Package ‘tpr’

April 10, 2019

Type Package

Title Temporal Process Regression

Version 0.3-1.2

Date 2019-04-10

Author Jun Yan <jyan@stat.uconn.edu>

Maintainer Jun Yan <jyan@stat.uconn.edu>

Description Regression models for temporal process responses with
time-varying coefficient.

Depends R (>= 2.10), stats, lgtdl

License GPL (>= 3)

Repository CRAN

Date/Publication 2019-04-10 07:08:09 UTC

NeedsCompilation yes

R topics documented:

  tpr-package .......................................................... 2
tpr ................................................................. 2
dnase ......................................................... 3
tpr.pfit .................................................. 7
tpr.test .................................................. 8

Index 10
tpr-package  *Temporal Process Regression*

**Description**

Fit regression models for temporal process responses with time-varying and time-independent coefficients.

**Details**

An overview of how to use the package, including the most important functions

**Author(s)**

Jun Yan <jyan@stat.uconn.edu>

**References**


---

**ci.plot  *Confidence Interval Plot***

**Description**

Plotting time-varying coefficient with pointwise confidence.

**Usage**

```r
ci.plot(x, y, se, level = 0.95, ylim = NULL, newplot = TRUE, fun = gaussian()$linkinv, dfun = gaussian()$mu, N)
```

**Arguments**

- **x**: the x coordinate
- **y**: the y coordinate
- **se**: the standard error of y
- **level**: confidence level
- **ylim**: the range of y axis
- **newplot**: if TRUE, draw a new plot
- **fun**: a transform function
- **dfun**: the derivative of the transform function
- **N**: arguments to be passed to plot

**Author(s)**

Jun Yan <jyan@stat.uconn.edu>
**Description**

Randomized trial of rhDNase for treatment of cystic fibrosis

**Usage**

data(dnase)

**Format**

A data frame with 767 observations on the following 6 variables.

- id  subject id
- rx  treatment arm: 0 = placebo, 1 = rhDNase
- fev  forced expiratory volume, a measure of lung capacity
- futime  follow time
- iv1  IV start time
- iv2  IV stop time

**Details**

During an exacerbation, patients received intravenous (IV) antibiotics and were considered unsusceptible until seven exacerbation-free days beyond the end of IV therapy.

A few subjects were infected at the time of enrollment, for instance a subject has a first infection interval of -21 to 7. We do not count this first infection as an "event", and the subject first enters the risk set at day 7.

**Source**


**References**

Yan and Fine (2008). Analysis of Episodic Data with Application to Recurrent Pulmonary Exacerbations in Cystic Fibrosis Patients. JASA.
Examples

## This example steps through how to set up for the tpr function.
## Three objects are needed:
## 1) response process (an object of "lgtdl")
## 2) data availability process (an object of "lgtdl")
## 3) a time-independent covariate matrix

data(dnase)

## extracting the unique id and subject level information
dat <- unique(dnase[,c("id", "futime", "fev", "rx")])

## construct temporal process response for recurrent event
rec <- lapply(split(dnase[,c("id", "iv1", "futime")], dnase$id),
              function(x) {
                v <- x$iv1
                maxfu <- max(x$futime)
                ## iv1 may be negative!!!
                if (is.na(v[1])) c(0, maxfu + 1)
                else if (v[1] < 0) c(v[1] - 1, v[!is.na(v)], maxfu + 1)
                else c(0, v[!is.na(v)], maxfu + 1)
              })

yrec <- lapply(rec,
                function(x) {
                  dat <- data.frame(time=x, cov=1:length(x)-1)
                  len <- length(x)
                  dat$cov[len] <- dat$cov[len - 1]
                  as.lgtdl(dat)
                })

## construct temporal process response for accumulative days exacerbation
do1.acc <- function(x) {
  gap <- x$iv2 - x$iv1 + 1
  if (all(is.na(gap))) yy <- tt <- NULL
  else {
    gap <- na.omit(gap)
    yy <- cumsum(rep(1, sum(gap)))
    tt <- unlist(sapply(1:length(gap), function(i)
                           seq(x$iv1[i], x$iv2[i], by=1.0)))
  }
  yy <- c(0, yy, rev(yy)[1])
  if (!is.null(tt[1]) && tt[1] < 0)
    tt <- c(tt[1] - 1, tt, max(x$futime) + 1)
  else tt <- c(0, tt, max(x$futime) + 1)
  as.lgtdl(data.frame(time=tt, cov=yy))
}

yacc <- lapply(split(dnase[,c("id", "iv1", "iv2", "futime")], dnase$id),
do1.acc)

## construct data availability (or at risk) indicator process
tu <- max(dat$futime) + 0.001
rt <- lapply(1:nrow(dat),
  function(i) {
    x <- dat[i, "futime"]
    time <- c(0, x, tu)
    cov <- c(1, 0, 0)
    as.lgtdl(data.frame(time=time, cov=cov))
  })

## time-independent covariate matrix
xmat <- model.matrix(~ rx + fev, data=dat)
## time-window in days
tlim <- c(10, 168)
good <- unlist(lapply(yrec, function(x) x$time[1] == 0))

## fully functional temporal process regression
## for recurrent event
m.rec <- tpr(yrec, rt, xmat[,1:3], list(), xmat[,-c(1:3), drop=FALSE], list(),
  tis=10:160, w = rep(1, 151), family = poisson(),
  evstr = list(link = 5, n = 3))
par(mfrow=c(1,3), mgp=c(2,1,0), mar=c(4,2,1,0), oma=c(0,2,0,0))
for(i in 1:3) ci.plot(m.rec$tis, m.rec$alpha[,i], sqrt(m.rec$valpha[,i]))

## hypothesis test of significance
## integral test, covariate index 2 and 3
sig.test.int.ff(m.rec, idx=2:3, ncut=2)
sig.test.boots.ff(m.rec, idx=2:3, nsim=1000)
## constant fit
cfit <- cst.fit.ff(m.rec, idx=2:3)

## goodness-of-fit test for constant fit
gof.test.int.ff(m.rec, idx=2:3, ncut=2)
gof.test.boots.ff(m.rec, idx=2:3, nsim=1000)

## for cumulative days in exacerbation
m.acc <- tpr(yacc, rt, xmat[,1:3], list(), xmat[,-c(1:3), drop=FALSE], list(),
  tis=10:160, w = rep(1, 151), family = gaussian(),
  evstr = list(link = 1, n = 1))
par(mfrow=c(1,3), mgp=c(2,1,0), mar=c(4,2,1,0), oma=c(0,2,0,0))
for(i in 1:3) ci.plot(m.acc$tis, m.acc$alpha[,i], sqrt(m.acc$valpha[,i]))

---

tpr

Temporal Process Regression
Description

Regression for temporal process responses and time-independent covariate. Some covariates have
time-varying coefficients while others have time-independent coefficients.

Usage

tpr(y, delta, x, xtv=list(), z, ztv=list(), w, tis,
   family = poisson(),
   evstr = list(link = 5, v = 3),
   alpha = NULL, theta = NULL,
   tidx = 1:length(tis),
   kernstr = list(kern=1, poly=1, band=range(tis)/50),
   control = list(maxit=25, tol=0.0001, smooth=0, intsmooth=0))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Response, a list of &quot;lgtdl&quot; objects.</td>
</tr>
<tr>
<td>delta</td>
<td>Data availability indicator, a list of &quot;lgtdl&quot; objects.</td>
</tr>
<tr>
<td>x</td>
<td>Covariate matrix for time-varying coefficients.</td>
</tr>
<tr>
<td>xtv</td>
<td>A list of list of &quot;lgtdl&quot; for time-varying covariates with time-varying coefficients.</td>
</tr>
<tr>
<td>z</td>
<td>NOT READY YET; Covariate matrix for time-independent coefficients.</td>
</tr>
<tr>
<td>ztv</td>
<td>NOT READY YET; A list of list of &quot;lgtdl&quot; for time-varying covariates with time-independent coefficients.</td>
</tr>
<tr>
<td>w</td>
<td>Weight vector with the same length of tis.</td>
</tr>
<tr>
<td>tis</td>
<td>A vector of time points at which the model is to be fitted.</td>
</tr>
<tr>
<td>family</td>
<td>Specification of the response distribution; see family for glm; this argument is used in getting initial estimates.</td>
</tr>
<tr>
<td>evstr</td>
<td>A list of two named components, link function and variance function. link: 1 = identity, 2 = logit, 3 = probit, 4 = cloglog, 5 = log; v: 1 = gaussian, 2 = binomial, 3 = poisson</td>
</tr>
<tr>
<td>alpha</td>
<td>A matrix supplying initial values of alpha.</td>
</tr>
<tr>
<td>theta</td>
<td>A numeric vector supplying initial values of theta.</td>
</tr>
<tr>
<td>tidx</td>
<td>indices for time points used to get initial values.</td>
</tr>
<tr>
<td>kernstr</td>
<td>A list of two names components: kern: 1 = Epanechnikov, 2 = triangular, 0 = uniform; band: bandwidth</td>
</tr>
<tr>
<td>control</td>
<td>A list of named components: maxit: maximum number of iterations; tol: tolerance level of iterations. smooth: 1 = smoothing; 0 = no smoothing.</td>
</tr>
</tbody>
</table>

Details

This rapper function can be made more user-friendly in the future. For example, evstr can be determined from the family argument.
An object of class "tpr":

- **tis**: same as the input argument
- **alpha**: estimate of time-varying coefficients
- **beta**: estimate of time-independent coefficients
- **valpha**: a matrix of variance of alpha at tis
- **vbeta**: a matrix of variance of beta at tis
- **niter**: the number of iterations used
- **infAlpha**: a list of influence functions for alpha
- **infBeta**: a matrix of influence functions for beta

**Author(s)**

Jun Yan <jyan@stat.uconn.edu>

**References**


---

**tpr.pfit**

*Constant fit of coefficients in a TPR model*

**Description**

Weighted least square estimate of a constant model for time-varying coefficients in a TPR model.

**Usage**

```r
cst.fit.ff(fit, idx)
```

**Arguments**

- **fit**: a fitted object from `tpr`
- **idx**: the index of the

**Value**

The estimated constant fit, standard error, z-value and p-value.

**Author(s)**

Jun Yan <jyan@stat.uconn.edu>
**References**


**See Also**

`tpr.test`

---

**tpr.test**  
*Significance and Goodness-of-fit Test of TPR*

**Description**

Two kinds of tests are provided for inference on the coefficients in a fully functional TRP model: integral test and bootstrap test.

**Usage**

```r
sig.test.int.ff(fit, chypo = 0, idx, weight = TRUE, ncut = 2)
sig.test.boots.ff(fit, chypo = 0, idx, nsim = 1000, plot = FALSE)
gof.test.int.ff(fit, cfitList = NULL, idx, weight = TRUE, ncut = 2)
gof.test.boots.ff(fit, cfitList = NULL, idx, nsim = 1000, plot = FALSE)
gof.test.boots.pf(fit1, fit2, nsim, p = NULL, q = 1)
```

**Arguments**

- `fit`: a fitted object from `tpr`
- `chypo`: hypothesized value of coefficients
- `idx`: the index of the coefficients to be tested
- `weight`: whether or not use inverse variation weight
- `ncut`: the number of cuts of the interval of interest in integral test
- `cfitList`: a list of fitted object from `cst.fit.ff`
- `nsim`: the number of bootstrap samples in bootstrap test
- `plot`: whether or not plot
- `fit1`: fit of H0 model (reduced)
- `fit2`: fit of H1 model (full)
- `p`: the index of the time-varying estimation in fit2
- `q`: the index of the time-independent estimation in fit1

**Value**

Test statistics and their p-values.
Author(s)

Jun Yan <jyan@stat.uconn.edu>

References


See Also

tpr

Examples

## see ?tpr
Index

*Topic **aplot**
ci.plot, 2

*Topic **datasets**
dnase, 3

*Topic **hplot**
ci.plot, 2

*Topic **htest**
tpr.test, 8

*Topic **models**
tpr.pfit, 7

*Topic **multivariate**
tpr.test, 8

*Topic **package**
tpr-package, 2

*Topic **regression**
tpr, 5

*Topic **robust**
tpr, 5
tpr.pfit, 7

ci.plot, 2
cst.fit.ff (tpr.pfit), 7
dnase, 3
gof.test.boots.ff (tpr.test), 8
gof.test.boots.pf (tpr.test), 8
gof.test.int.ff (tpr.test), 8

sig.test.boots.ff (tpr.test), 8
sig.test.int.ff (tpr.test), 8
tpr, 5, 9
tpr-package, 2
tpr.pfit, 7
tpr.test, 8, 8