Package ‘fdrci’

June 8, 2022

Type Package

Title Permutation-Based FDR Point and Confidence Interval Estimation

Version 2.3

Date 2022-6-1

Encoding UTF-8

Imports ggplot2, stats

Description FDR functions for permutation-based estimators, including pi0 as well as FDR confidence intervals. The confidence intervals account for dependencies between tests by the incorporation of an overdispersion parameter, which is estimated from the permuted data. Also included are options for an analog parametric approach.

License Artistic-2.0

LazyLoad yes

RoxygenNote 7.1.2

Suggests rmarkdown, knitr, GEOquery, dplyr, foreach

VignetteBuilder knitr

NeedsCompilation no

Author Joshua Millstein [aut, cre]

Maintainer Joshua Millstein <`joshua.millstein@usc.edu`>

Repository CRAN

Date/Publication 2022-06-07 22:40:02 UTC

R topics documented:

`fdrci-package` *Permutation-Based FDR Point and Confidence Interval Estimation*

Description

FDR functions for permutation-based estimators, including pi0 as well as FDR confidence intervals. The confidence intervals account for dependencies between tests by the incorporation of an overdispersion parameter, which is estimated from the permuted data. The package also includes a parametric analog of the same approach.

Details

Package:	fdrci
Type:	Package
Version:	2.3
Date:	2021-12-22
License:	Artistic-2.0
LazyLoad:	yes

This method is designed to compute FDR when a permutation-based approach has been utilized. The objective here is to identify a subset of positive tests that have corresponding statistics with a more exteme distribution than the permuted results, which are assumed to represent the null. The significance of the subset is described in terms of the FDR and uncertainty in the FDR estimate by computing a confidence interval. Say a set of p-values(or simply a set of test statistics) were recorded for a set of hypothesis tests, and data were permuted B times with test results generated for each permutation. The function fdr_od() can be used to estimate FDR and and a confidence interval along with pi0, the proportion of true null hypotheses, given a selected significance threshold. The function fdrTbl()uses fdr_od() to create a table of results over a sequence of possible significance thresholds. Finally, the function FDRplot will plot results from fdrTbl(), facilitating the selection of a final significance threshold.

Author(s)

Joshua Millstein, Eric S. Kawaguchi

Maintainer: Joshua Millstein <joshua.millstein@usc.edu> Joshua Millstein

References

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

FDRplot *Plot results of FDR table generated by fdrTbl()*

Description

This function plots FDR point and CI estimates over a sequence of possible significance thresholds. Results from fdrTbl() can be plotted directly as input to FDRplot.

Usage

```
FDRplot(
  plotdat,
  lowerbound,
  upperbound,
  ymax = 1,
  annot = "",
  xpos = 0.8,
  ypos = 0.8
)
```

Arguments

`plotdat`	a table that is returned from fdrTbl(), or results formated in the same way.
`lowerbound`	-log10(p-value) lower bound for the x-axis of the plot.
`upperbound`	-log10(p-value) upper bound for the x-axis of the plot.
`ymax`	upper limit for range of the y-axis.
`annot`	annotation text to be added to plot area.
`xpos`	x-axis position for annot
`ypos`	y-axis position for annot

Value

ggplot2 object

Author(s)

Joshua Millstein, `<joshua.millstein@usc.edu>`

Joshua Millstein

References

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

Examples

```
ss = 100
nvar = 100
X = as.data.frame(matrix(rnorm(ss*nvar),nrow=ss,ncol=nvar))
e = as.data.frame(matrix(rnorm(ss*nvar),nrow=ss,ncol=nvar))
Y = .1*X + e
nperm = 10

myanalysis = function(X,Y){
ntests = ncol(X)
rslts = as.data.frame(matrix(NA,nrow=ntests,ncol=2))
names(rslts) = c("ID","pvalue")
rslts[,"ID"] = 1:ntests
for(i in 1:ntests){
fit = cor.test(X[,i],Y[,i],na.action="na.exclude",
alternative="two.sided",method="pearson")
rslts[i,"pvalue"] = fit$p.value
}
return(rslts)
} # End myanalysis

# Generate observed results
obs = myanalysis(X,Y)

# Generate permuted results
perml = vector('list',nperm)
for(perm in 1:nperm){
X1 = X[order(runif(nvar)),]
perml[[perm]] = myanalysis(X1,Y)
}

# FDR results table
myfdrtbl = fdrTbl(obs$pvalue,perml,"pvalue",nvar,0,3)
# Plot results
FDRplot(myfdrtbl,0,3,annot="A. An Example")
```

---

fdrTbl *FDR Estimate and Confidence Interval Sequence Table*

Description

Computes FDR estimates and confidence intervals for a sequence of potential significance thresholds.

Usage

```
fdrTbl(
  obs.vec,
  perm.list = NULL,
  pname,
  ntests,
  lowerbound,
  upperbound,
  incr = 0.1,
  cl = 0.95,
  c1 = NA,
  correct = "none",
  meff = TRUE,
  seff = TRUE,
  mymat,
  nperms = 5
)
```

Arguments

`obs.vec`	observed vector of p-values.
`perm.list`	list of dataframes that include a column of permutation p-values (or statistics) in each. The length of the list permp = number of permutations.
`pname`	name of column in each list component dataframe that includes p-values (or statistics).
`ntests`	total number of observed tests, which is usually the same as the length of obs.vec and the number of rows in each perm.list dataframe. However, this may not be the case if results were filtered by a p-value threshold or statistic threshold. If filtering was conducted then lowerbound must be greater (more extreme) than the filtering criterion.
`lowerbound`	lowerbound refers to the range of -log10(p-value) over which fdr is computed for a sequence of thresholds
`upperbound`	upperbound refers to the range of -log10(p-value) over which fdr is computed for a sequence of thresholds
`incr`	value by which to increment the sequence from lowerbound to upperbound on a -log10(p-value) scale. Default is 0.1.
`cl`	confidence level (default is .95).
`c1`	overdispersion parameter to account for dependencies among tests. If all tests are known to be independent, then this parameter should be set to 1.
`correct`	"none", "BH", should confidence intervals be corrected for multiplicity using a modification of the Benjamini and Yekutieli (2005) approach for selecting and correcting intervals? (default is "none")
`meff`	(For parametric estimation, if `perm.list = NULL`.) Logical. To be passed into `fdr_od`. `TRUE` implies the calculation of the effective number of tests based on the JM estimator (Default is `TRUE`)
`seff`	(For parametric estimation, if `perm.list = NULL`.) Logical. To be passed into `fdr_od`. `TRUE` implies the calculation of the effective number of rejected hypotheses based on the JM estimator (Default is `TRUE`)

`mymat`	(For parametric estimation, if `perm.list = NULL`.) Matrix. To be passed into `fdr_od`. Design matrix used to calculate the p-values provided in `obsp`.
`nperms`	(For parametric estimation, if `perm.list = NULL`.) Integer. To be passed into `fdr_od`. Number of permutations needed to estimate the effective number of (rejected) tests. (Must be non-zero, default is 5)

Details

fdrTbl calls fdr_od for a series of discovery thresholds. Output from fdrTbl() can be used for FDRplot() input.

If correct = "BH", then confidence intervals will be corrected according to the thresholds specified by lowerbound, upperbound, and incr. Thresholds will be selected if FDR is determined to be significantly different than 1. First a Z-score test is conducted using the Millstein & Volfson standard error estimate. Then BH FDR is computed according to the Benjamini and Yekutieli (2005) approach. CIs for selected thresholds will be adjusted to account for multiple CI estimation. For thresholds that are not selected, NA values are returned.

Value

A dataframe is returned where rows correspond to p-value thresholds in the sequence from lowerbound to upperbound and columns are:

If permutation: c("threshold","fdr","ll","ul","pi0","odp","S","Sp")

`threshold`	p-value threshold chosen to define positive tests
`fdr`	estimated FDR at the chosen p-value threshold
`ll`	estimated lower 95% confidence bound for the FDR estimate
`ul`	estimated upper 95% confidence bound for the FDR estimate
`pi0`	estimated percent of true null hypotheses
`odp`	estimated over-dispersion parameter
`S`	observed number of positive tests
`Sp`	total number of positive tests summed across all permuted result sets

If parametric: c("threshold","fdr","ll","ul","M","M.eff","S","S.eff")

`threshold`	p-value threshold chosen to define positive tests
`fdr`	estimated FDR at the chosen p-value threshold
`ll`	estimated lower 95% confidence bound for the FDR estimate
`ul`	estimated upper 95% confidence bound for the FDR estimate
`M`	total number of tests
`M.eff`	Effective number of tests via the JM estimator
`S`	observed number of positive tests
`S.eff`	effective number of positive tests via the JM estimator

Author(s)

Joshua Millstein, Eric S. Kawaguchi

References

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

Benjamini, Yoav, and Daniel Yekutieli. "False discovery rate adjusted multiple confidence intervals for selected parameters." Journal of the American Statistical Association 100.469 (2005): 71-81.

Examples

```
n.row=100
n.col=100
X = as.data.frame(matrix(rnorm(n.row*n.col),nrow=n.row,ncol=n.col))
e = as.data.frame(matrix(rnorm(n.row*n.col),nrow=n.row,ncol=n.col))
Y = .1*X + e
nperm = 10

myanalysis = function(X,Y){
ntests = ncol(X)
rslts = as.data.frame(matrix(NA,nrow=ntests,ncol=2))
names(rslts) = c("ID","pvalue")
rslts[,"ID"] = 1:ntests
for(i in 1:ntests){
fit = cor.test(X[,i],Y[,i],na.action="na.exclude",
alternative="two.sided",method="pearson")
rslts[i,"pvalue"] = fit$p.value
}
return(rslts)
} # End myanalysis

## Generate observed results
obs = myanalysis(X,Y)

## Generate permuted results
perml = vector('list',nperm)
for(perm in 1:nperm){
X1 = X[order(runif(n.col)),]
perml[[perm]] = myanalysis(X1,Y)
}

## FDR results table
fdrTbl(obs$pvalue,perml,"pvalue",n.col,1,2)
fdrTbl(obs$pvalue,perml,"pvalue",n.col,1,2,correct="BH")

## FDR results table (parametric)
fdrTbl(obs$pvalue, NULL, "pvalue",n.col,1,2,meff = TRUE, seff = TRUE, mymat = X, nperms = 5)
```

fdr_od *Tail-area FDR and Confidence Interval*

Description

This function can be used to estimate FDR, corresponding confidence interval, and pi0, the proportion of true null hypotheses, given a selected significance threshold, and results from permuted data.

Usage

```
fdr_od(
  obsp,
  permp = NULL,
  pnm,
  ntests,
  thres,
  cl = 0.95,
  c1 = NA,
  meff = TRUE,
  seff = TRUE,
  mymat,
  nperms = 5
)
```

Arguments

`obsp`	observed vector of p-values.
`permp`	list of dataframes that include a column of permutation p-values (or statistics) in each. The length of the list permp = number of permutations. If `permp = NULL`, then the parametric estimator is calculated.
`pnm`	name of column in each list component dataframe that includes p-values (or statistics).
`ntests`	total number of observed tests, which is usually the same as the length of obsp and the number of rows in each permp dataframe. However, this may not be the case if results were filtered by a p-value threshold or statistic threshold. If filtering was conducted then thres must be smaller (more extreme) than the filtering criterion.
`thres`	significance threshold.
`cl`	confidence level (default is .95).
`c1`	overdispersion parameter. If this parameter is not specified (default initial value is NA), then the parameter is estimated from the data. If all tests are known to be independent, then this parameter should be set to 1.
`meff`	(For parametric estimation, if `permp = NULL`.) Logical. `TRUE` implies the calculation of the effective number of tests based on the JM estimator (Default is `TRUE`)

`seff`	(For parametric estimation, if `permp = NULL`.) Logical. `TRUE` implies the calculation of the effective number of rejected hypotheses based on the JM estimator (Default is `TRUE`)
`mymat`	(For parametric estimation, if `permp = NULL`.) Matrix. Design matrix used to calculate the p-values provided in `obsp`.
`nperms`	(For parametric estimation, if `permp = NULL`.) Integer. Number of permutations needed to estimate the effective number of (rejected) tests. (Must be non-zero, default is 5)

Details

If a very large number of tests are conducted, it may be useful to filter results, that is, save only results of those tests that meet some relaxed nominal significance threshold. This alleviates the need to record results for tests that are clearly non-significant. Results from fdr_od() are valid as long as thres < the relaxed nomimal significance threshold for both observed and permuted results. It is not necessary for the input to fdr_od() to be p-values, however, fdr_od() is designed for statistics in which smaller values are more extreme than larger values as is the case for p-values. Therefore, if raw statistics are used, then a transformation may be necessary to insure that smaller values are more likely associated with false null hypotheses than larger values. In certain situations, for instance when a large proportion of tests meet the significance threshold, `pi0` is estimated to be very small, and thus has a large influence on the FDR estimate. To limit this influence, `pi0` is constrained to be .5 or greater, resulting in a more conservative estimate under these conditions.

Value

For the permutation-based estimator: A list which includes:

`FDR`	FDR point estimate
`ll`	lower confidence limit
`ul`	upper confidence limit
`pi0`	proportion of true null hypotheses
`c1`	overdispersion parameter
`S`	observed number of positive tests
`Sp`	total number of positive tests summed across all permuted result sets

For the parametric-based estimator: A list which includes:

`FDR`	FDR point estimate
`ll`	lower confidence limit
`ul`	upper confidence limit
`M`	total number of tests
`M.eff`	effective number of tests. `NA` if `meff = FALSE`
`S`	observed number of positive tests
`S.eff`	effective number of positive tests. `NA` if `seff = FALSE`

Author(s)

Joshua Millstein, Eric S. Kawaguchi

References

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

Examples

```
ss=100
nvar=100
X = as.data.frame(matrix(rnorm(ss*nvar),nrow=ss,ncol=nvar))
e = as.data.frame(matrix(rnorm(ss*nvar),nrow=ss,ncol=nvar))
Y = .1*X + e
nperm = 10

myanalysis = function(X,Y){
ntests = ncol(X)
rslts = as.data.frame(matrix(NA,nrow=ntests,ncol=2))
names(rslts) = c("ID","pvalue")
rslts[,"ID"] = 1:ntests
for(i in 1:ntests){
fit = cor.test(X[,i],Y[,i],na.action="na.exclude",
alternative="two.sided",method="pearson")
rslts[i,"pvalue"] = fit$p.value
}
return(rslts)
} # End myanalysis

# Generate observed results
obs = myanalysis(X,Y)

## Generate permuted results
perml = vector('list',nperm)
for(p_ in 1:nperm){
X1 = X[order(runif(nvar)),]
perml[[p_]] = myanalysis(X1,Y)
}

## FDR results (permutation)
fdr_od(obs$pvalue,perml,"pvalue",nvar, thres = .05)

## FDR results (parametric)
fdr_od(obs$pvalue, permp = NULL, thres = 0.05, meff = FALSE, seff = FALSE, mymat = X)
```

meff.jm *Estimate the Effective Number of Tests*

Description

Estimate the effective number of tests using a permutation-based approach.

Usage

```
meff.jm(mydat, B = 1)
```

Arguments

`mydat`	A design matrix with n observations (rows) and p covariates (columns).
`B`	Integer. Number of permutations to perform. (Default is 1)

Details

The effective no of independent vars is the sum of the scaled eigenvalues for truly independent vars (n) minus the variance inflation of the top eigenvalues, that is, (observed - permuted) The sum of the eigenvalues of the correlation matrix = trace = sum of diag = n. The eigenvalues of variables X can be thought of as normalized Dependencies between variables will increase the magnitude of top eigenvalues, therefore, the effective number of independent variables is equal to the proportion of the sum of eigenvalues attributed to independence minus the proportion attributed to dependencies.

Value

Numeric. Returns the estimated effective number of tests averaged over `B` permutations.

Author(s)

Joshua Millstein, Eric S. Kawaguchi

References

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

Examples

```

# Independent
ss=300
nvar=100
X = as.data.frame(matrix(rnorm(ss * nvar), nrow = ss, ncol = nvar))
meff.jm(X, B = 5)

# High correlation
```

```
S = matrix(0.9, nvar, nvar)
diag(S) = 1
X = as.matrix(X) %*% chol(S)
meff.jm(X, B = 5)
```

`MV_q` *MV q-values and confidence intervals*

Description

q-values with confidence intervals are generated, based in the Millstein and Volfson (MV) estimators.

Usage

```
MV_q(obsp, permp, pnm, ntests, cl = 0.95, c1 = NA)
```

Arguments

`obsp`	observed vector of p-values.
`permp`	list of dataframes that include a column of permutation p-values (or statistics) in each. The length of the list permp = number of permutations.
`pnm`	name of column in each list component dataframe that includes p-values (or statistics).
`ntests`	total number of observed tests, which is usually the same as the length of obsp and the number of rows in each permp dataframe. However, this may not be the case if results were filtered by a p-value threshold or statistic threshold. If filtering was conducted then thres must be smaller (more extreme) than the filtering criterion.
`cl`	confidence level (default is .95).
`c1`	overdispersion parameter. If this parameter is not specified (default initial value is NA), then the parameter is estimated from the data. If all tests are known to be independent, then this parameter should be set to 1.

Details

Millstein and Volfson (2013) FDR is based on the idea that FDR is estimated at a level specified by the investigator. Storey and Tibshirani (2003) developed the q-value concept, where FDR is estimated at each observed p-value. However, Millstein and Volfson argued that in order to be informative, uncertainty in the estimate should be quantified, thus the development of confidence intervals for FDR. The MV FDR estimator is less conservative than the BH estimator.

Value

A dataframe which includes:

`q`	q-value corresponding to the respective p-value
`q.ll`	q-value lower limit
`q.ul`	q-value upper limit

Author(s)

Joshua Millstein

References

Millstein J, Volfson D. 2013. Computationally efficient permutation-based confidence interval estimation for tail-area FDR. Frontiers in Genetics | Statistical Genetics and Methodology 4(179):1-11.

Storey, John D., and Robert Tibshirani. "Statistical significance for genomewide studies." Proceedings of the National Academy of Sciences 100.16 (2003): 9440-9445.

Examples

```
ss=100
nvar=100
X = as.data.frame(matrix(rnorm(ss*nvar),nrow=ss,ncol=nvar))
e = as.data.frame(matrix(rnorm(ss*nvar),nrow=ss,ncol=nvar))
Y = .1*X + e
nperm = 10

myanalysis = function(X,Y){
ntests = ncol(X)
rslts = as.data.frame(matrix(NA,nrow=ntests,ncol=2))
names(rslts) = c("ID","pvalue")
rslts[,"ID"] = 1:ntests
for(i in 1:ntests){
fit = cor.test(X[,i],Y[,i],na.action="na.exclude",
alternative="two.sided",method="pearson")
rslts[i,"pvalue"] = fit$p.value
}
return(rslts)
} # End myanalysis

# Generate observed results
obs = myanalysis(X,Y)

# Generate permuted results
perml = vector('list',nperm)
for(p_ in 1:nperm){
X1 = X[order(runif(nvar)),]
perml[[p_]] = myanalysis(X1,Y)
}

q.values.MV = MV_q(obs$pvalue,perml,"pvalue",nvar)
```

Index