Package 'mhtboot'

October 13, 2022

Title Multiple Hypothesis Test Based on Distribution of p Values

Version 1.3.3

Author Abhirup Mallik [aut, cre]

Maintainer Abhirup Mallik <malli066@umn.edu>

Description A framework for multiple hypothesis testing based on distribution of p values. It is well known that the p values come from different distribution for null and alternatives, in this package we provide functions to detect that change. We provide a method for using the change in distribution of p values as a way to detect the true signals in the data.

Depends R (>= 3.0.0), ggplot2, reshape2

Suggests knitr

VignetteBuilder knitr

License GPL-3

LazyData true

RoxygenNote 5.0.1

ByteCompile true

NeedsCompilation no

Repository CRAN

Date/Publication 2016-10-30 22:14:24

R topics documented:

---

datgen *datgen*

Description

Function to generate data from multivariate normal with different mean.

Usage

```
datgen(n, m, m0, sigeff, Sigma)
```

Arguments

n	number of samples
m	number of cords
m0	number of non sparse elements
sigeff	magnitude of signal
Sigma	Covariance matrix

Details

This function generates data from multivariate normal distribution with given covariance matrix. The mean values are either zero or constant sigeff, randomly permuted among the coordinates.

Value

X data matrix of size nxm

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)

## End(Not run)
```

elbow *Finding corner of a vector of ordered transformed p values*

Description

Finds corner of a vector of ordered transformed p values.

Usage

```
elbow(zvec, rbuff = 25, h = 30)
```

Arguments

zvec	vector of ordered transformed p values
rbuff	scaler, by default 25. Controls the right buffer.
h	scaler, default 30. Controls the window size.

Details

The corner point of ordered p values indicate the point where the change from the alternative to null happens. So, by detecting that point we get an estimate of the number of true alternatives.

This function uses two methods for corner detection. One method is by transforming the vectors by taking their first difference and centering them around a theoretical mean for null case. The other method is by detecting the maximum change in gradient at each point. These methods will be denoted by dav and dlm respectively.

Value

vector with two elements, containing estimates of the index of corner

$dav: by average method. $dlm: by maximum gradient method.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample(X=X,B=500,ncpus = 1)
out <- elbow(zvec = porder[1,])
out

## End(Not run)
```

hitplots — *Plot area under p value cdf below a cutoff.*

Description

Function to plot the area under the cdf below a certain cutoff.

Usage

```
hitplots(porder, alpha = 0.005)
```

Arguments

porder	the feed from porder.1sample or porder.2sample. matrix of size Bxm. of ordered transformed p values.
alpha	the cutoff of ecd, by default 0.005.

Details

The alpha parameter specifies the cutoff, the plot is the ecdf under alpha. So the right tail of the ecdf would have probability alpha.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample(X, B = 100, ncpus = 1)
hitplots(porder)

## End(Not run)
```

mht.1sample — *Multiple hypothesis testing based on p value distribution for one sample test*

Description

Implements multiple hypothesis testing based on bootstrap distribution of p values.

Usage

```
mht.1sample(X, B = 100, test = t.test, nbx = NROW(X), ncpus = 8,
  rbuff = 25, h = 30, qi = 0.9)
```

Arguments

X	matrix of data
B	bootstrap sample size, default is 100
test	one sample test. by default t.test(), user can provide own function, must return p values in $p.value
nbx	size of the bootstrap sample
ncpus	number of cpu to use
rbuff	right buffer for change detection
h	window size for change detection
qi	the quantile to use for change detection

Details

This function takes the dataset and produces the bootstrap distribtution of the transformed and ordered p values using the user given parameters. Then detects the change in the bootstrap distribution using the corner detection method. This method requires the user to specify the quantile to use for change detection. The change point is an estimate of the location of change from alternative to null and used to get the coordinates of the true signals.

Value

list with two elements. cutoff: the location of corner, signal: the index of the detected coordinates.

Examples

```
n = 50;m = 100;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
out1 <- mht.1sample(X,B=100,ncpus = 1)
out1$cutoff
out1$signal
```

---

mhtboot *mhtboot: A package for multiple hypothesis testing using bootstrap distribution of p values.*

Description

The mhtboot package provides three categories of important functions: pboot, elbow and mht.

pboot functions

pboot functions provide bootstrap distribution of p values. The pvalues are ordered and transformed. Currently the default transformation is fn(p) = -log(1-p) and in future some more transformations would be provided. There are support for two type of tests. One sample and two sample tests. The corresponding two functions are pboot.1sample and pboot.2sample. The test function by default is taken to be t.test(), while the user can provide their own test function. Both of these functions are parallelized using multicore for better performance.

elbow functions

The purpose of elbow functions is to detect the change in distribution of the ordered transfromed p values. The basic function for detecting this change is elbow(), which takes in a particular p value curve and estimates the change point. We also provide a function to process the bootstrap distribution of p values and generate the estimate of the change point corresponding to a quantile of the empirical distribution.

mht

The general function implementing the proceedure for multiple hypothesis testing based on bootstrap distribution of the p values. All the controls associated with pboot functions and elbow functions are transferred in mht functions too. There are two functions corresponding to one sample and two sample tests. These functions are mht.1sample and mht.2sample.

---

pboot.1sample *Generate Bootstrap Distribution of p values for one sample tests.*

Description

Performs bootstrap to generate empirical distribution of order statistics of p values

Usage

```
pboot.1sample(X, B = 100, test = t.test, nbx = NROW(X), ncpus = 8)
```

Arguments

X	matrix of data, each row is an independent observation nxm
B	bootstrap sample size
test	function for testing. default is t.test(). Must return a data frame with p value in $p.value.
nbx	Sample size for the bootstrap samples. Default is NROW(X), which is same as the original data sample size.
ncpus	Number of cpus to use for bootstrap. We use parallel:multicore() to parallelize the bootstrap. For windows, use ncpus = 1, for any other machine, you can use the maximum permissible number for your system.

Details

We generate the bootstrap distribution of the order statistics of the p values. We are performing one sample test on each coordinate of the original dataset. The bootstrap used here is standard version with default bootstrap sample size being equal to data sample size. The default one sample test is t.test(), however the user can provide their own test functions. The only requirement is that it must return p values in $p.value column of the output. The bootstrap is parallelized using multicore from the library parallel. Windows machines at this point does not support using multiple cores, so the ncpus option should be equal to 1 for windows. For other systems, it can be higher to speed up the process. We also use a transofrmation of the p values, by default the transformation is -log(1-p). But the user can provide their own transformation function. They should be monotonically increasing functions.

Value

matrix of dimension Bxm. (Where m coordinates), each row indicates transformed p values for that bootstrap sample.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample(X=X,B=100,ncpus = 1)
plotpboot(porder)

## End(Not run)
```

`pboot.1sample.s` *Generate p value distributions and estimate of sample correlation matrix using bootstrap.*

Description

If the user chooses to keep sout as TRUE, then this function generates bootstrap distribution of p values and returns the mean of the correlation matrices of all the bootstrap samples generated.

Usage

```
pboot.1sample.s(X, B = 100, test = t.test, nbx = NROW(X), ncpus = 8,
  sout = FALSE)
```

Arguments

X	data matrix
B	Bootstrap size
test	test to perform
nbx	bootstrap sample size, by default same as the data sample size
ncpus	number of cpus to use
sout	if correlation matrix is needed or not

Value

a list with a matrix containing the p value distributions, and another matrix of correlation matrix.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample.s(X=X,B=100,sout = TRUE,ncpus = 1)
plotpboot(porder)

## End(Not run)
```

---

pboot.2sample *Generate bootstrap distribution of p values based on user given two sample tests.*

Description

Performs bootstrap to generate empirical distribution of order statistics of p values from two sample tests.

Usage

```
pboot.2sample(X, Y, B = 100, test = t.test, nbx = NROW(X),
  nby = NROW(Y), ncpus = 8)
```

Arguments

X	matrix of data, each row is an independent observation nxm
Y	matrix of data, sample 2. each row is an independent observation nxm.
B	bootstrap sample size
test	function for testing. default is t.test(). Must return a data frame with p value in $p.value.

nbx	Sample size for the bootstrap samples. Default is NROW(X), which is same as the original data sample size.
nby	Sample size for the bootstrap samples for second dataset. Default is NROW(X), which is same as the original data sample size.
ncpus	Number of cpus to use for bootstrap. We use parallel:multicore() to parallelize the bootstrap. For windows, use ncpus = 1, for any other machine, you can use the maximum permissible number for your system.

Details

We generate the bootstrap distribution of the order statistics of the p values. We are performing one sample test on each coordinate of the original dataset. The bootstrap used here is standard version with default bootstrap sample size being equal to data sample size. The default one sample test is t.test(), however the user can provide their own test functions. The only requirement is that it must return p values in $p.value column of the output. The bootstrap is parallelized using multicore from the library parallel. Windows machines at this point does not support using multiple cores, so the ncpus option should be equal to 1 for windows. For other systems, it can be higher to speed up the process. We also use a transofrmation of the p values, by default the transformation is -log(1-p). But the user can provide their own transformation function. They should be monotonically increasing functions.

Value

matrix of dimension Bxm. (Where m coordinates), each row indicates transformed p values for that bootstrap sample.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample(X=X,B=100,ncpus = 1)
plotpboot(porder)

## End(Not run)
```

`plotchange` *plotchange*

Description

Plot the change function that is maximized to find the change point.

Usage

```
plotchange(zvec, rbuff = 25, h = 30, ...)
```

Arguments

zvec	vector of transformed order statistic of p values
rbuff	right buffer
h	window size
...	any graphical parameters passed to the plot function

Details

Currently there are two types of change functions supported. The difference between first difference series and the difference in gradients at each point. Both of these functions should have a theoretical maximum at the change point. We plot these two series side by side along with indicating the change point.

Value

Nothing

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample(X=X,B=100,ncpus = 1)
plotchange(porder[1,])

## End(Not run)
```

---

plotpboot *Quantile plots for p value distributions.*

Description

Produces density plots of quantiles of transformed order statistics of p values

Usage

```
plotpboot(porder)
```

Arguments

porder	Matrix feeds from pboot. This is a matrix of p values from the bootstrap sampls. Of size Bxm, each row for one bootstrap. The columns indicate the coordinates for testing.

Details

Plot function for pboot

This function plots the order statistics of the quantiles of the transformed p values. As the distribution of the statistic changes as the number of coordinates increase, it should show a change in the curve.

This function uses ggplot2 and reshape library to manipulate data. The final object returned is a ggplot2 image that can be fed into ggsave or any other supported functions.

Value

ggplot2 object contatining the plot.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
Y <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.2sample(X=X,Y = Y, B=100,ncpus = 1)
plotpboot(porder)

## End(Not run)
```

`ptrans` *Transformation of order statistics of the p value distributions*

Description

This function applys transformation on the bootstrap distribution of order statistics of p values.

Usage

```
ptrans(porder, trans = "default")
```

Arguments

`porder`	matrix of p value order statistics, rows indicate replicates
`trans`	one of ("default","normal","none") indicating trnasformation of -log(1-p), which is by default. Or inverse normal cdf transformation or no transformation.

Details

The transformation of p values must be monotonically increasing. The user can use their own transofrmation, however, this function supports only the commonly used transformations. These are -log(1-p) transformation, inverse normal cdf and identiy transformation.

Value

matrix with transformed distribution.

Examples

```
## Not run:
X <- datgen(n=100,m=80,m0=20,sigeff=1,Sigma = 0.25*diag(80))
porder <- pboot.1sample(X=X,B=100,ncpus = 1)
porder.tr <- ptrans(porder,trans="normal")
plotpboot(porder.tr)

## End(Not run)
```

---

qelbow *Finding corner of a quantile of ordered transformed p values*

Description

Given a matrix of empirical distribution of ordered transformed p values, this function finds the corner point for a particular quantile.

Usage

```
qelbow(porder, rbuff = 25, h = 30, qi = 0.9)
```

Arguments

porder	matrix, usually feed from pboot functions. Bxm matrix of ordered p values, where B is the replication size and m is dimension.
rbuff	right buffer, scaler, control for elbow()
h	window size, default 30.
qi	number between 0 and 1, quantile of the distribution. default 0.9.

Details

In the distribution of the transformed ordered p values, we choose a particular quantile given by the user. We estimate the change point, which will be an estimate of the number of true alternatives corresponding to that quantile of the p values. As the values of the quantile increases, the estimates can only increasing, because we are dealing with ordered p values.

Value

vector with two elements. estimates of the corner point by two methods.

Examples

```
## Not run:
n = 50;m = 250;m0 = 20;
sigeff = 1;
Sigma <- 0.25*diag(m)
X <- datgen(n,m,m0,sigeff,Sigma = Sigma)
porder <- pboot.1sample(X=X,B=100,ncpus = 1)
out <- qelbow(porder = porder)
out

## End(Not run)
```

Index