Package: DgeaHeatmap (via r-universe)

July 21, 2026

Title Implementation of Functions to Analyze Nanostring and Other Data for DGE and to Generate Heatmaps

Version 0.99.0

Depends R (>= 4.4.0)

Description Package for data extraction from Nanostring GeoMx DSP data, also works for other data. Simple functions for Differential Expression Analysis. User-friendly and highly customizable functions for heatmap generation.

License GPL-3 + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

biocViews Bayesian,Clustering, DifferentialExpression, GeneExpression, Normalization, PrincipalComponent, RNASeq, Regression, Sequencing, Software, Transcription, AlternativeSplicing, BatchEffect,BiomedicalInformatics, CellBiology, Cheminformatics, DataImport, DifferentialSplicing, Epigenetics, ExonArray, FunctionalGenomics, GeneSetEnrichment, Genetics, ImmunoOncology, Metabolomics, MicroRNAArray, Microarray, MultipleComparison, OneChannel, Preprocessing, ProprietaryPlatforms, Proteomics, QualityControl, SystemsBiology, TimeCourse, TwoChannel, mRNAMicroarray

Imports BiocGenerics, ComplexHeatmap, RColorBrewer, purrr, magrittr, rlang, NanoStringNCTools, Biobase, GeomxTools, stats, limma, edgeR, utils, DESeq2, SummarizedExperiment, dplyr

URL https://gitlab.ub.uni-bielefeld.de/spittaulab/Dgea_Heatmap_Package

BugReports https://gitlab.ub.uni-bielefeld.de/spittaulab/Dgea_Heatmap_Package/-/issues

VignetteBuilder knitr

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0), tibble, tidyr, tidyverse, EnvStats, GeoMxWorkflows, VennDiagram, ggiraph, stringr, ggvenn, data.table

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev make libicu-dev libpng-dev libuv1-dev libssl-dev perl zlib1g-dev

Repository https://biocstaging.r-universe.dev

Date/Publication 2026-07-20 09:37:47 UTC

RemoteUrl https://github.com/BiocStaging/DgeaHeatmap

RemoteRef HEAD

RemoteSha 65a509be682ef287f8d7158d8a9555e8ad95e845

Contents

add_demoElem *Build a matrix from an input csv file, with one column as rownames*

Description

Build a matrix from an input csv file, with one column as rownames

Usage

```
add_demoElem(demoData)
```

Arguments

demoData	A NanostringGeoMxSet object.

Value

A NanostringGeoMxSet object.

Examples

```
demoData <- readRDS(
  system.file(
    "extdata",
    "miniGeoMx.rds",
    package = "DgeaHeatmap"
  )
)

demoData <- add_demoElem(demoData)
```

adv_Heatmap *Creating a color scheme based on the available color palettes of RColorBrewer for the heatmap.*

Description

Creating a color scheme based on the available color palettes of RColorBrewer for the heatmap.

Usage

```
adv_Heatmap(
  ncounts_matrix,
  column_name = "Heatmap",
  colorPalette = NULL,
  cluster_method = "hierarchical",
  distance_method = "euclidean",
  cluster_rows = TRUE,
  cluster_columns = FALSE,
  k_row = NULL,
  k_col = NULL,
  sample_metadata = NULL,
  annotation_colors = NULL,
  annotation_name_side = "right",
  show_row_names = FALSE,
  show_column_names = TRUE,
  row_annotation = FALSE,
  row_annotation_method = "auto",
  row_anno_names = NULL,
  row_anno_number = 5,
  fontsize_title = 15,
  fontsize_rowAnnotation = 10,
  fontsize_columnNames = 6,
  fontsize_rowNames = 4,
  fontsize_cluster_labels = 8,
  fontsize_group_annotation = 8,
  fontsize_group_annotation_legend = 10,
  fontsize_group_annotation_labels = 8,
  fontsize_heatmap_legend = 10,
  fontsize_heatmap_legend_labels = 8,
  title_heatmapLegend = "Expression",
  WidthNum = 4.5,
  HeightNum = 3,
  UnitSize = "cm"
)
```

Arguments

ncounts_matrix	An input matrix to create the heatmap.
column_name	A string to set the title of the heatmap, default <- "Heatmap".
colorPalette	Name of the colorPalette used for the heatmap, default <- NULL.
cluster_method	A string setting the cluster method for the heatmap, default <- "hierarchical".
distance_method	A string setting the distance method for clustering, default <- "euclidean".
cluster_rows	A Boolean switching the optional clustering of the rows on and off, default <- TRUE.

`cluster_columns`
: A Boolean switching the optional clustering of the columns on and off, default <- FALSE.

`k_row`
: An integer used to set number of clusters for row clustering in the heatmap, default <- NULL.

`k_col`
: An integer used to set number of clusters for column clustering in the heatmap, default <- NULL.

`sample_metadata`
: A dataframe containing the metadata information for the grouping of the columns, default <- NULL.

`annotation_colors`
: A list assigning choosen colors to the corresponding groups, default <- NULL.

`annotation_name_side`
: The side of the column annotation description, default <- "right".

`show_row_names`
: A Boolean switching rownames on the heatmap on and off, default <- FALSE.

`show_column_names`
: A Boolean switching colum names on the heatmap on and off, default <- TRUE.

`row_annotation`
: A Boolean switching row annotation on the heatmap on and off, default <- FALSE.

`row_annotation_method`
: A string setting the annotation method of the heatmap, default <- "auto".

`row_anno_names`
: A list containing choosen rownames to use for the row annotation, default <- NULL.

`row_anno_number`
: An integer setting the number of automatic annotations assigned per cluster, default <- 5.

`fontsize_title`
: An integer setting the font size of the heatmap title.

`fontsize_rowAnnotation`
: An integer setting the font size of the optional row annotation, default <- 10.

`fontsize_columnNames`
: An integer setting the font size of the column names, default <- 6.

`fontsize_rowNames`
: An integer setting the font size of the row names, default <- 4.

`fontsize_cluster_labels`
: An integer setting the font size of the cluster labels, default <- 8.

`fontsize_group_annotation`
: An integer setting the font size of the group annotation title, default <- 8.

`fontsize_group_annotation_legend`
: An integer setting the font size of the group annotation legend name, default <- 10.

`fontsize_group_annotation_labels`
: An integer setting the font size of the group annotation labels in the legend, default <- 8.

`fontsize_heatmap_legend`
: An integer setting the font size of the heatmap legend title, default <- 10.

`fontsize_heatmap_legend_labels`
: An integer setting the font size of the heatmap legend labels, default <- 8.

`title_heatmapLegend`
: A string setting the changeable title of the legend, default "Expression".

`WidthNum`
: A float setting the width of the heatmap, default <- 4.5.

`HeightNum`
: A float setting the height of the heatmap, default <- 3.

`UnitSize`
: A string such as "cm" or "inch" to set the unit of HeightNum and WidthNum, default <- "cm".

Value

An advanced and customizable heatmap.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
ncounts_matrix <- scale_counts(matrixCounts)
set.seed(1)
adv_Heatmap(ncounts_matrix)
```

---

aExprsDataQC *Function for automatized quality control.*

Description

Function for automatized quality control.

Usage

```
aExprsDataQC(demoData, vFlags)
```

Arguments

`demoData`
: A NanostringGeoMxSet object, storing all expression, annotation, and probe information linked together.

`vFlags`
: Character string to set the flags for quality control.

Value

A matrix.

Examples

```
demoData <- file
vFlags <- "QCFlags"
```

build_matrix *Builds a matrix from an input csv file, with one column as rownames*

Description

Builds a matrix from an input csv file, with one column as rownames

Usage

```
build_matrix(counts_data, x)
```

Arguments

counts_data	A data frame with floats or integers that has strings in one column.
x	Number of column which contains the rownames description.

Value

A matrix.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
```

color_setting *Function to set color scheme for a heatmap.*

Description

Function to set color scheme for a heatmap.

Usage

```
color_setting(colorPalette)
```

Arguments

colorPalette	Chosen color palette.

Value

No return.

Examples

```
colorPalette <- "RdBu"
color_setting(colorPalette)
```

`column_clustering` *Setting the column clustering*

Description

Setting the column clustering

Usage

```
column_clustering(
  ncounts_matrix,
  cluster_columns = TRUE,
  cluster_method = "hierarchical",
  distance_method = "euclidean",
  k_col = NULL,
  col_split = NULL,
  col_dend = TRUE
)
```

Arguments

`ncounts_matrix`	An input matrix to create the clustering.
`cluster_columns`	A Boolean switching the optional clustering of the rows on and off, default <- TRUE.
`cluster_method`	A string setting the cluster method for the heatmap.
`distance_method`	A string setting the distance method for clustering.
`k_col`	An integer used to set number of clusters for row clustering.
`col_split`	An integer seeting the column clustering by kmeans.
`col_dend`	A Boolean switching column clustering by hierarchical clustering on (default).

Value

col_split where the rows are split into clusters

Examples

```
x <- 1
seed <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
ncounts_matrix <- scale_counts(matrixCounts)
set.seed
col_split <- NULL
col_dend <- TRUE
columns_clustered <- column_clustering(ncounts_matrix,
  cluster_columns = TRUE, cluster_method = "hierarchical",
  distance_method = "euclidean", k_col = NULL, col_split = col_split,
  col_dend = col_dend
)
```

`create_contrast_matrix_edgeR` *create_contrast_matrix_edgeR*

Description

Creates a contrast matrix for edgeR

Usage

```
create_contrast_matrix_edgeR(metadata, comparisons = NULL, prefix = "DEA")
```

Arguments

`metadata`	A dataframe containing the meta data information of the raw counts matrix.
`comparisons`	An element of the class list, that contains optional information on specific comparisons that are to be conducted (simplifies the results).
`prefix`	A string used as a prefix for the contrast_names that are generated.

Value

A matrix containing the contrasts for DEA with edgeR.

Examples

```
rawCounts <- read.csv(system.file(
    "extdata/RawDataExamplePackageNanostring.csv",
    package = "DgeaHeatmap"
))
rawCounts <- build_matrix(rawCounts, 1)
metadata <- readRDS(system.file(
```

```
    "extdata/MetaDataPackageNanostring.rds",
    package = "DgeaHeatmap"
))
grouping_columns <- c("segment", "region", "class", "slide_name")
comparisons <- list(
    Comp1 = c(
        "Geometric_Segment_glomerulus_DKD_disease3",
        "PanCK_tubule_DKD_disease4"
    ),
    Comp2 = c("neg_tubule_DKD_disease4", "PanCK_tubule_DKD_disease4")
)
prefix <- "DEA"
results_edgeR <- DGEAedgeR(
    rawCounts, metadata, grouping_columns, comparisons,
    prefix = "DEA"
)
contrast_matrix <- create_contrast_matrix_edgeR(
    metadata = metadata, comparisons = comparisons, prefix = prefix
)
```

DGEADESeq2 *DGEADESeq2*

Description

Performs a differential expression analysis based on DESeq2

Usage

```
DGEADESeq2(rawCounts, metadata, grouping_columns, comparisons)
```

Arguments

rawCounts	A matrix containing raw counts (integers).
metadata	A dataframe containing the meta data information of the raw counts matrix.
grouping_columns	A list of the columns names in the metadata file that are used to generate the groups for the DEA that are compared with each other.
comparisons	An element of the class list, that contains optional information on specific comparisons that are to be conducted (simplifies the results).

Value

A list containing the voom data, fitting of the linear model, results of the contrasts, and the generated design.

Examples

```
set.seed(1)

counts <- matrix(rnbinom(80, mu = 20, size = 1), ncol = 4)
colnames(counts) <- paste0("Sample", 1:4)
rownames(counts) <- paste0("Gene", 1:20)
storage.mode(counts) <- "integer"

metadata <- data.frame(
    segment = factor(rep(c("A", "B"), each = 2)),
    row.names = colnames(counts)
)

comparisons <- list(
    Comp1 = c("A", "B")
)

results_DESeq2 <- DGEADESeq2(
    rawCounts = counts,
    metadata = metadata,
    grouping_columns = "segment",
    comparisons = comparisons
)
```

DGEAedgeR *DGEAedgeR*

Description

Performs a differential expression analysis based on edgeR

Usage

```
DGEAedgeR(
  rawCounts,
  metadata,
  grouping_columns,
  comparisons = NULL,
  prefix = "DEA"
)
```

Arguments

rawCounts	A matrix containing raw counts (integers).
metadata	A dataframe containing the meta data information of the raw counts matrix.
grouping_columns	A list of the columns names in the metadata file that are used to generate groups for the DEA that are compared with each other.

comparisons	An element of the class list, that contains optional information on specific comparisons that are to be conducted (simplifies the results).
prefix	A string used as a prefix for the contrast_names that are generated.

Value

A list containing the voom data, fitting of the linear model, results of the contrasts, and the generated design.

Examples

```
set.seed(1)
counts <- matrix(rnbinom(600, mu = 10, size = 1), ncol = 6)
colnames(counts) <- paste0("Sample", 1:6)
rownames(counts) <- paste0("Gene", 1:100)

# Create simple metadata
metadata <- data.frame(
    segment = factor(rep(c("A", "B"), each = 3)),
    row.names = colnames(counts)
)

comparisons <- list(
    Comp1 = c("A", "B")
)

results_edgeR <- DGEAedgeR(
    rawCounts = counts,
    metadata = metadata,
    grouping_columns = "segment",
    comparisons = comparisons,
    prefix = "DEA"
)

names(results_edgeR)
```

DGEALimma *DGEALimma*

Description

Performs a differential expression analysis based on limma voom

Usage

```
DGEALimma(
  rawCounts,
  metadata,
  grouping_columns,
```

```
  comparisons = NULL,
  prefix = "DEA"
)
```

Arguments

rawCounts	A matrix containing raw counts (integers).
metadata	A dataframe containing the meta data information of the raw counts matrix.
grouping_columns	A list of the columns names in the metadata file that are used to generate the groups for the DEA that are compared with each other.
comparisons	An element of the class list, that contains optional information on specific comparisons that are to be conducted (simplifies the results).
prefix	A string used as a prefix for the contrast_names that are generated.

Value

A list containing the voom data, fitting of the linear model, results of the contrasts, and the generated design.

Examples

```
rawCounts <- read.csv(
    system.file("extdata/RawDataExamplePackageNanostring.csv",
        package = "DgeaHeatmap"
    )
)
rawCounts <- build_matrix(rawCounts, 1)
metadata <- readRDS(system.file("extdata/MetaDataPackageNanostring.rds",
    package = "DgeaHeatmap"
))
grouping_columns <- c("segment", "region", "class", "slide_name")
comparisons <- list(Comp1 = c(
    "Geometric_Segment_glomerulus_DKD_disease3",
    "PanCK_tubule_DKD_disease4"
), Comp2 = c(
    "neg_tubule_DKD_disease4",
    "PanCK_tubule_DKD_disease4"
))
prefix <- "DEA"
DGEAListResults <- DGEALimma(
    rawCounts, metadata, grouping_columns,
    comparisons, prefix
)
```

draw_adv_heatmap *Draw the advanced heatmap*

Description

Draw the advanced heatmap

Usage

```
draw_adv_heatmap(
  ncounts_matrix,
  column_name = "Heatmap",
  colorPalette = NULL,
  col_ha = NULL,
  show_row_names = FALSE,
  show_column_names = TRUE,
  fontsize_title = 15,
  fontsize_columnNames = 6,
  fontsize_rowNames = 4,
  fontsize_cluster_labels = 8,
  fontsize_heatmap_legend = 10,
  fontsize_heatmap_legend_labels = 8,
  title_heatmapLegend = "Expression",
  WidthNum = 4.5,
  HeightNum = 3,
  UnitSize = "cm",
  row_annotation = FALSE,
  annotation_for_rows = NULL,
  row_split = NULL,
  col_split = NULL,
  row_dend = TRUE,
  col_dend = TRUE
)
```

Arguments

ncounts_matrix	An input matrix to create the heatmap.
column_name	A string to set the title of the heatmap.
colorPalette	Name of the colorPalette used for the heatmap, default <- NULL.
col_ha	A "HeamapAnnotation" object if "sample_metadata" is provided, otherwise NULL.
show_row_names	A Boolean switching rownames on the heatmap on and off, default <- FALSE.
show_column_names	A Boolean switching colum names on the heatmap on and off, default <- TRUE.
fontsize_title	An integer setting the font size of the heatmap title, default <- 15.

fontsize_columnNames
: An integer setting the font size of the column names, default <- 6.

fontsize_rowNames
: An integer setting the font size of the row names, default <- 4.

fontsize_cluster_labels
: An integer setting the font size of the cluster labels, default <- 8.

fontsize_heatmap_legend
: An integer setting the font size of the heatmap legend title, default <- 10.

fontsize_heatmap_legend_labels
: An integer setting the font size of the heatmap legend labels, default <- 8.

title_heatmapLegend
: A string setting the changeable title of the legend, default "Expression".

WidthNum
: A float setting the width of the heatmap, default <- 4.5.

HeightNum
: A float setting the height of the heatmap, default <- 3.

UnitSize
: A string such as "cm" or "inch" to set the unit of HeightNum and WidthNum, default <- "cm".

row_annotation
: A Boolean switching row annotation on the heatmap on and off, default <- FALSE.

annotation_for_rows
: Numeric index from matrix for the row annotation

row_split
: An integer seeting the row clustering by kmeans.

col_split
: An integer seeting the column clustering by kmeans.

row_dend
: A Boolean switching row clustering by hierarchical clustering on (default).

col_dend
: A Boolean switching column clustering by hierarchical clustering on (default).

Value

Row annotation

Examples

```
set.seed(1)
mat <- matrix(
  rnorm(25),
  nrow = 5,
  dimnames = list(
    paste0("Gene", 1:5),
    paste0("Sample", 1:5)
  )
)

sample_metadata <- data.frame(
  Group = rep(c("A", "B"), length.out = 5),
  row.names = colnames(mat)
)

group_colors <- list(
```

```
  Group = c(A = "#1b9e77", B = "#7570b3")
)

col_ha <- set_sample_annotation(
  sample_metadata = sample_metadata,
  annotation_colors = group_colors
)

ht <- draw_adv_heatmap(mat)
```

---

elbow_plot *Function to create an elbow plot to choose k for clustering by k-Means.*

Description

Function to create an elbow plot to choose k for clustering by k-Means.

Usage

```
elbow_plot(top_genes_matrix, maxK = 15)
```

Arguments

top_genes_matrix	A matrix for which the best number of clusters in k-Means Clustering is supposed to be calculated.
maxK	An integer determining the maximum number of possible clusters.

Value

An elbow plot to choose k.

Examples

```
x <- 1
seed <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
scaled_counts <- scale_counts(matrixCounts)
top_genes_matrix <- scaled_counts
set.seed(seed)
elbow_plot(top_genes_matrix)
```

`extract_genes_direction` *Get list of genes and their direction of regulation*

Description

Get list of genes and their direction of regulation

Usage

```
extract_genes_direction(
  results_list,
  only_up = FALSE,
  only_down = FALSE,
  only_sig = FALSE,
  padj = NULL,
  padj_cutoff = 0.05,
  lfc_cutoff = 0
)
```

Arguments

`results_list`	A list containing the results of the DEA.
`only_up`	A Boolean opting for only up regulated genes.
`only_down`	A Boolean opting for only down regulated genes.
`only_sig`	A Boolean opting for only significantly regulated genes.
`padj`	A Float, defining the adjusted p-value.
`padj_cutoff`	A Float, setting the adjusted p-values cutoff.
`lfc_cutoff`	A Float, setting the fold change cutoff.

Value

A character vector of contrast names.

Examples

```
rawCounts <- read.csv(
    system.file("extdata/RawDataExamplePackageNanostring.csv",
        package = "DgeaHeatmap"
    )
)
rawCounts <- build_matrix(rawCounts, 1)
metadata <- readRDS(system.file("extdata/MetaDataPackageNanostring.rds",
    package = "DgeaHeatmap"
))
grouping_columns <- c("segment", "region", "class", "slide_name")
```

```
comparisons <- list(
    Comp1 = c(
        "Geometric_Segment_glomerulus_DKD_disease3",
        "PanCK_tubule_DKD_disease4"
    ),
    Comp2 = c("neg_tubule_DKD_disease4", "PanCK_tubule_DKD_disease4")
)
storage.mode(rawCounts) <- "integer"
sum(!is.finite(as.matrix(rawCounts)))
results_list_DESeq2 <- DGEADESeq2(
    rawCounts, metadata, grouping_columns, comparisons
)
results_list <- results_list_DESeq2$results
gene_list <- extract_genes_direction(
    results_list,
    only_up = TRUE, only_down = FALSE, only_sig = FALSE,
    padj = NULL, padj_cutoff = 0.05, lfc_cutoff = 0
)
```

`extractDEGenes` *extractDEGenes*

Description

Extracts the results of the DEA with DESeq2

Usage

```
extractDEGenes(
  results_list,
  comparisons,
  only_up = FALSE,
  only_down = FALSE,
  up_down = FALSE,
  only_sig = FALSE,
  padj_cutoff = 0.05,
  lfc_cutoff = 0
)
```

Arguments

`results_list`	A list containing the results of the DEA.
`comparisons`	An element of the class list, that contains optional information on specific comparisons that are to be conducted (simplifies the results).
`only_up`	A Boolean opting for only up regulated genes.
`only_down`	A Boolean opting for only down regulated genes.
`up_down`	A Boolean opting for only up and down regulated genes.

only_sig	A Boolean opting for only significantly regulated genes.
padj_cutoff	A Float, setting the adjusted p-values cutoff.
lfc_cutoff	A Float, setting the fold change cutoff.

Value

A list containing the voom data, fitting of the linear model, results of the contrasts, and the generated design.

Examples

```
rawCounts <- read.csv(system.file(
    "extdata/RawDataExamplePackageNanostring.csv",
    package = "DgeaHeatmap"
))
rawCounts <- build_matrix(rawCounts, 1)
metadata <- readRDS(system.file(
    "extdata/MetaDataPackageNanostring.rds",
    package = "DgeaHeatmap"
))
grouping_columns <- c("segment", "region", "class", "slide_name")
comparisons <- list(
    Comp1 = c(
        "Geometric_Segment_glomerulus_DKD_disease3",
        "PanCK_tubule_DKD_disease4"
    ),
    Comp2 = c("neg_tubule_DKD_disease4", "PanCK_tubule_DKD_disease4")
)
storage.mode(rawCounts) <- "integer"
sum(!is.finite(as.matrix(rawCounts)))
results_list_DESeq2 <- DGEADESeq2(
    rawCounts, metadata, grouping_columns,
    comparisons
)
results_list <- results_list_DESeq2$results
down_genes <- extractDEGenes(results_list, comparisons, only_down = TRUE)
```

---

filtering_for_top_exprGenes

Function to filter a matrix to extract a chosen number of most variable rows through calculation of the variance.

Description

Function to filter a matrix to extract a chosen number of most variable rows through calculation of the variance.

Usage

```
filtering_for_top_exprGenes(counts_data, top_number_of_genes)
```

Arguments

counts_data	An input matrix from which the most variable rows are to be extracted.
top_number_of_genes	An integer to set the number of how many rows should be extracted from the original matrix by variance.

Value

A new matrix containing only the top_number_of_genes count of rows with the highest variance from input matrix.

Examples

```
x <- 1
top_number_of_genes <- 20
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
counts_data <- matrixCounts
varGenesMatrix <- filtering_for_top_exprGenes(
  counts_data,
  top_number_of_genes
)
```

---

function_complexHeatmap_var

Creating a heatmap with annotation of x most variable rows(genes).

Description

Creating a heatmap with annotation of x most variable rows(genes).

Usage

```
function_complexHeatmap_var(
  topGenes_matrix,
  probes,
  number_of_annotations_per_cluster,
  k,
  Title,
  fontsize_rowAnnotation,
  fontsize_columnNames,
  fontsize_rowNames,
  title_heatmapLegend,
  WidthNum,
  HeightNum,
```

```
  UnitSize,
  color_Palette
)
```

Arguments

topGenes_matrix	An input matrix to create the heatmap.
probes	A list of strings to summarize biological replicated in the columns.
number_of_annotations_per_cluster	An integer used to set the number of annotations per cluster in the heatmap.
k	An integer used to set number of cluster in the heatmap.
Title	A string to set the title of the heatmap.
fontsize_rowAnnotation	An integer used to set the font size of the row annotation in the heatmap
fontsize_columnNames	An integer used to set the font size of the columns in the heatmap.
fontsize_rowNames	An integer used to set the font size of the rownames in the heatmap.
title_heatmapLegend	A string setting the title of the legend of the heatmap.
WidthNum	A float setting the width of the heatmap.
HeightNum	A float setting the height of the heatmap.
UnitSize	A string such as "cm" or "inch" to set the unit of HeightNum and WidthNum.
color_Palette	Name of the colorPalette used for the heatmap.

Value

A plotted heatmap of the input data.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
topGenes_matrix <- scale_counts(matrixCounts)
probes <- list("disease3", "disease4", "disease1B")
k <- 1
seed <- 1
fontsize_rowAnnotation <- 8
number_of_annotations_per_cluster <- 5
Title <- "Heatmap of Data"
fontsize_columnNames <- 6
fontsize_rowNames <- 4
title_heatmapLegend <- "Expression"
```

```
WidthNum <- 4.5
HeightNum <- 3
UnitSize <- "cm"
color_Palette <- "RdBu"
set.seed(seed)
function_complexHeatmap_var(
  topGenes_matrix, probes,
  number_of_annotations_per_cluster, k, Title, fontsize_rowAnnotation,
  fontsize_columnNames, fontsize_rowNames, title_heatmapLegend, WidthNum,
  HeightNum, UnitSize, color_Palette
)
```

---

genRawReadCountTable *Function to generating a raw read count table.*

Description

Function to generating a raw read count table.

Usage

```
genRawReadCountTable(demoData)
```

Arguments

demoData	A NanostringGeoMxSet object, storing all expression, annotation, and probe information linked together.

Value

A dataframe containing only the raw read counts that have passed quality control.

Examples

```
demoData <- readRDS(
  system.file(
    "extdata",
    "miniGeoMx.rds",
    package = "DgeaHeatmap"
  )
)

demoData <- add_demoElem(demoData)

NanoStringNCTools::assayDataApply(
  demoData,
  MARGIN = 1,
  FUN = mean,
  elt = "demoElem"
)[seq_len(5)]
```

```
demoData <- split_data_by_column(
  demoData,
  vGroup = "aoi",
  vElt = "demoElem"
)
df_Exp <- genRawReadCountTable(demoData)
```

`get_dist` *Calculates the distance matrix*

Description

Calculates the distance matrix

Usage

```
get_dist(x, method)
```

Arguments

`x`	An input matrix to create the distance matrix.
`method`	Astring setting the distance method for clustering.

Value

A distance matrix.

Examples

```
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
x <- 1
matrixCounts <- build_matrix(input_data, x)
matrix <- scale_counts(matrixCounts)
method <- "euclidean"
distance_matrix <- get_dist(matrix, method)
```

get_heatmap_colors *Creating a color scheme based on the available color palettes of RColorBrewer for the heatmap.*

Description

Creating a color scheme based on the available color palettes of RColorBrewer for the heatmap.

Usage

```
get_heatmap_colors(colorPalette)
```

Arguments

colorPalette	An input string defining the color palette (available from RColorBrewer).

Value

The colors used in heatmap based on either ComplexHeatmap default or RColorBrewer.

Examples

```
colorPalette <- "RdBu"
get_heatmap_colors(colorPalette)
```

individual_matrix *Creates a matrix only containing chosen columns of an original matrix with more data.*

Description

Creates a matrix only containing chosen columns of an original matrix with more data.

Usage

```
individual_matrix(factors_for_matrix_devision, mmatrix)
```

Arguments

factors_for_matrix_devision	A list containing variables used to extract specific columns from the original matrix.
mmatrix	A matrix used as input, from which specifically chosen columns are extracted.

Value

A new matrix containing fewer columns than before, that have previously been specified.

Examples

```
factors_for_individual_matrix <- list("DKD", "glomerulus")
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
indiMatrix <- individual_matrix(factors_for_individual_matrix, matrixCounts)
```

Kmean_generation *Generates K-means for the columns and rows of a matrix.*

Description

Generates K-means for the columns and rows of a matrix.

Usage

```
Kmean_generation(m_top_genes_matrix, k)
```

Arguments

m_top_genes_matrix	An input matrix for which they K-Means are calculated next.
k	Number k of how many clusters are created.

Value

All calculated K-Means of the matrix.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
m_top_genes_matrix <- scale_counts(matrixCounts)
seed <- 1
k <- 1
set.seed(seed)
k_means <- Kmean_generation(m_top_genes_matrix, k)
```

most_variable_genes *Function to determine the most variable genes of each cluster to enable annotation..*

Description

Function to determine the most variable genes of each cluster to enable annotation..

Usage

```
most_variable_genes(m_kmeans, number_of_annotations_per_cluster, k)
```

Arguments

m_kmeans	Matrix of the k-Means for each row in a gene set.
number_of_annotations_per_cluster	Number of wanted annotations per cluster.
k	Number of clusters.

Value

A list of the most variable rows (genes) of each cluster.

Examples

```
x <- 1
number_of_annotations_per_cluster <- 5
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
m_top_genes_matrix <- scale_counts(matrixCounts)
seed <- 1
k <- 1
set.seed(seed)
m_kmeans <- Kmean_generation(m_top_genes_matrix, k)
top_x_variable_genes <- most_variable_genes(
  m_kmeans,
  number_of_annotations_per_cluster, k
)
```

pairwise_contrasts *Create pairwise contrasts*

Description

Create pairwise contrasts

Usage

```
pairwise_contrasts(comparisons, comp_factor, prefix = "Contrast")
```

Arguments

comparisons	An element of the class list, that contains optional information on specific comparisons that are to be conducted (simplifies the results).
comp_factor	A factor, for fallback all pairwise comparison.
prefix	A string used as a prefix for the contrast_names that are generated.

Value

A character vector of contrast names.

Examples

```
rawCounts <- read.csv(system.file(
    "extdata/RawDataExamplePackageNanostring.csv",
    package = "DgeaHeatmap"
))
rawCounts <- build_matrix(rawCounts, 1)
metadata <- readRDS(system.file(
    "extdata/MetaDataPackageNanostring.rds",
    package = "DgeaHeatmap"
))
grouping_columns <- c("segment", "region", "class", "slide_name")
comparisons <- list(
    Comp1 = c(
        "Geometric_Segment_glomerulus_DKD_disease3",
        "PanCK_tubule_DKD_disease4"
    ),
    Comp2 = c("neg_tubule_DKD_disease4", "PanCK_tubule_DKD_disease4")
)
comp <- apply(
    metadata[, grouping_columns, drop = FALSE], 1, paste,
    collapse = "_"
)
comp <- gsub(" ", "_", comp) # Replace spaces with underscores
metadata$comp <- make.names(comp) # Clean names for use in model.matrix
comp_factor <- factor(metadata$comp)
paired_contrasts <- pairwise_contrasts(comparisons, comp_factor)
```

`performing_kMeans` *Function to perform k-Means clustering for a matrix and setting split to split a heatmap into clusters.*

Description

Function to perform k-Means clustering for a matrix and setting split to split a heatmap into clusters.

Usage

```
performing_kMeans(m_top_genes_matrix, k)
```

Arguments

`m_top_genes_matrix`	A matrix for which K-Means are calculated.
`k`	Number of clusters for k-Means clustering (chosen by elbow-plot).

Value

A new dataframe containing information assigning each rows of the input matrix to a cluster.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
m_top_genes_matrix <- scale_counts(matrixCounts)
k <- 1
split <- performing_kMeans(m_top_genes_matrix, k)
```

`prepare_dge_list` *DGEAedgeR*

Description

Creates a contrast matrix for edgeR

Usage

```
prepare_dge_list(rawCounts, metadata)
```

Arguments

rawCounts	A matrix containing raw counts (integers).
metadata	A dataframe containing the meta data information of the raw counts matrix, with a column containing the comparison factors.

Value

A DGEList object.

Examples

```
rawCounts <- read.csv(system.file(
    "extdata/RawDataExamplePackageNanostring.csv",
    package = "DgeaHeatmap"
))
rawCounts <- build_matrix(rawCounts, 1)
metadata <- readRDS(system.file(
    "extdata/MetaDataPackageNanostring.rds",
    package = "DgeaHeatmap"
))
y <- prepare_dge_list(rawCounts, metadata)
```

---

print_heatmap *Function to build a heatmap using other functions.*

Description

Function to build a heatmap using other functions.

Usage

```
print_heatmap(
  m_top_genes_matrix,
  title,
  split,
  anno,
  fontsize_columnNames,
  fontsize_rowNames,
  title_heatmapLegend,
  WidthNum,
  HeightNum,
  UnitSize,
  color_Palette
)
```

Arguments

m_top_genes_matrix	An input matrix for which the heatmap is created.
title	A string used to set the title of the heatmap.
split	A dataframe containing information about the clustering of the rows of the input matrix.
anno	A numeric index containing row informations for annotation.
fontsize_columnNames	An integer used to set the font size of the columns in the heatmap.
fontsize_rowNames	An integer used to set the font size of the rownames in the heatmap.
title_heatmapLegend	A string setting the title of the legend of the heatmap.
WidthNum	A float setting the width of the heatmap.
HeightNum	A float setting the height of the heatmap.
UnitSize	A string such as "cm" or "inch" to set the unit of HeightNum and WidthNum.
color_Palette	Name of the colorPalette used for the heatmap.

Value

A plotted heatmap.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
m_top_genes_matrix <- scale_counts(matrixCounts)
k <- 1
split <- performing_kMeans(m_top_genes_matrix, k)
seed <- 1
fontsize_rowAnnotation <- 8
set.seed(seed)
m_kmeans <- Kmean_generation(m_top_genes_matrix, k)
number_of_annotations_per_cluster <- 5
top_x_genes_cluster <- most_variable_genes(
  m_kmeans,
  number_of_annotations_per_cluster, k
)
anno <- set_annotation(
  m_top_genes_matrix, top_x_genes_cluster,
  fontsize_rowAnnotation
)
title <- "Heatmap of Data"
fontsize_columnNames <- 6
fontsize_rowNames <- 4
```

```
title_heatmapLegend <- "Expression"
WidthNum <- 4.5
HeightNum <- 3
UnitSize <- "cm"
color_Palette <- "RdBu"
set.seed(seed)
print_heatmap(
  m_top_genes_matrix, title, split, anno, fontsize_columnNames,
  fontsize_rowNames, title_heatmapLegend, WidthNum, HeightNum,
  UnitSize, color_Palette
)
```

`row_clustering` *Setting the row clustering*

Description

Setting the row clustering

Usage

```
row_clustering(
  ncounts_matrix,
  cluster_rows = TRUE,
  cluster_method = "hierarchical",
  distance_method = "euclidean",
  k_row = NULL,
  row_split = NULL,
  row_dend = TRUE
)
```

Arguments

`ncounts_matrix`	An input matrix to create the clustering.
`cluster_rows`	A Boolean switching the optional clustering of the rows on and off, default <- TRUE.
`cluster_method`	A string setting the cluster method for the heatmap.
`distance_method`	A string setting the distance method for clustering.
`k_row`	An integer used to set number of clusters for row clustering.
`row_split`	An integer seeting the column clustering by kmeans.
`row_dend`	A Boolean switching column clustering by hierarchical clustering on (default).

Value

row_split where the rows are split into clusters

Examples

```
x <- 1
seed <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
ncounts_matrix <- scale_counts(matrixCounts)
set.seed
row_split <- NULL
row_dend <- TRUE
rows_clustered <- row_clustering(ncounts_matrix,
  cluster_rows = TRUE,
  cluster_method = "hierarchical", distance_method = "euclidean",
  k_row = NULL, row_split = row_split, row_dend = row_dend
)
```

---

scale_counts *Function to Z-count scale the values of a matrix.*

Description

Function to Z-count scale the values of a matrix.

Usage

```
scale_counts(countsmatrix)
```

Arguments

countsmatrix	An input matrix whose values are scaled by Z-count.

Value

A new matrix with Z-count scaled values.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
countsmatrix <- build_matrix(input_data, x)
scaled_counts <- scale_counts(countsmatrix)
```

set_annotation *Function to set row annotation for a heatmap.*

Description

Function to set row annotation for a heatmap.

Usage

```
set_annotation(m_top_genes_matrix, top_x_genes_cluster, fontsize_rowAnnotation)
```

Arguments

m_top_genes_matrix	An input matrix to used for heatmap generation and choosing annotation for a heatmap.
top_x_genes_cluster	A list of the most variable x genes of each cluster in a heatmap.
fontsize_rowAnnotation	An integer defining the font size of the row annotation

Value

A numeric index from the orginal matrix.

Examples

```
x <- 1
number_of_annotations_per_cluster <- 5
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
m_top_genes_matrix <- scale_counts(matrixCounts)
seed <- 1
k <- 1
fontsize_rowAnnotation <- 8
set.seed(seed)
m_kmeans <- Kmean_generation(m_top_genes_matrix, k)
top_x_genes_cluster <- most_variable_genes(
  m_kmeans,
  number_of_annotations_per_cluster, k
)
anno <- set_annotation(
  m_top_genes_matrix, top_x_genes_cluster,
  fontsize_rowAnnotation
)
```

set_row_annotation *Setting the row annotation*

Description

Setting the row annotation

Usage

```
set_row_annotation(
  ncounts_matrix,
  k_row = NULL,
  row_annotation = FALSE,
  row_annotation_method = "auto",
  row_anno_names = NULL,
  row_anno_number = 5,
  fontsize_rowAnnotation = 10
)
```

Arguments

ncounts_matrix	An input matrix to create the heatmap.
k_row	An integer used to set number of clusters for row clustering in the heatmap, default <- NULL.
row_annotation	A Boolean switching row annotation on the heatmap on and off, default <- FALSE.
row_annotation_method	A string setting the annotation method of the heatmap, default <- "auto".
row_anno_names	A list containing choosen rownames to use for the row annotation, default <- NULL.
row_anno_number	An integer setting the number of automatic annotations assigned per cluster, default <- 5.
fontsize_rowAnnotation	An integer setting the font size of the group annotation labels in the legend, default <- 8.

Value

Numeric index from matrix for the row annotation

Examples

```
x <- 1
seed <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
```

```
))
matrixCounts <- build_matrix(input_data, x)
ncounts_matrix <- scale_counts(matrixCounts)
set.seed(seed)
annotation_for_rows <- set_row_annotation(ncounts_matrix)
```

`set_sample_annotation` *Setting the sample annotation*

Description

Setting the sample annotation

Usage

```
set_sample_annotation(
  sample_metadata = NULL,
  annotation_colors = NULL,
  annotation_name_side = "right",
  fontsize_group_annotation = 8,
  fontsize_group_annotation_legend = 10,
  fontsize_group_annotation_labels = 8
)
```

Arguments

`sample_metadata`
: A dataframe containing the metadata information for the grouping of the columns, default <- NULL.

`annotation_colors`
: A list assigning choosen colors to the corresponding groups, default <- NULL.

`annotation_name_side`
: The side of the column annotation description, default <- "right".

`fontsize_group_annotation`
: An integer setting the font size of the group annotation title, default <- 8.

`fontsize_group_annotation_legend`
: An integer setting the font size of the group annotation legend name, default <- 10.

`fontsize_group_annotation_labels`
: An integer setting the font size of the group annotation labels in the legend, default <- 8.

Value

A "HeamapAnnotation" object if "sample_metadata" is provided, otherwise NULL.

Examples

```
x <- 1
seed <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
ncounts_matrix <- scale_counts(matrixCounts)
set.seed
groups <- c(
  "3_DKD_glomerulus_Geometric_S", "1B_DKD_glomerulus_Geometric_S",
  "2B_DKD_glomerulus_WT"
)
sample_names <- c(colnames(ncounts_matrix))
group_assignment <- sapply(sample_names, function(sample) {
  matched <- groups[sapply(groups, function(g) grepl(g, sample))]
  if (length(matched) > 0) matched[1] else NA
})
stopifnot(length(sample_names) == length(group_assignment))
sample_metadata <- data.frame(
  Group = group_assignment,
  row.names = sample_names
)
all(colnames(ncounts_matrix) == rownames(sample_metadata))
group_colors <- list(Group = c(
  "3_DKD_glomerulus_Geometric_S" = "#1b9e77",
  "1B_DKD_glomerulus_Geometric_S" = "#7570b3",
  "2B_DKD_glomerulus_WT" = "#e7298a"
))
col_ha <- set_sample_annotation(
  sample_metadata = sample_metadata,
  annotation_colors = group_colors
)
```

---

show_data_distribution

Function to visualize the data distribution within the data of a matrix.

Description

Function to visualize the data distribution within the data of a matrix.

Usage

```
show_data_distribution(scaled_counts)
```

Arguments

scaled_counts	An input matrix with Z-count scaled values.

Value

A plot visualizing the data distribution of Z-count scaled data.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
matrixCounts <- build_matrix(input_data, x)
scaled_counts <- scale_counts(matrixCounts)
show_data_distribution(scaled_counts)
```

---

split_data_by_column *Splitting data by group column with feature, pheno or protocol data to then get the mean.*

Description

Splitting data by group column with feature, pheno or protocol data to then get the mean.

Usage

```
split_data_by_column(demoData, vGroup, vElt)
```

Arguments

demoData	A NanostringGeoMxSet object, storing all expression, annotation, and probe information linked together.
vGroup	Group of features in column.
vElt	Character string of the expression matrix name.

Value

A NanostringGeoMxSet object.

Examples

```
demoData <- readRDS(
  system.file(
    "extdata",
    "miniGeoMx.rds",
    package = "DgeaHeatmap"
  )
)

demoData <- add_demoElem(demoData)
```

```
NanoStringNCTools::assayDataApply(
  demoData,
  MARGIN = 1,
  FUN = mean,
  elt = "demoElem"
)[seq_len(5)]

demoData <- split_data_by_column(
  demoData,
  vGroup = "aoi",
  vElt = "demoElem"
)
```

`summarise_bio_replicates` *Summarizes columns biological replicates of a matrix into one.*

Description

Summarizes columns biological replicates of a matrix into one.

Usage

```
summarise_bio_replicates(top_genes_matrix, probes)
```

Arguments

`top_genes_matrix`	An input matrix with columns representing biological replicates.
`probes`	List of strings by which columns of biological replicates can be identified.

Value

A new matrix where biological replicated are summarized as their mean.

Examples

```
x <- 1
input_data <- read.csv(system.file("extdata/testfile_counts.csv",
  package = "DgeaHeatmap"
))
top_genes_matrix <- build_matrix(input_data, x)
probes <- list("disease3", "disease4", "disease1B")
newMatrix <- summarise_bio_replicates(top_genes_matrix, probes)
```

summarize_edgeR_DEA *Summarize the results of the DEA with edgeR*

Description

Summarize the results of the DEA with edgeR

Usage

```
summarize_edgeR_DEA(
  results_edgeR,
  lfc_threshold = 1,
  fdr_threshold = 0.05,
  return_significant_genes = TRUE
)
```

Arguments

results_edgeR	A list containing the results of the DEA.
lfc_threshold	A Float, setting threshold value for the fold change.
fdr_threshold	A Float, setting the False Discovery Rate threshold.
return_significant_genes	A Boolean opting for the output of significantly regulated genes.

Value

A list containing the voom data, fitting of the linear model, results of the contrasts, and the generated design.

Examples

```
results_edgeR <- readRDS(
    system.file(
        "extdata",
        "mini_edgeR_results.rds",
        package = "DgeaHeatmap"
    )
)

summarize_edgeR_DEA(results_edgeR)
```

Index