Package 'saseR'

May 15, 2024

Type Package

Title Scalable Aberrant Splicing and Expression Retrieval

Date 2023-06-01

Version 1.0.0

Description saseR is a highly performant and fast framework for aberrant expression and splicing analyses. The main functions are:
\itemize{
\item \code{\link{BamtoAspliCounts}} - Process BAM files to ASpli counts
\item \code{\link{convertASpli}} -
Get gene, bin or junction counts from ASpli SummarizedExperiment
\item \code{\link{calculateOffsets}} -
Create an offsets assays for aberrant expression or splicing analysis
\item \code{\link{saseRfindEncodingDim}} -
Estimate the optimal number of latent factors to include when estimating the mean expression
\item \code{\link{saseRfit}} -
Parameter estimation of the negative binomial distribution and compute p-values for aberrant expression and splicing
}
For information upon how to use these functions, check out our vignette at \url{https://github.com/statOmics/saseR/blob/main/vignettes/Vignette.Rmd} and the saseR paper: Segers, A. et al. (2023). Juggling offsets unlocks RNA-seq tools for fast scalable differential usage, aberrant splicing and expression analyses. bioRxiv. \url{https://doi.org/10.1101/2023.06.29.547014}.

Depends R (>= 4.3.0)

Imports ASpli, S4Vectors, BiocGenerics, GenomicFeatures, MASS, PRROC, SummarizedExperiment, edgeR, pracma, precrec, BiocParallel, DESeq2, DEXSeq, data.table, limma, methods, GenomicRanges, GenomicAlignments, rrcov, MatrixGenerics, stats, IRanges, knitr, dplyr, igraph, parallel

VignetteBuilder knitr

License Artistic-2.0

URL https://github.com/statOmics/saseR, https://doi.org/10.1101/2023.06.29.547014

Encoding UTF-8

LazyData FALSE

RoxygenNote 7.2.3

biocViews DifferentialExpression, DifferentialSplicing, Regression, GeneExpression, AlternativeSplicing, RNASeq, Sequencing, Software

BugReports https://github.com/statOmics/saseR/issues

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

git_branch RELEASE_3_19

git_last_commit 83a2bb6

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-14

Author Alexandre Segers [aut, cre],
Jeroen Gilis [ctb],
Mattias Van Heetvelde [ctb],
Elfride De Baere [ctb],
Lieven Clement [ctb]

Maintainer Alexandre Segers <Alexandre.segers@ugent.be>

Contents

saseR-package *saseR: Scalable Aberrant Splicing and Expression Retrieval*

Description

saseR is a highly performant and fast framework for aberrant expression and splicing analyses. The main functions are:

- `BamtoAspliCounts` - Process BAM files to ASpli counts
- `convertASpli` - Get gene, bin or junction counts from ASpli SummarizedExperiment
- `calculateOffsets` - Create an offsets assays for aberrant expression or splicing analysis
- `saseRfindEncodingDim` - Estimate the optimal number of latent factors to include when estimating the mean expression
- `saseRfit` - Parameter estimation of the negative binomial distribution and compute p-values for aberrant expression and splicing

For information upon how to use these functions, check out our vignette at https://github.com/statOmics/saseR/blob/main/vignettes/Vignette.Rmd and the saseR paper: Segers, A. et al. (2023). Juggling offsets unlocks RNA-seq tools for fast scalable differential usage, aberrant splicing and expression analyses. bioRxiv. https://doi.org/10.1101/2023.06.29.547014.

Author(s)

Maintainer: Alexandre Segers <Alexandre.segers@ugent.be>

Other contributors:

- Jeroen Gilis <Jeroen.Gilis@ugent.be> [contributor]
- Mattias Van Heetvelde <Mattias.Vanheetvelde@ugent.be> [contributor]
- Elfride De Baere <Elfride.Debaere@ugent.be> [contributor]
- Lieven Clement <Lieven.Clement@ugent.be> [contributor]

See Also

Useful links:

- https://github.com/statOmics/saseR
- doi:10.1101/2023.06.29.547014
- Report bugs at https://github.com/statOmics/saseR/issues

ASpliSE *ASpliSE*

Description

SummarizedExperiment containing gene counts, bin counts and junction counts in the metadata slots, together with some rowData information. Output from BamtoAspliCounts.

Usage

```
data(saseRExample)
```

Value

Loads all data needed to run vignette and examples

BamtoAspliCounts *Converting BAM files to an ASpli-SummarizedExperiment, which contains gene, bin and junction counts.*

Description

BamtoAspliCounts is used to convert BAM files to a SummarizedExperiment that contains gene, bin and junction level counts in the metadata slot. This function is adapted from the ASpli package. More information can be found in the corresponding package and paper.

Usage

```
BamtoAspliCounts(
  features,
  targets,
  minReadLength = 100,
  maxISize = 50000,
  libType = "PE",
  strandMode = 0,
  minAnchor = 0,
  threshold = 5,
  BPPARAM = bpparam()
)
```

Arguments

features	An object of class ASpliFeatures, obtained by using the binGenome function of ASpli.
targets	A dataframe containing sample, bam and experimental factors columns.
minReadLength	Minimum read length of sequenced library. It is used for computing E1I and IE2 read summarization. Make sure this number is smaller than the maximum read length in every bam file, otherwise no E1I or IE2 will be found
maxISize	Maximum intron expected size. Junctions longer than this size will be dicarded.
libType	Defines how reads will be treated according their sequencing library type (paired (PE, default) or single end (SE)).
strandMode	controls the behavior of the strand getter. It indicates how the strand of a pair should be inferred from the strand of the first and last alignments in the pair. 0: strand of the pair is always *. 1: strand of the pair is strand of its first alignment. This mode should be used when the paired-end data was generated using one of the following stranded protocols: Directional Illumina (Ligation), Standard SOLiD. 2: strand of the pair is strand of its last alignment. This mode should be used when the paired-end data was generated using one of the following stranded protocols: dUTP, NSR, NNSR, Illumina stranded TruSeq PE protocol. For more information see ?strandMode
minAnchor	Minimum percentage of read that should be aligned to an exon-intron boundary.
threshold	Minimum number of reads supporting junctions. Default=5.
BPPARAM	object of class bpparamClass that specifies the back-end to be used for computations. See bpparam in BiocParallel package for details.

Value

A 'SummarizedExperiment' instance, containing 'gene', 'bin' and 'junction' counts in the metadata slot.

Examples

```
data(saseRExample, package = "saseR")

bamFileNames <- paste(rep(c("A", "B"), each = 6), rep(c("C",
 "D"), 2, each = 3), paste0(rep(0:2, 2), ".bam"), sep = "_")

BAMFiles <- system.file("extdata", bamFileNames, package = "saseR")

targets <- data.frame(
    row.names = paste0('Sample',c(1:12)),
    bam = BAMFiles,
    f1 = rep("A",12),
    stringsAsFactors = FALSE)

ASpliSE <- BamtoAspliCounts(
    features = features,
    targets = targets,
```

```
    minReadLength = 100,
    libType = "SE"
)
```

`calculateOffsets` *calculating the offsets to perform aberrant expression or splicing analysis.*

Description

calculateOffsets is used to calculate the offsets for aberrant expression or splicing analysis. it can use ordinary offsets such as 'edgeR' and 'DESeq2' offsets for aberrant expression analysis, or use the total counts aggregated per gene to perform aberrant splicing analysis. It returns a 'SummarizedExperiment' which includes an 'offsets' matrix in the assays slot.

Usage

```
calculateOffsets(
  se,
  method,
  aggregation = NULL,
  zeroCountOffsets = 1,
  zeroOffsets = 1,
  mergeGeneASpli = TRUE,
  filterna = TRUE,
  saveall = FALSE
)
```

Arguments

`se`	A 'SummarizedExperiment' instance generated with the SummarizedExperiment function of the SummarizedExperiment package. In the assay slot, provide the expression counts as an ordinary 'matrix', 'DataFrame', a 'sparseMatrix' or a 'DelayedMatrix'. When offsets are calculated for aberrant splicing ('AS') or other proportions ('proportion'), a rowData column is required which is used to group features over which the sum is taken as offset.
`method`	method used to calculate the offsets. 'TMM' uses edgeR's trimmed mean of M-values, 'geommean' uses DESeq2's size factors, 'unit' uses a unit matrix of offsets, 'AS' and 'proportion' use a rowData column to aggregate feature counts (e.g. calculating the total bin counts per gene) for aberrant splicing or other usage analyses.
`aggregation`	column of rowData used to aggregate feature counts. Only used when 'method' is 'AS' or 'proportion'. If not specified, the 'locus' column is used when available, followed by the 'symbol' column.

`zeroCountOffsets`
: When using the ‘AS‘ or ‘proportion‘ method, it is possible that some offsets are 0. As offsets need to be a strict positive value, these offsets are changed to a strict positive number (defined by ‘zeroOffsets‘, with a default of 1). The corresponding count can also be changed, with a default change to 1. Note that, although it is possible, it is not recommended to specify ‘zeroCountOffsets‘ greater than ‘zeroOffsets‘.

`zeroOffsets`
: When using the ‘AS‘ or ‘proportion‘ method, it is possible that some offsets are 0. As offsets need to be a strict positive value, these offsets are changed to a strict positive number (defined by ‘zeroOffsets‘, with a default of 1). The corresponding count can also be changed, with a default change to 1. Note that, although it is possible, it is not recommended to specify ‘zeroCountOffsets‘ greater than ‘zeroOffsets‘.

`mergeGeneASpli`
: logical value. When ASpli is used to obtain feature counts, one can aggregate feature counts to calculate offsets based on both gene-level or ASpli-cluster. When ‘mergeGeneASpli‘ is TRUE, one uses gene-level aggregation, except where no annotated gene is available, and use ASpli-cluster aggregation for these features.

`filterna`
: logical value to filter NA locus values in rowData. These are not necessarily features without annotated gene.

`saveall`
: logical value to save all aggregated counts and offsets. Default is FALSE to save memory. If TRUE, also counts and offsets of gene-level aggregation and ASpli-cluster aggregation are saved, instead of only using the final used counts and offsets.

Value

An updated ‘SummarizedExperiment‘ instance, now including an ‘offsets‘ matrix in the assays slot. When ‘method‘ is ‘AS‘ or ‘proportion‘, an ‘realCounts‘ matrix is available in the assays slot, which correspond to the original counts, while the ‘counts‘ matrix in the assays slot is updated with to ‘zeroCountOffsets‘ when the calculated offset is equal to 0. When ‘saveall‘ is TRUE, also matrices ‘Locuscounts‘, ‘Locusoffsets‘, ‘ASplicounts‘and ‘ASplioffsets‘ are saved in the assays slot, corresponding to the adapted counts and offsets using the ‘locus‘ rowData column, and using the ‘symbol‘ rowData column respectively.

Examples

```
data(saseRExample, package = "saseR")


SEgenes <- calculateOffsets(SEgenes, method = "TMM")
SEbins <- calculateOffsets(SEbins,
                           method = "AS",
                           aggregation = "locus")
SEjunctions <- calculateOffsets(SEjunctions,
                                method = "AS",
                                aggregation = "symbol",
                                mergeGeneASpli = TRUE)
```

convertASpli *Converting SummarizedExperiment with different ASpli-counts to gene, bin or junction level counts.*

Description

convertASpli is used to obtain gene, bin or junction level counts from a SummarizedExperiment object containing these three in the metadata slot.

Usage

```
convertASpli(ASpliSE, type = "none", filter = TRUE, ...)
```

Arguments

ASpliSE	An SummarizedExperiment object obtained by the 'BamtoAspliCounts' function, which contains gene, bin and junction level counts in the metadata slot.
type	character vector specifying the counts to be extracted. Can be 'gene', 'bin' and 'junction'.
filter	logical value specifying to filter bin counts based on the type of bin. Regions that contain both intronic and exonic regions are filtered as the intron and exon bins are already defined in other features.
...	extra parameters for filtering.

Value

A 'SummarizedExperiment' instance, representing the 'gene', 'bin' or 'junction' counts as specified.

Examples

```
data(saseRExample, package = "saseR")

SEgenes <- convertASpli(ASpliSE, type = "gene")
SEbins <- convertASpli(ASpliSE, type = "bin")
SEjunctions <- convertASpli(ASpliSE, type = "junction")
```

features *features*

Description

features used to count bam-files. These are obtained by binGenome from ASpli.

Usage

```
data(saseRExample)
```

Format

/

Value

Loads all data needed to run vignette and examples

Source

https://doi.org/10.1093/bioinformatics/btab141

References

Estefania Mancini, Andres Rabinovich, Javier Iserte, Marcelo Yanovsky, Ariel Chernomoretz, ASpli: integrative analysis of splicing landscapes through RNA-Seq assays, Bioinformatics, Volume 37, Issue 17, September 2021, Pages 2609–2616, https://doi.org/10.1093/bioinformatics/btab141

saseRExample *saseRExample*

Description

saseR example data, obtained from the ASpli package (https://bioconductor.org/packages/release/bioc/html/ASpli.html, 10.1093/bioinformatics/btab141).

Usage

```
data(saseRExample)
```

Value

Loads all data needed to run examples

`saseRfindEncodingDim` *Determine the optimal number of latent factors to detect outlier gene expression or splicing*

Description

saseRfindEncodingDim is used to search for the optimal number of latent factors included in the regression to search for aberrant expression or splicing. This optimal number of latent factors can be estimated by using the Gavish and Donoho threshold for singular values, or by using a denoising autoencoder, as described in our corresponding paper

Usage

```
saseRfindEncodingDim(
  se,
  method = "GD",
  analysis,
  dimensions = seq(2, min(100, ncol(se) - 2, nrow(se) - 1), 2),
  freq = 0.01,
  zScore = 3,
  sdlog = log(1.6),
  lnorm = TRUE,
  inj = "both",
  BPPARAM = bpparam(),
  aggregation,
  scale = TRUE,
  ...
)
```

Arguments

`se`	A 'SummarizedExperiment' instance generated with the SummarizedExperiment function of the SummarizedExperiment package. In the assay slot, provide the expression counts as an ordinary 'matrix', 'DataFrame', a 'sparseMatrix' or a 'DelayedMatrix'. 'colData' is a 'DataFrame' describing the samples in the experiment. Also, include 'offsets' in the assays slot, which can be calculated with the 'calculateOffsets' function. Finally, specify the experimental 'design' as a formula in the metadata slot. This formula must be based on the colData, and should be '~1' if no known covariates are included.
`method`	The method used to estimate the optimal number of latent factors included in the regression framework. Default is 'GD', which uses the Gavish and Donoho threshold for singular values. 'DAE' will use a Denoising autoencoder, but will require longer computation time without increased performance.
`analysis`	'AE' for aberrant expression analysis and 'AS' for aberrant splicing analysis. Used to insert corrupted counts when using the 'DAE' method.

dimensions	A vector containing the number of latent factors that are compared in the grid search for the optimal number of latent factors. Only used when using the 'DAE' method.
freq	the frequency of corrupted counts injected in the dateset. Only used when using the 'DAE' method.
zScore	The mean magnitude of the corrupted counts in the dataset. Only used when using the 'DAE' method.
sdlog	Standard deviation of the distribution of the corrupted counts on the log-scale. Only used when using the 'DAE' method and when lnorm is TRUE.
lnorm	If TRUE, the corrupted counts are simulated from a log-normal distribution with a mean of log(zScore) and standard deviation of sdlog. Only used when using the 'DAE' method.
inj	Injection of overexpression ('high'), underexpression ('low') or both ('both') types of corrupted counts. Only used when using the 'DAE' method.
BPPARAM	object of class bpparamClass that specifies the back-end to be used for computations. See bpparam in BiocParallel package for details.
aggregation	character vector representing the column in the rowData to be used to calculate offsets when injecting corrupted counts according to aberrant splicing. Only used when method is 'DAE' and when analysis is 'AE'.
scale	boolean. If TRUE, the deviance residuals upon will be scaled to mean 0 and sd = 1 before estimating the latent factors.
...	Extra arguments for .fitRUV.

Value

An updated 'SummarizedExperiment' instance, now including: 'optimalEncDim' in the metadata slot, representing the estimated optimal number of latent factors. 'LatentFactorControl' in the metadata slot, which represents the method used to estimate the optimal number of latent factors ('GD' for Gavish and Donoho threshold, 'DAE' for denoising autoencoder). 'deviances' in the assays slot when using the 'GD' method, representing the deviance residuals calculated using edgeR with an intercept and known covariates. 'svd_u', 'svd_d' and 'svd_v' matrices in the metadata slot when using the 'GD' method, which represent the singular value decomposition of the deviance residuals. 'encDimTable' a data.table in the metadata slot which represents the area under the curve to search for corrupted counts at the different dimensions when using the 'DAE' method.

Examples

```
data(saseRExample, package = "saseR")

SEgenes <- saseRfindEncodingDim(SEgenes, method = "GD")
SEbins <- saseRfindEncodingDim(SEbins, method = "GD")
SEjunctions <- saseRfindEncodingDim(SEjunctions, method = "GD")
```

saseRfit *Searching for aberrant expression or splicing.*

Description

saseRfit is used to perform a negative binomial regression and to search for aberrant expression or splicing in the context of rare Mendelian disorders. To model aberrant splicing, it uses adapted offsets to model proportions. It uses a predefined or an estimated optimal number of latent factors in the regression. It returns the p-values for the prioritisation of aberrant expression or splicing.

Usage

```
saseRfit(
  se,
  analysis,
  dimensions = NULL,
  padjust = "BH",
  BPPARAM = bpparam(),
  fit = "fast",
  scale = TRUE,
  robustPCA = FALSE,
  ignore_samples = NULL
)
```

Arguments

se	A 'SummarizedExperiment' instance generated with the SummarizedExperiment function of the SummarizedExperiment package. In the assay slot, provide the expression counts as an ordinary 'matrix', 'DataFrame', a 'sparseMatrix' or a 'DelayedMatrix'. 'colData' is a 'DataFrame' describing the samples in the experiment. Finally, specify the experimental design as a formula in the metadata slot. This formula must be based on the colData, or equal to ~1 in case of modelling only an intercept. If 'optimalEncDim' is available in the metadata slot, this number will be used as the number of latent factors.
analysis	'AE' for aberrant expression analysis, 'AS' for aberrant splicing analysis, and 'proportion' for any other kind of analysis using proportions.
dimensions	A single integer representing the number of latent factors that are used in the regression framework. If not specified, the 'optimalEncDim' in the metadata slot will be used. If nor 'optimalEncDim' is specified, the Gavish and Donoho threshold for singular values will be used.
padjust	the method used to compute FDR-adjusted p-values. Look at the documentation of the 'p.adjust' package for different options. Both Benjamini-Hochberg ('BH') or Benjamini-Yekutieli ('BY') are recommended.
BPPARAM	object of class bpparamClass that specifies the back-end to be used for computations. See bpparam in BiocParallel package for details.

fit	character value. Both 'fast' or 'edgeR' can be used. 'fast' uses a quadratic mean-variance relationship for fast estimation of the mean when many covariates or latent factors are included, without compromising upon power. 'edgeR' can be used to perform parameter estimation , although not scaling well with many covariates or latent factors.
scale	logical value to scale the deviance residuals before performing the singular value decomposition.
robustPCA	logical value to perform robust PCA using the 'PcaHubert' function, which decreases the influence of outlier values on the estimated latent factors.
ignore_samples	character vector with names of samples that are not used when performing the latent factor estimation and estimation of the regression coefficients.

Value

An updated 'SummarizedExperiment' instance, now including a matrix of p-values ('pValue') and a matrix of FDR-adjusted p-values ('pValueAdjust') in the assay slot. Also 'pValueAdjustMethod' is available in the metadata slot, stating which method was used to obtain FDR-adjusted pvalues. If analysis is 'AS' or 'proportion', also 'pValuesLocus', an aggregated p-value is provided based on the 'locus' column in the rowData. This is calculated by taking the minimal p-values.

Examples

```
data(saseRExample, package = "saseR")

SEgenes <- saseRfit(SEgenes,
                    analysis = "AE",
                    padjust = "BH",
                    fit = "fast")
SEbins <- saseRfit(SEbins,
                   analysis = "AS",
                   padjust = "BH",
                   fit = "fast")
SEjunctions <- saseRfit(SEjunctions,
                        analysis = "AS",
                        padjust = "BH",
                        fit = "fast")
```

SEbins *SEbins*

Description

SummarizedExperiment containing bin-level counts to perform an example aberrant splicing analysis.

Usage

```
data(saseRExample)
```

Value

Loads all data needed to run vignette and examples

`SEgenes` *SEgenes*

Description

SummarizedExperiment containing gene-level counts to perform an example aberrant expression analysis.

Usage

```
data(saseRExample)
```

Value

Loads all data needed to run vignette and examples

`SEjunctions` *SEjunctions*

Description

SummarizedExperiment containing junction-level counts to perform an example aberrant splicing analysis

Usage

```
data(saseRExample)
```

Value

Loads all data needed to run vignette and examples

Index