Package 'DiPhiSeq'

July 21, 2025

Type Package

Title Robust Tests for Differential Dispersion and Differential Expression in RNA-Sequencing Data

Version 0.2.0

Description Implements the algorithm described in Jun Li and Alicia T. Lamere, ``DiPhiSeq: Robust comparison of expression levels on RNA-Seq data with large sample sizes" (Unpublished). Detects not only genes that show different average expressions (``differential expression", DE), but also genes that show different diversities of expressions in different groups (``differentially dispersed", DD). DD genes can be important clinical markers. 'DiPhiSeq' uses a redescending penalty on the quasi-likelihood function, and thus has superior robustness against outliers and other noise. Updates from version 0.1.0: (1) Added the option of using adaptive initial value for phi. (2) Added a function for estimating the proportion of outliers in the data. (3) Modified the input parameter names for clarity, and modified the output format for the main function.

Depends R (>= 3.1.0)

Imports stats (>= 3.1.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Jun Li [aut, cre], Alicia T. Lamere [aut]

Maintainer Jun Li <jun.li@nd.edu>

Repository CRAN

Date/Publication 2018-10-24 22:40:07 UTC

Contents

`diphiseq` *Main function. For most users, this function is all what they need for the analysis.*

Description

Main function. For most users, this function is all what they need for the analysis.

Usage

```
diphiseq(countmat, classlab, depth = NULL, c.tukey.beta = 4,
  c.tukey.phi = 4, phi.ini = "adaptive")
```

Arguments

`countmat`	A count matrix. Rows are genes, and columns are samples. Each element/count is the number of reads mapped to a gene in a sample.
`classlab`	The class labels. A vector whose elements are of value 1 or 2.
`depth`	Sequencing depth. A vector of sequencing depth. Users are encouraged to provide estimated values from edgeR, DESeq, PoissonSeq, or other software that they prefer. If no values are provided, depth estimated by total counts will be used.
`c.tukey.beta`	The c value for beta in Tukey's biweight function. The default value, 4, is typically regarded as appropriate and should work for most datasets.
`c.tukey.phi`	The c value for phi in Tukey's biweight function. The default value, 4, is typically regarded as appropriate and should work for most datasets.
`phi.ini`	The initial value for phi to start search. If phi.in == 'adaptive' (the default value), the algorithm will adaptively choose a value (check Algorithm 1 for details). Otherwise, a positive numeric value (such as 0.5) should be given.

Value

A List that contains the following elements: `tab`: This is a data frame that contains the main results of this package. It has the following columns: `phi1`: the estimated dispersion of sample group 1. `phi2`: the estimated dispersion of sample group 2. `beta1`: the estimated (log) expression of sample group 1. `beta2`: the estimated (log) expression of sample group 2. `statistic.phi`: the z statistic for DD. `statistic.beta`: the z statistic for DE. `p.value.phi`: the p value for DD. `p.value.beta`:

the p value for DE. `fdr.phi`: the FDR for DD. `fdr.beta`: the FDR for DE. `log.depth`: A vector of the log sequencing depths. `countmat`: This is the count matrix that the user provided. `classlab`: This is the vecgtor of class labels that the user provided. `phi.ini`: The initial searching value of the dispersion parameter. `mumat`: This is the (estimated) mu (expected expression) matrix, of the same size as countmat. In another word, E(countmat)=mumat. `phimat`: This is the (estimated) phi matrix, of the same size as countmat. In another word, counmat ~ negative binomial(mumat, phimat).

Examples

```
countmat <- matrix(rnbinom(100, size=1, mu=50), nrow=4, ncol=25)
classlab <- c(rep(1, 10), rep(2, 15))
res <- diphiseq(countmat, classlab)

countmat <- matrix(rnbinom(100, size=1, mu=50), nrow=4, ncol=25)
classlab <- c(rep(1, 10), rep(2, 15))
res <- diphiseq(countmat, classlab, phi.ini=0.5)
```

---

example_data *Numeric count matrix of example data.*

Description

The first several rows of a dataset from the Cancer Genome Atlas Kidney Renal Clear Cell Carcinoma data collection, consisting of 5 genes from a dataset of raw RNA-Seq counts from 35 tumor and 35 normal samples.

Usage

```
example_data
```

Format

A numeric matrix with five genes observed across 70 experiments. The first 35 are tumor samples, the second 35 are normal samples.

Source

Akin, O. and Elnajjar, P. and Heller, M. and Jarosz, R. and Erickson, B. J. and Kirk, S. and ... Filippini, J. (2016) Data from The Cancer Genome Atlas Kidney Renal Clear Cell Carcinoma (TCGA-KIRC) collection. The Cancer Imaging Archive.

outprop *Give a rough estimate of the proportion of outliers in the data based on the results of DiPhiSeq.*

Description

Give a rough estimate of the proportion of outliers in the data based on the results of DiPhiSeq.

Usage

```
outprop(diphiseq.res, fdr.cutoff = 0.1)
```

Arguments

diphiseq.res	The results given by running diphiseq.
fdr.cutoff	The cutoff for FDR.

Value

a numeric value. The estimated proportion of outliers under the FDR cutoff in the data.

Examples

```
countmat <- matrix(rnbinom(100, size=1, mu=50), nrow=4, ncol=25)
classlab <- c(rep(1, 10), rep(2, 15))
res <- diphiseq(countmat, classlab)
outlier.proportion <- outprop(res)
```

robnb *Calculates the estimate and standard error of beta and phi. It takes as input counts from one group of samples for a single gene. This function is the core underlining function of the whole package. A significant part of the code is edited based on William H. Aeberhard's glmrob.nb R function; we appreciate them very much for sharing their code online. This function also implement Algorithm 1 of our submitted paper about DiPhiSeq. This function is called by robtest. Most users don't need to call this function directly.*

Description

Calculates the estimate and standard error of beta and phi. It takes as input counts from one group of samples for a single gene. This function is the core underlining function of the whole package. A significant part of the code is edited based on William H. Aeberhard's glmrob.nb R function; we appreciate them very much for sharing their code online. This function also implement Algorithm 1 of our submitted paper about DiPhiSeq. This function is called by robtest. Most users don't need to call this function directly.

Usage

```
robnb(y, log.depth, c.tukey.beta = 4, c.tukey.phi = 4, phi.ini = 0.5,
  alpha = 0.2, minphi = 0.01, maxphi = 5, maxit = 30, maxit.beta = 30,
  maxit.phi = 30, tol.beta = 0.01, tol.phi = 0.005)
```

Arguments

y	A count vector.
log.depth	Vector of log(sequencing depths).
c.tukey.beta	The c value for beta in Huber function. The default value should be appropriate for most datasets.
c.tukey.phi	The c value for phi in Huber function. The default value should be appropriate for most datasets.
phi.ini	The initial value of phi.
alpha	A positive value for setting initial values. The default value is usually appropriate.
minphi	A searching parameter for Algorithm 1 (check the algorithm for details.) The default value is usually appropriate.
maxphi	A searching parameter for Algorithm 1 (check the algorithm for details.) The default value is usually appropriate.
maxit	Maximum number of iterations for the outer loop. The default value is usually appropriate.
maxit.beta	Maximum number of iterations for the inner loop of solving beta. The default value is usually appropriate.
maxit.phi	Maximum number of iterations for the inner loop of solving phi. The default value is usually appropriate.
tol.beta	The numerical tolerance of solving beta. The default value is usually appropriate.
tol.phi	The numerical tolerance of solving phi. The default value is usually appropriate.

Value

A list that contains the elements: beta: the estimated (log) expression. phi: the estimated dispersion. fconv: flag of the convergence of the search. vars: the variance-covariance matrix of the estimates. sd.beta: the standard error of beta. sd.phi: the standard error of phi. y: the input y value. log.depth: log(sequencing depth).

Examples

```
d <- runif(10, min=1, max=2)
y <- rnbinom(10, size=1, mu=d*50)
res <- robnb(y, log(d))
```

robtest — *Calls the robnb function to estimate the coefficients, and then construct the statistical tests for DD and DE. It works for a single gene. y1 and y2 are count vectors for a single gene. diphiseq calls this function to do the calculation for each gene. Normal users often don't need to use this function directly.*

Description

Calls the robnb function to estimate the coefficients, and then construct the statistical tests for DD and DE. It works for a single gene. y1 and y2 are count vectors for a single gene. diphiseq calls this function to do the calculation for each gene. Normal users often don't need to use this function directly.

Usage

```
robtest(y1, log.depth1, y2, log.depth2, c.tukey.beta = 4, c.tukey.phi = 4)
```

Arguments

y1	counts from group 1. a vector.
log.depth1	log(sequencing depths) for samples in group 1. a vector.
y2	counts from group 2. a vector.
log.depth2	log(sequencing depths) for samples in group 2. a vector.
c.tukey.beta	The c value for beta in Huber function. The default value, 4, is typically regarded as appropriate and should work for most datasets.
c.tukey.phi	The c value for phi in Huber function. The default value, 4, is typically regarded as appropriate and should work for most datasets.

Value

A vector that contains the elements: phi1: the estimated dispersion of sample group 1. phi2: the estimated dispersion of sample group 2. beta1: the estimated (log) expression of sample group 1. beta2: the estimated (log) expression of sample group 2. statistic.phi: the z statistic for DD. statistic.beta: the z statistic for DE. p.value.phi: the p value for DD. p.value.beta: the p value for DE.

Examples

```
d1 <- runif(10, min=1, max=2)
d2 <- runif(15, min=1, max=2)
y1 <- rnbinom(10, size=1, mu=d1*50)
y2 <- rnbinom(15, size=1, mu=d2*50)
res <- robtest(y1, log(d1), y2, log(d2))
```

Index