Package: FastJM (via r-universe)

July 22, 2026

Type Package

Title Semi-Parametric Joint Modeling of Longitudinal and Survival Data

Version 1.7.0

Date 2026-07-20

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, tidycmprsk, rlang (>= 0.4.11), ggsurvfit, ggplot2, ggpubr

LinkingTo Rcpp, RcppEigen

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

RoxygenNote 7.3.2

LazyData true

Suggests testthat (>= 3.0.0), spelling, rmarkdown

Language en-US

Config/testthat/edition 3

Author Shanpeng Li [aut, cre], Ace Mejia-Sanchez [ctb], Emily Ouyang [ctb], Gang Li [ctb]

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

Date/Publication 2026-07-21 21:00:09 UTC

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

RemoteRef HEAD

RemoteSha 20068cee5c2363802044c4f0f8fa66a0a03ebe67

Contents

`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.

`combine_biomarkers` *Combine biomarker measurements across multiple data frames*

Description

Combines specified biomarker measurements across a list of data frames and returns a unified long-format data set restricted to subjects with at least one recorded measurement for EVERY requested biomarker.

The function searches each provided data frame for the biomarker names listed in `biomarkers`, extracts the matching measurement rows, and standardizes the output into a common structure. Subjects are retained only if they have at least one non-missing measurement for every requested biomarker across the supplied data frames. It is intended for settings where biomarker measurements may be distributed across multiple data frames, and where each data frame may contain one or more of the requested biomarkers. The returned data are in long format so that repeated measurements per subject and per biomarker are preserved. No collapsing, averaging, or other within-subject summarization is performed.

Usage

```
combine_biomarkers(
  data_list,
  biomarkers,
  id_col,
  time_col = NULL,
  dataset_names = NULL
)
```

Arguments

`data_list`	A non-empty list of data frames containing subject-level biomarker measurements.
`biomarkers`	A character vector of biomarker column names to search for across the supplied data frames.
`id_col`	A character string giving the subject ID column name. This column must be present in every data frame in `data_list`.

`time_col` An optional character string giving the time variable column name, such as days from baseline. If supplied, this column must be present in every data frame in `data_list`, and it is standardized to `"time"` in the returned long-format output.

`dataset_names` An optional character vector of names for the supplied data frames. If omitted, the function uses `names(data_list)` when available; otherwise it creates default names of the form `"dataset_1"`, `"dataset_2"`, and so on.

Value

A list with the following components:

`combined_long` A long-format data frame containing only included subjects. The output includes the subject ID column, a `dataset` column identifying the source data frame, a `biomarker` column identifying the biomarker name, a standardized `value` column containing the measurement values, and a standardized `time` column if `time_col` was supplied.

`included_ids` A vector of subject IDs retained in the final combined data. These are the subjects with at least one non-missing measurement for every requested biomarker.

`ids_by_biomarker` A named list giving the subject IDs with at least one non-missing measurement for each biomarker.

`presence_summary` A data frame summarizing biomarker presence among the included subjects, with one row per subject and one logical column per biomarker.

Author(s)

Shanpeng Li <lishanpeng0913@ucla.edu>

Examples

```
df_a <- data.frame(
  ID = c(1, 1, 2, 3, 4, 5),
  day = c(0, 30, 0, 0, 0, 0),
  sbp = c(120, 125, 130, NA, 110, 118)
)

df_b <- data.frame(
  ID = c(1, 2, 2, 3, 5, 6),
  day = c(0, 0, 20, 0, 0, 0),
  dbp = c(80, 85, 84, 90, NA, 88)
)

df_c <- data.frame(
  ID = c(1, 1, 2, 4, 5, 5),
  day = c(0, 40, 0, 0, 0, 10),
  bpm = c(70, 72, 68, 75, 77, 79)
)

res <- combine_biomarkers(
  data_list = list(df_a, df_b, df_c),
  biomarkers = c("sbp", "dbp", "bpm"),
  id_col = "ID",
```

```
  time_col = "day"
)

res$included_ids
head(res$combined_long)
res$presence_summary
```

`Concordance` *Concordance for joint models*

Description

Concordance for joint models

Usage

```
Concordance(
  seed = 100,
  object,
  n.cv = 3,
  maxiter = 10000,
  initial.para = TRUE,
  ...
)
```

Arguments

`seed`	A numeric value used to set the random seed for cross-validation. Default is `100`.
`object`	A fitted object of class `jmcs`, `JMMLSM`, or `mvjmcs`.
`n.cv`	Number of cross-validation folds. Default is `3`.
`maxiter`	Maximum number of EM iterations allowed when refitting the model within each cross-validation fold. Default is `10000`.
`initial.para`	Logical value indicating whether initial parameter values are used when refitting the model. Default is `TRUE`.
`...`	Further arguments passed to model-specific methods.

Value

A list containing:

`n.cv` The number of cross-validation folds.

`Concordance.cv` A list of fold-specific concordance estimates.

`PI.cv` A list of fold-specific Cindex value for the estimated time-independent prognostic indexes across subjects.

`CompetingRisk` Logical value indicating whether the fitted model accounts for competing risks.

`seed` The random seed used for cross-validation.

See Also

jmcs, JMMLSM

DynPredAcc *Dynamic prediction accuracy metrics for joint models*

Description

Dynamic prediction accuracy metrics for fitted joint models

Usage

```
DynPredAcc(
  seed = 100,
  object,
  landmark.time = NULL,
  horizon.time = NULL,
  obs.time = NULL,
  method = c("Laplace", "GH"),
  quadpoint = NULL,
  maxiter = NULL,
  n.cv = 3,
  quantile.width = 0.25,
  opt = c("nlminb", "optim"),
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  metrics = c("AUC", "Cindex", "Brier Score", "MAE", "MAEQ"),
  cpu.cores = 1,
  ...
)
```

Arguments

seed	A numeric value used to set the random seed for cross-validation. Default is 100.
object	A fitted object of class jmcs, JMMLSM, or mvjmcs.
landmark.time	A numeric value specifying the landmark time at which dynamic prediction begins.
horizon.time	A numeric vector of future times at which predicted probabilities are evaluated.
obs.time	A character string specifying the longitudinal time variable in the longitudinal data.
method	A character string specifying the approximation method used for dynamic prediction. Available options are "Laplace" and "GH". This argument is not used for objects of class mvjmcs.

quadpoint	Number of Gauss–Hermite quadrature points used when method = "GH". If NULL, the value stored in object is used.
maxiter	Maximum number of EM iterations allowed when refitting the model within each cross-validation fold. If NULL, the model-specific default is used.
n.cv	Number of cross-validation folds. Default is 3.
quantile.width	Numeric value specifying the width of the quantile groups used for "MAEQ" summaries. Default is 0.25. The reciprocal of quantile.width must be an integer.
opt	Optimization method used when refitting the model in each fold. Available options are "nlminb" and "optim".
LOCF	Logical value indicating whether the last-observation-carried- forward approach is applied for time-dependent survival covariates during prediction. Default is FALSE.
LOCFcovariate	A character vector specifying the time-dependent survival covariates to be updated by last observation carried forward when LOCF = TRUE. Default is NULL.
clongdata	A long-format data frame containing time-dependent survival covariates used when LOCF = TRUE. Default is NULL.
metrics	A character vector specifying which prediction accuracy metrics to compute. Available options are "AUC", "Cindex", "Brier Score", "MAE", and "MAEQ". Default is c("AUC", "Cindex", "Brier Score", "MAE", "MAEQ").
cpu.cores	a number of cpu cores used for parallel computing.
...	Further arguments passed to model-specific methods.

Details

Computes dynamic prediction accuracy measures for fitted joint models of class jmcs, JMMLSM, or mvjmcs using grouped cross-validation.

At each cross-validation fold, the function refits the joint model on the training set, obtains subject-specific dynamic predictions for validation subjects at the requested horizon times, and evaluates the requested prediction accuracy metrics.

Supported metrics include:

"AUC" Time-dependent area under the ROC curve.

"Cindex" Concordance index for dynamic predictions.

"Brier Score" Inverse-probability-of-censoring weighted Brier score.

"MAE" Inverse-probability-of-censoring weighted mean absolute error.

"MAEQ" Quantile-based calibration summaries comparing empirical and predicted cumulative incidence rates or survival probabilities.

For competing-risks models, event-specific cumulative incidence predictions are evaluated. For single-event models, conditional survival probabilities are evaluated.

The function performs grouped cross-validation at the subject level. Within each fold, the model is refit on the training data, and dynamic predictions are computed for validation subjects who remain under observation beyond the landmark time and who have longitudinal observations observed up to that time. For single-event models, prediction accuracy is based on conditional survival probabilities. For competing-risks models, prediction accuracy is based on event-specific cumulative incidence functions.

Value

An object of class DynPredAcc, returned as a list containing:

jm.class The class of the fitted joint model.

n.cv The number of cross-validation folds.

landmark.time The landmark time used for dynamic prediction.

horizon.time The vector of horizon times used for evaluation.

method The dynamic prediction approximation method used, if applicable.

quadpoint The number of quadrature points used, if applicable.

CompetingRisk Logical value indicating whether the fitted model accounts for competing risks.

seed The random seed used for cross-validation.

metrics The requested evaluation metrics.

quantile.width The width of quantile groups used for quantile-based calibration summaries.

AUC.cv A list of fold-specific time-dependent AUC estimates, returned when "AUC" is requested.

Cindex.cv A list of fold-specific concordance index estimates, returned when "Cindex" is requested.

Brier.cv A list of fold-specific inverse-probability-of- censoring weighted Brier scores, returned when "Brier Score" is requested.

MAE.cv A list of fold-specific inverse-probability-of- censoring weighted mean absolute errors, returned when "MAE" is requested.

MAEQ.cv A list of fold-specific quantile-based calibration summaries, returned when "MAEQ" is requested.

See Also

jmcs, JMMLSM, mvjmcs, survfitjmcs, survfitJMMLSM, survfitmvjmcs

---

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, JMMLSM, 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,
  control = jmcs_control()
)
```

Arguments

ydata	A longitudinal data frame in long format.
cdata	A survival data frame with one row per subject, containing the survival time, event indicator, and baseline covariates.
long.formula	A formula object specifying the response variable and fixed-effect covariates in the longitudinal submodel.

`random` A one-sided formula object specifying the random-effects structure in the longitudinal submodel. For example, a random-intercept model can be specified as `~ 1 | ID`, and a random-intercept and random-slope model can be specified as `~ x1 + ... + xn | ID`.

`surv.formula` A formula object specifying the survival time, event indicator, and covariates in the survival submodel.

`control` A list of fitting-control options, typically generated by `jmcs_control()`. Available options include:

- `quadpoint` Number of pseudo-adaptive or standard Gauss–Hermite quadrature points used for numerical integration. The default is `6`.
- `maxiter` Maximum number of EM iterations. The default is `10000`.
- `verbose` Logical value indicating whether to print detailed information at each iteration. The default is `FALSE`.
- `initial.para` Optional list of user-supplied initial parameter values for the EM algorithm. The default is `NULL`.
- `tol` Convergence tolerance for the EM algorithm. The default is `1e-4`.
- `method` Numerical integration method used in the E-step. Available options are `"pseudo-adaptive"` and `"standard"`. The default is `"pseudo-adaptive"`.
- `opt` Optimization method used to fit the initial linear mixed-effects model. Available options are `"nlminb"` and `"optim"`. The default is `"nlminb"`.

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 <- survfitJM(fit,
                     ynewdata = ND,
                     cnewdata = NDc,
                     u = seq(3, 4.8, by = 0.2),
                     method = "GH",
                     obs.time = "time")
survfit

res <- DynPredAcc(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 Score", "MAE", "MAEQ"))

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

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

`jmcs_control` *Control Options for jmcs*

Description

Control options for `jmcs()`

Usage

```
jmcs_control(
  quadpoint = NULL,
  maxiter = 10000,
  verbose = FALSE,
  initial.para = NULL,
  tol = 1e-04,
  method = c("pseudo-adaptive", "standard"),
  opt = c("nlminb", "optim")
)
```

Arguments

`quadpoint`	Number of Gauss–Hermite quadrature points used for numerical integration. The default is `NULL`.
`maxiter`	Maximum number of EM iterations. The default is `10000`.
`verbose`	Logical value indicating whether to print iteration details. The default is `FALSE`.
`initial.para`	Optional list of user-supplied initial parameter values for the EM algorithm. The default is `NULL`.
`tol`	Convergence tolerance. The default is `1e-4`.
`method`	Numerical integration method used in the E-step. Available options are `"pseudo-adaptive"` and `"standard"`. The default is `"pseudo-adaptive"`.
`opt`	Optimization method used to fit the initial linear mixed-effects model. Available options are `"nlminb"` and `"optim"`. The default is `"nlminb"`.

Details

Specifies numerical integration, convergence, initialization, and optimizer settings for `jmcs()`.

Value

A list of control parameters used by `jmcs()`.

`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,
  control = JMMLSM_control()
)
```

Arguments

`cdata`	A survival data frame with one row per subject, containing the survival time, event indicator, and baseline covariates.
`ydata`	A longitudinal data frame in long format.
`long.formula`	A formula object specifying the response variable and fixed-effect covariates in the longitudinal mean submodel.
`surv.formula`	A formula object specifying the survival time, event indicator, and covariates in the survival submodel.
`variance.formula`	A one-sided formula object specifying the fixed-effect covariates in the within-subject variance submodel.
`random`	A one-sided formula object specifying the random-effects structure in the longitudinal submodel. For example, a random-intercept model can be specified as `~ 1 | ID`, and a random-intercept and random-slope model can be specified as `~ x1 + ... + xn | ID`.
`control`	A list of fitting-control options, typically generated by `JMMLSM_control()`. Available options include: `maxiter` Maximum number of EM iterations. The default is `1000`. `tol` Convergence tolerance for the EM algorithm. The default is `1e-4`. `quadpoint` Number of Gauss–Hermite quadrature points used for numerical integration. The default is `15`.

`verbose` Logical value indicating whether to print detailed information at each iteration. The default is `FALSE`.

`initial.para` Optional list of user-supplied initial parameter values for the EM algorithm. The default is `NULL`.

`method` Numerical integration method used in the E-step. Available options are `"adaptive"` and `"standard"`. The default is `"adaptive"`.

`opt` Optimization method used to fit the initial linear mixed-effects model. Available options are `"nlminb"` and `"optim"`. The default is `"nlminb"`.

Details

Fits a joint mean and within-subject variability model for longitudinal continuous outcomes and competing risks or single-failure time-to-event outcomes.

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)

## make dynamic prediction of two subjects
cnewdata <- cdata[cdata$ID %in% c(122, 152), ]
ynewdata <- ydata[ydata$ID %in% c(122, 152), ]
survfit <- survfitJM(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)
```

JMMLSM_control *Control Options for JMMLSM*

Description

Control options for JMMLSM()

Usage

```
JMMLSM_control(
  maxiter = 1000,
  tol = 1e-04,
  quadpoint = NULL,
  verbose = FALSE,
  initial.para = NULL,
  method = c("adaptive", "standard"),
  opt = c("nlminb", "optim")
)
```

Arguments

maxiter	Maximum number of EM iterations. The default is 1000.
tol	Convergence tolerance. The default is 1e-4.
quadpoint	Number of Gauss–Hermite quadrature points. The default is 15.
verbose	Logical value indicating whether to print iteration details. The default is FALSE.
initial.para	Optional list of user-supplied initial parameter values. The default is NULL.
method	Numerical integration method used in the E-step. Available options are "adaptive" and "standard". The default is "adaptive".
opt	Optimization method used to fit the initial linear mixed-effects model. Available options are "nlminb" and "optim". The default is "nlminb".

Details

Specifies numerical integration, convergence, initialization, and optimizer settings for JMMLSM().

Value

A list of control parameters used by JMMLSM().

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 and competing risks data

Usage

```
mvjmcs(
  ydata,
  cdata,
  long.formula,
  random = NULL,
  surv.formula,
  control = mvjmcs_control(),
  latAsso = "sre",
  landmark = FALSE,
  s = NULL,
  ytime = 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. For example, a random-intercept model can be specified as ~ 1 | ID.
surv.formula	A formula object specifying the survival time, event indicator, and covariates in the survival submodel.
control	A list of fitting-control options, typically generated by mvjmcs_control(). Available options include: maxiter Maximum number of EM iterations. The default is 10000. opt Optimization method used to fit the initial mixed-effects models. Available options are "nlminb" and "optim". The default is "nlminb". tol Convergence tolerance for the EM algorithm. The default is 0.005. verbose Logical value indicating whether to print iteration details. The default is FALSE. initial.para Optional list of user-supplied initial parameter values for the EM algorithm. The default is NULL. cpu.cores Number of CPU cores used for parallel computation. The default is NULL.
latAsso	Latent association structure. Options are "sre", "present", and "presentlp". The default is "sre".
landmark	Logical value indicating whether landmarking is used. The default is FALSE.
s	Landmark time. Required when landmark = TRUE and latAsso is "present" or "presentlp".
ytime	Name of the longitudinal time variable. Required when landmarking is used.

Value

An object of class mvjmcs, returned as a list containing:

beta The vector of all biomarker-specific fixed effects for the linear mixed-effects submodels.

betaList The list of biomarker-specific fixed effects for the linear mixed-effects submodels.

gamma1 The vector of fixed effects for type 1 failure in the survival submodel.

gamma2 The vector of fixed effects for type 2 failure in the survival submodel. Valid only if CompetingRisk = TRUE.

alpha1 The vector of association parameters for type 1 failure.

alpha2 The vector of association parameters for type 2 failure. Valid only if CompetingRisk = TRUE.

H01 The matrix of baseline hazards evaluated at uncensored event times for type 1 failure.

H02 The matrix of baseline hazards evaluated at uncensored event times for type 2 failure. Valid only if CompetingRisk = TRUE.

`Sig` The variance-covariance matrix of the random effects.

`sigma` The vector of biomarker-specific measurement-error variances.

`iter` The total number of iterations until convergence.

`convergence` Convergence identifier: `1` indicates successful convergence, whereas `0` indicates a convergence problem, often requiring more iterations.

tol the convergence parameter.

`vcov` The variance-covariance matrix of all fixed-effect parameters.

`FisherInfo` The empirical Fisher information matrix.

`Score` A matrix of subject-specific score contributions.

`sebeta` The standard errors of `beta`.

`segamma1` The standard errors of `gamma1`.

`segamma2` The standard errors of `gamma2`. Valid only if `CompetingRisk = TRUE`.

`sealpha1` The standard errors of `alpha1`.

`sealpha2` The standard errors of `alpha2`. Valid only if `CompetingRisk = TRUE`.

`seSig` The vector of standard errors for covariance parameters of the random effects.

`sesigma` The standard errors of biomarker-specific measurement-error variances.

`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 indicating whether competing events are modeled.

`ydata` The input longitudinal dataset, reordered according to descending observation times in `cdata`.

`cdata` The input survival dataset, reordered according to descending observation times.

`PropEventType` A frequency table of event types.

`LongitudinalSubmodel` The longitudinal submodel specified by `long.formula`.

`SurvivalSubmodel` The survival submodel specified by `surv.formula`.

`random` The random-effects specification.

`control` The fitting-control options used.

`call` The matched function call.

`id` The grouping vector for the longitudinal outcomes.

`runtime` The total computation time.

`latAsso` The pre-specified latent association structure.

`landmark` The logical value indicating whether landmarking is used.

`s` The pre-specified landmark time.

`ytime` The name of the longitudinal time variable.

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)
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))

set.seed(08252025)
sampleID <- sample(mvcdata$ID, 2, replace = FALSE)
subcdata <- mvcdata %>%
  dplyr::filter(ID %in% sampleID)
subydata <- mvydata %>%
  dplyr::filter(ID %in% sampleID)
# Make predictions at the horizon times
survfit.mv <- survfitJM(fit, seed = 100, ynewdata = subydata, cnewdata = subcdata,
                        u = c(7, 8, 9), obs.time = "time")
survfit.mv
```

`mvjmcs_control` *Control Options for mvjmcs*

Description

Control options for `mvjmcs()`

Usage

```
mvjmcs_control(
  maxiter = 10000,
```

```
  opt = c("nlminb", "optim"),
  tol = 0.005,
  verbose = FALSE,
  initial.para = NULL,
  cpu.cores = NULL
)
```

Arguments

maxiter	Maximum number of EM iterations. The default is 10000.
opt	Optimization method used to fit the initial mixed-effects models. Available options are "nlminb" and "optim". The default is "nlminb".
tol	Convergence tolerance. The default is 0.005.
verbose	Logical value indicating whether to print iteration details. The default is FALSE.
initial.para	Optional list of user-supplied initial parameter values for the EM algorithm. The default is NULL.
cpu.cores	Number of CPU cores used for parallel computation. The default is NULL, in which case the function uses its default computation strategy.

Details

Specifies convergence, initialization, optimizer, verbosity, and parallel computing settings for mvjmcs().

Value

A list of control parameters used by mvjmcs().

---

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.

plot *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.

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

Plot subject-specific conditional survival probabilities or cumulative incidence probabilities from a `survfitmvjmcs` object. Longitudinal observations for multiple biomarkers are stacked vertically, with the predicted survival or cumulative incidence curve overlaid on each panel.

Usage

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

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

## S3 method for class 'survfitmvjmcs'
plot(
  x,
  subject = NULL,
  risk = 1,
  include.y = TRUE,
  xlab = NULL,
```

```
  ylab = NULL,
  xlim = NULL,
  ylim.surv = NULL,
  ...
)
```

Arguments

x	An object of class survfitmvjmcs.
include.y	Logical; retained for consistency with other plotting methods. The current method always displays longitudinal biomarker panels. Default is TRUE.
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.
...	Additional graphical arguments passed to the biomarker point plot.
subject	Optional subject or subjects to plot. Can be a subject ID, a numeric subject index, or a vector of IDs or indices. If NULL, all subjects in x$Last.time are plotted. Default is NULL.
risk	Failure type to plot when x$CompetingRisk = TRUE. Default is 1.

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.

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

survfitjmcs

survfitmvjmcs

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)

plot(fit)
```

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, JMMLSM, 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, JMMLSM, mvjmcs

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.jmcs *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")
))
```

---

simJMdata *Simulate single-biomarker joint model data*

Description

Simulates single longitudinal biomarker data and associated time-to-event data. The survival outcome may be generated either as a single-failure time-to-event outcome or as competing risks data with two failure types.

Usage

```
simJMdata(
  seed = 100,
  N = 200,
  increment = 0.7,
  beta = c(5, 1.5, 2, 1, 2),
  sigma2 = 1,
  gamma1 = c(1, 0.5, 0.5),
  gamma2 = c(-0.5, 0.5, 0.25),
  alpha1 = c(1, 0.7),
  alpha2 = c(-1, -0.5),
  lambda1 = 0.05,
  lambda2 = 0.025,
  CL = 5,
  CU = 10,
  covb = diag(rep(1, 2)),
  CR = TRUE
)
```

Arguments

seed	Integer random seed used for data generation. Default is 100.
N	Integer specifying the sample size. Default is 200.
increment	Numeric value specifying the increment of visit times for longitudinal measurements. Default is 0.7.
beta	Numeric vector of true fixed-effect parameters for the longitudinal sub-model.
sigma2	Scalar parameter of error variance for the longitudinal sub-model.
gamma1	Numeric vector of true fixed-effect parameters in the cause-specific hazard sub-model for failure type 1.
gamma2	Numeric vector of true fixed-effect parameters in the cause-specific hazard sub-model for failure type 2.
alpha1	Numeric vector of association parameters of individual mean linking the longitudinal process to the cause-specific hazard for failure type 1.
alpha2	Numeric vector of association parameters individual mean linking the longitudinal process to the cause-specific hazard for failure type 2.
lambda1	Baseline hazard rate for failure type 1. An exponential baseline hazard with rate lambda1 is assumed.
lambda2	Baseline hazard rate for failure type 2. An exponential baseline hazard with rate lambda2 is assumed.
CL	Lower bound of the uniform distribution used to generate censoring times.
CU	Upper bound of the uniform distribution used to generate censoring times.
covb	Variance-covariance matrix for the random effects in the longitudinal sub-model.
CR	Logical; if TRUE, competing risks data with two failure types are simulated. If FALSE, a single-failure time-to-event outcome is generated. Default is TRUE.

Details

Simulate single-biomarker joint model data

Value

A list with the following components:

ydata A long-format data frame containing the simulated longitudinal biomarker measurements.

cdata A data frame containing the simulated event-time data.

`simJMWSVdata` *Simulate joint model data with heterogeneous within-subject variability*

Description

Simulates longitudinal biomarker data with heterogeneous within-subject variability and associated time-to-event data. The survival outcome may be generated either as a single-failure time-to-event outcome or as competing risks data with two failure types.

Usage

```
simJMWSVdata(
  seed = 100,
  N = 200,
  increment = 0.7,
  beta = c(5, 1.5, 2, 1, 2),
  tau = c(0.5, 0.5, -0.2, 0.2, 0.05),
  gamma1 = c(1, 0.5, 0.5),
  gamma2 = c(-0.5, 0.5, 0.25),
  alpha1 = c(1, 0.7),
  alpha2 = c(-1, -0.5),
  vee1 = 0.5,
  vee2 = -0.5,
  lambda1 = 0.05,
  lambda2 = 0.025,
  CL = 5,
  CU = 10,
  covbw = diag(rep(1, 3)),
  CR = TRUE
)
```

Arguments

`seed`	Integer random seed used for data generation. Default is `100`.
`N`	Integer specifying the sample size. Default is `200`.
`increment`	Numeric value specifying the increment of visit times for longitudinal measurements. Default is `0.7`.
`beta`	Numeric vector of true fixed-effect parameters for the longitudinal mean sub-model.
`tau`	Numeric vector of true parameters for the within-subject variability sub-model.
`gamma1`	Numeric vector of true fixed-effect parameters in the cause-specific hazard sub-model for failure type 1.
`gamma2`	Numeric vector of true fixed-effect parameters in the cause-specific hazard sub-model for failure type 2.

alpha1	Numeric vector of association parameters of individual mean linking the longitudinal process to the cause-specific hazard for failure type 1.
alpha2	Numeric vector of association parameters individual mean linking the longitudinal process to the cause-specific hazard for failure type 2.
vee1	Numeric association parameter of within-subject variability linking the longitudinal process to the cause-specific hazard for failure type 1.
vee2	Numeric association parameter of within-subject variability linking the longitudinal process to the cause-specific hazard for failure type 2.
lambda1	Baseline hazard rate for failure type 1. An exponential baseline hazard with rate lambda1 is assumed.
lambda2	Baseline hazard rate for failure type 2. An exponential baseline hazard with rate lambda2 is assumed.
CL	Lower bound of the uniform distribution used to generate censoring times.
CU	Upper bound of the uniform distribution used to generate censoring times.
covbw	Variance-covariance matrix for the random effects in the longitudinal mean and within-subject variability sub-models.
CR	Logical; if TRUE, competing risks data with two failure types are simulated. If FALSE, a single-failure time-to-event outcome is generated. Default is TRUE.

Details

Simulate joint model data with heterogeneous within-subject variability

Value

A list with the following components:

ydatah A long-format data frame containing the simulated longitudinal biomarker measurements.

cdatah A data frame containing the simulated event-time data.

---

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), alpha12 = c(-0.5, 0.5)),
  alpha2 = list(alpha21 = c(0.5), alpha22 = c(-0.5, 0.5)),
  lambda1 = 0.05,
  lambda2 = 0.025,
  CL = 5,
  CU = 10,
  covb = diag(rep(1, 3)),
  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.

`simmvJMdatalm` *Data simulation for multivariate joint models conditional on a landmark time*

Description

Simulate data from a joint model with multiple longitudinal biomarkers and either a single time-to-event outcome or competing risks outcomes. The longitudinal biomarkers are generated from biomarker-specific linear mixed-effects models, allowing different random-effects structures across biomarkers. The event-time process is generated from cause-specific proportional hazards models with associations induced through the biomarker-specific random-effects contribution or current biomarker value at a landmark time.

Usage

```
simmvJMdatalm(
  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), alpha12 = c(-0.5)),
  alpha2 = list(alpha21 = c(0.5), alpha22 = c(-0.5)),
  lambda1 = 0.05,
  lambda2 = 0.025,
  CL = 5,
  CU = 10,
  covb = diag(rep(1, 4)),
  missprob = 0,
  landmark = TRUE,
  s = NULL,
  method = "presentlp",
  pRE = NULL,
  CR = TRUE
)
```

Arguments

`seed`	Integer specifying the random seed used for data simulation.
`N`	Integer specifying the sample size.
`increment`	Numeric scalar specifying the time increment between scheduled longitudinal measurements.
`beta`	A list of numeric vectors specifying the true fixed-effect parameters for the longitudinal submodels. Each list component corresponds to one biomarker.

	The entries are interpreted as the intercept, covariate effects, and time effect; if quadratic time is included, the final entry corresponds to the quadratic time effect.
sigma	Numeric vector specifying the measurement error variances for the longitudinal biomarkers. The length of sigma should equal the number of biomarkers.
gamma1	Numeric vector specifying the true survival fixed-effect parameters for failure type 1.
gamma2	Numeric vector specifying the true survival fixed-effect parameters for failure type 2. Used when CR = TRUE.
alpha1	A list of numeric vectors specifying the true association parameters between the longitudinal biomarkers and failure type 1. Each component corresponds to one biomarker and should have length matching the corresponding entry of pRE.
alpha2	A list of numeric vectors specifying the true association parameters between the longitudinal biomarkers and failure type 2. Each component corresponds to one biomarker and should have length matching the corresponding entry of pRE. Used when CR = TRUE.
lambda1	Numeric scalar specifying the constant baseline hazard rate for failure type 1. An exponential baseline hazard is assumed.
lambda2	Numeric scalar specifying the constant baseline hazard rate for failure type 2. An exponential baseline hazard is assumed. Used when CR = TRUE.
CL	Numeric scalar specifying the lower bound of the uniform distribution used to generate censoring times.
CU	Numeric scalar specifying the upper bound of the uniform distribution used to generate censoring times.
covb	Variance-covariance matrix for the subject-specific random effects. Its dimension must be equal to sum(pRE).
missprob	Numeric scalar between 0 and 1 specifying the probability that a scheduled longitudinal observation is missing. Default is 0.
landmark	Logical; if TRUE, event times and censoring times are generated conditional on survival beyond a landmark time s. Default is TRUE.
s	Numeric scalar specifying the landmark time. Required when landmark = TRUE. Default is NULL.
method	Character string specifying how the longitudinal process is linked to the event-time process. Options are "presentlp" and "present". If method = "presentlp", the event-time model depends on the subject-specific random-effects contribution at the landmark time. If method = "present", the event-time model depends on the full current biomarker value at the landmark time, including both fixed- and random-effects components. Default is "presentlp".
pRE	Integer vector specifying the number of random effects for each biomarker. For example, pRE = c(1, 2) specifies a random-intercept model for biomarker 1, a random intercept-and-slope model for biomarker 2. If NULL, a random-intercept model is assumed for each biomarker. The sum of pRE must equal the dimension of covb. Default is NULL.
CR	Logical; if TRUE, simulate competing risks data with two failure types. If FALSE, simulate a single failure type with independent censoring. Default is TRUE.

Details

Simulate multivariate longitudinal and time-to-event data from a joint model

The function generates two baseline covariates, `X1` and `X2`. The longitudinal biomarkers are generated from biomarker-specific linear mixed-effects models. The number of random effects for each biomarker is controlled by `pRE`. Currently, `pRE` values of 1 and 2 are supported, corresponding respectively to random intercept, random intercept and slope.

When `landmark = TRUE`, event and censoring times are generated after the landmark time `s`. Under `method = "presentlp"`, the survival model uses only the random-effects contribution evaluated at `s`. Under `method = "present"`, the survival model uses the full current biomarker value at `s`.

Value

A list with two elements:

`mvcdata`	A data frame containing survival data, including subject ID, observed event or censoring time, event indicator, and baseline covariates.
`mvydata`	A long-format data frame containing longitudinal biomarker measurements, visit times, subject ID, and baseline covariates.

Author(s)

Shanpeng Li <lishanpeng0913@ucla.edu>

Examples

```
## Not run:
dat <- simmvJMdatalm(
  seed = 100,
  N = 5000,
  increment = 0.7,
  beta = list(
    beta1 = c(5, 1.5, 2, 1),
    beta2 = c(10, 1, 2, 1),
    beta3 = c(8, 1.2, 1.5, 0.8)
  ),
  sigma = rep(1, 3),
  gamma1 = c(1, 0.5),
  gamma2 = c(-0.5, 0.5),
  alpha1 = list(
    alpha11 = -0.5,
    alpha12 = c(0.5, 0.7),
    alpha13 = c(0.3, 0.4)
  ),
  alpha2 = list(
    alpha21 = 0.5,
    alpha22 = c(0.5, 0.8),
    alpha23 = c(0.3, 0.1)
  ),
  lambda1 = 0.05,
  lambda2 = 0.05,
```

```
  covb = diag(c(5, 10, 1, 10, 1)),
  pRE = c(1, 2, 2),
  s = 2,
  landmark = TRUE,
  CR = TRUE
)

mvydata <- dat$mvydata
mvcdata <- dat$mvcdata

## End(Not run)
```

---

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 'DynPredAcc'
summary(
  object,
  metric = c("AUC", "Cindex", "Brier Score", "MAE", "MAEQ"),
  digits = 4,
  ...
)

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

## 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.
metric	a list to indicate what metric to summarize
digits	the number of significant digits to use when printing. Default is 4.
...	further arguments passed to or from other methods.

process	for which model (i.e., longitudinal model or survival model) to extract the estimated coefficients.

Value

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

A table to summarize the model results.

A table to summarize the model results.

A table to summarize the model results.

See Also

JMMLSM

jmcs

mvjmcs

---

summary.Concordance *Summarize Concordance*

Description

Summarize Concordance

Usage

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

Arguments

object	An object of class 'Concordance'.
digits	Number of decimal places to be printed. Default is 4.
...	Further arguments passed to or from other methods.

Value

A data frame containing the cross-validation averaged concordance estimate.

Author(s)

Shanpeng Li <lishanpeng0913@ucla.edu>

See Also

Concordance, jmcs, JMMLSM

`survfitJM.JMMLSM` *Dynamic predictions from fitted joint models*

Description

`survfitJM()` is the recommended user-facing function for dynamic prediction from fitted FastJM joint models. It dispatches automatically according to the class of the fitted model object.

Usage

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

survfitJM(object, ...)

## S3 method for class 'jmcs'
survfitJM(
  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,
  ...
)

## S3 method for class 'mvjmcs'
```

```
survfitJM(
  object,
  seed = 100,
  ynewdata = NULL,
  cnewdata = NULL,
  u = NULL,
  Last.time = NULL,
  obs.time = NULL,
  LOCF = FALSE,
  LOCFcovariate = NULL,
  clongdata = NULL,
  ...
)
```

Arguments

object	A fitted joint model object returned by jmcs(), JMMLSM(), or mvjmcs().
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.
...	Additional arguments passed to the model-specific prediction method.

Details

This function is an S3 generic. Depending on the class of `object`, it dispatches to the corresponding model-specific prediction routine: `survfitJM.jmcs()`, `survfitJM.JMMLSM()`, or `survfitJM.mvjmcs()`.

The model-specific functions `survfitjmcs()`, `survfitJMMLSM()`, and `survfitmvjmcs()` are retained as lower-level functions for backward compatibility.

Value

An object containing dynamic prediction results. The exact structure depends on the fitted model class.

See Also

`jmcs`, `JMMLSM`, `mvjmcs`, `survfitjmcs`, `survfitJMMLSM`, `survfitmvjmcs`

---

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.

Details

This is a model-specific dynamic prediction function for objects fitted by jmcs. It is retained for backward compatibility and for users who prefer direct access to the single-marker prediction routine. For routine use, users are encouraged to call survfitJM, which dispatches automatically according to the class of the fitted model object.

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.

Details

This is a model-specific dynamic prediction function for objects fitted by JMMLSM. It is retained for backward compatibility and for users who prefer direct access to the mixed-effects location-scale prediction routine. For routine use, users are encouraged to call survfitJM, which dispatches automatically according to the class of the fitted model object.

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.

Details

This is a model-specific dynamic prediction function for objects fitted by mvjmcs. It is retained for backward compatibility and for users who prefer direct access to the multivariate prediction routine.

For routine use, users are encouraged to call survfitJM, which dispatches automatically according to the class of the fitted model object.

Value

a list of matrices with conditional probabilities for subjects.

Author(s)

Shanpeng Li <lishanpeng0913@ucla.edu>

See Also

mvjmcs

`timeplot` *Diagnostic plots for the fitted joint model*

Description

Creates diagnostic plots for a fitted `jmcs` object. The function always produces a longitudinal biomarker trajectory plot and an event curve. Optionally, it can also produce a log residual variance plot.

1. a longitudinal biomarker trajectory plot, and
2. a plot of log residual variance over follow-up time (optional).

It also produces one event plot based on `object$CompetingRisk`:

- if `object$CompetingRisk = FALSE`, a Kaplan-Meier plot is shown;
- if `object$CompetingRisk = TRUE`, a cumulative incidence plot is shown.

The event time and status variables are inferred from `object$SurvivalSubmodel` when `event_time_col` and `event_status_col` are not supplied.

Usage

```
timeplot(
  object,
  biomarker,
  id_col,
  time_col,
  visit_col = NULL,
  n.obs = 100,
  seed = 100,
  window_days = 50,
  time_bin_width = 180,
  show_logvar = FALSE,
```

```
  biomarker_y_lab = NULL,
  logvar_y_lab = "Log Residual Variance",
  event_x_lab = NULL,
  event_y_lab = NULL,
  x_lab = NULL,
  x_break_by = NULL,
  ylim_mean = NULL,
  ylim_logvar = NULL,
  event_time_col = NULL,
  event_status_col = NULL,
  fail_code = NULL,
  cr_code = NULL,
  censor_code = 0,
  primary_event_label = "Primary event",
  competing_event_label = NULL,
  km_title = NULL,
  cif_title = NULL,
  center_event_plot = TRUE
)
```

Arguments

object	A fitted object of class jmcs.
biomarker	Unquoted name of the longitudinal biomarker variable in object$ydata.
id_col	Unquoted name of the subject identifier variable in object$ydata.
time_col	Unquoted name of the longitudinal follow-up time variable in object$ydata. This variable is used on the x-axis for the biomarker trajectory and log residual variance plots.
visit_col	Optional unquoted name of a visit variable in object$ydata. If supplied and found, visit-level summaries are used for the longitudinal diagnostic plots. If omitted, or if the supplied column is not found, derived time bins are used instead.
n.obs	Numeric value specifying the number of subjects randomly selected for the longitudinal biomarker trajectory plot. Default is 100.
seed	Optional integer random seed used for reproducible subject sampling in the longitudinal biomarker trajectory plot. Default is 100. If NULL, no seed is set.
window_days	Numeric value giving the half-width of the time window used when selecting observations around each visit- or bin-level mean time. Default is 50.
time_bin_width	Numeric value giving the width of derived time bins when visit_col is not supplied or not found. Default is 180. For small-scale time variables, such as time ranging from 0 to 5, use a smaller value such as 0.5 or 1.
show_logvar	Logical; if TRUE, produces an additional plot of log residual variance over follow-up time. Default is FALSE.
biomarker_y_lab	Optional character string for the y-axis label of the biomarker trajectory plot. If NULL, the name of biomarker is used.

logvar_y_lab Character string for the y-axis label of the log residual variance plot. Default is "Log Residual Variance".

event_x_lab Optional character string for the x-axis label of the event plot. If NULL, the name of the event-time variable is used.

event_y_lab Optional character string for the y-axis label of the event plot. If NULL, the default label from the Kaplan-Meier or cumulative incidence plotting function is used.

x_lab Optional character string for the x-axis label of the longitudinal plots. If NULL, the name of time_col is used.

x_break_by Numeric value controlling spacing of x-axis tick marks in the longitudinal plots. Default is NULL.

ylim_mean Optional numeric vector of length 2 giving y-axis limits for the biomarker trajectory plot.

ylim_logvar Optional numeric vector of length 2 giving y-axis limits for the log residual variance plot.

event_time_col Optional character string giving the event-time variable in object$cdata. If NULL, the function attempts to infer it as the first variable in object$SurvivalSubmodel, i.e. the first argument of Surv(time, status).

event_status_col Optional character string giving the event-status variable in object$cdata. If NULL, the function attempts to infer it as the second variable in object$SurvivalSubmodel, i.e. the second argument of Surv(time, status).

fail_code Numeric or integer code in event_status_col denoting the event of interest. For single-failure models, this is the event code used for the Kaplan-Meier plot. For competing-risk models, this is the primary event code used in the cumulative incidence plot. If the model is single-failure and there is exactly one non-censoring status code, fail_code can be inferred.

cr_code Numeric or integer code in event_status_col denoting the competing event. Required for competing-risk models unless there is exactly one other non-censoring status code besides fail_code, in which case it can be inferred. Ignored for single-failure Kaplan-Meier plots.

censor_code Numeric or integer code in event_status_col denoting censoring. Default is 0.

primary_event_label Character string used in event plot labels for the primary event. Default is "Primary event".

competing_event_label Character string used in cumulative incidence plot labels for the competing event. Required for informative competing-risk labeling. Ignored for single-failure Kaplan-Meier plots. Default is NULL.

km_title Optional character string for the Kaplan-Meier plot title. If NULL, a default title is constructed from primary_event_label.

cif_title Optional character string for the cumulative incidence plot title. If NULL, a default title is constructed from primary_event_label and competing_event_label.

center_event_plot
: Logical; if TRUE, the three-plot layout places the two longitudinal diagnostic plots on the first row and centers the event plot on the second row. If FALSE, the three plots are stacked in one column. Default is TRUE.

Details

The longitudinal trajectory plot displays individual biomarker trajectories for a random subset of subjects, along with a summary mean curve. If visit_col is supplied and found in object$ydata, summaries are calculated by observed visit. Otherwise, the function creates derived time bins using floor(time_col / time_bin_width).

The residual variance plot displays the log residual variance over visit- or bin-level follow-up time. Subject-level fitted values are obtained using fitted(object, type = "Subject", process = "Longitudinal").

The event plot is chosen automatically from the fitted model:

- For CompetingRisk = FALSE, the function creates a Kaplan-Meier plot using fail_code as the event code and censor_code as the censoring code.
- For CompetingRisk = TRUE, the function creates a cumulative incidence plot using fail_code as the primary event and cr_code as the competing event.

In competing-risk settings, the model identifies event codes, but it does not determine which event is scientifically "primary." The user should choose fail_code based on the scientific question. For example, if status = 1 is heart failure and status = 2 is death, then fail_code = 1 and cr_code = 2 answer the question: "What is the cumulative incidence of heart failure over time, accounting for death before heart failure?"

Value

Invisibly returns a list with components:

combined The combined plot object.

p1 The biomarker trajectory plot.

p2 The log residual variance plot.

p_cif The cumulative incidence plot if object$CompetingRisk = TRUE; otherwise NULL.

p_km The Kaplan-Meier plot if object$CompetingRisk = FALSE; otherwise NULL.

cif_fit The fitted cumulative incidence object if applicable; otherwise NULL.

km_fit The fitted Kaplan-Meier object if applicable; otherwise NULL.

event_time_col The event-time column used.

event_status_col The event-status column used.

fail_code The primary event code used.

cr_code The competing event code used, or NULL for single-failure models.

censor_code The censoring code used.

summary_data A data frame of visit- or bin-level longitudinal summary statistics.

obs_per_subject_visit A data frame giving the number of observations per subject and visit/bin after window-based filtering.

`grouping_used` A character string indicating whether summaries were based on an observed visit variable or derived time bins.

`show_logvar` a logic value to indicate whether a log residual variance plot is presented.

Author(s)

Shanpeng Li <lishanpeng0913@ucla.edu>

See Also

jmcs

Examples

```
library(FastJM)

data(ydata)
data(cdata)

# Single-failure example:
# Treat failure_type == 1 as the only event and all other observations as censored.
cdata_single <- cdata
cdata_single$event_single <- as.integer(cdata_single$failure_type == 1)

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

singleplot <- timeplot(
  object = fit_single,
  biomarker = response,
  id_col = ID,
  time_col = time,
  time_bin_width = 0.5,
  primary_event_label = "Event type 1"
)

singleplot$combined
singleplot$p_km

# Competing-risk example using FastJM simulated data.
fit_cr <- 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
)
```

```
crplot <- timeplot(
  object = fit_cr,
  biomarker = response,
  id_col = ID,
  time_col = time,
  time_bin_width = 0.5,
  fail_code = 1,
  cr_code = 2,
  censor_code = 0,
  primary_event_label = "Event type 1",
  competing_event_label = "Event type 2"
)

crplot$combined
crplot$p_cif
```

---

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.

Extract variance-covariance matrix for joint models.

Usage

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

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

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

Arguments

object	an object inheriting from class mvjmcs.
...	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.

a matrix of variance covariance of all parameter estimates.

Author(s)

Shanpeng Li <lishanpeng0913@ucla.edu>

See Also

JMMLSM

jmcs

mvjmcs

---

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