tweeDEseq

March 24, 2012

---

compareCountDist   *Compare count data distributions*

Description

Compares the empirical and estimated distributions for different count data models

Usage

```
compareCountDist(x, plot=TRUE, ...)
```

Arguments

x   numeric vector containing the read counts.

plot   If TRUE (the default) then the plot with the ECDF function for the counts and the three different Poisson-Tweedie distributions is produced, otherwise no graphical output is given and this only makes sense if one is interested in the returned value (see value section below).

...   Further arguments to be passed to the plot function.

Details

This function serves the purpose of comparing a empirical distribution of counts with three Poisson-Tweedie distributions arising from estimating mean, dispersion and setting $a = 1$ for comparing against a Poisson, $a = 0$ for comparing against a negative binomial and estimating the shape parameter a from data too. The legend shows the values of the a parameter and the P-value of the likelihood ratio test on whether the expression profile follows a negative binomial distribution ($H_0 : a = 0$).

Value

List with the following components:

a   shape parameter estimated from the input data x.

p.value   P-value for the test that the data follows a negative binomial distribution, i.e., $H_0 : a = 0$.

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

See Also

[qqchisq](#) [testShapePT](#)

Examples

```
# Generate 500 random counts following a Poisson Inverse Gaussian
# distribution with mean = 20 and dispersion = 5
randomCounts <- rPT(n = 500, mu = 20, D = 5, a = 0.5)

xx <- compareCountDist(randomCounts, plot=FALSE)
xx
```

---

dPT   *The Poisson-Tweedie family of distributions*

Description

Density function and random generation for the Poisson-Tweedie family of distributions.

Usage

```
dPT(x, mu, D, a, tol = 1e-15)
rPT(n, mu, D, a, max = 10*sqrt(mu*D), tol = 1e-4)
```

Arguments

x	an object of class 'mlePT' or a non-negative vector containing the integers in which the distribution should be evaluated.
mu	numeric positive scalar giving the mean of the distribution.
D	numeric positive scalar giving the dispersion of the distribution.
a	numeric scalar smaller than 1 giving the shape parameter of the distribution.
tol	numeric scalar giving the tolerance.
n	integer scalar giving number of random values to return.
max	numeric scalar containing the maximum number of counts to be used in the sampling process.

Value

If 'x' is of class 'mlePT', 'dPT' will return the Poisson-Tweedie distribution with parameters equal to the ones estimated by 'mlePoissonTweedie' evaluated on the data that was used to estimate the parameters. If 'x' is a numeric vector, 'dPT' will return the density of the specified Poisson-Tweedie distribution evaluated on 'x'.

'rPT' generates random deviates.

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

A.H. El-Shaarawi, R. Zhu, H. Joe (2010). Modelling species abundance using the Poisson-Tweedie family. Environmetrics 22, pages 152-164.

P. Hougaard, M.L. Ting Lee, and G.A. Whitmore (1997). Analysis of overdispersed count data by mixtures of poisson variables and poisson processes. Biometrics 53, pages 1225-1238.

See Also

compareCountDist testShapePT

Examples

```
# To compute the density function in 1:100 of the Polya-Aeppli
# distribution with mean = 20 and dispersion = 5
dPT(x = 1:100, mu = 20, D = 5, a = -1)

# To generate 100 random counts of the same distribution with same
# parameters
rPT(n = 100, mu = 20, D = 5, a = -1)
```

---

filterCounts   *Count data filtering*

Description

Filter count data to remove lowly expressed genes.

Usage

```
filterCounts(counts, cpm.cutoff=0.5, n.samples.cutoff=2)
```

Arguments

counts   numeric data.frame or matrix containing the count data.

cpm.cutoff   expression level cutoff defined as the minimum number of counts per million. By default this is set to 0.5 counts per million.

n.samples.cutoff
                 minimum number of samples where a gene should meet the counts per million cutoff (cpm.cutoff) in order to be kept as part of the count data matrix.

Details

This function removes genes with very low expression level defined in terms of a minimum number of coutns per million occurring in a minimum number of samples. Such a policy was described by Davis McCarthy in a message at the bioc-sig-sequencing mailing list. By default, this function keeps genes that are expressed at a level of 0.5 counts per million or greater in at least two samples.

Value

A matrix of filtered genes.

Author(s)

J.R. Gonzalez and R. Castelo

References

Davis McCarthy, https://stat.ethz.ch/pipermail/bioc-sig-sequencing/2011-June/002072.html.

See Also

normalizeCounts

Examples

```
# Generate a random matrix of counts
counts <- matrix(rPT(n=1000, a=0.5, mu=10, D=5), ncol = 40)

dim(counts)

# Filter genes with requiring the minimum expression level on every sample
filteredCounts <- filterCounts(counts, n.samples.cutoff=dim(counts)[2])

dim(filteredCounts)
```

---

gofTest   *Test the goodness of fit of every row in a matrix of counts*

Description

Function to test the goodness of fit of every row in a matrix of counts

Usage

```
gofTest(counts, a = 0, mc.cores = 1)
```

Arguments

counts	matrix of counts
a	numeric scalar smaller than 1. The function will test whether the shape parameter is equal to the introduced 'a' (default is 0).
mc.cores	number of cpu cores to be used. This option is only available when the 'multi-core' package is installed and loaded first. In such a case, if the default value of mc.cores=1 is not changed, all available cores will be used.

Details

By default a = 0, and therefore the function tests for every row of the input matrix of counts whether the count data follows a Negative-Binomial distribution. In this case, a Likelihood Ratio Test is performed. When the given value for 'a' is different from 0, a Wald test is performed. This function calls `testShapePT`.

Value

a vector of statistics that follows a χ^2 distribution with one degree of freedom under the null hypothesis.

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

A.H. El-Shaarawi, R. Zhu, H. Joe (2010). Modelling species abundance using the Poisson-Tweedie family. Environmetrics 22, pages 152-164.

P. Hougaard, M.L. Ting Lee, and G.A. Whitmore (1997). Analysis of overdispersed count data by mixtures of poisson variables and poisson processes. Biometrics 53, pages 1225-1238.

See Also

`testShapePT`

Examples

```
## Generate a random matrix of counts
counts <- matrix(rPT(n=2000, a=0.5, mu=10, D=5), nrow=20)

## Perform the goodness-of-fit tests for every row in the matrix
chi2gof <- gofTest(counts)

## Calculate and sort the corresponding P-values for the
## null hypothesis that counts follow a negative binomial distribution
sort(pchisq(chi2gof, df=1, lower.tail=FALSE))
```

---

mlePoissonTweedie   *Maximum likelihood estimation of the Poisson-Tweedie parameters*

Description

Maximum likelihood estimation of the Poisson-Tweedie parameters using L-BFGS-B quasi-Newton method.

Usage

```
mlePoissonTweedie(x, a, D.ini, a.ini, maxit = 100, loglik=TRUE, maxCount=20000,
getParam(object)
```

Arguments

x	numeric vector containing the read counts.
a	numeric scalar smaller than 1, if specified the PT shape parameter will be fixed.
D.ini	numeric positive scalar giving the initial value for the dispersion.
a.ini	numeric scalar smaller than 1 giving the initial value for the shape parameter (ignored if 'a' is specified).
maxit	numeric scalar providing the maximum number of 'L-BFGS-B' iterations to be performed (default is '100').
loglik	is log-likelihood computed? The default is TRUE
object	an object of class 'mlePT'.
maxCount	if max(x) > maxCount, then moment method is used to estimate model parameters to reduce computation time. The default is 20000.
...	additional arguments to be passed to the 'optim' 'control' options.

Details

The L-BFGS-B quasi-Newton method is used to calculate iteratively the maximum likelihood estimates of the three Poisson-Tweedie parameters. If 'a' argument is specified, this parameter will be fixed and the method will only estimate the other two.

Value

An object of class 'mlePT' containing the following information:

par: numeric vector giving the estimated mean ('mu'), dispersion ('D') and shape parameter 'a'.

se: numeric vector containing the standard errors of the estimated parameters 'mu', 'D' and 'a'.

loglik: numeric scalar providing the value of the loglikelihod for the estimated parameters.

iter: numeric scalar giving the number of performed iterations.

paramZhu: numeric vector giving the values of the estimated parameters in the Zhu parameterization 'a', 'b' and 'c'.

paramHou: numeric vector giving the values of the estimated parameters in the Hougaard parameterization 'alpha', 'delta' and 'theta'.

skewness: numeric scalar providing the estimate of the skewness given the estimated parameters.

x: numeric vector containing the count data introduced as the 'x' argument by the user.

convergence: A character string giving any additional information returned by the optimizer, or 'NULL'.

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

A.H. El-Shaarawi, R. Zhu, H. Joe (2010). Modelling species abundance using the Poisson-Tweedie family. Environmetrics 22, pages 152-164.

P. Hougaard, M.L. Ting Lee, and G.A. Whitmore (1997). Analysis of overdispersed count data by mixtures of poisson variables and poisson processes. Biometrics 53, pages 1225-1238.

See Also

testShapePT print.mlePT

Examples

```
# Generate 500 random counts following a Poisson Inverse Gaussian
# distribution with mean = 20 and dispersion = 5
randomCounts <- rPT(n = 500, mu = 20, D = 5, a = 0.5)

# Estimate all three parameters
res1 <- mlePoissonTweedie(x = randomCounts, a.ini = 0, D.ini
= 10)
res1
getParam(res1)

#Fix 'a = 0.5' and estimate the other two parameters
res2 <- mlePoissonTweedie(x = randomCounts, a = 0.5, D.ini
= 10)
res2
getParam(res2)
```

---

normalizeCounts *Count data normalization*

Description

Normalize count data to remove systematic technical effects.

Usage

```
normalizeCounts(counts, group, common.disp = FALSE, prior.n=1)
```

Arguments

counts	numeric data.frame or matrix containing the count data.
group	vector giving the experimental group/condition for each sample/library.
common.disp	logical indicating whether a common or tagwise (default) dispersions should be estimated and employed when adjusting counts.
prior.n	argument provided to the call of estimateTagwiseDisp which defines the amount of shrinkage of the estimated tagwise dispersions to the common one. By default prior.n=1 thus assumming no shrinkage toward that common dispersion. This argument is not used if common.disp=TRUE.

Details

This function aims at removing systematic technical effects from raw counts by using normalization procedures available in the edgeR package. In particular, the TMM method described in Robinson and Oshlack (2010) is employed to calculate normalization factors which are applied to estimate effective library sizes, then common and tagwise (only when the argument common.disp=TRUE) dispersions are calculated and finally counts are adjusted so that library sizes are approximately equal for the given dispersion values.

Value

A matrix of normalized counts.

Author(s)

J.R. Gonzalez and R. Castelo

References

M.D. Robinson and A. Oshlack. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biol*, 11:R25, 2010.

See Also

filterCounts

Examples

```
# Generate a random matrix of counts
counts <- matrix(rPT(n=1000, a=0.5, mu=10, D=5), ncol = 40)

colSums(counts)
counts[1:5, 1:5]

# Normalize counts
normCounts <- normalizeCounts(counts, rep(c(1,2), 20))

colSums(normCounts)
normCounts[1:5, 1:5]
```

Methods for objects of class 'mlePT'
Methods for objects of class 'mlePT'

Description

print, extract loglikelihood or compute confidence interval for an object of class 'mlePT'.

Usage

```
## S3 method for class 'mlePT'
print(x, dig = 3, ...)
## S3 method for class 'mlePT'
logLik(object, ...)
## S3 method for class 'mlePT'
confint(object, parm, level = 0.95, ...)
```

Arguments

x	object of class 'mlePT'.
object	object of class 'mlePT'.
dig	integer scalar giving the number of digits to be rounded the solution.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required (default is 0.95).
...	additional arguments.

Value

'logLik' returns the loglikelihood of the selected model.

'confint' returns a matrix (or vector) with columns giving lower and upper confidence limits for each parameter.

See Also

mlePoissonTweedie

Examples

```
# Load and aggregate the 'seizure' database
data(seizure)
aggCounts <- aggregate(x = cbind(seizure$count, seizure$trx), by =
list(seizure$id), FUN = sum)

# Estimate the parameters
mleSeizure <- mlePoissonTweedie(x = aggCounts[,2], a.ini = 0, D.ini =
10)

# Print
mleSeizure

# Extract loglikelihood
logLik(mleSeizure)

# Compute confidence inerval
confint(mleSeizure)
```

---

qqchisq	*Chi-square quantile-quantile plot*

Description

Make a chi-square quantile-quantile plot.

Usage

```
qqchisq(stat, df=1, normal=FALSE, rangeExpected=FALSE, obsQuantiles=c(0.50, 0.75
```

Arguments

stat   vector of χ^2 statistics.

df   degrees of freedom of stat.

normal   logical; set to TRUE if the χ^2 statistics in stat should be transform into normal z-scores in order to improve the display of lower quantiles. For this purpose, this function uses the [zscoreGamma](#) function from the limma package. Default is set to FALSE.

rangeExpected

      logical; set to TRUE if the displayed range of the observed χ^2 statistics is restricted to the range of their expected values. Default is set to FALSE.

obsQuantiles observed quantiles to indicate by horizontal dash lines. By default, these are set to 50%, 75% and 95%.

...   further arguments to pass to the [plot](#) function.

Details

The main purpose of this function in the tweeDEseq package is to provide means to assess the goodness of fit of count data to the negative binomial distribution. The main input argument stats should be the output of [gofTest](#).

Value

it returns invisibly a list with two components x and y corresponding to the coordinates of the plotted statistics.

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

See Also

[compareCountDist](#) [testShapePT](#)

Examples

```
## Generate a random matrix of counts
counts <- matrix(rPT(n=2000, a=0.5, mu=10, D=5), nrow=20)

## Perform the goodness-of-fit tests for every row in the matrix
chi2gof <- gofTest(counts)

## Not run:
qqchisq(chi2gof)

## End(Not run)
```

`seizure`	*Epileptic seizure counts*

Description

Data on seizure counts for 59 epileptics.

Usage

```
data(seizure)
```

Format

A data frame with 236 observations on the following 6 variables.

`id` a numeric vector, identification number for each patient

`count` a numeric vector, seizure counts

`visit` a numeric vector, visit number

`trx` a numeric vector, treatment: progabide (1) or placebo (0)

`baseline` a numeric vector, baseline 8 week seizure count

`age` a numeric vector, age of patient

Details

The data are from a placebo-controlled clinical trial of 59 epileptics. Patients with partial seizures were enrolled in a randomized clinical trial of the anti-epileptic drug, progabide. Participants in the study were randomized to either progabide or a placebo, as an adjuvant to the standard anti-epileptic chemotherapy. Progabide is an anti-epileptic drug whose primary mechanism of action is to enhance gamma-aminobutyric acid (GABA) content; GABA is the primary inhibitory neurotransmitter in the brain. Prior to receiving treatment, baseline data on the number of epileptic seizures during the preceding 8-week interval were recorded. Counts of epileptic seizures during 2-week intervals before each of four successive post-randomization clinic visits were recorded.

Source

P.F Thall, and S.C. Vail (1990). Some covariance models for longitudinal count data with overdispersion. Biometrics, 46, 657-671,

References

P. Hougaard, M.L. Ting Lee, and G.A. Whitmore (1997): Analysis of overdispersed count data by mixtures of poisson variables and poisson processes. Biometrics 53, pages 1225-1238.

See Also

testPoissonTweedie mlePoissonTweedie

Examples

```
# Although this is not a differential expression dataset, it is appropriate
# to illustrate the application of the Poisson-Tweedie in
# epidemiological studies

data(seizure)
summary(seizure)

# Aggregate
aggCounts <- aggregate(x = cbind(seizure$count, seizure$trx), by =
list(seizure$id), FUN = sum)

# Estimation of the three parameters for all individuals
mleSeizure <- mlePoissonTweedie(x = aggCounts[,2], a.ini = 0, D.ini = 10)
mleSeizure

#Poisson-Tweedie test
testPoissonTweedie(x = aggCounts[,2], group = aggCounts[,3])
```

---

testShapePT	*Test shape parameter of PT*

Description

Function to test whether the shape parameter is equal to a given value.

Usage

```
testShapePT(x, a = 0)
```

Arguments

x   object of class 'mlePT'.

a   numeric scalar smaller than 1. The function will test whether the shape param-
             eter is equal to the introduced 'a' (default is 0).

Details

By default a = 0, and therefore the function tests whether the count data follows a Negative-Binomial
distribution or not. In this case, a Likelihood Ratio Test is performed. When the given value for 'a'
is different from 0, a Wald test is performed.

If a = 1, the function tests whether the count data follows a Poisson distribution or not.

If a = 0.5, the function tests whether the count data follows a Poisson-inverse Gaussian distribution
or not.

If a = -1, the function tests whether the count data follows a Polya-Aeppli distribution or not.

Value

numeric p-value of the test.

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

A.H. El-Shaarawi, R. Zhu, H. Joe (2010). Modelling species abundance using the Poisson-Tweedie family. Environmetrics 22, pages 152-164.

P. Hougaard, M.L. Ting Lee, and G.A. Whitmore (1997). Analysis of overdispersed count data by mixtures of poisson variables and poisson processes. Biometrics 53, pages 1225-1238.

See Also

gofTest mlePoissonTweedie compareCountDist

Examples

```
# Generate a random matrix of counts
counts <- rPT(n=1000, a=0.5, mu=10, D=5)

# Maximum likelihood estimation of the Poisson-Tweedie parameters
mleEstimate <- mlePoissonTweedie(x = counts, a.ini = 0, D.ini
= 10)

# Test whether data comes from Negative-Binomial distribution
testShapePT(mleEstimate)

# Test whether data comes from Poisson-inverse Gaussian
testShapePT(mleEstimate, a = 0.5)
```

---

tweeDE *Score test for differences between two Poisson-Tweedie groups*

Description

Carry out a score test for differences between two Poisson-Tweedie groups.

Usage

```
tweeDE(object, group, mc.cores = 1, pair = NULL, ...)
testPoissonTweedie(x, group, saveModel = FALSE, ...)
MAplot(x, ...)
Vplot(x, ...)
## S3 method for class 'tweeDE'
print(x, n=6L, sort.by="pval", log2fc.cutoff=0, pval.adjust.cutoff=1, print=TRUE
## S3 method for class 'tweeDE'
MAplot(x, log2fc.cutoff=0, highlight=NULL, ...)
## S3 method for class 'tweeDE'
Vplot(x, log2fc.cutoff=0, pval.adjust.cutoff=1, highlight=NULL, ylab=expression(
```

Arguments

`object`	a `data.frame` or a `matrix` of RNA-seq counts.
`group`	vector giving the experimental group/condition for each sample/library.
`mc.cores`	number of cpu cores to be used. This option is only available when the 'multi-core' package is installed and loaded first. In such a case, if the default value of `mc.cores=1` is not changed, all available cores will be used.
`pair`	vector of two elements containing the representants of each of the two groups (default is 'NULL').
`n`	maximum number of genes printed.
`sort.by`	character string, indicating whether genes should be ranked by their P-value (`pval`), which is the default setting, or by absolute log2 fold-change (`log2fc`).
`log2fc.cutoff`	
	cutoff on the minimum value of the log2 fold change.
`pval.adjust.cutoff`	
	cutoff on the maximum adjusted P-value (FDR).
`print`	logical; it indicates whether the output should be printed on the terminal.
`highlight`	list of arguments to the `points()` plotting function in order to highlight genes in the MA or volcano plots. A component called `genes` is expected to have the identifiers of the genes to be higlighted.
`ylab`	label on the y-axis of the volcano plot set by default to -log10 of the raw P-value which is what this plot displays on that axis.
`x`	object returned by the function `tweeDE` in the case of `print` and vector of count data in the case of `testPoissonTweedie`.
`saveModel`	logical indicating whether the results of fitting the model should be saved or not (default is 'FALSE').
`...`	additional arguments.

Details

'testPoissonTweedie' performs the test for a vector of counts.

'tweeDE' performs the test for a whole 'data.frame'. The P-values are then corrected using the Benjamini and Hochberg method.

Value

'testPoissonTweedie' returns a list with:

'mean': means for each group 'pvalue': p-value for the test

'tweeDE' returns a 'data.frame' with columns

'overallMean': overall mean counts 'meanA': mean counts of the first group 'meanB': mean counts of the second group 'log2fc': logarigthm (base 2) of the fold-change (second group vs. first group) 'pval': p-value for the test 'pval.adjust': adjusted p-value using Benjamini-Hochberg method

References

M. Esnaola, P. Puig, D. Gonzalez, R. Castelo, J.R. Gonzalez. A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. Submitted.

A.H. El-Shaarawi, R. Zhu, H. Joe (2010). Modelling species abundance using the Poisson-Tweedie family. Environmetrics 22, pages 152-164.

P. Hougaard, M.L. Ting Lee, and G.A. Whitmore (1997). Analysis of overdispersed count data by mixtures of poisson variables and poisson processes. Biometrics 53, pages 1225-1238.

See Also

normalizeCounts mlePoissonTweedie

Examples

```
# Generate a random matrix of counts
counts <- matrix(rPT(n = 1000, a = 0.5, mu = 10, D = 5), ncol = 40)

# Test for differences between the two groups
tweeDE(counts, group = rep(c(1,2),20))
```

Index