Package 'FastJM'

August 6, 2022

Type Package

Title Semi-Parametric Joint Modeling of Longitudinal and Survival Data

Version 1.2.0

Date 2022-08-03

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

Description Maximum likelihood estimation for the semi-parametric joint modeling of competing risks and longitudinal data applying customized linear scan algorithms, proposed by Li and colleagues (2022) <doi:10.1155/2022/1362913>. The time-to-event data is modelled using a (cause-specific) Cox proportional hazards regression model with time-fixed covariates. The longitudinal outcome is modelled using a linear mixed effects model. The association is captured by shared random effects. The model is estimated using an Expectation Maximization algorithm.

License GPL (>= 3)

NeedsCompilation yes

Imports Rcpp (>= 1.0.7), survival, dplyr, nlme, mvtnorm, Hmisc

LinkingTo Rcpp, RcppEigen

Depends R (>= 3.5.0), MASS, statmod

RoxygenNote 7.2.1

LazyData true

Suggests testthat (>= 3.0.0), spelling

Language en-US

Config/testthat/edition 3

Author Shanpeng Li [aut, cre],
Ning Li [ctb],
Hong Wang [ctb],
Jin Zhou [ctb],
Hua Zhou [ctb],
Gang Li [ctb]

Repository CRAN

Date/Publication 2022-08-06 06:40:04 UTC

R topics documented:

`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)
```

`cdata` *Simulated competing risks data*

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.

`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`.
`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,
  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.
`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`.
`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`, `plot.survfitjmcs`,`vcov.jmcs`

Examples

```
require(FastJM)
# 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 = "survival")
# 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),
                       M = 100,
                       seed = 100)
survfit

oldpar <- par(mfrow = c(2, 2), mar = c(5, 4, 4, 4))
plot(survfit, estimator = "both", include.y = TRUE)
par(oldpar)
```

`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.survfitjmcs` *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 'survfitjmcs'
plot(
  x,
  estimator = c("both", "mean", "median"),
  conf.int = TRUE,
  include.y = FALSE,
  xlab = NULL,
  ylab = NULL,
  xlim = NULL,
  ylim.long = NULL,
  ylim.surv = NULL,
  ...
)
```

Arguments

`x`	x of class `survfitjmcs`.
`estimator`	character string specifying, whether to include in the plot the mean of the conditional probabilities of survival, the median or both. The mean and median are taken as estimates of these conditional probabilities over the M replications of the Monte Carlo scheme described in `survfitjmcs`.
`conf.int`	logical; if TRUE, then a pointwise confidence interval is included in the plot. Default is TRUE.
`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

`survfitjmcs`

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)

# Prediction of cumulative incidence for competing risks data
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),
```

```
                       M = 100,
                       seed = 100)

oldpar <- par(mfrow = c(2, 2), mar = c(5, 4, 4, 4))
plot(survfit, estimator = "both", include.y = TRUE)
par(oldpar)
```

`print` *Print jmcs*

Description

Print jmcs

Usage

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

Arguments

`x`	Object of class 'jmcs'.
`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.

Author(s)

Shanpeng Li `<lishanpeng0913@ucla.edu>`

See Also

`jmcs`

`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`

`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`.
`...`	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")
))
```

`summary` *Anova Method for Fitted Joint Models*

Description

Produce result summaries of a joint model fit.

Usage

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

Arguments

`object`	an object inheriting from class `jmcs`.
`process`	for which model (i.e., longitudinal model or survival model) to extract the estimated coefficients.
`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 model results.

See Also

jmcs

`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,
  M = 500,
  simulate = TRUE,
  quadpoint = 6,
  ...
)
```

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.
`M`	the number of Monte Carlo samples to be generated. Default is 500.
`simulate`	logical; if `TRUE`, a Monte Carlo approach is used to estimate conditional probabilities. Otherwise, a first order estimator is used instead. Default is `TRUE`.
`quadpoint`	number of quadrature points used for estimating conditional probabilities. Default is 6.
`...`	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`

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

Description

Extract variance-covariance matrix for joint models.

Usage

```
## 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.

Author(s)

Shanpeng Li <`lishanpeng0913@ucla.edu`>

See Also

`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.

Index