Package 'qsvaR'

May 2, 2024

Title Generate Quality Surrogate Variable Analysis for Degradation Correction

Version 1.9.0

Date 2023-06-27

Description The qsvaR package contains functions for removing the effect of degration in rna-seq data from postmortem brain tissue. The package is equipped to help users generate principal components associated with degradation. The components can be used in differential expression analysis to remove the effects of degradation.

License Artistic-2.0

URL https://github.com/LieberInstitute/qsvaR

BugReports https://support.bioconductor.org/t/qsvaR

biocViews Software, WorkflowStep, Normalization, BiologicalQuestion, DifferentialExpression, Sequencing, Coverage

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Suggests BiocFileCache, BiocStyle, covr, knitr, limma, RefManageR, rmarkdown, sessioninfo, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports sva, stats, ggplot2, methods

Depends R (>= 4.2), SummarizedExperiment

LazyData true

VignetteBuilder knitr

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

git_branch devel

git_last_commit e867a41

git_last_commit_date 2024-04-30

Repository Bioconductor 3.20

Date/Publication 2024-05-01

Author Joshua Stolz [aut] (<https://orcid.org/0000-0001-5694-5247>),
Hedia Tnani [ctb, cre] (<https://orcid.org/0000-0002-0380-9740>),
Leonardo Collado-Torres [ctb] (<https://orcid.org/0000-0003-2140-308X>)

Maintainer Hedia Tnani <hediatnani0@gmail.com>

Contents

`covComb_tx_deg` *RSE object of RNA-seq data that serves as output for degradation analysis*

Description

This data was generated from an experiment using degraded RNA-seq samples post-mortem brain tissue. The transcripts included are the result of the qsva expanded framework study and will be used to remove the effect of degradation in bulk RNA-seq data.

Format

A RangedSummarizedExperiment-class

See Also

getPCs k_qsvs getDegTx

degradation_tstats *Degradation time t-statistics*

Description

These t-statistics are derived from the same data that was used for covComb_tx_deg. They are the results from main model where we determined the relationship with degradation time adjusting for the brain region (so parallel degradation effects across brain regions). They are used for plotting in `DEqual()`.

Format

A `data.frame()` with the `t` statistics for degradation time. The `rownames()` are the GENCODE transcript IDs.

See Also

DEqual

DEqual *Differential expression quality (DEqual) plot*

Description

A DEqual plot compares the effect of RNA degradation from an independent degradation experiment on the y axis to the effect of the outcome of interest. They were orignally described by Jaffe et al, PNAS, 2017 https://doi.org/10.1073/pnas.1617384114. Other DEqual versions are included in Collado-Torres et al, Neuron, 2019 https://doi.org/10.1016/j.neuron.2019.05.013. This function compares your t-statistics of interest computed on transcripts against the t-statistics from degradation time adjusting for the six brain regions from degradation experiment data used for determining `covComb_tx_deg`.

Usage

```
DEqual(DE)
```

Arguments

DE	a `data.frame()` with one column containing the t-statistics from Differential Expression, typically generated with `limma::topTable()`. The `rownames(DE)` should be transcript GENCODE IDs.

Value

a `ggplot` object of the DE t-statistic vs the DE statistic from degradation

Examples

```
## Random differential expression t-statistics for the same transcripts
## we have degradation t-statistics for in `degradation_tstats`.
set.seed(101)
random_de <- data.frame(
    t = rt(nrow(degradation_tstats), 5),
    row.names = sample(
        rownames(degradation_tstats),
        nrow(degradation_tstats)
    )
)

## Create the DEqual plot
DEqual(random_de)
```

`getDegTx` *Obtain expression matrix for degraded transcripts*

Description

This function is used to obtain a RangedSummarizedExperiment-class of transcripts and their expression values #' These transcripts are selected based on a prior study of RNA degradation in postmortem brain tissues. This object can later be used to obtain the principle components necessary to remove the effect of degradation in differential expression.

Usage

```
getDegTx(
  rse_tx,
  type = "cell_component",
  sig_transcripts = select_transcripts(type),
  assayname = "tpm"
)
```

Arguments

`rse_tx`	A RangedSummarizedExperiment-class object containing the transcript data desired to be studied.
`type`	A `character(1)` specifying the transcripts set type. These were determined by Joshua M. Stolz et al, 2022. Here the names "cell_component", "top1500", and "standard" refer to models that were determined to be effective in removing degradation effects. The "standard" model involves taking the union of the top 1000 transcripts associated with degradation from the interaction model and the main effect model. The "top1500" model is the same as the "standard model except the union of the top 1500 genes associated with degradation is selected. The most effective of our models, "cell_component", involved deconvolution of the degradation matrix to determine the proportion of cell types within our

studied tissue. These proportions were then added to our `model.matrix()` and the union of the top 1000 transcripts in the interaction model, the main effect model, and the cell proportions model were used to generate this model of qSVs.

`sig_transcripts`
: A list of transcripts determined to have degradation signal in the qsva expanded paper.

`assayname`
: character string specifying the name of the assay desired in rse_tx

Value

A RangedSummarizedExperiment-class object.

Examples

```
getDegTx(covComb_tx_deg)
stopifnot(mean(rowMeans(assays(covComb_tx_deg)$tpm)) > 1)
```

---

getPCs *PCs from transcripts*

Description

This function returns the pcs from the obtained RangedSummarizedExperiment object of selected transcripts

Usage

```
getPCs(rse_tx, assayname = "tpm")
```

Arguments

`rse_tx`
: Ranged Summarizeed Experiment with only trancsripts selected for qsva

`assayname`
: character string specifying the name of the assay desired in rse_tx

Value

prcomp object generated by taking the pcs of degraded transcripts

Examples

```
getPCs(covComb_tx_deg, "tpm")
```

get_qsvs *Generate matrix of qsvs*

Description

Using the pcs and the k number of components be included, we generate the qsva matrix.

Usage

```
get_qsvs(qsvPCs, k)
```

Arguments

qsvPCs	prcomp object generated by taking the pcs of degraded transcripts
k	number of qsvs to be included.

Value

matrix with k principal components for each sample.

Examples

```
qsv <- list(x = matrix(seq_len(9), ncol = 3))
get_qsvs(qsv, 2)
```

k_qsvs *Apply num.sv algorithm to determine the number of pcs to be included*

Description

Apply num.sv algorithm to determine the number of pcs to be included

Usage

```
k_qsvs(rse_tx, mod, assayname)
```

Arguments

rse_tx	A RangedSummarizedExperiment-class object containing the transcript data desired to be studied.
mod	Model Matrix with necessary variables the you would model for in differential expression
assayname	character string specifying the name of the assay desired in rse_tx

Value

integer representing number of pcs to be included

Examples

```
## First we need to define a statistical model. We'll use the example
## covComb_tx_deg data. Note that the model you'll use in your own data
## might look different from this model.
mod <- model.matrix(~ mitoRate + Region + rRNA_rate + totalAssignedGene + RIN,
    data = colData(covComb_tx_deg)
)

## To ensure that the results are reproducible, you will need to set a
## random seed with the set.seed() function. Internally, we are using
## sva::num.sv() which needs a random seed to ensure reproducibility of the
## results.
set.seed(20230621)
k_qsvs(covComb_tx_deg, mod, "tpm")
```

---

qSVA *A wrapper function used to perform qSVA in one step.*

Description

A wrapper function used to perform qSVA in one step.

Usage

```
qSVA(
  rse_tx,
  type = "cell_component",
  sig_transcripts = select_transcripts(type),
  mod,
  assayname
)
```

Arguments

rse_tx	A RangedSummarizedExperiment-class object containing the transcript data desired to be studied.
type	a character string specifying which model you would like to use when selecting a degradation matrix.
sig_transcripts	A list of transcripts that are associated with degradation signal. Use `select_transcripts()` to select sets of transcripts identified by the qSVA expanded paper. Specifying a `character()` input of ENSEMBL transcript IDs (or whatever values you have at `rownames(rse_tx)`) obtained outside of `select_transcripts()` overrides the user friendly `type` argument. That is, this argument provides more fine tuning options for advanced users.

mod	Model Matrix with necessary variables the you would model for in differential expression
assayname	character string specifying the name of the assay desired in rse_tx

Value

matrix with k principal components for each sample

Examples

```
## First we need to define a statistical model. We'll use the example
## covComb_tx_deg data. Note that the model you'll use in your own data
## might look different from this model.
mod <- model.matrix(~ mitoRate + Region + rRNA_rate + totalAssignedGene + RIN,
    data = colData(covComb_tx_deg)
)

## To ensure that the results are reproducible, you will need to set a
## random seed with the set.seed() function. Internally, we are using
## sva::num.sv() which needs a random seed to ensure reproducibility of the
## results.
set.seed(20230621)
qSVA(rse_tx = covComb_tx_deg, type = "cell_component", mod = mod, assayname = "tpm")
```

---

select_transcripts *Select transcripts associated with degradation*

Description

Helper function to select which experimental model will be used to generate the qSVs.

Usage

```
select_transcripts(type = c("cell_component", "top1500", "standard"))
```

Arguments

type	A `character(1)` specifying the transcripts set type. These were determined by Joshua M. Stolz et al, 2022. Here the names "cell_component", "top1500", and "standard" refer to models that were determined to be effective in removing degradation effects. The "standard" model involves taking the union of the top 1000 transcripts associated with degradation from the interaction model and the main effect model. The "top1500" model is the same as the "standard model except the union of the top 1500 genes associated with degradation is selected. The most effective of our models, "cell_component", involved deconvolution of the degradation matrix to determine the proportion of cell types within our studied tissue. These proportions were then added to our `model.matrix()` and the union of the top 1000 transcripts in the interaction model, the main effect model, and the cell proportions model were used to generate this model of qSVs.

Value

A `character()` with the transcript IDs.

Examples

```
## Default set of transcripts associated with degradation
sig_transcripts <- select_transcripts()
length(sig_transcripts)
head(sig_transcripts)

## Example where match.arg() auto-completes
select_transcripts("top")
```

`transcripts` *Transcripts for Degradation Models*

Description

An object storing three lists of transcripts each corresponding to a model used in the degradation experiment. These were determined by Joshua M. Stolz et al, 2022. Here the names "cell_component", "top1500", and "standard" refer to models that were determined to be effective in removing degradation effects. The "standard" model involves taking the union of the top 1000 transcripts associated with degradation from the interaction model and the main effect model. The "top1500" model is the same as the "standard" model except the union of the top 1500 genes associated with degradation is selected. The most effective of our models, "cell_component", involved deconvolution of the degradation matrix to determine the proportion of cell types within our studied tissue. These proportions were then added to our `model.matrix()` and the union of the top 1000 transcripts in the interaction model, the main effect model, and the cell proportions model were used to generate this model of qSVs.

Usage

```
transcripts
```

Format

A `list()` with character strings containing the transcripts selected by each model. Each string is a GENCODE transcript IDs.

See Also

select_transcripts

Index