Package ‘Polyfit’

November 21, 2016

Type Package
Title Add-on to DESeq to improve p-values and q-values
Version 1.8.0
Date 2014-08-06
Author Conrad Burden
biocViews DifferentialExpression, Sequencing, RNASeq, GeneExpression
Maintainer Conrad Burden <conrad.burden@anu.edu.au>
Depends DESeq
Suggests BiocStyle
Description Polyfit is an add-on to the packages DESeq which ensures the p-value distribution is uniform over the interval [0, 1] for data satisfying the null hypothesis of no differential expression, and uses an adapted Storey-Tibshiran method to calculate q-values.
License GPL (>= 3)
NeedsCompilation no

R topics documented:

Polyfit-package .................................................. 1
levelPValues .................................................. 3
pfNbinomTest .................................................. 4
twoSidedPValueFromDiscrete ................................. 5

Index

Polyfit-package Polyfit add-on to DESeq

Description

implementation the Polyfit add-on to DESeq described in the paper "Improved error estimates for the analysis of differential expression from RNA-seq data"
Polyfit-package

Details

Package: Polyfit
Type: Package
Version: 0.99.3
Date: 2014-08-06
License: GPL(>=3)

Polyfit is an add-on to the negative-binomial based packages DESeq for two-class detection of differential expression which ensures the p-value distribution is uniform over the interval [0, 1] for data satisfying the null hypothesis of no differential expression. The first component is the function `pfNbinomTest` which replaces the function `nbinomTest` in DESeq. Its purpose is to smooth point singularities, particularly one at \( p = 1 \), in the p-value distribution caused by calculating calculating p-values from a discrete distribution. The output from this function should then be passed to the second component, the function `link{levelPValues}`. Its purpose is to apply a variant of the Storey-Tibshirani procedure to shift the p-values so that those corresponding to the null hypothesis have a uniform distribution, and to calculate corresponding q-values (or 'adjusted p-values') for controlling errors via the false discovery rate.

Author(s)

Conrad Burden

Maintainer: conrad.burden@anu.edu.au

References


Examples

```R
# Example using DESeq
cds <- makeExampleCountDataSet()
cds <- estimateSizeFactors( cds )
cds <- estimateDispersions( cds )
nbTPolyfit <- pfNbinomTest( cds, "A", "B" )

lp <- levelPValues(nbTPolyfit$pval)
pvalTab <- cbind( origPval=nbTPolyfit$pval, correctedPval=lp$pValueCorr, qval=lp$qValueCorr )
cat("n Original and corrected P-values from DESeq \n")
head(pvalTab)
```
**levelPValues**

**Level P-values**

**Description**

Function to level out a P-value spectrum generated by the Polyfit extension of DESeq by fitting a quadratic function to the right hand portion of the spectrum, produce ‘corrected’ p-values and q-values using an adapted version of the Storey-Tibsharini procedure

**Usage**

```r
levelPValues(oldPvals, plot = FALSE)
```

**Arguments**

- `oldPvals`: an array of p-values produced by the Polyfit replacement of the DESeq function `pfNbinomTest()` or the Plyfit replacement of the edgeR function `pfExactTest()`
- `plot`: TRUE to plot original and corrected pvalue spectra; FALSE not to plot

**Details**

`levelPValues` should only be used with P-values generated by the Polyfit function `pfNbinomTest`, and not with P-values generated by `nbinomTest`.

**Value**

List containing

- `pi0estimate`: an estimate of the proportion of genes not differentially expressed
- `lambdaOptimal`: the point in the p-value spectrum past which a quadratic is fitted
- `pValueCorr`: p-values calculated from the levelled spectrum
- `qValueCorr`: q-values calculated from the levelled spectrum
- `qValueCorrBH`: q-values calculated from `pValueCorr` using Benjamini-Hochberg

**Author(s)**

Conrad Burden

**References**


**Examples**

```r
cds <- makeExampleCountDataSet()
cds <- estimateSizeFactors( cds )
cds <- estimateDispersions( cds )
nbTPolyfit <- pfNbinomTest( cds, "A", "B" )
1P <- levelPValues(nbTPolyfit$pval, plot=TRUE)
pvalTab <- cbind(origPval=nbTPolyfit$pval, correctedPval=1P$pValueCorr)
cat("\n Original and corrected P-values from DESeq \n")
head(pvalTab)
```
pfNbinomTest

The Polyfit extension to the DESeq functions nbinomTest() and nbinomTestForMatrices()

Description

Polyfit extensions to the DESeq functions nbinomTest and nbinomTestForMatrices which test for differences between the base means of two conditions (i.e., for differential expression in the case of RNA-Seq).

Usage

pfNbinomTest(cds, condA, condB, pvals_only = FALSE, eps = NULL)
pfNbinomTestForMatrices(countsA, countsB, sizeFactorsA, sizeFactorsB, dispsA, dispsB)

Arguments

cds | a CountDataSet with size factors and raw variance functions
condA | one of the conditions in 'cds'
condB | another one of the conditions in 'cds'
pvals_only | return only a vector of (unadjusted) p values instead of the data frame described below
eps | This argument is no longer used. Do not use it
countsA | A matrix of counts, where each column is a replicate
countsB | Another matrix of counts, where each column is a replicate
sizeFactorsA | Size factors for the columns of the matrix 'countsA'
sizeFactorsB | Size factors for the columns of the matrix 'countsB'
dispsA | The dispersions for 'countsA', a vector with one value per gene
dispsB | The same for 'countsB'

Details

These functions have the same behaviour as the DESeq functions nbinomTest and nbinomTestForMatrices, except that the ‘flagpole’ in the P-value histogram, particularly at p = 1 is redistributed using the function twoSidedPValueFromDiscrete.

Value

pfNbinomTest gives a data frame with the following columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The ID of the observable, taken from the row names of the counts slots.</td>
</tr>
<tr>
<td>baseMean</td>
<td>The base mean (i.e., mean of the counts divided by the size factors) for the counts for both conditions</td>
</tr>
<tr>
<td>baseMeanA</td>
<td>The base mean (i.e., mean of the counts divided by the size factors) for the counts for condition A</td>
</tr>
<tr>
<td>baseMeanB</td>
<td>The base mean for condition B</td>
</tr>
<tr>
<td>foldChange</td>
<td>The ratio meanB/meanA</td>
</tr>
</tbody>
</table>
log2FoldChange  The log2 of the fold change
pval           The p value for rejecting the null hypothesis ‘meanA==meanB’
padj           The adjusted p values (adjusted with ‘p.adjust( pval, method="BH")’)

pfNbinomTestForMatrices gives a vector of unadjusted p values, one for each row in the counts matrices.

Author(s)

Conrad Burden, conrad.burden@anu.edu.au, based on software by Simon Anders

References


Examples

cds <- makeExampleCountDataSet()
cds <- estimateSizeFactors( cds )
cds <- estimateDispersions( cds )
nbT <- nbinomTest( cds, "A", "B" )
head( nbT )
nbTPolyfit <- pfNbinomTest( cds, "A", "B" )
head( nbTPolyfit )
oldpar <- par(mfrow=c(1,2))
hist(nbT$pval,breaks=seq(0,1,by=0.01),
xlab="P-value", main="DESeq")
hist(nbTPolyfit$pval,breaks=seq(0,1,by=0.01),
xlab="P-value", main="polyfit-DESeq")
par(oldpar)

twoSidedPValueFromDiscrete

Two sided P-value from discrete distribution

Description

Function to calculate a 2-sided p-value of an observation xobs for a finite discrete distribution

\[ \text{Prob}(X = xobs) = \text{probs}[xobs + 1] \]

over the range \( xobs \) in \((0, 1, \ldots, xmax)\) by “squaring off” the distribution to a continuous distribution

Usage

twoSidedPValueFromDiscrete(probs, xobs)
twoSidedPValueFromDiscrete

Arguments

- **probs**: an array containing the probabilities that $X$ takes the values 0, 1, ..., $x_{max}$
- **xobs**: a single observed value of $X$

Details

Note that the returned 2-sided p-value contains a random component, i.e. a given set of input parameters returns a different result each run.

Value

A real valued randomised p-value between 0 and 1. If $x_{obs}$ is generated with probability $probs[x_{obs} + 1]$ the returned value will be uniformly distributed on the interval [0, 1].

Author(s)

Conrad Burden

Examples

```r
pr <- dbinom(0:5, size=5, prob=0.4)
xSample <- rbinom(10000, size=5, prob=0.4)
pvalues <- c()
for(x in xSample){
  pvalues <- c(pvalues, twoSidedPValueFromDiscrete(pr, x))
}
hist(pvalues)
```
Index

*Topic \textasciitilde kwd1
  \twoSidedPValueFromDiscrete, 5
*Topic \textasciitilde kwd2
  \twoSidedPValueFromDiscrete, 5
*Topic \textbf{package}
  Polyfit\textendash package, 1

levelPValues, 3

nbinomTest, 2–4
nbinomTestForMatrices, 4

\pfnbinomTest, 2, 3, 4
\pfnbinomTestForMatrices (\pfnbinomTest), 4
Polyfit (Polyfit\textendash package), 1
Polyfit\textendash package, 1

twoSidedPValueFromDiscrete, 4, 5