Package 'SVAPLSseq'

October 16, 2019

Type Package

Title SVAPLSseq-An R package to estimate the hidden factors of unwanted variability and adjust for them to enable a more powerful and accurate differential expression analysis based on RNAseq data

Description The package contains functions that are intended for extracting the signatures of latent variation in RNAseq data and using them to perform an improved differential expression analysis for a set of features (genes, transcripts) between two specified biological groups.

Version 1.10.0

Date 2017-05-31

Author Sutirtha Chakraborty

Maintainer Sutirtha Chakraborty <sutirtha_sutir@yahoo.co.in>

Depends R (>= 3.4)

Imports methods, stats, SummarizedExperiment, edgeR, ggplot2, limma, lmtest, parallel, pls

Suggests BiocStyle

License GPL-3

biocViews ImmunoOncology, GeneExpression, RNASeq, Normalization, BatchEffect

Collate AllClasses.R AllGenerics.R Methods.R svplsSurr.R svplsTest.R

NeedsCompilation no

RoxygenNote 6.0.1

git_url https://git.bioconductor.org/packages/SVAPLSseq

git_branch RELEASE_3_9

git_last_commit c304927

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

R topics documented:

`SVAPLSseq-package` *SVAPLSseq: An R package to adjust for the hidden factors of variability in differential gene expression studies based on RNAseq data.*

Description

The package `SVAPLSseq` contains functions that are intended for the identification and correction of the hidden variability owing to a variety of unknown subject/sample specific and technical effects of residual heterogeneity in an RNAseq gene expression data.

Details

Package:	SVAPLSseq
Type:	Package
License:	GPL-3

The package can be used to find the genes that are truly differentially expressed between two groups of samples from an RNAseq data, after adjusting for different hidden factors of expression heterogeneity. The function `svplsSurr` operates on the raw data matrix of gene level read counts and extracts the signatures of the underlying hidden variability in the form of a set of surrogate variables. The function `svplsTest` detects the truly positive genes after correcting for the hidden signals (surrogate variables) extracted by `svplsSurr`.

Author(s)

Sutirtha Chakraborty.

Maintainer: Sutirtha Chakraborty <statistuta@gmail.com>

References

Boulesteix, A-L. and Strimmer, K. Partial least squares: a versatile tool for the analysis of high-dimensional genomic data. Briefings in Bioinformatics 2007; **8**(1):32–44.

See Also

svplsSurr, svplsTest

Examples

```
##Loading the Simulated Data
data(sim.dat)

## Fitting a linear model with the surrogate variables and detecting the differentially expressed genes
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv <- svplsSurr(dat = sim.dat, group = group, surr.select = "automatic")
surr = surr(sv)
fit <- svplsTest(dat = sim.dat, group = group, surr = surr, normalization = "TMM", test = "t-test")
head(sig.features(fit))
```

---

prop.vars *Accessor for the 'prop.vars' slot of a 'svplsSurr' object*

Description

Accessor for the 'prop.vars' slot of a 'svplsSurr' object

Usage

```
prop.vars(object)

## S4 method for signature 'svplsSurr'
prop.vars(object)
```

Arguments

object	a svplsSurr object

Examples

```
data(sim.dat)
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv = svplsSurr(sim.dat, group, surr.select = "automatic")
prop.vars(sv)
```

---

pvs.adj *Accessor for the 'pvs.adj' slot of a 'svplsTest' object*

Description

Accessor for the 'pvs.adj' slot of a 'svplsTest' object

Usage

```
pvs.adj(object)

## S4 method for signature 'svplsTest'
pvs.adj(object)
```

Arguments

object	a svplsTest object

Examples

```
data(sim.dat)
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv = svplsSurr(sim.dat, group, surr.select = "automatic")
surr = surr(sv)
fit = svplsTest(dat = sim.dat, group = group, surr = surr, normalization = "TMM", test = "t-test")
pvs.adj(fit)
```

---

pvs.unadj *Accessor for the 'pvs.unadj' slot of a 'svplsTest' object*

Description

Accessor for the 'pvs.unadj' slot of a 'svplsTest' object

Usage

```
pvs.unadj(object)

## S4 method for signature 'svplsTest'
pvs.unadj(object)
```

Arguments

object	a svplsTest object

Examples

```
data(sim.dat)
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv = svplsSurr(sim.dat, group, surr.select = "automatic")
surr = surr(sv)
fit = svplsTest(dat = sim.dat, group = group, surr = surr, normalization = "TMM", test = "t-test")
pvs.unadj(fit)
```

sig.features *Accessor for the 'sig.features' slot of a 'svplsTest' object*

Description

Accessor for the 'sig.features' slot of a 'svplsTest' object

Usage

```
sig.features(object)

## S4 method for signature 'svplsTest'
sig.features(object)
```

Arguments

object	a svplsTest object

Examples

```
data(sim.dat)
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv = svplsSurr(sim.dat, group, surr.select = "automatic")
surr = surr(sv)
fit = svplsTest(dat = sim.dat, group = group, surr = surr, normalization = "TMM", test = "t-test")
sig.features(fit)
```

sim.dat *A simulated RNAseq gene expression count data affected by multiple hidden variables.*

Description

The dataset contains simulated raw RNAseq expression counts corresponding to 1000 genes over 20 subjects S1, S2..S20, distributed equally between two groups 1 and 2. (S1, S2...S10 belong to group 1 and S11, S12...S20 belong to group 2). The data is affected by the unknown effects from several technical and sample-specific artefacts. The data has been created to illustrate usage of the functions in this package.

surr *Accessor for the 'surr' slot of a 'svplsSurr' object*

Description

Accessor for the 'surr' slot of a 'svplsSurr' object

Usage

```
surr(object)

## S4 method for signature 'svplsSurr'
surr(object)
```

Arguments

object	a svplsSurr object

Examples

```
data(sim.dat)
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv = svplsSurr(sim.dat, group, surr.select = "automatic")
surr(sv)
```

svplsSurr *svplsSurr*

Description

This function extracts the surrogated estimates of the hidden variables in the data by using the partial least squares (PLS) algorithm on two multivariate random matrices. It provides the user with two options:

(1) **Unsupervised SVAPLS**: Here a standard linear regression model is first used on a transformed version of the expression count matrix to estimate the primary signals of differential expression for all the features. The fitted model residuals and the transformed count matrix are then organized respectively into two multivariate matrices `E` and `Y`, in such a way that each column corresponds to a certain feature. `Y` is then regressed on `E` using a Non-linear partial least squares (NPLS) algorithm and the extracted factor estimates (scores) in the column-space of `Y` are deemed as the surrogate variables.

(2) **Supervised SVAPLS**: In case information on a set of control features (control genes, transcripts, spike-ins, etc.) is provided, this function uses a Non-linear partial least squares (NPLS) algorithm to regress `Y` on another expression matrix `Y.cont` corresponding to the set of controls and the factor estimates (scores) in the column-space of `Y.cont` are considered as the surrogate variables.

An optimal subset of these variables is then selected either manually by the user (manual selection) or by testing them for statistical significance (automatic selection). For the automatic selection the function regresses the first right singular vector of the residual matrix `E` (for Unsupervised SVAPLS) or the control matrix `Y.cont` (for Supervised SVAPLS), on all the surrogate variables and the estimated regression coefficients are used to perform a t-test with a certain user-specified pvalue cutoff. The variables yielding a pvalue below the cutoff are returned as the optimal surrogate variables.

Usage

```
svplsSurr(dat, group, controls = NULL, phi = function(x) log(x + const),
  const = 1, pls.method = "oscorespls", max.surrs = 3, opt.surrs = 1,
  surr.select = c("automatic", "manual"), cutoff = 10^-7,
  parallel = FALSE, num.cores = NULL, plot = FALSE)
```

Arguments

dat	The original feature expression count matrix.
group	a factor representing the sample indices belonging to the two different groups.
controls	The set of control features with no differential expression between the two groups (set to NULL by default).
phi	The transforming function to be applied on the original feature expression count data (set to be log function with an offset const).
const	The offset parameter for the transforming function phi (set to 1 by default).
pls.method	The non-linear partial least squares method to be used. The different options available are: the classical orthogonal scores algorithm ("oscorespls, default), the kernel algorithm ("kernelpls") and wide kernel algorithm ("widekernelpls"). Using the "oscorespls" option is recommended for producing mutually orthogonal surrogate variables.
max.surrs	The maximum number of factor estimates to be extracted from the NPLS algorithm (set to 3 by default).
opt.surrs	The index vector of factor estimates to be taken as the optimal surrogate variables (used for manual selection only).
surr.select	The method for selecting the optimal surrogate variables ("automatic" or "manual").
cutoff	The user-specified pvalue cutoff for testing the significance of the extracted surrogate variables (set to 1e-07 by default) (used for "automatic" selection only).
parallel	Logical, indicating if the computations should be parallelized or not (set to FALSE by default).
num.cores	The requested number of cores to be used in the parallel computations inside the function (used only when parallel is TRUE, NULL by default).
plot	Logical, if TRUE a barplot of the variance proportions explained by the significant surrogate variables is returned (set to FALSE by default).

Value

surr A data.frame of the optimal surrogate variables.

prop.vars A vector of the variance proportions explained by the variables in surr.

Examples

```
##Loading a simulated RNAseq gene expression count dataset
data(sim.dat)

##Extracting the optimal surrogate variables
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv <- svplsSurr(dat = sim.dat, group = group, surr.select = "automatic")
slotNames(sv)
```

```
head(surr(sv))
head(prop.vars(sv))
```

`svplsSurr-class` *svplsSurr*

Description

The `svplsSurr` class. An object of this class contains the following two slots:

surr A `data.frame` of the optimal surrogate variables.

prop.vars A vector of the variance proportions in the data space that are explained by the variables in `surr`.

`svplsTest` *svplsTest*

Description

This function incorporates the significant surrogate variables returned by the function `svplsSurr` in a linear model along with the group variable in order to estimate the group effects more accurately. The reestimated primary signals (group effects) are then used to test the features for differential expression. The resulting pvalues are further corrected for multiple hypothesis testing at a prespecified FDR level. The significantly differentially expressed features are finally returned along with their uncorrected and corrected pvalues.

Usage

```
svplsTest(dat, phi = function(x) log(x + const), const = 1,
  normalization = c("TMM", "RLE", "upperquartile", "none"), group, surr,
  test = c("t-test", "LRT"), mht.method = "BH", fdr.level = 0.05,
  parallel = FALSE, num.cores = NULL)
```

Arguments

`dat`	The original feature expression count matrix.
`phi`	The transforming function to be applied on the original feature expression count data (set to be log function with an offset `const`).
`const`	The offset parameter for the transforming function `phi` (set to 1 by default).
`normalization`	The method to use for normalizing the RNAseq feature expression count data (options are "TMM", "RLE", "upperquartile" or "none").
`group`	a factor representing the sample indices belonging to the two different groups.
`surr`	A `data.frame` of the significant surrogate variables.
`test`	The test to be used for detecting the differentially expressed features. Options are "t-test" and "Likelihood Ratio Test (LRT)".
`mht.method`	The method to be used for the multiple hypothesis correction (set to the Benjamini-Hochberg procedure ("BH") by default).

fdr.level	The specified level of the False Discovery Rate (FDR) for the multiple hypothesis testing (set to 0.05 by default).
parallel	Logical, indicating if the computations should be parallelized or not (set to FALSE by default).
num.cores	The requested number of cores to be used in the parallel computations inside the function (used only when parallel is TRUE, NULL by default).

Value

pvs.unadj The uncorrected pvalues corresponding to the genes after adjusting for the signatures of hidden variability.

pvs.adj The multiple hypothesis corrected pvalues after adjusting for the signatures of hidden variability.

sig.features The features detected to be significantly differentially expressed between the two groups.

Examples

```
##Loading a simulated RNAseq gene expression count dataset
data(sim.dat)

##Fitting a linear model with the surrogate variables and detecting the
##differentially expressed genes
group = as.factor(c(rep(1, 10), rep(-1, 10)))
sv <- svplsSurr(dat = sim.dat, group = group, surr.select = "automatic")
surr = surr(sv)
fit = svplsTest(dat = sim.dat, group = group, surr = surr, normalization = "TMM", test = "t-test")

##The detected genes, hidden effect adjusted pvalues, FDR-corrected pvalues and the positive genes
##detected from the fitted model are given by:
head(sig.features(fit))

head(pvs.unadj(fit))

head(pvs.adj(fit))
```

---

svplsTest-class *svplsTest*

Description

The svplsTest class. An object of this class contains the following three slots:

pvs.unadj The uncorrected pvalues corresponding to the features after adjusting for the signatures of hidden variability.

pvs.adj The multiple hypothesis corrected pvalues after adjusting for the signatures of hidden variability in the data.

sig.features The features detected to be significantly differentially expressed between the two groups.

Index