Package 'RiboDiPA'

April 18, 2024

Type Package

Title Differential pattern analysis for Ribo-seq data

Date 2024-03-19

Version 1.11.1

Description This package performs differential pattern analysis for Ribo-seq data. It identifies genes with significantly different patterns in the ribosome footprint between two conditions. RiboDiPA contains five major components including bam file processing, P-site mapping, data binning, differential pattern analysis and footprint visualization.

License LGPL (>= 3)

Encoding UTF-8

LazyData true

Imports Rcpp (>= 1.0.2), graphics, stats, data.table, elitism, methods, S4Vectors, IRanges, GenomicRanges, matrixStats, reldist, doParallel, foreach, parallel, qvalue, DESeq2, ggplot2, BiocFileCache, BiocGenerics, txdbmaker

LinkingTo Rcpp

Depends R (>= 4.1), Rsamtools, GenomicFeatures, GenomicAlignments

NeedsCompilation yes

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

biocViews RiboSeq, GeneExpression, GeneRegulation, DifferentialExpression, Sequencing, Coverage, Alignment, RNASeq, ImmunoOncology, QualityControl, DataImport, Software, Normalization

Author Keren Li [aut], Matt Hope [aut], Xiaozhong Wang [aut], Ji-Ping Wang [aut, cre]

Maintainer Ji-Ping Wang <jzwang@northwestern.edu>

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

git_branch devel

git_last_commit e475b56

git_last_commit_date 2024-03-19

Repository Bioconductor 3.19

Date/Publication 2024-04-17

Contents

binTrack 2
bpTrack 3
data.binned 4
data.psite 5
dataBinning 6
diffPatternTest 7
diffPatternTestExon 9
exonTrack 10
normFactor 11
plotTest 12
plotTrack 12
psiteCal 13
psiteMapping 14
result.exon 15
result.pst 16
RiboDiPA 17

Index **20**

`binTrack` *Visualization: generating tracks for* `igvR` *(per bin)*

Description

This function outputs a list of `GRanges` format objects of binned ribosome P-site footprints, as well as test results. It can be used for `igvR` visualization.

Usage

```
binTrack(data, exon.anno)
```

Arguments

`data`	Data object from `diffPatterbTest` function or wrapper function `RiboDiPA`.
`exon.anno`	`exons` value from `psiteMapping` function. Genomic start and end coordinates, and other gene information is included. See `psiteMapping` for details.

Details

R package igvR is not implemented in RiboDiPA. Users need install igvR through Bioconductor or relevant source package. A simple illustration example of how to use it for igvR visualization is given below.

Value

A list of GRanges format objects. Each element is a GRanges object of the binned P-site footprint tracks with differential pattern test results.

Examples

```
data(result.pst)
data(data.psite)
tracks.bin <- binTrack(data = result.pst,
    exon.anno = data.psite$exons)

# library(igvR)
# thred <- 0.05
# igv <- igvR()
# setBrowserWindowTitle(igv, "ribosome binned track example")
# setGenome(igv, "saccer3")

# for(track.name in names(tracks.bin)){
#     track.rep <- tracks.bin[[track.name]]
#     resize(track.rep, width(track.rep) + 1)
#     track <- GRangesQuantitativeTrack(trackName = paste(track.name,"binned"),
#         track.rep[,1], color = "black")
#     displayTrack(igv, track)
# }
# track.rep2 <- tracks.bin[[1]]
# sig.bin <- (values(track.rep2)[,5] <= thred)
# log10.padj <- - log10(values(track.rep2)[,5])
# mcols(track.rep2) <- data.frame(log10padj = log10.padj)
# track.rep2 <- track.rep2[which(sig.bin),]
# track <- GRangesQuantitativeTrack(trackName = "- log 10 of padj",
#     track.rep2, color = "red", trackHeight = 40)
# displayTrack(igv, track)
```

---

bpTrack *Visualization: generating tracks for* igvR *(per bp)*

Description

This function outputs a list of GRanges format objects of ribosome P-site footprints per bp. It can be used for igvR visualization.

Usage

```
bpTrack(data, names.rep = NULL, genes.list)
```

Arguments

data	Data object from psiteMapping function or wrapper function RiboDiPA.
names.rep	Customized names of the replicates. Default value uses the column names of data$counts.
genes.list	A list of genes for visualization. Default value uses all genes listed in data$coverage

Details

R package igvR is not implemented in RiboDiPA. Users need install igvR through Bioconductor or relevant source package. A simple illustration example of how to use it for igvR visualization is given below.

Value

A list of GRanges format objects. Each element is a GRanges object of the P-site footprint tracks of selected genes.

Examples

```
data(data.psite)
names.rep <- c("mutant1", "mutant2", "wildtype1", "wildtype2")
tracks.bp <- bpTrack(data = data.psite, names.rep = names.rep,
    genes.list = c("YDR050C", "YDR062W", "YDR064W"))

# library(igvR)
# igv <- igvR()
# setBrowserWindowTitle(igv, "ribosome footprint track example")
# setGenome(igv, "saccer3")
# for(track.name in names.rep){
#     track.rep <- tracks.bp[[track.name]]
#     track <- GRangesQuantitativeTrack(trackName = paste(track.name, "bp"),
#         track.rep[,1], color = "green")
#     displayTrack(igv, track)
# }
```

---

data.binned *An example of binned P-sites data*

Description

A example data containing binned ribosome P-site tracks of 4 replicates on 885 genes, two biological replicates each for wild type cells and New1 mutant cells, respectively. It is the output of the data binning function dataBinning on P-site coverage data, and input for diffPatternTest function for differential pattern analysis.

Usage

```
data("data.binned")
```

Format

A list of 885 matrices corresponding to 885 genes: in each matrix, rows correspond to replicates, columns correspond to bins. Bin names are set to "start-end" genomic coordinates.

Source

The raw data was adapted from Kasari et al 2019.

Examples

```
data(data.binned)
classlabel <- data.frame(condition = c("mutant", "mutant",
    "wildtype", "wildtype"), comparison = c(2, 2, 1, 1))
rownames(classlabel) <- c("mutant1", "mutant2", "wildtype1", "wildtype2")
result.pst <- diffPatternTest(data = data.binned,
    classlabel = classlabel, method = c('gtxr', 'qvalue'))
```

---

data.psite *An example of P-site coverage data*

Description

An example data set containing 4 ribo-seq replicates of 885 genes, two biological replicates each for wild type cells and New1 mutant cells, respectively. It is the output of P-site mapping funcion `psiteMapping`. It contains the p-site count at each location of the total transcript within each replicate.

Usage

```
data("data.psite")
```

Format

A list of size 4

coverage ribosome P-site coverage tracks

counts ribosome P-site total count, one count per gene

psite.mapping P-site mapping offset rule

exons relative start and end positions of each exon in the total transcript if a given gene, as well as genomic start and end coordinates

Source

Raw data was adapted from Kasari et al 2019.

Examples

```
data(data.psite)
data.binned <- dataBinning(data = data.psite$coverage, bin.width = 0,
    zero.omit = FALSE, bin.from.5UTR = TRUE, cores = 2)
```

`dataBinning` *Data binning*

Description

This function bins a mapped P-site data matrix for a given gene into a binned matrix, for statistical testing downstream. Data can be adaptively binned, where each gene has a different number of bins and bin widths, but the bin positions for a given gene are the same across different conditions and replicates. Alternatively, data can also be binned into bins of fixed width, down to the single-codon level.

Usage

```
dataBinning(data, bin.width = 0, zero.omit = FALSE,
    bin.from.5UTR = TRUE, cores = NULL)
```

Arguments

`data`	A list of mapped P-site position matrices from the `coverage` object of the `psiteMapping` function. In each element of the list, rows correspond to replicates, while columns correspond to nucleotides across the total transcript.
`bin.width`	Binning width per bin. If specified, it is the number of codons merged per bin; if not specified, an adaptive binning width method is used.
`zero.omit`	If the `zero.omit` argument is set to `TRUE`, bins with zero mapped P-site counts across all replicates are removed from the differential pattern analysis.
`bin.from.5UTR`	When the coding region length is not any integer multiple of binning width, and if value of `bin.from.5UTR` is set to `TRUE`, the uneven width bins will be arranged at the 3' end of the total transcript. If set to `FALSE`, binning will proceed from the 3' end.
`cores`	The number of cores to use for parallel execution. If not specified, the number of cores is set to the value of `detectCores(logical = FALSE)`.

Details

We recommend to use an adaptive bin width h following the Freedman-Diaconis rule,

$$h = 2 * IQR/m^(1/3)$$

. To see certain regions of transcripts in greater detail (e.g. near the start and stop codons), a specified `bin.width` per bin can be used to check the local differential pattern, though it may lead to low power at small fold change positions and potentially high computational time.

Value

A list of binned P-site footprint matrices: in each matrix, rows correspond to replicates, columns correspond to bins. Bin names are set to "start-end" genomic coordinates.

See Also

psiteMapping

Examples

```
data(data.psite)
data.binned <- dataBinning(data = data.psite$coverage, bin.width = 0,
    zero.omit = FALSE, bin.from.5UTR = TRUE, cores = 2)
data.codon <- dataBinning(data = data.psite$coverage, bin.width = 1,
    zero.omit = FALSE, bin.from.5UTR = TRUE, cores = 2)
```

---

diffPatternTest *Differential pattern analysis of Ribo-seq data*

Description

The normalized gene data are pooled into a large matrix, where parameter estimations and tests are performed. Within each gene, multiplicity correction are then performed for codon/bin-level p-values. The minimum of adjusted codon/bin-level p-value is defined to be the gene-level p-value.

Usage

```
diffPatternTest(data, classlabel, method = c('gtxr', 'qvalue'))
```

Arguments

data	A list of named matrices input from the dataBinning function. In each element of the list, rows correspond to replicates, columns correspond to bins.
classlabel	For matrix input: a DataFrame or data.frame with at least a column comparison. In comparison, 1s stand for the reference condition, 2s stand for the target condtion, and 0s represent replicates is not invloved in the test, if present. Rows of classlabel correspond to rows of data, which are biological replicates.
method	For a 2-component character vector input: the first argument is the multiplicity correction method for codon/bin-level p-value adjustment. The second argument is the multiplicity correction method for gene-level p-value adjustment. Methods include: "qvalue" for q-value from qvalue pacakge, "gtxr", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY","fdr", "none" from the elitism package.

Details

Using binned data, this function first estimates normalizing constant by exclusing outlier bins which may represent the true differential pattern. An outlier bin is defined as that whose log2-fold change value is more than 1.5 interquartile ranges below the first quartile or above the third quartile. For a given gene, the normalizing constant is defined based on the total read counts from each replicate.

It then performs differential pattern testing on P-site counts bin by bin for each gene. Briefly, counts are modeled by a negative binomial distribution to call bins with statistically significant differences across conditions, bin level p-values are adjusted for multiple hypothesis testing for a given gene, and then the smallest p-value for a gene is adjusted to control for multiple hypothesis testing across all genes.

Additionally, the T-value is a supplementary statistic that quantifies the magnitude of difference between conditions, with larger numbers indicating a greater difference. The T-value is defined to be 1-cosine of the angle between the first right singular vectors of the footprint matrices of the two conditions under comparison. It ranges from 0-1, with larger values representing larger differences between conditions, and practically speaking, can be used to identify genes with larger magnitude of pattern difference beyond statistical significance. This might be helpful to investigators to prioritize certain genes for investigation among many that may pass the significance test for differential pattern.

Value

`bin`	A List object of codon/bin-level results. Each element of list is of a gene, containing codon/bin results columns: `pvalue`, `log2FoldChange`, and the adjusted p-value named by the first string in `method`. Names of Bins are set to "start-end" genomic coordinates.
`gene`	A DataFrame object of gene-level results. It contains columns: `tvalue`, `pvalue`, and the adjusted p-value named by the second string in `method`.
`method`	The same as input `method`.
`small`	Names of genes without sufficient reads, not reported in `bin` and `gene`.
`data`	Subset of input `data`, including all genes reported in `bin` and `gene`.
`classlabel`	The same as input `classlabel`.

See Also

`p.adjust`

Examples

```
data(data.binned)
classlabel <- data.frame(condition = c("mutant", "mutant",
    "wildtype", "wildtype"), comparison = c(2, 2, 1, 1))
rownames(classlabel) <- c("mutant1", "mutant2", "wildtype1", "wildtype2")
result.pst <- diffPatternTest(data = data.binned,
    classlabel = classlabel, method = c('gtxr', 'qvalue'))
```

`diffPatternTestExon` *Main function for differential pattern analysis of exon-binned Ribo-seq data*

Description

An alternative version of `diffPatternTest` for exon level binning. Both data binning and differential pattern analysis are implemented. Instead of a fixed width or adaptive method, the positions of exons in the genome are used as bins. Therefore the number of exons per gene and their relative sizes determines the bins used for differential pattern testing.

Usage

```
diffPatternTestExon(psitemap, classlabel,
    method = c("gtxr", "qvalue"))
```

Arguments

`psitemap`	A list object from value of `psiteMapping` function. In `psitemap`, list elements `coverage` and `exons` are required.
`classlabel`	For matrix input: a DataFrame or data.frame with at least a column `comparison`. In `comparison`, `1`s stand for the reference condition, `2`s stand for target condtion, `0`s represent replicates not invloved in the test, if present. Rows of `classlabel` correspond to rows of `data`.
`method`	For a 2-component character vector input: the first argument is the multiplicity correction method for exon-level p-value adjustment. The second argument is the multiplicity correction method for gene-level p-value adjustment. Methods include: "qvalue" for q-value from `qvalue` pacakge, "gtxr", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY","fdr", "none" from `elitism` package.

Details

For mammalian species, when the reads are sparse, it's more meaningful to perform a exon level pattern analysis. `diffPatternTestExon()` provides the option of exon level pattern differentiation analysis by treating each exon as one bin. But for organisms such as yeast, as most genes only contain one exon, the exon-level analysis is not meaningful since the analysis will simply result in the RNA-seq type of analysis, i.e. differential abundance test instead of the pattern analysis. Using `diffPatternTestExon()` on yeast data is not for organisms with minimal alternative splicing or multiple exons. For a given gene, the normalizing constant is estimated at codon level.

Value

`bin`	A List object of exon-level results. Each element of list is of a gene, containing exon results columns: `pvalue`, `log2FoldChange`, and the adjusted p-value named by the first string in `method`.
`gene`	A DataFrame object of gene-level results. It contains columns: `tvalue`, `pvalue`, and the adjusted p-value named by the second string in `method`.

small	Names of genes without sufficient reads
classlabel	The same as input classlabel.
data	A list of exon-binned P-site footprint matrices: in each matrix, rows corrspond to replicates, columns corrspond to exons. All genes reported in bin and gene are included.
method	The same as input method.

See Also

diffPatternTest

Examples

```
data(data.psite)
classlabel <- data.frame(condition = c("mutant", "mutant",
    "wildtype", "wildtype"), comparison=c(2, 2, 1, 1))
rownames(classlabel) <- c("mutant1", "mutant2", "wildtype1", "wildtype2")
result.exon <- diffPatternTestExon(psitemap = data.psite,
    classlabel = classlabel, method = c('gtxr', 'qvalue'))
```

---

exonTrack *Visualization: generating tracks for* igvR *(per exon)*

Description

This function outputs a list of GRanges format objects of ribosome P-site footprints per exon, as well as corresponding test results. It can be used for igvR visualization.

Usage

```
exonTrack(data, gene)
```

Arguments

data	Data object from diffPatterbTestExon function or wrapper function RiboDiPA.
gene	One gene for visualization at a time, since different genes have different number of transcripts.

Details

R package igvR is not implemented in RiboDiPA. Users need install igvR through Bioconductor or relevant source package. A simple illustration example of how to use it for igvR visualization is given below.

Value

A list of lists. Each element is a list of GRanges objects representing replicates. Each second level list element is P-site footprint count per exon with differential pattern test results in a transcript.

Examples

```
data(result.exon)
tracks.exon <- exonTrack(data = result.exon, gene = "tY(GUA)D")

# library(igvR)
# igv <- igvR()
# setBrowserWindowTitle(igv, "ribosome footprint per exon example")
# setGenome(igv, "saccer3")

# for(track.name in names(tracks.exon)){
#     track.rep <- tracks.exon[[track.name]]
#     for(tx.name in names(track.rep)){
#         track.tx <- tracks.exon[[track.name]][[tx.name]]
#         track <- GRangesQuantitativeTrack(trackName =
#             paste(track.name, tx.name), track.tx[,1], color = track.name)
#         displayTrack(igv, track)
#     }
# }
```

`normFactor` *Size factors for normalization by sequencing depth*

Description

This function calculate the relative abundance of samples, in essence accounting for different sequencing depths across different Ribo-seq experiments.

Usage

```
normFactor(x, condition)
```

Arguments

`x`	A matrix of mapped P-site positions.
`condition`	A vector of indicators. `1`'s stand for reference condition, `2`'s stand for target condtion, `0`'s represent replicates not invloved in the abundance estimation, if present.

Value

A vector of relative abundances.

Examples

```
data(data.binned)
x <- data.binned$YDR050C
condition <- c(2, 2, 1, 1)
normFactor(x, condition)
```

plotTest *Plotting ribosome footprint data from mapped P-sites at the bin level.*

Description

This function visualizes the ribosome footprint in the form of mapped P-site frequency at the bin level along the total transcript. Bins that test positive for statistically significant differences are marked in black. Plotting is implmented with the ggplot2 package.

Usage

```
plotTest(result, genes.list = NULL, threshold = 0.05)
```

Arguments

result	Data object resulting from diffPatternTest or diffPatternTestExon functions or wrapper function RiboDiPA.
genes.list	genes.list is the list of genes for visualization. If genes.list is not specified, then only genes with significant differential patterns specified by q-value threshold will be plotted. If genes.list is not NULL, then threshold argument will be ignored.
threshold	The q-value threshold for genes whose footprint to be visualized. This argument is ignored if genes.list is not NULL.

Value

Bin-level tracks of genes and test results.

Examples

```
data(result.pst)
plotTest(result = result.pst, genes.list = NULL, threshold = 0.05)
```

plotTrack *Plotting ribosome footprint data at the mapped P-site level*

Description

This function visualizes the ribosome footprint in the form of P-site frequency at the per nucleotide level along the total transcript. Plotting is implemented with the ggplot2 package.

Usage

```
plotTrack(data, genes.list, replicates = NULL,exons = FALSE)
```

Arguments

data	Data object from psiteMapping function or wrapper function RiboDiPA.
genes.list	A list of genes for visualization.
replicates	Names of the replicates for which the footprint to visualize. The default is for all.
exons	If value is TRUE, Ribo-seq footprints per exon of specified genes are also output.

Value

Visualizes the Ribo-seq per nucleotide footprint on merged exons of the genes and replicates specified. If exons is TRUE, Ribo-seq footprint per exon of specified genes is also output.

Examples

```
data(data.psite)
plotTrack(data = data.psite, genes.list = c("YDR050C", "YDR064W"),
    replicates = NULL, exons = FALSE)
```

---

psiteCal *P-site position of reads*

Description

This function outputs the P-site position, provided the CIGAR string of the alignment and the start position of read. It is implemented as a C++ function using the Rcpp package.

Usage

```
psiteCal(cigar, start, psitemap)
```

Arguments

cigar	A vector of CIGAR strings.
start	A vector of read start positions.
psitemap	A vector of relative P-site positions, which describe the offset from the 5 prime most nucleotide of the read to the P-site.

Value

A vector of P-site positions for the reads.

Examples

```
ex.cigar <- c("21M74731N7M", "2S11M57302N12M3S", "28M", "27M1S")
ex.start <- c(177640, 249163, 249286, 249290)
ex.psitemap <- c(9, 18, 9, 9)
psiteCal(ex.cigar, ex.start, ex.psitemap)
```

psiteMapping *P-site mapping*

Description

This function computes the corresponding P-site locations of all RPFs and the total RPF read counts for all genes and samples.

Usage

```
psiteMapping(bam_file_list, gtf_file, psite.mapping = "auto",
    cores = NULL)
```

Arguments

bam_file_list	A vector of bam file names to be tested. Users should include path names if not located in the current working directory. Index files (.bai) will be generated if not already present.
gtf_file	Annotation file used to generate the BAM alignments. Note that a GTF file sourced from one organization (e.g. Ensembl) cannot be used with BAM files aligned with a GTF file sourced from another organization (e.g. UCSC).
psite.mapping	Rules for P-site offsets, to map a given read length of RPF to a P-site. Input for this parameter is a string input, or a user defined matrix assigning the P-site mapping rules. Strings include: "center" for taking center of the read as the P-site, and "auto" for an optimal P-site offset, which is the default. A user defined matrix should include two columns: "qwidth" and "psite", where "qwidth" is the range of possible read lengths and "psite" is the corresponding offset from the 5' end to map to the P-site.
cores	The number of cores to use for parallel execution. If not specified, the number of cores is set to the value of `detectCores(logical = FALSE)`.

Details

All exons from the same gene are concatenated into a total transcript in order to get a merged picture of translation, using the reduce() function from the GRanges package to accomplish the concatenation. Then, RPFs are mapped with respect to the total transcript and the P-site positions are inferred accordingly.

If 'psite.mapping' is unspecified, a two-step algorithm on start codons of CDS regions is used to compute optimal P-site offsets, following Lauria et al (2018). First, for a given read length, the offset is calculated by taking the distance between the first nucleotide of the start codon and the 5' most nucleotide of the read, and then defining the offset as the 5' position with the most reads mapped to it. This process is repeated for all read lengths and then the temporary global offset is defined to be the offset of the read length with the maximum count. Lastly, for each read length, the adjusted offset is defined to be the one corresponding to the local maximum found in the profiles of the start codons closest to the temporary global offset.

The function will return a list of matrices that can then be used for data binning and downstream analysis, among other data objects.

Value

coverage	A list object of matrices. Each element is a matrix representing the P-site footprints of a gene. Rows correspond to replicates and columns corrspond to nucleotide location with respect to the total transcript. Each column is named by its genomic coordinate.
counts	A matrix object of read counts. Rows correspond to genes and columns correspond to replicates.
psite.mapping	The P-site (or A-site) mapping rule used to map RPFs to P-site positions.
exons	A List object of matrices. Each element contains the relative start and end positions of exons in the gene with respect to the total transcript for that given gene, as well as genomic start and end coordinates.

References

Lauria, F., Tebaldi, T., Bernabò, P., Groen, E., Gillingwater, T. H., & Viero, G. (2018). riboWaltz: Optimization of ribosome P-site positioning in ribosome profiling data. PLoS computational biology, 14(8), e1006169.

Examples

```
library(BiocFileCache)
file_names <- c("WT1.bam", "WT2.bam", "MUT1.bam", "MUT2.bam", "eg.gtf")
url <- "https://github.com/jipingw/RiboDiPA-data/raw/master/"
bfc <- BiocFileCache()
bam_path <- bfcrpath(bfc,paste0(url,file_names))

classlabel <- data.frame(
    condition = c("mutant", "mutant", "wildtype", "wildtype"),
    comparison = c(2, 2, 1, 1)
)
rownames(classlabel) <- c("mutant1","mutant2","wildtype1","wildtype2")

data.psite <- psiteMapping(bam_file_list = bam_path[1:4],
    gtf_file = bam_path[5], psite.mapping = "auto", cores = 2)
```

---

result.exon *An example of exon-level differential pattern analysis result*

Description

An example output generated by the differential pattern analysis function `diffPatternTestExon`, including counts per exon, exon-level differential pattern results, etc.

Usage

```
data("result.exon")
```

Format

A list of 5

bin A list object of exon-level test results. Each element of the list is the result from a gene, containing columns: `pvalue`, `log2FoldChange`, the adjusted p-value by method "gtxr", and other exon genomic information.

gene Gene-level differential pattern results, including T-value, p-value, and q-value

method See `diffPatternTestExon`

exon Counts per exon, binned from `data`, for the final differential pattern analysis in the format of a list of named matrices. In each element of the list, rows correspond to replicates, columns correspond to exons.

data The ribosome P-site coverage tracks in the format of a list of named matrices. In each element of the list, rows correspond to replicates, columns correspond to bps.

classlabel See `diffPatternTestExon`

Source

The data was adapted from Kasari et al 2019.

Examples

```
data(result.exon)
tracks.exon <- exonTrack(data = result.exon, gene = "YDR050C")
```

---

result.pst *An example of differential pattern analysis result*

Description

An example output generated by the differential pattern analysis function `diffPatternTest`, including binned data, differential pattern results, etc.

Usage

```
data("result.pst")
```

Format

A list of size 5

bin A list object of codon/bin-level test results. Each element of the list is the result from a gene, containing columns: `pvalue`, `log2FoldChange`, and the adjusted p-value by method "gtxr"

gene Gene-level differential pattern results, including T-value, p-value, and q-value

classlabel See `diffPatternTest`

data The input data for differential pattern analysis in the format of a list of named matrices. In each element of the list, rows correspond to replicates,columns correspond to bins.

method See `diffPatternTest`.

small Names of genes without sufficient reads, not reported in `bin` and `gene`.

Source

The data was adapted from Kasari et al 2019.

Examples

```
data(result.pst)
plotTrack(data = data.psite, genes.list = c("YDR050C", "YDR064W"),
    replicates = NULL, exons = FALSE)
```

RiboDiPA *A wrapper function for the RiboDiPA pipeline*

Description

A wrapper function for the RiboDiPA pipeline, that will call PsiteMapping, DataBinning, and DPTest in order. This function is provided for users' convenience and requires BAM files (one per biological replicate), a GTF file, and a classlabel object describing what comparisons to make. The minimal output from the function is a list of genes with signficant differential patterns.

Usage

```
RiboDiPA(bam_file_list, gtf_file, classlabel, psite.mapping = "auto",
    exon.binning = FALSE, bin.width = 0,zero.omit = FALSE,
    bin.from.5UTR = TRUE, method = c("gtxr", "qvalue"), cores = NULL)
```

Arguments

`bam_file_list`	A vector of bam file names to be tested. Users should include path names if not located in the current working directory. Index files (.bai) will be generated if not already present.
`gtf_file`	Annotation file used to generate the BAM alignments. Note that a GTF file sourced from one organization (e.g. Ensembl) cannot be used with BAM files aligned with a GTF file sourced from another organization (e.g. UCSC).
`classlabel`	For matrix input: a DataFrame or data.frame with at least one column named `comparison`. In `comparison`, `1` stands for the reference condition, `2` stands for the treatment condtion, and `0` represents replicates not invloved in the test. Rows of `classlabel` correspond to the data, which is one row per BAM file.
`psite.mapping`	Rules for P-site offsets, to map a given read length of RPF to a P-site. See `psiteMapping` for details.
`exon.binning`	Logical indicator. If `exon.binning` is `TRUE`, use the exon boundaries indicated in the GTF file as bins for testing, otherwise, adaptive or fixed binning will be performed.
`bin.width`	Binning width per bin. `0` represents adaptive binning, which is the default method. The minimal value for fixed-width binning is `1`, which represent single-codon binning. See `dataBinning` for details.

zero.omit	If this parameter is TRUE, bins with zero reads across all replicates for a given gene are removed.
bin.from.5UTR	When the coding region length is not any integer multiple of binning width, and if value of bin.from.5UTR is TRUE, the uneven width bins will be arranged at the 3' end of the total transcript.
method	2-component character vector specifies the multiplicity correction method for codon/bin-level p-value adjustment. The default See diffPatternTest for details.
cores	The number of cores to use for parallel execution. If not specified, the number of cores is set to the value of detectCores(logical = FALSE).

Value

bin	A List object of codon/bin-level results. Each element of list is of a gene, containing codon/bin results columns: pvalue, log2FoldChange, and the adjusted p-value named by the first string in method.
gene	A DataFrame object of gene-level results. It contains columns: tvalue, pvalue, and the adjusted p-value named by the second string in method.
small	Names of genes without sufficient reads
classlabel	The same as input classlabel.
data	Tracks of binned data of all genes reported in bin and gene.
method	The same as input method.
coverage	A list object of matrices. Each element is a matrix representing the P-site footprints of a gene. Rows corrspond to replicates and columns corrspond to nucleotide location with reference to the total transcript.
counts	A matrix object of read counts. Rows corrspond to genes and columns corrspond to replicates.
exons	A List object of matrices. Each element contains the relative start and end positions of exons in the gene with reference to the total transcript
psite.mapping	The P-site mapping rule or A-site mapping rule used.

See Also

psiteMapping, dataBinning, diffPatternTest, diffPatternTestExon

Examples

```
library(BiocFileCache)
file_names <- c("WT1.bam", "WT2.bam", "MUT1.bam", "MUT2.bam", "eg.gtf")
url <- "https://github.com/jipingw/RiboDiPA-data/raw/master/"
bfc <- BiocFileCache()
bam_path <- bfcrpath(bfc,paste0(url,file_names))

classlabel <- data.frame(
    condition = c("mutant", "mutant", "wildtype", "wildtype"),
    comparison = c(2, 2, 1, 1)
```

```
)
rownames(classlabel) <- c("mutant1","mutant2","wildtype1","wildtype2")
result.pip <- RiboDiPA(bam_path[1:4], bam_path[5], classlabel, cores=2)
```

Index

∗ **A-site**
psiteMapping, 14
∗ **P-site**
psiteMapping, 14
∗ **bin width**
dataBinning, 6
∗ **data binning**
dataBinning, 6
∗ **datasets**
data.binned, 4
data.psite, 5
result.exon, 15
result.pst, 16
∗ **htest**
diffPatternTest, 7
∗ **pattern similarity test**
diffPatternTest, 7

binTrack, 2
bpTrack, 3

data.binned, 4
data.psite, 5
dataBinning, 6, *18*
diffPatternTest, 7, *18*
diffPatternTestExon, 9, *18*

exonTrack, 10

normFactor, 11

p.adjust, *8*
plotTest, 12
plotTrack, 12
psiteCal, 13
psiteMapping, *7*, 14, *18*

result.exon, 15
result.pst, 16
RiboDiPA, 17