Package ‘FDX’

January 19, 2022

Type Package

Title False Discovery Exceedance Controlling Multiple Testing Procedures

Version 1.0.4

Date 2022-01-19

Maintainer Florian Junge <florian.junge@h-da.de>

Language en-US

Description Multiple testing procedures for heterogeneous and discrete tests as described in Döhler and Roquain (2019) <arXiv:1912.04607v1>. The main algorithms of the paper are available as continuous, discrete and weighted versions.

License GPL-3

Encoding UTF-8

Depends R (>= 3.00), DiscreteFDR

Imports Rcpp (>= 1.0.3), methods, PoissonBinomial (>= 1.2.0), pracma

LinkingTo Rcpp, PoissonBinomial

RoxygenNote 7.1.2

URL https://github.com/DISOhda/FDX

BugReports https://github.com/DISOhda/FDX/issues

NeedsCompilation yes

Author Sebastian Döhler [aut],
Florian Junge [aut, cre],
Etienne Roquain [ctb]

Repository CRAN

Date/Publication 2022-01-19 16:22:53 UTC

\textbf{\texttt{R} topics documented:}

FDX-package ................................................................. 2
amnesia ........................................................................... 3
FDX-package

continuous.GR ................................................. 4
continuous.LR .................................................. 6
discrete.GR ...................................................... 8
discrete.LR ...................................................... 10
discrete.PB ...................................................... 13
fast.Discrete ................................................... 15
hist.FDX .......................................................... 18
kernel ............................................................. 19
plot.FDX .......................................................... 22
print.FDX ........................................................ 23
rejection.path ................................................... 24
summary.FDX ..................................................... 26
weighted.GR ...................................................... 27
weighted.LR ..................................................... 29
weighted.PB ..................................................... 31

Index 34

FDX-package

False Discovery Exceedance (FDX) Control for Heterogeneous and Discrete Tests

Description

This package implements the [HLR], [HGR] and [HPB] procedures for both heterogeneous and discrete tests (see Reference).

Details

The functions are reorganized from the reference paper in the following way. discrete.LR (for Discrete Lehmann-Romano) implements [DLR], discrete.GR (for Discrete Guo-Romano) implements [DGR] and discrete.PB (for Discrete Poisson-Binomial) implements [DPB]. DLR and NDLR are wrappers for discrete.LR to access [DLR] and its non-adaptive version directly. Likewise, DGR, NDGR, DPB and NDPB are wrappers for discrete.GR and discrete.PB, respectively. Their main parameters are a vector of raw observed p-values and a list of the same length, whose elements are the discrete supports of the CDFs of the p-values.

In the same fashion, weighted.LR (for Weighted Lehmann-Romano), weighted.GR (for Weighted Guo-Romano) and weighted.PB (for Weighted Poisson-Binomial) implement [wLR], [wGR] and [wGR], respectively. They also possess wrapper functions, namely wLR.AM, wGR.AM and wPB.AM for arithmetic weighting, and wLR.GM, wPB.GM and wPB.GM for geometric weighting.

The functions fast.Discrete.LR, fast.Discrete.GR and fast.Discrete.PB are wrappers for fisher.pvalues.support and discrete.LR, discrete.GR and discrete.PB, respectively, which allow to apply discrete procedures directly to a data set of contingency tables.

References

Description

For each of 2446 drugs in the MHRA database (column 1), the number of cases with amnesia as an adverse event (column 2), and the number of cases with other adverse event for this drug (column 3). In total, 682648 adverse drug reactions were reported, among them 2044 cases of amnesia.

Usage

data(amnesia)

Format

A data frame with 2446 rows representing drugs with the following 3 columns:

- **DrugName**  The name of the drug.
- **AmnesiaCases**  Number of the amnesia cases reported for the drug.
- **OtherAdverseCases**  Number of other adverse drug reactions reported for the drug.

Details

The data was collected from the Drug Analysis Prints published by the Medicines and Healthcare products Regulatory Agency (MHRA), by Heller & Gur. See references for more details.

References


Source

Drug Analysis Prints on MHRA site

Examples

data.amnesia <- amnesia[, 2:3]
DGR.amnesia <- fast.Discrete.GR(counts = data.amnesia, input = "HG2011")
summary(DGR.amnesia)
continuous.GR

Continuous Guo-Romano procedure

Description

Apply the usual continuous [GR] procedure, with or without computing the critical values, to a set of p-values. A non-adaptive version is available as well.

Usage

```r
continuous.GR(
  raw.pvalues,
  alpha = 0.05,
  zeta = 0.5,
  adaptive = TRUE,
  critical.values = FALSE
)

GR(raw.pvalues, alpha = 0.05, zeta = 0.5, critical.values = FALSE)

NGR(raw.pvalues, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

Arguments

- `raw.pvalues`: vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.
- `alpha`: the target FDP, a number strictly between 0 and 1. For *.fast kernels, it is only necessary, if stepUp = TRUE.
- `zeta`: the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.
- `adaptive`: a boolean specifying whether to conduct an adaptive procedure or not.
- `critical.values`: a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).

Details

GR and NGR are wrapper functions for continuous.GR. The first one simply passes all its parameters to continuous.GR with adaptive = TRUE and NGR does the same with adaptive = FALSE.

Value

A FDX S3 class object whose elements are:

- `Rejected`: Rejected raw p-values.
- `Indices`: Indices of rejected hypotheses.
**continuous.GR**

Num.rejected  Number of rejections.

Adjusted  Adjusted p-values (only for step-down direction).

Critical.values  Critical values (if requested).

Method  A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.

FDP.threshold  FDP threshold alpha.

Exceedance.probability  Probability $zeta$ of FDP exceeding alpha; thus, FDP is being controlled at level alpha with confidence $1 - zeta$.

Adaptive  A boolean specifying whether an adaptive procedure was conducted or not.

Data$raw.pvalues  The values of raw.pvalues.

Data$data.name  The respective variable names of raw.pvalues and pCDFlist.

See Also  

kernel, FDX-package, continuous.LR, discrete.LR, discrete.GR, discrete.PB, weighted.LR, weighted.GR, weighted.PB

Examples

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support # is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

GR.fast <- GR(raw.pvalues)
summary(GR.fast)

GR.crit <- GR(raw.pvalues, critical.values = TRUE)
summary(GR.crit)

NGR.fast <- NGR(raw.pvalues)
summary(NGR.fast)

NGR.crit <- NGR(raw.pvalues, critical.values = TRUE)
summary(NGR.crit)
```
**Description**

Apply the usual (continuous) [LR] procedure, with or without computing the critical values, to a set of p-values. A non-adaptive version is available as well.

**Usage**

```r
continuous.LR(
  raw.pvalues,
  alpha = 0.05,
  zeta = 0.5,
  adaptive = TRUE,
  critical.values = FALSE
)
```

```r
LR(raw.pvalues, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

```r
NLR(raw.pvalues, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

**Arguments**

- `raw.pvalues` vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.
- `alpha` the target FDP, a number strictly between 0 and 1. For *fast* kernels, it is only necessary, if `stepUp = TRUE`.
- `zeta` the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If `zeta=NULL` (the default), then `zeta` is chosen equal to `alpha`.
- `adaptive` a boolean specifying whether to conduct an adaptive procedure or not.
- `critical.values` a boolean. If `TRUE`, critical constants are computed and returned (this is computationally intensive).

**Details**

LR and NLR are wrapper functions for `continuous.LR`. The first one simply passes all its parameters to `continuous.LR` with `adaptive = TRUE` and NLR does the same with `adaptive = FALSE`.

**Value**

A `FDX` S3 class object whose elements are:

- `Rejected` Rejected raw p-values.
- `Indices` Indices of rejected hypotheses.
Num.rejected  Number of rejections.
Adjusted       Adjusted p-values (only for step-down direction).
Critical.values Critical values (if requested).
Method         A character string describing the used algorithm, e.g.
               'Discrete Lehmann-Romano procedure (step-up)'.
FDP.threshold  FDP threshold alpha.
Exceedance.probability
               Probability $\zeta$ of FDP exceeding $\alpha$; thus, FDP is being
               controlled at level $\alpha$ with confidence $1 - \zeta$.
Adaptive       A boolean specifying whether an adaptive procedure was conducted or not.
Data$raw.pvalues
               The values of raw.pvalues.
Data$data.name
               The respective variable names of raw.pvalues and pCDFlist.

See Also

kernel, FDX-package, continuous.GR, discrete.LR, discrete.GR, discrete.PB, weighted.LR,
weighted.GR, weighted.PB

Examples

X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support
# is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support
LR.fast <- LR(raw.pvalues)
summary(LR.fast)

LR.crit <- LR(raw.pvalues, critical.values = TRUE)
summary(LR.crit)

NLR.fast <- NLR(raw.pvalues)
summary(NLR.fast)

NLR.crit <- NLR(raw.pvalues, critical.values = TRUE)
summary(NLR.crit)
discrete.GR  

**Discrete Guo-Romano procedure**

**Description**

Apply the [DGR] procedure, with or without computing the critical values, to a set of p-values and their discrete support. A non-adaptive version is available as well.

**Usage**

```r
discrete.GR(
  raw.pvalues,
  pCDFlist,
  alpha = 0.05,
  zeta = 0.5,
  adaptive = TRUE,
  critical.values = FALSE
)
```

```r
DGR(raw.pvalues, pCDFlist, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

```r
NDGR(raw.pvalues, pCDFlist, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

**Arguments**

- `raw.pvalues` vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.
- `pCDFlist` a list of the supports of the CDFs of the p-values. Each support is represented by a vector that must be in increasing order.
- `alpha` the target FDP, a number strictly between 0 and 1. For *.fast kernels, it is only necessary, if stepUp = TRUE.
- `zeta` the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.
- `adaptive` a boolean specifying whether to conduct an adaptive procedure or not.
- `critical.values` a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).

**Details**

DGR and NDGR are wrapper functions for discrete.GR. The first one simply passes all its parameters to discrete.GR with adaptive = TRUE and NDGR does the same with adaptive = FALSE.
Discrete Generalized Right Tail (GR) distribution

Value

A FDX S3 class object whose elements are:

- **Rejected**
  - Rejected raw p-values.
- **Indices**
  - Indices of rejected hypotheses.
- **Num.rejected**
  - Number of rejections.
- **Adjusted**
  - Adjusted p-values (only for step-down direction).
- **Critical.values**
  - Critical values (if requested).
- **Method**
  - A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.
- **FDP.threshold**
  - FDP threshold alpha.
- **Exceedance.probability**
  - Probability $\zeta$ of FDP exceeding $\alpha$; thus, FDP is being controlled at level $\alpha$ with confidence $1 - \zeta$.
- **Adaptive**
  - A boolean specifying whether an adaptive procedure was conducted or not.

Data$raw.pvalues

The values of raw.pvalues.

Data$pCDFlist

The values of pCDFlist.

Data$data.name

The respective variable names of raw.pvalues and pCDFlist.

References


See Also

kernel, FDX-package, continuous.LR, continuous.GR, discrete.LR, discrete.PB, weighted.LR, weighted.GR, weighted.PB

Examples

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df
```

```r
# Construction of the p-values and their supports (fisher.pvalues.support
# is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support
```
Discrete Lehmann-Romano procedure

**Description**

Apply the [DLR] procedure, with or without computing the critical values, to a set of p-values and their discrete support. Both step-down and step-up procedures can be computed and non-adaptive versions are available as well.

**Usage**

```r
discrete.LR(
  raw.pvalues, pCDFlist, alpha = 0.05, zeta = 0.5,
  direction = "sd", adaptive = TRUE,
  critical.values = FALSE)
```

```r
DLR(
  raw.pvalues, pCDFlist, alpha = 0.05, zeta = 0.5,
  direction = "sd", critical.values = FALSE)
```

```r
NDLR(
  raw.pvalues, pCDFlist, alpha = 0.05,
)```
discrete.LR

```r
zeta = 0.5,
direction = "sd",
critical.values = FALSE
)
```

**Arguments**

- `raw.pvalues` vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.
- `pCDFList` a list of the supports of the CDFs of the p-values. Each support is represented by a vector that must be in increasing order.
- `alpha` the target FDP, a number strictly between 0 and 1. For `.fast` kernels, it is only necessary, if `stepUp = TRUE`.
- `zeta` the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If `zeta=NULL` (the default), then `zeta` is chosen equal to `alpha`.
- `direction` a character string specifying whether to conduct a step-up (`direction="su"`, the default) or step-down procedure (`direction="sd"`).
- `adaptive` a boolean specifying whether to conduct an adaptive procedure or not.
- `critical.values` a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).

**Details**

DLR and NDLR are wrapper functions for `discrete.LR`. The first one simply passes all its parameters to `discrete.LR` with `adaptive = TRUE` and NDLR does the same with `adaptive = FALSE`.

**Value**

A `FDX` S3 class object whose elements are:

- `Rejected` Rejected raw p-values.
- `Indices` Indices of rejected hypotheses.
- `Num.rejected` Number of rejections.
- `Adjusted` Adjusted p-values (only for step-down direction).
- `Critical.values` Critical values (if requested).
- `Method` A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.
- `FDP.threshold` FDP threshold `alpha`.
- `Exceedance.probability` Probability `zeta` of FDP exceeding `alpha`; thus, FDP is being controlled at level `alpha` with confidence `1 - zeta`.
- `Data$raw.pvalues` The values of `raw.pvalues`.
- `Data$pCDFlist` The values of `pCDFlist`.
- `Data$data.name` The respective variable names of `raw.pvalues` and `pCDFlist`. 
References


See Also

kernel, FDX-package, continuous.LR, continuous.GR, discrete.GR, discrete.PB, weighted.LR, weighted.GR, weighted.PB

Examples

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support # is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

DLR.sd.fast <- DLR(raw.pvalues, pCDFlist)
summary(DLR.sd.fast)
DLR.su.fast <- DLR(raw.pvalues, pCDFlist, direction = "su")
summary(DLR.su.fast)

DLR.sd.crit <- DLR(raw.pvalues, pCDFlist, critical.values = TRUE)
summary(DLR.sd.crit)
DLR.su.crit <- DLR(raw.pvalues, pCDFlist, direction = "su", critical.values = TRUE)
summary(DLR.su.crit)

NDLR.sd.fast <- NDLR(raw.pvalues, pCDFlist)
summary(NDLR.sd.fast)
NDLR.su.fast <- NDLR(raw.pvalues, pCDFlist, direction = "su")
summary(NDLR.su.fast)

NDLR.sd.crit <- NDLR(raw.pvalues, pCDFlist, critical.values = TRUE)
summary(NDLR.sd.crit)
NDLR.su.crit <- NDLR(raw.pvalues, pCDFlist, direction = "su", critical.values = TRUE)
summary(NDLR.su.crit)
```
Discrete Poisson-Binomial procedure

Description
Apply the [DPB] procedure, with or without computing the critical values, to a set of p-values and their discrete support. A non-adaptive version is available as well. Additionally, the user can choose between exact computation of the Poisson-Binomial distribution or a refined normal approximation.

Usage

```r
discrete.PB(
  raw.pvalues,
  pCDFlist,
  alpha = 0.05,
  zeta = 0.5,
  adaptive = TRUE,
  critical.values = FALSE,
  exact = TRUE
)
```

```r
DPB(
  raw.pvalues,
  pCDFlist,
  alpha = 0.05,
  zeta = 0.5,
  critical.values = FALSE,
  exact = TRUE
)
```

```r
NDPB(
  raw.pvalues,
  pCDFlist,
  alpha = 0.05,
  zeta = 0.5,
  critical.values = FALSE,
  exact = TRUE
)
```

Arguments
- **raw.pvalues**: vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.
- **pCDFlist**: a list of the supports of the CDFs of the p-values. Each support is represented by a vector that must be in increasing order.
- **alpha**: the target FDP, a number strictly between 0 and 1. For *.fast* kernels, it is only necessary, if stepUp = TRUE.
zeta
the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.
adaptive
a boolean specifying whether to conduct an adaptive procedure or not.
critical.values
a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).
exact
a boolean specifying whether to compute the Poisson-Binomial distribution exactly or by a normal approximation.

Details
DPB and NDPB are wrapper functions for discrete.PB. The first one simply passes all its parameters to discrete.PB with adaptive = TRUE and NDPB does the same with adaptive = FALSE.

Value
A FDX S3 class object whose elements are:
- Rejected
  Rejected raw p-values.
- Indices
  Indices of rejected hypotheses.
- Num.rejected
  Number of rejections.
- Adjusted
  Adjusted p-values (only for step-down direction).
- Critical.values
  Critical values (if requested).
- Method
  A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.
- FDP.threshold
  FDP threshold alpha.
- Exceedance.probability
  Probability zeta of FDP exceeding alpha; thus, FDP is being controlled at level alpha with confidence 1 - zeta.
- Data$raw.pvalues
  The values of raw.pvalues.
- Data$pCDFlist
  The values of pCDFlist.
- Data$data.name
  The respective variable names of raw.pvalues and pCDFlist.

References

See Also
kernel, FDX-package, continuous.LR, continuous.GR, discrete.LR, discrete.GR, weighted.LR, weighted.GR, weighted.PB
Examples

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support
# is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

DPB.fast <- DPB(raw.pvalues, pCDFlist)
summary(DPB.fast)

DPB.crit <- DPB(raw.pvalues, pCDFlist, critical.values = TRUE)
summary(DPB.crit)

NDPB.fast <- NDPB(raw.pvalues, pCDFlist)
summary(NDPB.fast)

NDPB.crit <- NDPB(raw.pvalues, pCDFlist, critical.values = TRUE)
summary(NDPB.crit)
```

---

**fast.Discrete**

*Fast application of discrete procedures*

**Description**

Applies the [DLR], [DGR] or [DPB] procedures, without computing the critical values, to a data set of 2 x 2 contingency tables using Fisher’s exact test.

**Usage**

```r
fast.Discrete.LR(
  counts,
  alternative = "greater",
  input = "noassoc",
  alpha = 0.05,
  zeta = 0.5,
  direction = "sd",
  adaptive = TRUE
)
```
fast.Discrete.PB(
  counts,
  alternative = "greater",
  input = "noassoc",
  alpha = 0.05,
  zeta = 0.5,
  adaptive = TRUE,
  exact = FALSE
)

fast.Discrete.GR(
  counts,
  alternative = "greater",
  input = "noassoc",
  alpha = 0.05,
  zeta = 0.5,
  adaptive = TRUE
)

Arguments

counts        a data frame of 2 or 4 columns and any number of lines, each line representing a 2 x 2 contingency table to test. The number of columns and what they must contain depend on the value of the input argument, see Details of fisher.pvalues.support.

alternative   same argument as in fisher.test. The three possible values are "greater" (default), "two.sided" or "less"; may be abbreviated.

input         the format of the input data frame, see Details of fisher.pvalues.support. The three possible values are "noassoc" (default), "marginal" or "HG2011"; may be abbreviated.

alpha         the target FDP, a number strictly between 0 and 1. For *.fast kernels, it is only necessary, if stepUp = TRUE.

zeta          the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.

direction     a character string specifying whether to conduct a step-up (direction="su", the default) or step-down procedure (direction="sd").

adaptive      a boolean specifying whether to conduct an adaptive procedure or not.

exact         a boolean specifying whether to compute the Poisson-Binomial distribution exactly or by a normal approximation.

Value

A FDX S3 class object whose elements are:

Rejected  Rejected raw p-values.
Indices    Indices of rejected hypotheses.
Num.rejected Number of rejections.
**fast.Discrete**

**Adjusted**
Adjusted p-values (only for step-down direction).

**Method**
A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.

**FDP.threshold**
FDP threshold alpha.

**Exceedance.probability**
Probability zeta of FDP exceeding alpha; thus, FDP is being controlled at level alpha with confidence 1 - zeta.

**Adaptive**
A boolean specifying whether an adaptive procedure was conducted or not.

**Data$raw.pvalues**
The values of raw.pvalues.

**Data$data.name**
The respective variable names of raw.pvalues and pCDFlist.

**Examples**

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

DLR.sd <- fast.Discrete.LR(counts = df, input = "noassoc")
DLR.sd$Adjusted
summary(DLR.sd)

DLR.su <- fast.Discrete.LR(counts = df, input = "noassoc", direction = "su")
summary(DLR.su)

NDLR.sd <- fast.Discrete.LR(counts = df, input = "noassoc", adaptive = FALSE)
NDLR.sd$Adjusted
summary(NDLR.sd)

NDLR.su <- fast.Discrete.LR(counts = df, input = "noassoc", direction = "su", adaptive = FALSE)
summary(NDLR.su)

DGR <- fast.Discrete.GR(counts = df, input = "noassoc")
DGR$Adjusted
summary(DGR)

NDGR <- fast.Discrete.GR(counts = df, input = "noassoc", adaptive = FALSE)
NDGR$Adjusted
summary(NDGR)

DPB <- fast.Discrete.PB(counts = df, input = "noassoc")
DPB$Adjusted
summary(DPB)

NDPB <- fast.Discrete.PB(counts = df, input = "noassoc", adaptive = FALSE)
NDPB$Adjusted
```
hist.FDX

Summary of NDPB

hist.FDX

Histogram of Raw p-Values

Description

Computes a histogram of the raw p-values of a FDX object.

Usage

## S3 method for class 'FDX'
hist(x, breaks = "FD", main = NULL, xlab = NULL, ylab = NULL, plot = TRUE, ...)

Arguments

x an object of class "FDX".
breaks as in hist; here, the Friedman-Diaconis algorithm("FD") is used as default.
main main title. If NULL (default), a description string is used.
xlab, ylab labels for x and y axis.
plot a boolean If TRUE (the default), a histogram is plotted, otherwise a list of breaks and counts is returned.
...

Details

If x contains results of a weighted approach, a histogram of the weighted p-values is constructed. Otherwise, it is constituted by the raw ones.

Value

An object of class histogram.

Examples

X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support
# is from 'DiscreteFDR' package!)
Kernel functions

Description

Kernel functions transform observed p-values or their support according to [HLR], [PB] and [HGR]. The output is used by `discrete.LR`, `discrete.PB` and `discrete.GR`, respectively. For each procedure, there is a kernel for fast computation and one for calculation of critical values. Kernels followed by ".crit", e.g. `kernel.DGR.crit`, compute and return these critical values, while kernels ending in ".fast" only transform p-values and are therefore faster. The end user should not use these functions directly.

Usage

```r
kernel_DLR_fast(
  pCDFlist, pvalues, adaptive = TRUE, alpha = 0.05, stepUp = FALSE, zeta = 0.5, support = 0L
)

kernel_DLR_crit(
  pCDFlist, pvalues, sorted_pv, adaptive = TRUE, alpha = 0.05, zeta = 0.5, stepUp = FALSE
)

kernel_DGR_fast(pCDFlist, pvalues, adaptive = TRUE, alpha = 0.05)

kernel_DGR_crit(
  pCDFlist, pvalues,
)```
sorted_pv,
  adaptive = TRUE,
  alpha = 0.05,
  zeta = 0.5
)

kernel_DPB_fast(pCDFlist, pvalues, adaptive = TRUE, alpha = 0.05, exact = TRUE)

kernel_DPB_crit(
  pCDFlist,
  pvalues,
  sorted_pv,
  adaptive = TRUE,
  alpha = 0.05,
  zeta = 0.5,
  exact = TRUE
)

kernel_wLR_fast(qvalues, weights, alpha = 0.05, geom_weighting = FALSE)

kernel_wGR_fast(qvalues, weights, alpha = 0.05, geom_weighting = FALSE)

kernel_wPB_fast(
  qvalues,
  weights,
  alpha = 0.05,
  geom_weighting = FALSE,
  exact = TRUE
)

Arguments

pCDFlist a list of the supports of the CDFs of the p-values. Each support is represented by a vector that must be in increasing order.
pvalues a numeric vector. Contains all values of the p-values supports if we search for the critical constants. If not, contains only the observed p-values. Must be sorted in increasing order!
adaptive a boolean specifying whether to conduct an adaptive procedure or not.
alpha the target FDP, a number strictly between 0 and 1. For *.fast kernels, it is only necessary, if stepUp = TRUE.
stepUp a numeric vector. Identical to pvalues for a step-down procedure. Equals c.m for a step-up procedure.
zeta the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.
support a numeric vector. Contains all values of the p-values supports. Ignored, if stepUp = FALSE. Must be sorted in increasing order!
sorted_pv a vector of observed p-values, in increasing order.
exact: a boolean specifying whether to compute the Poisson-Binomial distribution exactly or by a normal approximation.


weights: a numeric vector. Contains the weights of the p-values.

gem_weighting: a boolean specifying whether to conduct geometric (TRUE) or arithmetic (FALSE) weighting.

Value

For "fast" kernels, a vector of transformed p-values is returned; "crit" kernels return a list object with critical constants (crit.consts) and transformed p-values (pval.transf).

See Also

FDX-package, discrete.LR, discrete.GR, discrete.PB, weighted.LR, weighted.GR, discrete.PB

Examples

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support # is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

alpha <- 0.05

# If not searching for critical constants, we use only the observed p-values
sorted.pvals <- sort(raw.pvalues)
y.DLR.fast <- kernel_DLR_fast(pCDFlist, sorted.pvals, TRUE)
y.NDGR.fast <- kernel_DGR_fast(pCDFlist, sorted.pvals, FALSE)$pval.transf

# transformed values
y.DLR.fast
y.NDGR.fast

# compute support
pv.list <- sort(unique(unlist(pCDFlist)))
y.DGR.crit <- kernel_DGR_crit(pCDFlist, pv.list, sorted.pvals, TRUE)
y.NDPB.crit <- kernel_DPB_crit(pCDFlist, pv.list, sorted.pvals, FALSE)

# critical constants
y.DGR.crit$crit.consts
y.NDPB.crit$crit.consts

# transformed values
```
Description

Plots raw p-values of a FDX object and highlights rejected and accepted p-values. If present, the critical values are plotted, too.

Usage

```r
## S3 method for class 'FDX'
plot(
x,
col = c(2, 4, 1),
pch = c(1, 1, 1),
lwd = c(1, 1, 1),
type.crit = "b",
legend = NULL,
...
)
```

Arguments

- `x`: an object of class "FDX".
- `col`: a numeric or character vector of length 3 indicating the colors of the
  1. rejected p-values
  2. accepted p-values
  3. critical values (if present).
- `pch`: a numeric or character vector of length 3 indicating the point characters of the
  1. rejected p-values
  2. accepted p-values
  3. critical values (if present and `type.crit` is a plot type like 'p', 'b' etc.).
- `lwd`: a numeric vector of length 3 indicating the thickness of the points and lines.
- `type.crit`: 1-character string giving the type of plot desired for the critical values (e.g.: 'p', 'l' etc; see `plot`).
- `legend`: if NULL, no legend is plotted; otherwise expecting a character string like "topleft" etc. or a numeric vector of two elements indicating (x, y) coordinates.
- `...`: further arguments to `plot.default`.
Details
If x contains results of a weighted approach, the Y-axis of the plot is derived from the weighted p-values. Otherwise, it is constituted by the raw ones.

Examples
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support # is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

DLR.sd.fast <- DLR(raw.pvalues, pCDFlist)
DLR.sd.crit <- DLR(raw.pvalues, pCDFlist, critical.values = TRUE)
DLR.su.fast <- DLR(raw.pvalues, pCDFlist, direction = "su")
DLR.su.crit <- DLR(raw.pvalues, pCDFlist, direction = "su", critical.values = TRUE)

plot(DLR.su.fast)
plot(DLR.su.crit, xlim = c(1, 5), ylim = c(0, 0.4))
plot(DLR.sd.fast, col = c(2, 4), pch = c(2, 3), lwd = c(2, 2),
     legend = "topleft", xlim = c(1, 5), ylim = c(0, 0.4))
plot(DLR.sd.crit, col = c(2, 4, 1), pch = c(1, 1, 4), lwd = c(1, 1, 2),
     type.crit = 'o', legend = c(1, 0.4), lty = 1, xlim = c(1, 5),
     ylim = c(0, 0.4))

print.FDX

Description
Prints the results of discrete FDX analysis, stored in a FDX S3 class object.

Usage
## S3 method for class 'FDX'
print(x, ...)

rejection.path

Arguments

x

an object of class "FDX".

... further arguments to be passed to or from other methods. They are ignored in this function.

Value

The respective input object is invisibly returned via invisible(x).

Examples

X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support # is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

DPB.crit <- DPB(raw.pvalues, pCDFlist, critical.values = TRUE)
print(DPB.crit)

rejection.path  Rejection Path Plot (for FDX objects)

Description

Displays the number of rejections of the raw p-values in a FDX object in dependence of the exceedance probability zeta.

Usage

rejection.path(
  x,
  xlim = NULL,
  ylim = NULL,
  main = NULL,
  xlab = expression(zeta),
  ylab = "Number of Rejections",
  verticals = FALSE,
pch = 19,
ref.show = FALSE,
ref.col = "gray",
ref.lty = 2,
ref.lwd = 2,
...
)

Arguments

x an object of class "FDX".
xlim the x limits of the plot. If NULL (default), the (0, 1) range is used.
ylim the y limits of the plot. If NULL (default), the double of the median of the number of possible rejections is used as upper limit.
main main title. If NULL (default), a description string is used.
xlab, ylab labels for x and y axis.
verticals logical; if TRUE, draw vertical lines at steps.
pch jump point character.
ref.show logical; if TRUE a vertical reference line is plotted, whose height is the number of rejections of the original Benjamini-Hochberg (BH) procedure.
ref.col color of the reference line.
ref.lty, ref.lwd line type and thickness for the reference line.
... further arguments to plot.stepfun.

Value

Invisibly returns a stepfun object that computes the number of rejections in dependence on the exceedance probability \( \zeta \).

Examples

X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support
# is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

DLR <- DLR(raw.pvalues, pCDFlist)
NDLR <- NDLR(raw.pvalues, pCDFlist)

rejection.path(DLR, xlim = c(0, 1), ref.show = TRUE, ref.col = "green", ref.lty = 4)
rejection.path(NDLR, col = "red", add = TRUE)

---

**summary.FDX**

**Summarizing Discrete FDX Results**

**Description**

summary method for class "FDX"

**Usage**

```r
## S3 method for class 'FDX'
summary(object, ...)

## S3 method for class 'summary.FDX'
print(x, max = NULL, ...)
```

**Arguments**

- `object` an object of class "FDX".
- `...` further arguments passed to or from other methods.
- `x` an object of class "summary.FDX".
- `max` numeric or NULL, specifying the maximal number of rows of the p-value table to be printed. By default, when NULL, `getOption("max.print")` is used.

**Details**

`summary.FDX` objects include all data of an FDX object, but also include an additional table which includes the raw p-values, their indices, the respective critical values (if present), the adjusted p-values (if present) and a logical column to indicate rejection. The table is sorted in ascending order by the raw p-values.

`print.summary.FDX` simply prints the same output as `print.FDX`, but also prints the p-value table.

**Value**

`summary.FDX` computes and returns a list that includes all the data of an input FDX, plus

Table a data.frame, sorted by the raw p-values, that contains the indices, that raw p-values themselves, their respective critical values (if present), their adjusted p-values (if present) and a logical column to indicate rejection.

`print.summary.FDX` returns that object invisibly.
Examples

```r
X1 <- c(4, 2, 2, 14, 6, 9, 4, 0, 1)
X2 <- c(0, 0, 1, 3, 2, 1, 2, 2, 2)
N1 <- rep(148, 9)
N2 <- rep(132, 9)
Y1 <- N1 - X1
Y2 <- N2 - X2
df <- data.frame(X1, Y1, X2, Y2)
df

# Construction of the p-values and their supports (fisher.pvalues.support
# is from 'DiscreteFDR' package!)
df.formatted <- fisher.pvalues.support(counts = df, input = "noassoc")
raw.pvalues <- df.formatted$raw
pCDFlist <- df.formatted$support

DGR.crit <- DGR(raw.pvalues, pCDFlist, critical.values = TRUE)
DGR.crit.summary <- summary(DGR.crit)
print(DGR.crit.summary)
```

weighted.GR

Weighted Guo-Romano Procedure

Description

Apply the weighted [wGR] procedure, with or without computing the critical values, to a set of p-values. Both arithmetic and geometric weighting are available.

Usage

```r
weighted.GR(
  raw.pvalues,
  weights,
  alpha = 0.05,
  zeta = 0.5,
  weighting.method = "AM",
  critical.values = FALSE
)
```

```r
wGR.AM(raw.pvalues, weights, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

```r
wGR.GM(raw.pvalues, weights, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>raw.pvalues</td>
<td>vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.</td>
</tr>
</tbody>
</table>
weights a numeric vector. Contains the weights of the p-values.
alpha the target FDP, a number strictly between 0 and 1. For \( * \). fast kernels, it is only necessary, if stepUp = TRUE.
zeta the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.
weighting.method a character string specifying whether to conduct arithmetic (direction="AM", the default) or geometric weighting (direction="GM") of p-values.
critical.values a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).

Details

wGR.AM and wGR.GM are wrapper functions for weighted.GR. The first one simply passes all its parameters to weighted.GR with weighting.method = "AM" and wGR.GM does the same with weighting.method = "GM".

Value

A FDX S3 class object whose elements are:

- **Rejected** Rejected raw p-values.
- **Indices** Indices of rejected hypotheses.
- **Num.rejected** Number of rejections.
- **Adjusted** Adjusted p-values (only for step-down direction).
- **Weighted** Weighted p-values.
- **Critical.values** Critical values (if requested).
- **Method** A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.
- **FDP.threshold** FDP threshold alpha.
- **Exceedance.probability** Probability zeta of FDP exceeding alpha; thus, FDP is being controlled at level alpha with confidence 1-zeta.
- **Weighting** A character string describing the weighting method.

Data$raw.pvalues

The values of raw.pvalues.

Data$weights

The values of weights.

Data$data.name

The respective variable names of raw.pvalues and pCDFlist.

References

See Also

kernel, FDX-package, continuous.LR, continuous.GR, discrete.LR, discrete.GR, discrete.PB, weighted.LR, weighted.PB

Examples

# Construction of the p-values and their supports for weighted methods
raw.pvalues.weighted <- c(0.7389727, 0.1882310, 0.1302457, 0.9513677,
                         0.7592122, 0.0100559, 0.0000027, 0.1651034)
weights <- c(0.7947122, 1.2633867, 2.8097858, 2.2112801,
             2.3878654, 1.2389620, 2.3878654, 0.7947122)

wGR.AM.fast <- wGR.AM(raw.pvalues.weighted, weights)
summary(wGR.AM.fast)

wGR.AM.crit <- wGR.AM(raw.pvalues.weighted, weights, critical.values = TRUE)
summary(wGR.AM.crit)

wGR.GM.fast <- wGR.GM(raw.pvalues.weighted, weights)
summary(wGR.GM.fast)

wGR.GM.crit <- wGR.GM(raw.pvalues.weighted, weights, critical.values = TRUE)
summary(wGR.GM.crit)

weighted.LR

Weighted Lehmann-Romano Procedure

Description

Apply the weighted [wLR] procedure, with or without computing the critical values, to a set of p-values. Both arithmetic and geometric weighting are available.

Usage

weighted.LR(
    raw.pvalues, weights,
    alpha = 0.05, zeta = 0.5,
    weighting.method = "AM",
    critical.values = FALSE
)

wLR.AM(raw.pvalues, weights, alpha = 0.05, zeta = 0.5, critical.values = FALSE)

wLR.GM(raw.pvalues, weights, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
Arguments

raw.pvalues vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.

weights a numeric vector. Contains the weights of the p-values.

alpha the target FDP, a number strictly between 0 and 1. For *.fast kernels, it is only necessary, if stepUp = TRUE.

zeta the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.

weighting.method a character string specifying whether to conduct arithmetic (direction="AM", the default) or geometric weighting (direction="GM") of p-values.

critical.values a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).

Details

wLR.AM and wLR.GM are wrapper functions for weighted.LR. The first one simply passes all its parameters to weighted.LR with weighting.method = "AM" and wLR.GM does the same with weighting.method = "GM".

Value

A FDX S3 class object whose elements are:

Rejected Rejected raw p-values.
Indices Indices of rejected hypotheses.
Num.rejected Number of rejections.
Adjusted Adjusted p-values (only for step-down direction).
Weighted Weighted p-values.
Critical.values Critical values (if requested).
Method A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.
FDP.threshold FDP threshold alpha.
Exceedance.probability Probability zeta of FDP exceeding alpha; thus, FDP is being controlled at level alpha with confidence \(1 - zeta\).
Weighting A character string describing the weighting method.
Data$raw.pvalues The values of raw.pvalues.
Data$weights The values of weights.
Data$data.name The respective variable names of raw.pvalues and pCDFlist.
See Also

kernel, FDX-package, continuous.LR, continuous.GR, discrete.LR, discrete.GR, discrete.PB, weighted.GR, weighted.PB

Examples

# Construction of the p-values and their supports for weighted methods
raw.pvalues.weighted <- c(0.7389727, 0.1882310, 0.1302457, 0.9513677,
                         0.7592122, 0.0100559, 0.0000027, 0.1651034)
weights <- c(0.7947122, 1.2633867, 2.8097858, 2.2112801,
             2.3878654, 1.2389620, 2.3878654, 0.7947122)

wLR.AM.fast <- wLR.AM(raw.pvalues.weighted, weights)
summary(wLR.AM.fast)

wLR.AM.crit <- wLR.AM(raw.pvalues.weighted, weights, critical.values = TRUE)
summary(wLR.AM.crit)

wLR.GM.fast <- wLR.GM(raw.pvalues.weighted, weights)
summary(wLR.GM.fast)

wLR.GM.crit <- wLR.GM(raw.pvalues.weighted, weights, critical.values = TRUE)
summary(wLR.GM.crit)

---

weighted.PB                  Weighted Poisson-Binomial Procedure

Description

Apply the weighted [wPB] procedure, with or without computing the critical values, to a set of
p-values. Both arithmetic and geometric weighting are available. Additionally, the user can choose
between exact computation of the Poisson-Binomial distribution or a refined normal approximation.

Usage

weighted.PB(  
  raw.pvalues,  
  weights,  
  alpha = 0.05,  
  zeta = 0.05,  
  weighting.method = "AM",  
  critical.values = FALSE,  
  exact = TRUE  
)

wPB.AM(raw.pvalues, weights, alpha = 0.05, zeta = 0.5, critical.values = FALSE)

wPB.GM(raw.pvalues, weights, alpha = 0.05, zeta = 0.5, critical.values = FALSE)
Arguments

*raw.pvalues* vector of the raw observed p-values, as provided by the end user and before matching with their nearest neighbor in the CDFs supports.

*weights* a numeric vector. Contains the weights of the p-values.

*alpha* the target FDP, a number strictly between 0 and 1. For *.fast kernels, it is only necessary if stepUp = TRUE.

*zeta* the target probability of not exceeding the desired FDP, a number strictly between 0 and 1. If zeta=NULL (the default), then zeta is chosen equal to alpha.

*weighting.method* a character string specifying whether to conduct arithmetic (direction="AM", the default) or geometric weighting (direction="GM") of p-values.

*critical.values* a boolean. If TRUE, critical constants are computed and returned (this is computationally intensive).

*exact* a boolean specifying whether to compute the Poisson-Binomial distribution exactly or by a normal approximation.

Details

`wPB.AM` and `wPB.GM` are wrapper functions for `weighted.PB`. The first one simply passes all its parameters to `weighted.PB` with `weighting.method = "AM"` and `wPB.GM` does the same with `weighting.method = "GM"`.

Value

A FDX S3 class object whose elements are:

- **Rejected** Rejected raw p-values.
- **Indices** Indices of rejected hypotheses.
- **Num.rejected** Number of rejections.
- **Adjusted** Adjusted p-values (only for step-down direction).
- **Weighted** Weighted p-values.
- **Critical.values** Critical values (if requested).
- **Method** A character string describing the used algorithm, e.g. 'Discrete Lehmann-Romano procedure (step-up)'.
- **FDP.threshold** FDP threshold alpha.
- **Exceedance.probability** Probability zeta of FDP exceeding alpha; thus, FDP is being controlled at level alpha with confidence 1 -zeta.
- **Weighting** A character string describing the weighting method.
- **Data$raw.pvalues** The values of `raw.pvalues`.
- **Data$weights** The values of `weights`.
- **Data$data.name** The respective variable names of `raw.pvalues` and `pCDFlist`. 
See Also

kernel, FDX-package, continuous.LR, continuous.GR, discrete.LR, discrete.GR, discrete.PB, weighted.LR, weighted.GR

Examples

```r
# Construction of the p-values and their supports for weighted methods
raw.pvalues.weighted <- c(0.7389727, 0.1882310, 0.1302457, 0.9513677, 0.7592122, 0.0100559, 0.0000027, 0.1651034)
weights <- c(0.7947122, 1.2633867, 2.8097858, 2.2112801, 2.3878654, 1.2389620, 2.3878654, 0.7947122)

wPB.AM.fast <- wPB.AM(raw.pvalues.weighted, weights)
summary(wPB.AM.fast)

wPB.AM.crit <- wPB.AM(raw.pvalues.weighted, weights, critical.values = TRUE)
summary(wPB.AM.crit)

wPB.GM.fast <- wPB.GM(raw.pvalues.weighted, weights)
summary(wPB.GM.fast)

wPB.GM.crit <- wPB.GM(raw.pvalues.weighted, weights, critical.values = TRUE)
summary(wPB.GM.crit)
```
Index

* datasets
  - amnesia, 3
  - continuous.GR, 4, 7, 9, 12, 14, 29, 31, 33
  - continuous.LR, 5, 6, 9, 12, 14, 29, 31, 33
  - discrete.GR, 5, 7, 8, 12, 14, 19, 21, 29, 31, 33
  - discrete.LR, 5, 7, 9, 10, 14, 19, 21, 29, 31, 33
  - discrete.PB, 5, 7, 9, 12, 13, 19, 21, 29, 31, 33
  - DGR (discrete.GR), 8
  - DLR (discrete.LR), 10
  - DPB (discrete.PB), 13
    - fast.Discrete, 15
    - FDX-package, 2
    - fisher.pvalues.support, 2, 16
    - fisher.test, 16
    - GR (continuous.GR), 4
    - hist, 18
    - hist.FDX, 18
    - kernel, 5, 7, 9, 12, 14, 19, 29, 31, 33
      - kernel_DGR_crit (kernel), 19
      - kernel_DGR_fast (kernel), 19
      - kernel_DLR_crit (kernel), 19
      - kernel_DLR_fast (kernel), 19
      - kernel_DPB_crit (kernel), 19
      - kernel_DPB_fast (kernel), 19
      - kernel_wGR_fast (kernel), 19
      - kernel_wLR_fast (kernel), 19
      - kernel_wPB_fast (kernel), 19
    - LR (continuous.LR), 6
    - NDGR (discrete.GR), 8
    - NDLR (discrete.LR), 10
    - NDPB (discrete.PB), 13
  - NGR (continuous.GR), 4
  - NLR (continuous.LR), 6
    - plot, 22
    - plot.default, 22
    - plot.FDX, 22
    - plot.histogram, 18
    - plot.stepfun, 25
    - print.FDX, 23
    - print.summary.FDX (summary.FDX), 26
    - rejection.path, 24
    - summary.FDX, 26
    - weighted.GR, 5, 7, 9, 12, 14, 21, 27, 31, 33
    - weighted.LR, 5, 7, 9, 12, 14, 21, 29, 29, 33
    - weighted.PB, 5, 7, 9, 12, 14, 29, 31, 31
    - wGR.AM (weighted.GR), 27
    - wGR.GM (weighted.GR), 27
    - wLR.AM (weighted.LR), 29
    - wLR.GM (weighted.LR), 29
    - wPB.AM (weighted.PB), 31
    - wPB.GM (weighted.PB), 31

34