Package 'MitoHEAR'

March 1, 2022

Type Package

Title Quantification of Mitochondrial DNA Heteroplasmy

Version 0.1.0

Author Gabriele Lubatti

Maintainer Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

Description Allows the estimation and downstream statistical analysis of the mitochondrial DNA Heteroplasmy calculated from single-cell datasets <https://github.com/ScialdoneLab/MitoHEAR/tree/master>.

License Artistic-2.0

Depends R (>= 4.0)

Imports Biostrings, circlize, ComplexHeatmap, dynamicTreeCut, GenomicRanges, ggplot2, gridExtra, IRanges, magrittr, mcclust, rdist, reshape2, rlist, Rsamtools,

Suggests clustree, fmsb, gam, karyoploteR, knitr, plotly, regioneR, rmarkdown, testthat

VignetteBuilder knitr

biocViews software

Encoding UTF-8

Config/testthat/edition 3

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2022-03-01 21:20:02 UTC

R topics documented:

`choose_features_clustering`
choose_features_clustering

Description

choose_features_clustering

Usage

```
choose_features_clustering(
  heteroplasmy_matrix,
  allele_matrix,
  cluster,
  top_pos,
  deepSplit_param,
  minClusterSize_param,
  min_value_vector,
  threshold = 0.2,
  index,
  max_frac = 0.7
)
```

Arguments

heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
allele_matrix	Fourth element returned by *get_heteroplasmy*.
cluster	Vector specifying a partition of the samples.
top_pos	Numeric value. Number of bases sorted with decreasing values of distance variance (see section *Details* below) among samples. If *relevant_bases*=NULL, then the bases for performing hierarchical clustering are the ones whose relative variance (variance of the base divided sum of variance among *top_pos* bases) is above *min_value*.
deepSplit_param	Integer value between 0 and 4 for the *deepSplit* parameter of the function *cutreeHybrid*. See section *Details* below.
minClusterSize_param	Integer value specifying the *minClusterSize* parameter of the function *cutreeHybrid*. See section *Details* below.
min_value_vector	Numeric vector. For each value in the vector, the function *clustering_angular_distance* is run with parameter *min_value* equal to one element of the vector *min_value_vector*.
threshold	Numeric value. If a base has heteroplasmy greater or equal to *threshold* in more than *max_frac* of cells, then the base is not considered for down stream analysis.
index	Fifth element returned by *get_heteroplasmy*.
max_frac	Numeric value.If a base has heteroplasmy greater or equal to *threshold* in more than *max_frac* of cells, then the base is not considered for down stream analysis.

Value

Clustree plot returned by function *clustree* from package *clustree*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://cran.r-project.org/package=clustree

---

clustering_angular_distance *clustering_angular_distance*

Description

For each pair of samples and for each base, an angular distance matrix is computed based on the four allele frequencies. Then only the angular distances corresponding to the relevant_bases are kept. If relevant bases is NULL, then only the angular distances corresponding to the bases with relative distance variance among samples above *min_value* are kept . Finally the distance between each pair of samples is defined as the euclidean distance of the angular distances corresponding to the bases that pass the previous filtering step. On this final distance matrix, a hierarchical clustering approach is performed using the function *cutreeHybrid* of the package *dynamicTreeCut*.

Usage

```
clustering_angular_distance(
  heteroplasmy_matrix,
  allele_matrix,
  cluster,
  top_pos,
  deepSplit_param,
  minClusterSize_param,
  threshold = 0.2,
  min_value,
  index,
  relevant_bases = NULL,
  max_frac = 0.7
)
```

Arguments

`heteroplasmy_matrix`	Third element returned by *get_heteroplasmy*.
`allele_matrix`	Fourth element returned by *get_heteroplasmy*.
`cluster`	Vector specifying a partition of the samples.
`top_pos`	Numeric value. Number of bases sorted with decreasing values of distance variance (see section *Details* below) among samples. If *relevant_bases*=NULL, then the bases for performing hierarchical clustering are the ones whose relative variance (variance of the base divided sum of variance among *top_pos* bases) is above *min_value*.
`deepSplit_param`	Integer value between 0 and 4 for the *deepSplit* parameter of the function *cutreeHybrid*. See section *Details* below.
`minClusterSize_param`	Integer value specifying the *minClusterSize* parameter of the function *cutreeHybrid*. See section *Details* below.
`threshold`	Numeric value. If a base has heteroplasmy greater or equal to *threshold* in more than *max_frac* of cells, then the base is not considered for down stream analysis.
`min_value`	Numeric value. If *relevant_bases*=NULL, then the bases for performing hierarchical clustering are the ones whose relative variance (variance of the base divided sum of variance among *top_pos* bases) is above *min_value*.

index	Fifth element returned by *get_heteroplasmy*.
relevant_bases	Character vector of bases to consider as features for performing hierarchical clustering on samples.Default=NULL.
max_frac	Numeric value.If a base has heteroplasmy greater or equal to *threshold* in more than *max_frac* of cells, then the base is not considered for down stream analysis.

Value

It returns a list with 4 elements:

classification	Dataframe with two columns and n_row equal to n_row in heteroplasmy_matrix. The first column is the old cluster annotation provided by cluster. The second columns is the new cluster annotation obtained with hierarchical clustering on distance matrix based on heteroplasmy values.
dist_ang_matrix	Distance matrix based on heteroplasmy values as defined in the section *Details*
top_bases_dist	Vector of bases used for hierarchical clustering. If *relevant_bases* is not NULL, then *top_bases_dist*=NULL
common_idx	Vector of indices of samples for which hierarchical clustering is performed. If *index* is NULL, then *common_idx*=NULL

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/dynamicTreeCut/versions/1.63-1/topics/cutreeHybrid

---

`detect_insertion` *detect_insertion*

Description

detect_insertion

Usage

```
detect_insertion(ref_sequence, different_sequence, length_comparison = 10)
```

Arguments

ref_sequence	Character vector whose elements are the bases of a DNA sequence to use as reference.
different_sequence	Character vector whose elements are the bases of a DNA sequence different from the reference.
length_comparison	Integer number. Number of bases to consider for the comparison between the two DNA sequences in order to detect and remove insertions in the non-reference sequence.

Value

Character vector of the different_sequence with length equal to ref_sequence, after having removed the insertions.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

---

dpt_test *dpt_test*

Description

dpt_test

Usage

```
dpt_test(heteroplasmy_matrix, time, index = NULL, method = "GAM")
```

Arguments

heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
time	Vector of diffusion pseudo time.
index	index returned by *get_heteroplasmy*.
method	Character name denoting the method to choose for assigning an adjusted p value to each of the bases. Can be one of GAM, pearson and spearman. GAM: For each base, a GAM fit with formula z ~ lo(t) is performed between the heteroplasmy values (z) and the time (t). The p value from the table "Anova for Parametric Effects" is then assigned to the base. pearson,spearman:for each base, a pearson or spearman correlation test is performed between the heteroplasmy values and the time . The p value obtained from the test is then assigned to the base. In all the three possible methods, all the p values are then corrected with the method FDR.

Value

A data frame with 2 columns and number of rows equal to n_col in *heteroplasmy_matrix*. In the first column there are the names of the bases while in the second column there are the adjusted p value.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/gam/versions/1.20/topics/gam

---

`filter_bases` *filter_bases*

Description

filter_bases

Usage

```
filter_bases(heteroplasmy_matrix, min_heteroplasmy, min_cells, index = NULL)
```

Arguments

`heteroplasmy_matrix`	Third element returned by *get_heteroplasmy*.
`min_heteroplasmy`	Numeric value.
`min_cells`	Numeric value.
`index`	Fifth element returned by *get_heteroplasmy*.

Value

Character vector of bases that have an heteroplasmy greater than *min_heteroplasmy* in more than *min_cells*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

get_distribution *get_distribution*

Description

get_distribution

Usage

```
get_distribution(heteroplasmy_matrix, FUNCTION, index = NULL)
```

Arguments

heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
FUNCTION	A character specifying the function to be applied on each column of *matrix*. The possible values are: *mean*,*max*,*min*,*median* and *sum*.
index	index returned by *get_heteroplasmy*.

Value

It returns a numeric vector with length equal to n_col of *matrix* where each element contains the result of the operation defined by *FUNCTION*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

get_heteroplasmy *get_heteroplasmy*

Description

It is one of the two main functions of the **MitoHEAR** package (together with *get_raw_counts_allele*). It computes the allele frequencies and the heteroplasmy matrix starting from the counts matrix obtained with *get_raw_counts_allele*.

Usage

```
get_heteroplasmy(
  raw_counts_allele,
  name_position_allele,
  name_position,
  number_reads,
  number_positions,
  filtering = 1,
  my.clusters = NULL
)
```

Arguments

`raw_counts_allele`
: A raw counts matrix obtained from *get_raw_counts_allele*.

`name_position_allele`
: A character vector with elements specifying the genomic coordinate of the base and the allele (obtained from *get_raw_counts_allele*).

`name_position`
: A character vector with elements specifying the genomic coordinate of the base (obtained from *get_raw_counts_allele*).

`number_reads`
: Integer specifying the minimum number of counts above which we consider the base covered by the sample.

`number_positions`
: Integer specifying the minimumnumber of bases that must be covered by the sample (with counts>*number_reads*), in order to keep the sample for down-stream analysis.

`filtering`
: Numeric value equal to 1 or 2. If 1 then only the bases that are covered by all the samples are kept for the downstream analysis. If 2 then all the bases that are covered by more than 50% of the the samples in each cluster (specified by *my.clusters*) are kept for the down-stream analysis. Default is 1.

`my.clusters`
: Character vector specifying a partition of the samples. It is only used when filtering is equal to 2. Default is NULL

Details

Starting from *raw counts allele matrix*, the function performed two consequentially filtering steps. The first one is on the samples, keeping only the ones that cover a number of bases above number_positions. The second one is on the bases, defined by the parameter filtering. The heteroplasmy for each sample-base pair is computed as *1-max(f)*, where f are the frequencies of the four alleles.

Value

It returns a list with 5 elements:

`sum_matrix`
: A matrix (n_row=number of sample, n_col=number of bases) with the counts for each sample/base, for all the initial samples and bases included in the *raw counts allele matrix*.

`sum_matrix_qc`
: A matrix (n_row=number of sample, n_col=number of bases) with the counts for each sample/base, for all the samples and bases that pass the two consequentially filtering steps.

`heteroplasmy_matrix`
: A matrix with the same dimension of *sum_matrix_qc* where each entry (i,j) is the heteroplasmy for sample i in base j.

`allele_matrix`
: A matrix (n_row=number of sample, n_col=4*number of bases) with allele frequencies, for all the samples and bases that pass the two consequentially filtering steps.

`index`
: Indices of the samples that cover a base, for all bases and samples that pass the two consequentially filtering steps; if all the samples cover all the bases, then *index* is NULL

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

Examples

```
# Two samples and two bases whose reference allele is A and C.
# The two samples have 100 reads in the reference allele and 0 in all the others.
sample1_A <- c(100, 0, 0, 0)
names_A <- rep("1_A", length(sample1_A))
sample1_C <- c(100, 0, 0, 0)
names_C <- rep("2_C", length(sample1_C))
allele <- c("A", "C", "T", "G")
names_A_allele <- paste(names_A, allele, sep = " ")
names_C_allele <- paste(names_C, allele, sep = " ")
sample1 <- c(sample1_A, sample1_C)
sample2_A <- c(100, 0, 0, 0)
sample2_C <- c(100, 0, 0, 0)
sample2 <- c(sample2_A, sample2_C)
test_allele <- matrix(c(sample1, sample2), byrow = TRUE, ncol = 8, nrow = 2)
colnames(test_allele) <- c(names_A_allele, names_C_allele)
row.names(test_allele) <- c("sample1", "sample2")
name_position_allele_test <- c(names_A_allele, names_C_allele)
name_position_test <- c(names_A, names_C)
test <- get_heteroplasmy(test_allele, name_position_allele_test, name_position_test, 50, 1, 1)
```

---

get_raw_counts_allele *get_raw_counts_allele*

Description

It is one the two main function of the **MitoHEAR** package (together with *get_heteroplasmy*). The function allows to obtain a matrix of counts (n_row = number of sample, n_col= 4*number of bases) of the four alleles in each base, for every sample. It takes as input a vector of sorted bam files (one bam file for each sample) and a fasta file for the genomic region of interest. It is based on the *pileup* function of the package Rsamtools.

Usage

```
get_raw_counts_allele(bam_input, path_fasta, cell_names, cores_number = 1)
```

Arguments

bam_input	Character vector of sorted bam files (full path). Each sample is defined by one bam file. For each bam file it is needed also the index bam file (.bai) at the same path.
path_fasta	Character string with full path to the fasta file of the genomic region of interest.
cell_names	Character vector of sample names.
cores_number	Number of cores to use.

Value

A list with three elements:

`matrix_allele_counts`
: Matrix of counts (n_row = number of sample, n_col= 4*number of bases) of the four alleles in each base, for every sample. The row names is equal to cell_names.

`name_position_allele`
: Character vector with length equal to n_col of matrix_allele_counts. Each element specifies the coordinate of genomic position for a base and the allele.

`name_position`
: Character vector with length equal to n_col of matrix_allele_counts. Each element specifies the coordinate of genomic position for a base.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/Rsamtools/versions/1.24.0/topics/pileup

---

`get_wilcox_test` *get_wilcox_test*

Description

get_wilcox_test

Usage

```
get_wilcox_test(heteroplasmy_matrix, cluster, label_1, label_2, index = NULL)
```

Arguments

`heteroplasmy_matrix`
: Third element returned by *get_heteroplasmy*.

`cluster`
: Vector specifying a partition of the samples.

`label_1`
: Character name of a first label included in cluster. It denotes the first group used for the Wilcoxon test

`label_2`
: Character name of a second label included in cluster and different from label_1. it denotes the second group used for the Wilcoxon test.

`index`
: Fifth element returned by *get_heteroplasmy*.

Value

It returns a numeric vector of length equal to n_row in matrix. Each element stands for a base and it contains the adjusted p-value (FDR), obtained in unpaired two-samples Wilcoxon test from the comparison of the heteroplasmy between the label_1 and label_2 group.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/wilcox.test

---

`plot_allele_frequency` *plot_allele_frequency*

Description

plot_allele_frequency

Usage

```
plot_allele_frequency(
  position,
  heteroplasmy_matrix,
  allele_matrix,
  cluster,
  names_allele_qc,
  names_position_qc,
  size_text,
  index
)
```

Arguments

`position`	Character name of the base to plot.
`heteroplasmy_matrix`	Third element returned by *get_heteroplasmy*.
`allele_matrix`	Fourth element returned by *get_heteroplasmy*.
`cluster`	Vector specifying a partition of the samples.
`names_allele_qc`	Character vector with length equal to n_col of *allele_matrix*. Each element specifies the name of the base and the allele.
`names_position_qc`	Character vector with length equal to n_col of *allele_matrix*. Each element specifies the name of the base.
`size_text`	Character specifying the size of the text for *gridExtra* function *grid.arrange*)
`index`	Fifth element returned by *get_heteroplasmy*.

Value

grid.arrange plot of allele frequencies of a specific base across samples divided according to cluster.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://cran.r-project.org/package=gridExtra

`plot_base_coverage` *plot_base_coverage*

Description

plot_base_coverage

Usage

```
plot_base_coverage(
  sum_matrix,
  sum_matrix_qc,
  selected_cells,
  interactive = FALSE,
  text_size = 10
)
```

Arguments

`sum_matrix`	First element returned by the function *get_heteroplasmy*.
`sum_matrix_qc`	Second element returned by the function *get_heteroplasmy*.
`selected_cells`	Character vector with cells used fro plotting the coverage.
`interactive`	Logical. If TRUE an interactive plot is produced.
`text_size`	Character specifying the size of the text for ggplot2.

Value

ggplot2 object (if *interactive*=FALSE) or plotly object (if *interactive*=TRUE).

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://plotly.com/r/

plot_batch *plot_batch*

Description

plot_batch

Usage

```
plot_batch(position, heteroplasmy_matrix, batch, cluster, text_size, index)
```

Arguments

position	Character name of the base to plot.
heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
batch	Vector of batch names,with length equal to n_row of *heteroplasmy_matrix*.
cluster	Vector specifying a partition of the samples.
text_size	Character specifying the size of the text for ggplot2.
index	Fifth element returned by *get_heteroplasmy*.

Value

ggplot2 object of the heteroplasmy level of a specific base across samples divided according to batch.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

plot_cells_coverage *plot_cells_coverage*

Description

plot_cells_coverage

Usage

```
plot_cells_coverage(sum_matrix, cells_selected, cluster, interactive = FALSE)
```

Arguments

sum_matrix	First element returned by the function *get_heteroplasmy*.
cells_selected	Character vector of cells for which the coverage is computed.
cluster	Character vector with partition information for cells specified in *cells_selected*
interactive	Logical. If TRUE an interactive plot is produced.

Value

ggplot2 object (if *interactive*=FALSE) or plotly object (if *interactive*=TRUE).

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://plotly.com/r/

---

plot_condition *plot_condition*

Description

plot_condition

Usage

```
plot_condition(
  distribution_1,
  distribution_2,
  label_1,
  label_2,
  name_x,
  name_y,
  name_title
)
```

Arguments

distribution_1, distribution_2	Numeric vector
label_1	Character vector of length equal to distribution_1
label_2	Character vector of length equal to distribution_2
name_x	Character name specifying the xlab argument in *ggplot2*.
name_y	Character name specifying the ylab argument in ggplot2.
name_title	Character name specifying the ggtitle argument in *ggplot2*.

Value

ggplot2 boxplot of the quantities specified by *distribution_1* and *distribution_2*, separated by the conditions denoted by *label_1* and *label_2*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

`plot_coordinate_cluster`
plot_coordinate_cluster

Description

plot_coordinate_cluster

Usage

```
plot_coordinate_cluster(coordinate_dm, cluster)
```

Arguments

`coordinate_dm`	Dataframe whit samples on the rows and coordinates names on the columns.
`cluster`	Vector specifying a partition of the samples.

Value

ggplot2 object.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

`plot_coordinate_heteroplasmy`
plot_coordinate_heteroplasmy

Description

plot_coordinate_heteroplasmy

Usage

```
plot_coordinate_heteroplasmy(
  coordinate_dm,
  heteroplasmy_matrix,
  index,
  name_base
)
```

Arguments

coordinate_dm	Dataframe whit samples on the rows and coordinates names on the columns.
heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
index	Fifth element returned by *get_heteroplasmy*.
name_base	Character name specifying the base.

Value

ggplot2 object.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

---

plot_correlation_bases

plot_correlation_bases

Description

plot_correlation_bases

Usage

```
plot_correlation_bases(bases_vector, index, heteroplasmy_matrix)
```

Arguments

bases_vector	Character vector specifying the bases for which the spearman correlation across samples is computed.
index	Fifth element returned by *get_heteroplasmy*.
heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.

Value

Heatmap plot produced by function *Heatmap* from package *ComplexHeatmap*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/ComplexHeatmap/versions/1.10.2/topics/Heatmap

---

plot_distance_matrix *plot_distance_matrix*

Description

plot_distance_matrix

Usage

```
plot_distance_matrix(dist_ang_matrix, cluster)
```

Arguments

dist_ang_matrix	Distance matrix obtained from *clustering_angular_distance* (second element of the output).
cluster	Vector.Can be one of the two partitions returned by function *clustering_angular_distance* (first element of the output).

Value

Heatmap plot produced by function *Heatmap* from package *ComplexHeatmap*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/ComplexHeatmap/versions/1.10.2/topics/Heatmap

plot_distribution *plot_distribution*

Description

plot_distribution

Usage

```
plot_distribution(quantity_counts_cell, name_x, name_title)
```

Arguments

quantity_counts_cell	Vector returned by *get_distribution*
name_x	Character name specifying the xlab argument in *ggplot2*.
name_title	Character name specifying the ggtitle argument in *ggplot2*.

Value

ggplot2 density plot of the vector *quantity_counts_cell*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

plot_dpt *plot_dpt*

Description

plot_dpt

Usage

```
plot_dpt(position, heteroplasmy_matrix, cluster, time, gam_fit_result, index)
```

Arguments

position	Character name of the base to plot.
heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
cluster	Vector specifying a partition of the samples.
time	Vector of diffusion pseudo time,with length equal to n_row of *heteroplasmy_matrix*.
gam_fit_result	Data frame returned by *dpt_test*.
index	Fifth element returned by *get_heteroplasmy*.

Value

ggplot2 object of the heteroplasmy level of a specific base across samples and the GAM fitted curve. The title shows the adjusted p value (FDR) for the position obtained from *get_heteroplasmy*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://cran.r-project.org/package=gam

`plot_genome_coverage` *plot_genome_coverage*

Description

plot_genome_coverage

Usage

```
plot_genome_coverage(biomart_file, path_fasta, chr_name, heteroplasmy_matrix)
```

Arguments

`biomart_file`	Character string with full path to the txt file downloaded from BioMart https://m.ensembl.org/info/data/biomart/index.html . It must have the following five columns:Gene.stable.ID, Gene.name, Gene.start..bp., Gene.end..bp., Chromosome.scaffold.name
`path_fasta`	Character string with full path to the fasta file of the genomic region of interest. It should be the same file used in *get_raw_counts_allele*.
`chr_name`	Character specifying the name of the chromosome of interest. It must be one of the names in the *Chromosome.scaffold.name* column from the *biomart_file*.
`heteroplasmy_matrix`	Third element returned by *get_heteroplasmy*.

Value

KaryoPlot object as returned by *plotKaryotype* function from package *karyoploteR*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

http://bioconductor.org/packages/release/bioc/html/karyoploteR.html

plot_heatmap *plot_heatmap*

Description

plot_heatmap

Usage

```
plot_heatmap(
  new_classification,
  old_classification,
  dist_ang_matrix,
  cluster_columns = FALSE,
  cluster_rows = TRUE,
  name_legend
)
```

Arguments

new_classification	Character vector.Second column of the dataframe returned by function *clustering_angular_distance* (first element of the output).
old_classification	Character vector. First column of the dataframe returned by function *clustering_angular_distance* (first element of the output).
dist_ang_matrix	Distance matrix obtained from *clustering_angular_distance* (second element of the output).
cluster_columns	Logical. Parameter for cluster_columns argument of the function *Heatmap* in the package *ComplexHeatmap*
cluster_rows	Logical. Parameter for cluster_rows argument of the function *Heatmap*
name_legend	Character value.Parameter for name argument of the function *Heatmap*

Value

Heatmap plot produced by function *Heatmap* from package *ComplexHeatmap*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/ComplexHeatmap/versions/1.10.2/topics/Heatmap

plot_heteroplasmy *plot_heteroplasmy*

Description

plot_heteroplasmy

Usage

```
plot_heteroplasmy(position, heteroplasmy_matrix, cluster, index)
```

Arguments

position	Character name of the base to plot.
heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
cluster	Vector specifying a partition of the samples.
index	Fifth element returned by *get_heteroplasmy*.

Value

ggplot2 object of the heteroplasmy level of a specific base across samples divided according to cluster.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

plot_heteroplasmy_variability *plot_heteroplasmy_variability*

Description

plot_heteroplasmy_variability

Usage

```
plot_heteroplasmy_variability(
  heteroplasmy_matrix,
  cluster,
  threshold = 0.1,
  frac = FALSE,
  index
)
```

Arguments

heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
cluster	Vector specifying a partition of the samples.
threshold	Numeric value.
frac	Logical. If FALSE the absolute number of cells that have at least one base with heteroplasmy above *threshold* are shown separated by *cluster*. If TRUE, then the fraction of cells are shown.
index	Fifth element returned by *get_heteroplasmy*.

Value

ggplot2 object.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

---

plot_spider_chart *plot_spider_chart*

Description

plot_spider_chart

Usage

```
plot_spider_chart(name_base, cluster, heteroplasmy_matrix, index)
```

Arguments

name_base	Character name specifying the base.
cluster	Vector specifying a partition of the samples.
heteroplasmy_matrix	Third element returned by *get_heteroplasmy*.
index	Fifth element returned by *get_heteroplasmy*.

Value

radarchart plot produced by function *radarchart* from package *fmsb*.

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://rdrr.io/cran/fmsb/man/radarchart.html

vi_comparison	*vi_comparison We compute the variation of information (VI) between the partition provided by* new_classification *and* old_classification. *The VI between a random partitions (obtained with re-shuffle from original labels in* old_classification*) and* old_classification *is also computed. A distribution of VI values from random partitions is built. Finally, from the comparison with this distribution, an empirical p value is given to the VI of the unsupervised cluster analysis.*

Description

vi_comparison We compute the variation of information (VI) between the partition provided by *new_classification* and *old_classification*. The VI between a random partitions (obtained with re-shuffle from original labels in *old_classification*) and *old_classification* is also computed. A distribution of VI values from random partitions is built. Finally, from the comparison with this distribution, an empirical p value is given to the VI of the unsupervised cluster analysis.

Usage

```
vi_comparison(old_classification, new_classification, number_iter)
```

Arguments

old_classification	Character vector. First column of the dataframe returned by function *clustering_angular_distance* (first element of the output).
new_classification	Character vector.Second column of the dataframe returned by function *clustering_angular_distance* (first element of the output).
number_iter	Integer value. Specify how many random partition are generated (starting from re-shuffle of labels in *old_classification*).

Value

Numeric value (empirical p value).

Author(s)

Gabriele Lubatti <gabriele.lubatti@helmholtz-muenchen.de>

See Also

https://www.rdocumentation.org/packages/mcclust/versions/1.0/topics/vi.dist

Index