Package 'rnaEditr'

November 14, 2025

Title Statistical analysis of RNA editing sites and hyper-editing regions

Version 1.21.0

Description RNAeditr analyzes site-specific RNA editing events, as well as hyper-editing regions. The editing frequencies can be tested against binary, continuous or survival outcomes. Multiple covariate variables as well as interaction effects can also be incorporated in the statistical models.

Depends R (>= 4.0)

License GPL-3

Encoding UTF-8

LazyData false

RoxygenNote 7.1.1

Imports GenomicRanges, IRanges, BiocGenerics, Seqinfo, bumphunter, S4Vectors, stats, survival, logistf, plyr, corrplot

Suggests knitr, rmarkdown, testthat

biocViews GeneTarget, Epigenetics, DimensionReduction, FeatureExtraction, Regression, Survival, RNASeq

VignetteBuilder knitr

URL https://github.com/TransBioInfoLab/rnaEditr

BugReports https://github.com/TransBioInfoLab/rnaEditr/issues

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

git_branch devel

git_last_commit 6261cfb

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-11-13

Author Lanyu Zhang [aut, cre],
Gabriel Odom [aut],
Tiago Silva [aut],
Lissette Gomez [aut],
Lily Wang [aut]

Maintainer Lanyu Zhang <jennyzly2016@gmail.com>

Contents

AddMetaData . . . 2
AllCloseByRegions . . . 4
AllCoeditedRegions . . . 5
AnnotateResults . . . 7
CountSamplesPerGroup . . . 9
CreateEditingTable . . . 10
CreateOutputDF . . . 11
CreateRdrop . . . 13
FindCorrelatedRegions . . . 14
GetMinPairwiseCor . . . 15
GetSitesLocations . . . 16
MakeModelFormula . . . 17
MarkCoeditedSites . . . 18
OrderSitesByLocation . . . 20
PlotEditingCorrelations . . . 21
RegionSummaryMethod . . . 22
rnaedit_df . . . 23
SingleCloseByRegion . . . 23
SingleCoeditedRegion . . . 24
SitesToRegion . . . 26
SummarizeAllRegions . . . 28
SummarizeSingleRegion . . . 29
TestAssociations . . . 30
TestSingleRegion . . . 32
TransformToGR . . . 33
t_rnaedit_df . . . 35

Index **36**

`AddMetaData` *Add metadata columns to GRanges object.*

Description

Add metadata information to GRanges object.

Usage

```
AddMetaData(
  target_gr,
  annot_gr = NULL,
  annotType_char = c("geneSymbol", "region"),
  annotLabel_char = "symbol",
```

```
  genome = c("hg38", "hg19")
)
```

Arguments

target_gr	A GRanges object that will be annotated with metadata
annot_gr	A GRanges object that includes the metadata information. When annotType_char = "geneSymbol", this argument can be left as NULL, and the gene annotation file saved in the package will be used to annotate target_gr. When annotType_char = "region", this argument must be specified, each row in target_gr will be annotated with rows in annot_gr that overlap with it.
annotType_char	Type of the metadata column, defaults to "geneSymbol".
annotLabel_char	Name of the metadata column, defaults to "symbol" which corresponds to default setting "geneSymbol" for argument annotType_char.
genome	Use "hg19" or "hg38" gene reference. Defaults to "hg38".

Value

A GRanges object with seqnames, ranges, region, and supplied metadata information.

Examples

```
data(rnaedit_df)

input_gr <- TransformToGR(
  genes_char = "PHACTR4",
  type = "symbol",
  genome = "hg19"
)

# identifies co-edited region within input_gr
coedited_gr <- AllCoeditedRegions(
  regions_gr = input_gr,
  rnaEditMatrix = rnaedit_df,
  output = "GRanges",
  method = "spearman"
)

# identify input regions for co-edited regions
AddMetaData(
  target_gr = coedited_gr,
  annot_gr = input_gr,
  annotType_char = "region",
  annotLabel_char = "inputRegion",
  genome = "hg19"
)
```

`AllCloseByRegions` *Extract clusters of RNA editing sites located closely in genomic regions.*

Description

A wrapper function to extract clusters of RNA editing sites that are located closely in genomic regions.

Usage

```
AllCloseByRegions(
  regions_gr,
  rnaEditMatrix,
  maxGap = 50,
  minSites = 3,
  progressBar = "time"
)
```

Arguments

`regions_gr`	A GRanges object of input genomic regions.
`rnaEditMatrix`	A matrix (or data frame) of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (`data(rnaedit_df)`).
`maxGap`	An integer, genomic locations within `maxGap` from each other are placed into the same cluster. Defaults to 50.
`minSites`	An integer, minimum number of RNA editing sites within each resulting cluster. Defaults to 3.
`progressBar`	Name of the progress bar to use. There are currently five types of progress bars: `"time"`, `"none"`, `"text"`, `"tk"`, and `"win"`. Defaults to `"time"`. See `create_progress_bar` for more details.

Details

The algorithm of this function is based on the `clusterMaker` function in the `bumphunter` R package. Each cluster is essentially a group of site locations such that two consecutive locations in the cluster are separated by less than `maxGap`.

Value

A GRanges object containing genomic regions of RNA editing sites located closely within each input pre-defined genomic region.

See Also

TransformToGR, AllCoeditedRegions, CreateEditingTable, SummarizeAllRegions, TestAssociations, AnnotateResults

Examples

```
data(rnaedit_df)

exm_regions <- TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

AllCloseByRegions(
  regions_gr = exm_regions,
  rnaEditMatrix = rnaedit_df,
  maxGap = 50,
  minSites = 3,
  progressBar = "time"
)
```

AllCoeditedRegions *Extracts contiguous co-edited genomic regions from input genomic regions .*

Description

A wrapper function to extract contiguous co-edited genomic regions from input genomic regions.

Usage

```
AllCoeditedRegions(
  regions_gr,
  rnaEditMatrix,
  output = c("GRanges", "dataframe"),
  rDropThresh_num = 0.4,
  minPairCorr = 0.1,
  minSites = 3,
  method = c("spearman", "pearson"),
  returnAllSites = FALSE,
  progressBar = "time",
  verbose = TRUE
)
```

Arguments

regions_gr	A GRanges object of input genomic regions.
rnaEditMatrix	A matrix (or data frame) of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (data(rnaedit_df)).
output	Type of output data. Defaults to "GRanges".
rDropThresh_num	Threshold for minimum correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites. Please set a number between 0 and 1. Defaults to 0.4.
minPairCorr	Threshold for minimum pairwise correlation of sites within a selected cluster. To use this filter, set a number between -1 and 1 (defaults to 0.1). To select all clusters (i.e. no filter), please set this argument to -1.
minSites	Minimum number of sites to be considered as a region. Only regions with more than minSites number of sites will be returned.
method	Method for computing correlation. Defaults to "spearman".
returnAllSites	When no contiguous co-edited regions are found in an input genomic region, returnAllSites = TRUE indicates returning all the sites in the input region, while returnAllSites = FALSE indicates not returning any site from input region. Defaults to FALSE.
progressBar	Name of the progress bar to use. There are currently five types of progress bars: "time", "none", "text", "tk", and "win". Defaults to "time". See create_progress_bar for more details.
verbose	Should messages and warnings be displayed? Defaults to FALSE, but is set to TRUE when called from within SingleCoeditedRegion().

Value

When output is set as "GRanges", a GRanges object with seqnames, ranges and strand of the contiguous co-edited regions will be returned. When output is set as "dataframe", a data frame with following columns will be returned:

- site : site ID.
- chr : chromosome number.
- pos : genomic position number.
- r_drop : the correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites.
- keep : indicator for co-edited sites, the sites with keep = 1 belong to the contiguous and co-edited region.
- keep_contiguous : contiguous co-edited region number.
- regionMinPairwiseCor : the pairwise correlation of a subregion.
- keep_regionMinPairwiseCor : indicator for contiguous co-edited subregions, the regions with keepminPairwiseCor = 1 passed the minimum correlation and will be returned as a contiguous co-edited subregion.

See Also

TransformToGR, AllCloseByRegions, CreateEditingTable, SummarizeAllRegions, TestAssociations, AnnotateResults

Examples

```
data(rnaedit_df)

genes_gr <- TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

AllCoeditedRegions(
  regions_gr = genes_gr,
  rnaEditMatrix = rnaedit_df,
  output = "GRanges",
  method = "spearman"
)
```

AnnotateResults *Add Annotations to site-specific or region-based analysis results.*

Description

Add annotations to site-specific or region-based analysis results from function TestAssociations.

Usage

```
AnnotateResults(
  results_df,
  closeByRegions_gr = NULL,
  inputRegions_gr = NULL,
  genome = c("hg38", "hg19"),
  analysis = c("region-based", "site-specific")
)
```

Arguments

results_df	An output data frame from function TestAssociations, which includes variables for locations and result of statistical tests for the genomic sites or regions.
closeByRegions_gr	An output GRanges object from function AllCloseByRegions, defaults to NULL.
inputRegions_gr	A GRanges object for input genomic regions, defaults to NULL.

genome	Use "hg19" or "hg38" gene reference. Defaults to "hg38".
analysis	Results type. Defaults to "region-based". When it's set to "site-specific", arguments closeByRegions_gr and inputRegions_gr will not be used and set to NULL automatically.

Value

A data frame with locations of the genomic sites or regions (seqnames, start, end, width), annotations for locations (inputRegion, closeByRegion, symbol), test statistics (estimate, stdErr or coef, exp_coef, se_coef), pValue and false discovery rate (fdr).

See Also

TransformToGR, AllCloseByRegions, AllCoeditedRegions, CreateEditingTable, SummarizeAllRegions, TestAssociations

Examples

```
data(rnaedit_df)

# get GRanges for genes
genes_gr <- TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

# find close-by regions within the genes
closebyRegions_gr <- AllCloseByRegions(
  regions_gr = genes_gr,
  rnaEditMatrix = rnaedit_df
)

# identify co-edited regions within the genes
coedited_gr <- AllCoeditedRegions(
  regions_gr = closebyRegions_gr,
  rnaEditMatrix = rnaedit_df,
  output = "GRanges",
  method = "spearman"
)

# summarize editing levels within each gene by maximum
summarizedRegions_df <- SummarizeAllRegions(
  regions_gr = coedited_gr,
  rnaEditMatrix = rnaedit_df,
  selectMethod = MaxSites
)

exm_pheno <- readRDS(
  system.file(
  "extdata",
  "pheno_df.RDS",
```

```
  package = 'rnaEditr',
  mustWork = TRUE
  )
)

# test summarized editing levels against survival outcome
results_df <- TestAssociations(
  rnaEdit_df = summarizedRegions_df,
  pheno_df = exm_pheno,
  responses_char = "sample_type",
  covariates_char = NULL,
  respType = "binary"
)

AnnotateResults(
  results_df = results_df,
  closeByRegions_gr = closebyRegions_gr,
  inputRegions_gr = genes_gr,
  genome = "hg19"
)
```

`CountSamplesPerGroup` *Find minimum sample Size per group.*

Description

Find minimum sample size for each group which is decided by the combination of variables with class character or factor.

Usage

```
CountSamplesPerGroup(pheno_df, responses_char, covariates_char)
```

Arguments

`pheno_df`	A data frame with phenotype and covariates, which should include all the samples in `rnaEdit_df`. Please make sure the input `pheno_df` has the variable named `"sample"` to indicate sample IDs.
`responses_char`	A character vector of names of response variables in `pheno_df`. When respType is set as `"survival"`, `responses_char` should have length 2. The first element must be the name of the variable with follow up time, and the second element must be the status indicator. Status indicator should be coded as 0/1(1=death), TRUE/FALSE(TRUE=death), or 1/2(2=death). Please make sure variable names are in this order. This code has not been tested for interval-censored data yet.
`covariates_char`	A character vector of names of covariate variables in `pheno_df`.

Value

An integer.

Examples

```
exm_pheno <- readRDS(
  system.file(
  "extdata",
  "pheno_df.RDS",
  package = 'rnaEditr',
  mustWork = TRUE
  )
)

CountSamplesPerGroup(
  pheno_df = exm_pheno,
  responses_char = "sample_type",
  covariates_char = "race"
)
```

`CreateEditingTable` *Convert RNA editing matrix into a special data frame with class* `rnaEdit_df`.

Description

Convert RNA editing matrix to a special data frame with class `rnaEdit_df`, which is then used to identify differentially co-edited regions with function `TestAssociations`.

Usage

```
CreateEditingTable(rnaEditMatrix)
```

Arguments

`rnaEditMatrix`	A matrix of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (`data(rnaedit_df)`).

Value

A dataset of class `rnaEdit_df`, includes variables `seqnames`, `start`, `end`, `width` and summarized RNA editing levels in each sample.

See Also

`TransformToGR`, `AllCloseByRegions`, `AllCoeditedRegions`, `SummarizeAllRegions`, `TestAssociations`, `AnnotateResults`

Examples

```
data(rnaedit_df)
CreateEditingTable(rnaEditMatrix = rnaedit_df)[1:3, 1:5]
```

`CreateOutputDF` *Create output data in the format of data frame.*

Description

Output all the contiguous co-edited subregions found by `FindCorrelatedRegions` function and filtered by `GetMinPairwiseCor` function.

Usage

```
CreateOutputDF(
  keepSites_df,
  keepContiguousSites_df,
  keepminPairwiseCor_df,
  returnAllSites = FALSE,
  verbose = TRUE
)
```

Arguments

`keepSites_df`	An output data frame from function `MarkCoeditedSites`, with variables `site`, `keep`, `ind`, `r_drop`. Please see `MarkCoeditedSites` for details.
`keepContiguousSites_df`	An output data frame from function `FindCorrelatedRegions` with variables `site`, `subregion`. Please see `FindCorrelatedRegions` for details.
`keepminPairwiseCor_df`	An output data frame from function `GetMinPairwiseCor` with variables `subregion`, `keepminPairwiseCor` and `minPairwiseCor`. Please see `GetMinPairwiseCor` for details.
`returnAllSites`	When no contiguous co-edited regions are found in a input genomic region, `returnAllSites` = `TRUE` indicates outputting all the sites in this input region, while `returnAllSites` = `FALSE` indicates not returning any site in this input region. Defaults to FALSE.
`verbose`	Should messages and warnings be displayed? Defaults to TRUE.

Value

A data frame with following columns:

- `site` : site ID.
- `chr` : chromosome number.

- pos : genomic location.
- r_drop : the correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites.
- keep : indicator for co-edited sites, The sites with keep = 1 belong to the contiguous and co-edited region.
- keep_contiguous : contiguous co-edited region number.
- regionMinPairwiseCor : the minimum pairwise correlation between sites within a sub-region.
- keep_regionMinPairwiseCor : indicator for contiguous co-edited subregions, The regions with keepminPairwiseCor = 1 are the ones that passed regionMinPairwiseCor filter and will be returned as a contiguous co-edited sub-region.

Examples

```
data(t_rnaedit_df)

ordered_cols <- OrderSitesByLocation(
  sites_char = colnames(t_rnaedit_df),
  output = "vector"
)
exm_data <- t_rnaedit_df[, ordered_cols]

exm_sites <- MarkCoeditedSites(
  rnaEditCluster_mat = exm_data,
  method = "spearman"
)

exm_regions <- FindCorrelatedRegions(
  sites_df = exm_sites,
  featureType = "site"
)

exm_probes <- split(
  x = exm_regions$site,
  f = exm_regions$subregion
)

exm_cor <- GetMinPairwiseCor(
  rnaEditCluster_mat = exm_data,
  minPairCorr = 0.1,
  probes_ls = exm_probes,
  method = "spearman"
)

CreateOutputDF(
  keepSites_df = exm_sites,
  keepContiguousSites_df = exm_regions,
  keepminPairwiseCor_df = exm_cor$keepminPairwiseCor_df
)
```

`CreateRdrop` *Calculates R-drop values for RNA editing sites.*

Description

Calculates the correlation coefficient between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites in a region.

Usage

```
CreateRdrop(
  data,
  method = c("spearman", "pearson"),
  minEditFreq = 0.05,
  verbose = TRUE
)
```

Arguments

`data`	A data frame of RNA editing level values on individual sites, with row names as sample IDs and column names as site IDs in the form of "chrAA:XXXXXXXX".
`method`	Method for computing correlation. Defaults to `"spearman"`.
`minEditFreq`	Threshold for minimum percentage of edited samples for a given site. The `r_drop` value of the sites with frequency lower than `minEditFreq` will be set as NA. Please set a number between 0 and 1. Defaults to 0.05.
`verbose`	Should messages and warnings be displayed? Defaults to TRUE.

Value

A data frame with the following columns:

- `site` : site ID.
- `r_drop` : the correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites.

Examples

```
data(t_rnaedit_df)

ordered_cols <- OrderSitesByLocation(
  sites_char = colnames(t_rnaedit_df),
  output = "vector"
)
exm_data <- t_rnaedit_df[, ordered_cols]

CreateRdrop(
  data = exm_data,
```

```
  method = "spearman"
)
```

`FindCorrelatedRegions` *Find contiguous co-edited subregions.*

Description

Find contiguous co-edited subregions based on the output file from function `MarkCoeditedSites`.

Usage

```
FindCorrelatedRegions(
  sites_df,
  featureType = c("site", "cpg"),
  minSites_int = 3
)
```

Arguments

`sites_df`	An output data frame from function `MarkCoeditedSites`, with variables `site`, `keep`, `ind`, `r_drop`. Please see `MarkCoeditedSites` for details.
`featureType`	Feature type, Defaults to `"site"`.
`minSites_int`	An integer indicates the minimum number of sites to be considered a contiguous co-edited region.

Value

A data frame with the following columns:

- `site` : site ID.
- `subregion` : index for each output contiguous co-edited region.

Examples

```
data(t_rnaedit_df)

ordered_cols <- OrderSitesByLocation(
  sites_char = colnames(t_rnaedit_df),
  output = "vector"
)
exm_data <- t_rnaedit_df[, ordered_cols]

exm_sites <- MarkCoeditedSites(
  rnaEditCluster_mat = exm_data,
  method = "spearman"
)
```

```
FindCorrelatedRegions(
  sites_df = exm_sites,
  featureType = "site"
)
```

`GetMinPairwiseCor` *Calculate minimum pairwise correlation for sub-regions.*

Description

Filter the contiguous co-edited subregions found from `FindCorrelatedRegions`, by calculating pairwise correlations and then selecting subregions passing the minimum correlation filter.

Usage

```
GetMinPairwiseCor(
  rnaEditCluster_mat,
  minPairCorr = 0.1,
  probes_ls,
  method = c("spearman", "pearson")
)
```

Arguments

`rnaEditCluster_mat`	A matrix of RNA editing level values on individual sites, with row names as sample IDs and column names as site IDs in the form of "chrAA:XXXXXXXX".
`minPairCorr`	Minimum pairwise correlation coefficient of sites within a cluster, used as a filter. To use this filter, set a number between -1 and 1 (defaults to 0.1). To turn it off, please set the number to -1.
`probes_ls`	A list of regions with sites. Please note that probes in each list need to be ordered by their locations.
`method`	Method for computing correlation. Defaults to `"spearman"`.

Value

A list with a list of probes passing the minPairCorr and a data frame with the following columns:

- `subregion` : index for each output contiguous co-edited region.
- `keepminPairwiseCor` : indicator for contiguous co-edited subregions, The regions with `keepminPairwiseCor = 1` passed the minimum correlation and will be returned as a contiguous co-edited subregion.
- `minPairwiseCor` : the minimum pairwise correlation of sites within a subregion.

Examples

```
data(t_rnaedit_df)

ordered_cols <- OrderSitesByLocation(
  sites_char = colnames(t_rnaedit_df),
  output = "vector"
)
exm_data <- t_rnaedit_df[, ordered_cols]

exm_sites <- list(
  "1" = c("chr1:28661656", "chr1:28661718", "chr1:28662148")
)

GetMinPairwiseCor(
  rnaEditCluster_mat = exm_data,
  minPairCorr = 0.1,
  probes_ls = exm_sites,
  method = "spearman"
)
```

`GetSitesLocations` *Extract RNA editing sites located in a genomic region.*

Description

Extract and order RNA editing sites located within an input genomic region.

Usage

```
GetSitesLocations(
  region_df,
  rnaEditMatrix,
  output = c("locationsOnly", "locationsAndValues")
)
```

Arguments

`region_df`	A data frame with the input genomic region. Please make sure columns `seqnames`, `start`, and `end` are included in the data frame.
`rnaEditMatrix`	A matrix (or data frame) of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (`data(rnaedit_df)`).
`output`	Type of output data. Defaults to `"locationsOnly"`.

Value

When output is set to "locationsOnly", a data frame of extracted and ordered RNA editing sites with columns chr and pos will be returned.

When output is set to "locationsAndValues", a data frame of RNA editing level values from the extracted and ordered sites will be returned. Please note that site IDs will be in row names of the output data frame.

Examples

```
data (rnaedit_df)

exm_region <- data.frame(
   seqnames = "chr1",
   start =  28000000,
   end = 28826881,
   stringsAsFactors = FALSE
)

GetSitesLocations(
   region_df = exm_region,
   rnaEditMatrix = rnaedit_df,
   output = "locationsOnly"
)[1:3, ]
```

MakeModelFormula *Make model formula.*

Description

Make model formula for different types of phenotype responses.

Usage

```
MakeModelFormula(
  responses_char,
  covariates_char = NULL,
  respType = c("binary", "continuous", "survival")
)
```

Arguments

responses_char	A character vector of the response variable.
covariates_char	A character vector of the covariate variables.
respType	Type of outcome. Defaults to "binary".

Details

When `respType` is set as `"survival"`, `"surv_object"` is only a placeholder here , which will be defined later in `TestSingleRegion()`.

Value

A character vector of the model formula.

Examples

```
MakeModelFormula(
  responses_char = "age",
  covariates_char = c("sex", "tumor_type"),
  respType = "continuous"
)

MakeModelFormula(
  responses_char = "sample_type",
  covariates_char = c("sex", "tumor_type"),
  respType = "binary"
)

MakeModelFormula(
  responses_char = c("OS.time", "OS"),
  covariates_char = c("sex", "tumor_type"),
  respType = "survival"
)
```

`MarkCoeditedSites` *Mark RNA editing sites in contiguous and co-edited region.*

Description

Mark RNA editing sites in contiguous and co-edited region by selecting sites for which `r_drop` values calculated from inner function `CreateRdrop` is greater than `rDropThresh_num`.

Usage

```
MarkCoeditedSites(
  rnaEditCluster_mat,
  rDropThresh_num = 0.4,
  method = c("spearman", "pearson"),
  minEditFreq = 0.05,
  verbose = TRUE
)
```

Arguments

rnaEditCluster_mat	A matrix of RNA editing level values on individual sites, with row names as sample IDs and column names as site IDs in the form of "chrAA:XXXXXXXX".
rDropThresh_num	Threshold for minimum correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites. Please set a number between 0 and 1. Defaults to 0.4.
method	Method for computing correlation. Defaults to "spearman".
minEditFreq	Threshold for minimum percentage of edited samples for a given site. The r_drop value of the sites with frequency lower than minEditFreq will be set as NA. Please set a number between 0 and 1. Defaults to 0.05.
verbose	Should messages and warnings be displayed? Defaults to TRUE.

Details

r_drop statistic is used to identify co-edited sites. An outlier site (keep = 0) in a genomic region typically has low correlation with the rest of the sites in a genomic region. The sites with r_drop value greater than rDropThresh_num are marked to have keep = 1. Please see CreateRdrop for more details.

Value

A data frame with the following columns:

- site : site ID.
- r_drop : The correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites.
- keep : indicator for co-edited sites, The sites with keep = 1 belong to the contiguous and co-edited region.
- keep_contiguous : contiguous co-edited region number

- site : site ID.
- keep : indicator for co-edited sites, The sites with keep = 1 belong to the contiguous and co-edited region.
- ind : index for the sites.
- r_drop : the correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites.

See Also

CreateRdrop

Examples

```
data(t_rnaedit_df)

ordered_cols <- OrderSitesByLocation(
  sites_char = colnames(t_rnaedit_df),
  output = "vector"
)
exm_data <- t_rnaedit_df[, ordered_cols]

MarkCoeditedSites(
  rnaEditCluster_mat = exm_data,
  method = "spearman"
)
```

`OrderSitesByLocation` *Order RNA editing sites by their genomic locations.*

Description

Split RNA editing sites locations into chromosomes and positions, and then order the sites by their genomic locations.

Usage

```
OrderSitesByLocation(sites_char, output = c("dataframe", "vector"))
```

Arguments

`sites_char`	A character vector of RNA editing sites. site IDs should be in the form of "chrAA:XXXXXXXX".
`output`	Type of output data. Defaults to `"dataframe"`.

Value

Depends on the output type. When `output` is set as `"vector"`, a character vector of ordered input RNA editing sites will be returned. When `output` is set as `"dataframe"`, a data frame of ordered RNA editing sites with following columns will be returned:

- `site` : site ID.
- `chr` : chromosome number.
- `pos` : genomic location.

Examples

```
exm_sites <- c(
  "chr22:41327462", "chr22:24969087",
  "chr22:29538891", "chr22:45736763"
)

OrderSitesByLocation(
  sites_char = exm_sites,
  output = "dataframe"
)
```

`PlotEditingCorrelations` *Plotting correlations of RNA editing levels within a region.*

Description

Plotting correlations of RNA editing levels within a region.

Usage

```
PlotEditingCorrelations(region_gr, rnaEditMatrix, ...)
```

Arguments

`region_gr`	A GRanges object of a region.
`rnaEditMatrix`	A matrix (or data frame) of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (`data(rnaedit_df)`).
`...`	Dots for additional internal arguments, see `corrplot` for details.

Value

(Invisibly) returns a reordered correlation matrix.

Examples

```
data(rnaedit_df)

genes_gr <- TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

exm_regions <- AllCoeditedRegions(
  regions_gr = genes_gr,
```

```
  rnaEditMatrix = rnaedit_df,
  output = "GRanges",
  method = "spearman"
)

PlotEditingCorrelations(
  region_gr = exm_regions[1],
  rnaEditMatrix = rnaedit_df
)
```

`RegionSummaryMethod` *Methods to summarize RNA editing levels from multiple sites within a single region.*

Description

Summarize RNA editing sites in a single region by taking maximum, mean, median or first principal component.

Usage

```
MaxSites(rnaEditMatrix, ...)

MeanSites(rnaEditMatrix, ...)

MedianSites(rnaEditMatrix, ...)

PC1Sites(rnaEditMatrix, ...)
```

Arguments

`rnaEditMatrix`	A matrix (or data frame) of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (`data(rnaedit_df)`).
`...`	Dots for additional internal arguments (currently unused).

Value

A named numeric vector of summarized RNA editing levels with sample IDs as names.

Examples

```
data(rnaedit_df)
MedianSites(rnaEditMatrix = rnaedit_df)[1:3]
```

rnaedit_df *Example breast cancer RNA editing dataset.*

Description

A subset of the TCGA breast cancer RNA editing dataset for 272 edited sites on genes PHACTR4, CCR5, METTL7A and a few randomly sampled sites for 221 subjects.

Usage

```
rnaedit_df
```

Format

A data frame containing RNA editing levels for 272 sites (in the rows) for 221 subjects (in the columns). Row names are site IDs and column names are sample IDs.

Source

Synapse database ID: syn2374375.

SingleCloseByRegion *Extracts clusters of RNA editing sites located closely in a single genomic region.*

Description

Extracts clusters of RNA editing sites located closely in an input genomic region.

Usage

```
SingleCloseByRegion(region_df, rnaEditMatrix, maxGap = 50, minSites = 3)
```

Arguments

region_df	A data frame with the input genomic region. Please make sure columns seqnames, start, and end are included in the data frame.
rnaEditMatrix	A matrix (or data frame) of RNA editing level values for individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (data(rnaedit_df)).
maxGap	An integer, genomic locations within maxGap from each other are placed into the same cluster. Defaults to 50.
minSites	An integer, minimum number of edited sites for a cluster to be selected for output. Defaults to 3.

Details

The algorithm of this function is based on the clusterMaker function in the bumphunter R package. Each cluster is essentially a group of sites such that two consecutive sites in the cluster are separated by less than maxGap.

Value

A GRanges object containing genomic locations of RNA editing sites located closely within the single input pre-defined genomic region.

Examples

```
data(rnaedit_df)

exm_region <- data.frame(
  seqnames = "chr1",
  start =  28691093,
  end = 28826881,
  stringsAsFactors = FALSE
)

SingleCloseByRegion(
  region_df = exm_region,
  rnaEditMatrix = rnaedit_df,
  maxGap = 50,
  minSites = 3
)
```

SingleCoeditedRegion *Extracts contiguous co-edited genomic regions from a single genomic region.*

Description

Extracts contiguous co-edited genomic regions from an input genomic region.

Usage

```
SingleCoeditedRegion(
  region_df,
  rnaEditMatrix,
  output = c("GRanges", "dataframe"),
  rDropThresh_num = 0.4,
  minPairCorr = 0.1,
  minSites = 3,
  method = c("spearman", "pearson"),
  minEditFreq = 0.05,
```

```
  returnAllSites = FALSE,
  verbose = TRUE
)
```

Arguments

region_df	A data frame with the input genomic region. Please make sure columns seqnames, start, and end are included in the data frame.
rnaEditMatrix	A matrix (or data frame) of RNA editing level values on individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (data(rnaedit_df)).
output	Type of output data, can be "GRanges" or "dataframe". Defaults to "GRanges".
rDropThresh_num	Threshold for minimum correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites. Please set a number between 0 and 1. Defaults to 0.4.
minPairCorr	Minimum pairwise correlation coefficient of a cluster is used as a filter to select clusters for output. Only clusters with all pairwise correlations between sites more than minPairCorr will be selected for output. To use this filter, set this argument to a number between -1 and 1 (defaults to 0.1). To turn it off, please set the argument to -1.
minSites	Minimum number of sites to be considered a region. Only regions with more than minSites number of sites will be returned.
method	Method for computing correlations. Defaults to "spearman".
minEditFreq	Threshold for minimum percentage of samples for a given site. The r_drop value of the sites with frequency lower than minEditFreq will be set as NA. Please set a number between 0 and 1. Defaults to 0.05.
returnAllSites	When no co-edited region is found in an input genomic region, returnAllSites = TRUE indicates outputting all the sites from the input region, while returnAllSites = FALSE indicates not returning any site from the input region. Defaults to FALSE.
verbose	Should messages and warnings be displayed? Defaults to TRUE, but is set to FALSE when called from within AllCoeditedRegions().

Value

When output is set to "GRanges", a GRanges object with seqnames, ranges and strand of the contiguous co-edited regions will be returned.

When output is set to "dataframe", a data frame with following columns will be returned:

- site : site ID.
- chr : chromosome.
- pos : genomic location.
- r_drop : the correlation between RNA editing levels of one site and the mean RNA editing levels of the rest of the sites.

- keep : indicator for co-edited sites, The sites with keep = 1 belong to the contiguous and co-edited region.
- keep_contiguous : contiguous co-edited region number.
- regionMinPairwiseCor : the minimum pairwise correlation of a co-edited region.
- keep_regionMinPairwiseCor : equals 1 for contiguous co-edited subregions. The regions with keepminPairwiseCor = 1 are the ones that passed the regionMinPairwiseCor filter and will be returned as a co-edited sub-region.

Examples

```
data(rnaedit_df)

exm_region <- data.frame(
  seqnames = "chr1",
  start =  28691093,
  end = 28826881,
  stringsAsFactors = FALSE
)

SingleCoeditedRegion(
  region_df = exm_region,
  rnaEditMatrix = rnaedit_df,
  minPairCorr = 0.25,
  output = "dataframe",
  method = "spearman"
)
```

SitesToRegion *Create output data in the format of GRanges.*

Description

Output contiguous co-edited subregions found by FindCorrelatedRegions function and filtered by GetMinPairwiseCor function.

Usage

```
SitesToRegion(
  sitesSubregion_df,
  sitesAreOrdered = TRUE,
  keepminPairwiseCor_df,
  returnAllSites = FALSE,
  verbose = TRUE
)
```

Arguments

sitesSubregion_df	An output data frame from function FindCorrelatedRegions with variables site, subregion. Please see FindCorrelatedRegions for details.
sitesAreOrdered	Are the sites in sitesSubregion_df ordered by location? Defaults to FALSE.
keepminPairwiseCor_df	An output data frame from function GetMinPairwiseCor with variables subregion, keepminPairwiseCor and minPairwiseCor. Please see GetMinPairwiseCor for details.
returnAllSites	When no contiguous co-edited regions are found in a input genomic region, returnAllSites = TRUE indicates outputting all the sites in this input region, while returnAllSites = FALSE indicates not returning any site in this input region. Defaults to FALSE.
verbose	Should messages and warnings be displayed? Defaults to TRUE.

Value

A GRanges object with seqnames, ranges and strand of the contiguous co-edited regions.

Examples

```
data(t_rnaedit_df)

ordered_cols <- OrderSitesByLocation(
  sites_char = colnames(t_rnaedit_df),
  output = "vector"
)
exm_data <- t_rnaedit_df[, ordered_cols]

exm_sites <- MarkCoeditedSites(
  rnaEditCluster_mat = exm_data,
  method = "spearman"
)

exm_regions <- FindCorrelatedRegions(
  sites_df = exm_sites,
  featureType = "site"
)

exm_sites <- split(
  x = exm_regions$site,
  f = exm_regions$subregion
)

exm_cor <- GetMinPairwiseCor(
  rnaEditCluster_mat = exm_data,
  minPairCorr = 0.1,
  probes_ls = exm_sites,
  method = "spearman"
```

```
)

SitesToRegion(
  sitesSubregion_df = exm_regions,
  keepminPairwiseCor_df = exm_cor$keepminPairwiseCor_df
)
```

SummarizeAllRegions *Summarize RNA editing levels from multiple sites in regions.*

Description

A wrapper function to summarize RNA editing levels from multiple sites in regions.

Usage

```
SummarizeAllRegions(
  regions_gr,
  rnaEditMatrix,
  selectMethod = MedianSites,
  progressBar = "time",
  ...
)
```

Arguments

regions_gr	A GRanges object of input genomic regions.
rnaEditMatrix	A matrix (or data frame) of RNA editing level values for individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (`data(rnaedit_df)`).
selectMethod	Method for summarizing regions. Available options are `"MaxSites"`, `"MeanSites"`, `"MedianSites"`, `"PC1Sites"`. Please see RegionSummaryMethod for more details.
progressBar	Name of the progress bar to use. There are currently five types of progress bars: `"time"`, `"none"`, `"text"`, `"tk"`, and `"win"`. Defaults to `"time"`. See create_progress_bar for more details.
...	Dots for additional internal arguments (currently unused).

Value

A data frame of the class `rnaEdit_df`, includes variables `seqnames`, `start`, `end`, `width` and summarized RNA editing levels in each sample.

See Also

TransformToGR, AllCloseByRegions, AllCoeditedRegions, CreateEditingTable, TestAssociations, AnnotateResults

Examples

```
data(rnaedit_df)

genes_gr <- TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

exm_regions <- AllCoeditedRegions(
  regions_gr = genes_gr,
  rnaEditMatrix = rnaedit_df,
  output = "GRanges",
  method = "spearman"
)

SummarizeAllRegions(
  regions_gr = exm_regions,
  rnaEditMatrix = rnaedit_df
)[1:3, 1:6]
```

SummarizeSingleRegion *Summarizes RNA editing levels from multiple sites in a single region.*

Description

Summarizes RNA editing levels from multiple sites in an input region.

Usage

```
SummarizeSingleRegion(
  region_df,
  rnaEditMatrix,
  selectMethod = MedianSites,
  ...
)
```

Arguments

region_df	A data frame with the input genomic region. Please make sure columns seqnames, start, and end are included in the data frame.
rnaEditMatrix	A matrix (or data frame) of RNA editing level values for individual sites, with row names as site IDs in the form of "chrAA:XXXXXXXX", and column names as sample IDs. Please make sure to follow the format of example dataset (data(rnaedit_df)).
selectMethod	Method for summarizing regions. Available options are "MaxSites", "MeanSites", "MedianSites", "PC1Sites". Please see RegionSummaryMethod for more details.
...	Dots for additional internal arguments (currently unused).

Value

A named numeric vector of summarized RNA editing levels with sample IDs as column names.

Examples

```
data(rnaedit_df)

exm_region <- data.frame(
  seqnames = "chr1",
  start =  28691093,
  end = 28826881,
  stringsAsFactors = FALSE
)

SummarizeSingleRegion(
  region_df = exm_region,
  rnaEditMatrix = rnaedit_df
)[1:3]
```

`TestAssociations` *Test associations between phenotype and RNA editing levels.*

Description

A wrapper function to test associations between phenotype and RNA editing levels in single-site analysis or summarized RNA editing levels in region-based analysis.

Usage

```
TestAssociations(
  rnaEdit_df,
  pheno_df,
  responses_char,
  covariates_char = NULL,
  respType = c("binary", "continuous", "survival"),
  progressBar = "time",
  orderByPval = TRUE
)
```

Arguments

`rnaEdit_df`	A data frame with class `rnaEdit_df`, which is a output from function `CreateEditingTable()` or function `SummarizeAllRegions()`. This data frame should include RNA editing level values, with row names as site IDs or region IDs, and column names as sample IDs.

pheno_df	A data frame with phenotype and covariates, which should include all the samples in rnaEdit_df. Please make sure the input pheno_df has the variable named "sample" to indicate sample IDs.
responses_char	A character vector of names of response variables in pheno_df. When respType is set as "survival", responses_char should have length 2. The first element must be the name of the variable with following up time, and the second element must be status indicator. Status indicator should be coded as 0/1(1=death), TRUE/FALSE(TRUE=death), or 1/2(death). Please make sure variable names are in this order. We have not tested this code on interval-censored data; use at your own risk. See Surv for more details.
covariates_char	A character vector of names of covariate variables in pheno_df.
respType	Type of outcome. Defaults to "binary".
progressBar	Name of the progress bar to use. There are currently five types of progress bars: "time", "none", "text", "tk", and "win". Defaults to "time". See create_progress_bar for more details.
orderByPval	Sort co-edited regions by model p-value or not? Defaults to TRUE.

Value

A data frame with locations of the genomic regions or sites (seqnames, start, end, width), test statistics (estimate, stdErr or coef, exp_coef, se_coef), pValue and false discovery rate (fdr).

See Also

TransformToGR, AllCloseByRegions, AllCoeditedRegions, CreateEditingTable, SummarizeAllRegions, AnnotateResults

Examples

```
data(rnaedit_df)

genes_gr <- TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

exm_regions <- AllCoeditedRegions(
  regions_gr = genes_gr,
  rnaEditMatrix = rnaedit_df,
  output = "GRanges",
  method = "spearman"
)

sum_regions <- SummarizeAllRegions(
  regions_gr = exm_regions,
  rnaEditMatrix = rnaedit_df,
```

```
  selectMethod = MaxSites
)

exm_pheno <- readRDS(
  system.file(
  "extdata",
  "pheno_df.RDS",
  package = 'rnaEditr',
  mustWork = TRUE
  )
)

TestAssociations(
  rnaEdit_df = sum_regions,
  pheno_df = exm_pheno,
  responses_char = "sample_type",
  covariates_char = NULL,
  respType = "binary"
)
```

`TestSingleRegion` *Test associations between phenotype and RNA editing levels.*

Description

Test associations between phenotype and RNA editing levels in a single site or summarized RNA editing levels in a single region.

Usage

```
TestSingleRegion(
  rnaEdit_num,
  modelPrep_ls,
  respType = c("binary", "continuous", "survival")
)
```

Arguments

`rnaEdit_num`	A named numeric vector of (summarized) RNA editing level values with sample IDs as names.
`modelPrep_ls`	A list includes `modelFormula_char` which is created by function `MakeModelFormula`, `pheno_df` which is the input phenotype data frame in `TestAssociations`, and `minSize` (minimum sample size per group to use regular logistic regression) which is created by function `CountSamplesPerGroup` when `respType` is `"binary"`.
`respType`	Type of outcome. Defaults to `"binary"`.

Details

minSize is used by function TestSingleRegion to decide on whether to use regular logistic regression or Firth corrected logistic regression ("https://www.jstor.org/stable/2336755").

Value

a dataframe with test statistics (estimate, stdErr, pValue or coef, exp_coef, se_coef, pValue).

Examples

```
data(rnaedit_df)

exm_pheno <- readRDS(
  system.file(
  "extdata",
  "pheno_df.RDS",
  package = 'rnaEditr',
  mustWork = TRUE
  )
)

exm_model <- list(
  modelFormula_char = "age_at_diagnosis ~ rnaEditSummary",
  pheno_df = exm_pheno,
  minSize = NULL
)

TestSingleRegion(
  rnaEdit_num = unlist(rnaedit_df[2,]),
  modelPrep_ls = exm_model,
  respType = "continuous"
)
```

TransformToGR *Transform gene symbols or region ranges into GRanges object.*

Description

Transform a character vector of gene symbols or region ranges into a GRanges object.

Usage

```
TransformToGR(
  genes_char,
  type = c("symbol", "region"),
  genome = c("hg38", "hg19")
)
```

Arguments

genes_char	A character vector of gene symbols or region ranges. If you select type to be "symbol", then please make sure your input of genes_char is in the format of c("ABCB10", "PEX26"). If you select type to be "region", then please make sure your input of genes_char is in the format of c("chr1:33772367-33791699", "chr22:18555686-18573797").
type	What is the type of genes_char. Can be "symbol" (default) or "region".
genome	Use "hg19" or "hg38" gene reference. Defaults to "hg38". It's only used when type is set to "symbol"

Details

TransformToGR() uses the hg19/hg38 genes to associate gene symbols with their genomic region ranges. The pre-processed dataset is saved in inst/extdata in this package.

Users who wish to add gene symbols to the GRanges created using function TransformToGR() can use function AddMetaData(). Please see AddMetaData for details.

Value

A GRanges object with seqnames, ranges and strand.

See Also

AllCloseByRegions, AllCoeditedRegions, CreateEditingTable, SummarizeAllRegions, TestAssociations, AnnotateResults

Examples

```
TransformToGR(
  genes_char = c("PHACTR4", "CCR5", "METTL7A"),
  type = "symbol",
  genome = "hg19"
)

TransformToGR(
  genes_char = c("chr22:18555686-18573797", "chr22:36883233-36908148"),
  type = "region",
  genome = "hg19"
)
```

t_rnaedit_df *Transposed breast cancer example dataset.*

Description

A subset of the TCGA breast cancer RNA editing dataset for 20 randomly selected RNA editing sites and 50 randomly selected subjects from example dataset `rnaedit_df`. Please note that this is only a computational testing dataset for inner functions of this package. To test main functions, please use dataset `rnaedit_df` instead.

Usage

```
t_rnaedit_df
```

Format

A data frame containing RNA editing levels for 50 subjects (in the rows) at 20 edited sites (in the columns). Row names are sample IDs and column names are site IDs.

Source

Synapse database ID: syn2374375.

Index

∗ **datasets**
rnaedit_df, 23
t_rnaedit_df, 35
∗ **internal**
AddMetaData, 2
CountSamplesPerGroup, 9
CreateOutputDF, 11
CreateRdrop, 13
FindCorrelatedRegions, 14
GetMinPairwiseCor, 15
GetSitesLocations, 16
MakeModelFormula, 17
MarkCoeditedSites, 18
OrderSitesByLocation, 20
PlotEditingCorrelations, 21
RegionSummaryMethod, 22
SingleCloseByRegion, 23
SingleCoeditedRegion, 24
SitesToRegion, 26
SummarizeSingleRegion, 29
TestSingleRegion, 32

AddMetaData, 2, *34*
AllCloseByRegions, 4, *7, 8*, *10*, *28*, *31*, *34*
AllCoeditedRegions, *5*, 5, *8*, *10*, *28*, *31*, *34*
AnnotateResults, *5*, *7*, 7, *10*, *28*, *31*, *34*

clusterMaker, *4*, *24*
corrplot, *21*
CountSamplesPerGroup, 9, *32*
create_progress_bar, *4*, *6*, *28*, *31*
CreateEditingTable, *5*, *7, 8*, 10, *28*, *30, 31*, *34*
CreateOutputDF, 11
CreateRdrop, 13, *18, 19*

FindCorrelatedRegions, *11*, 14, *15*, *26, 27*

GetMinPairwiseCor, *11*, 15, *26, 27*
GetSitesLocations, 16

MakeModelFormula, 17, *32*
MarkCoeditedSites, *11*, *14*, 18
MaxSites *(*RegionSummaryMethod*)*, 22
MeanSites *(*RegionSummaryMethod*)*, 22
MedianSites *(*RegionSummaryMethod*)*, 22

OrderSitesByLocation, 20

PC1Sites *(*RegionSummaryMethod*)*, 22
PlotEditingCorrelations, 21

RegionSummaryMethod, 22, *28, 29*
rnaedit_df, 23

SingleCloseByRegion, 23
SingleCoeditedRegion, 24
SitesToRegion, 26
SummarizeAllRegions, *5*, *7, 8*, *10*, 28, *30, 31*, *34*
SummarizeSingleRegion, 29
Surv, *31*

t_rnaedit_df, 35
TestAssociations, *5*, *7, 8*, *10*, *28*, 30, *32*, *34*
TestSingleRegion, 32
TransformToGR, *5*, *7, 8*, *10*, *28*, *31*, 33