Package `chngpt’

May 13, 2020

**LazyLoad** yes

**LazyData** yes

**Version** 2020.5-13

**Title** Estimation and Hypothesis Testing for Threshold Regression

**Maintainer** Youyi Fong &lt;youyifong@gmail.com&gt;

**Depends** R (>= 3.5.0)

**Suggests** R.rsp, RUnit, mvtnorm

**Imports** survival, splines, kyotil, boot, MASS, methods

**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, with and without interaction terms. 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],

Adam Elder [aut],

Hyunju Son [aut],

Zonglin He [aut],

Tao Yang [aut]

**Repository** CRAN

**Date/Publication** 2020-05-13 18:20:13 UTC

**R topics documented:**

chngpt ............................................................... 2
chngpt.test ....................................................... 2
chngptom ........................................................... 5
coeff.0.ls .......................................................... 10
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)

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

### 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.

**by.percentile**  
... arguments passed to or from methods.

**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-e)_+.

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

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)

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)

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

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

## End(Not run)

---

**chngptm**  
*Estimate change point logistic model*

### Description

Estimate change point logistic model

### Usage

```r
chngptm(formula.1, formula.2, family, data, type = c("hinge", "M02", "M03", "M04", "upperhinge", "M20", "M30", "M40", "M21", "M12", "M21c", "M12c", "M22", "M22c", "M31", "M13", "M33c", "segmented", "segmented2", "step", "stegmented"), formula.strat = NULL, weights = NULL, offset = NULL, est.method = c("default", "fastgrid2", "fastgrid", "grid", "smoothapprox"), var.type = c("default", "none", "robust", "model", "bootstrap", "all"), aux.fit = NULL, lb.quantile = 0.1, ub.quantile = 0.9, grid.search.max = Inf, test.inv.ci = TRUE, boot.test.inv.ci = FALSE, ci.bootstrap.size = 1000, alpha = 0.05, save.boot = TRUE, m.out.of.n = 0, subsampling=0, b.transition = Inf, tol = 1e-04, maxit = 100, chngpt.init = NULL, search.bound = 10, keep.best.fit = TRUE, ncpus = 1, verbose = FALSE, ...)
```

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

## S3 method for class 'chngptm'
```r
test
plot(test)
```
chngptm

## S3 method for class 'chngptm'
vcov(object, var.type=NULL, ...)

## S3 method for class 'chngptm'
print(x, ...)

## S3 method for class 'chngptm'
plot(x, which = NULL, xlim = NULL, lwd = 2, lcol = "red", 
     lty = 1, add = FALSE, add.points = TRUE, add.ci = 
     TRUE, breaks = 20, mark.chngpt = FALSE, xlab = NULL, 
     ylab = NULL, ...)

## S3 method for class 'chngptm'
summary(object, var.type = NULL, expo = FALSE, 
show.slope.post.threshold = FALSE, verbose = FALSE, 
boot.type = "symm", ...)

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 es-
            timate 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 parameteriza-
            tion.
b.transition Numeric. Controls whether threshold model or smooth transition model. De-
            fault to Inf, which corresponds to threshold model
est.method  default: estimation algorithm will be chosen optimally; fastgrid2: a super fast 
            grid search algorithm, limited to linear regression; grid: plain grid search, works 
            for almost all models; smoothapprox: approximates the likelihood function using 
            a smooth function, only works for some models. fastgrid = fastgrid2, kept 
            for backward compatibility
var.type    string. Different methods for estimating covariance matrix and constructing con-
            fidence intervals
aux.fit     a model fit object that is needed for model-robust estimation of covariance ma-
            trix
grid.search.max The maximum number of grid points used in grid search. When doing fast grid 
            search, grid.search.max is set to Inf internally because it does not take more time 
            to examine all potential thresholds.
test.inv.ci  Boolean, whether or not to find test-inversion confidence interval for threshold 
ci.bootstrap.size integer, number of bootstrap
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alpha</td>
<td>double, normal type I error rate</td>
</tr>
<tr>
<td>save.boot</td>
<td>Boolean, whether to save bootstrap samples</td>
</tr>
<tr>
<td>lb.quantile</td>
<td>lower bound of the search range for change point estimate</td>
</tr>
<tr>
<td>ub.quantile</td>
<td>upper bound of the search range for change point estimate</td>
</tr>
<tr>
<td>tol</td>
<td>Numeric. Stopping criterion on the coefficient estimate.</td>
</tr>
<tr>
<td>maxit</td>
<td>integer. Maximum number of iterations in the outer loop of optimization.</td>
</tr>
<tr>
<td>chngpt.init</td>
<td>numeric. Initial value for the change point.</td>
</tr>
<tr>
<td>weights</td>
<td>passed to glm</td>
</tr>
<tr>
<td>verbose</td>
<td>Boolean.</td>
</tr>
<tr>
<td>add.points</td>
<td>Boolean.</td>
</tr>
<tr>
<td>add.ci</td>
<td>Boolean.</td>
</tr>
<tr>
<td>add</td>
<td>Boolean.</td>
</tr>
<tr>
<td>breaks</td>
<td>integer.</td>
</tr>
<tr>
<td>ncpus</td>
<td>Number of cores to use if the OS is not Windows.</td>
</tr>
<tr>
<td>keep.best.fit</td>
<td>Boolean.</td>
</tr>
<tr>
<td>y</td>
<td>outcome</td>
</tr>
<tr>
<td>x</td>
<td>chngptm fit object</td>
</tr>
<tr>
<td>object</td>
<td>chngptm fit object</td>
</tr>
<tr>
<td>...</td>
<td>arguments passed to glm or coxph</td>
</tr>
<tr>
<td>m.out.of.n</td>
<td>sample size for m-out-of-n bootstrap, default 0 for not doing this type of bootstrap</td>
</tr>
<tr>
<td>subsampling</td>
<td>sample size for subsampling bootstrap, default 0 for not doing this type of bootstrap</td>
</tr>
<tr>
<td>boot.test.inv.ci</td>
<td>whether to get test inversion CI under bootstrap</td>
</tr>
<tr>
<td>search-bound</td>
<td>bounds for search for sloping parameters</td>
</tr>
<tr>
<td>which</td>
<td>an integer</td>
</tr>
<tr>
<td>xlim</td>
<td>xlim</td>
</tr>
<tr>
<td>lwd</td>
<td>lwd</td>
</tr>
<tr>
<td>lcol</td>
<td>line col</td>
</tr>
<tr>
<td>mark.chngpt</td>
<td>mark.chngpt</td>
</tr>
<tr>
<td>xlab</td>
<td>xlab</td>
</tr>
<tr>
<td>ylab</td>
<td>ylab</td>
</tr>
<tr>
<td>offset</td>
<td>offset</td>
</tr>
<tr>
<td>lty</td>
<td>lty</td>
</tr>
<tr>
<td>boot.type</td>
<td>lty</td>
</tr>
<tr>
<td>comb</td>
<td>a vector of combination coefficients that will be used to form an inner product with the estimated slope</td>
</tr>
<tr>
<td>expo</td>
<td>If family is binomial and expo is TRUE, coefficients summary will be shown on the scale of odds ratio instead of slopes</td>
</tr>
</tbody>
</table>
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.

When var.method is bootstrap and the OS is not Windows, the boot package we use under the hood takes advantage of ncpus cores through parallel::.mclapply.

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
types=c("hinge", "segmented", "M02", "M03")
for (type in types) {
  fit=chngptm(formula.1=logratio~1, formula.2=~range, lidar, type=type, family="gaussian",
               var.type="bootstrap")
  summary(fit)
  plot(fit)
}

# with weights
dat.1=sim.chngpt("thresholded", "segmented", n=200, seed=1, beta=1, alpha=-1, x.distr="norm", e.=4,
                 family="gaussian")
fit.1=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))
# threshold logistic regression
dat.2=sim.chngpt("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)

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

# Poisson example
counts <- c(18,17,15,20,10,25,13,12,33,35)
x <- 1:length(counts)
print(d.AD <- data.frame(x, counts))
fit.4=chngptm(formula.1=counts ~ 1, formula.2=~x, data=d.AD, family="poisson", type="segmented", var.type="bootstrap", verbose=1, ci.bootstrap.size=1)
summary(fit.4)

# coxph example
library(survival)
fit=chngptm(formula.1=Surv(time, status) ~ ph.ecog, formula.2=~age, data=lung, family="coxph", type="segmented", var.type="bootstrap", ci.bootstrap.size=10)
summary(fit)

# one interaction term (mtcars is part of R default installation)
# est.method will be grid as fastgrid not available for models with interaction terms yet
fit=chngptm(formula.1=mpg ~ hp, formula.2=~hp*drat, mtcars, type="segmented", family="gaussian", var.type="bootstrap", ci.bootstrap.size=10)
summary(fit)

# interaction, upperhinge model, bootstrap
fit=chngptm(formula.1=mpg ~ hp, formula.2=~hp*drat, mtcars, type="M10", family="gaussian", var.type="bootstrap", ci.bootstrap.size=10)
summary(fit)

# more than one interaction term
# subsampling bootstrap confidence interval for step model
fit=chngptm(formula.1=mpg~hp+wt, formula.2=-hp*drat+wt*drat, mtcars, type="step",
family="gaussian", var.type="bootstrap", ci.bootstrap.size=10)
summary(fit)

# step model, subsampling bootstrap confidence intervals
fit=chngptm(formula.1=mpg~hp, formula.2=-drat, mtcars, type="step",
family="gaussian", var.type="bootstrap", ci.bootstrap.size=10, verbose=TRUE)
summary(fit)

# higher order threshold models
dat=sim.chngpt(mean.model="thresholded", threshold.type="M22", n=500, seed=1,
beta=c(32,2,10, 10), x.distr="norm", e.=6, b.transition=Inf, family="gaussian",
alpha=0, sd=0, coef.z=0)
fit.0=chngptm(formula.1=y~z, formula.2=-x, dat, type="M22", family="gaussian",
est.method="fastgrid2"); plot(fit.0)

dat=sim.chngpt(mean.model="thresholded", threshold.type="M22c", n=500, seed=1,
beta=c(32,2,32, 10), x.distr="norm", e.=6, b.transition=Inf, family="gaussian",
alpha=0, sd=0, coef.z=0)
fit.0=chngptm(formula.1=y~z, formula.2=-x, dat, type="M22c", family="gaussian",
est.method="fastgrid2"); plot(fit.0)

# examples of aux.fit
fit.0=glm(y~zz+ns(xx,df=3), data, family="binomial")
fit = chngptm (formula.1=yy~zz, formula.2="xx, family="binomial", data, type="hinge",
est.method="smoothapprox", var.type="all", verbose=verbose, aux.fit=fit.0,
lb.quantile=0.1, ub.quantile=0.9, tol=1e-4, maxit=1e3)

## End(Not run)

development version

---

**Simulation Study Parameters**

### Description

The true parameters used in the simulation studies.

### Usage

data("coef.0.ls")
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)+" .. -..$ sigmoid2b_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)+" .. -..$ 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 -0.1054 4.5 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2b5_norm : Named num [1:5] -0.0943 0.3365 0.4005 -0.1625 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 0.0583 4.5 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2b9_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)+" .. -..$ sigmoid2b10_norm : Named num [1:5] -0.0943 0.3365 0.4005 0.005 4.5 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2b16_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)+" .. -..$ sigmoid2b17_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)+" .. -..$ sigmoid2b18_norm : Named num [1:5] -0.0943 0.3365 0.4005 0.0583 4.5 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ smooth2b_norm : Named num [1:5] 0.336 0.4 0.48 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ smooth2b1_norm : Named num [1:5] 0.336 0.4 0.005 4.8 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ smooth2b2_norm : Named num [1:5] 0.336 0.4 0.01 4.8 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ smooth2b3_norm : Named num [1:5] 0.336 0.4 0.02 4.8 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ smooth2b4_norm : Named num [1:5] 0.336 0.4 0.05 4.8 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ smooth2b5_norm : Named num [1:5] 0.336 0.4 0.1 4.8 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2_gam : Named num [1:5] 0.336 0.4 -0.916 2.2 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2_gam1 : Named num [1:5] -1 0.336 0.4 -0.916 1.5 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2_gam2 : Named num [1:5] -0.6 0.336 0.4 -0.916 1 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ sigmoid2a_gam : Named num [1:5] -0.5 0.336 0 -0.916 2.2 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ quadratic_gam : Named num [1:5] -1.6355 0.3363 -0.0398 1.4869 2.8154 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ exp_gam : Named num [1:5] -2.753 0.336 0.513 0.936 3.607 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ quadratic_norm_gaussian: Named num [1:5] -3.735 0.336 0.898 1.845 4.7 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ quadratic_norm : Named num [1:5] -2.83 0.338 0.553 1.341 3.754 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ exp_norm :Named num [1:5] -6.235 0.337 1.012 1.325 5.057 .. -attr(*, "names")= chr [1:5] "(Intercept)" "z" "x" "(x-chngpt)+" .. -..$ hinge :List of 29 .
<table>
<thead>
<tr>
<th>Variable</th>
<th>Coefficient</th>
<th>Coefficient</th>
<th>Coefficient</th>
<th>Coefficient</th>
<th>Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1.0</td>
<td>0.3</td>
<td>0.5</td>
<td>-1.0</td>
<td>4.5</td>
</tr>
<tr>
<td>z</td>
<td>0.2</td>
<td>0.6</td>
<td>0.3</td>
<td>-0.1</td>
<td>1.5</td>
</tr>
<tr>
<td>x</td>
<td>0.4</td>
<td>0.2</td>
<td>0.7</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>(x-chngpt+)</td>
<td>0.5</td>
<td>0.3</td>
<td>0.4</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>cubic2_lin</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>sigmoid2_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>-0.9163</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0198</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b1_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0392</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b2_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0583</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b3_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.1625</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b4_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0198</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b5_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0392</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b6_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0583</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b7_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0198</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b8_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0392</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b9_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0583</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b10_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0198</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b11_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0392</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b12_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0583</td>
<td>4.5</td>
</tr>
<tr>
<td>sigmoid2b13_norm</td>
<td>-0.0943</td>
<td>0.3365</td>
<td>0.4005</td>
<td>0.0198</td>
<td>4.5</td>
</tr>
</tbody>
</table>

cof.0.ls

\[
\text{cof.0.ls} = \begin{align*}
\text{Intercept} & : 1.0 \\
\text{z} & : 0.2 \\
\text{x} & : 0.4 \\
\text{(x-chngpt+)} & : 0.5 \\
\text{cubic2_lin} & : 0.5 \\
\text{sigmoid2_norm} & : -0.0943 \\
\text{sigmoid2b_norm} & : -0.0943 \\
\text{sigmoid2b1_norm} & : -0.0943 \\
\text{sigmoid2b2_norm} & : -0.0943 \\
\text{sigmoid2b3_norm} & : -0.0943 \\
\text{sigmoid2b4_norm} & : -0.0943 \\
\text{sigmoid2b5_norm} & : -0.0943 \\
\text{sigmoid2b6_norm} & : -0.0943 \\
\text{sigmoid2b7_norm} & : -0.0943 \\
\text{sigmoid2b8_norm} & : -0.0943 \\
\text{sigmoid2b9_norm} & : -0.0943 \\
\text{sigmoid2b10_norm} & : -0.0943 \\
\text{sigmoid2b11_norm} & : -0.0943 \\
\text{sigmoid2b12_norm} & : -0.0943 \\
\text{sigmoid2b13_norm} & : -0.0943 \\
\end{align*}
\]
convert.coef

Helper functions

Description

Some helper functions.

Usage

convert.coef(coef.0, threshold.type)

predictx(fit, boot.type, alpha = 0.05, xx = NULL, verbose = FALSE, return.boot = FALSE, include.intercept=FALSE)

threshold.func(threshold.type, coef, xx, x.name, include.intercept=FALSE)

Arguments

include.intercept
**dat.mtct**

- **coef.0**
- **threshold.type**
- **return.boot**
- **fit**
- **boot.type**
- **alpha**
- **verbose**
- **coef**
- **xx**
- **x.name**

---

**dat.mtct**  
An Example Dataset

---

**Description**

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

**Usage**

```r
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

**An Example Dataset**

**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**

### double.hinge

**Fit Double Hinge Models**

**Description**
Fit double hinge models.

**Usage**

double.hinge(x, y, lower.y = NULL, upper.y = NULL, var.type = c("none", "bootstrap"), ci.bootstrap.size = 1000, alpha = 0.05, save.boot = TRUE, ncpus = 1)

## S3 method for class 'double.hinge'
plot(x, which = NULL, xlim = NULL, lwd = 2, lcol = "red", lty = 1, add.points = TRUE, add.ci = TRUE, breaks = 20, mark.chngpt = FALSE, xlab = NULL, ylab = NULL, ...)

## S3 method for class 'double.hinge'
fitted(object, ...)

## S3 method for class 'double.hinge'
residuals(object, ...)
**Arguments**

- object
- x
- y
- lower.y
- upper.y
- var.type
- ci.bootstrap.size
- alpha
- save.boot
- ncpus
- lcol
- lwd
- 'which'
- 'xlim'
- 'lty'
- 'add.points'
- 'add.ci'
- 'breaks'
- 'mark.chngpt'
- 'xlab'
- 'ylab'
- ... arguments passed along

**Details**

If lower.y and upper.y are not supplied, \( \min(y) \) is taken as the function value when \( x \) is less than or equal to the first threshold, and \( \max(y) \) is taken as the function value when \( x \) is greater than or equal to the second threshold.

If the function is expected to be decreasing between the two thresholds, lower.y and upper.y should be supplied to ensure the correct fit.

mse is residual sum of squares
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

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

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

lidar          Light Detection and Ranging Data

Description

LIDAR

Usage

data("lidar")

Format

A data frame with 221 observations on the following 2 variables.

range    a numeric vector
logratio a numeric vector

Source

**nutrition**  

---

**Infant Nutrition Data**

**Description**

The infant nutrition dataset comprises data collected in a study on the nutrition of infants and preschool children in the north central region of the United States of America.

**Usage**

```r
data("nutrition")
```

**Format**

A data frame with 72 observations on the following 2 variables.

- **woh** weight/height ratio
- **age** a numeric vector

**Source**


---

**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**
**sim.chngpt**

---

### sim.alphas Simulation Parameters

**Description**

Simulation Parameters

**Usage**

```r
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

---

### sim.chngpt Simulation Function

**Description**

Generate simulation datasets for change point Monte Carlo studies.

**Usage**

```r
```
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**: numeric.
- **seed**: numeric.
- **weighted**: numeric.
- **beta**: numeric.
- **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).
- **e.**:
- **verbose**: Boolean.
- **b.transition**: numeric.
- **sd**: numeric.
- **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
cat("Examples")

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)
cat("\n")
```

---

**sim.hinge**

*Simulation function*

**Description**

Simulate data for Monte Carlo study.

**Usage**

```r
sim.hinge(threshold.type = c("NA", "hinge"), family = c("binomial", "gaussian"),
           thres = "NA", X.ditr = "norm", mu.X, coef.X, cov.X, eps.sd, seed, n)
```

**Arguments**

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

Usage

```r
sim.my(n, seed, label, alpha, beta, e. = NULL, b. = NULL, tr. = 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
- `e.`: inflection point for the logistic transformation (the log scale)
- `b.`: slope for the logistic transformation
- `tr.`: 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 

\[
\logit(Pr(Y=1)) = \alpha + \beta \cdot (\text{transformed } x.\text{star})
\]

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=sim.my(n=250, seed=1, label="sigmoid4", alpha, beta, e.=e., b.=b.)
```
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

```r
dat=sim.pastor(seed=1)
```
Index

*Topic distribution
  chngpt, 2
  antoch.test (chngpt.test), 2
  chngpt, 2
  chngpt.test, 2
  chngptm, 5
  coef.0.ls, 10
  coef.chngptm (chngptm), 5
  convert.coef, 13

  dat.mtct, 14
  dat.mtct.2, 15
  double.hinge, 15

  fitted.double.hinge (double.hinge), 15

  hinge.test, 17

  lidar, 18
  lincomb (chngptm), 5

  nutrition, 19

  performance.unit.test, 19
  plot.chngpt.test (chngpt.test), 2
  plot.chngptm (chngptm), 5
  plot.double.hinge (double.hinge), 15
  predictx (convert.coef), 13
  print.chngptm (chngptm), 5

  residuals.chngptm (chngptm), 5
  residuals.double.hinge (double.hinge), 15

  sim.alphas, 20
  sim.chngpt, 20
  sim.hinge, 22
  sim.my, 23
  sim.pastor, 24

summary.chngptm (chngptm), 5
threshold.func (convert.coef), 13
vcov.chngptm (chngptm), 5