bivrec — *Survival analysis for bivariate recurrent event processes with BLUP frailties*

Description

Tools for fitting proportional hazards models to clustered bivariate recurrent events data. Nested frailties are modeled by their best linear unbiased predictors under an auxiliary Poisson model.

Usage

```
## S3 method for class 'formula':
bivrec(formula, data = parent.frame(), K1 = 10, K2 = 10,
        excludevars1 = NULL, excludevars2 = NULL, verbose = 1,
        alternating = FALSE, dispest = "pearson", correction = "none",
        computesd = TRUE, fullS = TRUE, fixzero = NULL, smooth = FALSE,
        maxiter = 200, convergence = 1e-3, initial = NULL)
```

Arguments

`formula`	a formula object, similar to coxph. The response to the left of the ~ should be a survival object generated by Surv2. The right side must contain an `id(x)` term, where `x` is a variable that takes a unique value for each subject within a cluster. It may also contain a `cluster(y)` term if `y` is the variable that indicates cluster membership, and a `strata(z)` term.
`data`	a `data.frame` with columns corresponding to the terms in the formula.
`K1`	either an integer, a vector of integers, or a value between 0 and 1, to determine the level of discretization. If it is an integer, `K1` gives the number of breakpoints in the baseline hazard for the first process. If it is a vector of integers, `K1` should have length equal to the number of strata, and each value gives the number of breakpoints in the baseline hazard for each stratum. If it is a number between 0 and 1, it gives the ratio of the number of breakpoints relative to the maximum possible. Defaults to 10.
`K2`	analogous to `K1`, for the second process.
`excludevars1`	a vector of strings giving the names of variables or interactions that should be excluded from the model for the first process.
`excludevars2`	analogous to `excludevars1` for the second process.
`verbose`	an integer from 0 to 3 that determines the quantity of output printed to the screen. Setting `verbose=0` is completely silent.
`alternating`	logical, describing the at-risk function for the model. If `FALSE`, patients are assumed to be constantly at risk for both processes, if `TRUE`, they are only at-risk for one process at a time. Defaults to `FALSE`.

`dispest`	a string, determining the method used to estimate the dispersion parameters. Possible values are `"pearson", "marginal", "ohlsson"`, defaults to `"pearson"`.
`correction`	a string describing the degree-of-freedom correction proposed by Ma. It only applies when `dispest="pearson"`. Possible values are `"single", "double", "none"`, defaults to `"none"`
`computesd`	a boolean determining whether standard errors should be computed. Defaults to `TRUE`.
`fullS`	logical determining whether to use the full sensitivity matrix in computing standard errors, or only the covariate portion. Using `fullS=TRUE` leads to more accurate results but increases computer time. Defaults to `TRUE`.
`fixzero`	a vector of strings, listing any dispersion parameters that should be set to 0. Can contain any subset of `"clust1", "clust2", "subj1", "subj2", "cov"`, all else is ignored. Defaults to `NULL`.
`smooth`	logical, determines whether the baseline hazard should be smoothed at each iteration. Defaults to `FALSE`.
`maxiter`	an integer giving the maximum number of iterations permitted.
`convergence`	double, determines the value of the convergence criterion required at termination.
`initial`	a list of initial values in the format used by the code internally.

Value

An object of class `bivrec` with the following components:

`call` the original call to the model-fitting function

`regression` a list containing results from the regression fit in the last iteration. It has components

- `coefficients1` a vector of regression coefficients for the first process
- `coefficients2` a vector of regression coefficients for the second process
- `loglik1` conditional loglikelihood for the first process.
- `loglik2` conditional loglikelihood for the second process.

`frailty` a list containing results from the frailty estimation in the last iteration. It has components

- `clust1` cluster frailties for process 1
- `clust2` cluster frailties for process 2
- `subj1` subject frailties for process 1
- `subj2` subject frailties for process 2

`dispersion` a list containing results from the dispersion parameter estimation in the last iteration. It has components

- `clust1` cluster frailty variance for process 1
- `clust2` cluster frailty variance for process 2
- `subj1` subject frailty variance for process 1

	`subj2` subject frailty variance for process 2
	`cov` subject frailty covariance
`hazard`	a list describing the baseline hazard. It has components
	`breaks1` matrix of breakpoints in the hazard for each stratum for process 1
	`breaks2` matrix of breakpoints in the hazard for each stratum for process 2
	`hazard1` matrix of hazards in each interval for each stratum for process 1
	`hazard2` matrix of hazards in each interval for each stratum for process 1
`summaries`	a list of summary matrices. It has components
	`regression` summary for the regression coefficients
	`dispersion` summary for the dispersion parameters

Author(s)

Emmanuel Sharef ⟨ess28@cornell.edu⟩

References

E. Sharef and R. Strawderman. “A nested frailty model for clustered bivariate recurrent events”, *in preparation.*

See Also

`summary.bivrec`, `plot.bivrec`

Examples

```
data(m10Ji5)

fit <- bivrec( Surv2(start, stop, delta, Delta) ~
            Z1 + cluster(i) + id(j), data = m10Ji5 )

summary(fit)
plot(fit)
```

`id` *Identify subjects*

Description

A function used in the context of specifying recurrent event models to identify individual subjects.

Usage

```
id(x)
```

Arguments

`x`	a character, factor or numeric variable that should be unique for each subject (or each subject within a cluster).

Value

x

See Also

`Surv2`, `bivrec`

`m10Ji5` *Simulated data set with 10 clusters of 5 patients*

Description

A simulated data set generated using the following settings:

- 10 clusters
- 5 subjects per cluster
- 1 stratum
- lognormal cluster frailties with mean 1, variance 0.25
- lognormal subject frailties with variance .25 and covariance .125
- one time-fixed covariate, generated as Normal(0,.5)
- true regression coefficients are 1 for both processes
- both baseline hazards are Weibull with lambda=10, gamma=1.8
- censoring times are Weibull with lambda=1, gamma=1.8

used for package examples. The object `m10Ji5.fit` contains the fitted bivariate model for this data with default settings.

Usage

```
m10Ji5
```

Format

`m10Ji5` is a data frame of 9 columns and 298 rows, `m10Ji5.fit` is an object of class `bivrec`.

See Also

`bivrec`

plot.bivrec — *Plot of survivor function for bivrec or unirec objects*

Description

Function to plot the the survivor function estimated by `bivrec` or `unirec`.

Usage

```
## S3 method for class 'bivrec':
plot(x, which = c(0,1,2), main=NULL,
    xscale = 1, hazscale = 1, add = FALSE, legend = NULL, ...)

## S3 method for class 'unirec':
plot(x, main=NULL, xscale = 1, hazscale = 1, add = FALSE,...)
```

Arguments

`x`	an object of type `bivrec` or `unirec`.
`which`	an integer that determines which plot to draw. If `0`, plots are drawn for both processes, if `1` or `2`, a plot is drawn for process 1 or 2 respectively.
`main`	for `bivrec`, a character vector of length 2, with two plot titles, for `unirec`, a string giving the main title.
`xscale`	amount by which to scale the x axis (for example to convert time from days to years). The hazard is automatically scaled accordingly.
`hazscale`	additional scaling for the hazard.
`add`	logical, whether to add the lines to the current plot or create a new plot.
`legend`	logical, whether to draw a legend.
`...`	additional parameters passed on to `plot`

See Also

`bivrec`, `unirec`

Examples

```
data(m10Ji5.fit) # Example fitted model (see example(bivrec))

# Default plot of the survivor function of the first process
plot(m10Ji5.fit, which=1, main="Example Plot")

# Add a line showing the effect of the covariate
plot(m10Ji5.fit, which=1,
    hazscale=exp(m10Ji5.fit$regression$coefficients1["Z1"]),
    add=TRUE, lty=2)
```

```
legend("topright", c("Z1 = 0", "Z1 = 1"), lty=c(1,2))
```

`summary.bivrec` *Summary method for bivrec objects*

Description

Prints a summary of a fit from `bivrec`, including pretty-printed p-values and significance indicators.

Usage

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

Arguments

`object`	an object of type `bivrec`.
`digits`	number of digits to be used in pretty-printing output.
`...`	additional parameters for `print`.

Value

An item of type `summary.bivrec` with components `call`, `summary.reg`, `summary.disp` as documented in `bivrec`.

See Also

`bivrec`

`summary.unirec` *Summary method for unirec objects*

Description

Prints a summary of a fit from `unirec`, including pretty-printed p-values and significance indicators.

Usage

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

Arguments

`object`	an object of type `unirec`.
`digits`	number of digits to be used in pretty-printing output.
`...`	additional parameters for `print`.

Value

An item of type `summary.unirec` with components `call`, `summary.reg`, `summary.disp` as documented in `unirec`.

See Also

`unirec`

`Surv2` *Bivariate survival object*

Description

Creates a survival object for bivariate data, to be used in `bivrec`.

Usage

```
Surv2(start, stop, status1, status2)
```

Arguments

`start`	starting time for the interval.
`stop`	ending time for the interval.
`status1`	status indicator for the first event process, 1=event, 0=no event.
`status2`	analogous to `status1`, for the second event process.

Value

an object of class `Surv2`, implemented as a data frame of 4 columns.

See Also

`Surv`, `bivrec`

`unirec`	*Survival analysis for univariate recurrent event processes with BLUP frailties*

Description

Tools for fitting proportional hazards models to clustered recurrent events data. Nested frailties are modeled by their best linear unbiased predictors under an auxiliary Poisson model. The computations are done using the code for `bivrec`, but effectively reduce to the method of Ma et al (2001).

Usage

```
## S3 method for class 'formula':
unirec(formula, data = parent.frame(), K1 = 10,
         excludevars1 = NULL, verbose = 1, dispest = "pearson",
         correction = "none", computesd = TRUE, fullS = TRUE,
         fixzero = NULL, smooth = FALSE, maxiter = 200,
         convergence = 1e-3, initial = NULL)
```

Arguments

`formula`	a formula object, similar to `coxph`. The response to the left of the `~` should be a survival object generated by `Surv`, with three components (start, stop, status). The right side must contain an `id(x)` term, where `x` is a variable that takes a unique value for each subject within a cluster. It may also contain a `cluster(y)` term if `y` is the variable that indicates cluster membership, and a `strata(z)` term.
`data`	a `data.frame` with columns corresponding to the terms in the formula.
`K1`	either an integer, a vector of integers, or a value between 0 and 1, to determine the level of discretization. If it is an integer, `K1` gives the number of breakpoints in the baseline hazard for the process. If it is a vector of integers, `K1` should have length equal to the number of strata, and each value gives the number of breakpoints in the baseline hazard for each stratum. If it is a number between 0 and 1, it gives the ratio of the number of breakpoints relative to the maximum possible. Defaults to 10.
`excludevars1`	a vector of strings giving the names of variables or interactions that should be excluded from the model.
`verbose`	an integer from 0 to 3 that determines the quantity of output printed to the screen. Setting `verbose=0` is completely silent.
`dispest`	a string, determining the method used to estimate the dispersion parameters. Possible values are `"pearson"`, `"marginal"`, `"ohlsson"`, defaults to `"pearson"`.

`correction`	a string describing the degree-of-freedom correction proposed by Ma. It only applies when `dispest="pearson"`. Possible values are `"single", "double", "none"`, defaults to `"none"`
`computesd`	a boolean determining whether standard errors should be computed. Defaults to `TRUE`.
`fullS`	logical determining whether to use the full sensitivity matrix in computing standard errors, or only the covariate portion. Using `fullS=TRUE` leads to more accurate results but increases computer time. Defaults to `TRUE`.
`fixzero`	a vector of strings, listing any dispersion parameters that should be set to 0. Can contain any subset of `"clust1", "subj1"`, all else is ignored. Defaults to `NULL`.
`smooth`	logical, determines whether the baseline hazard should be smoothed at each iteration. Defaults to `FALSE`.
`maxiter`	an integer giving the maximum number of iterations permitted.
`convergence`	double, determines the value of the convergence criterion required at termination.
`initial`	a list of initial values in the format used by the code internally.

Value

An object of class `unirec` with the following components:

`call`	the original call to the model-fitting function
`regression`	a list containing results from the regression fit in the last iteration. It has components `coefficients` a vector of regression coefficients for the process `loglik` conditional loglikelihood.
`frailty`	a list containing results from the frailty estimation in the last iteration. It has components `clust` cluster-level frailty estimates `subj` subject-level frailty estimates
`dispersion`	a list containing results from the dispersion parameter estimation in the last iteration. It has components `clust` cluster-level frailty variance `subj` subject-level frailty variance
`hazard`	a list describing the baseline hazard. It has components `breaks` matrix of breakpoints in the hazard for each stratum `hazard` matrix of hazards in each interval for each stratum
`summaries`	a list of summary matrices. It has components `regression` summary for the regression coefficients `dispersion` summary for the dispersion parameters

Author(s)

Emmanuel Sharef ⟨ess28@cornell.edu⟩

References

R. Ma. "Random effects Cox models: A Poisson modelling approach", *Biometrika*, 90 (1) 157-169, 2001.

E. Sharef and R. Strawderman. "A nested frailty model for clustered bivariate recurrent events", *in preparation.*

See Also

summary.unirec, plot.unirec, bivrec

Examples

```
data(m10Ji5)

fit <- unirec( Surv(start, stop, delta) ~
            Z1 + cluster(i) + id(j), data = m10Ji5 )

summary(fit)
plot(fit)
```

---

vigndata   *Simulated data set used in the blupsurv package vignette*

Description

A simulated data set for illustrative use in the package vignette. To make the discussion less abstract, covariates were given real-world names. The data represent a 10-year study during which patients in the 50 US states were monitored for events of severe pain or fever. The data were generated as follows:

- 50 clusters, labeled with US state names
- between 5 and 25 subjects per cluster
- 1 stratum
- frailties for the pain process are lognormal with cluster and subject- level variance 0.25
- frailties for the fever process are lognormal with cluster and subject- level variance 0.5
- subject-level frailty covariance is 0.15
- age is a rounded Normal variable with mean 50 and variance 10
- sex is a Bernoulli variable with mean 0.5
- true regression coefficients for the pain process are 0 for age and 1 for sex
- true regression coefficient for the fever process are 0.025 for age and 2 for sex

- both baseline hazards are Weibull with lambda=0., gamma=1.8
- censoring times are fixed at 10

used for package examples. The object `vigndata.fit` contains the fitted bivariate model for this data with full discretization.

Usage

```
vigndata
```

Format

`vigndata` is a data frame of 8 columns and 1982 rows, `vigndata.fit` and `vigndata.quickfit` are objects of class `bivrec`.

See Also

`bivrec`