Package ‘qsvaR’

November 1, 2025

Title Generate Quality Surrogate Variable Analysis for Degradation Correction

Version 1.14.0

Date 2025-09-29

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 7.3.2

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

Config/testthat/edition 3

Imports dplyr, sva, stats, ggplot2, rlang, methods

Depends R (>= 4.2), SummarizedExperiment

LazyData true

VignetteBuilder knitr

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

git_branch RELEASE_3_22

git_last_commit f960098

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2025-10-31

Author Joshua Stolz [aut] (ORCID: <https://orcid.org/0000-0001-5694-5247>),
Hedia Tnani [ctb] (ORCID: <https://orcid.org/0000-0002-0380-9740>),
Leonardo Collado-Torres [ctb] (ORCID:
<https://orcid.org/0000-0003-2140-308X>),
Nicholas J. Eagles [aut, cre] (ORCID:
<https://orcid.org/0000-0002-9808-5254>)

Maintainer Nicholas J. Eagles <`nickeagles77@gmail.com`>

Contents

`degradation_tstats` *Degradation time t-statistics*

Description

These t-statistics are derived from the degradation timepoints data built into qsvaR. They are the results from multiple models where we determined the association of transcripts with degradation time adjusting for 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 `rse_tx`.

Usage

```
DEqual(
  DE,
  deg_tstats = qsvaR::degradation_tstats,
  show.legend = TRUE,
  show.cor = c("caption", "corner-top", "corner-bottom", "none"),
  font.size = 12,
  cor.size = font.size/2,
  cor.label = "cor: "
)
```

Arguments

`DE`	a `data.frame()` with a column "t" containing the t-statistics from Differential Expression, typically generated with `limma::topTable()`. `rownames(DE)` must have transcript Ensembl/Gencode IDs.
`deg_tstats`	an optional `data.frame()` with a column "t" containing t-statistics resulted from a degradation experiment. Default is the internal `qsvaR::degradation_tstats` from the package authors.
`show.legend`	logical (default TRUE) to show legend in the plot
`show.cor`	specify where to show the correlation value. Can be one of "caption", "corner-top", "corner-bottom", or "none".
`font.size`	numeric value to set the base font size of the plot
`cor.size`	numeric (default font.size/2) to set the font size for the correlation text
`cor.label`	character (default "cor: ") to set the text preceding the correlation value

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,
  sig_transcripts = select_transcripts(),
  assayname = "tpm",
  verbose = TRUE
)
```

Arguments

`rse_tx`	A RangedSummarizedExperiment-class object containing the transcript data desired to be studied.
`sig_transcripts`	A `character()` vector of transcripts that should be associated with degradation, expected to be present in `rownames(rse_tx)`.
`assayname`	character string specifying the name of the assay desired in rse_tx
`verbose`	specify if the function should report how many model transcripts were matched

Value

A RangedSummarizedExperiment-class object.

Examples

```
degTx <- getDegTx(rse_tx)
```

`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(rse_tx, "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 <- getPCs(rse_tx, "tpm")
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
## rse_tx 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(rse_tx)
)

## 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(rse_tx, mod, "tpm")
```

`normalize_tx_names` *Remove version number from Gencode/Ensembl transcript names*

Description

This function removes the Gencode/ENSEMBL version from the transcript ID, while protecting _PAR_Y suffixes if present

Usage

```
normalize_tx_names(txnames)
```

Arguments

`txnames`	A `character()` vector of GENCODE or ENSEMBL transcript IDs

Value

A `character()` vector of transcript names without versioning

Examples

```
ensIDs <- normalize_tx_names(rownames(rse_tx))
```

`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, sig_transcripts = select_transcripts(), mod, assayname)
```

Arguments

`rse_tx`	A RangedSummarizedExperiment-class object containing the transcript data desired to be studied.
`sig_transcripts`	A `character()` vector of transcripts that should be associated with degradation, expected to be present in `rownames(rse_tx)`.
`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
## rse_tx 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(rse_tx)
)

## 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 = rse_tx, mod = mod, assayname = "tpm")
```

`rse_tx` *Example of RSE object with RNA-seq transcript quantification data*

Description

This data is a RangedSummarizedExperiment-class with transcript quantification data stored in an "tpm" assay. It is used to demonstrate the use of qsvaR in bulk RNA-seq data.

Format

A RangedSummarizedExperiment-class

See Also

getPCs k_qsvs getDegTx qSVA

`select_transcripts` *Select transcripts associated with degradation*

Description

Helper function to select which experimental model(s) will be used to generate the qSVs. Degradation-associated transcripts are derived in four different models (transcripts). The `cell_component` parameter controls whether the models with cell-type proportions are included. This function extract the top `top_n` transcripts found to be significant in each considered model, then returns the union of transcripts across all considered models. Up to 10,000 transcripts are available to select from each model prior to taking the union.

Usage

```
select_transcripts(top_n = 1000, cell_component = FALSE)
```

Arguments

`top_n`	An `integer(1)` specifying how many significant transcripts to extract from each model prior to taking a union across models.
`cell_component`	A `logical(1)`. If `FALSE`, only include transcripts from the main and interaction models (see `main_model` and `int_model` here: transcripts). If `TRUE`, additionally include main and interaction models that include cell-type proportions (a total of 4 models).

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)

## Select more transcripts if desired
length(select_transcripts(top_n = 5000))
```

`transcripts` *Transcripts for Degradation Models*

Description

This object is a list of four `tibbles` where each element corresponds to the top 10,000 transcripts (by significance) and their adjusted p-values for a given degradation model. The `main_model` model is a linear model modelling expression against a sample's degradation time, with brain region as a covariate. The `int_model` model is similar but includes an interaction term with degradation time and brain region. The `cell_main_model` and `cell_int_model` models are like the respective `main_model` and `int_model` models, but including cell-type fractions from deconvolution as linear terms.

Usage

```
transcripts
```

Format

A `list()` of `tibble()`s containing the transcripts and adjusted p-values selected by each model. Each string is a GENCODE transcript IDs.

See Also

select_transcripts

`which_tx_names` *Check validity of transcript vectors and return a vector matching indexes in tx1*

Description

This function is used to check if tx1 and tx2 are GENCODE or ENSEMBL transcript IDs and return an integer vector of tx1 transcript indexes that are in tx2.

Usage

```
which_tx_names(txnames, sig_tx)
```

Arguments

`txnames`	A `character()` vector of GENCODE or ENSEMBL transcript IDs.
`sig_tx`	A `character()` vector of GENCODE or ENSEMBL signature transcript IDs.

Value

A `integer()` vector of `txnames` transcript indexes in `sig_tx`.

Examples

```
sig_tx <- select_transcripts(cell_component = TRUE)
whichTx <- which_tx_names(rownames(rse_tx), sig_tx)
```

Index