Package ‘qch’

October 13, 2022

Title Query Composed Hypotheses

Version 1.0.0

Description Provides functions for the joint analysis of K sets of p-values obtained for a same list of items. This joint analysis is performed by querying a composed hypothesis, i.e. an arbitrary complex combination of simple hypotheses, as described in Mary-Huard et al. (2021) <arXiv:2104.14601>. The null distribution corresponding to the composed hypothesis of interest is obtained by fitting non-parametric mixtures models (one for each of the simple hypothesis of the complex combination). Type I error rate control is achieved through Bayesian False Discovery Rate control. The 3 main functions of the package GetHinfo(), qch.fit() and qch.test() correspond to the 3 steps for querying a composed hypothesis (composed H0/H1 formulation, inferring the null distribution and testing the null hypothesis).

Depends R (>= 2.10)

Imports graphics, ks, mclust, stats

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Tristan Mary-Huard [aut, cre] (<`https://orcid.org/0000-0002-3839-9067`>)

Maintainer Tristan Mary-Huard `<tristan.mary-huard@agroparistech.fr>`

Repository CRAN

Date/Publication 2021-05-07 12:10:02 UTC

R topics documented:

`FastKerFdr` *FastKerFdr*

Description

FastKerFdr

Usage

```
FastKerFdr(Pval, p0 = NULL, plotting = FALSE, NbKnot = 1e+05, tol = 1e-05)
```

Arguments

`Pval`	a vector of p-values (corresponding to a p-value serie)
`p0`	a priori proportion of H0 hypotheses
`plotting`	boolean, should some diagnostic graphs be plotted. Default is FALSE.
`NbKnot`	The (maximum) number of knot for the kde procedure. Default is 1e5
`tol`	a tolerance value for convergence. Default is 1e-5

Value

A list of 3 objects. Object p0 is an estimate of the proportion of H0 hypotheses., tau is the vector of H1 posteriors. f1 is a numeric vector, each coordinate i corresponding to the evaluation of the H1 density at point pi, where pi is the ith p-value in Pval.

`GetHinfo` *Generate H0/H1 configurations and specify the ones corresponding to the composed H1*

Description

Generate H0/H1 configurations and specify the ones corresponding to the composed H1

Usage

```
GetHinfo(Q, AtLeast, Consecutive = FALSE)
```

Arguments

`Q`	number of test series to be combined
`AtLeast`	How many H1 hypotheses at least for the item to be of interest ?
`Consecutive`	Should the significant test series be consecutive ? Default=FALSE

Value

A list of two objects 'Hconfig' and 'Hconfig.H1'. Hconfig is the list of all possible combination of H0 and H1 hypotheses among Q hypotheses tested. Hconfig.H1 is the vector of components of Hconfig that correspond to the 'AtLeast' specification.

See Also

`GetHinfoEqual()`

Examples

```
GetHinfo(4,2)
```

`GetHinfoEqual` *Generate H0/H1 configurations and specify the ones corresponding to the composed H1*

Description

Generate H0/H1 configurations and specify the ones corresponding to the composed H1

Usage

```
GetHinfoEqual(Q, Equal, Consecutive = FALSE)
```

Arguments

`Q`	number of test series to be combined
`Equal`	How many H1 hypotheses exactly for the item to be of interest ?
`Consecutive`	Should the significant test series be consecutive ? Default=FALSE

Value

A list of two objects 'Hconfig' and 'Hconfig.H1'. Hconfig is the list of all possible combination of H0 and H1 hypotheses among Q hypotheses tested. Hconfig.H1 is the vector of components of Hconfig that correspond to the 'Equal' specification.

See Also

`GetHinfo()`

Examples

```
GetHinfoEqual(4,2)
```

`PvalSets` *Synthetic example to illustrate the main qch functions*

Description

PvalSets is a data.frame with 10,000 rows and 3 columns. Each row corresponds to an item, columns 'Pval1' and 'Pval2' each correspond to a test serie over the items, and column 'Class' provides the truth, i.e. if item i belongs to class 1 then the H0 hypothesis is true for the 2 tests, if item i belongs to class 2 (resp. 3) then the H0 hypothesis is true for the first (resp. second) test only, and if item i belongs to class 4 then both H0 hypotheses are false (for the first and the second test).

Usage

```
PvalSets
```

Format

A data.frame

`qch.fit` *Infer Hconfig posteriors*

Description

Infer Hconfig posteriors

Usage

```
qch.fit(pValMat, Hconfig, plotting = FALSE)
```

Arguments

`pValMat`	a matrix of p-values, each column corresponding to a p-value serie.
`Hconfig`	an Hconfig list as generated by the `GetHinfo()` function.
`plotting`	a boolean. Should some diagnostic graphs be plotted ? Default is FALSE.

Value

A list of 2 objects 'prior' and 'posterior'. Object 'prior' is a vector of estimated prior probabilities for each of the H-configurations. Object 'posterior' is a matrix providing for each item (in row) its posterior probability to belong to each of the H-configurations (in columns).

Examples

```
data(PvalSets)
PvalMat <- as.matrix(PvalSets[,-3])
## Build the Hconfig objects
Q <- 2
AtLeast <- 2
Hconfig <- GetHinfo(Q,AtLeast)$Hconfig

## Run the function
res.fit <- qch.fit(PvalMat,Hconfig)

## Display the prior of each class of items
res.fit$prior

## Display the first posteriors
head(res.fit$posterior)
```

`qch.test` *Perform composed hypothesis testing with FDR control*

Description

Perform composed hypothesis testing with FDR control

Usage

```
qch.test(posterior, Hconfig.H1, Alpha = 0.05)
```

Arguments

`posterior`	a matrix of posterior probabilities for each item to belong the different H-configurations, as provided by the `qch.fit()` function.
`Hconfig.H1`	a list of H1 config, as created by the `GetHinfo()` function.
`Alpha`	the nominal Type I error rate for FDR control.

Value

A list of 2 objects 'Rejection' and 'lFDR'. Object 'Rejection' is a vector providing for each item the result of the composed hypothesis test, after multiple testing correction. Object 'lFDR' is a vector providing for each item its local FDR estimate.

Examples

```
data(PvalSets)
PvalMat <- as.matrix(PvalSets[,-3])
Truth <- PvalSets[,3]

## Build the Hconfig objects
```

```
Q <- 2
AtLeast <- 2
Hconfig <- GetHinfo(Q,AtLeast)$Hconfig
Hconfig.H1 <- GetHinfo(Q,AtLeast)$Hconfig.H1

## Infer the posteriors
res.fit <- qch.fit(PvalMat,Hconfig)

## Run the test procedure with FDR control
res.test <- qch.test(res.fit$posterior,Hconfig.H1)
table(res.test$Rejection,Truth==4)
```

Index