Package ‘SurvSparse’

October 31, 2023

Type Package

Title Survival Analysis with Sparse Longitudinal Covariates

Version 0.1

Description Survival analysis with sparse longitudinal covariates under right censoring scheme. Different hazards models are involved. Please cite the manuscripts corresponding to this package: Sun, Z. et al. (2022) <doi:10.1007/s10985-022-09548-6>, Sun, Z. and Cao, H. (2023) <arXiv:2310.15877> and Sun, D. et al. (2023) <arXiv:2308.15549>.

License GPL-3

Encoding UTF-8

Imports splines, stats, dplyr, MASS, nloptr, nleqslv, tibble,foreach, gaussquad, tidyr , purrr

RoxygenNote 7.2.3

NeedsCompilation no

Author Zhuowei Sun [aut, cre, cph],
Dayu sun [aut, ctb],
Chen Li [aut],
Hongyuan Cao [aut, ctb],
Xingqiu Zhao [aut]

Maintainer Zhuowei Sun <sunzw21@mails.jlu.edu.cn>

Repository CRAN

Date/Publication 2023-10-31 17:30:02 UTC

R topics documented:

add.haz — *Additive hazards model with sparse longitudinal covariates*

Description

Regression analysis of additive hazards model with sparse longitudinal covariates. Three different weighting schemes are provided to impute the missing values.

Usage

```
add.haz(data, n, tau, h, method)
```

Arguments

`data`	An object of class tibble. The structure of the tibble must be: tibble(id = id, X = failure time, covariates = observation for covariates, obs_times = observation times, delta = censoring indicator).
`n`	An object of class integer. The sample size.
`tau`	An object of class numeric. The pre-specified time endpoint.
`h`	An object of class vector. If use auto bandwidth selection, the structure of the vector must be: h = c(the maximum bandwidth, the minimum bandwidth, the number of bandwidth divided). If use fixed bandwidth, h is the chosen bandwidth.
`method`	An object of class integer. If use weighted LVCF, method = 1. If use half kernel, method = 2. If use full kernel, method = 3.

Value

a list with the following elements:

`est`	The estimation for the corresponding parameters.
`se`	The estimation for the standard error of the estimated parameters.

References

Sun, Z. et al. (2022) <doi:10.1007/s10985-022-09548-6>

Examples

```
library(gaussquad)
library(dplyr)
library(nleqslv)
library(MASS)
n=500
lqrule64 <- legendre.quadrature.rules(64)[[64]]
simdata <- function(alpha,beta ) {
```

```
cen=1
nstep=20
Sigmat_z <- exp(-abs(outer(1:nstep, 1:nstep, "-")) / nstep)
z  <-   c(mvrnorm(  1, c(1: nstep)/2, Sigmat_z  ))
left_time_points <- (0:(nstep - 1)) / nstep
z_fun <- stepfun(left_time_points, c(0,z  ))
lam_fun <- function(tt) {  alpha(tt)+beta*z_fun(tt)}
u <- runif(1)
fail_time <- nleqslv( 0 , function(ttt)
 legendre.quadrature(lam_fun,
                   lower = 0,
                   upper = ttt,
                   lqrule64) + log(u))$x

X <- min(fail_time, cen)
obs=rpois(1,5)+1
tt= sort(runif(obs, min = 0, max = 1))
obs_times <- tt[which(tt<=cen)]
if (length(obs_times) == 0)
 obs_times <- cen
 covariates_obscov <-z_fun(obs_times)
 return( tibble(X = X,delta = fail_time < cen,
 covariates = covariates_obscov,obs_times = obs_times, censoring = cen  )  ) }
data <- replicate(n, simdata(alpha = function(tt)  tt, 1  ),
              simplify = FALSE ) %>% bind_rows(.id = "id")

add.haz(data,n,1,n^(-0.5),3)
```

`trans.haz` *Transformed hazards model with sparse longitudinal covariates*

Description

Statistical inference on transformed hazards model with sparse longitudinal covariates. Kernel-weighted log-likelihood and sieve maximum log-likelihood estimation are combined to conduct statistical inference on the hazards function.

Usage

```
trans.haz(data, n, nknots, norder, tau, s, h)
```

Arguments

`data`	An object of class tibble. The structure of the tibble must be: tibble(id = id, X = failure time, covariates = observation for covariates, obs_times = observation times, delta = censoring indicator).
`n`	An object of class integer. The sample size.
`nknots`	An object of class integer. The number of knots for B-spline.
`norder`	An object of class integer. The order of B-spline.

tau	An object of class numeric. The maximum follow-up time.
s	An object of class numeric. The parameter for Box-Cox transformation.
h	An object of class vector. If use auto bandwidth selection, the structure of the vector must be: h = c(the maximum bandwidth, the minimum bandwidth, the number of bandwidth divided). If use fixed bandwidth, h is the chosen bandwidth.

Value

a list with the following elements:

est	The estimation for the corresponding parameters.
se	The estimation for the standard error of the estimated parameters.

References

Sun, D. et al. (2023) <arXiv:2308.15549>

Examples

```
library(dplyr)
library(gaussquad)
library(nleqslv)
library(MASS)
n=200
lqrule64 <- legendre.quadrature.rules(64)[[64]]
simdata <- function(  beta ) {
cen=1
nstep=20
Sigmat_z <- exp(-abs(outer(1:nstep, 1:nstep, "-")) / nstep)
z  <-  2*(pnorm(c(mvrnorm(  1, rep(0,20), Sigmat_z  )))-0.5)
left_time_points <- (0:(nstep - 1)) / nstep
z_fun <- stepfun(left_time_points, c(0,z  ))
h_fun <- function(x) { beta  * z_fun(x)  }
lam_fun <- function(tt) 2 * exp(h_fun(tt))
u <- runif(1)
fail_time <- nleqslv(0, function(ttt)
legendre.quadrature(lam_fun, lower = 0,upper = ttt, lqrule64) + log(u))$x
X <- min(fail_time, cen)
obs=rpois(1,  5)+1
tt= sort(runif(obs, min = 0, max = 1))
obs_times <- tt[which(tt<=cen)]
if (length(obs_times) == 0)
 obs_times <- cen
 covariates_obscov <-z_fun(obs_times)
 return( tibble(X = X,delta = fail_time < cen,
 covariates = covariates_obscov,obs_times = obs_times, censoring = cen  )  )
  }
beta=1
```

```
data <- replicate( n, simdata( beta ), simplify = FALSE ) %>% bind_rows(.id = "id")
trans.haz(data,n,3,3,1,s=0,n^(-0.35))
```

`tv.co.Cox` *Multiplicative hazards model with sparse longitudinal covariates*

Description

Regression analysis of multiplicative hazards model with sparse longitudinal covariates. The kernel weighting approach is employed to impute the missing value and localize the estimating equation. A wild bootstrap-based simultaneous confidence band for the nonparametric function is also provided.

Usage

```
tv.co.Cox(data, n, l, times, bd, scb)
```

Arguments

`data`	An object of class tibble. The structure of the tibble must be: tibble(id = id, X = failure time, covariates = observation for covariates, obs_times = observation times, delta = censoring indicator).
`n`	An object of class integer. The sample size.
`l`	An object of class vector. The selection vector. For example, for the p dimensional regression coefficient function, if we want to construct simultaneous confidence band for the first regression coefficient function, we can take l=c(1,0,...,0).
`times`	An object of class vector. The interest time.
`bd`	An object of class vector. If use auto bandwidth selection, the structure of the vector must be: bd=c(the maximum bandwidth, the minimum bandwidth, the number of bandwidth divided). If use fixed bandwidth, bd is the chosen bandwidth.
`scb`	An object of class vector. If need to construct the simultaneous confidence band, the structure of the vector must be: c(desirable confidence level, repeat times). Otherwise, scb=0.

Value

a list with the following elements:

`est`	The estimation for the corresponding parameters.
`se`	The estimation for the standard error of the estimated parameters.
`scb`	The quantile used to construct simultaneous confidence band.

References

Sun, Z. and Cao, H. (2023) <arXiv:2310.15877>

Examples

```
library(dplyr)
library(gaussquad)
library(MASS)
library(nleqslv)
n=500
beta<-function(t){
  0.5*(t+0.5)^2
}
lqrule64 <- legendre.quadrature.rules(64)[[64]]
simdata <- function(  beta ) {
cen=1
nstep=20
Sigmat_z <- exp(-abs(outer(1:nstep, 1:nstep, "-")) / nstep)
z  <-c(mvrnorm(  1, rep(0,20), Sigmat_z  ))
left_time_points <- (0:(nstep - 1)) / nstep
z_fun <- stepfun(left_time_points, c(0,z  ))
h_fun <- function(x) { beta(x) * z_fun(x)  }
lam_fun <- function(tt) 2 * exp(h_fun(tt))
u <- runif(1)
fail_time <- nleqslv(0, function(ttt)
legendre.quadrature(lam_fun, lower = 0,upper = ttt, lqrule64) + log(u))$x
X <- min(fail_time, cen)
obs=rpois(1,  5)+1
tt= sort(runif(obs, min = 0, max = 1))
obs_times <- tt[which(tt<=cen)]
if (length(obs_times) == 0)
 obs_times <- cen
 covariates_obscov <-z_fun(obs_times)
 return( tibble(X = X,delta = fail_time < cen,
  covariates = covariates_obscov,obs_times = obs_times   )  ) }


data <- replicate(  n, simdata( beta ), simplify = FALSE ) %>% bind_rows(.id = "id")
tv.co.Cox(data,n,l,0.2,bd=c(n^(-0.4),n^(-0.4)),scb=0)
```

Index