Package ‘NanoStringDiff’

February 1, 2017

Type Package
Title Differential Expression Analysis of NanoString nCounter Data
Version 1.5.0
Date 2015-08-3
Imports matrixStats, methods
Depends Biobase
Suggests testthat, BiocStyle
Author hong wang <hong.wang@uky.edu>, chi wang <chi.wang@uky.edu>
Maintainer hong wang <hong.wang@uky.edu>
Description This Package utilizes a generalized linear model(GLM) of
  the negative binomial family to characterize count data and
  allows for multi-factor design. NanoStrongDiff incorporate size
  factors, calculated from positive controls and housekeeping
  controls, and background level, obtained from negative
  controls, in the model framework so that all the normalization
  information provided by NanoString nCounter Analyzer is fully
  utilized.
License GPL
biocViews DifferentialExpression, Normalization
NeedsCompilation no

R topics documented:

NanoStringDiff-package .................................................. 2
estNormalizationFactors ............................................... 2
glm.LRT ................................................................. 3
housekeepingControl .................................................. 4
housekeepingFactor .................................................... 5
NanoStringData .......................................................... 6
NanoStringSet-class ..................................................... 6
negativeControl ........................................................ 8
negativeFactor .......................................................... 9
positiveControl ........................................................ 10
positiveFactor .......................................................... 11

Index 12
NanoStringDiff-package

NanoStringDiff package for differential expression analysis of NanoString nCounter data

Description

NanoStringDiff is an R package for differential expression analysis of NanoString nCounter data, and the main function for differential analysis is `glm.LRT`. See the examples at `glm.LRT` for basic analysis steps. NanoStringDiff utilizes a generalized linear model (GLM) of the negative binomial family to characterize count data and allows for multi-factor design.

Author(s)

hong Wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

estNormalizationFactors

estimate normalization factors, include positive size factors, background noise, housekeeping size factors.

Description

This function estimates positive size factors, background noise and housekeeping size factors for the input "NanoStringSet" object and return the same object with positiveFactor, negativeFactor and housekeepingFactor slots filled or replaced.

Usage

`estNormalizationFactors(NanoStringData)`

Arguments

NanoStringData An object of "NanoStringSet" class.

Value

The same "NanoStringSet" object with positiveFactor, negativeFactor and housekeepingFactor field filled or replaced.

Author(s)

hong wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

Examples

data(NanoStringData)
NanoStringData=estNormalizationFactors(NanoStringData)
pf=positiveFactor(NanoStringData)
nf=negativeFactor(NanoStringData)
hf=housekeepingFactor(NanoStringData)
glm.LRT

perform gene-wise likelihood ratio test for NanoString Data

Description

The method considers a generalized linear model of the negative binomial family to characterize count data and allows for multi-factor design. The method propose an empirical Bayes shrinkage approach to estimate the dispersion parameter and use likelihood ratio test to obtain p-value.

Usage

```r
glm.LRT(NanoStringData, design.full, Beta=ncol(design.full), contrast=NULL)
```

Arguments

- **NanoStringData**: An object of "NanoStringSet" class.
- **design.full**: numeric matrix giving the design matrix for the generalized linear models under full model. must be of full column rank.
- **Beta**: integer or character vector indicating which coefficients of the linear model are to be tested equal to zero. Values must be columns or column names of design. Defaults to the last coefficient. Ignored if contrast is specified.
- **contrast**: numeric vector or matrix specifying one or more contrasts of the linear model coefficients to be tested equal to zero.

Value

A list of:
- **table**: A data frame with each row corresponding to a gene. Rows are sorted according to likelihood ratio test statistics. The columns are: logFC: log fold change between two groups. lr: likelihood ratio test statistics. pvalue: p-value. qvalue: adjust p-value using the procedure of Benjamini and Hochberg.
- **dispersion**: a vetor of dispersion
- **log.dispersion**: a vector of log dispersion: log.dispersion=log(dispersion)
- **design.full**: numeric matrix giving the design matrix under full generalized linear model.
- **design.reduce**: numeric matrix giving the design matrix under reduced generalized linear model.
- **Beta.full**: coefficients under full model.
- **mean.full**: mean value under full model.
- **Beta.reduce**: coefficients under reduced model.
- **mean.reduce**: mean value under reduced model.
- **m0**: hyper-parameter: mean value of the prior distribution of log dispersion
- **sigma**: hyper-parameter: standard deviation of the prior distribution of log dispersion

Author(s)

hong wang<hong.wang@uky.edu> chi wang <chi.wang@uky.edu>
Examples

```r
data(NanoStringData)
NanoStringData=estNormalizationFactors(NanoStringData)
group=pData(NanoStringData)
design.full=model.matrix(~0+factor(group$group))
contrast=c(1,-1)
result=glm.LRT(NanoStringData,design.full,
    Beta=ncol(design.full),contrast=contrast)
head(result$table)
```

Description

user-defined housekeeping control genes can be used to estimate housekeeping factors to adjust variation caused by different sample input.

Usage

```r
## S4 method for signature 'NanoStringSet'
housekeepingControl(object)
## S4 replacement method for signature 'NanoStringSet,matrix'
housekeepingControl(object) <- value
```

Arguments

- `object`: A NanoStringSet object.
- `value`: A matrix with housekeeping control genes.

Details

NanoString nCounter analyzer also contains probes for a set of species-specific mRNA housekeeping (reference) genes that are not spike-in the system. Nanostring recommends at least three housekeeping genes, but the more that are included, the more accurate the normalization will be. Housekeeping control genes are expected consistent in their expression levels.

Value

A matrix contain housekeeping control genes

Author(s)

Hong Wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

See Also

housekeepingFactor
Examples

```r
data(NanoStringData)
## obtain housekeeping control genes
housekeepingControl(NanoStringData)

## assign a matrix
n=ncol(exprs(NanoStringData))
r=nrow(housekeepingControl(NanoStringData))
housekeeping=matrix(rpois(r*n,100),ncol=n)
housekeepingControl(NanoStringData)=housekeeping
```

Description

Housekeeping size factors can be used to adjust the variance caused by different sample input.

Usage

```r
## S4 method for signature 'NanoStringSet'
housekeepingFactor(object)
## S4 replacement method for signature 'NanoStringSet,numeric'
housekeepingFactor(object) <- value
```

Arguments

- `object`: A NanoStringSet object.
- `value`: A vector of housekeeping size factors.

Details

Housekeeping gene normalization corrects for different in sample input between assays, since reference genes are supposed to have the same expression rate between samples. So the read counts from housekeeping genes, after subtracting background noise and adjusting by positive size factors, are not expected to vary between samples. If there exist difference, which should be caused by sample input variation.

Value

A vector contain housekeeping factors

Author(s)

Hong Wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

See Also

housekeepingControl
Examples

data(NanoStringData)
## obtain housekeeping factors
housekeepingFactor(NanoStringData)

## assign a vector
n=ncol(exprs(NanoStringData))
housekeepingFactor(NanoStringData)=rep(1,n)

NanoStringData  A real 'NanoStringSet' object.

Description

The object is created based on Mori Data with normal and tumor groups and 2 samples in each group. The object contain 599 endogenes, 6 positive control, 6 negative control and 4 housekeeping control.

Usage

data(NanoStringData)

Value

An object of NanoStringSet

Examples

data(NanoStringData)
NanoStringData

NanoStringSet-class  NanoStringSet object and constructors

Description

The NanoStringSet is a s4 class used to store data from NanoString nCounter analyzer. This class a subclass of ExpressionSet, with six more slots: positiveControl, negativeControl, housekeepingControl, positiveFactor, negativeFactor and housekeepingFactor. The constructor functions createNanoStringSet and createNanoStringSetFromCsv create a NanoStringSet object from two types of input: separate matrix or csv files. See the vignette for examples of construction from these two input types.

Usage

createNanoStringSet(endogenous,positiveControl,negativeControl,
    housekeepingControl,designs)

createNanoStringSetFromCsv(path, header=TRUE, designs)
**Arguments**

- **endogenous**
  - for matrix input: a matrix of non-negative integers of endogenes

- **positiveControl**
  - for matrix input: a matrix of non-negative integers of positive control genes.
  - There must have 6 positive control genes order by concentrations form high to low

- **negativeControl**
  - for matrix input: a matrix of non-negative integers of negative control genes

- **housekeepingControl**
  - for matrix input: a matrix of non-negative integers of housekeeping control genes

- **designs**
  - for data.frame input: phenotype data for NanoString nCounter data with at least one column. Each row is one sample, that is the number of rows must equal number of samples or replicates in the data.

- **path**
  - path to the csv file.

- **header**
  - a logical value indicating whether the file contains the names of the variables as its first line. The default value is TRUE.

**Value**

A NanoStringSet object.

**Methods**

- **positiveControl, positiveControl<-**
  - Access and set positive control genes.

- **negativeControl, negativeControl<-**
  - Access and set negative control genes.

- **housekeepingControl, housekeepingControl<-**
  - Access and set housekeeping control genes.

- **positiveFactor, positiveFactor<-**
  - Access and set positive factors.

- **negativeFactor, negativeFactor<-**
  - Access and set negative factors.

- **housekeepingFactor, housekeepingFactor<-**
  - Access and set housekeeping factors.

**Author(s)**

hong wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

**See Also**

- positiveControl, negativeControl, housekeepingControl, positiveFactor, negativeFactor, housekeepingFactor

**Examples**

```r
endogenous = matrix(rpois(100, 50), 25, 4)
positive = matrix(rpois(24, c(128, 32, 8, 2, 0.5, 0.125) * 80), 6, 4)
negative = matrix(rpois(32, 10), 8, 4)
housekeeping = matrix(rpois(12, 100), 3, 4)
designs = data.frame(group = c(0, 0, 1, 1), gender = c("male", "female", "female", "male"), age = c(20, 40, 39, 37))
NanoStringData = createNanoStringSet(endogenous, positive, negative, housekeeping, designs)
NanoStringData
```

NanoStringData
negativeControl

Accessor functions for the 'negativeControl' slot in a NanoStringSet object.

Description

Negative control genes are provided by nCounter Analyzer which can be used to estimate background noise for each sample.

Usage

```r
## S4 method for signature 'NanoStringSet'
negativeControl(object)
## S4 replacement method for signature 'NanoStringSet,matrix'
negativeControl(object) <- value
```

Arguments

- `object` A NanoStringSet object.
- `value` A matrix with negative control genes.

Details

Each code set in the nCounter Analyzer includes several negatives control genes for which no transcript is expected to be present. We use these spike-in negative control genes to estimate background noise for each sample.

Value

A matrix contain negative control genes

Author(s)

Hong Wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

See Also

negativeFactor

Examples

```r
data(NanoStringData)
## obtain negative control genes
negativeControl(NanoStringData)
## assign a matrix
n=ncol(exprs(NanoStringData))
r=nrow(negativeControl(NanoStringData))
```
negativeFactor

```r
negative = matrix(rpois(r*n, 10), ncol = n)
negativeControl(NanoStringData) = negative
```

### negativeFactor

**Accessor functions for the 'negativeFactor' slot in a NanoStringSet object.**

**Description**

Negative size factors can be used to adjust background noise for each sample.

**Usage**

```r
## S4 method for signature 'NanoStringSet'
negativeFactor(object)

## S4 replacement method for signature 'NanoStringSet, numeric'
negativeFactor(object) <- value
```

**Arguments**

- `object`: A NanoStringSet object.
- `value`: A vector of background noise.

**Details**

Accurate estimation of system background is essential for DE detection analysis. Each code set in the nCounter Analyzer includes several negative control genes for which no transcript is expected to be present. We use these spike-in negative control genes to estimate background noise for each sample.

**Value**

A vector containing background noise.

**Author(s)**

Hong Wang <hong.wang@uky.edu>; Chi Wang <chi.wang@uky.edu>

**See Also**

negativeControl

**Examples**

```r
data(NanoStringData)
## obtain negative factors
negativeFactor(NanoStringData)

## assign a vector
n = ncol(exprs(NanoStringData))
lambda = rpois(n, 10)
negativeFactor(NanoStringData) = lambda
```
positiveControl

Acessor functions for the 'positiveControl' slot in a NanoStringSet object.

Description

nCounter Analyzer has positive spike-in RNA hybridization controls for each sample which can be used to estimate the overall efficiency of hybridization and recovery for each sample.

Usage

## S4 method for signature 'NanoStringSet'
positiveControl(object)
## S4 replacement method for signature 'NanoStringSet,matrix'
positiveControl(object) <- value

Arguments

- object: A NanoStringSet object.
- value: A matrix with six positive control genes.

Details

Positive control genes are provided by NanoString nCounter technology. For each sample, nCounter provide six positive controls corresponding to six different concentrations in the 30 ul hybridization: 128fM, 32fM, 8fM, 2fM, 0.5fM, and 0.125fM. Six positive control genes must be order by concentrations from high to low.

Value

A matrix contain positive control genes

Author(s)

Hong Wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

See Also

positiveFactor

Examples

data(NanoStringData)
## obtain positive control genes
positiveControl(NanoStringData)

## assign a matrix
n=ncol(exprs(NanoStringData))
x=matrix(c(128,32,8,2,0.5,0.125)*80,ncol=1)
positive=matrix(rpois(6*n,x),ncol=n)
positiveControl(NanoStringData)=positive
positiveFactor

Description
Positive size factors can be used to adjust all platform associated sources of variation.

Usage
## S4 method for signature 'NanoStringSet'
positiveFactor(object)
## S4 replacement method for signature 'NanoStringSet,numeric'
positiveFactor(object) <- value

Arguments

object       A NanoStringSet object.
value        A vector of positive size factors.

Details
The observed counts including negative control genes and housekeeping control genes might be
effect by some experimental factors like hybridization and binding efficiency. In order to get the
ture rate of gene expression, these variations must be normalized. Positive size factors can normalize
this kind of variation.

Value
A vector contain positive size factors

Author(s)
Hong Wang <hong.wang@uky.edu> chi wang <chi.wang@uky.edu>

See Also
positiveControl

Examples

data(NanoStringData)
## obtain positive factors
positiveFactor(NanoStringData)

## assign a vector
n=ncol(exprs(NanoStringData))
positiveFactor(NanoStringData)=rep(1,n)
Index

*Topic classes
  NanoStringSet-class, 6

*Topic datasets
  NanoStringData, 6

*Topic models
  glm.LRT, 3

*Topic normalization
  estNormalizationFactors, 2

*Topic package
  NanoStringDiff-package, 2

createNanoStringSet
  (NanoStringSet-class), 6
createNanoStringSetFromCsv
  (NanoStringSet-class), 6
estNormalizationFactors, 2
estNormalizationFactors,estNormalizationFactors-method
  (estNormalizationFactors), 2

glm.LRT, 2, 3
glm.LRT,NanoStringSet-method (glm.LRT),
  3

housekeepingControl, 4
housekeepingControl,NanoStringSet-method
  (housekeepingControl), 4
housekeepingControl<-
  (housekeepingControl), 4
housekeepingControl<-,NanoStringSet,matrix-method
  (housekeepingControl), 4
housekeepingFactor, 5
housekeepingFactor,NanoStringSet-method
  (housekeepingFactor), 5
housekeepingFactor<-
  (housekeepingFactor), 5
housekeepingFactor<-,NanoStringSet,numeric-method
  (housekeepingFactor), 5

NanoStringData, 6
NanoStringDiff
  (NanoStringDiff-package), 2
NanoStringDiff-package, 2
NanoStringSet (NanoStringSet-class), 6
NanoStringSet-class, 6

negativeControl, 8
negativeControl,NanoStringSet-method
  (negativeControl), 8
negativeControl<-(negativeControl), 8
negativeControl<-,NanoStringSet,matrix-method
  (negativeControl), 8
negativeFactor, 9
negativeFactor,NanoStringSet-method
  (negativeFactor), 9
negativeFactor<-(negativeFactor), 9
negativeFactor<-,NanoStringSet,numeric-method
  (negativeFactor), 9

positiveControl, 10
positiveControl,NanoStringSet-method
  (positiveControl), 10
positiveControl<-(positiveControl), 10
positiveControl<-,NanoStringSet,matrix-method
  (positiveControl), 10
positiveFactor, 11
positiveFactor,NanoStringSet-method
  (positiveFactor), 11
positiveFactor<-(positiveFactor), 11
positiveFactor<-,NanoStringSet,numeric-method
  (positiveFactor), 11