

SSPA

February 8, 2012

PilotData-class *Class PilotData*

Description

The "[PilotData](#)" class is a container class. It contains all necessary information of the pilot data for performing the power and sample size analysis.

Objects from the Class

Objects can be created by calling the function [pilotData](#).

Slots

name: name of the pilot data, type "character"

testStatistics: vector of test statistics, type "numeric"

sampleSize: effective sample size $(\frac{1}{n_A} + \frac{1}{n_B})^{-1}$ with n_A the sample size in group A, type "numeric"

pValues: vector of p-values, type "numeric"

dof: degrees of freedom only used for Student t distribution, type "numeric"

nullDist: null distribution either normal or Student t, type "character"

Methods

[show](#), [PilotData-method](#), [hist](#) and [plot](#)

Author(s)

Maarten van Iterson

See Also

[pilotData](#), [SampleSize-class](#)

Power

Estimates the average power of the pilot data

Description

The function `Power` estimates the average power of the pilot data at a given false discovery rate. The average power can be estimated for sample sizes other than the pilot data.

Usage

```
Power(x, threshold = 0, fdr = 0.1, samplesizes = NULL,
      plot = FALSE, type = "l", ylim = c(0, 1), xlim = c(0, 1), xlab = "p-value",
      ylab = "average power", main, sub, ...)
```

Arguments

<code>x</code>	object of class <code>SampleSize-class</code>
<code>threshold</code>	threshold for truncation of the density of effect-sizes. A small symmetrical region around zero can be defined that will be excluded from the density of effect sizes.
<code>fdr</code>	numeric given the desired false discovery rate threshold, possibly a vector like: <code>c(0.1, 0.05)</code>
<code>samplesizes</code>	numeric vector, giving the samplesizes for which the power will be estimated
<code>plot</code>	logical if <code>TRUE</code> plots the power curve and intersection for the given <code>fdr</code> (default <code>plot=FALSE</code>).
<code>type</code>	what type of plot should be drawn
<code>ylim</code>	range of y values
<code>xlim</code>	range of x values
<code>xlab</code>	a title for the x axis
<code>ylab</code>	a title for the y axis
<code>main</code>	an overall title for the plot
<code>sub</code>	a sub title for the plot
<code>...</code>	additional arguments to <code>plot</code> or <code>par</code>

Details

Having estimated the proportion of non-differentially expressed genes and the density of effect-sizes an estimate of the average power is calculated at a given false discovery rate and sample size. Power and sample size analysis can be performed by estimating the average power for sample sizes other than the pilot data. Ferreira and Zwinderman (2006) show that the average power (the proportion of rejected hypothesis out of all alternative hypothesis) of the adaptive Benjamini-Hochberg procedure is estimated by the point where the CDF of p-values under the alternative hypothesis intersects the straight line with a slope determined by the given FDR and the estimated proportion of non-differentially expressed genes. Using `plot=TRUE` the CDF of p-values under the alternative hypothesis is shown with the intersecting lines for different FDRs and sample sizes. Don't use more than 2 FDRs and more than 3 different sample sizes because the plot will then be unclear.

Value

estimated average power, probably a named matrix with on the rows different sample sizes and on the columns the different false discovery rates.

Author(s)

Maarten van Iterson

References

Ferreira, F.A., Zwinderman, A., (2006). Approximate Power and Sample Size Calculations with Microarray Data: An Illustration. *Statistical Applications in Genetics and Molecular Biology* 5, (1).

Examples

```
library(multtest)
data(golub)
teststat <- mt.teststat(golub, golub.cl)
table(golub.cl)
pd <- pilotData(name="golub", testStatistics=teststat, sampleSizeA=11, sampleSizeB=27)
hist(pd)
plot(pd)
ss <- sampleSize(pd)
plotEffectSize(ss)
Power(ss)

##creating a plot estimate power vs. sample size
pwr <- Power(ss, plot = FALSE, samplesizes = c(5, 10, 15, 20), fdr=0.01)
plot(c(5, 10, 15, 20), pwr, ylim = c(0, 1), type = "b", ylab = "Power", xlab = "Sample size",
legend("bottomright", colnames(pwr), col=c(1:ncol(pwr)), pch=1, lty=1)

##creating a plot estimate power vs. sample size for different false discovery rates
pwr <- Power(ss, plot = FALSE, samplesizes = c(5, 10, 15, 20), fdr=c(0.01, 0.05))
matplot(c(5, 10, 15, 20), pwr, ylim = c(0, 1), type = "b", pch=1, ylab = "Power", xlab = "Sample size",
legend("bottomright", colnames(pwr), col=c(1:ncol(pwr)), pch=1, lty=1)
```

Internal functions *internal functions*

Description

internal functions; these are not to be called by the user.

SampleSize-class *Class SampleSize*

Description

By constructing an object of `SampleSize` the power and sample size analysis will be performed.

Objects from the Class

Objects can be created by calling the function `sampleSize`.

Slots

`pi0`: proportion of non-differentially expressed genes, "named-list"
`lambda`: vector containing the estimated density of effect sizes, "numeric"
`theta`: vector containing the range over which the density of effect sizes is estimated, "numeric"
`adjust`: is the density of effect sizes adjusted (`adjust=TRUE`) or not (`adjust=FALSE`), "logical"
`method`: method used for the estimation of the density of effect sizes, "character"
`bandwidth`: Default bandwidth given by $\log_{10}(n)^{-\frac{1}{2}}$ where n is the number of genes.
`kernel`: kernel used for the deconvolution estimator, "character"
`nKnots`: number of knots used for the B-splines (only visible if `method="Ruppert"`)
`bDegree`: B-spline degree(only visible if `method="Ruppert"`)
`name`: name of the pilot data, type "character"
`testStatistics`: vector of test statistics, type "numeric"
`sampleSize`: effective sample size $(\frac{1}{n_A} + \frac{1}{n_B})^{-1}$ with n_A the sample size in group A, type "numeric"
`pValues`: vector of p-values, type "numeric"
`dof`: degrees of freedom only used for Student t distribution, type "numeric"
`nullDist`: null distribution either normal or Student t, type "character"

Extends

Class `PilotData`, directly.

Methods

`show`, `SampleSize-method`, `Power` and `plotEffectSize`

Author(s)

Maarten van Iterson

See Also

`pilotData`, `PilotData`

`hist`*Histogram of p-values of the pilot data*

Description

The function `hist` computes a histogram of p-values of the given pilot data. Two-sided p-values are computed either using the normal or Student t distribution based on the test statistics that are stored in an object of `PilotData`.

Usage

```
hist(x, ...)
```

Arguments

<code>x</code>	object of <code>PilotData</code>
<code>...</code>	additional arguments to <code>hist</code>

Details

p-values calculated under the null hypothesis (non-differentially expressed genes) are assumed to be uniformly distributed on [0,1]. p-values calculated under the alternative hypothesis (differentially expressed genes) are assumed to be accumulated near zero. The height of the flat part of the histogram is an indication of how many hypothesis are calculated under the null hypothesis.

Value

an object of class "histogram", see `hist`.

Author(s)

Maarten van Iterson

See Also

`hist` and `plot`

Examples

```
library(multtest)
data(golub)
teststat <- mt.teststat(golub, golub.cl)
table(golub.cl)
pd <- pilotData(name="golub", testStatistics=teststat, sampleSizeA=11, sampleSizeB=27)
hist(pd)
```

pilotData	<i>Creates an object of class "PilotData"</i>
-----------	---

Description

The function `pilotData` initializes a `PilotData`-object. Information of the pilot data and the null distribution is stored e.g. name of pilot experiment, test statistics and number of samples used.

Usage

```
pilotData(name = "Unknown Experiment", testStatistics = double(1), sampleSizeA =
```

Arguments

<code>name</code>	character string giving the experiment name
<code>testStatistics</code>	vector of type numeric containing the set of test statistics
<code>sampleSizeA</code>	the samplesize of group A
<code>sampleSizeB</code>	the samplesize of group B
<code>dof</code>	degree of freedom for a Student t distribution
<code>nullDist</code>	distribution under the null hypothesis either one of: <ul style="list-style-type: none">• "normal",• "student", (only in case of method="Ruppert" see <code>sampleSize</code>)

Details

Based on the given null distribution two-sided p-values are calculated from the test statistics. Some additional checks on the data are performed. Once an object of `PilotData` is created the `sampleSize`-function can be called.

Value

object of class `PilotData`

Author(s)

Maarten van Iterson

See Also

`hist`, `plot` and `PilotData` use `class?PilotData`

Examples

```
library(multtest)
data(golub)
teststat <- mt.teststat(golub, golub.cl)
table(golub.cl)
pd <- pilotData(name="golub", testStatistics=teststat, sampleSizeA=11, sampleSizeB=27)
```

`plot`*Empirical cumulative distribution function of p-values of the pilot data*

Description

The function `plot` plots an empirical cumulative distribution function of p-values of the pilot data. The test statistics that are given as input to `PilotData-class` are transformed to two-sided p-values either using the normal or Student t distribution.

Usage

```
plot(x, y, ...)
```

Arguments

<code>x</code>	object of class <code>PilotData-class</code>
<code>y</code>	unused item
<code>...</code>	additional arguments given to <code>plot</code> or <code>par</code>

Details

Empirical cumulative distribution function of p-values. The line at angle of 45 degrees represents the theoretical CDF of a uniform distribution as expected when all genes are non-differentially expressed. An accumulation of p-values near zero indicates a certain number of differentially expressed genes.

Author(s)

Maarten van Iterson

Examples

```
library(multtest)
data(golub)
teststat <- mt.teststat(golub, golub.cl)
table(golub.cl)
pd <- pilotData(name="golub", testStatistics=teststat, sampleSizeA=11, sampleSizeB=27)
hist(pd)
plot(pd)
```

`plotEffectSize`*Plots the density of effect sizes of the pilot data*

Description

The function `plotEffectSize` plots density of effect sizes of the pilot data.

Usage

```
plotEffectSize(x, threshold = 0, xlab = "effect size", ylab = "density of effect
```

Arguments

x	object of class <code>SampleSize-class</code>
threshold	threshold for truncation of the density of effect-sizes. The threshold will be taken symmetrical around the y-axis.
xlab	a title for the x axis
ylab	a title for the y axis
main	an overall title for the plot
sub	a sub title for the plot
...	additional arguments given to <code>plot</code> or <code>par</code>

Details

The density of effect sizes describes the effects observed in the pilot data. Usually a bimodal density is observed representing up- and down-regulated genes. The way in which the test statistics is calculated determines what is meant by up- and down-regulation. A small symmetrical region around zero can be defined that will be excluded from the density of effect sizes and thereby increases the estimated average power.

References

Ferreira, F.A., Zwinderman, A., (2006). Approximate Power and Sample Size Calculations with Microarray Data: An Illustration. *Statistical Applications in Genetics and Molecular Biology* 5, (1).

Examples

```
library(multtest)
data(golub)
teststat <- mt.teststat(golub, golub.cl)
table(golub.cl)
pd <- pilotData(name="golub", testStatistics=teststat, sampleSizeA=11, sampleSizeB=27)
hist(pd)
plot(pd)
ss <- sampleSize(pd)
plotEffectSize(ss)
```

sampleSize

Creates an object of class SampleSize

Description

The function `sampleSize` initializes a `SampleSize`-object. The density of effect-sizes is estimated using a deconvolution estimator or constrained optimization using B-splines.

Usage

```
sampleSize(PilotData, method = c("Langaas", "Storey", "Ferreira", "Ruppert", "Us
```

Arguments

PilotData	object of class <code>PilotData</code> on which the sample size and power analysis will be performed
method	character string giving the method for estimation of the fraction of non-differentially expressed genes either one of: <ul style="list-style-type: none"> • "Langaas" (default) • "Storey" • "Ferreira" • "Ruppert" • "Userdefined"
from	Lower bound of the density of effect-sizes (the range should be symmetric)
to	Upper bound of the density of effect-sizes.
resolution	the number of points on which the density of effect-sizes will be estimated (must be a power of 2)
kernel	the kernel type used in the deconvolution estimator either one of: <ul style="list-style-type: none"> • "fan" (default) • "wand" • "sinc"
pi0	numeric or a vector of type numeric giving the fraction of non-differentially expressed genes. If <code>method="Userdefined"</code> a userdefined <code>pi0</code> is obligated. If <code>method="Ferreira"</code> a range of values should be given e.g. <code>seq(0.01, 0.99, 0.01)</code>
adjust	is the density of effect sizes adjusted (<code>adjust=TRUE</code> , default) or not (<code>adjust=FALSE</code>)
a	For better upper bound estimates of <code>pi0</code> as described by Efron <i>et al.</i> (2001).
nKnots	number of knots used in Rupperts method for estimating the proportion of non-differentially expressed genes and density of effect-sizes
bDegree	degree of B-spline basis used in Rupperts method for estimating the proportion of non-differentially expressed genes and density of effect-sizes
...	Additional parameters for method for the estimation of the fraction of non-differentially expressed genes (Doesn't work yet!)

Details

The `sampleSize` function performs the estimation of the proportion of non-differentially expressed genes using one of the three methods, "Langaas", "Storey", "Ferreira" if `method="Userdefined"` a userdefined `pi0` is needed and estimation of the proportion of non-differentially expressed genes will be skipped. A deconvolution estimator is implemented using the Fast Fourier Transform Algorithm `fft()` for estimations of the density of effect sizes. If `method="Ruppert"` constrained optimization using B-splines is used, for this method two additional packages needs to be installed namely `quadprog` and `splines`. Both the proportion of non-differentially expressed genes and the density of effect sizes are estimated with Ruppert's method. In contrast to the original method of Ruppert *et al.* (2007) we made a modification on the estimation of the density of effect sizes allowing for negative effect sizes as well.

Value

Object of class `SampleSize`

Author(s)

Maarten van Iterson

References

- Ferreira, F.A., Zwinderman, A., (2006). Approximate Power and Sample Size Calculations with Microarray Data: An Illustration. *Statistical Applications in Genetics and Molecular Biology* 5, (1).
- Ferkingstad, E., Langaas, M., and Lindqvist, B. (2005). Estimating the proportion of true null hypotheses, with application to DNA microarray data. *Journal of the Royal Statistical Society Series B*, 67, 555-572.
- Storey, J.D., (2002). A direct approach to false discovery rates. *J.R. Statist. Med.* 27, 1960-1972.
- Ruppert, D. and Nettleton, D. and Hwang, J.T.G., (2007). Exploring the information in p-values for the analysis and planning of multiple-test experiments. *Biometrics*, 63, 2, 483-95.
- Efron, B. and Tibshirani, R. and Storey, J.D. and Tusher, V. (2001) Empirical Bayes Analysis of a Microarray Experiment. *Journal of the American Statistical Association*, 96, 456, 1151-1160.

See Also

[convest](#), [qvalue](#) and [SampleSize](#) use `class?SampleSize`

Examples

```
library(multtest)
data(golub)
teststat <- mt.teststat(golub, golub.cl)
table(golub.cl)
pd <- pilotData(name="golub", testStatistics=teststat, sampleSizeA=11, sampleSizeB=27)
hist(pd)
plot(pd)
ss <- sampleSize(pd)
```

show-methods

Nicely showing objects of class `pilotData` and `sampleSize`

Description

Nicely showing objects from "[PilotData](#)" and "[SampleSize](#)"

Methods

object = "PilotData" object from class "[PilotData](#)"

object = "SampleSize" object from class "[SampleSize](#)"

Index

*Topic classes

pilotData, 6
PilotData-class, 1
sampleSize, 8
SampleSize-class, 4

*Topic hplot

hist, 5
plot, 7
plotEffectSize, 7
Power, 2

*Topic methods

Internal functions, 3
show-methods, 10

bspline(*Internal functions*), 3

constrainedOptimization
(*Internal functions*), 3
convest, 10

deconvolution(*Internal
functions*), 3
Dn(*Internal functions*), 3

Gnhat(*Internal functions*), 3
Gntilde(*Internal functions*), 3

hist, 1, 5, 5
hist, PilotData-method(*hist*), 5
Hntilde(*Internal functions*), 3

Internal functions, 3

par, 2, 7, 8
PilotData, 1, 4–6, 9, 10
pilotData, 1, 4, 6
PilotData-class, 7
PilotData-class, 1
plot, 1, 2, 5, 7, 7, 8
plot, ANY, ANY-method(*plot*), 7
plot, PilotData, missing-method
(*plot*), 7
plot-methods(*plot*), 7
plotEffectSize, 4, 7
Power, 2, 4

psreg(*Internal functions*), 3

qvalue, 10

SampleSize, 8–10
sampleSize, 4, 6, 8
SampleSize-class, 1, 2, 8
SampleSize-class, 4
show, PilotData-method, 1
show, PilotData-method
(*show-methods*), 10
show, SampleSize-method, 4
show, SampleSize-method
(*show-methods*), 10
show-methods, 10