Package ‘HTSFilter’

January 21, 2016

Type Package
Title Filter replicated high-throughput transcriptome sequencing data
Version 1.10.0
Date 2015-03-04
Author Andrea Rau, Melina Gallopin, Gilles Celeux, and Florence Jaffrezic
Maintainer Andrea Rau <andrea.rau@jouy.inra.fr>
Depends methods, Biobase (>= 2.27.3), R (>= 3.2)
Imports DESeq (>= 1.19.0), edgeR (>= 3.9.14), DESeq2 (>= 1.6.3)
Suggests EDASeq (>= 2.1.4), BiocStyle
Description This package implements a filtering procedure for replicated transcriptome sequencing data based on a global Jaccard similarity index in order to identify genes with low, constant levels of expression across one or more experimental conditions.
License Artistic-2.0
LazyLoad yes
biocViews Sequencing, RNASeq, Preprocessing, DifferentialExpression, GeneExpression, Normalization

NeedsCompilation no

R topics documented:

HTSFilter-package .................................................... 2
HTSBasicFilter ...................................................... 4
HTSFilter ............................................................ 7
normalizeData ......................................................... 11
sultan ................................................................. 12

Index 14
HTSFilter-package  
*Filter replicated high-throughput transcriptome sequencing data*

**Description**

This package implements a filtering procedure for replicated transcriptome sequencing data based on a global Jaccard similarity index in order to identify genes with low, constant levels of expression across one or more experimental conditions.

**Details**

- **Package:** HTSFilter
- **Type:** Package
- **Version:** 1.7.1
- **Date:** 2015-03-04
- **License:** Artistic-2.0

**Author(s)**

Andrea Rau, Melina Gallopin, Gilles Celeux, and Florence Jaffrezic

Maintainer: Andrea Rau &lt;andrea.rau@jouy.inra.fr&gt;

**References**


**Examples**

```r
data("sultan")
conds &lt;- pData(sultan)$cell.line

#------------------------------------
## Matrix or data.frame
#------------------------------------

filter &lt;- HTSFilter(exprs(sultan), conds, s.len=25, plot=FALSE)
```
```r
# CountDataSet
library(DESeq)
cds <- newCountDataSet(exprs(sultan), conds)
cds <- estimateSizeFactors(cds)
cds <- estimateDispersions(cds)
cds <- HTSFilter(cds, s.len=25, plot=FALSE)$filteredData
class(cds)
## res <- nbinomTest(cds, levels(conds)[1], levels(conds)[2])

# DGEExact
library(edgeR)
dge <- DGEList(counts=exprs(sultan), group=conds)
dge <- calcNormFactors(dge)
dge <- estimateCommonDisp(dge)
dge <- estimateTagwiseDisp(dge)
et <- exactTest(dge)
et <- HTSFilter(et, DGEList=dge, s.len=25, plot=FALSE)$filteredData
## topTags(et)

# DESeq2
library(DESeq2)
dds <- DESeqDataSetFromMatrix(countData = exprs(sultan),
   colData = data.frame(cell.line = conds),
   design = ~ cell.line)
## Not run:
##
## dds <- DESeq(dd)
##
## filter <- HTSFilter(dd, s.len=25, plot=FALSE)$filteredData
## class(filter)
```
## HTSBasicFilter

Implement basic filters for transcriptome sequencing data.

### Description

This function implements a variety of basic filters for transcriptome sequencing data.

### Usage

```r
## S4 method for signature 'matrix'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("TMM", "DESeq", "none"))

## S4 method for signature 'data.frame'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("TMM", "DESeq", "none"))

## S4 method for signature 'CountDataSet'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("DESeq", "TMM", "none"))

## S4 method for signature 'DGEList'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("TMM", "DESeq", "pseudocounts", "none"))

## S4 method for signature 'DGEExact'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("TMM", "DESeq", "pseudocounts", "none"))

## S4 method for signature 'DGEGLM'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("TMM", "DESeq", "none"))

## S4 method for signature 'DGElrt'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("TMM", "DESeq", "none"))

## S4 method for signature 'DESeqDataSet'
HTSBasicFilter(x, method, cutoff.type="value", cutoff=10, length=NA, normalization=c("DESeq", "TMM", "none"), pAdjustMethod="BH")
```
Arguments

x A numeric matrix or data.frame representing the counts of dimension \((g \times n)\), for \(g\) genes in \(n\) samples, a CountDataSet object, a DGEList object, a DGEExact object, a DGEGLM object, a DGEList object, or a DESeqDataSet object.

method Basic filtering method to be used: “mean”, “sum”, “rpkm”, “variance”, “cpm”, “max”, “cpm.mean”, “cpm.sum”, “cpm.variance”, “cpm.max”, “rpkm.mean”, “rpkm.sum”, “rpkm.variance”, or “rpkm.max”

cutoff.type Type of cutoff to be used: a numeric value indicating the number of samples to be used for filtering (when \(\text{method} = \text{“cpm”} \text{ or } \text{“rpkm”}) \), or one of “value”, “number”, or “quantile”

cutoff Cutoff to be used for chosen filter

length Optional vector of length \(n\) containing the lengths of each gene in \(x\); optional except in the case of \(\text{method} = \text{“rpkm”}\)

normalization Normalization method to be used to correct for differences in library sizes, with choices “TMM” (Trimmed Mean of M-values), “DESeq” (normalization method proposed in the DESeq package), “pseudo.counts” (pseudo-counts obtained via quantile-quantile normalization in the edgeR package, only available for objects of class DGEList and DGEExact), and “none” (to be used only if user is certain no normalization is required, or if data have already been pre-normalized by an alternative method)

p.adjustMethod The method used to adjust p-values, see ?p.adjust

Details

This function implements a basic filter for high-throughput sequencing data for a variety of filter types: mean, sum, RPKM, variance, CPM, maximum, mean CPM values, the sum of CPM values, the variance of CPM values, maximum CPM value, mean RPKM values, the sum of RPKM values, the variance of RPKM values, or the maximum RPKM value. The filtering criteria used may be for a given cutoff value, a number of genes, or a given quantile value.

Value

filteredData An object of the same class as \(x\) containing the data that passed the filter

on A binary vector of length \(g\), where 1 indicates a gene with normalized expression greater than the optimal filtering threshold \(s_{\text{optimal}}\) in at least one sample (irrespective of condition labels), and 0 indicates a gene with normalized expression less than or equal to the optimal filtering threshold in all samples

normFactor A vector of length \(n\) giving the estimated library sizes estimated by the normalization method specified in normalization

removedData A matrix containing the filtered data

filterCrit A vector or matrix containing the criteria used to perform filtering
Author(s)
Andrea Rau, Melina Gallopin, Gilles Celeux, and Florence Jaffrezic

References

Examples

data("sultan")
conds <- pData(sultan)$cell.line

library(deseq)
library(edgeR)
library(dgeExact)

### Matrix or data.frame

```
## Filter genes with total (sum) normalized gene counts < 10
filter <- HTSBasicFilter(exprs(sultan), method="sum", cutoff.type="value",
cutoff = 10)
```

```
## CountDataSet

library(DESeq)

## Filter genes with mean normalized gene counts less than the 40% quantile
cds <- newCountDataSet(exprs(sultan), conds)
filter <- HTSBasicFilter(cds, method="mean", cutoff.type="quantile",
cutoff = 0.4)
```

```
## DGEExact

library(edgeR)

## Filter genes with CPM values less than 100 in more than 2 samples
dge <- DGEList(counts=.exprs(sultan), group=conds)
dge <- calcNormFactors(dge)
filter <- HTSBasicFilter(dge, method="cpm", cutoff.type=2, cutoff=100)
```
## DESeq2

```r
library(DESeq2)

dds <- DESeqDataSetFromMatrix(countData = exprs(sultan),
                               colData = data.frame(cell.line = conds),
                               design = ~ cell.line)

## Not run: Filter genes with mean normalized gene counts < 40% quantile
##
## dds <- DESeq(dds)
## filter <- HTSBasicFilter(dds, method="mean", cutoff.type="quantile",
##                          cutoff = 0.4)
## res <- results(filter, independentFiltering=FALSE)
```

---

### HTSFILTER

**Calculate data-based filtering threshold for replicated transcriptome sequencing data.**

#### Description

Calculate a data-based filtering threshold for replicated transcriptome sequencing data through the pairwise Jaccard similarity index between pairs of replicates within each experimental condition.

#### Usage

- **S4 method for signature 'matrix'**
  ```r
  HTSFILTER(x, conds, s.min=1, s.max=200, s.len=100,
            loess.span=0.3, normalization=c("TMM", "DESeq", "none"),
            plot=TRUE, plot.name=NA)
  ```

- **S4 method for signature 'data.frame'**
  ```r
  HTSFILTER(x, conds, s.min=1, s.max=200, s.len=100,
            loess.span=0.3, normalization=c("TMM", "DESeq", "none"),
            plot=TRUE, plot.name=NA)
  ```

- **S4 method for signature 'CountDataSet'**
  ```r
  HTSFILTER(x, conds=NA, s.min=1, s.max=200, s.len=100,
            loess.span=0.3, normalization=c("DESeq", "TMM", "none"),
            plot=TRUE, plot.name=NA)
  ```

- **S4 method for signature 'DGEList'**
  ```r
  HTSFILTER(x, conds=NA, s.min=1, s.max=200, s.len=100,
            loess.span=0.3, normalization=c("DESeq", "TMM", "none"),
            plot=TRUE, plot.name=NA)
  ```
Arguments

**x**
A numeric matrix or data.frame representing the counts of dimension \((g \times n)\), for \(g\) genes in \(n\) samples, a CountDataSet object, a DGEList object, a DGEExact object, a DGEGLM object, a DGELRT object, or a DESeqDataSet object.

**conds**
Vector of length \(n\) identifying the experimental condition of each of the \(n\) samples; required when `sQuote(x)` is a numeric matrix.

**s.min**
Minimum value of filtering threshold to be considered, with default value equal to 1

**s.max**
Maximum value of filtering threshold to be considered, with default value equal to 200

**s.len**
Length of sequence of filtering thresholds to be considered (from `s.min` to `s.max`) for the calculation of the global similarity index

**loess.span**
Span of the loess curve to be fitted to the filtering thresholds and corresponding global similarity indices, with default value equal to 0.3

**normalization**
Normalization method to be used to correct for differences in library sizes, with choices “TMM” (Trimmed Mean of M-values), “DESeq” (normalization method proposed in the DESeq package), “pseudo.counts” (pseudo-counts obtained via quantile-quantile normalization in the edgeR package, only available for objects of class DGEList and DGEExact), and “none” (to be used only if user is certain no normalization is required, or if data have already been pre-normalized by an alternative method)
HTSFilter

plot If “TRUE”, produce a plot of the calculated global similarity indices against the filtering threshold with superimposed loess curve
plot.name If plot = “TRUE”, the name of the PDF file to be saved to the current working directory. If plot.name = NA, the plot is drawn in the current window.
DGEList Object of class DGEList, to be used when filtering objects of class DGEExact
DGEGLM Object of class DGEGLM, to be used when filtering objects of class DGELRT
padjustmethod The method used to adjust p-values, see ?p.adjust

Details

The Jaccard similarity index, which measures the overlap of two sets, is calculated as follows. Given two binary vectors, each of length \( n \), we define the following values:

- \( a \) = the number of attributes with a value of 1 in both vectors
- \( b \) = the number of attributes with a value of 1 in the first vector and 0 in the second
- \( c \) = the number of attributes with a value of 0 in the first vector and 1 in the second
- \( d \) = the number of attributes with a value of 0 in both vectors

We note that all attributes fall into one of these four quantities, so \( a + b + c + d = n \). Given these quantities, we may calculate the Jaccard similarity index between the two vectors as follows:

\[
J = \frac{a}{a + b + c}.
\]

Value

filteredData An object of the same class as \( x \) containing the data that passed the filter
on A binary vector of length \( g \), where 1 indicates a gene with normalized expression greater than the optimal filtering threshold \( s_{\text{optimal}} \) in at least one sample (irrespective of condition labels), and 0 indicates a gene with normalized expression less than or equal to the optimal filtering threshold in all samples
s The optimal filtering threshold as identified by the global similarity index
indexValues A matrix of dimension \((s\cdot\text{len} \times 2)\) giving the tested filtering thresholds and the corresponding global similarity indices. Note that the threshold values are equally spaced on the log scale, and thus unequally spaced on the count scale (i.e., we test more threshold values at very low levels of expression, and fewer at very high levels of expression).
normFactor A vector of length \( n \) giving the estimated library sizes estimated by the normalization method specified in normalization
removedData A matrix containing the filtered data

Note

Filter should only be calculated on REPLICATED high-throughput sequencing data.
Author(s)

Andrea Rau, Melina Gallopin, Gilles Celeux, and Florence Jaffrezic

References


Examples

data("sultan")
conds <- pData(sultan)$cell.line

library(DESeq)
cds <- newCountDataSet(exprs(sultan), conds)
cds <- estimateSizeFactors(cds)
cds <- estimateDispersion(cds)
cds <- HTSFilter(cds, s.len=25, plot=FALSE)$filteredData
class(cds)

library(edgeR)
dge <- DGEList(counts=exprs(sultan), group=conds)
dge <- calcNormFactors(dge)
dge <- estimateCommonDisp(dge)
dge <- estimateTagwiseDisp(dge)
et <- exactTest(dge)
et <- HTSFilter(et, DGEList=dge, s.len=25, plot=FALSE)$filteredData

# toptags(et)
normalizeData

 Normalize transcriptome sequencing data.

Description

Normalize count-based measures of transcriptome sequencing data using the Trimmed Means of M-values (TMM) or DESeq approach.

Usage

normalizeData(data, normalization)

Arguments

data A numeric matrix representing the counts of dimension \((g \times n)\), for \(g\) genes in \(n\) samples.

normalization Normalization method to be used to correct for differences in library sizes, with choices “TMM” (Trimmed Mean of M-values), “DESeq” (normalization method proposed in the DESeq package), and “none”
Value

data.norm A numeric matrix representing the normalized counts of dimension (g x n), for g genes in n samples.
norm.factor A vector of length n giving the estimated library sizes estimated by the normalization method specified in normalization

Author(s)

Andrea Rau, Melina Gallopin, Gilles Celeux, and Florence Jaffrezic

References


Examples

data("sultan")
normData <- normalizeData(exprs(sultan), norm="DESeq")

sultan 

RNA-seq data from humans in Sultan et al. (2008)

Description

This dataset represents RNA-seq data from humans in two conditions (Ramos B cell line and HEK293T), with two biological replicates per condition. The ExpressionSet was downloaded from the ReCount online resource.

Usage

sultan

Format

An ExpressionSet named sultan.eset containing the phenotype data and expression data for the Sultan et al. (2008) experiment. Phenotype data may be accessed using the pData function, and expression data may be accessed using the exprs function.
**Source**

ReCount online resource (http://bowtie-bio.sourceforge.net/recount).

**References**


Index

*Topic datasets
  sultan, 12

*Topic methods
  HTSBasicFilter, 4
  HTSFFilter, 7
  normalizeData, 11

*Topic package
  HTSFFilter-package, 2

HTSBasicFilter, 4
  HTSBasicFilter, CountDataSet-method
    (HTSBasicFilter), 4
  HTSBasicFilter, data.frame-method
    (HTSBasicFilter), 4
  HTSBasicFilter, DESeqDataSet-method
    (HTSBasicFilter), 4
  HTSBasicFilter, DGEExact-method
    (HTSBasicFilter), 4
  HTSBasicFilter, DGEGLM-method
    (HTSBasicFilter), 4
  HTSBasicFilter, DGEList-method
    (HTSBasicFilter), 4
  HTSBasicFilter, DGERT-method
    (HTSBasicFilter), 4
  HTSBasicFilter-methods
    (HTSBasicFilter), 4

HTSFFilter, 7
  HTSFFilter, CountDataSet-method
    (HTSFFilter), 7
  HTSFFilter, data.frame-method
    (HTSFFilter), 7
  HTSFFilter, DESeqDataSet-method
    (HTSFFilter), 7
  HTSFFilter, DGEExact-method (HTSFFilter), 7
  HTSFFilter, DGEGLM-method (HTSFFilter), 7
  HTSFFilter, DGEList-method (HTSFFilter), 7
  HTSFFilter, DGERT-method (HTSFFilter), 7
  HTSFFilter, matrix-method (HTSFFilter), 7

HTSFFilter-methods (HTSFFilter), 7
HTSFFilter-package, 2
normalizeData, 11
sultan, 12