  obs.time = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  maxiter = NULL,
  n.cv = 3,
  survinitial = TRUE,
  quantile.width = 0.25,
  initial.para = FALSE,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  metrics = c("AUC", "Cindex", "Brier", "MAE", "MAEQ"),
  ...
)
```

Arguments

`seed`	a numeric value of seed to be specified for cross validation.
`object`	object of class 'jmcs'.
`landmark.time`	a numeric value of time for which dynamic prediction starts.
`horizon.time`	a numeric vector of future times for which predicted probabilities are to be computed.
`obs.time`	a character string of specifying a longitudinal time variable.
`method`	estimation method for predicted probabilities. If `Laplace`, then the empirical empirical estimates of random effects is used. If `GH`, then the pseudo-adaptive Gauss-Hermite quadrature is used.
`quadpoint`	the number of pseudo-adaptive Gauss-Hermite quadrature points if `method = "GH"`.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`n.cv`	number of folds for cross validation. Default is 3.

`survinitial`	Fit a Cox model to obtain initial values of the parameter estimates. Default is TRUE.
`quantile.width`	a numeric value of width of quantile to be specified. Default is 0.25.
`initial.para`	Initial guess of parameters for cross validation. Default is FALSE.
`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`metrics`	a list to indicate which metric is used.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

See Also

`jmcs`, `survfitjmcs`

`fitted` *Fitted values for joint models*

Description

Extract fitted values for joint models.

Usage

```
## S3 method for class 'jmcs'
fitted(
  object,
  type = c("Marginal", "Subject"),
  process = c("Longitudinal", "Event"),
  ...
)
```

Arguments

`object`	an object inheriting from class `jmcs`.
`type`	for which type of fitted values to calculate.
`process`	for which sub-model to calculate the fitted values.
`...`	further arguments passed to or from other methods.

Value

a numeric vector of fitted values.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

Examples

```
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)

# fitted for the longitudinal process
head(cbind(
  "Marg" = fitted(fit, type = "Marginal", process = "Longitudinal"),
  "Subj" = fitted(fit, type = "Subject", process = "Longitudinal")
))
# fitted for the levent process - marginal survival function
head(fitted(fit, type = "Marginal", process = "Event"))
```

`fixef` *Estimated coefficients estimates for joint models*

Description

Extracts the fixed effects for a fitted joint model.

Usage

```
fixef(object, process = c("Longitudinal", "Event"), ...)
```

Arguments

`object`	an object inheriting from class `jmcs` or `mvjmcs`.
`process`	for which sub-model to extract the estimated coefficients.
`...`	further arguments passed to or from other methods.

Value

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

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

Examples

```
# a joint model fit
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)

# fixed effects for the longitudinal process
fixef(fit, process = "Longitudinal")
# fixed effects for the event process
fixef(fit, process = "Event")
```

`jmcs` *Joint modeling of longitudinal continuous data and competing risks*

Description

Joint modeling of longitudinal continuous data and competing risks

Usage

```
jmcs(
  ydata,
  cdata,
  long.formula,
  random = NULL,
  surv.formula,
  REML = TRUE,
  quadpoint = NULL,
  maxiter = 10000,
  print.para = FALSE,
  initial.para = NULL,
  survinitial = TRUE,
  tol = 1e-04,
  method = "pseudo-adaptive",
  opt = "nlminb"
)
```

Arguments

`ydata`	a longitudinal data frame in long format.
`cdata`	a survival data frame with competing risks or single failure. Each subject has one data entry.
`long.formula`	a formula object with the response variable and fixed effects covariates to be included in the longitudinal sub-model.

`random`	a one-sided formula object describing the random effects part of the longitudinal sub-model. For example, fitting a random intercept model takes the form `~ 1|ID`. Alternatively. Fitting a random intercept and slope model takes the form `~ x1 + ... + xn|ID`.
`surv.formula`	a formula object with the survival time, event indicator, and the covariates to be included in the survival sub-model.
`REML`	a logic object that indicates the use of REML estimator. Default is TRUE.
`quadpoint`	the number of pseudo-adaptive Gauss-Hermite quadrature points. to be chosen for numerical integration. Default is 6 which produces stable estimates in most dataframes.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`print.para`	Print detailed information of each iteration. Default is FALSE, i.e., not to print the iteration details.
`initial.para`	a list of initialized parameters for EM iteration. Default is NULL.
`survinitial`	Fit a Cox model to obtain initial values of the parameter estimates. Default is TRUE.
`tol`	Tolerance parameter. Default is 0.0001.
`method`	Method for proceeding numerical integration in the E-step. Default is pseudo-adaptive.
`opt`	Optimization method to fit a linear mixed effects model, either `nlminb` (default) or `optim`.

Value

Object of class `jmcs` with elements

`beta`	the vector of fixed effects for the linear mixed effects model.
`gamma1`	the vector of fixed effects for type 1 failure for the survival model.
`gamma2`	the vector of fixed effects for type 2 failure for the survival model. Valid only if `CompetingRisk = TRUE`.
`nu1`	the vector of association parameter(s) for type 1 failure.
`nu2`	the vector of association parameter(s) for type 2 failure. Valid only if `CompetingRisk = TRUE`.
`H01`	the matrix that collects baseline hazards evaluated at each uncensored event time for type 1 failure. The first column denotes uncensored event times, the second column the number of events, and the third columns the hazards obtained by Breslow estimator.
`H02`	the matrix that collects baseline hazards evaluated at each uncensored event time for type 2 failure. The data structure is the same as `H01`. Valid only if `CompetingRisk = TRUE`.
`Sig`	the variance-covariance matrix of the random effects.
`sigma`	the variance of the measurement error for the linear mixed effects model.

`iter`	the total number of iterations until convergence.
`convergence`	convergence identifier: 1 corresponds to successful convergence, whereas 0 to a problem (i.e., when 0, usually more iterations are required).
`vcov`	the variance-covariance matrix of all the fixed effects for both models.
`sebeta`	the standard error of `beta`.
`segamma1`	the standard error of `gamma1`.
`segamma2`	the standard error of `gamma2`. Valid only if `CompetingRisk = TRUE`.
`senu1`	the standard error of `nu1`.
`senu2`	the standard error of `nu2`. Valid only if `CompetingRisk = TRUE`.
`seSig`	the vector of standard errors of covariance of random effects.
`sesigma`	the standard error of variance of measurement error for the linear mixed effects model.
`loglike`	the log-likelihood value.
`fitted`	a list with the fitted values: **resid** the vector of estimated residuals for the linear mixed effects model. **fitted** the vector of fitted values for the linear mixed effects model. **fittedmar** the vector of marginal fitted values for the linear mixed effects model. **residmar** the vector of estimated marginal residuals for the linear mixed effects model.
`fittedSurv`	the estimated survival rate evaluated at each uncensored event time.
`FUNB`	the estimated random effects for each subject.
`CompetingRisk`	logical value; TRUE if a competing event are accounted for.
`quadpoint`	the number of Gauss Hermite quadrature points used for numerical integration.
`ydata`	the input longitudinal dataset for fitting a joint model. It has been re-ordered in accordance with descending observation times in `cdata`.
`cdata`	the input survival dataset for fitting a joint model. It has been re-ordered in accordance with descending observation times.
`PropEventType`	a frequency table of number of events.
`LongitudinalSubmodel`	the component of the `long.formula`.
`SurvivalSubmodel`	the component of the `surv.formula`.
`random`	the component of the `random`.
`tol`	the convergence parameter.
`call`	the matched call.
`Quad.method`	the quadrature rule used for integration. If pseudo-adaptive quadrature rule is used, then return `pseudo-adaptive`. Otherwise return `standard`.
`id`	the grouping vector for the longitudinal outcome.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

ranef, fixef, fitted.jmcs,residuals.jmcs, survfitjmcs, plot.jmcs,vcov.jmcs

Examples

```
require(FastJM)
require(survival)
# Load a simulated longitudinal dataset
data(ydata)
# Load a simulated survival dataset with two competing events
data(cdata)

# Fit a joint model
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)
fit
# Extract the parameter estimates of longitudinal sub-model fixed effects
fixef(fit, process = "Longitudinal")
# Extract the parameter estimates of survival sub-model fixed effects
fixef(fit, process = "Event")
# Obtain the random effects estimates for first 6 subjects
head(ranef(fit))
# Obtain the variance-covariance matrix of all parameter estimates
vcov(fit)
# Obtain the result summaries of the joint model fit
summary(fit, process = "Longitudinal")
summary(fit, process = "Event")
# Prediction of cumulative incidence for competing risks data
# Predict the conditional probabilities for two patients who are alive (censored)
ND <- ydata[ydata$ID %in% c(419, 218), ]
ID <- unique(ND$ID)
NDc <- cdata[cdata$ID  %in% ID, ]
survfit <- survfitjmcs(fit,
                       ynewdata = ND,
                       cnewdata = NDc,
                       u = seq(3, 4.8, by = 0.2),
                       method = "GH",
                       obs.time = "time")
survfit

res <- DynPredAccjmcs(object = fit,
                      landmark.time = 3,
                      horizon.time = c(3.6, 4, 4.4),
                      obs.time = "time",
                      method = "GH",
                      maxiter = 1000,
                      n.cv = 3,
                      metrics = c("AUC", "Cindex", "Brier", "MAE", "MAEQ"))

# Print all available evaluation metrics for the fitted joint model
```

```
summary(res, metric = "Brier")
summary(res, metric = "MAE")
summary(res, metric = "MAEQ")
summary(res, metric = "AUC")
summary(res, metric = "Cindex")
```

`JMMLSM` *Joint Modeling for Continuous outcomes*

Description

Joint modeling of longitudinal continuous data and competing risks

Usage

```
JMMLSM(
  cdata,
  ydata,
  long.formula,
  surv.formula,
  variance.formula,
  random,
  maxiter = 1000,
  epsilon = 1e-04,
  quadpoint = NULL,
  print.para = FALSE,
  survinitial = TRUE,
  initial.para = NULL,
  method = "adaptive",
  opt = "nlminb",
  initial.optimizer = "BFGS"
)
```

Arguments

`cdata`	a survival data frame with competing risks or single failure. Each subject has one data entry.
`ydata`	a longitudinal data frame in long format.
`long.formula`	a formula object with the response variable and fixed effects covariates to be included in the longitudinal sub-model.
`surv.formula`	a formula object with the survival time, event indicator, and the covariates to be included in the survival sub-model.
`variance.formula`	an one-sided formula object with the fixed effects covariates to model the variance of longitudinal sub-model.

`random`	a one-sided formula object describing the random effects part of the longitudinal sub-model. For example, fitting a random intercept model takes the form ~ 1|ID. Alternatively. Fitting a random intercept and slope model takes the form ~ x1 + ... + xn|ID.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`epsilon`	Tolerance parameter. Default is 0.0001.
`quadpoint`	the number of Gauss-Hermite quadrature points to be chosen for numerical integration. Default is 15 which produces stable estimates in most dataframes.
`print.para`	Print detailed information of each iteration. Default is FALSE, i.e., not to print the iteration details.
`survinitial`	Fit a Cox model to obtain initial values of the parameter estimates. Default is TRUE.
`initial.para`	a list of initialized parameters for EM iteration. Default is NULL.
`method`	Method for proceeding numerical integration in the E-step. Default is adaptive.
`opt`	Optimization method to fit a linear mixed effects model, either nlminb (default) or optim.
`initial.optimizer`	Method for numerical optimization to be used. Default is `BFGS`.

Value

Object of class `JMMLSM` with elements

`ydata`	the input longitudinal dataset for fitting a joint model. It has been re-ordered in accordance with descending observation times in `cdata`.
`cdata`	the input survival dataset for fitting a joint model. It has been re-ordered in accordance with descending observation times.
`PropEventType`	a frequency table of number of events.
`beta`	the vector of fixed effects for the mean trajectory in the mixed effects location and scale model.
`tau`	the vector of fixed effects for the within-subject variability in the mixed effects location and scale model.
`gamma1`	the vector of fixed effects for type 1 failure for the survival model.
`gamma2`	the vector of fixed effects for type 2 failure for the survival model. Valid only if `CompetingRisk = TRUE`.
`alpha1`	the vector of association parameter(s) for the mean trajectory for type 1 failure.
`alpha2`	the vector of association parameter(s) for the mean trajectory for type 2 failure. Valid only if `CompetingRisk = TRUE`.
`vee1`	the vector of association parameter(s) for the within-subject variability for type 1 failure.
`vee2`	the vector of association parameter(s) for the within-subject variability for type 2 failure. Valid only if `CompetingRisk = TRUE`.

`H01`	the matrix that collects baseline hazards evaluated at each uncensored event time for type 1 failure. The first column denotes uncensored event times, the second column the number of events, and the third columns the hazards obtained by Breslow estimator.
`H02`	the matrix that collects baseline hazards evaluated at each uncensored event time for type 2 failure. The data structure is the same as `H01`. Valid only if `CompetingRisk = TRUE`.
`Sig`	the variance-covariance matrix of the random effects.
`iter`	the total number of iterations until convergence.
`convergence`	convergence identifier: 1 corresponds to successful convergence, whereas 0 to a problem (i.e., when 0, usually more iterations are required).
`vcov`	the variance-covariance matrix of all the fixed effects for both models.
`sebeta`	the standard error of `beta`.
`setau`	the standard error of `tau`.
`segamma1`	the standard error of `gamma1`.
`segamma2`	the standard error of `gamma2`. Valid only if `CompetingRisk = TRUE`.
`sealpha1`	the standard error of `alpha1`.
`sealpha2`	the standard error of `alpha2`. Valid only if `CompetingRisk = TRUE`.
`sevee1`	the standard error of `vee1`.
`sevee2`	the standard error of `vee2`. Valid only if `CompetingRisk = TRUE`.
`seSig`	the vector of standard errors of covariance of random effects.
`loglike`	the log-likelihood value.
`EFuntheta`	a list with the expected values of all the functions of random effects.
`CompetingRisk`	logical value; TRUE if a competing event are accounted for.
`quadpoint`	the number of Gauss Hermite quadrature points used for numerical integration.
`LongitudinalSubmodelmean`	the component of the `long.formula`.
`LongitudinalSubmodelvariance`	the component of the `variance.formula`.
`SurvivalSubmodel`	the component of the `surv.formula`.
`random`	the component of the `random`.
`call`	the matched call.

Examples

```
require(FastJM)
data(ydata)
data(cdata)
## fit a joint model
## Not run:
fit <- JMMLSM(cdata = cdata, ydata = ydata,
```

```
              long.formula = Y ~ Z1 + Z2 + Z3 + time,
              surv.formula = Surv(survtime, cmprsk) ~ var1 + var2 + var3,
              variance.formula = ~ Z1 + Z2 + Z3 + time,
              quadpoint = 6, random = ~ 1|ID, print.para = FALSE)

## make dynamic prediction of two subjects
cnewdata <- cdata[cdata$ID %in% c(122, 152), ]
ynewdata <- ydata[ydata$ID %in% c(122, 152), ]
survfit <- survfitJMMLSM(fit, seed = 100, ynewdata = ynewdata, cnewdata = cnewdata,
                         u = seq(5.2, 7.2, by = 0.5), Last.time = "survtime",
                         obs.time = "time", method = "GH")
oldpar <- par(mfrow = c(2, 2), mar = c(5, 4, 4, 4))
plot(survfit, include.y = TRUE)
par(oldpar)

## End(Not run)
```

`MAEQJMMLSM` *A metric of prediction accuracy of joint model by comparing the predicted risk with the empirical risks stratified on different predicted risk group.*

Description

A metric of prediction accuracy of joint model by comparing the predicted risk with the empirical risks stratified on different predicted risk group.

Usage

```
MAEQJMMLSM(
  seed = 100,
  object,
  landmark.time = NULL,
  horizon.time = NULL,
  obs.time = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  maxiter = 1000,
  survinitial = TRUE,
  n.cv = 3,
  quantile.width = 0.25,
  opt = "nlminb",
  initial.para = FALSE,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  ...
)
```

Arguments

`seed`	a numeric value of seed to be specified for cross validation.
`object`	object of class 'JMMLSM'.
`landmark.time`	a numeric value of time for which dynamic prediction starts..
`horizon.time`	a numeric vector of future times for which predicted probabilities are to be computed.
`obs.time`	a character string of specifying a longitudinal time variable.
`method`	estimation method for predicted probabilities. If `Laplace`, then the empirical empirical estimates of random effects is used. If `GH`, then the standard Gauss-Hermite quadrature is used.
`quadpoint`	the number of standard Gauss-Hermite quadrature points if `method = "GH"`.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.
`survinitial`	Fit a Cox model to obtain initial values of the parameter estimates. Default is TRUE.
`n.cv`	number of folds for cross validation. Default is 3.
`quantile.width`	a numeric value of width of quantile to be specified. Default is 0.25.
`opt`	Optimization method to fit a linear mixed effects model, either nlminb (default) or optim.
`initial.para`	Initial guess of parameters for cross validation. Default is FALSE.
`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`JMMLSM`, `survfitJMMLSM`

`mvcdata` *Simulated competing risks data correlated with mvydata*

Description

The `mvcdata` data frame has 500 rows and 5 columns.

Usage

```
data(mvcdata)
```

Format

This data frame contains the following columns:

`ID`	patient identifier.
`survtime`	event time.
`cmprsk`	event indicator. `0` denotes censoring, `1` risk 1, and `2` risk 2.
`X21`	X21.
`X22`	X22.

`mvjmcs` *Joint modeling of multivariate longitudinal and competing risks data*

Description

Joint modeling of multivariate longitudinal continuous data and competing risks

Usage

```
mvjmcs(
  ydata,
  cdata,
  long.formula,
  random = NULL,
  surv.formula,
  maxiter = 10000,
  opt = "nlminb",
  tol = 0.005,
  print.para = TRUE,
  initial.para = NULL,
  cpu.cores = NULL
)
```

Arguments

`ydata`	A longitudinal data frame in long format.
`cdata`	A survival data frame with competing risks or single failure. Each subject has one data entry.
`long.formula`	A list of formula objects specifying fixed effects for each longitudinal outcome.
`random`	A formula or list of formulas describing random effects structures (e.g., `~ 1|ID`).
`surv.formula`	A formula for the survival sub-model, including survival time and event indicator.
`maxiter`	Maximum number of EM iterations. Default is 10000.
`opt`	Optimization method for mixed model. Default is `"nlminb"`.
`tol`	Convergence tolerance for EM algorithm. Default is 0.0001.
`print.para`	Logical; if `TRUE`, prints parameter values at each iteration.
`initial.para`	Optional list of initialized parameters. Default is `NULL`.
`cpu.cores`	Number of CPU cores for parallel computation. Default is 1.

Details

Function fits a joint model for multiple longitudinal outcomes and competing risks using a fast EM algorithm.

Value

Object of class `mvjmcs` with elements

`beta`	the vector of all biomarker-specific fixed effects for the linear mixed effects sub-models.
`betaList`	the list of biomarker-specific fixed effects for the linear mixed effects sub-model.
`gamma1`	the vector of fixed effects for type 1 failure for the survival model.
`gamma2`	the vector of fixed effects for type 2 failure for the survival model. Valid only if `CompetingRisk = TRUE`.
`alpha1`	the vector of association parameter(s) for type 1 failure.
`alpha2`	the vector of association parameter(s) for type 2 failure. Valid only if `CompetingRisk = TRUE`.
`H01`	the matrix that collects baseline hazards evaluated at each uncensored event time for type 1 failure. The first column denotes uncensored event times, the second column the number of events, and the third columns the hazards obtained by Breslow estimator.
`H02`	the matrix that collects baseline hazards evaluated at each uncensored event time for type 2 failure. The data structure is the same as `H01`. Valid only if `CompetingRisk = TRUE`.
`Sig`	the variance-covariance matrix of the random effects.
`sigma`	the vector of the variance of the biomarker-specific measurement error for the linear mixed effects sub-models.

`iter`	the total number of iterations until convergence.
`convergence`	convergence identifier: 1 corresponds to successful convergence, whereas 0 to a problem (i.e., when 0, usually more iterations are required).
`vcov`	the variance-covariance matrix of all the fixed effects for both models.
`FisherInfo`	the Empirical Fisher information matrix.
`Score`	a matrix of the score function for all subjects.
`sebeta`	the standard error of `beta`.
`segamma1`	the standard error of `gamma1`.
`segamma2`	the standard error of `gamma2`. Valid only if `CompetingRisk = TRUE`.
`sealpha1`	the standard error of `nu1`.
`sealpha2`	the standard error of `nu2`. Valid only if `CompetingRisk = TRUE`.
`seSig`	the vector of standard errors of covariance of random effects.
`sesigma`	the standard error of variance of biomarker-specific measurement error for the linear mixed effects sub-models.
`pos.mode`	the posterior mode of the conditional distribution of random effects.
`pos.cov`	the posterior covariance of the conditional distribution of random effects.
`CompetingRisk`	logical value; TRUE if a competing event are accounted for.
`ydata`	the input longitudinal dataset for fitting a joint model. It has been re-ordered in accordance with descending observation times in `cdata`.
`cdata`	the input survival dataset for fitting a joint model. It has been re-ordered in accordance with descending observation times.
`PropEventType`	a frequency table of number of events.
`LongitudinalSubmodel`	the component of the `long.formula`.
`SurvivalSubmodel`	the component of the `surv.formula`.
`random`	the component of the `random`.
`call`	the matched call.
`id`	the grouping vector for the longitudinal outcome.
`opt`	the numerical optimizer for obtaining the initial guess of the parameters in the linear mixed effects sub-models.
`runtime`	the total computation time.

Examples

```
require(FastJM)
require(survival)
require(future)
require(future.apply)

data(mvcdata)
```

```
data(mvydata)


# Fit joint model with two biomarkers
fit <- mvjmcs(ydata = mvydata, cdata = mvcdata,
              long.formula = list(Y1 ~ X11 + X12 + time,
                                  Y2 ~ X11 + X12 + time),
              random = list(~ time | ID,
                            ~ 1 | ID),
         surv.formula = Surv(survtime, cmprsk) ~ X21 + X22, maxiter = 1000, opt = "optim",
              tol = 1e-3, print.para = FALSE)
fit

# Extract the parameter estimates of longitudinal sub-model fixed effects
fixef(fit, process = "Longitudinal")

# Extract the parameter estimates of survival sub-model fixed effects
fixef(fit, process = "Event")

# Obtain the random effects estimates for first 6 subjects
head(ranef(fit))
```

`mvydata` *Simulated bivariate longitudinal data*

Description

The `mvydata` data frame has 4060 rows and 6 columns.

Usage

`data(mvydata)`

Format

This data frame contains the following columns:

`ID` patient identifier.

`time` visit time.

`Y1` response variable of biomarker 1.

`Y2` response variable of biomarker 2.

`X11` X11.

`X12` X12.

`PEJMMLSM` *A metric of prediction accuracy of joint model by comparing the predicted risk with the counting process.*

Description

A metric of prediction accuracy of joint model by comparing the predicted risk with the counting process.

Usage

```
PEJMMLSM(
  seed = 100,
  object,
  landmark.time = NULL,
  horizon.time = NULL,
  obs.time = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  maxiter = 1000,
  n.cv = 3,
  survinitial = TRUE,
  opt = "nlminb",
  initial.para = FALSE,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  ...
)
```

Arguments

`seed`	a numeric value of seed to be specified for cross validation.
`object`	object of class 'JMMLSM'.
`landmark.time`	a numeric value of time for which dynamic prediction starts..
`horizon.time`	a numeric vector of future times for which predicted probabilities are to be computed.
`obs.time`	a character string of specifying a longitudinal time variable.
`method`	estimation method for predicted probabilities. If `Laplace`, then the empirical empirical estimates of random effects is used. If `GH`, then the pseudo-adaptive Gauss-Hermite quadrature is used.
`quadpoint`	the number of pseudo-adaptive Gauss-Hermite quadrature points if `method = "GH"`.
`maxiter`	the maximum number of iterations of the EM algorithm that the function will perform. Default is 10000.

`n.cv`	number of folds for cross validation. Default is 3.
`survinitial`	Fit a Cox model to obtain initial values of the parameter estimates. Default is TRUE.
`opt`	Optimization method to fit a linear mixed effects model, either nlminb (default) or optim.
`initial.para`	Initial guess of parameters for cross validation. Default is FALSE.
`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`JMMLSM`, `survfitJMMLSM`

`plot.jmcs` *Fitted values for joint models*

Description

Plot Diagnostics for Joint Models.

Usage

```
## S3 method for class 'jmcs'
plot(x, add.smooth = getOption("add.smooth"), ...)
```

Arguments

`x`	x of class 'jmcs'.
`add.smooth`	logical; if `TRUE` a smooth line is superimposed in the "Residuals vs Fitted" plot.
`...`	further arguments passed to or from other methods.

Value

The first two plots are longitudinal sub-model diagnostics and the last two are marginal survival function and marginal cumulative hazard.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

Examples

```
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)

oldpar <- par(mfrow = c(2, 2))
plot(fit)
par(oldpar)
```

`plot.survfitJMMLSM` *Plot conditional probabilities for new subjects*

Description

Plot conditional probabilities for new subjects. If `CompetingRisk` = `FALSE`, print the survival probabilities. Otherwise, print the cumulative incidence probabilities for each failure type.

Usage

```
## S3 method for class 'survfitJMMLSM'
plot(
  x,
  include.y = FALSE,
  xlab = NULL,
  ylab = NULL,
  xlim = NULL,
  ylim.long = NULL,
  ylim.surv = NULL,
  ...
)
```

Arguments

`x`	x of class `survfitJMMLSM`.
`include.y`	include longitudinal responses of this subject versus time. Default is FALSE.
`xlab`	X axis label.
`ylab`	Y axis label.
`xlim`	X axis support.
`ylim.long`	Y axis support for the longitudinal outcome.
`ylim.surv`	Y axis support for the event / survival probability.
`...`	further arguments passed to or from other methods.

Value

plots of conditional probabilities over different pre-specified time points for subjects. If single failure type, then survival probabilities will be returned. Otherwise, cumulative incidence probabilities for each failure type will be returned.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`survfitJMMLSM`

`print` *Print jmcs*

Description

Print jmcs

Print mvjmcs

Usage

```
## S3 method for class 'jmcs'
print(x, digits = 4, ...)

## S3 method for class 'mvjmcs'
print(x, digits = 4, ...)
```

Arguments

`x`	Object of class ’mvjmcs’.
`digits`	the number of significant digits to use when printing.
`...`	Further arguments passed to or from other methods.

Value

a summary of data, joint model, log likelihood, and parameter estimates.

a summary of data, joint model, log likelihood, and parameter estimates.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`jmcs`

`mvjmcs`

`print.JMMLSM` *Print JMMLSM*

Description

Print contents of JMMLSM object.

Usage

```
## S3 method for class 'JMMLSM'
print(x, digits = 4, ...)
```

Arguments

`x`	Object of class 'JMMLSM'.
`digits`	number of digits of decimal to be printed.
`...`	Further arguments passed to or from other methods.

Value

a summary of data, joint model, log likelihood, and parameter estimates.

Author(s)

Shanpeng Li

See Also

`JMMLSM`

`print.survfitjmcs` *Print survfitjmcs*

Description

Print survfitjmcs

Usage

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

Arguments

`x`	x of class 'survfitjmcs'.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`jmcs`, `survfitjmcs`

`print.survfitJMMLSM` *Print survfitJMMLSM*

Description

Print survfitJMMLSM

Usage

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

Arguments

`x`	x of class 'survfitJMMLSM'.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`JMMLSM`, `survfitJMMLSM`

`print.survfitmvjmcs` *Print survfitmvjmcs*

Description

Print survfitmvjmcs

Usage

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

Arguments

`x`	x of class 'survfitmvjmcs'.
`...`	Further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`mvjmcs`, `survfitmvjmcs`

`ranef` *Random effects estimates for joint models*

Description

Extracts the posterior mean of the random effects for a fitted joint model.

Usage

```
ranef(object, ...)
```

Arguments

`object`	an object inheriting from class `jmcs` or `mvjmcs`.
`...`	further arguments passed to or from other methods.

Value

a matrix of random effects estimates.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`jmcs`

Examples

```
# a joint model fit
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)

# extract random effects estimates
head(ranef(fit))
```

`residuals` *Residuals for joint models*

Description

Extract residuals for joint models.

Usage

```
## S3 method for class 'jmcs'
residuals(object, type = c("Marginal", "Subject"), ...)
```

Arguments

`object`	an object inheriting from class `jmcs`.
`type`	what type of residuals to calculate.
`...`	further arguments passed to or from other methods.

Value

a vector of residuals of the longitudinal sub-model.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`jmcs`

Examples

```
# a joint model fit
fit <- jmcs(ydata = ydata, cdata = cdata,
            long.formula = response ~ time + gender + x1 + race,
            surv.formula = Surv(surv, failure_type) ~ x1 + gender + x2 + race,
            random =  ~ time| ID)

# residuals of the longitudinal sub-model
head(cbind(
  "Marg" = residuals(fit, type = "Marginal"),
  "Subj" = residuals(fit, type = "Subject")
))
```

`simmvJMdata` *Joint modeling of multivariate longitudinal and competing risks data*

Description

Data simulation from the joint model of multivariate longitudinal biomarkers and time-to-event data

Usage

```
simmvJMdata(
  seed = 100,
  N = 200,
  increment = 0.7,
  beta = list(beta1 = c(5, 1.5, 2, 1), beta2 = c(10, 1, 2, 1)),
  sigma = c(0.5, 0.5),
  gamma1 = c(1, 0.5),
  gamma2 = c(-0.5, 0.5),
  alpha1 = list(alpha11 = c(0.5, 0.7), alpha12 = c(-0.5, 0.5)),
  alpha2 = list(alpha21 = c(0.5, 0.7), alpha22 = c(-0.5, 0.5)),
  lambda1 = 0.05,
  lambda2 = 0.025,
  CL = 5,
  CU = 10,
  covb = diag(rep(1, 4)),
  missprob = 0,
  CR = TRUE
)
```

Arguments

`seed`	a random seed number specified for simulating a joint model dataset.
`N`	an integer to specify the sample size.
`increment`	a scalar to specify the increment of visit time for longitudinal measurements.
`beta`	a list of true parameters for the linear mixed effects sub-models. Each component of the list correspond to a specific biomarker.
`sigma`	a vector of true error variance for all biomarkers.
`gamma1`	a vector of true parameters of survival fixed effects for failure 1.
`gamma2`	a vector of true parameters of survival fixed effects for failure 2.
`alpha1`	a list of true parameters for the association parameters for failure 1. Each component of the list correspond to a specific biomarker.
`alpha2`	a list of true parameters for the association parameters for failure 2. Each component of the list correspond to a specific biomarker.
`lambda1`	the baseline hazard rate of failure 1. An exponential distribution with a rate parameter of `lambda1` is assumed.

`lambda2`	the baseline hazard rate of failure 2. An exponential distribution with a rate parameter of `lambda2` is assumed.
`CL`	a lower limit of a uniform distribution to be specified for the censoring time.
`CU`	an upper limit of a uniform distribution to be specified for the censoring time.
`covb`	a matrix of variance-covariance matrix of random effects.
`missprob`	a scalar (ranging from 0 to 1) to specify the probability of missing longitudinal observations. Default is 0.
`CR`	logical; if `TRUE`, simulate competing risks time-to-event data with 2 failures. Default is `TRUE`.

Value

a list of datasets for both longitudinal and survival data with the elements

`mvydata`	a long-format data frame of longitudinal data.
`mvcdata`	a dataframe of survival data.

`summary` *Summaries of evaluation metrics for joint models*

Description

Produce result summaries of a joint model fit.

Produce result summaries of a joint model fit.

Produce result summaries of a joint model fit.

Usage

```
## S3 method for class 'AUCJMMLSM'
summary(object, digits = 4, ...)

## S3 method for class 'ConcordanceJMMLSM'
summary(object, digits = 4, ...)

## S3 method for class 'Concordancejmcs'
summary(object, digits = 4, ...)

## S3 method for class 'DynPredAccjmcs'
summary(
  object,
  metric = c("AUC", "Cindex", "Brier", "MAE", "MAEQ"),
  digits = 4,
  ...
)
```

```
## S3 method for class 'JMMLSM'
summary(object, process = c("longitudinal", "survival"), digits = 4, ...)

## S3 method for class 'MAEQJMMLSM'
summary(object, digits = 3, ...)

## S3 method for class 'PEJMMLSM'
summary(object, error = c("MAE", "Brier"), ...)

## S3 method for class 'jmcs'
summary(object, process = c("Longitudinal", "Event"), digits = 4, ...)

## S3 method for class 'mvjmcs'
summary(object, process = c("Longitudinal", "Event"), digits = 4, ...)
```

Arguments

`object`	an object inheriting from class `mvjmcs`.
`digits`	the number of significant digits to use when printing. Default is 4.
`...`	further arguments passed to or from other methods.
`metric`	a list to indicate what metric to summarize
`process`	for which model (i.e., longitudinal model or survival model) to extract the estimated coefficients.
`error`	a character string that specifies the loss function.

Value

a list of matrices with conditional probabilities for subjects.

a list of matrices with conditional probabilities for subjects.

a list of matrices with conditional probabilities for subjects.

a summary of the list of matrices with conditional probabilities for subjects.

A table to summarize the model results.

a list of matrices with conditional probabilities for subjects.

a list of matrices with conditional probabilities for subjects.

A table to summarize the model results.

A table to summarize the model results.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`JMMLSM`, `survfitJMMLSM`

`JMMLSM`, `survfitJMMLSM`

`jmcs`, `survfitjmcs`

`jmcs`, `survfitjmcs`

`JMMLSM`

`JMMLSM`, `survfitJMMLSM`

`JMMLSM`, `survfitJMMLSM`

`jmcs`

`mvjmcs`

`survfitjmcs` *Prediction in Joint Models*

Description

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

Usage

```
survfitjmcs(
  object,
  seed = 100,
  ynewdata = NULL,
  cnewdata = NULL,
  u = NULL,
  Last.time = NULL,
  obs.time = NULL,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  ...
)
```

Arguments

`object`	an object inheriting from class `jmcs`.
`seed`	a random seed number to proceed Monte Carlo simulation. Default is 100.
`ynewdata`	a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required.

`cnewdata`	a data frame that contains the survival and covariate information for the subjects for which prediction of survival probabilities is required.
`u`	a numeric vector of times for which prediction survival probabilities are to be computed.
`Last.time`	a numeric vector or character string. This specifies the known time at which each of the subjects in cnewdata was known to be alive. If NULL, then this is automatically taken as the survival time of each subject. If a numeric vector, then it is assumed to be greater than or equals to the last available longitudinal time point for each subject. If a character string, then it should be a variable in cnewdata.
`obs.time`	a character string of specifying a longitudinal time variable in ynewdata.
`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`method`	a character string specifying the type of probability approximation; if `Laplace`, then a first order estimator is computed. If `GH`, then the standard Gauss-Hermite quadrature is used instead.
`quadpoint`	number of quadrature points used for estimating conditional probabilities when `method = "GH"`. Default is NULL. If `method = "GH"`, then use the same amount of quadrature points obtained from `object`.
`...`	further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`jmcs`

`survfitJMMLSM` *Prediction in Joint Models*

Description

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

Usage

```
survfitJMMLSM(
  object,
  seed = 100,
  ynewdata = NULL,
  cnewdata = NULL,
  u = NULL,
  Last.time = NULL,
  obs.time = NULL,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  ...
)
```

Arguments

`object`	an object inheriting from class `JMMLSM`.
`seed`	a random seed number to proceed non-parametric bootstrap. Default is 100.
`ynewdata`	a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required.
`cnewdata`	a data frame that contains the survival and covariate information for the subjects for which prediction of survival probabilities is required.
`u`	a numeric vector of times for which prediction survival probabilities are to be computed.
`Last.time`	a numeric vector or character string. This specifies the known time at which each of the subjects in cnewdata was known to be alive. If NULL, then this is automatically taken as the survival time of each subject. If a numeric vector, then it is assumed to be greater than or equals to the last available longitudinal time point for each subject. If a character string, then it should be a variable in cnewdata.
`obs.time`	a character string of specifying a longitudinal time variable in ynewdata.

`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`method`	a character string specifying the type of probability approximation; if `Laplace`, then a first order estimator is computed. If `GH`, then the standard Gauss-Hermite quadrature is used instead.
`quadpoint`	number of quadrature points used for estimating conditional probabilities when `method = "GH"`. Default is NULL. If `method = "GH"`, then 15 is used.
`...`	further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`JMMLSM`

`survfitmvjmcs` *Prediction in Joint Models*

Description

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

Usage

```
survfitmvjmcs(
  object,
  seed = 100,
  ynewdata = NULL,
  cnewdata = NULL,
  u = NULL,
  Last.time = NULL,
  obs.time = NULL,
  LOCF = FALSE,
```

```
  LOCFcovariate = NULL,
  clongdata = NULL,
  ...
)
```

Arguments

`object`	an object inheriting from class `mvjmcs`.
`seed`	a random seed number to proceed Monte Carlo simulation. Default is 100.
`ynewdata`	a data frame that contains the longitudinal and covariate information for the subjects for which prediction of survival probabilities is required.
`cnewdata`	a data frame that contains the survival and covariate information for the subjects for which prediction of survival probabilities is required.
`u`	a numeric vector of times for which prediction survival probabilities are to be computed.
`Last.time`	a numeric vector or character string. This specifies the known time at which each of the subjects in cnewdata was known to be alive. If NULL, then this is automatically taken as the survival time of each subject. If a numeric vector, then it is assumed to be greater than or equals to the last available longitudinal time point for each subject. If a character string, then it should be a variable in cnewdata.
`obs.time`	a character string of specifying a longitudinal time variable in ynewdata.
`LOCF`	a logical value to indicate whether the last-observation-carried-forward approach applies to prediction. If `TRUE`, then `LOCFcovariate` and `clongdata` must be specified to indicate which time-dependent survival covariates are included for dynamic prediction. Default is FALSE.
`LOCFcovariate`	a vector of string with time-dependent survival covariates if `LOCF = TRUE`. Default is NULL.
`clongdata`	a long format data frame where time-dependent survival covariates are incorporated. Default is NULL.
`...`	further arguments passed to or from other methods.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`mvjmcs`

`vcov` *Variance-covariance matrix of the estimated parameters for joint models*

Description

Extract variance-covariance matrix for joint models.

Extract variance-covariance matrix for joint models.

Usage

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

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

Arguments

`object`	an object inheriting from class `jmcs`.
`...`	further arguments passed to or from other methods.

Value

a matrix of variance covariance of all parameter estimates.

a matrix of variance covariance of all parameter estimates.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`JMMLSM`

`jmcs`

`ydata` *Simulated longitudinal data*

Description

The `ydata` data frame has 3067 rows and 6 columns.

Usage

```
data(ydata)
```

Format

This data frame contains the following columns:

`ID` patient identifier.

`response` response variable.

`time` visit time.

`x1` treatment indicator. `0` denotes the placebo group and `1` the treatment group.

`gender` gender indicator.

`race` race indicator.

`ydatah` *Simulated longitudinal data with within-subject variance*

Description

The `ydatah` data frame has 1353 rows and 6 columns.

Usage

```
data(ydatah)
```

Format

This data frame contains the following columns:

`ID` patient identifier.

`Y` response variable.

`time` visit time.

`Z1` treatment indicator. `0` denotes the placebo group and `1` the treatment group.

`Z2` continuous variable..

`Z3` continuous variable..

Index