Package ‘chngpt’

October 30, 2018

LazyLoad yes
LazyData yes
Version 2018.10-17
Title Estimation and Hypothesis Testing for Threshold Regression
Maintainer Youyi Fong <youyifong@gmail.com>
Depends R (>= 3.2.0)
Suggests R.rsp, RUnit, mvtnorm
Imports survival, splines, kyotil, boot, MASS, rgl
VignetteBuilder R.rsp
Description Threshold regression models are also called two-phase regression, broken-stick regression, split-point regression, structural change models, and regression kink models. Methods for both continuous and discontinuous threshold models are included, but the support for the former is much greater. This package is described in Fong, Huang, Gilbert and Permar (2017) chngpt: threshold regression model estimation and inference, BMC Bioinformatics, in press, <DOI:10.1186/s12859-017-1863-x>.
License GPL (>= 2)
NeedsCompilation yes
Author Youyi Fong [cre],
    Zonglin He [aut],
    Adam Elder [aut]
Repository CRAN
Date/Publication 2018-10-30 19:30:03 UTC

R topics documented:

  chngpt ................................................................. 2
  chngpt.test ......................................................... 2
  chngptm .............................................................. 5
  coef.0.ls ............................................................ 9
  dat.mtct ............................................................ 12
  dat.mtct.2 ....................................................... 12
chngpt.test

hinge.test ......................................................... 13
performance.unit.test ............................................ 14
sim.alphas ......................................................... 15
sim.chngpt ......................................................... 15
sim.hinge ......................................................... 17
sim.my ............................................................ 18
sim.pastor ......................................................... 19

Index

20

chngpt   chngpt Package

Description

Please see the Index link below for a list of available functions. The main testing function is chngpt.test(). The main estimation function is chngptm().

chngpt.test   Change Point Tests

Description

Hypothesis testing for change point covariate in logistic regression and linear regression.

Usage

chngpt.test (formula.null, formula.chngpt, family=c("binomial","gaussian"), data,
  type=c("step","hinge","segmented","stegmented"),
  test.statistic=c("lr","score"), # support for score is gradually descreasing
  chngpts=NULL, lb.quantile=.1, ub.quantile=.9,
  chngpts.cnt=50, #this is set to 25 if int is weighted.two.sided or weighted.one.sided
  prec.weights=NULL,
  p.val.method=c("MC","param.boot"),
  mc.n=5e4, # 1e3 won't cut it, the p values estimated could be smaller than nominal
  boot.B=1e4,
  robust=FALSE,
  keep.fits=FALSE, verbose=FALSE
)

antoch.test (formula, data, chngpt.var, plot.=FALSE)

## S3 method for class 'chngpt.test'
plot(x, by.percentile=TRUE, both=FALSE, main=NULL, ...)

Index

20
**Arguments**

- `formula.null`: formula for the null model.
- `formula.chngpt`: formula for the change point model. For example, suppose `formula.null=y~z` and we want to test whether `I(x>cutoff)` is a significant predictor, `formula.chngpt=~x`. If instead we are interested in testing the null that neither `I(x>cutoff)` nor `z*I(x>cutoff)` is a significant predictor, `formula.chngpt=-x*z`
- `data`: data frame.
- `family`: Currently only linear and logistic regression are supported.
- `type`: step: flat before and after change point; hinge: flat before and slope after change point; segmented: slope before and after change point
- `test.statistic`: method for testing main effects of some threshold model.
- `chngpts`: A grid of potential change points to maximize over. If not supplied, they will be set to a vector of length chngpt.cnt equally spaced between lb.quantile and ub.quantile.
- `robust`: Boolean.
- `lb.quantile`: number. The lower bound in the search for change point in the unit of quantile.
- `ub.quantile`: number. The upper bound in the search for change point in the unit of quantile.
- `chngpts.cnt`: integer. Number of potential change points to maximize over.
- `mc.n`: integer. Number of multivariate normal samples to generate in the Monte Carlo procedure to evaluate p-value.
- `verbose`: Boolean.
- `chngpt.var`: string. Name of the predictor to detect change point.
- `plot`: Boolean. Whether to make a plot.
- `formula`: formula.
- `x`: An object of type chngpt.test.
- `...`: arguments passed to or from methods
- `by.percentile`
- `both`
- `main`
- `prec.weights`
- `p.val.method`
- `boot.B`
- `keep.fits`

**Details**

The model under the alternative is the model under the null plus terms involving the threshold. For example, when the type is segmented and `formula.null=~z`, `formula.chngpt=~x`, the model under the null is `~z+x` and the model under the alternative is `~z+x+(x-c)_+`.

If there are missing values in the chngpt formula, those rows will be removed from the whole dataset, including null model and chngpt model.

`antoch.test` is only implemented for main effect only and is based on Antoch et al. (2004). Also see Fong et al. (2014).
**Value**

A list of class htest and chngpt.test

- `p.value` P-value
- `family` Family from input
- `method` Method from input

**References**


**Examples**

```r
# Examples of using chngpt.test

dat=sim.chngpt("thresholded", "step", n=200, seed=1, beta=1, alpha=-1, x.distr="norm", e.=4, family="binomial")
test=chngpt.test(formula.null=y~z, formula.chngpt=-x, dat, type="step", family="binomial", mc.n=10)
test
plot(test)

# segmented example

dat=sim.chngpt("thresholded", "segmented", n=200, seed=1, beta=1, alpha=-1, x.distr="norm", e.=4, family="binomial")
test=chngpt.test(formula.null=y~z, formula.chngpt=-x, dat, type="segmented", family="binomial", mc.n=10)
test
plot(test)

# segmented example with interaction

test = chngpt.test(formula.null=Volume~1, formula.chngpt=-Girth, family="gaussian", data=trees, type="segmented", mc.n=1e4, verbose=FALSE, chngpts.cnt=100, test.statistic="lr")
test
plot(test)

# Not run:
# not run because otherwise the examples take >5s and that is a problem for R CMD check

# has interaction

test = chngpt.test(formula.null=y~z, formula.chngpt=-x*z, dat, type="step", family="binomial")
test
plot(test)

# End(Not run)
```
Estimate change point logistic model

Usage

```r
chngptm (formula.1, formula.2, family, data, type = c("step", "hinge", "upperhinge", "segmented", "segmented2", "segmented"), formula.strat = NULL, weights = NULL, est.method = c("default", "smoothapprox", "grid", "fastgrid", "fastgrid2", "gridC"), var.type = c("none", "robust", "model", "robusttruth", "bootstrap", "all"), aux.fit = NULL, lb.quantile = 0.1, ub.quantile = 0.9, grid.search.max = 5000, test.inv.ci = TRUE, boot.test.inv.ci = FALSE, ci.bootstrap.size = 1000, alpha = 0.05, save.boot = FALSE, m.out.of.n = FALSE, b.transition = Inf, tol = 1e-04, maxit = 100, chngpt.init = NULL, search.bound = 10, keep.best.fit = TRUE, verbose = FALSE, ...)
```

```r
chngptm.xy(x, y, type=c("step","hinge","segmented","segmented2","segmented"), ...)
```

### S3 method for class 'chngptm'

- `coef(object, ...)`
- `residuals(object, ...)`
- `vcov(object, var.type=NULL, ...)`
- `print(x, ...)`
- `plot(x, which = NULL, xlim = NULL, lwd = 2, lcol = "red", add = FALSE, add.points = TRUE, add.ci = TRUE, breaks = 20, ...)`
- `summary(object, var.type = NULL, expo = FALSE, show.slope.post.threshold = FALSE, verbose = FALSE, ...)`

```r
lincomb(object, comb, alpha=0.05)
```
Arguments

- **formula.1**: The part of formula that is free of terms involving thresholded variables
- **formula.2**: The part of formula that is only composed of thresholded variables
- **formula.strat**: stratification formula
- **family**: string. `coxph` or any valid argument that can be passed to `glm`. But variance estimate is only available for binomial and gaussian (only model-based for latter)
- **data**: data frame.
- **type**: types of threshold effects. `segmented2` differs from `segmented` in parameterization.
- **b.transition**: Numeric. Controls whether threshold model or smooth transition model. Default to Inf, which corresponds to threshold model
- **est.method**: string. Better leave it at NULL. grid: grid search; smoothapprox: smooth approximation
- **var.type**: string. Different methods for estimating covariance matrix and constructing confidence intervals
- **aux.fit**: a model fit object that is needed for model-robust estimation of covariance matrix
- **grid.search.max**: integer.
- **test.inv.ci**: Boolean, whether or not to find test-inversion confidence interval for threshold
- **ci.bootstrap.size**: integer, number of bootstrap
- **alpha**: double, nominal type I error rate
- **save.boot**: Boolean, whether to save bootstrap samples
- **lb.quantile**: lower bound of the search range for change point estimate
- **ub.quantile**: upper bound of the search range for change point estimate
- **tol**: Numeric. Stopping criterion on the coefficient estimate.
- **maxit**: integer. Maximum number of iterations in the outer loop of optimization.
- **chngpt.init**: numeric. Initial value for the change point.
- **weights**: passed to `glm`
- **verbose**: Boolean.
- **add.points**: Boolean.
- **add.ci**: Boolean.
- **add**: Boolean.
- **breaks**: integer.
- **keep.best.fit**: Boolean.
- **y**: outcome
- **show.slope.post.threshold**: boolean
- **x**: `chngptm` fit object.
Object `chngptm` fit object.

... arguments passed to glm or coxph

`m.out.of.n` whether to perform m out of n bootstrap

`boot.test.inv.ci` whether to get test inversion CI under bootstrap

`search.bound` bounds for search for sloping parameters

`which` an integer

`xlim` xlim

`lwd` lwd

`lcol` line col

`comb` a vector of combination coefficients that will be used to form an inner product with the estimated slope

`expo` If family is binomial and expo is TRUE, coefficients summary will be shown on the scale of odds ratio instead of slopes

Details

Without lb.quantile and ub.quantile, finite sample performance of estimator drops considerably! When est.method is smoothapprox, Newton-Raphson is done with initial values chosen by change point hypothesis testing. The testing procedure may be less subjective to finite sample volatility.

If var.method is bootstrap, summary of fitted model contains p values for each estimated slope. These p values are approximate p-values, obtained assuming that the bootstrap distributions are normal.

Value

A an object of type chngptm with the following components

- `converged` Boolean
- `coefficients` vector. Estimated coefficients. The last element, named ".chngpt", is the estimated change point
- `test` `htest`. Max score test results
- `iter` integer. Number of iterations

References


Examples

# threshold linear regression
dat.1=sim.chngptm("thresholded", "segmented", n=200, seed=1, beta=1, alpha=-1, x.distr="norm", e.=4, family="gaussian")
# default search method is fastgrid
fit.1=chngptm(formula.1=y~z, formula.2=-x, family="gaussian", dat.1, type="segmented", 
est.method="fastgrid", var.type="bootstrap", ci.bootstrap.size=10)
summary(fit.1)
head(resid(fit.1))
plot(fit.1)
# with weights
fit.1.a=chngptm(formula.1=y~z, formula.2=-x, family="gaussian", dat.1, type="segmented", 
est.method="fastgrid", var.type="bootstrap", weights=ifelse(dat.1$x<3.5,100,1)
, ci.bootstrap.size=10)
summary(fit.1.a)
plot(fit.1.a)

# threshold logistic regression
dat.2=sim.chngptm("thresholded", "step", n=200, seed=1, beta=1, alpha=-1, x.distr="norm", e.=4,
family="binomial")
fit.2=chngptm(formula.1=y~z, formula.2=-x, family="binomial", dat.2, type="step", est.method="grid")
summary(fit.2)
# no variance estimates available for discontinuous threshold models such as step
# vcov(fit.2$best.fit) gives the variance estimates for the best model conditional on threshold est

# also supports cbind() formula on left hand side
set.seed(1)
dat.2$succ=rbinom(nrow(dat.2), 10, 1/(1 + exp(-dat.2$eta)))
dat.2$fail=10-dat.2$succ
fit.2a=chngptm(formula.1=cbind(succ,fail)~z, formula.2=-x, family="binomial", dat.2, 
type="step")

# threshold Cox regression, no variance estimates available
library(survival)
dat.3=data.frame(time=c(4,3,1,1,2,2,3), status=c(1,1,1,0,1,1,0), x=c(0,0,0,0,1,1,1))
fit.3=chngptm(formula.1=Surv(time, status)~1, formula.2=-x, data=dat.3, family="coxph", type="step")
summary(fit.3)

# Poisson
counts = c(18,17,15,20,10,20,25,13,12)
outcome = as.integer(gl(3,1,9))
treatment = gl(3,3)
print(d.AD <- data.frame(treatment, outcome, counts))
fit.4=chngptm(formula.1=counts ~ treatment, formula.2=-outcome, data=d.AD, family="poisson", 
type="segmented", var.type="bootstrap", verbose=1, ci.bootstrap.size=10)
## coef.0.ls

```r
summary(fit.4)
```

```r
## Not run:
# not run because otherwise the examples take >5s and that is a problem for R CMD check

# an interaction model. est.method will be set to grid inside the function. fastgrid not available for models with interaction terms yet
fit=chngptm(formula.l=mpg~cyl + disp + hp, formula.z=-hp+drat, mtcars, type="segmented", family="gaussian", var.type="bootstrap", ci.bootstrap.size=100)
summary(fit)
```

## End(Not run)

---

### coef.0.ls

#### Simulation Study Parameters

**Description**

The true parameters used in the simulation studies.

**Usage**

data("coef.0.ls")

**Format**

The format is: List of 3 $ segmented : List of 32 ...

- **quadratic2b_norm**: Named num [1:5] 0 1 0 0 0...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **cubic2b_lin**: Named num [1:5] 0 1 7 0 0...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2_norm**: Named num [1:5] -0.0943 0.3365 0.4005 -0.9163 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b_norm**: Named num [1:5] -0.0943 0.3365 0.4005 0 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b1_norm**: Named num [1:5] -0.0943 0.3365 0.4005 -0.2231 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b2_norm**: Named num [1:5] -0.0943 0.3365 0.4005 -0.5108 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b3_norm**: Named num [1:5] -0.0943 0.3365 0.4005 -0.0513 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b4_norm**: Named num [1:5] -0.0943 0.3365 0.4005 -0.1625 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b5_norm**: Named num [1:5] -0.0943 0.3365 0.4005 -0.1054 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b6_norm**: Named num [1:5] -0.0943 0.3365 0.4005 0.0198 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b7_norm**: Named num [1:5] -0.0943 0.3365 0.4005 0.0392 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...

- **sigmoid2b8_norm**: Named num [1:5] -0.0943 0.3365 0.4005 0.0198 4.5...
  - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+"...
coef.0.ls

0.3365 0.4005 0.0583 4.5 ... attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" 

sigmoid2b9_norm: Named num [1:5] -0.0943 0.3365 0.4005 0.9163 4.5 ...

sigmoid2b10_norm: Named num [1:5] -0.0943 0.3365 0.4005 0.3924 4.5 ...

sigmoid2b16_norm: Named num [1:5] -0.0943 0.3365 0.4005 0.0198 4.5 ...

sigmoid2b17_norm: Named num [1:5] -0.0943 0.3365 0.4005 0.0392 4.5 ...

sigmoid2b18_norm: Named num [1:5] -0.0943 0.3365 0.4005 0.0583 4.5 ...

sigmoid2b21_norm: Named num [1:5] 0.3365 0.4 0.4 0.48 ...

sigmoid2_gam: Named num [1:5] -1.33 0.336 0.4 -0.916 2.2 ...

sigmoid2_gam1: Named num [1:5] -1 0.336 0.4 -0.916 1.5 ...

sigmoid2_gam2: Named num [1:5] -0.5 0.336 -0.916 2.2 ...

sigmoid2a_gam: Named num [1:5] -0.5 0.336 -0.916 2.2 ...

quadratic_gam: Named num [1:5] -1.6355 0.3363 -0.0398 1.4869 2.8154 ...

exp_gam: Named num [1:5] -2.753 0.336 0.513 0.936 3.607 ...

quadratic_norm_gaussian: Named num [1:5] -3.735 0.336 0.898 1.845 4.7 ...

quadratic_norm: Named num [1:5] -2.83 0.338 0.553 1.341 3.754 ...

exp_norm: Named num [1:5] -6.235 0.337 1.012 1.325 5.057 ...

hinge: List of 29 ...

sigmoid2b2_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b3_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b4_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b5_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b6_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b7_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b8_norm: Named num [1:5] 0 0 0 0 0 ...

sigmoid2b9_norm: Named num [1:5] 0 0 0 0 0 ...
```
coefs.0_ls

<table>
<thead>
<tr>
<th>Term</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>sigmoid2b10_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>sigmoid2b16_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>sigmoid2b17_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>sigmoid2b18_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>smooth2b_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>smooth2b1_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>smooth2b2_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>smooth2b3_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>smooth2b4_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>smooth2b5_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>quadratic_gam</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>quadratic2b_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b_lin</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b2_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b3_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b4_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b5 norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b6_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b7_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b8_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b9_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b10_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
<tr>
<td>cubic2b16_norm</td>
<td>(x-chngpt)+ 0.3365 0.4005 0.45 0.9163 4.5...</td>
</tr>
</tbody>
</table>
```

smooth2b_norm : Named num [1:5] 0 0.336 0.4 0.8 .. - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. smooth2b1_norm : Named num [1:5] 0 0.336 0.4 0.005 4.8 .. - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. smooth2b2_norm : Named num [1:5] 0 0.336 0.4 0.01 4.8 .. - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. smooth2b3_norm : Named num [1:5] 0 0.336 0.4 0.02 4.8 .. - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. smooth2b4_norm : Named num [1:5] 0 0.336 0.4 0.05 4.8 .. - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. smooth2b5_norm : Named num [1:5] 0 0.336 0.4 0.1 4.8 .. - attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" ..

dat.mtct

Description
A dataset from the immune correlates study of Maternal To Child Transmission of HIV-1

Usage
data("dat.mtct")

Format
A data frame with 236 observations on the following 3 variables.

y a numeric vector
birth a factor with levels C-section Vaginal
NAb_SF162LS a numeric vector

References

dat.mtct.2

Description
A dataset from the immune correlates study of Maternal To Child Transmission of HIV-1

Usage
dat.mtct.2
Format

A data frame with 248 observations on the following 2 variables.

NAb_score  a numeric vector
V3_BioV3B  a numeric vector

References


hinge.test A non-nested hypothesis testing problem for threshold regression models

Description

Test a hinge model against a linear model.

Usage

hinge.test(formula, cov.interest, family = c("binomial", "gaussian"), data, thres = NA, lb.quantile = 0.1, ub.quantile = 0.9, chngpts.cnt = 10, method = c("FDB", "B", "DB"), boot.B = 10000, B2 = NA, verbose = FALSE)

Arguments

formula
cov.interest
family
data
thres If supplied, this will be the threshold value to use in the hinge model.
lb.quantile lower bound of threshold candidates in quantile
ub.quantile upper bound of threshold candidates in quantile
chngpts.cnt number of candidate thresholds
method type of test. FDB: false double bootstrap, B: parametric bootstrap, DB: double bootstrap.
boot.B number of parametric bootstrap replicates for B and FDB
B2 number of inner bootstrap replicates for DB
verbose

hingeNtest A non-nested hypothesis testing problem for threshold regression models

Description

Test a hinge model against a linear model.

Usage

hingeNtest(formula, covNinterest, family = c("binomial", "gaussian"), data, thres = NA, lb.quantile = 0.1, ub.quantile = 0.9, chngpts.cnt = 10, method = c("FDB", "B", "DB"), bootNb = 1PPPPL bR = naL verbose = falseI

Arguments

formula
cov.interest
family
data
thres If supplied, this will be the threshold value to use in the hinge model.
lb.quantile lower bound of threshold candidates in quantile
ub.quantile upper bound of threshold candidates in quantile
chngpts.cnt number of candidate thresholds
method type of test. FDB: false double bootstrap, B: parametric bootstrap, DB: double bootstrap.
bootNb number of parametric bootstrap replicates for B and FDB
bR number of inner bootstrap replicates for DB
verbose
Value

A list of class htest

- `p.value`: P-value
- `chngpts`: Vector of change points evaluated
- `tt`: Standardized absolute score statistics
- `V.S.hat`: Estimated variance-covariance matrix of the score statistics

Author(s)

Zonglin He

References

He, Fong, Fouda, Permar. A non-nested hypothesis testing problem for threshold regression model, under review

Examples

```r
dat=sim.hinge(threshold.type = 'NA', family = 'binomial', thres='NA', X.ditr = 'norm', mu.X = c(0,0,0),
    coef.X = c(0,5,5,4), cov.X = diag(3), eps.sd = 1, seed = 1, n=100)
test=hinge.test(Y-X1+X2, "x", family="binomial", data=dat,'method'='FDB',boot.B=10)
test
```

performance.unit.test  Perform unit testing for performance evaluation.

Description

This function performs unit testing for performance evaluation.

Usage

```r
performance.unit.test(formula.1, formula.2, family, data, B, I)
```

Arguments

- `formula.1`
- `formula.2`
- `family`
- `data`
- `B`
- `I`
**Description**

Simulation Parameters

**Usage**

data(sim.alphas)

**Format**

List of 6. Names: sigmoid2_norm, sigmoid2_norm3, sigmoid3_norm, sigmoid3_norm3, sigmoid4_norm, sigmoid4_norm3. Each element is a 5x4 matrix

---

**Description**

Generate simulation datasets for change point Monte Carlo studies.

**Usage**

```r
sim.chngpt (mean.model = c("thresholded", "thresholdedItxn", "quadratic", "quadratic2b", "cubic2b", "exp", "flatHyperbolic", "z2", "z2hinge", "z2segmented", "z2linear"), threshold.type = c("NA", "step", "hinge", "segmented", "segmented2", "segmented", "upperhinge"), b.transition = Inf, family = c("binomial", "gaussian"), x.distr = c("norm", "norm3", "norm6", "imb", "lin", "mix", "gam", "zbinary", "gam1", "gam2", "fixnorm"), e. = NULL, mu.x = 4.7, sd.x = NULL, sd = 0.3, mu.z = 0, alpha = NULL, alpha.candidate = NULL, coef.z = log(1.4), beta = NULL, beta.itxn = NULL, n, seed, weighted = FALSE, heteroscedastic = FALSE, verbose = FALSE)
```
Arguments

threshold.type  string. Types of threshold effect to simulate, only applicable when label does not start with sigmoid.

family  string. Glm family.

n
mu.z

seed

weighted

beta

coef.z  numeric. Coefficient for z.

beta.itxn  numeric. Coefficient for z.

alpha  numeric, intercept.

mu.x  numeric

sd.x  numeric

mean.model  numeric

x.distr  string. Possible values: norm (normal distribution), gam (gamma distribution). gam1 is a hack to allow e. be different

e.

verbose  Boolean

b.transition

sd

alpha.candidate  Candidate values of alpha, used in code to determine alpha values

heteroscedastic  Boolean.

Details

mean.model, threshold.type and b.transition all affect mean models.

Value

A data frame with following columns:

y  0/1 outcome

x  observed covariate that we are interested in

x.star  unobserved covariate that underlies x

z  additional covariate

In addition, columns starting with ‘w’ are covariates that we also adjust in the model; columns starting with ‘x’ are covariates derived from x.
**Examples**

```r
seed=2
par(mfrow=c(2,2))
dat=sim.chngpt(mean.model="thresholded", threshold.type="hinge", family="gaussian", beta=0, n=200, seed=seed, alpha=1, x.distr="norm", e.=4, heteroscedastic=FALSE)
plot(y~z, dat)
dat=sim.chngpt(mean.model="thresholded", threshold.type="hinge", family="gaussian", beta=0, n=200, seed=seed, alpha=1, x.distr="norm", e.=4, heteroscedastic=TRUE)
plot(y~z, dat)
dat=sim.chngpt(mean.model="z2", threshold.type="hinge", family="gaussian", beta=1, n=200, seed=seed, alpha=1, x.distr="norm", e.=4, heteroscedastic=FALSE)
plot(y~z, dat)
dat=sim.chngpt(mean.model="z2", threshold.type="hinge", family="gaussian", beta=1, n=200, seed=seed, alpha=1, x.distr="norm", e.=4, heteroscedastic=TRUE)
plot(y~z, dat)
```

---

**Description**

Simulate data for Monte Carlo study.

**Usage**

```r
```

**Arguments**

- `threshold.type`
- `family`
- `thres`
- `X.ditr`
- `mu.X`
- `coef.X`
- `cov.X`
- `eps.sd`
- `seed`
- `n`
simNmy

Simulate data

Description

Simulate data

Usage

```r
simNmy(n, seed, label, alpha, beta, eN = NULL, bN = NULL, trN = NULL)
```

Arguments

- `n` Sample size
- `seed` Seed for random number generator
- `label` A character string which specifies the simulation scenario. sigmoid4, sigmoidgam4, elbow4
- `alpha` regression parameter
- `beta` regression parameter
- `eN` inflection point for the logistic transformation (the log scale)
- `bN` slope for the logistic transformation
- `trN` threshold point

Details

When the label starts with elbow, the transformation on x.star is elbow shaped. When the label starts with sigmoid, the transformation on x.star is sigmoid shaped. Data simulated from \( \logit(Pr(Y=1)) = \alpha + \beta \times (\text{transformed } x\text{.star}) \).

Value

A data frame with columns: y, x.star, x.star.expit (if label starts with sigmoid), x.star.tr (if label starts with elbow), x.bin.med (x.star dichotomized at median), x.tri (x.star trichotomized at tertiles).

Examples

```r
alpha=-1; beta=log(0.2)
e=5; b=-30; t=1
dat=simNmy(n=250, seed=1, label="sigmoid4", alpha, beta, e=e, b=b)
```
Simulate data according to one of the scenarios considered in Pastor-Barriuso et al 2003

Description

Simulate data according to one of the scenarios considered in Pastor-Barriuso et al 2003

Usage

sim.pastor(seed)

Arguments

seed Seed for the random number generator.

Value

A data frame with columns: y, x.star, x.star.expit, and x.bin.med (x.star dichotomized at median).

Examples

dat=sim.pastor(seed=1)
Index

*Topic \textasciitilde kwd1
  performance.unit.test, 14
*Topic \textasciitilde kwd2
  performance.unit.test, 14
*Topic \textbf{distribution}
  chngpt, 2

antoch.test (chngpt.test), 2

chngpt, 2
chngpt.test, 2
chngptm, 5
coef.0.1s, 9
coef.chngptm(chngptm), 5

dat.mtct, 12
dat.mtct.2, 12

hinge.test, 13

lincomb (chngptm), 5

performance.unit.test, 14
plot.chngpt.test(chngpt.test), 2
plot.chngptm(chngptm), 5
print.chngptm(chngptm), 5

residuals.chngptm(chngptm), 5

sim.alphas, 15
sim.chngpt, 15
sim.hinge, 17
sim.my, 18
sim.pastor, 19
summary.chngptm(chngptm), 5

vcov.chngptm(chngptm), 5