Package ‘RUvSeq’

March 29, 2017

Version 1.8.0

Title Remove Unwanted Variation from RNA-Seq Data

Description This package implements the remove unwanted variation (RUV) methods of Risso et al. (2014) for the normalization of RNA-Seq read counts between samples.

Author Davide Risso [aut, cre, cph], Sandrine Dudoit [aut], Lorena Panton [ctb], Kamil Slowikowski [ctb]

Maintainer Davide Risso <risso.davide@gmail.com>

Date 04-15-2014

Imports methods, MASS

Depends Biobase, EDASeq (>= 1.99.1), edgeR

Suggests BiocStyle, knitr, RColorBrewer, zebrafishRNASeq, DESeq2

VignetteBuilder knitr

License Artistic-2.0

LazyLoad yes

biocViews DifferentialExpression, Preprocessing, RNASeq, Software

URL https://github.com/drisso/RUVSeq

BugReports https://github.com/drisso/RUVSeq/issues

NeedsCompilation no

R topics documented:

RUvSeq-package .................................................. 2
makeGroups ...................................................... 2
residuals.DGEGLM .............................................. 3
RUvg-methods ................................................... 4
RUvr-methods ................................................... 6
RUvs-methods ................................................... 8

Index 10
RUVSeq-package  Remove Unwanted Variation from RNA-Seq Data

Description

This package implements the remove unwanted variation (RUV) methods of Risso et al. (2014) for the normalization of RNA-Seq read counts between samples.

Details

Package: RUVSeq
Type: Package
Version: 0.99.1
Date: 2014-04-15
License: Artistic-2.0

The RUVg function implements the RUVg normalization procedure of Risso et al. (2014), by using control genes to remove unwanted variation from the RNA-Seq read counts.

See also RUVr and RUVs for the "residual" and "sample" methods, based, respectively, on residuals (e.g., deviance residuals from a first-pass GLM regression of the unnormalized counts on the covariates of interest) and replicate/negative control samples for which the covariates of interest are constant.

Author(s)

Davide Risso and Sandrine Dudoit
Maintainer: Davide Risso <risso.davide@gmail.com>

References


See Also

RUVg, RUVr, RUVs

makeGroups

Make a matrix suitable for use with RUVSeq methods such as RUVs.

Description

Each row in the returned matrix corresponds to a set of replicate samples. The number of columns is the size of the largest set of replicates; rows for smaller sets are padded with -1 values.
residuals.DGEGLM

Usage
makeGroups(xs)

Arguments
xs A vector indicating membership in a group.

Author(s)
Kamil Slowikowski

See Also
RUVs

Examples
makeGroups(c("A","B","B","C","C","D","D","D","A"))

Description
This function implements the residuals method for the edgeR function glmFit.

Usage
## S3 method for class 'DGEGLM'
residuals(object, type = c("deviance", "pearson"), ...)

Arguments
object An object of class DGEGLM as created by the glmFit function of edgeR.
type Compute deviance or Pearson residuals..
... Additional arguments to be passed to the generic function.

Value
A genes-by-samples numeric matrix with the negative binomial residuals for each gene and sample.

Author(s)
Davide Risso

References
Examples

```r
library(edgeR)
library(zebrafishRNASeq)
data(zfGenes)

## run on a subset genes for time reasons
## (real analyses should be performed on all genes)
genes <- rownames(zfGenes)[grep("^ENS", rownames(zfGenes))]
spikes <- rownames(zfGenes)[grep("^ERCC", rownames(zfGenes))]
set.seed(123)
idx <- c(sample(genes, 1000), spikes)
seq <- newSeqExpressionSet(as.matrix(zfGenes[idx,]))

x <- as.factor(rep(c("Ctl", "Trt"), each=3))
design <- model.matrix(~x)
y <- DGEList(counts=counts(seq), group=x)
y <- calcNormFactors(y, method="upperquartile")
y <- estimateGLMCommonDisp(y, design)
y <- estimateGLMTagwiseDisp(y, design)

fit <- glmFit(y, design)
res <- residuals(fit, type="deviance")
head(res)
```

---

**RUVg-methods**  
*Remove Unwanted Variation Using Control Genes*

**Description**

This function implements the RUVg method of Risso et al. (2014).

**Usage**

```r
RUVg(x, cIdx, k, drop=0, center=TRUE, round=TRUE, epsilon=1, tolerance=1e-8, isLog=FALSE)
```

**Arguments**

- `x`: Either a genes-by-samples numeric matrix or a `SeqExpressionSet` object containing the read counts.
- `cIdx`: A character, logical, or numeric vector indicating the subset of genes to be used as negative controls in the estimation of the factors of unwanted variation.
- `k`: The number of factors of unwanted variation to be estimated from the data.
- `drop`: The number of singular values to drop in the estimation of the factors of unwanted variation. This number is usually zero, but might be set to one if the first singular value captures the effect of interest. It must be less than `k`.
- `center`: If TRUE, the counts are centered, for each gene, to have mean zero across samples. This is important to ensure that the first singular value does not capture the average gene expression.
- `round`: If TRUE, the normalized measures are rounded to form pseudo-counts.
- `epsilon`: A small constant (usually no larger than one) to be added to the counts prior to the log transformation to avoid problems with log(0).
tolerance  Tolerance in the selection of the number of positive singular values, i.e., a singular value must be larger than tolerance to be considered positive.

isLog  Set to TRUE if the input matrix is already log-transformed. Ignored if x is a SeqExpressionSet.

Details

The RUVg procedure performs factor analysis of the read counts based on a suitably-chosen subset of negative control genes known a priori not be differentially expressed (DE) between the samples under consideration.

Several types of controls can be used, including housekeeping genes, spike-in sequences (e.g., ERCC), or “in-silico” empirical controls (e.g., least significantly DE genes based on a DE analysis performed prior to RUV normalization).

Note that one can relax the negative control gene assumption by requiring instead the identification of a set of positive or negative controls, with a priori known expression fold-changes between samples. RUVg can then simply be applied to control-centered log counts, as detailed in the vignette.

Methods

signature(x = "matrix", cIdx = "ANY", k = "numeric") It returns a list with
- A samples-by-factors matrix with the estimated factors of unwanted variation (W).
- The genes-by-samples matrix of normalized expression measures (possibly rounded) obtained by removing the factors of unwanted variation from the original read counts (normalizedCounts).

signature(x = "SeqExpressionSet", cIdx = "character", k="numeric") It returns a SeqExpressionSet with
- The normalized counts in the normalizedCounts slot.
- The estimated factors of unwanted variation as additional columns of the phenoData slot.

Author(s)

Davide Risso

References


See Also

RUVr, RUVs.

Examples

library(zebrafishRNASeq)
data(zfGenes)

## run on a subset of genes for time reasons
## (real analyses should be performed on all genes)
genes <- rownames(zfGenes)[grep("^ENS", rownames(zfGenes))]
spikes <- rownames(zfGenes)[grep("^ERCC", rownames(zfGenes))]
set.seed(123)
idx <- c(sample(genes, 1000), spikes)
seq <- newSeqExpressionSet(as.matrix(zfGenes[idx,]))

# RUVg normalization
seqRUVg <- RUVg(seq, spikes, k=1)
pData(seqRUVg)
head(normCounts(seqRUVg))
plotRLE(seq, outline=FALSE, ylim=c(-3, 3))
plotRLE(seqRUVg, outline=FALSE, ylim=c(-3, 3))
barplot(as.matrix(pData(seqRUVg)), beside=TRUE)

---

**RUVr-methods**

*Remove Unwanted Variation Using Residuals*

**Description**

This function implements the RUVr method of Risso et al. (2014).

**Usage**

```
RUVr(x, cIdx, k, residuals, center=TRUE, round=TRUE, epsilon=1, tolerance=1e-8, isLog=FALSE)
```

**Arguments**

- **x**: Either a genes-by-samples numeric matrix or a *SeqExpressionSet* object containing the read counts.
- **cIdx**: A character, logical, or numeric vector indicating the subset of genes to be used as negative controls in the estimation of the factors of unwanted variation.
- **k**: The number of factors of unwanted variation to be estimated from the data.
- **residuals**: A genes-by-samples matrix of residuals obtained from a first-pass regression of the counts on the covariates of interest, usually the negative binomial deviance residuals obtained from *edgeR* with the *residuals* method.
- **center**: If `TRUE`, the residuals are centered, for each gene, to have mean zero across samples.
- **round**: If `TRUE`, the normalized measures are rounded to form pseudo-counts.
- **epsilon**: A small constant (usually no larger than one) to be added to the counts prior to the log transformation to avoid problems with log(0).
- **tolerance**: Tolerance in the selection of the number of positive singular values, i.e., a singular value must be larger than `tolerance` to be considered positive.
- **isLog**: Set to `TRUE` if the input matrix is already log-transformed. Ignored if `x` is a *SeqExpressionSet*.

**Details**

The RUVr procedure performs factor analysis on residuals, such as deviance residuals from a first-pass GLM regression of the counts on the covariates of interest using *edgeR*. The counts may be either unnormalized or normalized with a method such as upper-quartile (UQ) normalization.
Methods

signature(x = "matrix", cIdx = "ANY", k = "numeric", residuals = "matrix") It returns a list with

- A samples-by-factors matrix with the estimated factors of unwanted variation (\(W\)).
- The genes-by-samples matrix of normalized expression measures (possibly rounded) obtained by removing the factors of unwanted variation from the original read counts (normalizedCounts).

signature(x = "SeqExpressionSet", cIdx = "character", k="numeric", residuals = "matrix")
It returns a SeqExpressionSet with

- The normalized counts in the normalizedCounts slot.
- The estimated factors of unwanted variation as additional columns of the phenoData slot.

Author(s)

Davide Risso

References


See Also

RUVg, RUVs, residuals.

Examples

library(edgeR)
library(zebrafishRNASeq)
data(zfGenes)

## run on a subset of genes for time reasons
## (real analyses should be performed on all genes)
genres <- rownames(zfGenes)[grep("^ENS", rownames(zfGenes))]
spikes <- rownames(zfGenes)[grep("^ERCC", rownames(zfGenes))]
set.seed(123)
idx <- c(sample(genres, 1000), spikes)
seq <- newSeqExpressionSet(as.matrix(zfGenes[idx,]))

# Residuals from negative binomial GLM regression of UQ-normalized
counts on covariates of interest, with edgeR
x <- as.factor(rep(c("Ctl", "Trt"), each=3))
design <- model.matrix(~x)
y <- DGEList(counts=counts(seq), group=x)
y <- calcNormFactors(y, method="upperquartile")
y <- estimateGLMCommonDisp(y, design)
y <- estimateGLMTagwiseDisp(y, design)
fit <- glmFit(y, design)
res <- residuals(fit, type="deviance")
Removing Unwanted Variation Using Replicate/Negative Control Samples

**Description**

This function implements the RUVs method of Risso et al. (2014).

**Usage**

```r
RUVs(x, cIdx, k, scIdx, round=TRUE, epsilon=1, tolerance=1e-8, isLog=FALSE)
```

**Arguments**

- `x`: Either a genes-by-samples numeric matrix or a `SeqExpressionSet` object containing the read counts.
- `cIdx`: A character, logical, or numeric vector indicating the subset of genes to be used as negative controls in the estimation of the factors of unwanted variation.
- `k`: The number of factors of unwanted variation to be estimated from the data.
- `scIdx`: A numeric matrix specifying the replicate samples for which to compute the count differences used to estimate the factors of unwanted variation (see details).
- `round`: If `TRUE`, the normalized measures are rounded to form pseudo-counts.
- `epsilon`: A small constant (usually no larger than one) to be added to the counts prior to the log transformation to avoid problems with log(0).
- `tolerance`: Tolerance in the selection of the number of positive singular values, i.e., a singular value must be larger than `tolerance` to be considered positive.
- `isLog`: Set to `TRUE` if the input matrix is already log-transformed. Ignored if `x` is a `SeqExpressionSet`.

**Details**

The RUVs procedure performs factor analysis on a matrix of count differences for replicate/negative control samples, for which the biological covariates of interest are constant.

Each row of `scIdx` should correspond to a set of replicate samples. The number of columns is the size of the largest set of replicates; rows for smaller sets are padded with `-1` values.

For example, if the sets of replicate samples are `(1,11,21), (2,3), (4,5), (6,7,8)`, then `scIdx` should be:

```
  1 11 21
  2 3 -1
  4 5 -1
  6 7 8
```
Methods

signature(x = "matrix", cIdx = "ANY", k = "numeric", scIdx = "matrix") It returns a list with
  • A samples-by-factors matrix with the estimated factors of unwanted variation (W).
  • The genes-by-samples matrix of normalized expression measures (possibly rounded) obtained by removing the factors of unwanted variation from the original read counts (normalizedCounts).

signature(x = "SeqExpressionSet", cIdx = "character", k="numeric", scIdx = "matrix")
It returns a SeqExpressionSet with
  • The normalized counts in the normalizedCounts slot.
  • The estimated factors of unwanted variation as additional columns of the phenoData slot.

Author(s)

Davide Risso (building on a previous version by Laurent Jacob).

References


See Also

RUVg, RUVr.

Examples

library(zebrafishRNASeq)
data(zfGenes)

## run on a subset of genes for time reasons
## (real analyses should be performed on all genes)
genesis <- rownames(zfGenes)[grep("^ENS", rownames(zfGenes))]
spikes <- rownames(zfGenes)[grep("^ERCC", rownames(zfGenes))]
set.seed(123)
idx <- c(sample(genes, 1000), spikes)
seq <- newSeqExpressionSet(as.matrix(zfGenes[idx,]))

# RUVs normalization
controls <- rownames(seq)
differences <- matrix(data=c(1:3, 4:6), byrow=TRUE, nrow=2)
seqRUVs <- RUVs(seq, controls, k=1, differences)

pData(seqRUVs)
head(normCounts(seqRUVs))
Index

*Topic package*
  RUVSeq-package, 2

DGEGLM, 3
edgeR, 3, 6
glmFit, 3
makeGroups, 2
residuals, 3, 6, 7
residuals.DGEGLM, 3
RUVg, 2, 7, 9
RUVg (RUVg-methods), 4
RUVg, matrix, ANY, numeric-method
  (RUVg-methods), 4
RUVg, SeqExpressionSet, character, numeric-method
  (RUVg-methods), 4
RUVg-methods, 4
RUVr, 2, 5, 9
RUVr (RUVr-methods), 6
RUVr, matrix, ANY, numeric, matrix-method
  (RUVr-methods), 6
RUVr, SeqExpressionSet, character, numeric, matrix-method
  (RUVr-methods), 6
RUVr-methods, 6
RUVs, 2, 3, 5, 7
RUVs (RUVs-methods), 8
RUVs, matrix, ANY, numeric, matrix-method
  (RUVs-methods), 8
RUVs, SeqExpressionSet, character, numeric, matrix-method
  (RUVs-methods), 8
RUVs-methods, 8
RUVSeq (RUVSeq-package), 2
RUVSeq-package, 2
SeqExpressionSet, 4–9