Package ‘coxinterval’

February 19, 2015

Title Cox-type models for interval-censored data
Version 1.0
Date 2014-07-19
Depends Matrix, parallel, survival, timereg. R (>= 2.13.0)
LazyData Yes
LazyLoad Yes
Author Audrey Boruvka and Richard J. Cook
Maintainer Audrey Boruvka <audrey.boruvka@gmail.com>
Description Fits Cox-type models based on interval-censored data from a survival
or illness-death process
SystemRequirements GNU make
NeedsCompilation Yes
License GPL (>= 2)
URL https://github.com/aboruvka/coxinterval
BugReports https://github.com/aboruvka/coxinterval/issues
Repository CRAN
Date/Publication 2014-07-20 20:52:59

R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cosmesis</td>
<td>2</td>
</tr>
<tr>
<td>coxaalenic</td>
<td>2</td>
</tr>
<tr>
<td>coxaalenic.control</td>
<td>5</td>
</tr>
<tr>
<td>coxic</td>
<td>6</td>
</tr>
<tr>
<td>coxic.control</td>
<td>9</td>
</tr>
<tr>
<td>dualic</td>
<td>10</td>
</tr>
<tr>
<td>dualrc</td>
<td>11</td>
</tr>
<tr>
<td>print.coxinterval</td>
<td>12</td>
</tr>
<tr>
<td>summary.coxinterval</td>
<td>12</td>
</tr>
<tr>
<td>trans</td>
<td>13</td>
</tr>
</tbody>
</table>

Index 14
cosmesis  

**Breast cosmesis data**

**Description**

Interval-censored times to cosmetic deterioration for breast cancer patients undergoing radiation or radiation plus chemotherapy.

**Usage**

data(cosmesis)

**Format**

A data frame with 94 observations on the following 3 variables.

- **left**: left endpoint of the censoring interval in months
- **right**: right endpoint of the censoring interval in months
- **treat**: factor with levels **rt** and **rct** representing radiotherapy-only and radiation plus chemotherapy treatments, respectively

**Source**


**References**


**Examples**

data(cosmesis)

coxaalenic  

**Cox-Aalen model for interval-censored survival data**

**Description**

Fit a Cox-Aalen model to interval-censored survival data with fixed covariates.

**Usage**

coxaalenic(formula, data = parent.frame(), subset, init = NULL, formula.timerreg = NULL, init.timerreg = FALSE, close.cplex = TRUE, control, ...)
Arguments

- **formula**: an expression of the form `response ~ terms`, where `response` is a survival object returned by the `Surv` function and `terms` contains at least one multiplicative term identified by the `prop` function from the `timereg` package.

- **data**: an optional data frame in which to interpret the variables named in the arguments `formula` and `formulaNtimereg`.

- **subset**: expression specifying which rows of `data` should be used in the fit. All observations are included by default.

- **init**: a list with elements named `coef` and `basehaz`. The `coef` element should be a scalar or vector specifying the initial values of the multiplicative regression coefficient. If `init = NULL` or `coef = NULL`, this coefficient will be initialized to zero. The element `basehaz` should be a matrix or data frame whose columns represent time and the corresponding value for the cumulative baseline hazard function and any remaining additive cumulative coefficients (also referred to as "regression functions"). The cumulative regression functions are then initialized by linearly interpolating or extrapolating these values. If `init = NULL` or `basehaz = NULL`, the cumulative baseline hazard function is initialized to a linear function of time and any cumulative coefficients are started at zero.

- **formulaNtimereg**: a list of formula objects specifying models to fit with the `timereg` package’s `cox.aalen` function using right-censored observations in `data`. Here the shorthand "." refers to the same terms given in `formula`.

- **initNtimereg**: a logical value indicating that `init` should be overrided by estimates based on the `cox.aalen` fit to the first model in `formulaNtimereg`.

- **closeNcplex**: a logical value indicating that CPLEX data structures should be freed when coxaalenic exits.

- **control**: a named list of parameters controlling the model fit, as returned by the function `coxaalenic.control`. This defaults to `coxaalenic.control()`.

- **...**: additional arguments to be passed to `coxaalenic.control`.

Details

A valid response in the `formula` argument can be expressed as

```r
Surv(<left>, <right>, type = "interval2")
```

where `<left>, <right>` is the censoring interval for the survival time. Following the `survival` package’s `Surv` function, we use the convention that any right-censoring times are provided in the variable `<left>` and `<right>` is set to the NA value.

Terms in the `formula` have either time-varying additive effects on the survival hazard as in Aalen’s additive regression model, or fixed multiplicative effects as in the Cox model. Multiplicative terms are distinguished by applying `timereg`’s `prop` function to each corresponding variable.

coxaalenic requires C functions, which are loaded only if `coxinterval` is installed from source on a system with CPLEX. Refer to the package’s README for detailed instructions.
Value

An object of the class "coxinterval" and "coxaalenic", which is a list with the following components.

call  
the matched call to coxic.

n    
size of the sample used in the model fit.

p    
number of (multiplicative) regression coefficients.

coef 
a named p vector of regression coefficients.

var  
a named p by p covariance matrix of the regression coefficients.

basehaz 
a data frame giving the cumulative regression functions evaluated at time points given by the maximal intersections of the censoring intervals.

init 
list of initial values used in the model fit.

loglik 
a vector giving the initial and final log-likelihood values.

iter 
number of iterations needed to achieve the model fit.

maxnorm 
the maximum norm of the difference between the penultimate and final parameter values.

gradnorm 
the inner product between the final parameter value and the score function.

cputime 
the processing time for parameter and variance estimation.

fit.timereg 
an unnamed list whose components give the cox.aalen fit to any models specified by the formula.timereg argument.

na.action 
the "na.action" attribute of the model frame.

censor.rate 
a vector giving the rates of left-, interval- and right-censoring among the observations used in the model fit.

control 
a named list of arguments passed to coxic.control.

References


See Also

cox.aalen, prop, Surv

Examples

# Fit a Cox model to the breast cosmesis dataset
if (is.loaded("coxaalenic", "coxinterval")) {
  fit <- coxaalenic(Surv(left, right, type = "interval") ~ prop(treat),
                    data = cosmesis)
  fit
  plot(fit$basehaz, type = "s")
}
coxaalenic.control

Control a Cox-Aalen model fit

Description

Set parameters controlling the model fit returned by coxaalenic.

Usage

coxaalenic.control(eps = 1e-07, eps.norm = c("max", "grad"),
                 iter.max = 5000, armijo = 1/3, coef.typ = 1,
                 coef.max = 10, trace = FALSE, thread.max = 1)

Arguments

eps threshold value for the norm used to measure convergence in the parameter estimates.
eps.norm a character string identifying the norm to use in the convergence criteria—either the maximum norm between the current and previous parameter values (eps.norm = "max") or the inner product between the current value and the score (eps.norm = "grad").
iter.max maximum number of iterations to attempt. This ensures that coxaalenic will eventually exit, even when the stopping rule is not met. A warning is issued whenever the estimation routine has stopped before converging on a final parameter value.
armijo a scale factor in (0, 1/2) for Armijo’s (1966) rule—a step-halving line search used to ensure that each iteration achieves an adequate increase in the log-likelihood. The model fit is not particularly sensitive to this value.
coef.typ a scalar or vector of typical (absolute) values for the multiplicative regression coefficient.
coef.max a scalar or vector of probable upper bounds for the multiplicative regression coefficient. This and the coef.typ arguments tune variance estimation via the curvature in the profile log-likelihood.
trace a logical value indicating that CPLEX should print its results to the screen.
thread.max maximum number of CPU threads to allocate to CPLEX. Default value disables multithreading. A value of zero allows CPLEX to set the number of threads automatically. Any value exceeding the total number of logical cores on the system is reset by coxaalenic.control to detectCores(logical = TRUE). If this new value does not reflect the total number of available threads, it is later adjusted by CPLEX.

Value

A list of the above arguments with their final values.
References


See Also
coxaalenic

Examples

```r
if (is.loaded("coxaalenic", "coxinterval"))
  coxaalenic(Surv(left, right, type = "interval2") ~ prop(treat),
  data = cosmesis, control = coxaalenic.control(iter.max = 2,
  trace = TRUE))
```

Description

Fit a Cox model to a progressive Markov illness-death process observed under right-censored survival times and interval- or right-censored progression times.

Usage

```r
coxic(formula, data = parent.frame(), subset, init = NULL,
  formulaNcoxph = NULL, initNcoxph = FALSE, control, ...)
```

Arguments

- `formula`: an expression of the form `response ~ terms`, where `response` is a survival object returned by the `Surv` function and `terms` contain clustering and state-transition variables identified by the `cluster` and `trans` functions, respectively.
- `data`: an optional data frame in which to interpret the variables named in the arguments `formula` and `formulaNcoxph`.
- `subset`: expression specifying which rows of `data` should be used in the fit. All observations are included by default.
- `init`: a named list of the vector `coef`, specifying the initial coefficient values, and matrix or data frame `basehaz`, on which to base the initial cumulative baseline transition intensities. The `basehaz` element should contain columns representing the cumulative transition intensity, time and transition type. If the columns do not appear in this order, they should be indicated by the column names "hazard", "time" and "trans", respectively. The sorted values used to represent the transition types should respectively denote the initial to intermediate, initial to terminal, and intermediate to terminal state transitions. The `NULL` value for `init`
or its components enables default values. For coef the default is zero. For basehaz the default corresponds to linear functions of time with an upper bound of one. Under any alternatives arising from this or the init.coxph arguments, basehaz is interpreted as a step function of time and the initial value is its piecewise linear approximation.

**formula.coxph**

a list of formula objects specifying models to fit with coxph using singly right-censored observations in data. The shorthand indicates the same terms given in formula, with the function strata in place by trans. Under dual-right-censoring, the first component of formula.coxph defaults to same model in formula, which is fit to observations singly-right-censored at the earlier censoring time.

**init.coxph**

a logical value indicating that init should be overridden by estimates based on the coxph fit to the first model specified in formula.coxph.

**control**

a named list of parameters controlling the model fit, returned by the function coxic.control. This defaults to coxic.control().

... additional arguments to be passed to coxic.control.

**Details**

A valid formula argument can be expressed as

```r
Surv(<start>, <stop>, <status>) ~ cluster(<id>) + trans(<from>, <to>) + <covariate terms>
```

where `<start>, <stop>` is largest known time interval over which individual `<id>` is at risk for a transition between the states `<from>` and `<to>`. The variable `<status>` indicates whether or not a transition is observed to occur at `<stop>`.

Under dual censoring (Boruvka and Cook, 2014b), both the originating state and the left endpoint of an at-risk interval may be unknown. This case is handled with `<start> = NA, <from> = NA, <to>` equal to the index of the terminal state and any transition-type-specific covariates taking on the values assumed when `<from>` is equal to the intermediate state index. Under discrete observation of non-terminal events, the right-endpoint of some at-risk intervals may be unknown. For these `<start>` is the initial observation time (zero, unless left-truncated), `<stop> = NA` and `<from>` is equal to the initial state index. Missing values are retained by the NA action `na.coxic`. The default NA action is used to handle any missing values passed to coxph.

Dual censoring typically arises in two scenarios: (1) dual right-censoring, where intermediate events are right-censored before terminal events, and (2) interval-censored intermediate events. For examples of these refer to dualrc and dualic, respectively.

A consequence of dual censoring is that any discrete maximum likelihood estimator has ambiguous support at any failure times associated with these NA values. To resolve this, the cumulative baseline transition intensities are restricted to piecewise linear functions on a sieve with size controlled by arguments passed to coxic.control. This approach requires that both types of transitions to the terminal state are, at least for some subjects, observed exactly.

**Value**

An object of the classes "coxinterval" and "coxic", which is a list with the following components.
call the matched call to coxic.
censor a string indicating the dual censoring type. The value "right" corresponds to strictly dual right-censored data. All other cases return "interval".
n size of the sample used in the model fit.
m number of at-risk intervals used in the model fit.
p number of regression coefficients.
coef a named p vector of regression coefficients.
var a named p by p covariance matrix of the regression coefficients.
basehaz a data frame giving the cumulative baseline transition intensities evaluated over the sieve.
init list of initial values used in the model fit.
loglik a vector giving the initial and final log-likelihood values.
iter number of iterations needed to achieve the model fit.
gradnorm the maximum norm of the score function at the final parameter value.
maxnorm the maximum norm of the difference between the penultimate and final parameter values.
cputime the processing time used for parameter and variance estimation.
fit.coxph an unnamed list whose components give the coxph fit to any models specified by the formula.coxph argument.
na.action the "na.action" attribute of the model frame. Here this corresponds to the result from the custom NA action na.coxic.
censor.rate a named vector of censoring rates.
control a list of arguments passed to coxic.control.

References


See Also

cluster, dualic, dualrc, Surv, trans

Examples

# Fit Cox model to dual-right--censored data
fit <- coxic(Surv(start, stop, status) ~ cluster(id) + trans(from, to)
  + I(z == 1) + I( (z < 0 & to == 2))
  + I(z < (from & to == 2)), data = dualrc,
  sieve.rate = 2/5)

fit
par(mfrow = c(1, 3))
by(fit$basehaz, fit$basehaz$trans, function(x) plot(x[, 2:1],
    type = "l", main = paste(x[, 1]), xlim = c(0, 2), ylim = c(0, 4)))
# Fit Cox model to data with interval-censored progression times
fit <- coxic(Surv(start, stop, status) ~ cluster(id) + trans(from, to) + I(z * (to == 1)) + I(z * (from == 0 & to == 2)) + I(z * (from == c(NA, 1) & to == 2)), data = dualic)
fit

### coxic.control

**Control Cox model fit**

**Description**

Set parameters controlling the model fit returned by `coxic`.

**Usage**

```r
coxic.control(eps = 1e-07, iter.max = 50000, coef.typ = 1,
              coef.max = 10, sieve.const = 1, sieve.rate = 1/3)
```

**Arguments**

- **eps**: maximum threshold for the relative change in the model fit required to meet the iterative estimation routine’s stopping rule.
- **iter.max**: maximum number of iterations to attempt. This ensures that `coxic` will eventually exit, even when there are convergence issues.
- **coef.typ**: a scalar or vector of typical (absolute) values for the regression coefficient.
- **coef.max**: a scalar or vector of probable upper bounds for the regression coefficient. This and the `coef.typ` arguments tune variance estimation via the curvature in the profile log-likelihood.
- **sieve.const**: a constant factor that, in part, determines the sieve size. The factor can be made specific to the transition type with `sieve.const` a vector of length three. Indexing the states from zero, this vector’s components correspond to the state 0 to state 1, 0 to 2, and 1 to 2 transition types, respectively.
- **sieve.rate**: a scalar in (1/8, 1/2) determining the rate at which the sieve increases with the sample size.

**Details**

For a given sample size $n$, the resulting sieve has size at most $sieve.const * n^{sieve.rate}$. Any reduction in size from this value is applied to ensure that each subinterval in the sieve’s time partition captures at least one support point from the semiparametric maximum likelihood estimator based on the subsample with known progression status (Boruvka and Cook, 2014b).
Value

A list of the above arguments with their final values.

References


See Also
coxic

tcoxic

Examples
coxic(Surv(start, stop, status) - cluster(id) + trans(from, to)
+ I(z * (to == 1)) + I(z * (from %in% 0 & to == 2))
+ I(z * (from %in% c(NA, 1) & to == 2)), data = dualrc,
control = coxic.control(eps = 1e-5, sieve.rate = 2/5))

dualic Simulated dual-censored data from an illness-death process

Description

Data from a Markov illness-death process with interval-censored progression times, simulated according to the initial scenario described in Boruvka and Cook (2014b).

Usage
data(dualic)

Format

A data frame with 723 observations on the following 7 variables.

id subject identifier.
from originating state index with 0 denoting the initial state, 1 the intermediate state, 2 the terminal state and NA an unknown state.
to subsequent state index.
start left-endpoint of the at-risk interval at which the subject is known to be at risk for a transition between state from and state to.
stop right-endpoint of the at-risk interval.
status indicator that a transition between state from and state to was observed at stop.
z a binary covariate.
References


See Also

coxic

Examples

data(dualic)

dualrc Simulated dual-right–censored data from an illness-death process

Description

Data from a Markov illness-death process with progression and death observed under dual right censoring, simulated according to the initial scenario described in Boruvka and Cook (2014b).

Usage

data(dualrc)

Format

A data frame with 644 observations on the following 7 variables.

id subject identifier.
from originating state index with 0 denoting the initial state, 1 the intermediate state, 2 the terminal state and NA an unknown state.
to subsequent state index.
start left-endpoint of the time interval at which the subject is known to be at risk for a transition between state from and state to.
stop right-endpoint of the at-risk interval.
status indicator that a transition between state from and state to was observed at stop.
z a binary covariate.

References


See Also

coxic
Examples

```r
data(dualrc)
```

### print.coxinterval

Print method for model fit

#### Description

Prints an object the class "coxinterval".

#### Usage

```r
# S3 method for class 'coxinterval'
print(x, ..., 
```

#### Arguments

- `x`: an object returned by `coxalenic` or `coxic`.
- `...`: further arguments for other methods.

#### See Also

- `summary.coxinterval`

### summary.coxinterval

Summary method for Cox model fit

#### Description

summary method for the class "coxinterval".

#### Usage

```r
# S3 method for class 'coxinterval'
summary(object, conf.int = 0.95, scale = 1, ...) 
```

#### Arguments

- `object`: an object returned by `coxalenic` or `coxic`.
- `conf.int`: level for confidence intervals. If FALSE, no confidence intervals are provided.
- `scale`: scale factor for the confidence intervals, whose limits represent the change in risk associated with one scale unit increase in the corresponding variable.
- `...`: further arguments for other methods.
Value

An object of the class "summary.coxinterval".

Examples

```R
# Fit Cox model to dual-right--censored data
fit <- coxic(Surv(start, stop, status) ~ cluster(id) + trans(from, to)
  + I(z * (to == 1)) + I(z * (from %in% 0 & to == 2))
  + I(z* (from %in% c(NA, 1) & to == 2)), data = dualrc,
  sieve.rate = 2/5)
fit
```

trans

Identify transition type in model terms

Description

A special function for Cox models fit with `coxic` that identifies formula terms specifying the state-transition type.

Usage

`trans(from, to)`

Arguments

- `from` a variable representing the originating state.
- `to` a variable representing the subsequent state.

Value

A combination of the `from` and `to` arguments by column with two attributes:

- "states" a vector giving the unique non-missing values in the `from` and `to` arguments ordered so that the initial state appears first, the intermediate state second, and the terminal state last.
- "types" a vector of transition type labels in terms of the values in the `from` and `to` arguments ordered so that the intermediate transition appears first, the terminal transition directly from the initial state second, and the terminal transition from the intermediate state last.

See Also

- `coxic`

Examples

```R
with(dualrc, trans(from, to))
```
Index

*Topic **datasets**
  - cosmesis, 2
  - dualic, 10
  - dualrc, 11

*Topic **survival**
  - coxaalenic, 2
  - coxaalenic.control, 5
  - coxic, 6
  - coxic.control, 9
  - print.coxinterval, 12
  - summary.coxinterval, 12
  - trans, 13

  cluster, 6, 8
  cosmesis, 2
  coxaalenic, 3, 4
  coxaalenic, 2, 5, 6, 12
  coxaalenic.control, 3, 5
  coxic, 6, 9–13
  coxic.control, 4, 7, 8, 9
  coxph, 7, 8

  dualic, 7, 8, 10
  dualrc, 7, 8, 11

  na.coxic, 7, 8
  na.coxic(coxic), 6

  print.coxinterval, 12
  print.summary.coxinterval
     (summary.coxinterval), 12
  prop, 3, 4

  strata, 7
  summary.coxinterval, 12, 12
  Surv, 3, 4, 6, 8

  trans, 6–8, 13