

Package ‘kerfdr’

April 17, 2009

Type Package

Title semi-parametric kernel-based approach to local fdr estimations

Version 1.0.1

Date 2007-10-23

Author M Guedj <mickael.guedj@gmail.com> and G Nuel
<Gregory.Nuel@math-info.univ-paris5.fr>, with contributions from S Robin and A Celisse.

Maintainer M Guedj <mickael.guedj@gmail.com>

Description semi-parametric kernel-based approaches to local fdr estimations useful for the testing of multiple hypothesis (in large-scale genetic, genomic and post-genomic studies for instance).

License GPL

Depends R (>= 2.6.0)

URL <http://stat.genopole.cnrs.fr/sg/software/kerfdr>

Repository CRAN

Date/Publication 2009-01-09 19:21:10

R topics documented:

kerfdr 2

Index 4

kerfdr	<i>kerfdr</i>
--------	---------------

Description

This function computes local fdr values by using a two-components mixture model with a semi-parametric density estimation. The code is freely inspired from the [density](#) function. For a simple use, we recommend the default setting (most parameters are optional).

Usage

```
kerfdr(pv, x=NULL, trans=c("probit", "log", "none"), f0=NULL, localfdr=NULL, pil="storey",
```

Arguments

pv	the vector of raw p-values.
x	a transformation of pv. It can be given by the user or (if NULL) computed via the <code>trans</code> parameter
trans	the transformation to apply on pv to produce x: "probit" (by default) returns <code>qnorm(pv)</code> and "log" returns <code>log10(pv)</code> .
f0	the sample density under the null hypothesis. Can be specified by the user. If NULL (by default) the density under H0 is determined according to <code>trans</code> : if <code>trans = "probit"</code> then f0 is a standard Gaussian distribution; if <code>trans = "log"</code> then f0 is a standard Exponential distribution; if <code>trans = "none"</code> then f0 is a standard Uniform distribution
localfdr	values to initiate the iterative algorithm. If NULL (by default) initial values are then sampled in a Uniform distribution [0,1]
pil	a priori proportion of alternative hypothesis or a method (string) to compute it; by default it uses the method proposed by Storey and Tibshirani (2003).
lambda	p-value threshold for the Storey's calculation of pil (0.5 by default). See qvalue for more details.
bw	a bandwidth value or a method to determine it among "nrd0", "nrd", "ucv", "bcv", "sj-ste", "sj-dpi". See bandwidth for more details.
kernel	the kernel used (string) among "gaussian" (by default), "epanechnikov", "rectangular", "triangular", "biweight", "cosine". For more details on kernels: http://stat.genopole.cnrs.fr/sg/software/kerfdr/kernels
truncat	an interval on p-values to deal with truncated distributions such as those obtained with Monte-Carlo simulations.
plot	if TRUE, it returns graphics of local fdr estimations. Some plots are inspired from qvalue .
cuts	vector of significance values to use in summary (see below)

Value

A list of parameters (`pv`, `x`, `pi1`, `bw`, `f0` ...) and the following results:

<code>f</code>	the observed mixture density
<code>f1</code>	the estimated density under H1
<code>localfdr</code>	the local fdr values resulting from the algorithm
<code>summary</code>	a summary table comparing the number of significant calls for the raw p-values, Bonferroni and Benjamini-Hochberg corrections and for the calculated local fdr, using a set of cutoffs given by <code>cuts</code>

Author(s)

M Guedj, A Celisse, S Robin, G Nuel

References

<http://stat.genopole.cnrs.fr/sg/software/kerfdr>, Robin et al (2007), Strimmer (2008), Guedj et al (under review)

Examples

```
# Example 1: kerfdr with different plots
n = 10000
pi0 = 0.8
# plot in a probit scale (default)
pv = 1-pnorm(c(rnorm(n*pi0), rnorm(n*(1-pi0), 4)))
res = kerfdr(pv)
res$pi0
res$summary
# plot in a log scale
kerfdr(pv, trans = "log")
# plot in the raw p-values scale
kerfdr(pv, trans = "none")
# Example 2: truncation on a vector of null p-values (resulting local fdr should be 1 for ea
n = 10000
pv = runif(n)
# truncation on [0.1;0.9]
pv[which(pv < 0.1)] = 0.1
pv[which(pv > 0.9)] = 0.9
# kerfdr WITHOUT taking the truncation into account (local fdr is hence badly estimated)
kerfdr(pv, trans = "log")
# kerfdr by taking the truncation into account (local fdr is then well estimated)
kerfdr(pv, truncat = c(0.1, 0.9), trans = "log")
```

Index

*Topic **nonparametric**

kerfdr, [2](#)

bandwidth, [2](#)

density, [2](#)

kerfdr, [2](#)

qvalue, [2](#)