Package: penPHcure (via r-universe)

October 10, 2024

Type Package

Title Variable Selection in PH Cure Model with Time-Varying Covariates

Version 1.0.2

Date 2019-12-03

Description Implementation of the semi-parametric proportional-hazards (PH) of Sy and Taylor (2000) <doi:10.1111/j.0006-341X.2000.00227.x> extended to time-varying covariates. Estimation and variable selection are based on the methodology described in Beretta and Heuchenne (2019) <doi:10.1080/02664763.2018.1554627>; confidence intervals of the parameter estimates may be computed using a bootstrap approach. Moreover, data following the PH cure model may be simulated using a method similar to Hendry (2014) <doi:10.1002/sim.5945>, where the event-times are generated on a continuous scale from a piecewise exponential distribution conditional on time-varying covariates.

License GPL-2 | GPL-3

Copyright Copyright (C) 2019 University of Liège (Belgium).

Encoding UTF-8

Imports Rcpp, MASS, Rdpack

Depends survival, R (>= 3.5)

LinkingTo Rcpp, RcppArmadillo

URL https://github.com/a-beretta/penPHcure

BugReports https://github.com/a-beretta/penPHcure/issues

RoxygenNote 7.0.0

RdMacros Rdpack

LazyData true

NeedsCompilation yes

Author Alessandro Beretta [aut, cre] (<https://orcid.org/0000-0002-0427-8785>), Cédric Heuchenne [aut] (<https://orcid.org/0000-0002-3150-3044>)

Maintainer Alessandro Beretta <a.beretta@uliege.be>

Repository CRAN

Date/Publication 2019-12-03 17:20:05 UTC

Contents

`penPHcure-package` *Variable Selection in Proportional-Hazards Cure Model with Time-Varying Covariates*

Description

Contrary to standard survival analysis models, which rely on the assumption that the entire population will eventually experience the event of interest, mixture cure models allow to split the population in susceptible and non-susceptible (cured) individuals.

In this R package, we implement the semi-parametric proportional-hazards (PH) cure model of Sy and Taylor (2000) extended to time-varying covariates. If we define T as the time-to-event, the survival function for the entire population is given by

$$S(t) = (1 - p) + pS(t|Y = 1)$$

where p is the incidence (i.e. probability of being susceptible) and $S(t|Y = 1)$ is the latency (i.e. survival function conditional on being susceptible).

The incidence is modeled by a logistic regression model:

$$p = P(Y = 1|\mathbf{x}_i) = \exp(\mathbf{x}_i'\mathbf{b})/(1 + \exp(\mathbf{x}_i'\mathbf{b})),$$

where $\mathbf{x}_i$ is a vector of time-invariant covariates (including the intercept) and $\mathbf{b}$ a vector of unknown coefficients. Whereas, the latency is modeled by a Cox's PH model:

$$\lambda(t|Y = 1, \mathbf{z}_i(t)) = \lambda_0(t|Y = 1)e^{\mathbf{z}_i'(t)\boldsymbol{\beta}},$$

where $\mathbf{z}_i(t)$ is a vector of time-varying covariates, $\lambda_0(t|Y = 1)$ is an arbitrary conditional baseline hazard function and $\boldsymbol{\beta}$ is a vector of unknown coefficients.

The function `penPHcure` allows to:

- estimate the regression coefficients (**b**, β) and the baseline hazard function $\lambda_0(t|Y=1)$;
- compute confidence intervals for the estimated regression coefficients using the basic/percentile bootstrap method;
- perform variable selection with the SCAD-penalized likelihood technique proposed by Beretta and Heuchenne (2019).

Moreover, the function `penPHcure.simulate` allows to simulate data from a PH cure model, where the event-times are generated on a continuous scale from a piecewise exponential distribution conditional on time-varying covariates, using a method similar to the one described in Hendry (2014).

References

Beretta A, Heuchenne C (2019). "Variable selection in proportional hazards cure model with time-varying covariates, application to US bank failures." *Journal of Applied Statistics*, **46**(9), 1529-1549. doi: 10.1080/02664763.2018.1554627.

Hendry DJ (2014). "Data generation for the Cox proportional hazards model with time-dependent covariates: a method for medical researchers." *Statistics in Medicine*, **33**(3), 436-454. doi: 10.1002/sim.5945.

Sy JP, Taylor JM (2000). "Estimation in a Cox proportional hazards cure model." *Biometrics*, **56**(1), 227-236. doi: 10.1111/j.0006341X.2000.00227.x.

See Also

`penPHcure`, `penPHcure.simulate`

`cpRossi` *Criminal Recidivism Data*

Description

A sample of 432 inmates released from Maryland state prisons followed for one year after release (Rossi et al. 1980). The aim of this study was to investigate the relationship between the time to first arrest after release and some covariates observed during the follow-up period. Most of the variables are constant over time, except one binary variable denoting whether the individual was working full time during the follow-up period.

Usage

```
cpRossi

data(cpRossi,package="penPHcure")
```

Format

A `data.frame` in counting process format with 1405 observations for 432 individuals on the following 13 variables.

`id` integer. Identification code for each individual.

`(tstart, tstop ]` integers. Time interval of the observation (in weeks). Observation for each individual start after the first release.

`arrest` factor with 2 levels ("no", "yes"). Denote whether the individual has been arrested during the 1 year follow-up period or not.

`fin` factor with 2 levels ("no", "yes"). Denote whether the inmate received financial aid after release.

`age` integer. Age in years at the time of release.

`race` factor with 2 levels ("black", "other"). Denote whether the race of the individual is black or not.

`wexp` factor with 2 levels ("no", "yes"). Denote whether the individual had full-time work experience before incarceration or not.

`mar` factor with 2 levels ("yes", "no"). Denote whether the inmate was married at the time of release or not.

`paro` factor with 2 levels ("no", "yes"). Denote whether the inmate was released on parole or not.

`prio` integer. The number of convictions an inmate had prior to incarceration.

`educ` factor with 3 levels ("3", "4", "5"). Level of education:

- "3": <=9th degree;
- "4": 10th or 11th degree; and
- "5": >=12 degree.

`emp` factor with 2 levels ("no", "yes"). Denote whether the individual was working full time during the observed time interval.

Source

The `Rossi` dataset in the `RcmdrPlugin.survival` package (Fox and Carvalho 2012) is the source of these data, which have been converted into counting process format.

References

Fox J, Carvalho MS (2012). "The RcmdrPlugin.survival Package: Extending the R Commander Interface to Survival Analysis." *Journal of Statistical Software*, **49**(7), 1–32. http://www.jstatsoft.org/v49/i07/.

Rossi PH, Berk RA, Lenihan KJ (1980). *Money, Work, and Crime: Experimental Evidence*. New York: Academic Press. doi: 10.1016/C20130114122.

penPHcure *Variable selection in PH cure model with time-varying covariates*

Description

This function allows to fit a PH cure model with time varying covariates, to compute confidence intervals for the estimated regression coefficients or to make variable selection through a LASSO/SCAD-penalized model.

Usage

```
penPHcure(
  formula,
  cureform,
  data,
  X = NULL,
  maxIterNR = 500,
  maxIterEM = 500,
  tol = 1e-06,
  standardize = TRUE,
  ties = c("efron", "breslow"),
  SV = NULL,
  which.X = c("last", "mean"),
  inference = FALSE,
  nboot = 100,
  constraint = TRUE,
  pen.type = c("none", "SCAD", "LASSO"),
  pen.weights = NULL,
  pen.tuneGrid = NULL,
  epsilon = 1e-08,
  pen.thres.zero = 1e-06,
  print.details = TRUE,
  warnings = FALSE
)
```

Arguments

`formula`	a formula object, with the response on the left of a ~ operator and the variables to be included in the latency (survival) component on the right. The response must be a survival object returned by the `Surv(time,time2,status)` function.
`cureform`	a one-sided formula object of the form ~ `x1 + x2 + ...` with the covariates to be included in the incidence (cure) component.
`data`	a data.frame (in a counting process format) in which to interpret the variables named in the `formula` and `cureform` arguments.
`X`	a matrix of time-invariant covariates to be included in the incidence (cure) component. If the user provide such matrix, the arguments `cureform` and `which.X` will be ignored. By default, `X = NULL`.

`maxIterNR` a positive integer: the maximum number of iterations to attempt for convergence of the Newton-Raphson (NR) algorithm (Cox's and logistic regression model). By default `maxIterNR = 500`.

`maxIterEM` a positive integer: the maximum number of iterations to attempt for convergence of the Expectation-Maximization (EM) algorithm. By default `maxIterEM = 500`.

`tol` a positive numeric value used to determine convergence of the NR and EM algorithms. By default, `tol = 1e-6`.

`standardize` a logical value. If `TRUE`, the values of the covariates are standardized (centered and scaled), such that their mean and variance will be equal to 0 and 1, respectively. By default, `standardize = TRUE`.

`ties` a character string used to specify the method for handling ties: either `"efron"` or `"breslow"`. By default, `ties = "efron"`.

`SV` a list with elements `b` and `beta`, numeric vectors of starting values for the regression coefficients in the incidence (cure) and latency (survival) component, respectively. By default `SV = NULL`.

`which.X` character string used to specify the method used to transform the covariates included in the incidence (cure) component from time-varying to time-invariant. There are two options: either take the last observation (`"last"`) or the mean over the full history of the covariates (`"mean"`). By default, `which.X = "last"`.

`inference` a logical value. If `TRUE` and `pen.type == "none"`, confidence intervals for the regression coefficient estimates are computed using the basic/percentile bootstrap method. By default `inference = FALSE`.

`nboot` a positive integer: the number of bootstrap resamples for the construction of the confidence intervals (used only when `inference = TRUE`). By default, `nboot = 100`.

`constraint` a logical value. If `TRUE`, the model makes use of the zero-tail constraint, classifying the individuals with censoring times grater than the largest event time as non-susceptible. For more details, see Sy and Taylor (2000). By default `constraint = TRUE`.

`pen.type` a character string used to specify the type of penalty used to make variable selection: either `"none"`, `"SCAD"` or `"LASSO"`. By default, `pen.type="none"`, only a standard model is fitted without performing variable selection.

`pen.weights` a list with elements named `CURE` and `SURV`, positive numeric vectors of penalty weights for the covariates in the incidence (cure) and latency (survival) component, respectively. By default, all weights are set equal to 1, except for the intercept in the incidence (cure) component (always equal to 0).

`pen.tuneGrid` a list with elements named `CURE` and `SURV`, named lists of tuning parameter vectors. If `pen.type == "SCAD"` they should contain two numeric vectors of possible tuning parameters: `lambda` and `a`. Whereas, if `pen.type == "LASSO"`, only one vector `lambda`. By default `lambda = exp(seq(-7,0,length.out = 10))` and `a = 3.7`.

`epsilon` a positive numeric value used as a perturbation of the penalty function. By default, `epsilon = 1e-08`.

`pen.thres.zero` a positive numeric value used as a threshold. After fitting the penalized PH cure model, the estimated regression coefficients with an absolute value lower than this threshold are set equal to zero. By default, `pen.thres.zero = 1e-06`.

`print.details` a logical value. If `TRUE`, tracing information on the progress of the routines is produced. By default `print.details = TRUE`.

`warnings` a logical value. If `TRUE`, possible warnings from the NR and EM algorithms are produced. By default `warnings = FALSE`.

Details

When the starting values (`SV`) are not specified and `pen.type == "none"`:

- `SV$b` is set equal to the estimates of a logistic regression model with the event indicator (0=censored, 1=event) as dependent variable; and
- `SV$beta` is set equal to the estimates of a standard Cox's model.

Whereas, if `pen.type == "SCAD" | "LASSO"`, both vectors are filled with zeros.

When performing variable selection (`pen.type == "SCAD" | "LASSO"`), a penalized PH cure model is fitted for each possible combination of the tuning parameters in `pen.tuneGrid`. Two models are selected on the basis of the Akaike and Bayesian Information Criteria:

$$AIC = -ln(\hat{L}) + 2df,$$

$$BIC = -ln(\hat{L}) + ln(n)df,$$

where $ln(\hat{L})$ is the value of the log-likelihood at the penalized MLEs, df is the value of the degrees of freedom (number of non-zero coefficients) and n is the sample size.

Regarding the possible tuning parameters in `pen.tuneGrid`, the numeric vectors `lambda` and `a` should contain values >= 0 and > 2, respectively.

Value

If the argument `pen.type = "none"`, this function returns a `PHcure.object`. Otherwise, if `pen.type == "SCAD" | "LASSO"`, it returns a `penPHcure.object`.

References

Beretta A, Heuchenne C (2019). "Variable selection in proportional hazards cure model with time-varying covariates, application to US bank failures." *Journal of Applied Statistics*, **46**(9), 1529-1549. doi: 10.1080/02664763.2018.1554627.

Sy JP, Taylor JM (2000). "Estimation in a Cox proportional hazards cure model." *Biometrics*, **56**(1), 227-236. doi: 10.1111/j.0006341X.2000.00227.x.

See Also

`penPHcure-package`, `PHcure.object`, `penPHcure.object`

Examples

```
# Generate some data (for more details type ?penPHcure.simulate in your console)
data <- penPHcure.simulate()

### Standard PH cure model

# Fit standard cure model (without inference)
fit <- penPHcure(Surv(time = tstart,time2 = tstop,
                      event = status) ~ z.1 + z.2 + z.3 + z.4,
                 cureform = ~ x.1 + x.2 + x.3 + x.4,data = data)
# The returned PHcure.object has methods summary and predict,
#  for more details type ?summary.PHcure or ?predict.PHcure in your console.

# Fit standard cure model (with inference)
fit2 <- penPHcure(Surv(time = tstart,time2 = tstop,
                       event = status) ~ z.1 + z.2 + z.3 + z.4,
                  cureform = ~ x.1 + x.2 + x.3 + x.4,data = data,
                  inference = TRUE)
# The returned PHcure.object has methods summary and predict,
#  for more details type ?summary.PHcure or ?predict.PHcure in your console.


### Tune penalized cure model with SCAD penalties

# First define the grid of possible values for the tuning parameters.
pen.tuneGrid <- list(CURE = list(lambda = exp(seq(-7,-2,length.out = 10)),
                                 a = 3.7),
                     SURV = list(lambda = exp(seq(-7,-2,length.out = 10)),
                                 a = 3.7))
# Tune the penalty parameters.
tuneSCAD <- penPHcure(Surv(time = tstart,time2 = tstop,
                           event = status) ~ z.1 + z.2 + z.3 + z.4,
                      cureform = ~ x.1 + x.2 + x.3 + x.4,
                      data = data,pen.type = "SCAD",
                      pen.tuneGrid = pen.tuneGrid)
# The returned penPHcure.object has methods summary and predict, for more
#  details type ?summary.penPHcure or ?predict.penPHcure in your console.

### Tune penalized cure model with LASSO penalties

# First define the grid of possible values for the tuning parameters.
pen.tuneGrid <- list(CURE = list(lambda = exp(seq(-7,-2,length.out = 10))),
                     SURV = list(lambda = exp(seq(-7,-2,length.out = 10))))
# Tune the penalty parameters.
tuneLASSO <- penPHcure(Surv(time = tstart,time2 = tstop,
                            event = status) ~ z.1 + z.2 + z.3 + z.4,
                       cureform = ~ x.1 + x.2 + x.3 + x.4,
                       data = data,pen.type = "LASSO",
                       pen.tuneGrid = pen.tuneGrid)
# The returned penPHcure.object has methods summary and predict, for more
#  details type ?summary.penPHcure or ?predict.penPHcure in your console.
```

penPHcure.object *Penalized PH cure model object*

Description

This class of objects is returned by the function penPHcure when is called with the argument pen.type = "SCAD" | "LASSO". Objects of this class have methods for the functions summary and predict.

Arguments

AIC	a list with elements containing the results of the selected model based on the Akaike information criterion (AIC). See Details.
BIC	a list with elements containing the results of the selected model based on the Bayesian Information Criterion (BIC). See Details.
pen.type	a character string indicating the type of penalty used, either "SCAD" or "LASSO".
tuneGrid	a data.frame containing the values of the AIC and BIC criteria for each combination of the tuning parameters.
pen.weights	a list with elements named CURE and SURV, containing the penalty weights. For more details, see penPHcure.
N	the sample size (number of individuals).
K	the number of unique failure times.
isTies	logical value: TRUE in case of tied event times.
censoring	the proportion of censored individuals.
which.X	character string indicating the method used to transform the covariates included in the incidence (cure) component from time-varying to time-invariant. See penPHcure for more details.
survform	a formula object with all variables involved in the latency (survival) component of the model.
cureform	a formula object with all variables involved in the incidence (survival) component of the model.
call	object of class call, with all the specified arguments.

Details

The lists AIC and BIC contain the results of the selected model based on the Akaike information criterion (AIC) and Bayesian Information Criterion (BIC), respectively. They are composed by the following elements:

- crit: value of the minimized AIC/BIC criterion.
- b: a numeric vector with the estimated regression coefficients in the cure (incidence) component.

- `beta`: a numeric vector with the true estimated coefficients in the survival (latency) component.
- `cumhaz`: a numeric vector with the estimated cumulative baseline hazard function at the unique event times (reported in the `"names"` attribute).
- `tune_params`: a list with elements named `CURE` and `SURV` containing the selected tuning parameters, which minimize the AIC/BIC criterion.

See Also

`penPHcure`

`penPHcure.simulate` *Simulation of a PH cure model with time-varying covariates*

Description

This function allows to simulate data from a PH cure model with time-varying covariates:

- the event-times are generated on a continuous scale from a piecewise exponential distribution conditional on time-varying covariates and regression coefficients `beta0`, using a method similar to the one described in *Hendry (2014)*. The time varying covariates are constant in the intervals $(s_{j-1}, s_j]$, for $j = 1, , , , J$.
- the censoring times are generated from an exponential distribution truncated above s_J;
- the susceptibility indicators are generated from a logistic regression model conditional on time-invariant covariates and regression coefficients `b0`.

Usage

```
penPHcure.simulate(
  N = 500,
  S = seq(0.1, 5, by = 0.1),
  b0 = c(1.2, -1, 0, 1, 0),
  beta0 = c(1, 0, -1, 0),
  gamma = 1,
  lambdaC = 1,
  mean_CURE = rep(0, length(b0) - 1L),
  mean_SURV = rep(0, length(beta0)),
  sd_CURE = rep(1, length(b0) - 1L),
  sd_SURV = rep(1, length(beta0)),
  cor_CURE = diag(length(b0) - 1L),
  cor_SURV = diag(length(beta0)),
  X = NULL,
  Z = NULL,
  C = NULL
)
```

Arguments

N	the sample size (number of individuals). By default, N = 500.
S	a numeric vector containing the end of the time intervals, in ascending order, over which the time-varying covariates are constant (the first interval start at 0). By default, S = seq(0.1, 5, by=0.1).
b0	a numeric vector with the true coefficients in the incidence (cure) component, used to generate the susceptibility indicators. By default, b0 = c(1.2,-1,0,1,0).
beta0	a numeric vector with the true regression coefficients in the latency (survival) component, used to generate the event times. By default, beta0 = c(1,0,-1,0).
gamma	a positive numeric value, parameter controlling the shape of the baseline hazard function: $\lambda_0(t) = \gamma t^{\gamma-1}$. By default, gamma = 1.
lambdaC	a positive numeric value, parameter of the truncated exponential distribution used to generate the censoring times. By default, lambdaC = 1.
mean_CURE	a numeric vector of means for the variables used to generate the susceptibility indicators. By default, all zeros.
mean_SURV	a numeric vector of means for the variables used to generate the event-times. By default, all zeros.
sd_CURE	a numeric vector of standard deviations for the variables used to generate the susceptibility indicators. By default, all ones.
sd_SURV	a numeric vector of standard deviations for the variables used to generate the event-times. By default, all ones.
cor_CURE	the correlation matrix of the variables used to generate the susceptibility indicators. By default, an identity matrix.
cor_SURV	the correlation matrix of the variables used to generate the event-times. By default, an identity matrix.
X	[optional] a matrix of time-invariant covariates used to generate the susceptibility indicators, with dimension N by length(b0)-1L. By default, X = NULL.
Z	[optional] an array of time-varying covariates used to generate the censoring times, with dimension length(S) by length(beta) by N. By default, Z = NULL.
C	[optional] a vector of censoring times with N elements. By default, C = NULL.

Details

By default, the time-varying covariates in the latency (survival) component are generated from a multivariate normal distribution with means mean_SURV, standard deviations sd_SURV and correlation matrix cor_SURV. Otherwise, they can be provided by the user using the argument Z. In this case, the arguments mean_SURV, sd_SURV and cor_SURV will be ignored.

By default, the time-invariant covariates in the incidence (cure) component are generated from a multivariate normal distribution with means mean_CURE, standard deviations sd_CURE and correlation matrix cor_CURE. Otherwise, they can be provided by the user using the argument X. In this case, the arguments mean_CURE, sd_CURE and cor_CURE will be ignored.

Value

A `data.frame` with columns:

`id`	unique ID number associated to each individual.
`tstart`	start of the time interval.
`tstop`	end of the time interval.
`status`	event indicator, 1 if the event occurs or 0, otherwise.
`z.?`	one or more columns of covariates used to generate the survival times.
`x.?`	one or more columns of covariates used to generate the susceptibility indicator (constant over time).

In addition, it contains the following attributes:

`perc_cure`	Percentage of individuals not susceptible to the event of interest.
`perc_cens`	Percentage of censoring.

References

Hendry DJ (2014). "Data generation for the Cox proportional hazards model with time-dependent covariates: a method for medical researchers." *Statistics in Medicine*, **33**(3), 436-454. doi: 10.1002/sim.5945.

Examples

```
### Example 1:
###  - event-times generated from a Cox's PH model with unit baseline hazard
###    and time-varying covariates generated from independent standard normal
###    distributions over the intervals (0,s_1], (s_1,s_2], ..., (s_1,s_J].
###  - censoring times generated from an exponential distribution truncated
###    above s_J.
###  - covariates in the incidence (cure) component generated from independent
###    standard normal distributions.

# Define the sample size
N <- 250
# Define the time intervals for the time-varying covariates
S <- seq(0.1, 5, by=0.1)
# Define the true regression coefficients (incidence and latency)
b0 <- c(1,-1,0,1,0)
beta0 <- c(1,0,-1,0)
# Define the parameter of the truncated exponential distribution (censoring)
lambdaC <- 1.5
# Simulate the data
data1 <- penPHcure.simulate(N = N,S = S,
                            b0 = b0,
                            beta0 = beta0,
                            lambdaC = lambdaC)


### Example 2:
```

```
###  Similar to the previous example, but with a baseline hazard function
###   defined as lambda_0(t) = 3t^2.

# Define the sample size
N <- 250
# Define the time intervals for the time-varying covariates
S <- seq(0.1, 5, by=0.1)
# Define the true regression coefficients (incidence and latency)
b0 <- c(1,-1,0,1,0)
beta0 <- c(1,0,-1,0)
# Define the parameter controlling the shape of the baseline hazard function
gamma <- 3
# Simulate the data
data2 <- penPHcure.simulate(N = N,S = S,
                            b0 = b0,
                            beta0 = beta0,
                            gamma = gamma)


### Example 3:
###  Simulation with covariates in the cure and survival components generated
###   from multivariate normal (MVN) distributions with specific means,
###   standard deviations and correlation matrices.

# Define the sample size
N <- 250
# Define the time intervals for the time-varying covariates
S <- seq(0.1, 5, by=0.1)
# Define the true regression coefficients (incidence and latency)
b0 <- c(-1,-1,0,1,0)
beta0 <- c(1,0,-1,0)
# Define the means of the MVN distribution (incidence and latency)
mean_CURE <- c(-1,0,1,2)
mean_SURV <- c(2,1,0,-1)
# Define the std. deviations of the MVN distribution (incidence and latency)
sd_CURE <- c(0.5,1.5,1,0.5)
sd_SURV <- c(0.5,1,1.5,0.5)
# Define the correlation matrix of the MVN distribution (incidence and latency)
cor_CURE <- matrix(NA,4,4)
for (p in 1:4)
  for (q in 1:4)
    cor_CURE[p,q] <- 0.8^abs(p - q)
cor_SURV <- matrix(NA,4,4)
for (p in 1:4)
  for (q in 1:4)
    cor_SURV[p,q] <- 0.8^abs(p - q)
# Simulate the data
data3 <- penPHcure.simulate(N = N,S = S,
                            b0 = b0,
                            beta0 = beta0,
                            mean_CURE = mean_CURE,
                            mean_SURV = mean_SURV,
                            sd_CURE = sd_CURE,
```

```
                            sd_SURV = sd_SURV,
                            cor_CURE = cor_CURE,
                            cor_SURV = cor_SURV)


### Example 4:
###  Simulation with covariates in the cure and survival components from a
###   data generating process specified by the user.

# Define the sample size
N <- 250
# Define the time intervals for the time-varying covariates
S <- seq(0.1, 5, by=0.1)
# Define the true regression coefficients (incidence and latency)
b0 <- c(1,-1,0,1,0)
beta0 <- c(1,0,-1,0)
# As an example, we simulate data with covariates following independent
#  standard uniform distributions. But the user could provide random draws
#  from any other distribution. Be careful!!! X should be a matrix of size
#  N x length(b0) and Z an array of size length(S) x length(beta0) x N.
X <- matrix(runif(N*(length(b0)-1)),N,length(b0)-1)
Z <- array(runif(N*length(S)*length(beta0)),c(length(S),length(beta0),N))
data4 <- penPHcure.simulate(N = N,S = S,
                            b0 = b0,
                            beta0 = beta0,
                            X = X,
                            Z = Z)


### Example 5:
###  Simulation with censoring times from a data generating process
###   specified by the user

# Define the sample size
N <- 250
# Define the time intervals for the time-varying covariates
S <- seq(0.1, 5, by=0.1)
# Define the true regression coefficients (incidence and latency)
b0 <- c(1,-1,0,1,0)
beta0 <- c(1,0,-1,0)
# As an example, we simulate data with censoring times following
#  a standard uniform distribution between 0 and S_J.
#  Be careful!!! C should be a numeric vector of length N.
C <- runif(N)*max(S)
data5 <- penPHcure.simulate(N = N,S = S,
                            b0 = b0,
                            beta0 = beta0,
                            C = C)
```

PHcure.object *Standard PH cure model object*

Description

This class of objects is returned by the function penPHcure when is called with the argument pen.type = "none". Objects of this class have methods for the functions summary and predict.

Arguments

b	a numeric vector with the estimated regression coefficients in the cure (incidence) component.
beta	a numeric vector with the true estimated regression coefficients in the survival (latency) component.
cumhaz	a numeric vector with the estimated cumulative baseline hazard function at the unique event times (reported in the "names" attribute).
logL	the value of the log-likelihood for the estimated model.
converged	an integer to indicate if the Expectation-Maximization (EM) algorithm converged. The possible values are: 1 if it converged, -1 if it exceeded the maximum number of iterations or -2 if it stopped due to non-finite elements in the regression coefficients.
iter	the maximum number of iteration before the convergence of the Expectation-Maximization (EM) algorithm.
N	the sample size (number of individuals).
K	the number of unique failure times.
isTies	logical value: TRUE in case of tied event times.
censoring	the proportion of censored individuals.
which.X	character string indicating the method used to transform the covariates included in the incidence (cure) component from time-varying to time-invariant. See penPHcure for more details.
survform	a formula object with all variables involved in the latency (survival) component of the model.
cureform	a formula object with all variables involved in the incidence (survival) component of the model.
inference	[optional] a list with elements named bs, betas and nboot. The elements bs and betas are matrices containing on each row the estimated regression coefficients in the incidence and latency components, respectively, for each of the nboot bootstrap resamples. This object is returned only if the function penPHcure was called with the argument inference = TRUE.
call	object of class call, with all the specified arguments.

See Also

penPHcure

`predict.penPHcure` *Predict method for penPHcure.object*

Description

Compute probabilities to be susceptible and survival probabilities (conditional on being susceptible) for a model fitted by `penPHcure` with the argument `pen.type = "SCAD" | "LASSO"`.

Usage

```
## S3 method for class 'penPHcure'
predict(object, newdata, crit.type=c("BIC","AIC"), X = NULL,...)
```

Arguments

`object`	an object of class `PHcure.object`.
`newdata`	a data.frame in counting process format.
`crit.type`	a character string indicating the criterion used to select the tuning parameters, either `"AIC"` or `"BIC"`. By default `crit.type = "BIC"`.
`X`	[optional] a matrix of time-invariant covariates.
`...`	ellipsis to pass extra arguments.

Details

If the model selected by means of the BIC criterion differs from the one selected by the AIC criterion, with the argument `crit.type` it is possible to specify which model to use for the calculation of the probabilities.

If argument `X` was not supplied in the call to the `penPHcure` function, the probabilities to be susceptible are computed using the covariates retrieved using the same `which.X` method as in the `penPHcure` function call.

Value

An object of class `predict.penPHcure`, a list including the following elements:

`CURE` a numeric vector containing the probabilities to be susceptible to the event of interest:

$$P(Y_i = 1|x_i) = \frac{e^{\mathbf{x}_i'\hat{\mathbf{b}}}}{1 + e^{\mathbf{x}_i'\hat{\mathbf{b}}}},$$

where $\mathbf{x}_i$ is a vector of time-invariant covariates and $\hat{\mathbf{b}}$ is a vector of estimated coefficients.

SURV a numeric vector containing the survival probabilities (conditional on being susceptible to the event of interest):

$$S(t_i|Y_i = 1, \bar{\mathbf{z}}_i(t)) = \exp\left(-\sum_{j=1}^{K}(t_{(j-1)} - t_{(j)})\hat{\lambda}_{0j}I(t_{(j)} \leq t_i)e^{\mathbf{z}_i(t_{(j)})\hat{\boldsymbol{\beta}}}\right),$$

where $t_{(1)} < t_{(2)} < ... < t_{(K)}$ denotes the K ordered event-times, $\mathbf{z}_i(t)$ is a vector of time-varying covariates, $\hat{\boldsymbol{\beta}}$ is a vector of estimated coefficients and $\hat{\lambda}_{0j}$ is the estimated baseline hazard function (constant in the interval $(t_{(j-1)}, t_{(j)}]$).

Examples

```
# Generate some data (for more details type ?penPHcure.simulate in your console)
set.seed(12) # For reproducibility
data <- penPHcure.simulate(N=250)

### Tune penalized cure model with SCAD penalties
# First define the grid of possible values for the tuning parameters.
pen.tuneGrid <- list(CURE = list(lambda = c(0.01,0.03,0.05,0.07,0.09),
                                 a = 3.7),
                     SURV = list(lambda = c(0.01,0.03,0.05,0.07,0.09),
                                 a = 3.7))
# Tune the penalty parameters.
tuneSCAD <- penPHcure(Surv(time = tstart,time2 = tstop,
                           event = status) ~ z.1 + z.2 + z.3 + z.4,
                      cureform = ~ x.1 + x.2 + x.3 + x.4,
                      data = data,pen.type = "SCAD",
                      pen.tuneGrid = pen.tuneGrid,
                      print.details = FALSE)

# Use the predict method to obtain the probabilities for the selected model.
#  By default, the model is the one selected on the basis of the BIC criterion.
pred.tuneSCAD.BIC <- predict(tuneSCAD,data)
# Otherwise, to return the probabilities for the model selected on the basis
#  of the AIC criterion, the user can set argument crit.type = "AIC":
pred.tuneSCAD.AIC <- predict(tuneSCAD,data,crit.type="AIC")

# Use the predict method to make prediction for new observations.
#  For example, two individuals censored at time 0.5 and 1.2, respectively,
#  and all cavariates equal to 1.
newdata <- data.frame(tstart=c(0,0),tstop=c(0.5,1.2),status=c(0,0),
                      z.1=c(1,1),z.2=c(1,1),z.3=c(1,1),z.4=c(1,1),
                      x.1=c(1,1),x.2=c(1,1),x.3=c(1,1),x.4=c(1,1))
pred.tuneSCAD.newdata.BIC <- predict(tuneSCAD,newdata)
pred.tuneSCAD.newdata.AIC <- predict(tuneSCAD,newdata,crit.type="AIC")
# The probabilities to be susceptible for the BIC selected model are:
pred.tuneSCAD.newdata.BIC$CURE
# [1] 0.6456631 0.6456631
# The probabilities to be susceptible for the AIC selected model are:
pred.tuneSCAD.newdata.BIC$CURE
# [1] 0.6456631 0.6456631
# The survival probabilities (conditional on being susceptible) for the BIC
```

```
#  selected model are:
pred.tuneSCAD.newdata.BIC$SURV
# [1] 0.5624514 0.1335912
# The survival probabilities (conditional on being susceptible) for the AIC
#  selected model are:
pred.tuneSCAD.newdata.AIC$SURV
# [1] 0.5624514 0.1335912
```

`predict.PHcure` *Predict method for PHcure.object*

Description

Compute probabilities to be susceptible and survival probabilities (conditional on being susceptible) for a model fitted by `penPHcure` with the argument `pen.type = "none"`.

Usage

```
## S3 method for class 'PHcure'
predict(object, newdata, X = NULL,...)
```

Arguments

`object`	an object of class `PHcure.object`.
`newdata`	a data.frame in counting process format.
`X`	[optional] a matrix of time-invariant covariates. It is not required, unless argument `X` was supplied in the call to the `penPHcure` function.
`...`	ellipsis to pass extra arguments.

Details

If argument `X` was not supplied in the call to the `penPHcure` function, the probabilities to be susceptible are computed using the covariates retrieved using the same `which.X` method as in the `penPHcure` function call.

Value

An object of class `predict.PHcure`, a list including the following elements:

`CURE` a numeric vector containing the probabilities to be susceptible to the event of interest:

$$P(Y_i = 1|x_i) = \frac{e^{\mathbf{x}_i'\hat{\mathbf{b}}}}{1 + e^{\mathbf{x}_i'\hat{\mathbf{b}}}},$$

where $\mathbf{x}_i$ is a vector of time-invariant covariates and $\hat{\mathbf{b}}$ is a vector of estimated coefficients.

SURV a numeric vector containing the survival probabilities (conditional on being susceptible to the event of interest):

$$S(t_i|Y_i = 1, \bar{\mathbf{z}}_i(t)) = \exp\left(-\sum_{j=1}^{K}(t_{(j-1)} - t_{(j)})\hat{\lambda}_{0j}I(t_{(j)} \leq t_i)e^{\mathbf{z}_i(t_{(j)})\hat{\boldsymbol{\beta}}}\right),$$

where $t_{(1)} < t_{(2)} < ... < t_{(K)}$ denotes the K ordered event-times, $\mathbf{z}_i(t)$ is a vector of time-varying covariates, $\hat{\boldsymbol{\beta}}$ is a vector of estimated coefficients and $\hat{\lambda}_{0j}$ is the estimated baseline hazard function (constant in the interval $(t_{(j-1)}, t_{(j)}]$).

Examples

```
# Generate some data (for more details type ?penPHcure.simulate in your console)
set.seed(12) # For reproducibility
data <- penPHcure.simulate(N=250)

# Fit standard cure model (without inference)
fit <- penPHcure(Surv(time = tstart,time2 = tstop,
                      event = status) ~ z.1 + z.2 + z.3 + z.4,
                 cureform = ~ x.1 + x.2 + x.3 + x.4,data = data)

# Use the predict method to obtain the probabilities for the fitted model
pred.fit <- predict(fit,data)

# Use the predict method to make prediction for new observations.
#  For example, two individuals censored at time 0.5 and 1.2, respectively,
#  and all cavariates equal to 1.
newdata <- data.frame(tstart=c(0,0),tstop=c(0.5,1.2),status=c(0,0),
                      z.1=c(1,1),z.2=c(1,1),z.3=c(1,1),z.4=c(1,1),
                      x.1=c(1,1),x.2=c(1,1),x.3=c(1,1),x.4=c(1,1))
pred.fit.newdata <- predict(fit,newdata)
# The probabilities to be susceptible are:
pred.fit.newdata$CURE
# [1] 0.6761677 0.6761677
# The survival probabilities (conditional on being susceptible) are:
pred.fit.newdata$SURV
# [1] 0.5591570 0.1379086
```

---

summary.penPHcure *Summary method for penPHcure.object*

Description

Produces a summary of a fitted penalized PH cure model, after selection of the tuning parameters, based on AIC or BIC criteria.

Usage

```
## S3 method for class 'penPHcure'
summary(object,crit.type=c("BIC","AIC"),...)
```

Arguments

object	an object of class penPHcure.object.
crit.type	a character string indicating the criterion used to select the tuning parameters, either "AIC" or "BIC". By default crit.type = "BIC".
...	ellipsis to pass extra arguments.

Value

An object of class summary.penPHcure, a list including the following elements:

N	the sample size (number of individuals).
censoring	the proportion of censored individuals.
K	the number of unique failure times.
isTies	logical value: TRUE in case of tied event times.
pen.type	a character string indicating the type of penalty used, either "SCAD" or "LASSO".
crit.type	a character string indicating the criterion used to select tuning parameters, either "AIC" or "BIC". By default crit.type = "BIC".
tune_params	a list with elements named CURE and SURV containing the selected tuning parameters, which minimize the AIC/BIC criterion.
crit	value of the minimized AIC/BIC criterion.
CURE	a matrix where in the first column the estimated regression coefficients in the cure (incidence) component are provided. If the argument inference (in the penPHcure function) was set equal to TRUE, two additional columns for the confidence intervals are provided.
SURV	a matrix where in the first column the estimated regression coefficients in the survival (latency) component are provided. If the argument inference (in the penPHcure function) was set equal to TRUE, two additional columns for the confidence intervals are provided.

Examples

```
# Generate some data (for more details type ?penPHcure.simulate in your console)
set.seed(12) # For reproducibility
data <- penPHcure.simulate(N=250)

### Tune penalized cure model with SCAD penalties
# First define the grid of possible values for the tuning parameters.
pen.tuneGrid <- list(CURE = list(lambda = c(0.01,0.03,0.05,0.07,0.09),
                                 a = 3.7),
                     SURV = list(lambda = c(0.01,0.03,0.05,0.07,0.09),
                                 a = 3.7))
```

```
# Tune the penalty parameters.
tuneSCAD <- penPHcure(Surv(time = tstart,time2 = tstop,
                           event = status) ~ z.1 + z.2 + z.3 + z.4,
                      cureform = ~ x.1 + x.2 + x.3 + x.4,
                      data = data,pen.type = "SCAD",
                      pen.tuneGrid = pen.tuneGrid,
                      print.details = FALSE)
# Use the summary method to see the results
summary(tuneSCAD)
#
# ------------------------------------------------------
# +++   PH cure model with time-varying covariates   +++
# +++             [ Variable selection ]             +++
# ------------------------------------------------------
# Sample size:  250
# Censoring proportion:  0.5
# Number of unique event times: 125
# Tied failure times:  FALSE
# Penalty type:  SCAD
# Selection criterion:  BIC
#
# ------------------------------------------------------
# +++               Tuning parameters                +++
# ------------------------------------------------------
#  Cure (incidence) --- lambda:  0.07
#                            a:  3.7
#
#  Survival (latency) - lambda:  0.07
#                            a:  3.7
#
#  BIC =  -118.9359
#
# ------------------------------------------------------
# +++                Cure (incidence)                +++
# +++     [ Coefficients of selected covariates ]    +++
# ------------------------------------------------------
#              Estimate
# (Intercept)  0.872374
# x.1         -0.958260
# x.3          0.685916
#
# ------------------------------------------------------
# +++              Survival (latency)                +++
# +++     [ Coefficients of selected covariates ]    +++
# ------------------------------------------------------
#      Estimate
# z.1  0.991754
# z.3 -1.008180

# By default, the summary method for the penPHcure.object returns the selected
#  variables based on the BIC criterion. For AIC, the user can set the
#  argument crit.type equal to "AIC".
summary(tuneSCAD,crit.type = "AIC")
```

```
#
# ------------------------------------------------------
# +++   PH cure model with time-varying covariates   +++
# +++             [ Variable selection ]             +++
# ------------------------------------------------------
# Sample size:  250
# Censoring proportion:  0.5
# Number of unique event times: 125
# Tied failure times:  FALSE
# Penalty type:  SCAD
# Selection criterion:  AIC
#
# ------------------------------------------------------
# +++               Tuning parameters                +++
# ------------------------------------------------------
#  Cure (incidence) --- lambda:  0.07
#                            a:  3.7
#
#  Survival (latency) - lambda:  0.07
#                            a:  3.7
#
#  AIC =  -136.5432
#
# ------------------------------------------------------
# +++                Cure (incidence)                +++
# +++     [ Coefficients of selected covariates ]    +++
# ------------------------------------------------------
#             Estimate
# (Intercept)  0.872374
# x.1         -0.958260
# x.3          0.685916
#
# ------------------------------------------------------
# +++              Survival (latency)                +++
# +++     [ Coefficients of selected covariates ]    +++
# ------------------------------------------------------
#      Estimate
# z.1  0.991754
# z.3 -1.008180
```

---

summary.PHcure *Summary method for PHcure.object*

Description

Produces a summary of a fitted PH cure model

Usage

```
## S3 method for class 'PHcure'
summary(object, conf.int = c("basic","percentile"), conf.int.level = 0.95,...)
```

Arguments

object	an object of class PHcure.object.
conf.int	a character string indicating the method to compute bootstrapped confidence intervals: "percentile" or "basic". By default conf.int = "basic".
conf.int.level	confidence level. By default conf.int.level = 0.95.
...	ellipsis to pass extra arguments.

Value

An object of class summary.PHcure, a list including the following elements:

N	the sample size (number of individuals).
censoring	the proportion of censored individuals.
K	the number of unique failure times.
isTies	a logical value, equal to TRUE in case of tied event times.
conf.int	a character string indicating the method used to compute the bootstrapped confidence intervals: "percentile", "basic" or "no". The latter is returned when the penPHcure function was called with the argument inference = FALSE.
conf.int.level	confidence level used to compute the bootstrapped confidence intervals.
nboot	the number of bootstrap resamples for the construction of the confidence intervals.
logL	the value of the log-likelihood for the estimated model.
CURE	a matrix with one column containing the estimated regression coefficients in the incidence (cure) component. In case the function penPHcure was called with the argument inference = TRUE, two additional columns for the confidence intervals are provided.
SURV	a matrix where in the first column the estimated regression coefficients in the latency (survival) component. In case the function penPHcure was called with the argument inference = TRUE, two additional columns for the confidence intervals are provided.

Examples

```
# For reproducibility
set.seed(12)
# If you use R v3.6 or greater, uncomment the following line
# RNGkind(sample.kind="Rounding")

# Generate some data (for more details type ?penPHcure.simulate in your console)
data <- penPHcure.simulate(N=250)
```

```
# Fit standard cure model (without inference)
fit <- penPHcure(Surv(time = tstart,time2 = tstop,
                      event = status) ~ z.1 + z.2 + z.3 + z.4,
                 cureform = ~ x.1 + x.2 + x.3 + x.4,data = data)

# Use the summary method to see the results
summary(fit)
#
# ------------------------------------------------------
# +++   PH cure model with time-varying covariates   +++
# ------------------------------------------------------
# Sample size: 250
# Censoring proportion: 0.5
# Number of unique event times: 125
# Tied failure times: FALSE
#
# log-likelihood:  74.11
#
# ------------------------------------------------------
# +++         Cure (incidence) coefficients          +++
# ------------------------------------------------------
#              Estimate
# (Intercept)  0.889668
# x.1         -0.972653
# x.2         -0.051580
# x.3          0.714611
# x.4          0.156169
#
# ------------------------------------------------------
# +++         Survival (latency) coefficients         +++
# ------------------------------------------------------
#      Estimate
# z.1  0.996444
# z.2 -0.048792
# z.3 -1.013562
# z.4  0.079422

# Fit standard cure model (with inference). nboot = 30 bootstrap resamples
#  are used to compute the confidence intervals.
fit2 <- penPHcure(Surv(time = tstart,time2 = tstop,
                       event = status) ~ z.1 + z.2 + z.3 + z.4,
                  cureform = ~ x.1 + x.2 + x.3 + x.4,data = data,
                  inference = TRUE,print.details = FALSE,nboot = 30)
# Use the summary method to see the results
summary(fit2)
#
# ------------------------------------------------------
# +++   PH cure model with time-varying covariates   +++
# ------------------------------------------------------
# Sample size: 250
# Censoring proportion: 0.5
# Number of unique event times: 125
# Tied failure times: FALSE
```

```
#
# log-likelihood:  74.11
#
# ------------------------------------------------------
# +++     Cure (incidence) coefficient estimates     +++
# +++          and 95% confidence intervals *        +++
# ------------------------------------------------------
#              Estimate      2.5%     97.5%
# (Intercept)  0.889668  0.455975  1.092495
# x.1         -0.972653 -1.414194 -0.503824
# x.2         -0.051580 -0.557843  0.304632
# x.3          0.714611  0.206211  1.081819
# x.4          0.156169 -0.011555  0.464841
#
# ------------------------------------------------------
# +++    Survival (latency) coefficient estimates    +++
# +++          and 95% confidence intervals *        +++
# ------------------------------------------------------
#      Estimate      2.5%     97.5%
# z.1  0.996444  0.750321  1.130650
# z.2 -0.048792 -0.204435  0.073196
# z.3 -1.013562 -1.127882 -0.780339
# z.4  0.079422 -0.100677  0.193522
#
# ------------------------------------------------------
# * Confidence intervals computed by the basic
#   bootstrap method, with 30 replications.
# ------------------------------------------------------

# By default, confidence intervals are computed by the basic bootstrap method.
#  Otherwise, the user can specify the percentile bootstrap method.
summary(fit2,conf.int = "percentile")
#
# ------------------------------------------------------
# +++   PH cure model with time-varying covariates   +++
# ------------------------------------------------------
# Sample size: 250
# Censoring proportion: 0.5
# Number of unique event times: 125
# Tied failure times: FALSE
#
# log-likelihood:  74.11
#
# ------------------------------------------------------
# +++     Cure (incidence) coefficient estimates     +++
# +++          and 95% confidence intervals *        +++
# ------------------------------------------------------
#              Estimate      2.5%     97.5%
# (Intercept)  0.889668  0.686842  1.323362
# x.1         -0.972653 -1.441483 -0.531112
# x.2         -0.051580 -0.407791  0.454684
# x.3          0.714611  0.347404  1.223011
# x.4          0.156169 -0.152503  0.323893
```

```
#
# ------------------------------------------------------
# +++    Survival (latency) coefficient estimates    +++
# +++          and 95% confidence intervals *        +++
# ------------------------------------------------------
#      Estimate      2.5%     97.5%
# z.1  0.996444  0.862238  1.242567
# z.2 -0.048792 -0.170779  0.106852
# z.3 -1.013562 -1.246785 -0.899242
# z.4  0.079422 -0.034678  0.259521
#
# ------------------------------------------------------
# * Confidence intervals computed by the percentile
#   bootstrap method, with 30 replications.
# ------------------------------------------------------

# By default, a 95% confidence level is used. Otherwise, the user can specify
#  another confidence level: e.g. 90%.
summary(fit2,conf.int.level = 0.90)
#
# ------------------------------------------------------
# +++   PH cure model with time-varying covariates   +++
# ------------------------------------------------------
# Sample size: 250
# Censoring proportion: 0.5
# Number of unique event times: 125
# Tied failure times: FALSE
#
# log-likelihood:  74.11
#
# ------------------------------------------------------
# +++     Cure (incidence) coefficient estimates     +++
# +++          and 90% confidence intervals *        +++
# ------------------------------------------------------
#              Estimate        5%       95%
# (Intercept)  0.889668  0.467864  1.074423
# x.1         -0.972653 -1.397088 -0.618265
# x.2         -0.051580 -0.527389  0.249460
# x.3          0.714611  0.314140  1.028425
# x.4          0.156169  0.033802  0.436361
#
# ------------------------------------------------------
# +++    Survival (latency) coefficient estimates    +++
# +++          and 90% confidence intervals *        +++
# ------------------------------------------------------
#      Estimate        5%       95%
# z.1  0.996444  0.767937  1.125745
# z.2 -0.048792 -0.158821  0.050965
# z.3 -1.013562 -1.120989 -0.800606
# z.4  0.079422 -0.086063  0.180392
#
# ------------------------------------------------------
# * Confidence intervals computed by the basic
```

```
#   bootstrap method, with 30 replications.
# -----------------------------------------------------
```

Index