Package 'plac'

May 9, 2026

Type Package

Title A Pairwise Likelihood Augmented Cox Estimator for Left-Truncated Data

Version 0.1.3

Description A semi-parametric estimation method for the Cox model with left-truncated data using augmented information from the marginal of truncation times.

Depends R (>= 3.2.0), survival (>= 2.38-3)

Imports Rcpp (>= 0.12.1),

Suggests testthat

License GPL (>= 3)

URL https://github.com/942kid/plac

BugReports https://github.com/942kid/plac/issues

LinkingTo Rcpp, RcppEigen

RoxygenNote 7.2.3

Encoding UTF-8

NeedsCompilation yes

Author Fan Wu [aut, cre]

Maintainer Fan Wu <fannwu@umich.edu>

Repository CRAN

Date/Publication 2023-07-02 04:00:02 UTC

Contents

`plac-package` *A Package for Computating the Pairwise Likelihood Augmented Cox Estimator for Left-Truncated Data.*

Description

This package provides both lower-level `C++` functions (`PLAC_TI()`, `PLAC_TV()` and `PLAC_TvR()`) and an R wrapper function `PLAC()` to calculate the pairwise likelihood augmented Cox estimator for left-truncated survival data as proposed by Wu et al. (2018).

Wrapper Function `PLAC()`

This `R` wrapper function calls different `C++` function depending on the covariate types `data` has.

C++ Functions

The three `C++` functions `PLAC_TI()`, `PLAC_TV()` and `PLAC_TvR()` provide a direct interface to the algorithm in case that users need to supply more flexible time-dependent coavriates other than indicator functions.

References

Wu, F., Kim, S., Qin, J., Saran, R., & Li, Y. (2018). A pairwise likelihood augmented Cox estimator for left-truncated data. Biometrics, 74(1), 100-108.

`cum.haz` *Calulate the Values of the cumulative Hazard at Fixed Times*

Description

Calulate the Values of the cumulative Hazard at Fixed Times

Usage

```
cum.haz(est, t.eval = c(0.25, 0.75))
```

Arguments

`est`	an object of the class `plac.fit`.
`t.eval`	time points at which the Lambda(t) is evaluated (for both conditional apporach and the PLAC estimator).

Value

a list containing the estiamtes and SEs of Lambda(t) for both conditional apporach and the PLAC estimator.

Examples

```
dat1 = sim.ltrc(n = 50)$dat
est = PLAC(ltrc.formula = Surv(As, Ys, Ds) ~ Z1 + Z2,
     ltrc.data = dat1, td.type = "none")
H = cum.haz(est, t.eval = seq(0.1, 0.9, 0.1))
H$L
H$se.L
```

`PLAC` *Calculate the PLAC estimator when a time-dependent indicator presents*

Description

Both a conditional approach Cox model and a pairwise likelihood augmented estimator are fitted and the corresponding results are returned in a list.

Usage

```
PLAC(
  ltrc.formula,
  ltrc.data,
  id.var = "ID",
  td.type = "none",
  td.var = NULL,
  t.jump = NULL,
  init.val = NULL,
  max.iter = 100,
  print.result = TRUE,
  ...
)
```

Arguments

ltrc.formula	a formula of of the form Surv(A, Y, D) ~ Z, where Z only include the time-invariate covariates.
ltrc.data	a data.frame of the LTRC dataset including the responses, time-invariate covariates and the jump times for the time-depnencent covariate.
id.var	the name of the subject id in data.
td.type	the type of the time-dependent covariate. Either one of c("none", "independent", "post-trunc", "pre-post-trunc"). See Details.
td.var	the name of the time-dependent covariate in the output.
t.jump	the name of the jump time variable in data.
init.val	a list of the initial values of the coefficients and the baseline hazard function for the PLAC estimator.
max.iter	the maximal number of iteration for the PLAC estimator
print.result	logical, if a brief summary of the regression coefficient estiamtes should be printed out.
...	other arguments

Details

ltrc.formula should have the same form as used in coxph(); e.g., Surv(A, Y, D) ~ Z1 + Z2. where (A, Y, D) are the truncation time, the survival time and the status indicator ((tstart, tstop, event) as in coxph). td.type is used to determine which C++ function will be invoked: either PLAC_TI (if td.type = "none"), PLAC_TD (if td.type = "independent") or PLAC_TDR) (if td.type %in% c("post-trunc", "pre-post-trunc")). For td.type = "post-trunc", the pre-truncation values for the time-dependent covariate will be set to be zero for all subjects.

Value

a list of model fitting results for both conditional approach and the PLAC estimators.

Event.Time Ordered distinct observed event times

b Regression coefficients estiamtes

se.b Model-based SEs of the regression coefficients estiamtes

H0 Estimated cumulative baseline hazard function

se.H0 Model-based SEs of the estimated cumulative baseline hazard function

sandwich The sandwich estimator for (beta, lambda)

k The number of iteration for used for the PLAC estimator

summ A brief summary of the covariates effects

References

Wu, F., Kim, S., Qin, J., Saran, R., & Li, Y. (2018). A pairwise likelihood augmented Cox estimator for left-truncated data. Biometrics, 74(1), 100-108.

Examples

```
# When only time-invariant covariates are involved
dat1 = sim.ltrc(n = 40)$dat
PLAC(ltrc.formula = Surv(As, Ys, Ds) ~ Z1 + Z2,
     ltrc.data = dat1, td.type = "none")
# When there is a time-dependent covariate that is independent of the truncation time
dat2 = sim.ltrc(n = 40, time.dep = TRUE,
               distr.A = "binomial", p.A = 0.8, Cmax = 5)$dat
PLAC(ltrc.formula = Surv(As, Ys, Ds) ~ Z,
     ltrc.data = dat2, td.type = "independent",
     td.var = "Zv", t.jump = "zeta")
# When there is a time-dependent covariate that depends on the truncation time
dat3 = sim.ltrc(n = 40, time.dep = TRUE, Zv.depA = TRUE, Cmax = 5)$dat
PLAC(ltrc.formula = Surv(As, Ys, Ds) ~ Z,
     ltrc.data = dat3, td.type = "post-trunc",
     td.var = "Zv", t.jump = "zeta")
```

PLAC_TD *C++ Function for Solving the PLAC Estimator. (with time-dependent convariates independent of A^*)*

Description

C++ Function for Solving the PLAC Estimator. (with time-dependent convariates independent of A^*)

Usage

```
PLAC_TD(Z, ZFV_, X, W, Ind1, Ind2, Dn, b, h, K = 100L)
```

Arguments

Z	matrix for all the covariates history.
ZFV_	matrix for all covariates at the each individual's observed survival time.
X	the response matrix (As, Xs, Ds).
W	the ordered observed event times.
Ind1	risk-set indicators.
Ind2	truncation pair indicators.
Dn	number of ties at each observed event time.
b	initial values of the regression coefficients.
h	initial values of the baseline hazard function.
K	maximal iteration number, the default is K = 100.

Value

list of model fitting results for both conditional approach and the PLAC estimator.

PLAC_TDR *C++ Function for Solving the PLAC Estimator. (with time-dependent convariates depending on A^*)*

Description

C++ Function for Solving the PLAC Estimator. (with time-dependent convariates depending on A^*)

Usage

```
PLAC_TDR(ZF, ZFV_, Z, X, W, Ind1, Ind2, Dn, b, h, K = 100L)
```

Arguments

ZF	matrix for all the time-invariant covariates.
ZFV_	matrix for all covariates at the each individual's observed survival time.
Z	matrix for all the covariates history.
X	the response matrix (As, Xs, Ds).
W	the ordered observed event times.
Ind1	risk-set indicators.
Ind2	truncation pair indicators.
Dn	number of ties at each observed event time.
b	initial values of the regression coefficients.
h	initial values of the baseline hazard function.
K	maximal iteration number, the default is K = 100.

Value

list of model fitting results for both conditional approach and the PLAC estimator.

PLAC_TI *C++ Function for Solving the PLAC Estimator. (with time-invariant convariates only)*

Description

C++ Function for Solving the PLAC Estimator. (with time-invariant convariates only)

Usage

```
PLAC_TI(Z, X, W, Ind1, Ind2, Dn, b, h, K = 100L)
```

Arguments

Z	matrix for all the covariates history.
X	the response matrix (As, Xs, Ds).
W	the ordered observed event times.
Ind1	risk-set indicators.
Ind2	truncation pair indicators.
Dn	number of ties at each observed event time.
b	initial values of the regression coefficients.
h	initial values of the baseline hazard function.
K	maximal iteration number, the default is K = 100.

Value

list of model fitting results for both conditional approach and the PLAC estimator.

---

plr *Perform the paired log-rank test.*

Description

Perform the paired log-rank test on the truncation times and the residual survival times to check the stationarity assumption (uniform truncation assumption) of the left-truncated right-censored data.

Usage

```
plr(dat, A.name = "As", Y.name = "Ys", D.name = "Ds")
```

Arguments

dat	a data.frame of left-truncated right-censored data.
A.name	the name of the truncation time variable in dat.
Y.name	the name of the survival time variable in dat.
D.name	the name of the event indicator in dat.

Value

a list containing the test statistic and the p-value of the paired log-rant test.

References

Jung, S.H. (1999). Rank tests for matched survival data. *Lifetime Data Analysis, 5(1):67-79*.

Examples

```
dat = sim.ltrc(n = 50, distr.A = "weibull")$dat
plr(dat)
```

PwInd *Generate truncation-pair indicators*

Description

Generate truncation-pair indicators

Usage

```
PwInd(X, W)
```

Arguments

X	the response matrix (As, Xs, Ds).
W	the ordered observed event times.

Value

the truncation-pair indicators of the form I(w_k <= A_i)

- I(w_k <= XA_j).

SgInd *Generate risk-set indicators*

Description

Generate risk-set indicators

Usage

```
SgInd(X, W)
```

Arguments

X	the response matrix (As, Xs, Ds).
W	the ordered observed event times.

Value

risk-set indicators Y_i(w_k) of the form I(A_i <= w_k <= X_i).

`sim.ltrc` *Generate left-truncated (and right-cencored) data from the Cox model.*

Description

Various baseline survival functions and truncation distribution are available. Censoring rate can be designated through tuning the parameter `Cmax`; `Cmas = Inf` means no censoring.

Usage

```
sim.ltrc(
  n = 200,
  b = c(1, 1),
  Z.type = c("C", "B"),
  time.dep = FALSE,
  Zv.depA = FALSE,
  A.depZ = FALSE,
  distr.T = "weibull",
  shape.T = 2,
  scale.T = 1,
  meanlog.T = 0,
  sdlog.T = 1,
  distr.A = "weibull",
  shape.A = 1,
  scale.A = 5,
  p.A = 0.3,
  b.A = c(0, 0),
  Cmax = Inf,
  fix.seed = NULL
)
```

Arguments

`n`	the sample size.
`b`	a numeric vector for true regression coefficients.
`Z.type`	a vector indicating the type of the time-invariant covariates; `"C"` = uniform[-1, 1], `"B"` = binary(0.5).
`time.dep`	logical, whether there is the time-dependent covariate (only one indicator function Zv = I(t >= zeta) is supported); the default is FALSE.
`Zv.depA`	logical, whether the time-dependent covariate `Zv` depends on A^* (the only form supported is Zv = I(t >= zeta + A^*)); the default is FALSE.
`A.depZ`	logical, whether the truncation times depends on the covariate Z.
`distr.T`	the baseline survival time (T*) distribution ("exp" or "weibull").
`shape.T`	the shape parameter for the Weibull distribution of T*.
`scale.T`	the scale parameter for the Weibull distributiof of T*.

meanlog.T	the mean for the log-normal distribution of T*.
sdlog.T	the sd for the log-normal distribution of T*.
distr.A	the baseline truncation time (A*) distribution: either of "weibull" (the default), "unif" (Length-Biased Sampling), "binomial" or "dunif"). Note: If distribution name other than these are provided, "unif" will be used.
shape.A	the shape parameter for the Weibull distribution of A*.
scale.A	the scale parameter for the Weibull distribution of A*.
p.A	the success probability for the binomial distribution of A*.
b.A	the vector of coefficients for the model of A on the covariates.
Cmax	the upper bound of the uniform distribution of the censoring time (C).
fix.seed	an optional random seed for simulation.

Value

a list with a data.frame containing the biased sample of survival times (Ys) and truncation times (As), the event indicator (Ds) and the covariates (Zs); a vector of certain quantiles of Ys (taus); the censoring proportion (PC) and the truncation proportion (PT).

Examples

```
# With time-invariant covariates only
sim1 = sim.ltrc(n = 40)
head(sim1$dat)
# With one time-dependent covariate
sim2 = sim.ltrc(n = 40, time.dep = TRUE,
         distr.A = "binomial", p.A = 0.8, Cmax = 5)
head(sim2$dat)
# With one time-dependent covariate with dependence on the truncation time
sim3 = sim.ltrc(n = 40, time.dep = TRUE, Zv.depA = TRUE, Cmax = 5)
head(sim3$dat)
```

TvInd *Generate time-depependent covariate indicators*

Description

Generate time-depependent covariate indicators

Usage

```
TvInd(zeta, W)
```

Arguments

zeta	the change point (jump time) of Z_v(t).
W	the ordered observed event times.

Value

the time-depependent covariate of the form Z_v(t) = I(w_k > zeta).

`TvInd2` *Generate time-depependent covariate indicators*

Description

Generate time-depependent covariate indicators

Usage

```
TvInd2(eta, zeta, W)
```

Arguments

`eta`	a random variable of the Z_v(t) value before the change point.
`zeta`	the change point (jump time).
`W`	the ordered observed event times.

Value

the time-depependent covariate indicators of the form Z_v(t) = eta * I(w_k <= zeta).

Index