The fdrtool Package

October 21, 2007

Version 1.2.0

Date 2007-10-21

Title Estimation and Control of (Local) False Discovery Rates

Author Korbinian Strimmer.

Maintainer Korbinian Strimmer <strimmer@uni-leipzig.de>

Depends R (>= 2.2.1)

Suggests

Description This package allows to estimate both tail area-based false discovery rates (Fdr) as well as local false discovery rates (fdr) for a variety of null models (p-values, z-scores, correlation coefficients, t-scores). The proportion of null values and the parameters of the null distribution are adaptively estimated from the data. In addition, the package contains functions for non-parametric density estimation (Grenander estimator), for monotone regression (isotonic regression and antitonic regression with weights), for computing the greatest convex minorant (GCM) and the least concave majorant (LCM), and for the half-normal and correlation distributions.

License GPL (>= 3)

URL http://strimmerlab.org/software/fdrtool/

R topics documented:

censored.fit ................................................................. 2
dcor0 ................................................................. 3
fdrtool-internal ......................................................... 4
fdrtool ................................................................. 5
gcmlcm ............................................................... 7
grenader ............................................................... 8
halfnormal ............................................................ 10
monoreg ............................................................... 12
pval.estimate.eta0 ................................................... 14
pvalues ............................................................... 16
censored.fit

**Fit Null Distribution To Censored Data by Maximum Likelihood**

**Description**

censored.fit fits a null distribution to censored data.

fndr.cutoff finds a suitable cutoff point based on the (approximate) false non-discovery rate (FNDR).

**Usage**

```r

censored.fit(x, cutoff, statistic=c("normal", "correlation", "pvalue", "studentt"))
fndr.cutoff(x, statistic=c("normal", "correlation", "pvalue", "studentt"))
```

**Arguments**

- `x`: vector of test statistics.
- `cutoff`: truncation point (this may a single value or a vector).
- `statistic`: type of statistic - normal, correlation, or student t.

**Details**

As null model truncated normal, truncated student t or a truncated correlation density is assumed. The truncation point is specified by the cutoff parameter. All data points whose absolute value are larger than the cutoff point are ignored when fitting the truncated null model via maximum likelihood. The total number of data points is only used to estimate the fraction of null values eta0.

**Value**

- `censored.fit` returns a matrix whose rows contain the estimated parameters and corresponding errors for each cutoff point.
- `fndr.cutoff` returns a tentative cutoff point.

**See Also**

- `fdrtool`

**Examples**

```r

# load "fdrtool" library
library("fdrtool")

# simulate normal data
sd.true = 2.232
n = 5000
z = rnorm(n, sd=sd.true)
```
# simulate contaminated mixture of correlation distribution
r <- rcor0(700, kappa=10)
u1 <- runif(200, min=-1, max=-0.7)
u2 <- runif(200, min=0.7, max=1)
rc <- c(r, u1, u2)
censored.fit(r, 0.7, statistic="correlation")
censored.fit(rc, 0.7, statistic="correlation")

# pvalue example
data(pvalues)
c = findR.cutoff(pvalues, statistic="pvalue")
c
censored.fit(pvalues, cutoff=c, statistic="pvalue")

dcor0

Distribution of the Vanishing Correlation Coefficient (rho=0) and Related Functions

Description

The above functions describe the distribution of the Pearson correlation coefficient r assuming that there is no correlation present (\(\rho = 0\)).

Note that the distribution has only a single parameter: the degree of freedom kappa, which is equal to the inverse of the variance of the distribution.

The theoretical value of kappa depends both on the sample size n and the number p of considered variables. If a simple correlation coefficient between two variables (p=2) is considered the degree of freedom equals kappa = n-1. However, if a partial correlation coefficient is considered (conditioned on p-2 remaining variables) the degree of freedom is kappa = n-1-(p-2) = n-p+1.

Usage

dcor0(x, kappa, log=FALSE)
pkor0(q, kappa, lower.tail=TRUE, log.p=FALSE)
qkor0(p, kappa, lower.tail=TRUE, log.p=FALSE)
rcor0(n, kappa)

Arguments

x, q
vector of sample correlations

p
vector of probabilities

kappa
the degree of freedom of the distribution (= inverse variance)

n
number of values to generate. If n is a vector, length(n) values will be generated

log, log.p
logical vector; if TRUE, probabilities p are given as log(p)
lower.tail logical vector; if TRUE (default), probabilities are \( P[R \leq r] \), otherwise, \( P[R > r] \).

Details

For density and distribution functions as well as a corresponding random number generator of the correlation coefficient for arbitrary non-vanishing correlation \( \rho \) please refer to the SuppDists package by Bob Wheeler (bwheeler@echip.com) (available on CRAN). Note that the parameter \( N \) in his dPearson function corresponds to \( N=\kappa+1 \).

Value

dcor0 gives the density, pcor0 gives the distribution function, qcor0 gives the quantile function, and rcor0 generates random deviates.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also
cor.

Examples

```r
# load fdrtool library
library("fdrtool")

# distribution of r for various degrees of freedom
x <- seq(-1,1,0.01)
y1 <- dcor0(x, kappa=7)
y2 <- dcor0(x, kappa=15)
plot(x,y2,type="l", xlab="r", ylab="pdf",
     xlim=c(-1,1), ylim=c(0,2))
lines(x,y1)

# simulated data
r <- rcor0(1000, kappa=7)
hist(r, freq=FALSE,
     xlim=c(-1,1), ylim=c(0,5))
lines(x,y1,type="l")

# distribution function
pcor0(-0.2, kappa=15)
```
fdrtool-internal  

Internal fdrtool Functions

Description

Internal fdrtool functions.

Note

These are not to be called by the user (or in some cases are just waiting for proper documentation to be written).

fdrtool  

Estimate (Local) False Discovery Rates For Diverse Test Statistics

Description

fdrtool takes a vector of z-scores (or of correlations, p-values, or t-statistics), and estimates for each case both the tail area-based Fdr as well as the density-based fdr (=q-value resp. local false discovery rate). The parameters of the null distribution are estimated adaptively from the data (except for the case of p-values where this is not necessary).

Usage

fdrtool(x, statistic=c("normal", "correlation", "pvalue", "studentt"), plot=TRUE, verbose=TRUE, cutoff.method=c("fndr", "pct0", "locfdr"), pct0=0.75)

Arguments

x  
vector of the observed test statistics.

statistic  
one of "normal" (default), "correlation", "pvalue", "studentt". This species the null model.

plot  
plot a figure with estimated densities, distribution functions, and (local) false discovery rates.

verbose  
print out status messages.

cutoff.method  
one of "fndr" (default), "pct0", "locfdr".

pct0  
fraction of data used for fitting null model - only if cutoff.method="pct0"
Details

The algorithm implemented in this function proceeds as follows:

1. A suitable cutoff point is determined. If cutoff.method is "fnr" then first an approximate null model is fitted and subsequently a cutoff point is sought with false nondiscovery rate as small as possible (see fnr.cutoff). If cutoff.method is "pct0" then a specified quantile (default value: 0.75) of the data is used as the cutoff point. If cutoff.method equals "lofdr" then the heuristic of the "lofdr" package (version 1-1.5) is employed to find the cutoff (z-scores only).

2. The parameters of the null model are estimated from the data using censored.fit. This results in estimates for scale parameters und and proportion of null values (eta0).

3. Subsequently the corresponding p-values are computed, and a modified grenander algorithm is employed to obtain the overall density and distribution function (note that this respects the estimated eta0).

4. Finally, q-values and local fdr values are computed for each case.

The assumed null models all have (except for p-values) one free scale parameter. Note that the z-scores and the correlations are assumed to have zero mean.

Value

A list with the following components:

- pval a vector with p-values for each case.
- qval a vector with q-values (Fdr) for each case.
- lfdc a vector with local fdr values for each case.
- statistic the specified type of null model.
- param a vector containing the estimated parameters (the null proportion eta0 and the free parameter of the null model).

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also

pval.estimate.eta0,censored.fit.

Examples

# load "fdrtool" library and p-values
library("fdrtool")
data(pvalues)

# estimate fdr and Fdr from p-values

data(pvalues)
fdr <- fdrtool(pvalues, statistic="pvalue")
R code:
```r
fdr$qval # estimated Fdr values
fdr$lfdr # estimated local fdr

# estimate fdr and Fdr from z-scores
sd.true = 2.232
n = 500
z = rnorm(n, sd=sd.true)
z = c(z, runif(30, 5, 10)) # add some contamination
fdr <- fdrtool(z)

# you may change some parameters of the underlying functions
fdr <- fdrtool(z, cutoff.method="pct0", pct0=0.9)
```

**gcmlcm**

### Greatest Convex Minorant and Least Concave Majorant

**Description**

gcmlcm computes the greatest convex minorant (GCM) or the least concave majorant (LCM) of a piece-wise linear function.

**Usage**

gcmlcm(x, y, type=c("gcm", "lcm"))

**Arguments**

- **x, y** coordinate vectors of the piece-wise linear function. Note that the x values need to be unique and be arranged in sorted order.
- **type** specifies whether to compute the greatest convex minorant (type="gcm", the default) or the least concave majorant (type="lcm").

**Details**

The GCM is obtained by isotonic regression of the raw slopes, whereas the LCM is obtained by antitonic regression. See Robertson et al. (1988).

**Value**

A list with the following entries:

- **x.knots** the x values belonging to the knots of the LCM/GCM curve
- **y.knots** the corresponding y values
- **slope.knots** the slopes of the corresponding line segments

**Author(s)**

Korbinian Strimmer ([http://strimmerlab.org](http://strimmerlab.org)).
References


See Also

monoreg.

Examples

```r
# load "fdrtool" library
library("fdrtool")

# generate some data
x = 1:20
y = rexp(20)
plot(x, y, type="l", lty=3, main="GCM (red) and LCM (blue)")
points(x, y)

# greatest convex minorant (red)
gg = gcmlcm(x,y)
lines(gg$x.knots, gg$y.knots, col=2, lwd=2)

# least concave majorant (blue)
ll = gcmlcm(x,y, type="lcm")
lines(ll$x.knots, ll$y.knots, col=4, lwd=2)
```

---

grenader

-Grenander Estimator of a Decreasing or Increasing Density-

Description

The function `grenader` computes the Grenander estimator of a one-dimensional decreasing or increasing density.

Usage

`grenader(F, type=c("decreasing", "increasing"))`

Arguments

- `F` is an `ecdf` containing the empirical cumulative density.
- `type` specifies whether the distribution is decreasing (the default) or increasing.
Details

The Grenander (1956) density estimator is given by the slopes of the least concave majorant (LCM) of the empirical distribution function (ECDF). It is a decreasing piecewise-constant function and can be shown to be the non-parametric maximum likelihood estimate (NPMLE) under the assumption of a decreasing density (note that the ECDF is the NPMLE without this assumption). Similarly, an increasing density function is obtained by using the greatest convex minorant (GCM) of the ECDF.

Value

A list of class `grenander` with the following components:

- `F`: the empirical distribution function specified as input.
- `x.knots`: x locations of the knots of the least concave majorant of the ECDF.
- `F.knots`: the corresponding y locations of the least concave majorant of the ECDF.
- `f.knots`: the corresponding slopes (=density).

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

References


See Also

cedf, gcmlcm, density.

Examples

```r
# load "fdrtool" library
library("fdrtool")

# samples from random exponential variable
z = rexp(30,1)
e = ecdf(z)
g = grenander(e)
g

# plot ecdf, concave cdf, and Grenander density estimator (on log scale)
plot(g, log="y")

# for comparison the kernel density estimate
plot(density(z))

# area under the Grenander density estimator
sum( g$f.knots[-length(g$f.knots)] * diff(g$x.knots) )
```
The Half-Normal Distribution

Description

Density, distribution function, quantile function and random generation for the half-normal distribution with parameter \( \theta \).

Usage

- \( dhalfnorm(x, \theta = \sqrt{\pi/2}, \text{log} = \text{FALSE}) \)
- \( phalfnorm(q, \theta = \sqrt{\pi/2}, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE}) \)
- \( qhalfnorm(p, \theta = \sqrt{\pi/2}, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE}) \)
- \( rhalfnorm(n, \theta = \sqrt{\pi/2}) \)
- \( \text{sd2theta}(sd) \)
- \( \text{theta2sd}(\theta) \)

Arguments

- \( x, q \) vector of quantiles.
- \( p \) vector of probabilities.
- \( n \) number of observations. If \( \text{length}(n) > 1 \), the length is taken to be the number required.
- \( \theta \) parameter of half-normal distribution.
- \( \text{log, log.p} \) logical; if \( \text{TRUE} \), probabilities \( p \) are given as \( \log(p) \).
- \( \text{lower.tail} \) logical; if \( \text{TRUE} \) (default), probabilities are \( P[X \leq x] \), otherwise, \( P[X > x] \).
- \( sd \) standard deviation of the zero-mean normal distribution that corresponds to the half-normal with parameter \( \theta \).

Details

\( x = \text{abs}(z) \) follows a half-normal distribution with if \( z \) is a normal variate with zero mean. The half-normal distribution has density

\[
f(x) = \frac{2\theta}{\pi} e^{-x^2\theta^2/\pi}
\]

It has mean \( E(x) = \frac{1}{\theta} \) and variance \( \text{Var}(x) = \frac{\pi-2}{2\theta^2} \).

The parameter \( \theta \) is related to the standard deviation \( \sigma \) of the corresponding zero-mean normal distribution by the equation \( \theta = \sqrt{\pi/2}/\sigma \).

If \( \theta \) is not specified in the above functions it assumes the default values of \( \sqrt{\pi/2} \), corresponding to \( \sigma = 1 \).
Value

dhalfnorm gives the density, phalfnorm gives the distribution function, qhalfnorm gives the quantile function, and rhalfnorm generates random deviates. sd2theta computes a theta parameter. theta2sd computes a sd parameter.

See Also

Normal.

Examples

# load "fdrtool" library
library("fdrtool")

## density of half-normal compared with a corresponding normal
par(mfrow=c(1,2))

sd.norm = 0.64
x = seq(0, 5, 0.01)
x2 = seq(-5, 5, 0.01)
plot(x, dhalfnorm(x, sd2theta(sd.norm)), type="l", xlim=c(-5, 5), lwd=2,
    main="Probability Density", ylab="pdf(x)")
lines(x2, dnorm(x2, sd=sd.norm), col=8)

plot(x, phalfnorm(x, sd2theta(sd.norm)), type="l", xlim=c(-5, 5), lwd=2,
    main="Distribution Function", ylab="cdf(x)")
lines(x2, pnorm(x2, sd=sd.norm), col=8)

legend("topleft",
c("half-normal", "normal"), lwd=c(2,1),
col=c(1, 8), bty="n", cex=1.0)

par(mfrow=c(1,1))

## distribution function

integrate(dhalfnorm, 0, 1.4, theta = 1.234)
phalfnorm(1.4, theta = 1.234)

## quantile function
qhalfnorm(-1) # NaN
qhalfnorm(0)
qhalfnorm(.5)
qhalfnorm(1)
qhalfnorm(2) # NaN

## random numbers
theta = 0.72
hz = rhalfnorm(10000, theta)
hist(hz, freq=FALSE)
lines(x, dhalfnorm(x, theta))
monoreg

Monotone Regression: Isotonic Regression and Antitonic Regression

Description

monoreg performs monotone regression (either isotonic or antitonic) with weights.

Usage

monoreg(x, y=NULL, w=rep(1, length(x)), type=c("isotonic", "antitonic"))

Arguments

x, y coordinate vectors of the regression points. Alternatively a single “plotting”
structure can be specified: see xy.coords.

w data weights (default values: 1).

type fit a monotonely increasing ("isotonic") or monotonely decreasing ("antitonic")
function.

Details

monoreg is similar to isoreg, with the addition that monoreg accepts weights.

If several identical x values are given as input, the corresponding y values and the weights w are
automatically merged, and a warning is issued.

The plot.monoreg function optionally plots the cumulative sum diagram with the greatest conv-
ex minorant (isotonic regression) or the least concave majorant (antitonic regression), see the ex-
amples below.
**monoreg**

**Value**

A list with the following entries:

- `x` the sorted and unique x values
- `y` the corresponding y values
- `w` the corresponding weights
- `yf` the fitted y values
- `type` the type of monotone regression ("isotonic" or "antitonic"
- `call` the function call

**Author(s)**

Korbinian Strimmer (http://strimmerlab.org).

Part of this function is C code that has been ported from R code originally written by Kaspar Rufibach.

**References**


**See Also**

isoreg.

**Examples**

```r
# load "fdrtool" library
library("fdrtool")

# an example with weights

# Example 1.1.1. (dental study) from Robertson, Wright and Dykstra (1988)
age = c(14, 14, 8, 8, 8, 10, 10, 12, 12, 12)
size = c(23.5, 25, 21, 23.5, 23, 24, 21, 25, 21.5, 22, 19)
mr = monoreg(age, size)

# sorted x values
mr$x # 8 10 12 14
# weights and merged y values
mr$w # 3 3 3 2
mr$y # 22.50000 23.33333 20.83333 24.25000
# fitted y values
mr$yf # 22.22222 22.22222 22.22222 24.25000
fitted(mr)
residuals(mr)

plot(mr, ylim=c(18, 26)) # this shows the averaged data points
```
points(age, size, pch=2)  # add original data points

###

y = c(1,0,1,0,0,1,0,1,0,1,0,0)
x = 1:length(y)
mr = monoreg(y)

# plot with greatest convex minorant
plot(mr, plot.type="row.wise")

# this is the same
mr = monoreg(x,y)
plot(mr)

# antitonic regression and least concave majorant
mr = monoreg(-y, type="a")
plot(mr, plot.type="row.wise")

# the fit yf is independent of the location of x and y
plot(monoreg(x + runif(1, -1000, 1000),
          y +runif(1, -1000, 1000)) )

###

y = c(0,0,2/4,1/5,2/4,1/2,4/5,5/8,7/11,10/11)
x = c(5,9,13,18,22,24,29,109,120,131)

mr = monoreg(x,y)
plot(mr, plot.type="row.wise")

# the fit (yf) only depends on the ordering of x
monoreg(1:length(y), y)$yf
monoreg(x, y)$yf

---

**pval.estimate.eta0**  
*Estimate the Proportion of Null p-Values*

**Description**

`pval.estimate.eta0` estimates the proportion `eta0` of null p-values in a given vector of p-values.

**Usage**

```r
pval.estimate.eta0(p, method=c("smoother", "bootstrap", "conservative", "adaptive", "quantile"), lambda=seq(0,0.9,0.05), diagnostic.plot=TRUE, q=0.1)
```
Arguments

- **p**: vector of p-values
- **method**: algorithm used to estimate the proportion of null p-values. Available options are "conservative", "adaptive", "bootstrap", quantile, and "smoother" (default).
- **lambda**: optional tuning parameter vector needed for "bootstrap" and "smoothing" methods (defaults to seq(0, 0.9, 0.05)) - see Storey (2002) and Storey and Tibshirani (2003).
- **diagnostic.plot**: if TRUE (the default) the histogram of the p-values together with the estimate of \( \eta_0 \) null line is plotted. This is useful to visually check the fit of the estimated proportion of null p-values.
- **q**: quantile used for estimating \( \eta_0 \) - only if method="quantile"

Details

This quantity \( \eta_0 \), i.e. the proportion \( \eta_0 \) of null p-values in a given vector of p-values, is an important parameter when controlling the false discovery rate (FDR). A conservative choice is \( \eta_0 = 1 \) but a choice closer to the true value will increase efficiency and power - see Benjamini and Hochberg (1995, 2000) and Storey (2002) for details.

The function `pval.estimate.eta0` provides five algorithms: the "conservative" method always returns \( \eta_0 = 1 \) (Benjamini and Hochberg, 1995), "adaptive" uses the approach suggested in Benjamini and Hochberg (2000), "bootstrap" employs the method from Storey (2002), "smoother" uses the smoothing spline approach in Storey and Tibshirani (2003), and "quantile" is a simplified version of "smoother".

Value

The estimated proportion \( \eta_0 \) of null p-values.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

Adapted in part from S-PLUS code by Y. Benjamini (http://www.math.tau.ac.il/~roee/FDR_Splus.txt) and R code from J.D. Storey (http://faculty.washington.edu/~jstorey/).

References


"quantile" procedure: similar to smoother, except that the lower q quantile of all \( \eta_0 \) computed in dependence of lambda is taken as overall estimate of \( \eta_0 \).

See Also

*fdrtool*.

Examples

```r
# load "fdrtool" library and p-values
library("fdrtool")
data(pvalues)

# Proportion of null p-values for different methods
pval.estimate.eta0(pvalues, method="conservative")
pval.estimate.eta0(pvalues, method="adaptive")
pval.estimate.eta0(pvalues, method="bootstrap")
pval.estimate.eta0(pvalues, method="smoother")
pval.estimate.eta0(pvalues, method="quantile")
```

pvalues

**Example p-Values**

Description

This data set contains 4,289 p-values. These data are used to illustrate the functionality of the functions *fdrtool* and *pval.estimate.eta0*.

Usage

```r
data(pvalues)
```

Format

pvalues is a vector with 4,289 p-values.

Examples

```r
# load fdrtool library
library("fdrtool")

# load data set
data(pvalues)

# estimate density and distribution function,
# and compute corresponding (local) false discovery rates
fdrtool(pvalues, statistic="pvalue")
```
Index

* Topic datasets
  pvalues, 16
* Topic distribution
dcor0, 3
  halfnormal, 10
* Topic htest
censored.fit, 2
  fdrtool, 5
  pval.estimate.eta0, 14
* Topic internal
  fdrtool-internal, 4
* Topic regression
  monoreg, 12
* Topic smooth
gcmlcm, 7
  monoreg, 12
* Topic univar
grenader, 8
  grenader, 8
  grenader, 6
  grenader(grenader), 8
  halfnormal, 10
  iqr.fit(fdrtool-internal), 4
  isoreg, 12, 13
  monoreg, 7, 12
  Normal, 11
  num.curv(fdrtool-internal), 4
  pcor0(dcor0), 3
  phalfnorm(halfnormal), 10
  plot.grenander(grenader), 8
  plot.monoreg(monoreg), 12
  pval.estimate.eta0, 6, 14, 16
  pvalues, 16
  pvt.fit.nullmodel
    (fdrtool-internal), 4
  pvt.isoMean(fdrtool-internal), 4
  pvt.nullfunction
    (fdrtool-internal), 4
  pvt.plotlabels
    (fdrtool-internal), 4
  qcor0(dcor0), 3
  qhalfnorm(halfnormal), 10
  r2t(fdrtool-internal), 4
  rcor0(dcor0), 3
  residuals.monoreg(monoreg), 12
  rhalfnorm(halfnormal), 10
  sd2theta(halfnormal), 10
  t2r(fdrtool-internal), 4
  theta2sd(halfnormal), 10
  xy.coords, 12

approximate.fit
  (fdrtool-internal), 4
censored.fit, 2, 5, 6
cor, 4
dcor0, 3
density, 9
dhalfnorm(halfnormal), 10
ecdf, 8, 9
ecdf.pval(fdrtool-internal), 4
fdrtool, 2, 5, 16
fdrtool-internal, 4
fitted.monoreg(monoreg), 12
fnr.cutoff, 5
fnr.cutoff(censored.fit), 2
gcmlcm, 7, 9
get.nullmodel(fdrtool-internal), 4

17