Package ‘FASE’

January 10, 2023

Type Package

Title RNA-Seq data analysis pipeline

Version 0.1.24

Author Harsh Sharma, Ravi Datta Sharma

Maintainer Harsh Sharma <harsh.sharma34@student.amity.edu>

Description Pipeline for RNA Sequencing data analysis and Alternative Splicing. FASE can be used for finding Cassette Exon/Intron Retention events, Structure and Concentration of different transcripts expressed in treated and normal condition(s). It also contains wrapper functions to find Differentially Expressed Genes and Differentially Expressed Junctions.

Encoding UTF-8

LazyData True

RoxygenNote 7.1.1

Citation Sharma, H., Pani, T., Dasgupta, U., Batra, J., and Sharma, R.D. (2023). Prediction of transcript structure and concentration using RNA-Seq data. Briefings in Bioinformatics. DOI: 10.1093/bib/bbad022

Imports Biobase, Matrix, RBGL, Rsubread, edgeR, graph, limma, matrixStats, BioPhysConnectoR, openxlsx, parallel, survival, survivalAnalysis, dplyr, igraph, purrr, stringr, MASS, methods, stats

Dependencies BioPhysConnectoR (R >= 3.6.2); Biobase (R >= 3.6.2); Matrix (R >= 3.6.2); RBGL (R >= 3.6.2); Rsubread (R >= 3.6.2); edgeR (R >= 3.6.2); graph (R >= 3.6.2); limma (R >= 3.6.2); matrixStats (R >= 3.6.2); parallel (R >= 3.6.2); openxlsx (R >= 3.6.2); survival (R >= 3.6.2); survivalAnalysis (R >= 3.6.2); dplyr (R >= 3.6.2); purrr (R >= 3.6.2); igraph (R >= 3.6.2); stringr (R >= 3.6.2); MASS (R >= 3.6.2), methods(R >= 3.6.2), stats(R >= 3.6.2)

NeedsCompilation no

R topics documented:

`.cmm` *Title*

Description

Title

Usage

```
.cmm(expression = expression, RMM = RMM, iMM = iMM)
```

Arguments

`expression`

`RMM`

`iMM`

`geneID`

`.correctJunctionCoordinate` *correctJunctionCoordinate*

Description

Internal function for `getJunctionCountMatrix`, not to be run separately.

Usage

```
.correctJunctionCoordinate(bed)
```

`.epip.exp` *Prepare files for hs and simscore*

Description

Prepare files for hs and simscore

Usage

```
.epip.exp(ep, ip, ep.exp, ip.exp)
```

Arguments

`ep`

`ip`

`ep.exp`

`exp`

.ePrnaseqFunction *EP function*

Description

Internal function of EPrnaseq, not to be called separately.

Usage

```
.ePrnaseqFunction(
  counts = y,
  RMM = RMmeber,
  contrastM = contrastM,
  designM = designM,
  Groups = Groups
)
```

.extractIntrons *extractIntrons*

Description

Internal function of intronGTFparser, not to be called separately.

Usage

```
.extractIntrons(sGTF)
```

.fitEPrnaseqModel *EP model fitting*

Description

Internal function of EPrnaseq, not to be called separately.

Usage

```
.fitEPrnaseqModel(gene, counts, RMmeber, contrastM, designM, Groups, threshold)
```

.fitiPrnaseqModel *fitiPrnaseqModel*

Description

Internal function of iPrnaseq, not to be called separately.

Usage

```
.fitiPrnaseqModel(gene, counts, iMMeber, contrastM, designM, Groups, threshold)
```

`.getEIJcounts` *.getEIJcounts*

Description

Internal function of `countMatrixGenes`, not to be called separately.

Usage

```
.getEIJcounts(Gannotation, GjunctionCount, GintronCount, GexonCount)
```

`.getNumericCount` *.getNumericCount*

Description

Internal function of `countMatrixGenes`, not to be run separately.

Usage

```
.getNumericCount(counts)
```

`.Gstructure` *Gene structure*

Description

Internal function for `readMembershipMatrix`, not to be called separately.

Usage

```
.Gstructure(sGTF, sJM)
```

`.iPrnaseqFunction` *iPrnaseqFunction*

Description

Internal function of `iPrnaseq`, not to be called separately.

Usage

```
.iPrnaseqFunction(
  counts = y,
  iMM = iMMeber,
  contrastM = contrastM,
  designM = designM,
  Groups = Groups
)
```

`.iStructure` *intron Structure*

Description

Internal function for `intronMembershipMatrix`, not to be called separately.

Usage

```
.iStructure(sGTF)
```

`.prepareCounts` *Prepare counts for ExonPointer and IntronPointer*

Description

Internal function for `EPrnaseq`/`iPrnaseq`, not to be run separately.

Usage

```
.prepareCounts(y, designM, Groups = NULL, threshold = NULL, ...)
```

References

Henrik Bengtsson (2017). matrixStats: Functions that Apply to Rows and Columns of Matrices (and to Vectors). R package version 0.52.2. https://github.com/HenrikBengtsson/matrixStats

`.propagation` *Transcript Structure: seed propagation function*

Description

Transcript Structure: seed propagation function

Usage

```
.propagation(
  cmm = cmm,
  ts.p = ts.p,
  cjmm = cjmm,
  se = se,
  s_exon = s_exon,
  event = event,
  eventtype,
  connected.exon = connected.exon,
  s.exon = NULL,
  keep.intron = FALSE
)
```

Arguments

`cmm`

`ts.p`

`cjmm`

`se`

`s_exon`

`event`

`eventtype`

`connected.exon`

`s.exon`

`keep.intron`

`.r.a.bias` *Removing algebraic bias*

Description

Removing algebraic bias

Usage

```
.r.a.bias(ts, tc)
```

`.r.dup.ts` *Removing duplicate transcript*

Description

Removing duplicate transcript

Usage

```
.r.dup.ts(transtruct)
```

Arguments

`transtruct`

.removeLECounts *Remove low-expressed reads*

Description

Internal function of EPrnaseq/iPrnaseq, not to be called separately.

Usage

```
.removeLECounts(y, designM, Groups = NULL, threshold = 6.32, ...)
```

References

Henrik Bengtsson (2017). matrixStats: Functions that Apply to Rows and Columns of Matrices (and to Vectors). R package version 0.52.2. https://github.com/HenrikBengtsson/matrixStats

.removeLECountsTS *Title*

Description

Title

Usage

```
.removeLECountsTS(
  Gcount = Gcount,
  designM = designM,
  Groups = Groups,
  threshold = 6.32,
  ...
)
```

Arguments

`Gcount`

`designM`

`Groups`

`threshold`

`...`

`.s.exon.ip.incl` *Title*

Description

Title

Usage

```
.s.exon.ip.incl(ip_event = ip_event, cmm = cmm, annotation = annotation)
```

Arguments

`ip_event`

`cmm`

`annotation`

`.seed.exon.ep.excl` *Title*

Description

Title

Usage

```
.seed.exon.ep.excl(ep_event = ep_event, cmm = cmm)
```

Arguments

`ep_event`

`cmm`

`.seed.exon.ep.incl` *Seed exons*

Description

Seed exons

Usage

```
.seed.exon.ep.incl(cmm = cmm, ep_event = ep_event, keep.intron = keep.intron)
```

Arguments

`cmm`

`ep_event`

`keep.intron`

.seed.exon.ip.excl *Title*

Description

Title

Usage

```
.seed.exon.ip.excl(ip_event = ip_event, cmm = cmm, annotation = annotation)
```

Arguments

ip_event

cmm

annotation

.sumPvalsMethod *Sum of p-values*

Description

Internal function of EPrnaseq/iPrnaseq, not to be called separately.

Usage

```
.sumPvalsMethod(x, n)
```

Arguments

x

.transtruct.prep *Transtruct preparation function*

Description

Transtruct preparation function

Usage

```
.transtruct.prep(
  cmm = cmm,
  event,
  eventlist,
  se = s.exon$se,
  ls_exon = s.exon$ls_exon,
  rs_exon = s.exon$rs_exon,
  eventtype,
  junc.mat = NULL,
  p = p,
  transtruct.ep.incl = NULL,
  transtruct.ep.excl = NULL,
  transtruct.ip.incl = NULL,
  transtruct.ip.excl = NULL,
  s.exon = NULL,
  keep.intron = keep.intron
)
```

Arguments

`cmm`

`event`

`eventlist`

`se`

`ls_exon`

`rs_exon`

`eventtype`

`junc.mat`

`p`

`transtruct.ep.incl`

`transtruct.ep.excl`

`transtruct.ip.incl`

`transtruct.ip.excl`

`s.exon`

`keep.intron`

`addAnnotationDEG` *Annotation of differentially expressed genes*

Description

Add gene annotation to ranked differentially expressed genes for a given contrast, using output of `DEG` function.

Usage

```
addAnnotationDEG(geneCount, fit, contrast)
```

Arguments

`geneCount`	summarized read counts of genes.
`fit`	output of `DEG` function that contains ranking of differentially expressed genes.
`contrast`	a contrast from contrast matrix, whose ranking is required.

Value

Annotated ranking of differentially expressed genes of given contrast. The output can be saved using write.csv or `write.xlsx`.

`addAnnotationDEJ` *Annotation of differentially expressed junctions*

Description

Add annotation to ranked differentially expressed junctions of given contrast returned by `DEJ`.

Usage

```
addAnnotationDEJ(JunctionMatrixA, fit, contrast)
```

Arguments

`JunctionMatrixA`	annotated junction matrix containing junction read counts, produced by `JunctionMatrixAnnota`
`fit`	output of `DEJ`, that contains ranking of differentially expressed junctions.
`contrast`	contrast from contrast matrix, whose ranking is required.

Value

Annotated ranking of differentially expressed junctions of a given contrast. The output can be saved using write.csv or `write.xlsx`.

`addAnnotationRnaSeq` *Add annotation to EP/IP events*

Description

Adds the associated information for each ranked cassette exon/intron retention event generated by `EPrnaseq`/`iPrnaseq`. The information includes location of the event (chromosome, start, stop, strand), position in genome and the associated gene name.

Usage

```
addAnnotationRnaSeq(fit, annotation = annotation)
```

Arguments

`fit`	output of `EPrnaseq`/`iPrnaseq`.
`annotation`	matrix; contains annotation of exons and introns, created by `readMembershipMatrix`.

Value

Annotated matrix of ranked cassette exon/intron retention events. Output of addAnnotationRnaSeq can be passed to `getPvaluesByContrast` to find differentially expressed intron retention events in a given contrast.

`connected.exons.igraph` *Connected Exons using igraph*

Description

Connected Exons using igraph

Usage

```
connected.exons.igraph(
  cmm = cmm,
  event = event,
  ls_exon = ls_exon,
  rs_exon = rs_exon,
  s.exon = NULL,
  eventtype = NULL
)
```

Arguments

cmm

event

ls_exon

rs_exon

s.exon

eventtype

---

countMatrixGenes *Count Matrix Genes*

Description

This function creates association of metafeatures such as exons, introns and junctions times sample for each gene. It requires a junction matrix, annotation matrix (generated by default using `readMembershipMatrix`) and summarized exon and intron read counts. It should be run only after running `readMembershipMatrix` and `intronMembershipMatrix`.

Usage

```
countMatrixGenes(
  JunctionMatrix,
  annotation,
  intronList = c(intron_A, intron_B, intron_C),
  exonList = c(out_A, out_B, out_C)
)
```

Arguments

`JunctionMatrix`
: matrix of junction read counts times samples, generated by `getJunctionCountMatrix`.

`annotation`
: gene features and meta-featues annotation file generated by `readMembershipMatrix`, saved by default as Annotation.Rdata.

`intronList`
: intron read counts per gene generated by Rsubread package and saved in counts_introns.Rdata file, as per preprocessing instructions.

`exonList`
: exon read counts per gene generated by Rsubread package and saved in counts_exons.Rdata file, as per preprocessing instructions.

Value

Gcount list contains gene-wise read count summarization of meta-features times samples in the study. The output is saved as Gcount.Rdata.

cpmCountsDEG *cpmCountsDEG*

Description

Generates read counts and log2cpm expression for differentially expressed genes for a given contrast, using output of `addAnnotationDEG` function.

Usage

```
cpmCountsDEG(
  geneCount,
  filename,
  designM = designM,
  contrastM = contrastM,
  Groups = Groups
)
```

Arguments

`geneCount`	summarized read counts of genes.
`filename`	filename in which output of addAnnotationDEG is saved.
`designM`	design matrix required by limma
`contrastM`	contrast matrix required by limma.
`Groups`	list of sample groups. Example: If there are two sample groups with three samples each, 'Groups' should be formed as: 1. numeric: c(1, 1, 1, 2, 2, 2)

Value

Read counts and log2cpm expression of given contrast for ranked differentially expressed genes.

References

1. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009)

cpmCountsDEJ *cpmCountsDEJ*

Description

Save read counts and log2cpm expression of differentially expressed junctions.

Usage

```
cpmCountsDEJ(
  JunctionMatrix,
  filename,
  designM = designM,
  contrastM = contrastM,
  Groups = Groups
)
```

Arguments

`JunctionMatrix`
: matrix containing read counts for junctions.

`filename`
: filename in which output of `addAnnotationDEJ` is saved.

`designM`
: design matrix required by limma

`contrastM`
: contrast matrix required by limma.

`Groups`
: list of sample groups.
Example: If there are two sample groups with three samples each, 'Groups' should be formed as:

  1. numeric: c(1, 1, 1, 2, 2, 2)

Value

Read counts and logcpm expression of given contrast.

References

1. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009)

`cpmCountsEP` *cpmCountsEP*

Description

This function requires ranking of cassette exon/intron retention events as generated by `getPvaluesByContrast` to generate raw read counts and log2cpm expression for the ranking of events in a given contrast only.

Usage

```
cpmCountsEP(filename, designM = designM, Groups, Gcount = Gcount)
```

Arguments

`filename`	file in which ExonPointer/IntronPointer ranking of a contrast is saved. *(Filename should be in csv format.)
`designM`	design matrix.
`Groups`	list of sample groups. Example: If there are two sample groups with three samples each, 'Groups' should be formed as: c(1, 1, 1, 2, 2, 2).
`Gcount`	list; contains gene-wise matrix of meta-features read counts times samples, generated by `countMatrixGenes`.

Value

Read counts and log2cpm expression of ranked cassette exon/intron retention events for the given contrast.

References

1. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009)

`DEG` *Differentially Expressed Genes*

Description

A wrapper function of limma package to find differentially expressed genes, given summarized read counts of genes obtained from `featureCounts` function or preprocessing intructions.

Usage

```
DEG(geneCount, designM = designM, contrastM = contrastM, Groups = Groups)
```

Arguments

`geneCount`	summarized read counts of genes.
`designM`	design matrix required by limma.
`contrastM`	contrast matrix required by limma.
`Groups`	list of sample groups. Example: If there are two sample groups with three samples each, 'Groups' should be formed as: 1. numeric: c(1, 1, 1, 2, 2, 2)

Value

Saves raw gene counts and log2cpm expression for all genes. Meta-data generated through this function is saved in fit2.Rdata file. Further, annotation of this meta-data is performed by `addAnnotationDEG` function. Contrast-wise ranking of annotated differentially expressed genes can be obtained using `cpmCountsDEG` function.

References

1. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009)

`DEJ` *Differentially Expressed Junctions*

Description

Find differentially expressed junctions, provided a junction matrix as input (generated by `getJunctionCountMatri`
This function uses standard limma package (`eBayes`) to find differentially expressed junctions.

Usage

```
DEJ(JunctionMatrix, designM = designM, contrastM = contrastM, Groups = Groups)
```

Arguments

`JunctionMatrix`	matrix containing read counts for junctions, obtained using `getJunctionCountMatrix` .
`designM`	design matrix required by limma
`contrastM`	contrast matrix required by limma.
`Groups`	list of sample groups. Example: If there are two sample groups with three samples each, 'Groups' should be formed as: 1. numeric: c(1, 1, 1, 2, 2, 2)

Value

The output is ranked differentially expressed junctions. Meta-data is saved as fit2.Rdata in folderSRA directory. The ranking can be annotated using `addAnnotationDEJ`. The annotated and ranked differentially expressed junctions for a given contrast (as given in contrast matrix) can be saved using `cpmCountsDEJ`

References

1. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009)

EPrnaseq *ExonPointer*

Description

Prediction of cassette exons events by utilizing information of meta-features (flanking junctions, skipping junctions and introns) associated with the exon in context ofs a given gene.

Usage

```
EPrnaseq(
  Gcount,
  RMM,
  designM,
  contrastM,
  Groups = NULL,
  p = 1,
  threshold = 3,
  annotation = annotation,
  ...
)
```

Arguments

`Gcount`	list; contains gene-wise matrix of meta-features read counts times samples, generated by `countMatrixGenes`.
`RMM`	gene-wise list that represents the association of exons with other meta-features of genes (introns and junctions (skipping/flanking)). It is generated using `readMembershipMatri`
`designM`	design matrix required by limma.
`contrastM`	contrast matrix required by limma.
`Groups`	list of sample groups. Example: If there are two sample groups with three samples each, 'Groups' should be formed as: 1. numeric: c(1, 1, 1, 2, 2, 2) 2. alphabetical: c('A', 'A', 'A', 'B', 'B', 'B')
`p`	number of threads to be used if running in parallel. (default=1)
`threshold`	minimum number of reads that should map to a meta-feature (default=3). If number of reads<threshold, meta-feature would be discarded.
`annotation`	matrix; contains annotation of exons and introns, created using `readMembershipMatrix`.
`...`	other parameters to be passed to `eBayes`, `voom`, `calcNormFactors` and `lmFit` .

Details

ExonPointer algorithm finds cassette exon events using metafeatures (exons, introns and junctions). The read counts of meta-features are present in Gcount and the association of an exon with introns and junctions (skipping/flanking) is given by Read Membership Matrix (RMM).
In order to find a cassette exon event, one-tailed p-values of metafeatures are summarized using

Irwin-Hall method to find the equivalent P-value (EqP). EqP determines if an event is differentially alternatively spliced. For more details, please refer: S. S. Tabrez, R. D. Sharma, V. Jain, A. A. Siddiqui & A. Mukhopadhyay. Differential alternative splicing coupled to nonsense-mediated decay of mRNA ensures dietary restriction-induced longevity. Nature Communications volume 8, Article number: 306 (2017).

Value

ExonPointer gives a list of ranked cassette exon events with equivalent p-value and t-statistic.

References

1. S. S. Tabrez, R. D. Sharma, V. Jain, A. A. Siddiqui & A. Mukhopadhyay. Differential alternative splicing coupled to nonsense-mediated decay of mRNA ensures dietary restriction-induced longevity. Nature Communications volume 8, Article number: 306 (2017).
2. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009).
3. Ritchie, M. E., Phipson, B., Wu, D., Hu, Y., Law, C. W., Shi, W., & Smyth, G. K. limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic acids research, 43(7), e47 (2015).
4. Henrik Bengtsson (2017). matrixStats: Functions that Apply to Rows and Columns of Matrices (and to Vectors). R package version 0.52.2. https://github.com/HenrikBengtsson/matrixStats
5. https://git.bioconductor.org/packages/Biobase
6. Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, Bravo HC, Davis S, Gatto L, Girke T, Gottardo R, Hahne F, Hansen KD, Irizarry RA, Lawrence M, Love MI, MacDonald J, Obenchain V, Ole's AK, Pag'es H, Reyes A, Shannon P, Smyth GK, Tenenbaum D, Waldron L, Morgan M (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." Nature Methods, 12(2), 115–121.
7. https://CRAN.R-project.org/view=HighPerformanceComputing

`FASE` *FASE: Finding Alternative Splicing Events (0.1.24)*

Description

RNA-Seq data analysis pipeline.

Author(s)

Harsh Sharma and Dr. Ravi Datta Sharma

`getJunctionCountMatrix` *Generate junction count matrix*

Description

This function combines tophat2 pipeline output junctions.bed files after mapping reads to genome/trancriptome. It can be called separately for combining junction.bed files for the FASE pipeline.

Usage

```
getJunctionCountMatrix(files)
```

Arguments

`files`	junction bed files.

Value

Junction read counts matrix.

References

1. F. Hoffgaard, P. Weil, K. Hamacher. BioPhysConnectoR: Connecting Sequence Information and Biophysical Models. BMC Bioinformatics volume 11, Article number: 199 (2010).

`getPvaluesByContrast` *Cassette exon/intron retention event ranking by contrast*

Description

Takes the annotated and fitted object of `EPrnaseq`/`iPrnaseq` and the name or number of contrast as given in contrast matrix as input and finds differentially alternatively spliced cassette exon/intron retention events for that contrast.

Usage

```
getPvaluesByContrast(fit, contrast = NULL)
```

Arguments

`fit`	output of addAnnotationRnaSeq function.
`contrast`	contrast whose ranking is required, for example, 'NormalvsTumor' (as used in contrast matrix).

Value

Data frame that contains ranking of cassette exon/intron retention events in the given contrast or comparision with their annotation. The output can be saved as csv/xlsx file.

GTFnomencJunctionM *Junction Matrix Annotation*

Description

This function is used for correcting the annotation of junction matrix in case junction.bed files are obtained from "BAM" files using "regtools" software. This is the case where "BAM" files are downloaded some repository instead of running tophat2 pipeline. This function then produces a correctly annotated junction matrix on the basis of chromosome nomenclature as used in standard "GTF" file.

Usage

```
GTFnomencJunctionM(gtf, JunctionMatrix)
```

Arguments

`gtf`	gtf file of the organism.
`JunctionMatrix`	matrix with read counts of junctions.

Value

Annotated junction matrix file: JunctionMatix.

heatscore *Heatscore using Splice Index.*

Description

heatscore uses expression of ranked EP and IP events

Usage

```
heatscore(ep, ip, ep.exp, ip.exp)
```

Arguments

`ep`	ExonPointer file containing cassette exon event ranking.
`ip`	IntronPointer file containing intron retention event ranking.
`ep.exp`	ExonPointer cassete exon expression file.
`ip.exp`	IntronPointer intron retention expression file.

Value

Heat Score/Splice Index required by HotNet software.

`intronGTFparser` *intronGTFparser*

Description

Parse intron location given in a gtf file and updated gtf will be written. Intron information can be used then for counting reads with Rsubread package (check wrapper functions: `ppAuto` and `ppSumEIG` for read count summarization). However, information associated with these introns (related to transcripts) can not be used as annotation since this transcript information is added in the corresponding field to avoid unnecessary errors.

Usage

```
intronGTFparser(gtf)
```

Arguments

`gtf`	gtf file of the organism.

Value

gtf file with intron information.

References

1. http://Matrix.R-forge.R-project.org/
2. Carey V, Long L, Gentleman R (2019). RBGL: An interface to the BOOST graph library. https://bioconductor.org/packages/RBGL/
3. Gentleman R, Whalen E, Huber W, Falcon S (2019). graph: graph: A package to handle graph data structures. http://www.bioconductor.org/packages/release/bioc/html/graph.html

---

`intronMembershipMatrix`
Intron Membership Matrix

Description

iMM describes association of each intron with meta-features (exons, skipping junctions and flanking junctions) of that gene. It can be generated using a gtf file and a combined junction matrix generated via `getJunctionCountMatrix`. iMM is a pre-requisite matrix for running `iPrnaseq`. It should be run only after running `readMembershipMatrix`.

Usage

```
intronMembershipMatrix(verbose = TRUE, annotation = annotation)
```

Arguments

`verbose`	TRUE
`annotation`	matrix; contains annotation of exons and introns, created using `readMembershipMatrix`.

Value

intronMembershipMatrix creates gene-wise list which is saved by default as iMM.Rdata. Each gene is represented by a matrix of meta-features times the number of introns in gene.
A number is assigned for each meta-feature association to introns in the gene as:

- 0 : No association
- 1 : Exon associated with the intron
- 2 : Intron with itself
- 3 : Junction associated with the intron

References

1. F. Hoffgaard, P. Weil, K. Hamacher. BioPhysConnectoR: Connecting Sequence Information and Biophysical Models. BMC Bioinformatics volume 11, Article number: 199 (2010).

---

iPrnaseq *Intron Pointer*

Description

Prediction of intron retention events by utilizing information of meta-features (flanking junctions, skipping junctions and introns) associated with the intron in context for a given gene.

Usage

```
iPrnaseq(
  Gcount,
  iMM,
  designM,
  contrastM,
  Groups = NULL,
  p = 1,
  threshold = 3,
  annotation = annotation,
  ...
)
```

Arguments

Gcount	list; contains gene-wise matrix of meta-features read counts times samples, generated by countMatrixGenes.
iMM	gene-wise list that represents the association of intron with other meta-features of genes (exons and junctions (skipping/flanking)). It is generated using intronMembershipMat
designM	design matrix required by limma.
contrastM	contrast matrix required by limma.
Groups	list of sample groups. Example: If there are two sample groups with three samples each, 'Groups' should be formed as:

1. numeric: c(1, 1, 1, 2, 2, 2)
2. alphabetical: c('A', 'A', 'A', 'B', 'B', 'B')

`p` number of threads to be used if running in parallel. (default=1)

`threshold` minimum number of reads that should map to a meta-feature (default=3). If number of reads<threshold, meta-feature would be discarded.

`annotation` matrix; contains annotation of exons and introns, created using `readMembershipMatrix`.

`...` other parameters to be passed to `eBayes`, `voom`, `calcNormFactors` and `lmFit` .

Details

IntronPointer algorithm finds intron retention events using metafeatures (exons, introns and junctions). The read counts of meta-features are present in Gcount and the association of an intron with exons and junctions is given by Intron Membership Matrix (iMM).
In order to find an intron retention event, one-tailed p-values of metafeatures are summarized using Irwin-Hall method to find the equivalent P-value (EqP). EqP determines if an event is differentially alternatively spliced.For more details, please refer: S. S. Tabrez, R. D. Sharma, V. Jain, A. A. Siddiqui & A. Mukhopadhyay. Differential alternative splicing coupled to nonsense-mediated decay of mRNA ensures dietary restriction-induced longevity. Nature Communications volume 8, Article number: 306 (2017).

Value

IntronPointer gives a list of ranked intron retention events with equivalent p-value and t-statistics. The output of iPrnaseq can be passed to `addAnnotationRnaSeq` to add annotation to the ranked intron retention events.

References

1. S. S. Tabrez, R. D. Sharma, V. Jain, A. A. Siddiqui & A. Mukhopadhyay. Differential alternative splicing coupled to nonsense-mediated decay of mRNA ensures dietary restriction-induced longevity. Nature Communications volume 8, Article number: 306 (2017).
2. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140 (2009).
3. Ritchie, M. E., Phipson, B., Wu, D., Hu, Y., Law, C. W., Shi, W., & Smyth, G. K. limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic acids research, 43(7), e47 (2015).
4. Henrik Bengtsson (2017). matrixStats: Functions that Apply to Rows and Columns of Matrices (and to Vectors). R package version 0.52.2. https://github.com/HenrikBengtsson/matrixStats
5. https://git.bioconductor.org/packages/Biobase
6. Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, Bravo HC, Davis S, Gatto L, Girke T, Gottardo R, Hahne F, Hansen KD, Irizarry RA, Lawrence M, Love MI, MacDonald J, Obenchain V, Ole's AK, Pag'es H, Reyes A, Shannon P, Smyth GK, Tenenbaum D, Waldron L, Morgan M (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." Nature Methods, 12(2), 115–121.
7. https://CRAN.R-project.org/view=HighPerformanceComputing

JunctionMatrixAnnotation *Junction Matrix Annotation*

Description

Annotation of junction matrix using gtf file.

Usage

```
JunctionMatrixAnnotation(gtf, JunctionMatrix)
```

Arguments

`gtf`	gtf file of the organism.
`JunctionMatrix`	matrix containing junction read counts.

Value

Annotated junction matrix file: JunctionMatixA.

ppAuto *RNA Seq and Alternative Splicing preprocessing function*

Description

ppAuto is a wrapper function for several tools and functions that perform preprocessing of RNA Sequencing data. This function performs preprocessing that includes mapping of reads, sorting and indexing of bam files, to summarization of read counts for exons, introns, genes and junctions. ppAuto also creates several prerequisite matrices including junction matrix, ReadMembershipMatrix (RMM), IntronMembershipMatrix (iMM) and Gcount matrix in order to run ExonPointer and IntronPointer algorithms.

System requirements for ppAuto include:

1. fastq-dump (if files='SRA')
2. tophat2
3. samtools

Usage

```
ppAuto(
  folderSRA = FALSE,
  srlist = NULL,
  pairedend = FALSE,
  genomeBI,
  gtf,
  files = "fastq",
  p = 1,
  N = 6,
  r = 44,
  mate_std_dev = 30,
  read_edit_dist = 6,
  max_intron_length = 10000,
  min_intron_length = 50,
  segment_length = NULL,
  ...
)
```

Arguments

`folderSRA`	path of directory containing fastq or SRA files. (default=current directory)
`srlist`	list of unique sample names of fastq/SRA files created by default in the function. Please follow naming convention for the sample files: For SRA files : "Sample-S1_1" "Sample-S1_2" (for paired-end reads) and "Sample-S1" (for single-end reads). For fastq files: "Sample-S1_1.fastq" "Sample-S1_2.fastq" (for paired-end reads) and "Sample-S1.fastq" (for single-end reads).
`pairedend`	boolean, TRUE if reads are paired-end and FALSE if reads are single-end. All files should be either single-end or paired-end. (default=FALSE)
`genomeBI`	path of genome build of the organism created using bowtie2-build command.
`gtf`	intron parsed gtf file of the organism. Please check `intronGTFparser` to generate intron parsed gtf file (to generate intron read counts).
`files`	type of raw read file: fastq or sra (downloaded from NCBI). All files should be in same format and have same read length. (default=fastq)
`p`	number of threads to be utilized by samtools and Rsubread package. (default=1)
`N`	accepted read mismatches. Reads with more than N mismatches are discarded. (default=6) [tophat2 parameter]
`r`	expected inner distance between mate pair. (default=44) [tophat2 parameter]
`mate_std_dev`	the standard deviation for the distribution on inner distances between mate pairs. (default=30) [tophat2 parameter]
`read_edit_dist`	final read alignments having more than these many edit distance are discarded. (default=6) [tophat2 parameter]
`max_intron_length`	when searching for junctions ab initio, TopHat2 will ignore donor/acceptor pairs farther than this many bases apart, except when such a pair is supported by a split segment alignment of a long read. (default=10000) [tophat2 parameter]

`min_intron_length`
: topHat2 will ignore donor/acceptor pairs closer than this many bases apart. (default=50) [tophat2 parameter]

`segment_length`
: each read is divided into this length and mapped independently to find junctions. [tophat2 parameter]

`...`
: other parameter to be passed to tophat2.

Value

1. Mapped, sorted and indexed bam files. (Can be run separately using tophat2 and samtools or wrapper function: `ppRawData`)
2. Lists of gene counts, exon counts and intron counts saved in folderSRA directory as respective Rdata files. (Can be run separately using `featureCounts` or wrapper function: `ppSumEIG`)
3. Junction Matrix: Matrix with annotated junction count reads. (Can be run separately using `getJunctionCountMatrix` or wrapper function: `ppRawData`)
4. RMM : ReadMembershipMatrix. (Can be run separately using `readMembershipMatrix` or wrapper function: `ppFASE`)
5. iMM : intronMembershipMatrix. (Can be run separately using `intronMembershipMatrix` or wrapper function: `ppFASE`)
6. Gcount : A list of gene-wise read count summarization of meta-features times samples in the study. (Can be run separately using `countMatrixGenes` or wrapper function: `ppFASE`)

References

1. Liao Y, Smyth GK, Shi W. The R package Rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads. Nucleic Acids Research, 47, e47 (2019).

`ppFASE` *Alternative Splicing Pre-processing*

Description

Alternative Splicing preprocessing function. This function creates several prerequisite matrices for related to meta-features:junction matrix (by combining output of tophat2: junction.bed), ReadMembershipMatrix (RMM), IntronMembershipMatrix (IMM) and Gcount matrix in order to run ExonPointer and IntronPointer algorithms. `ppFASE` should be run only after tophat2 (or its wrapper function: `ppRawData`) has mapped all the raw read files and the reads have been summarized using `featureCounts` (or its wrapper function: `ppSumEIG`).

Usage

```
ppFASE(
  folderSRA = FALSE,
  gtf = gtf,
  exonCount = exonCount,
  intronCount = intronCount,
  JunctionMatrix = JunctionMatrix
)
```

Arguments

`folderSRA`	directory containing fastq or SRA files.
`gtf`	intron parsed gtf file of the organism.
`JunctionMatrix`	junction count matrix. If ppRawData has been run, JunctionMatrix is saved in JunctionCounts.Rdata.
`exonCounts`	list of summarized exon counts. If ppSumEIG has been run, exonCounts are saved in counts_exons.Rdata.
`intronCounts`	list of summarized intron counts. If ppSumEIG has been run, intronCounts are saved in counts_introns.Rdata.

Value

1. Junction Matrix: Matrix with Junction count reads and their annotation. (Can be run separately using `getJunctionCountMatrix`)
2. RMM : ReadMembershipMatrix. (Can be run separately using `readMembershipMatrix` or wrapper function: `ppAuto`)
3. iMM : intronMembershipMatrix. (Can be run separately using `intronMembershipMatrix` or wrapper function: `ppAuto`)
4. Gcount : A list of gene-wise read count summarization of meta-features times samples in the study. (Can be run separately using `countMatrixGenes` or wrapper function: `ppAuto`)

References

Liao Y., Smyth G.K., Shi W. The R package Rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads. Nucleic Acids Research, 47, e47 (2019).

`ppRawData` *RNA Sequencing raw data preprocessing*

Description

Manual function to map reads with the reference genome, given SRA/fastq files. It also sorts and indexes the mapped reads for further processing. Reads produced by ppRawData can be summarized for genes, exons and introns using `ppSumEIG`. `ppAuto` is not required if ppRawData has been called.

System requirements for ppRawData include:

1. fastq-dump (if files='SRA')
2. tophat2
3. samtools

Usage

```
ppRawData(
  folderSRA = FALSE,
  srlist = NULL,
  pairedend = FALSE,
  genomeBI,
  files = "fastq",
  p = 1,
  N = 6,
  r = 44,
  mate_std_dev = 30,
  read_edit_dist = 6,
  max_intron_length = 10000,
  min_intron_length = 50,
  segment_length = NULL,
  ...
)
```

Arguments

`folderSRA`	path of directory containing fastq or SRA files. (default=current directory)
`srlist`	list of unique sample names of fastq/SRA files created by default in the function. Please follow naming convention for the sample files: For SRA files : "Sample-S1_1" "Sample-S1_2" (for paired-end reads) and "Sample-S1" (for single-end reads). For fastq files: "Sample-S1_1.fastq" "Sample-S1_2.fastq" (for paired-end reads) and "Sample-S1.fastq" (for single-end reads).
`pairedend`	boolean, TRUE if reads are paired-end and FALSE if reads are single-end. All files should be either single-end or paired-end. (default=FALSE)
`genomeBI`	path of genome build of the organism created using bowtie2-build command.
`files`	type of raw read file: fastq or sra (downloaded from NCBI). All files should be in same format and have same read length. (default=fastq)
`p`	number of threads to be utilized by samtools and Rsubread package. (default=1)
`N`	accepted read mismatches. Reads with more than N mismatches are discarded. (default=6) [tophat2 parameter]
`r`	expected inner distance between mate pair. (default=44) [tophat2 parameter]
`mate_std_dev`	the standard deviation for the distribution on inner distances between mate pairs. (default=30) [tophat2 parameter]
`read_edit_dist`	final read alignments having more than these many edit distance are discarded. (default=6) [tophat2 parameter]
`max_intron_length`	when searching for junctions ab initio, TopHat2 will ignore donor/acceptor pairs farther than this many bases apart, except when such a pair is supported by a split segment alignment of a long read. (default=10000) [tophat2 parameter]
`min_intron_length`	topHat2 will ignore donor/acceptor pairs closer than this many bases apart. (default=50) [tophat2 parameter]

segment_length
: each read is divided into this length and mapped independently to find junctions. [tophat2 parameter]

...
: other parameters to be passed to tophat2.

gtf
: intron parsed gtf file of the organism. Please check `intronGTFparser` to generate intron parsed gtf file (to generate intron read counts).

Value

1. Mapped, sorted and indexed bam files. (Can be run separately using tophat2 and samtools or automatic wrapper function: `ppAuto`)
2. Junction Matrix: Matrix with junction count reads. (Can be run separately using `getJunctionCountMatrix` or wrapper function: `ppAuto`)

References

1. https://CRAN.R-project.org/view=HighPerformanceComputing
2. Sequence Read Archive Submissions Staff. Using the SRA Toolkit to convert .sra files into other formats. In: SRA Knowledge Base [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2011-. Available from: https://www.ncbi.nlm.nih.gov/books/NBK158900/.
3. https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=toolkit_doc&f=fastq-dump
4. Kim D, Pertea G, Trapnell C, Pimentel H, Kelley R, Salzberg SL. TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. Genome Biol. 25;14(4):R36 (2013 Apr). http://ccb.jhu.edu/software/tophat.
5. Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R, and 1000 Genome Project Data Processing Subgroup, The Sequence alignment/map (SAM) format and SAMtools, Bioinformatics (2009) 25(16) 2078-9.

---

ppSumEIG *RNA Seq Preprocessing Read Summarization*

Description

ppSumEIG is a manual wrapper function that provides summarization of read counts for exons, introns and genes using `featureCounts`. The gtf file passed to this function should first be passed to `intronGTFparser` to find the location of introns. Reads used for summarization by ppSumEIG should already be mapped, sorted and index using tophat2 and samtools or their wrapper function: `ppRawData`. The summarized counts produced by ppSumEIG can be further processed using `ppFASE`, which produces several matrices required by ExonPointer and IntronPointer algorithms for finding alternative splicing events. ppSumEIG need not be run if `ppAuto` has already been run.

Usage

```
ppSumEIG(
  folderSRA = FALSE,
  pairedend = FALSE,
  p = 1,
  gtf = gtf,
  srlist = NULL,
  ...
)
```

Arguments

`folderSRA`	path of directory containing aligned and indexed bam file folders. (default=current directory)
`pairedend`	boolean, TRUE if reads are paired-end and FALSE if reads are single-end. (default=FALSE).
`p`	number of threads to be utilized by Rsubread package. (default=1)
`gtf`	intron parsed gtf file of the organism.
`...`	other parameters to be passed to `featureCounts`.

Value

Lists of gene counts, exon counts and intron counts saved in folderSRA directory as respective Rdata files. (Can be run separately using `featureCounts` or automatic wrapper function for entire pre-processing: `ppAuto`)

References

1. Liao Y, Smyth GK, Shi W. The R package Rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads. Nucleic Acids Research, 47, e47 (2019).

`readMembershipMatrix`

Read Membership Matrix

Description

RMM describes association of each exon with meta-features (introns, skipping junctions and flanking junctions) of that gene. It can be generated using a gtf file and a combined junction matrix generated via `getJunctionCountMatrix`. RMM is a pre-requisite matrix for running `EPrnaseq`.

Usage

```
readMembershipMatrix(gtf, JunctionMatrix = JunctionMatrix)
```

Arguments

`gtf`	gtf file of the organism.
`JunctionMatrix`	junction matrix contains read counts of each junction mapped by tophat2 alongwith their annotation.

Value

readMembershipMatrix creates a gene-wise list which is saved by default as RMM.Rdata. Each gene is represented by a matrix of meta-features times the number of exons in gene.
A number is assigned for each meta-feature association to exons in the gene as:

- 0 : No association

- 0.5: Skipping junction to the exon
- 1 : Exon with itself
- 2 : Flanking junction to the exon
- 3 : Intron associated with the exon

References

1. F. Hoffgaard, P. Weil, K. Hamacher. BioPhysConnectoR: Connecting Sequence Information and Biophysical Models. BMC Bioinformatics volume 11, Article number: 199 (2010).

`simscore` *Similarity Score for network edge weight*

Description

Similarity Score for network edge weight

Usage

```
simscore(ep, ep.exp, ip, ip.exp)
```

Arguments

`ep`	ExonPointer file containing cassette exon event ranking.
`ep.exp`	ExonPointer cassete exon expression file.
`ip`	IntronPointer file containing intron retention event ranking.
`ip.exp`	IntronPointer intron retention expression file.

Value

Similarity Score gives cosine similarity between the splice index of two genes. It can be used as edge weight for an unweighted network.

`survFASE` *Survival analysis using metafeatures*

Description

survFASE finds survival rate of patients using altenative splicing data. It requires exon, intron and junction expression of the samples generated using FASE (check `EPrnaseq`, `iPrnaseq` and `DEJ`). rownames of all expression files and clinical data should have the same identifier/sample ID, otherwise survFASE would not be able to perform survival analysis. survFASE uses RMM/iMM to find metafeature(s) associated with the given exonID/intronID and incorporate the expression of those metafeatures with respective exonID/intronID.

Usage

```
survFASE(
  Time = Time,
  Status = Status,
  Gcount,
  clinical.data,
  rmm = NULL,
  imm = NULL,
  eventID,
  threshold = 0.6,
  designM,
  Groups
)
```

Arguments

`Time`	column name of survival time in clinical data. (default = Time)
`Status`	column name of survival status (alive/dead status) in clinical data. 1 = dead and 0 = alive. (default = Status)
`Gcount`	Gene count matrix for the given gene. It should contain raw meta-feature counts of only patients. Rownames of gcount should be unique sample IDs that can be mapped to the clinical data.
`clinical.data`	clinical data of patients. It should contain survival time (days to last follow up) and survival status (0/1) of the patients. Rownames of clinical data should be unique, non-repeating sample IDs that can be mapped to the Gcount.
`rmm`	readMembershipMatrix matrix of the gene. It contains association between exons and other metafeatures in a gene and is generated by default as RMM.Rdata using `readMembershipMatrix` function. RMM is required only for finding survival rate associated with a cassette exon event.
`imm`	intronMembershipMatrix matrix of the gene. It contains association between introns and other metafeatures in a gene. It is generated by default as iMM.Rdata using `intronMembershipMatrix` function. iMM is required only for finding survival rate associated with an intron retention event.
`eventID`	exonID or intronID of the AS event for survival analysis.
`designM`	design matrix
`Groups`	list of sample groups

Value

survFASE returns an overall p-value, concordance index and Cox-PH statistics. The overall p-value suggests whether or not the given exon/intron significantly affects patient survival. C-index signifies goodness-of-fit of the model. Cox-PH results show which of the metafeatures associated with the exon/intron affect survival rate and their statistical inferences like hazard-ratio, beta-coefficient, etc.

`transconc` *Transcript Concentration*

Description

Transcript concentration gives the abundance of transcripts expressed in different samples and conditions. It uses transcript structure generated by transtruct method and superimposes expression values of corresponding meta-features.

Usage

```
transconc(transtruct, designM)
```

Arguments

`transtruct`	transcript structure for EP/IP inclusion and exclusion for a gene.
`designM`	design matrix for the two conditions in question. It should be a binary matrix where 1 represents samples of the condition and 0 represents the samples of other condition.

Value

transconc returns three matrices:

1. transconc.samples: transcript concentration by samples.
2. transconc.condition: transcript concentration by condition.
3. transconc.TS: structures of transcripts and concentration of their meta-features, used for evaluating transcript abundance (transconc combines all structures by transtruct and retains only unique structures).

`transtruct` *Transcript Stucture*

Description

transtruct uses alternative splicing information to find structures of transcripts generated in two conditions (usually control and treated). It can generate stuctures corresponding to multiple AS events in a gene at a time. These can be passed as a list or vector to ep.event or ip.event parameter. transtruct primarily requires RMM and iMM of the gene to find seed exons and extend them to form continuous paths of exons connected via flanking junctions.

Usage

```
transtruct(
  ep.event = NULL,
  ip.event = NULL,
  Gcount = Gcount,
  RMM,
  iMM = NULL,
  designM,
  annotation,
  Groups,
  keep.intron = FALSE
)
```

Arguments

`ep.event`	list/vector of Cassette Exon event(s), for which Transcript Structure is required. (default = NULL)
`ip.event`	list/vector of Intron Retention event(s), for which Transcript Structure is required. (default = NULL)
`Gcount`	Gcount matrix of given geneID. Gcount matrix generated via `countMatrixGenes`/`ppAuto`/`ppFA` as Gcount.Rdata. It contains gene-wise read count summarization of meta-features times samples in the study.
`RMM`	RMM of given geneID. readMembershipMatrix contains association of each exon with meta-features (exons, introns, skipping junctions and flanking junctions) of that gene. It is generated by `readMembershipMatrix` function as RMM.Rdata. It is required for both Cassette Exon as well as Intron Retention event.
`iMM`	iMM of given geneID. intronMembershipMatrix contains association of each intron with meta-features (introns, exons and skipping junctions) of that gene. It is generated by `intronMembershipMatrix` function as iMM.Rdata. It is required only when Transcript Structure for Intron Retention event is required.
`designM`	design matrix required by edgeR. Example: If there are four samples, two corresponding each to control and treated condition, design matrix should be prepared as: designM <- matrix(c(rep(1,2), rep(0,4), rep(1,2)), byrow = T, ncol=2, dimnames=list(c('Sample1', 'Sample2', 'Sample3', 'Sample4'), c('Normal', 'Treated')))
`annotation`	annotation matrix of given geneID. annotation is generated by `readMembershipMatrix`/`ppFASE` function as Annotation.Rdata. It contains gene-wise annotation of each meta-feature (exons, introns and junctions).
`keep.intron`	if a cassette exon event is selected by ExonPointer due to flanking introns, instead of or along with flanking junction(s), retain intron in the transcript structure and propagate structure using that intron. logical. (default = FALSE)
`geneID`	geneID of the Cassette Exon and/or Intron Retention event(s). Only one geneID can be passed.

Value

list of three matrices:

1. numeric: transtruct.condition: It contains transcript structure(s) of given gene for corresponding Cassette Exon and/or Intron Retention event(s) (as specified by rownames), in the treated samples.
2. numeric: transtruct.normal: It contains transcript structure(s) of given gene for corresponding Cassette Exon and/or Intron Retention event(s) (as specified by rownames), in the normal/untreated samples.
3. numeric: expression: it contains expression values of meta-features in the given gene

Index