Package 'clustifyr'

October 17, 2020

Title Classifier for Single-cell RNA-seq Using Cell Clusters

Version 1.0.0

Description Package designed to aid in classifying cells from single-cell RNA sequencing data using external reference data (e.g., bulk RNA-seq, scRNA-seq, microarray, gene lists). A variety of correlation based methods and gene list enrichment methods are provided to assist cell type assignment.

License MIT + file LICENSE

Depends R (>= 3.6)

Imports cowplot, dplyr, entropy, fgsea, ggplot2, Matrix, readr, rlang, scales, stringr, tibble, tidyr, stats, methods, SingleCellExperiment, SummarizedExperiment, matrixStats, S4Vectors

Suggests ComplexHeatmap, covr, knitr, rmarkdown, testthat, ggrepel, BiocStyle

biocViews SingleCell, Annotation, Sequencing, Microarray

BugReports https://github.com/rnabioco/clustifyr/issues

URL http://github.com/rnabioco/clustifyr#readme,
https://rnabioco.github.io/clustifyr/

VignetteBuilder knitr

ByteCompile true

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.0

LazyData true

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

git_branch RELEASE_3_11

git_last_commit bede830

git_last_commit_date 2020-04-27

Date/Publication 2020-10-16

Author Rui Fu [aut, cre],
Kent Riemondy [aut],
RNA Bioscience Initiative [fnd],

Austin Gillen [ctb],
Chengzhe Tian [ctb],
Jay Hesselberth [ctb],
Yue Hao [ctb],
Michelle Daya [ctb]

Maintainer Rui Fu <raysinensis@gmail.com>

R topics documented:

assign_ident *manually change idents as needed*

Description

manually change idents as needed

Usage

```
assign_ident(
  metadata,
  cluster_col = "cluster",
  ident_col = "type",
  clusters,
  idents
)
```

Arguments

metadata column of ident

cluster_col column in metadata containing cluster info

ident_col column in metadata containing identity assignment

clusters names of clusters to change, string or vector of strings

idents new idents to assign, must be length of 1 or same as clusters

Value

new dataframe of metadata

average_clusters *Average expression values per cluster*

Description

Average expression values per cluster

Usage

```
average_clusters(
  mat,
  metadata,
  cluster_col = "cluster",
  if_log = TRUE,
  cell_col = NULL,
  low_threshold = 0,
  method = "mean",
  output_log = TRUE,
  subclusterpower = 0,
  cut_n = NULL
)
```

Arguments

mat expression matrix

metadata data.frame or vector containing cluster assignments per cell. Order must match column order in supplied matrix. If a data.frame provide the cluster_col parameters.

cluster_col column in metadata with cluster number

if_log input data is natural log, averaging will be done on unlogged data

cell_col if provided, will reorder matrix first

low_threshold option to remove clusters with too few cells

method whether to take mean (default) or median

output_log whether to report log results

subclusterpower whether to get multiple averages per original cluster

cut_n set on a limit of genes as expressed, lower ranked genes are set to 0, considered unexpressed

Value

average expression matrix, with genes for row names, and clusters for column names

Examples

```
mat <- average_clusters(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    if_log = FALSE
)
mat[1:3, 1:3]
```

`binarize_expr` *Binarize scRNAseq data*

Description

Binarize scRNAseq data

Usage

```
binarize_expr(mat, n = 1000, cut = 0)
```

Arguments

`mat`	single-cell expression matrix
`n`	number of top expressing genes to keep
`cut`	cut off to set to 0

Value

matrix of 1s and 0s

Examples

```
pbmc_avg <- average_clusters(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified"
)

mat <- binarize_expr(pbmc_avg)
mat[1:3, 1:3]
```

calculate_pathway_gsea

Convert expression matrix to GSEA pathway scores (would take a similar place in workflow before average_clusters/binarize)

Description

Convert expression matrix to GSEA pathway scores (would take a similar place in workflow before average_clusters/binarize)

Usage

```
calculate_pathway_gsea(
  mat,
  pathway_list,
  n_perm = 1000,
  scale = TRUE,
  no_warnings = TRUE
)
```

Arguments

mat	expression matrix
pathway_list	a list of vectors, each named for a specific pathway, or dataframe
n_perm	Number of permutation for fgsea function. Defaults to 1000.
scale	convert expr_mat into zscores prior to running GSEA?, default = FALSE
no_warnings	suppress warnings from gsea ties

Value

matrix of GSEA NES values, cell types as row names, pathways as column names

Examples

```
gl <- list(
    "n" = c("PPBP", "LYZ", "S100A9"),
    "a" = c("IGLL5", "GNLY", "FTL")
)

pbmc_avg <- average_clusters(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified"
)

calculate_pathway_gsea(
    mat = pbmc_avg,
    pathway_list = gl
)
```

`calc_similarity` *compute similarity*

Description

compute similarity

Usage

```
calc_similarity(query_mat, ref_mat, compute_method, rm0 = FALSE, ...)
```

Arguments

`query_mat`	query data matrix
`ref_mat`	reference data matrix
`compute_method`	method(s) for computing similarity scores
`rm0`	consider 0 as missing data, recommended for per_cell
`...`	additional parameters

Value

matrix of numeric values

`call_consensus` *get concensus calls for a list of cor calls*

Description

get concensus calls for a list of cor calls

Usage

```
call_consensus(list_of_res)
```

Arguments

`list_of_res`	list of call dataframes from cor_to_call_rank

Value

dataframe of cluster, new ident, and mean rank

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    ref_mat = cbmc_ref
)

res2 <- cor_to_call_rank(res, threshold = "auto")
res3 <- cor_to_call_rank(res)
call_consensus(list(res2, res3))
```

call_to_metadata *Insert called ident results into metadata*

Description

Insert called ident results into metadata

Usage

```
call_to_metadata(
  res,
  metadata,
  cluster_col,
  per_cell = FALSE,
  rename_prefix = NULL
)
```

Arguments

res	dataframe of idents, such as output of cor_to_call
metadata	input metadata with tsne or umap coordinates and cluster ids
cluster_col	metadata column, can be cluster or cellid
per_cell	whether the res dataframe is listed per cell
rename_prefix	prefix to add to type and r column names

Value

new metadata with added columns

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    ref_mat = cbmc_ref
)
```

```
res2 <- cor_to_call(res, cluster_col = "classified")

call_to_metadata(
    res = res2,
    metadata = pbmc_meta,
    cluster_col = "classified",
    rename_prefix = "assigned"
)
```

cbmc_m — *reference marker matrix from seurat citeseq CBMC tutorial*

Description

reference marker matrix from seurat citeseq CBMC tutorial

Usage

```
cbmc_m
```

Format

An object of class `data.frame` with 3 rows and 13 columns.

Source

https://satijalab.org/seurat/v3.0/multimodal_vignette.html#identify-differentially-expressed-pro

See Also

Other data: `cbmc_ref`, `downrefs`, `object_loc_lookup`, `pbmc_markers_M3Drop`, `pbmc_markers`, `pbmc_matrix_small`, `pbmc_meta`, `pbmc_vargenes`, `s_small3`, `s_small`, `sce_small`

cbmc_ref — *reference matrix from seurat citeseq CBMC tutorial*

Description

reference matrix from seurat citeseq CBMC tutorial

Usage

```
cbmc_ref
```

Format

An object of class `matrix` with 2000 rows and 13 columns.

Source

https://satijalab.org/seurat/v3.0/multimodal_vignette.html#identify-differentially-expressed-pro

See Also

Other data: cbmc_m, downrefs, object_loc_lookup, pbmc_markers_M3Drop, pbmc_markers, pbmc_matrix_small, pbmc_meta, pbmc_vargenes, s_small3, s_small, sce_small

---

clustify *Compare scRNA-seq data to reference data.*

Description

Compare scRNA-seq data to reference data.

Usage

```
clustify(input, ...)

## Default S3 method:
clustify(
  input,
  ref_mat,
  metadata = NULL,
  cluster_col = NULL,
  query_genes = NULL,
  per_cell = FALSE,
  n_perm = 0,
  compute_method = "spearman",
  verbose = FALSE,
  lookuptable = NULL,
  rm0 = FALSE,
  obj_out = TRUE,
  seurat_out = TRUE,
  rename_prefix = NULL,
  threshold = "auto",
  low_threshold_cell = 10,
  exclude_genes = c(),
  if_log = TRUE,
  ...
)

## S3 method for class 'seurat'
clustify(
  input,
  ref_mat,
  cluster_col = NULL,
  query_genes = NULL,
  per_cell = FALSE,
  n_perm = 0,
  compute_method = "spearman",
  use_var_genes = TRUE,
  dr = "umap",
  seurat_out = TRUE,
  obj_out = TRUE,
  threshold = "auto",
  verbose = FALSE,
  rm0 = FALSE,
  rename_prefix = NULL,
  exclude_genes = c(),
  if_log = TRUE,
  ...
)

## S3 method for class 'Seurat'
clustify(
  input,
  ref_mat,
  cluster_col = NULL,
  query_genes = NULL,
  per_cell = FALSE,
  n_perm = 0,
  compute_method = "spearman",
  use_var_genes = TRUE,
  dr = "umap",
  seurat_out = TRUE,
  obj_out = TRUE,
  threshold = "auto",
  verbose = FALSE,
  rm0 = FALSE,
  rename_prefix = NULL,
  exclude_genes = c(),
  if_log = TRUE,
  ...
)

## S3 method for class 'SingleCellExperiment'
clustify(
  input,
  ref_mat,
  cluster_col = NULL,
  query_genes = NULL,
  per_cell = FALSE,
  n_perm = 0,
  compute_method = "spearman",
  use_var_genes = TRUE,
  dr = "umap",
  seurat_out = TRUE,
  obj_out = TRUE,
  threshold = "auto",
  verbose = FALSE,
  rm0 = FALSE,
  rename_prefix = NULL,
  exclude_genes = c(),
  if_log = TRUE,
  ...
```

```
)
```

Arguments

`input` single-cell expression matrix or Seurat object

`...` additional arguments to pass to compute_method function

`ref_mat` reference expression matrix

`metadata` cell cluster assignments, supplied as a vector or data.frame. If data.frame is supplied then `cluster_col` needs to be set. Not required if running correlation per cell.

`cluster_col` column in metadata that contains cluster ids per cell. Will default to first column of metadata if not supplied. Not required if running correlation per cell.

`query_genes` A vector of genes of interest to compare. If NULL, then common genes between the expr_mat and ref_mat will be used for comparision.

`per_cell` if true run per cell, otherwise per cluster.

`n_perm` number of permutations, set to 0 by default

`compute_method` method(s) for computing similarity scores

`verbose` whether to report certain variables chosen

`lookuptable` if not supplied, will look in built-in table for object parsing

`rm0` consider 0 as missing data, recommended for per_cell

`obj_out` whether to output object instead of cor matrix

`seurat_out` output cor matrix or called seurat object (deprecated, use obj_out instead)

`rename_prefix` prefix to add to type and r column names

`threshold` identity calling minimum correlation score threshold, only used when obj_out = TRUE

`low_threshold_cell`
option to remove clusters with too few cells

`exclude_genes` a vector of gene names to throw out of query

`if_log` input data is natural log, averaging will be done on unlogged data

`use_var_genes` if providing a seurat object, use the variable genes (stored in seurat_object@var.genes) as the query_genes.

`dr` stored dimension reduction

Value

single cell object with identity assigned in metadata, or matrix of correlation values, clusters from input as row names, cell types from ref_mat as column names

Examples

```
# Annotate a matrix and metadata
clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified",
    verbose = TRUE
)

# Annotate using a different method
clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified",
    compute_method = "cosine"
)

# Annotate a Seurat object
clustify(
    s_small,
    cbmc_ref,
    cluster_col = "res.1",
    obj_out = TRUE,
    per_cell = FALSE,
    dr = "tsne"
)

# Annotate (and return) a Seurat object per-cell
clustify(
    input = s_small,
    ref_mat = cbmc_ref,
    cluster_col = "res.1",
    obj_out = TRUE,
    per_cell = TRUE,
    dr = "tsne"
)
```

clustifyr_methods *Correlation functions available in clustifyr*

Description

Correlation functions available in clustifyr

Usage

```
clustifyr_methods
```

Format

An object of class `character` of length 5.

clustify_lists *Main function to compare scRNA-seq data to gene lists.*

Description

Main function to compare scRNA-seq data to gene lists.

Usage

```
clustify_lists(input, ...)

## Default S3 method:
clustify_lists(
  input,
  marker,
  marker_inmatrix = TRUE,
  metadata = NULL,
  cluster_col = NULL,
  if_log = TRUE,
  per_cell = FALSE,
  topn = 800,
  cut = 0,
  genome_n = 30000,
  metric = "hyper",
  output_high = TRUE,
  lookuptable = NULL,
  obj_out = TRUE,
  seurat_out = TRUE,
  rename_prefix = NULL,
  threshold = 0,
  low_threshold_cell = 10,
  ...
)

## S3 method for class 'seurat'
clustify_lists(
  input,
  metadata = NULL,
  cluster_col = NULL,
  if_log = TRUE,
  per_cell = FALSE,
  topn = 800,
  cut = 0,
  marker,
  marker_inmatrix = TRUE,
  genome_n = 30000,
  metric = "hyper",
  output_high = TRUE,
  dr = "umap",
  seurat_out = TRUE,
  obj_out = TRUE,
  threshold = 0,
  rename_prefix = NULL,
  ...
)

## S3 method for class 'Seurat'
clustify_lists(
  input,
  metadata = NULL,
  cluster_col = NULL,
  if_log = TRUE,
  per_cell = FALSE,
  topn = 800,
  cut = 0,
  marker,
  marker_inmatrix = TRUE,
  genome_n = 30000,
  metric = "hyper",
  output_high = TRUE,
  dr = "umap",
  seurat_out = TRUE,
  obj_out = TRUE,
  threshold = 0,
  rename_prefix = NULL,
  ...
)

## S3 method for class 'SingleCellExperiment'
clustify_lists(
  input,
  metadata = NULL,
  cluster_col = NULL,
  if_log = TRUE,
  per_cell = FALSE,
  topn = 800,
  cut = 0,
  marker,
  marker_inmatrix = TRUE,
  genome_n = 30000,
  metric = "hyper",
  output_high = TRUE,
  dr = "umap",
  seurat_out = TRUE,
  obj_out = TRUE,
  threshold = 0,
  rename_prefix = NULL,
  ...
)
```

Arguments

`input` single-cell expression matrix or Seurat object

`...` passed to matrixize_markers

`marker` matrix or dataframe of candidate genes for each cluster

`marker_inmatrix` whether markers genes are already in preprocessed matrix form

`metadata` cell cluster assignments, supplied as a vector or data.frame. If data.frame is supplied then `cluster_col` needs to be set. Not required if running correlation per cell.

`cluster_col` column in metadata with cluster number

`if_log` input data is natural log, averaging will be done on unlogged data

`per_cell` compare per cell or per cluster

`topn` number of top expressing genes to keep from input matrix

`cut` expression cut off from input matrix

`genome_n` number of genes in the genome

`metric` adjusted p-value for hypergeometric test, or jaccard index

`output_high` if true (by default to fit with rest of package), -log10 transform p-value

`lookuptable` if not supplied, will look in built-in table for object parsing

`obj_out` whether to output object instead of cor matrix

`seurat_out` output cor matrix or called seurat object (deprecated, use obj_out instead)

`rename_prefix` prefix to add to type and r column names

`threshold` identity calling minimum correlation score threshold, only used when obj_out = T

`low_threshold_cell` option to remove clusters with too few cells

`dr` stored dimension reduction

Value

matrix of numeric values, clusters from input as row names, cell types from marker_mat as column names

Examples

```
# Annotate a matrix and metadata
clustify_lists(
    input = pbmc_matrix_small,
    marker = cbmc_m,
    metadata = pbmc_meta,
    cluster_col = "classified",
    verbose = TRUE
)

# Annotate using a different method
clustify_lists(
    input = pbmc_matrix_small,
    marker = cbmc_m,
    metadata = pbmc_meta,
    cluster_col = "classified",
    verbose = TRUE,
    metric = "jaccard"
)
```

clustify_nudge *Combined function to compare scRNA-seq data to bulk RNA-seq data and marker list*

Description

Combined function to compare scRNA-seq data to bulk RNA-seq data and marker list

Usage

```
clustify_nudge(input, ...)

## Default S3 method:
clustify_nudge(
  input,
  ref_mat,
  marker,
  metadata = NULL,
  cluster_col = NULL,
  query_genes = NULL,
  compute_method = "spearman",
  weight = 1,
  seurat_out = FALSE,
  threshold = -Inf,
  dr = "umap",
  norm = "diff",
  call = TRUE,
  marker_inmatrix = TRUE,
  mode = "rank",
  obj_out = FALSE,
  rename_prefix = NULL,
  lookuptable = NULL,
  ...
)

## S3 method for class 'seurat'
clustify_nudge(
  input,
  ref_mat,
  marker,
  cluster_col = NULL,
  query_genes = NULL,
  compute_method = "spearman",
  weight = 1,
  seurat_out = TRUE,
  obj_out = FALSE,
  threshold = -Inf,
  dr = "umap",
  norm = "diff",
  marker_inmatrix = TRUE,
  mode = "rank",
  rename_prefix = NULL,
  ...
)

## S3 method for class 'Seurat'
clustify_nudge(
  input,
  ref_mat,
  marker,
  cluster_col = NULL,
  query_genes = NULL,
  compute_method = "spearman",
  weight = 1,
  seurat_out = TRUE,
  obj_out = FALSE,
  threshold = -Inf,
  dr = "umap",
  norm = "diff",
  marker_inmatrix = TRUE,
  mode = "rank",
  rename_prefix = NULL,
  ...
)
```

Arguments

`input` express matrix or object

`...` passed to matrixize_markers

`ref_mat` reference expression matrix

`marker` matrix of markers

`metadata` cell cluster assignments, supplied as a vector or data.frame. If data.frame is supplied then `cluster_col` needs to be set.

`cluster_col` column in metadata that contains cluster ids per cell. Will default to first column of metadata if not supplied. Not required if running correlation per cell.

`query_genes` A vector of genes of interest to compare. If NULL, then common genes between the expr_mat and ref_mat will be used for comparision.

`compute_method` method(s) for computing similarity scores

`weight` relative weight for the gene list scores, when added to correlation score

`seurat_out` output cor matrix or called seurat object

`threshold` identity calling minimum score threshold, only used when obj_out = T

`dr` stored dimension reduction

`norm` whether and how the results are normalized

`call` make call or just return score matrix

`marker_inmatrix` whether markers genes are already in preprocessed matrix form

`mode` use marker expression pct or ranked cor score for nudging

`obj_out` whether to output object instead of cor matrix

`rename_prefix` prefix to add to type and r column names

`lookuptable` if not supplied, will look in built-in table for object parsing

Value

single cell object, or matrix of numeric values, clusters from input as row names, cell types from ref_mat as column names

Examples

```
# Seurat2
clustify_nudge(
    input = s_small,
    ref_mat = cbmc_ref,
    marker = cbmc_m,
    cluster_col = "res.1",
    threshold = 0.8,
    seurat_out = FALSE,
    mode = "pct",
    dr = "tsne"
)

# Seurat3
clustify_nudge(
    input = s_small3,
    ref_mat = cbmc_ref,
    marker = cbmc_m,
    cluster_col = "RNA_snn_res.1",
    threshold = 0.8,
    seurat_out = FALSE,
    mode = "pct",
    dr = "tsne"
)

# Matrix
clustify_nudge(
    input = pbmc_matrix_small,
    ref_mat = cbmc_ref,
    metadata = pbmc_meta,
    marker = as.matrix(cbmc_m),
    query_genes = pbmc_vargenes,
    cluster_col = "classified",
    threshold = 0.8,
    call = FALSE,
    marker_inmatrix = FALSE,
    mode = "pct"
)
```

---

collapse_to_cluster *From per-cell calls, take highest freq call in each cluster*

Description

From per-cell calls, take highest freq call in each cluster

Usage

```
collapse_to_cluster(res, metadata, cluster_col, threshold = 0)
```

Arguments

res	dataframe of idents, such as output of cor_to_call
metadata	input metadata with tsne or umap coordinates and cluster ids
cluster_col	metadata column for cluster
threshold	minimum correlation coefficent cutoff for calling clusters

Value

new metadata with added columns

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    ref_mat = cbmc_ref,
    per_cell = TRUE
)

res2 <- cor_to_call(res)

collapse_to_cluster(
    res2,
    metadata = pbmc_meta,
    cluster_col = "classified",
    threshold = 0
)
```

---

compare_lists *Calculate adjusted p-values for hypergeometric test of gene lists or jaccard index*

Description

Calculate adjusted p-values for hypergeometric test of gene lists or jaccard index

Usage

```
compare_lists(
  bin_mat,
  marker_mat,
  n = 30000,
  metric = "hyper",
  output_high = TRUE
)
```

Arguments

bin_mat	binarized single-cell expression matrix, feed in by_cluster mat, if desired
marker_mat	matrix or dataframe of candidate genes for each cluster
n	number of genes in the genome
metric	adjusted p-value for hypergeometric test, or jaccard index
output_high	if true (by default to fit with rest of package), -log10 transform p-value

Value

matrix of numeric values, clusters from expr_mat as row names, cell types from marker_mat as column names

Examples

```
pbmc_mm <- matrixize_markers(pbmc_markers)

pbmc_avg <- average_clusters(
    pbmc_matrix_small,
    pbmc_meta,
    cluster_col = "classified"
)

pbmc_avgb <- binarize_expr(pbmc_avg)

compare_lists(
    pbmc_avgb,
    pbmc_mm,
    metric = "spearman"
)
```

cor_to_call *get best calls for each cluster*

Description

get best calls for each cluster

Usage

```
cor_to_call(
  cor_mat,
  metadata = NULL,
  cluster_col = "cluster",
  collapse_to_cluster = FALSE,
  threshold = 0,
  rename_prefix = NULL,
  carry_r = FALSE
)
```

Arguments

`cor_mat` input similarity matrix

`metadata` input metadata with tsne or umap coordinates and cluster ids

`cluster_col` metadata column, can be cluster or cellid

`collapse_to_cluster` if a column name is provided, takes the most frequent call of entire cluster to color in plot

`threshold` minimum correlation coefficent cutoff for calling clusters

`rename_prefix` prefix to add to type and r column names

`carry_r` whether to include threshold in unassigned names

Value

dataframe of cluster, new ident, and r info

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    ref_mat = cbmc_ref
)

cor_to_call(res)
```

`cor_to_call_rank` *get ranked calls for each cluster*

Description

get ranked calls for each cluster

Usage

```
cor_to_call_rank(
  cor_mat,
  metadata = NULL,
  cluster_col = "cluster",
  collapse_to_cluster = FALSE,
  threshold = 0,
  rename_prefix = NULL,
  top_n = NULL
)
```

Arguments

cor_mat	input similarity matrix
metadata	input metadata with tsne or umap coordinates and cluster ids
cluster_col	metadata column, can be cluster or cellid
collapse_to_cluster	if a column name is provided, takes the most frequent call of entire cluster to color in plot
threshold	minimum correlation coefficent cutoff for calling clusters
rename_prefix	prefix to add to type and r column names
top_n	the number of ranks to keep, the rest will be set to 100

Value

dataframe of cluster, new ident, and r info

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    ref_mat = cbmc_ref
)

cor_to_call_rank(res, threshold = "auto")
```

cor_to_call_topn *get top calls for each cluster*

Description

get top calls for each cluster

Usage

```
cor_to_call_topn(
  cor_mat,
  metadata = NULL,
  col = "cluster",
  collapse_to_cluster = FALSE,
  threshold = 0,
  topn = 2
)
```

Arguments

`cor_mat` input similarity matrix

`metadata` input metadata with tsne or umap coordinates and cluster ids

`col` metadata column, can be cluster or cellid

`collapse_to_cluster` if a column name is provided, takes the most frequent call of entire cluster to color in plot

`threshold` minimum correlation coefficent cutoff for calling clusters

`topn` number of calls for each cluster

Value

dataframe of cluster, new potential ident, and r info

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified"
)

cor_to_call_topn(
    cor_mat = res,
    metadata = pbmc_meta,
    col = "classified",
    collapse_to_cluster = FALSE,
    threshold = 0.5
)
```

cosine *Cosine distance*

Description

Cosine distance

Usage

```
cosine(vec1, vec2)
```

Arguments

`vec1` test vector

`vec2` reference vector

Value

numeric value of cosine distance between the vectors

`downrefs` *table of references stored in clustifyrdata*

Description

table of references stored in clustifyrdata

Usage

```
downrefs
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 9 rows and 6 columns.

Source

various packages

See Also

Other data: `cbmc_m`, `cbmc_ref`, `object_loc_lookup`, `pbmc_markers_M3Drop`, `pbmc_markers`, `pbmc_matrix_small`, `pbmc_meta`, `pbmc_vargenes`, `s_small3`, `s_small`, `sce_small`

`downsample_matrix` *downsample matrix by cluster or completely random*

Description

downsample matrix by cluster or completely random

Usage

```
downsample_matrix(
  mat,
  n = 1,
  keep_cluster_proportions = TRUE,
  metadata = NULL,
  cluster_col = "cluster"
)
```

Arguments

`mat` expression matrix

`n` number per cluster or fraction to keep

`keep_cluster_proportions`
whether to subsample

`metadata` data.frame or vector containing cluster assignments per cell. Order must match column order in supplied matrix. If a data.frame provide the cluster_col parameters.

`cluster_col` column in metadata with cluster number

Value

new smaller mat with less cell_id columns

Examples

```
set.seed(42)
mat <- downsample_matrix(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta$classified,
    n = 10,
    keep_cluster_proportions = TRUE
)
mat[1:3, 1:3]
```

feature_select_PCA *Returns a list of variable genes based on PCA*

Description

Extract genes, i.e. "features", based on the top loadings of principal components formed from the bulk expression data set

Usage

```
feature_select_PCA(
  mat = NULL,
  pcs = NULL,
  n_pcs = 10,
  percentile = 0.99,
  if_log = TRUE
)
```

Arguments

mat	Expression matrix. Rownames are genes, colnames are single cell cluster name, and values are average single cell expression (log transformed).
pcs	Precalculated pcs if available, will skip over processing on mat.
n_pcs	Number of PCs to selected gene loadings from. See the explore_PCA_corr.Rmd vignette for details.
percentile	Select the percentile of absolute values of PCA loadings to select genes from. E.g. 0.999 would select the top point 1 percent of genes with the largest loadings.
if_log	whether the data is already log transformed

Value

vector of genes

Examples

```
feature_select_PCA(
    cbmc_ref,
    if_log = FALSE
)
```

file_marker_parse *takes files with positive and negative markers, as described in garnett, and returns list of markers*

Description

takes files with positive and negative markers, as described in garnett, and returns list of markers

Usage

```
file_marker_parse(filename)
```

Arguments

filename	txt file to load

Value

list of positive and negative gene markers

Examples

```
marker_file <- system.file(
    "extdata",
    "hsPBMC_markers.txt",
    package = "clustifyr"
)

file_marker_parse(marker_file)
```

find_rank_bias *Find rank bias*

Description

Find rank bias

Usage

```
find_rank_bias(
  mat,
  metadata,
  type_col,
  ref_mat,
  query_genes = NULL,
  filter_out = TRUE,
  threshold = 0.33,
  expr_cut = 3000,
  consensus_cut = 1
)
```

Arguments

mat	original query expression matrix
metadata	metadata after calling types
type_col	column name in metadata that contains ids
ref_mat	reference expression matrix
query_genes	original vector of genes used to clustify
filter_out	whether to only report filtered results
threshold	diff threshold
expr_cut	consider all lower expressing genes as off
consensus_cut	filter out if lower han number of types show large diff

Value

matrix of rank diff values

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified"
)
call1 <- cor_to_call(
    res,
    metadata = pbmc_meta,
    cluster_col = "classified",
    collapse_to_cluster = FALSE,
    threshold = 0.8
)
pbmc_meta2 <- call_to_metadata(
    call1,
    pbmc_meta,
    "classified"
)
find_rank_bias(
    pbmc_matrix_small,
    pbmc_meta2, "type",
    cbmc_ref,
    query_genes = pbmc_vargenes
)
```

gene_pct *pct of cells in each cluster that express genelist*

Description

pct of cells in each cluster that express genelist

Usage

```
gene_pct(matrix, genelist, clusters, returning = "mean")
```

Arguments

`matrix`	expression matrix
`genelist`	vector of marker genes for one identity
`clusters`	vector of cluster identities
`returning`	whether to return mean, min, or max of the gene pct in the gene list

Value

vector of numeric values

`gene_pct_markerm` *pct of cells in every cluster that express a series of genelists*

Description

pct of cells in every cluster that express a series of genelists

Usage

```
gene_pct_markerm(matrix, marker_m, metadata, cluster_col = NULL, norm = NULL)
```

Arguments

`matrix`	expression matrix
`marker_m`	matrixized markers
`metadata`	data.frame or vector containing cluster assignments per cell. Order must match column order in supplied matrix. If a data.frame provide the cluster_col parameters.
`cluster_col`	column in metadata with cluster number
`norm`	whether and how the results are normalized

Value

matrix of numeric values, clusters from mat as row names, cell types from marker_m as column names

Examples

```
gene_pct_markerm(
    matrix = pbmc_matrix_small,
    marker_m = cbmc_m,
    metadata = pbmc_meta,
    cluster_col = "classified"
)
```

`get_best_match_matrix` *Function to make best call from correlation matrix*

Description

Function to make best call from correlation matrix

Usage

```
get_best_match_matrix(cor_mat)
```

Arguments

`cor_mat`	correlation matrix

Value

matrix of 1s and 0s

`get_best_str` *Function to make call and attach score*

Description

Function to make call and attach score

Usage

```
get_best_str(name, best_mat, cor_mat, carry_cor = TRUE)
```

Arguments

`name`	name of row to query
`best_mat`	binarized call matrix
`cor_mat`	correlation matrix
`carry_cor`	whether the correlation score gets reported

Value

string with ident call and possibly cor value

get_common_elements *Find entries shared in all vectors*

Description

return entries found in all supplied vectors. If the vector supplied is NULL or NA, then it will be excluded from the comparision.

Usage

```
get_common_elements(...)
```

Arguments

`...`	vectors

Value

vector of shared elements

get_similarity *Compute similarity of matrices*

Description

Compute similarity of matrices

Usage

```
get_similarity(
  expr_mat,
  ref_mat,
  cluster_ids,
  compute_method,
  per_cell = FALSE,
  rm0 = FALSE,
  if_log = TRUE,
  ...
)
```

Arguments

`expr_mat`	single-cell expression matrix
`ref_mat`	reference expression matrix
`cluster_ids`	vector of cluster ids for each cell
`compute_method`	method(s) for computing similarity scores
`per_cell`	run per cell?
`rm0`	consider 0 as missing data, recommended for per_cell
`if_log`	input data is natural log, averaging will be done on unlogged data
`...`	additional parameters not used yet

Value

matrix of numeric values, clusters from expr_mat as row names, cell types from ref_mat as column names

get_unique_column *Generate a unique column id for a dataframe*

Description

Generate a unique column id for a dataframe

Usage

```
get_unique_column(df, id = NULL)
```

Arguments

df	dataframe with column names
id	desired id if unique

Value

character

get_vargenes *Generate variable gene list from marker matrix*

Description

Variable gene list is required for `clustify` main function. This function parses variables genes from a matrix input.

Usage

```
get_vargenes(marker_mat)
```

Arguments

marker_mat	matrix or dataframe of candidate genes for each cluster

Value

vector of marker gene names

Examples

```
get_vargenes(cbmc_m)
```

gmt_to_list *convert gmt format of pathways to list of vectors*

Description

convert gmt format of pathways to list of vectors

Usage

```
gmt_to_list(
  path,
  cutoff = 0,
  sep = "\thttp://www.broadinstitute.org/gsea/msigdb/cards/.*?\t"
)
```

Arguments

path	gmt file path
cutoff	remove pathways with less genes than this cutoff
sep	sep used in file to split path and genes

Value

list of genes in each pathway

Examples

```
gmt_file <- system.file(
    "extdata",
    "c2.cp.reactome.v6.2.symbols.gmt.gz",
    package = "clustifyr"
)

gl <- gmt_to_list(path = gmt_file)
length(gl)
```

insert_meta_object *more flexible metadata update of single cell objects*

Description

more flexible metadata update of single cell objects

Usage

```
insert_meta_object(
  input,
  new_meta,
  type = class(input),
  meta_loc = NULL,
  lookuptable = NULL
)
```

Arguments

`input`	input object
`new_meta`	new metadata table to insert back into object
`type`	look up predefined slots/loc
`meta_loc`	metadata location
`lookuptable`	if not supplied, will look in built-in table for object parsing

Value

new object with new metadata inserted

Examples

```
## Not run:
insert_meta_object(s_small3, seurat_meta(s_small3, dr = "tsne"))

## End(Not run)
```

`kl_divergence` *KL divergence*

Description

Use package entropy to compute Kullback-Leibler divergence. The function first converts each vector's reads to pseudo-number of transcripts by normalizing the total reads to total_reads. The normalized read for each gene is then rounded to serve as the pseudo-number of transcripts. Function entropy::KL.shrink is called to compute the KL-divergence between the two vectors, and the maximal allowed divergence is set to max_KL. Finally, a linear transform is performed to convert the KL divergence, which is between 0 and max_KL, to a similarity score between -1 and 1.

Usage

```
kl_divergence(vec1, vec2, if_log = FALSE, total_reads = 1000, max_KL = 1)
```

Arguments

`vec1`	Test vector
`vec2`	Reference vector
`if_log`	Whether the vectors are log-transformed. If so, the raw count should be computed before computing KL-divergence.
`total_reads`	Pseudo-library size
`max_KL`	Maximal allowed value of KL-divergence.

Value

numeric value, with additional attributes, of kl divergence between the vectors

marker_select *decide for one gene whether it is a marker for a certain cell type*

Description

decide for one gene whether it is a marker for a certain cell type

Usage

```
marker_select(row1, cols, cut = 1, compto = 1)
```

Arguments

row1	a numeric vector of expression values (row)
cols	a vector of cell types (column)
cut	an expression minimum cutoff
compto	compare max expression to the value of next 1 or more

Value

vector of cluster name and ratio value

Examples

```
pbmc_avg <- average_clusters(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified",
    if_log = FALSE
)

marker_select(
    row1 = pbmc_avg["PPBP", ],
    cols = names(pbmc_avg["PPBP", ])
)
```

matrixize_markers *Convert candidate genes list into matrix*

Description

Convert candidate genes list into matrix

Usage

```
matrixize_markers(
  marker_df,
  ranked = FALSE,
  n = NULL,
  step_weight = 1,
  background_weight = 0,
  unique = FALSE,
  metadata = NULL,
  cluster_col = "classified",
  remove_rp = FALSE
)
```

Arguments

marker_df	dataframe of candidate genes, must contain "gene" and "cluster" columns, or a matrix of gene names to convert to ranked
ranked	unranked gene list feeds into hyperp, the ranked gene list feeds into regular corr_coef
n	number of genes to use
step_weight	ranked genes are tranformed into pseudo expression by descending weight
background_weight	ranked genes are tranformed into pseudo expression with added weight
unique	whether to use only unique markers to 1 cluster
metadata	vector or dataframe of cluster names, should have column named cluster
cluster_col	column for cluster names to replace original cluster, if metadata is dataframe
remove_rp	do not include rps, rpl, rp1-9 in markers

Value

matrix of unranked gene marker names, or matrix of ranked expression

Examples

```
matrixize_markers(pbmc_markers)
```

---

not_pretty_palette *black and white palette for plotting continous variables*

Description

black and white palette for plotting continous variables

Usage

```
not_pretty_palette
```

Format

An object of class `character` of length 9.

Value

vector of colors

`object_data` *Function to access object data*

Description

Function to access object data

Usage

```
object_data(object, ...)

## S3 method for class 'seurat'
object_data(object, slot, ...)

## S3 method for class 'Seurat'
object_data(object, slot, ...)

## S3 method for class 'SingleCellExperiment'
object_data(object, slot, ...)
```

Arguments

`object`	object after tsne or umap projections and clustering
`...`	additional arguments
`slot`	data to access

Value

expression matrix, with genes as row names, and cell types as column names

Examples

```
mat <- object_data(
    object = s_small,
    slot = "data"
)
mat[1:3, 1:3]
mat <- object_data(
    object = s_small3,
    slot = "data"
)
mat[1:3, 1:3]
mat <- object_data(
    object = sce_small,
    slot = "data"
)
mat[1:3, 1:3]
```

object_loc_lookup *lookup table for single cell object structures*

Description

lookup table for single cell object structures

Usage

```
object_loc_lookup
```

Format

An object of class `data.frame` with 4 rows and 6 columns.

Source

various packages

See Also

Other data: `cbmc_m`, `cbmc_ref`, `downrefs`, `pbmc_markers_M3Drop`, `pbmc_markers`, `pbmc_matrix_small`, `pbmc_meta`, `pbmc_vargenes`, `s_small3`, `s_small`, `sce_small`

object_ref *Function to convert labelled object to avg expression matrix*

Description

Function to convert labelled object to avg expression matrix

Usage

```
object_ref(input, ...)

## Default S3 method:
object_ref(
  input,
  cluster_col = NULL,
  var_genes_only = FALSE,
  assay_name = NULL,
  method = "mean",
  lookuptable = NULL,
  if_log = TRUE,
  ...
)

## S3 method for class 'Seurat'
object_ref(
  input,
  cluster_col = NULL,
  var_genes_only = FALSE,
  assay_name = NULL,
  method = "mean",
  lookuptable = NULL,
  if_log = TRUE,
  ...
)

## S3 method for class 'seurat'
object_ref(
  input,
  cluster_col = NULL,
  var_genes_only = FALSE,
  assay_name = NULL,
  method = "mean",
  lookuptable = NULL,
  if_log = TRUE,
  ...
)

## S3 method for class 'SingleCellExperiment'
object_ref(
  input,
  cluster_col = NULL,
  var_genes_only = FALSE,
  assay_name = NULL,
  method = "mean",
  lookuptable = NULL,
  if_log = TRUE,
  ...
)
```

Arguments

`input`	object after tsne or umap projections and clustering
`...`	additional arguments
`cluster_col`	column name where classified cluster names are stored in seurat meta data, cannot be "rn"
`var_genes_only`	whether to keep only var.genes in the final matrix output, could also look up genes used for PCA
`assay_name`	any additional assay data, such as ADT, to include. If more than 1, pass a vector of names
`method`	whether to take mean (default) or median
`lookuptable`	if not supplied, will look in built-in table for object parsing
`if_log`	input data is natural log, averaging will be done on unlogged data

Value

reference expression matrix, with genes as row names, and cell types as column names

Examples

```
object_ref(
    s_small3,
    cluster_col = "RNA_snn_res.1"
)
```

overcluster *Overcluster by kmeans per cluster*

Description

Overcluster by kmeans per cluster

Usage

```
overcluster(mat, cluster_id, power = 0.15)
```

Arguments

mat	expression matrix
cluster_id	list of ids per cluster
power	decides the number of clusters for kmeans

Value

new cluster_id list of more clusters

Examples

```
res <- overcluster(
    mat = pbmc_matrix_small,
    cluster_id = split(colnames(pbmc_matrix_small), pbmc_meta$classified)
)
length(res)
```

overcluster_test *compare clustering parameters and classification outcomes*

Description

compare clustering parameters and classification outcomes

Usage

```
overcluster_test(
  expr,
  metadata,
  ref_mat,
  cluster_col,
  x_col = "UMAP_1",
  y_col = "UMAP_2",
  n = 5,
  ngenes = NULL,
  query_genes = NULL,
  threshold = 0,
  do_label = TRUE,
  do_legend = FALSE,
  newclustering = NULL
)
```

Arguments

`expr`	expression matrix
`metadata`	metadata including cluster info and dimension reduction plotting
`ref_mat`	reference matrix
`cluster_col`	column of clustering from metadata
`x_col`	column of metadata for x axis plotting
`y_col`	column of metadata for y axis plotting
`n`	expand n-fold for over/under clustering
`ngenes`	number of genes to use for feature selection, use all genes if NULL
`query_genes`	vector, otherwise genes with be recalculated
`threshold`	type calling threshold
`do_label`	whether to label each cluster at median center
`do_legend`	whether to draw legend
`newclustering`	use kmeans if NULL on dr or col name for second column of clustering

Value

faceted ggplot object

Examples

```
set.seed(42)
overcluster_test(
    expr = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    cluster_col = "classified",
    x_col = "UMAP_1",
    y_col = "UMAP_2"
)
```

parse_loc_object *more flexible parsing of single cell objects*

Description

more flexible parsing of single cell objects

Usage

```
parse_loc_object(
  input,
  type = class(input),
  expr_loc = NULL,
  meta_loc = NULL,
  var_loc = NULL,
  cluster_col = NULL,
  lookuptable = NULL
)
```

Arguments

`input`	input object
`type`	look up predefined slots/loc
`expr_loc`	expression matrix location
`meta_loc`	metadata location
`var_loc`	variable genes location
`cluster_col`	column of clustering from metadata
`lookuptable`	if not supplied, will look in built-in table for object parsing

Value

list of expression, metadata, vargenes, cluster_col info from object

Examples

```
obj <- parse_loc_object(s_small3)
length(obj)
```

pbmc_markers *Marker genes identified by Seurat from single-cell RNA-seq PBMCs.*

Description

Dataframe of markers from Seurat FindAllMarkers function

Usage

```
pbmc_markers
```

Format

An object of class `data.frame` with 2304 rows and 7 columns.

Source

[pbmc_matrix] processed by Seurat

See Also

Other data: `cbmc_m`, `cbmc_ref`, `downrefs`, `object_loc_lookup`, `pbmc_markers_M3Drop`, `pbmc_matrix_small`, `pbmc_meta`, `pbmc_vargenes`, `s_small3`, `s_small`, `sce_small`

`pbmc_markers_M3Drop` *Marker genes identified by M3Drop from single-cell RNA-seq PBMCs.*

Description

Selected features of 3k pbmcs from Seurat3 tutorial

Usage

```
pbmc_markers_M3Drop
```

Format

A data frame with 3 variables:

Source

[pbmc_matrix] processed by [M3Drop]

See Also

Other data: `cbmc_m`, `cbmc_ref`, `downrefs`, `object_loc_lookup`, `pbmc_markers`, `pbmc_matrix_small`, `pbmc_meta`, `pbmc_vargenes`, `s_small3`, `s_small`, `sce_small`

`pbmc_matrix_small` *Matrix of single-cell RNA-seq PBMCs.*

Description

Count matrix of 3k pbmcs from Seurat3 tutorial, with only var.features

Usage

```
pbmc_matrix_small
```

Format

A sparseMatrix with genes as rows and cells as columns.

Source

https://satijalab.org/seurat/v3.0/pbmc3k_tutorial.html

See Also

Other data: cbmc_m, cbmc_ref, downrefs, object_loc_lookup, pbmc_markers_M3Drop, pbmc_markers, pbmc_meta, pbmc_vargenes, s_small3, s_small, sce_small

pbmc_meta *Meta-data for single-cell RNA-seq PBMCs.*

Description

Metadata, including umap, of 3k pbmcs from Seurat3 tutorial

Usage

```
pbmc_meta
```

Format

An object of class `data.frame` with 2638 rows and 9 columns.

Source

[pbmc_matrix] processed by Seurat

See Also

Other data: cbmc_m, cbmc_ref, downrefs, object_loc_lookup, pbmc_markers_M3Drop, pbmc_markers, pbmc_matrix_small, pbmc_vargenes, s_small3, s_small, sce_small

pbmc_vargenes *Variable genes identified by Seurat from single-cell RNA-seq PBMCs.*

Description

Top 2000 variable genes from 3k pbmcs from Seurat3 tutorial

Usage

```
pbmc_vargenes
```

Format

An object of class `character` of length 2000.

Source

[pbmc_matrix] processed by Seurat

See Also

Other data: cbmc_m, cbmc_ref, downrefs, object_loc_lookup, pbmc_markers_M3Drop, pbmc_markers, pbmc_matrix_small, pbmc_meta, s_small3, s_small, sce_small

percent_clusters *Percentage detected per cluster*

Description

Percentage detected per cluster

Usage

```
percent_clusters(mat, metadata, cluster_col = "cluster", cut_num = 0.5)
```

Arguments

mat	expression matrix
metadata	data.frame with cells
cluster_col	column in metadata with cluster number
cut_num	binary cutoff for detection

Value

matrix of numeric values, with genes for row names, and clusters for column names

permute_similarity *Compute a p-value for similarity using permutation*

Description

Permute cluster labels to calculate empirical p-value

Usage

```
permute_similarity(
  expr_mat,
  ref_mat,
  cluster_ids,
  n_perm,
  per_cell = FALSE,
  compute_method,
  rm0 = FALSE,
  ...
)
```

Arguments

`expr_mat`	single-cell expression matrix
`ref_mat`	reference expression matrix
`cluster_ids`	clustering info of single-cell data assume that genes have ALREADY BEEN filtered
`n_perm`	number of permutations
`per_cell`	run per cell?
`compute_method`	method(s) for computing similarity scores
`rm0`	consider 0 as missing data, recommended for per_cell
`...`	additional parameters

Value

matrix of numeric values

`plot_best_call` *Plot best calls for each cluster on a tSNE or umap*

Description

Plot best calls for each cluster on a tSNE or umap

Usage

```
plot_best_call(
  cor_mat,
  metadata,
  cluster_col = "cluster",
  collapse_to_cluster = FALSE,
  threshold = 0,
  x = "UMAP_1",
  y = "UMAP_2",
  plot_r = FALSE,
  per_cell = FALSE,
  ...
)
```

Arguments

`cor_mat`	input similarity matrix
`metadata`	input metadata with tsne or umap coordinates and cluster ids
`cluster_col`	metadata column, can be cluster or cellid
`collapse_to_cluster`	if a column name is provided, takes the most frequent call of entire cluster to color in plot
`threshold`	minimum correlation coefficent cutoff for calling clusters
`x`	x variable

`y`	y variable
`plot_r`	whether to include second plot of cor eff for best call
`per_cell`	whether the cor_mat was generate per cell or per cluster
`...`	passed to plot_dims

Value

ggplot object, cells projected by dr, colored by cell type classification

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified"
)

plot_best_call(
    cor_mat = res,
    metadata = pbmc_meta,
    cluster_col = "classified"
)
```

plot_call *Plot called clusters on a tSNE or umap, for each reference cluster given*

Description

Plot called clusters on a tSNE or umap, for each reference cluster given

Usage

```
plot_call(cor_mat, metadata, data_to_plot = colnames(cor_mat), ...)
```

Arguments

`cor_mat`	input similarity matrix
`metadata`	input metadata with tsne or umap coordinates and cluster ids
`data_to_plot`	colname of data to plot, defaults to all
`...`	passed to plot_dims

Value

list of ggplot object, cells projected by dr, colored by cell type classification

plot_cor *Plot similarity measures on a tSNE or umap*

Description

Plot similarity measures on a tSNE or umap

Usage

```
plot_cor(
  cor_mat,
  metadata,
  data_to_plot = colnames(cor_mat),
  cluster_col = NULL,
  x = "UMAP_1",
  y = "UMAP_2",
  scale_legends = FALSE,
  ...
)
```

Arguments

`cor_mat`	input similarity matrix
`metadata`	input metadata with per cell tsne or umap coordinates and cluster ids
`data_to_plot`	colname of data to plot, defaults to all
`cluster_col`	colname of clustering data in metadata, defaults to rownames of the metadata if not supplied.
`x`	metadata column name with 1st axis dimension. defaults to "UMAP_1".
`y`	metadata column name with 2nd axis dimension. defaults to "UMAP_2".
`scale_legends`	if TRUE scale all legends to maximum values in entire correlation matrix. if FALSE scale legends to maximum for each plot. A two-element numeric vector can also be passed to supply custom values i.e. c(0, 1)
`...`	passed to plot_dims

Value

list of ggplot objects, cells projected by dr, colored by cor values

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified"
)

plot_cor(
    cor_mat = res,
    metadata = pbmc_meta,
    data_to_plot = colnames(res)[1:2],
    cluster_col = "classified",
    x = "UMAP_1",
    y = "UMAP_2"
)
```

plot_cor_heatmap *Plot similarity measures on heatmap*

Description

Plot similarity measures on heatmap

Usage

```
plot_cor_heatmap(
  cor_mat,
  metadata = NULL,
  cluster_col = NULL,
  col = not_pretty_palette,
  legend_title = NULL,
  ...
)
```

Arguments

cor_mat	input similarity matrix
metadata	input metadata with per cell tsne or umap cooordinates and cluster ids
cluster_col	colname of clustering data in metadata, defaults to rownames of the metadata if not supplied.
col	color ramp to use
legend_title	legend title to pass to Heatmap
...	passed to Heatmap

Value

complexheatmap object

Examples

```
res <- clustify(
    input = pbmc_matrix_small,
    metadata = pbmc_meta,
    ref_mat = cbmc_ref,
    query_genes = pbmc_vargenes,
    cluster_col = "classified",
    per_cell = FALSE
)

plot_cor_heatmap(res)
```

plot_dims *Plot a tSNE or umap colored by feature.*

Description

Plot a tSNE or umap colored by feature.

Usage

```
plot_dims(
  data,
  x = "UMAP_1",
  y = "UMAP_2",
  feature = NULL,
  legend_name = "",
  c_cols = pretty_palette2,
  d_cols = NULL,
  pt_size = 0.25,
  alpha_col = NULL,
  group_col = NULL,
  scale_limits = NULL,
  do_label = FALSE,
  do_legend = TRUE,
  do_repel = TRUE
)
```

Arguments

data	input data
x	x variable
y	y variable
feature	feature to color by
legend_name	legend name to display, defaults to no name
c_cols	character vector of colors to build color gradient for continuous values, defaults to clustifyr::pretty_palette
d_cols	character vector of colors for discrete values. defaults to RColorBrewer paired palette
pt_size	point size
alpha_col	whether to refer to data column for alpha values
group_col	group by another column instead of feature, useful for labels
scale_limits	defaults to min = 0, max = max(data$x), otherwise a two-element numeric vector indicating min and max to plot
do_label	whether to label each cluster at median center
do_legend	whether to draw legend
do_repel	whether to use ggrepel on labels

Value

ggplot object, cells projected by dr, colored by feature

Examples

```
plot_dims(
    pbmc_meta,
    feature = "classified"
)
```

---

plot_gene *Plot gene expression on to tSNE or umap*

Description

Plot gene expression on to tSNE or umap

Usage

```
plot_gene(expr_mat, metadata, genes, cell_col = NULL, ...)
```

Arguments

`expr_mat`	input single cell matrix
`metadata`	data.frame with tSNE or umap coordinates
`genes`	gene(s) to color tSNE or umap
`cell_col`	column name in metadata containing cell ids, defaults to rownames if not supplied
`...`	additional arguments passed to [clustifyr::plot_dims()]

Value

list of ggplot object, cells projected by dr, colored by gene expression

Examples

```
genes <- c(
    "RP11-314N13.3",
    "ARF4"
)

plot_gene(
    expr_mat = pbmc_matrix_small,
    metadata = tibble::rownames_to_column(pbmc_meta, "rn"),
    genes = genes,
    cell_col = "rn"
)
```

plot_pathway_gsea *plot GSEA pathway scores as heatmap, returns a list containing results and plot.*

Description

plot GSEA pathway scores as heatmap, returns a list containing results and plot.

Usage

```
plot_pathway_gsea(
  mat,
  pathway_list,
  n_perm = 1000,
  scale = TRUE,
  topn = 5,
  returning = "both"
)
```

Arguments

mat	expression matrix
pathway_list	a list of vectors, each named for a specific pathway, or dataframe
n_perm	Number of permutation for fgsea function. Defaults to 1000.
scale	convert expr_mat into zscores prior to running GSEA?, default = TRUE
topn	number of top pathways to plot
returning	to return "both" list and plot, or either one

Value

list of matrix and plot, or just plot, matrix of GSEA NES values, cell types as row names, pathways as column names

Examples

```
gl <- list(
    "n" = c("PPBP", "LYZ", "S100A9"),
    "a" = c("IGLL5", "GNLY", "FTL")
)

pbmc_avg <- average_clusters(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified"
)

plot_pathway_gsea(
    pbmc_avg,
    gl,
    5
)
```

pos_neg_marker *generate pos and negative marker expression matrix from a list/dataframe of positive markers*

Description

generate pos and negative marker expression matrix from a list/dataframe of positive markers

Usage

```
pos_neg_marker(mat)
```

Arguments

`mat`	matrix or dataframe of markers

Value

matrix of gene expression

Examples

```
m1 <- pos_neg_marker(cbmc_m)
```

pos_neg_select *adapt clustify to tweak score for pos and neg markers*

Description

adapt clustify to tweak score for pos and neg markers

Usage

```
pos_neg_select(
  input,
  ref_mat,
  metadata,
  cluster_col = "cluster",
  cutoff_n = 0,
  cutoff_score = 0.5
)
```

Arguments

`input`	single-cell expression matrix
`ref_mat`	reference expression matrix with positive and negative markers(set expression at 0)
`metadata`	cell cluster assignments, supplied as a vector or data.frame. If data.frame is supplied then `cluster_col` needs to be set. Not required if running correlation per cell.

`cluster_col`	column in metadata that contains cluster ids per cell. Will default to first column of metadata if not supplied. Not required if running correlation per cell.
`cutoff_n`	expression cutoff where genes ranked below n are considered non-expressing
`cutoff_score`	positive score lower than this cutoff will be considered as 0 to not influence scores

Value

matrix of numeric values, clusters from input as row names, cell types from ref_mat as column names

Examples

```
pn_ref <- data.frame(
    "Myeloid" = c(1, 0.01, 0),
    row.names = c("CD74", "clustifyr0", "CD79A")
)

pos_neg_select(
    input = pbmc_matrix_small,
    ref_mat = pn_ref,
    metadata = pbmc_meta,
    cluster_col = "classified",
    cutoff_score = 0.8
)
```

`pretty_palette` *Color palette for plotting continous variables*

Description

Color palette for plotting continous variables

Usage

```
pretty_palette
```

Format

An object of class `character` of length 6.

Value

vector of colors

`pretty_palette2` *Color palette for plotting continous variables, starting at gray*

Description

Color palette for plotting continous variables, starting at gray

Usage

```
pretty_palette2
```

Format

An object of class `character` of length 9.

Value

vector of colors

`pretty_palette_ramp_d` *Expanded color palette ramp for plotting discrete variables*

Description

Expanded color palette ramp for plotting discrete variables

Usage

```
pretty_palette_ramp_d(n)
```

Arguments

`n`	number of colors to use

Value

color ramp

`ref_feature_select` *feature select from reference matrix*

Description

feature select from reference matrix

Usage

```
ref_feature_select(mat, n = 3000, mode = "var", rm.lowvar = TRUE)
```

Arguments

`mat`	reference matrix
`n`	number of genes to return
`mode`	the method of selecting features
`rm.lowvar`	whether to remove lower variation genes first

Value

vector of genes

Examples

```
pbmc_avg <- average_clusters(
    mat = pbmc_matrix_small,
    metadata = pbmc_meta,
    cluster_col = "classified"
)

ref_feature_select(
    mat = pbmc_avg[1:100, ],
    n = 5
)
```

`ref_marker_select` *marker selection from reference matrix*

Description

marker selection from reference matrix

Usage

```
ref_marker_select(mat, cut = 0.5, arrange = TRUE, compto = 1)
```

Arguments

mat	reference matrix
cut	an expression minimum cutoff
arrange	whether to arrange (lower means better)
compto	compare max expression to the value of next 1 or more

Value

dataframe, with gene, cluster, ratio columns

Examples

```
ref_marker_select(
    cbmc_ref,
    cut = 2
)
```

reverse_marker_matrix *generate negative markers from a list of exclusive positive markers*

Description

generate negative markers from a list of exclusive positive markers

Usage

```
reverse_marker_matrix(mat)
```

Arguments

mat	matrix or dataframe of markers

Value

matrix of gene names

Examples

```
reverse_marker_matrix(cbmc_m)
```

run_gsea — *Run GSEA to compare a gene list(s) to per cell or per cluster expression data*

Description

Use fgsea algorithm to compute normalized enrichment scores and pvalues for gene set ovelap

Usage

```
run_gsea(
  expr_mat,
  query_genes,
  cluster_ids = NULL,
  n_perm = 1000,
  per_cell = FALSE,
  scale = FALSE,
  no_warnings = TRUE
)
```

Arguments

`expr_mat`	single-cell expression matrix or Seurat object
`query_genes`	A vector or named list of vectors of genesets of interest to compare via GSEA. If supplying a named list, then the gene set names will appear in the output.
`cluster_ids`	vector of cell cluster assignments, supplied as a vector with order that matches columns in `expr_mat`. Not required if running per cell.
`n_perm`	Number of permutation for fgsea function. Defaults to 1000.
`per_cell`	if true run per cell, otherwise per cluster.
`scale`	convert expr_mat into zscores prior to running GSEA?, default = FALSE
`no_warnings`	suppress warnings from gsea ties

Value

dataframe of gsea scores (pval, NES), with clusters as rownames

Examples

```
run_gsea(
    expr_mat = pbmc_matrix_small,
    query_genes = pbmc_vargenes[1:100],
    n_perm = 10,
    cluster_ids = pbmc_meta$classified,
    no_warnings = TRUE
)
```

`sce_small` *Small SingleCellExperiment object*

Description

Small SingleCellExperiment object

Usage

```
sce_small
```

Format

An object of class `SingleCellExperiment` with 200 rows and 200 columns.

Source

`"https://scrnaseq-public-datasets.s3.amazonaws.com/scater-objects/segerstolpe.rds"`

See Also

Other data: `cbmc_m`, `cbmc_ref`, `downrefs`, `object_loc_lookup`, `pbmc_markers_M3Drop`, `pbmc_markers`, `pbmc_matrix_small`, `pbmc_meta`, `pbmc_vargenes`, `s_small3`, `s_small`

`seurat_meta` *Function to convert labelled seurat object to fully prepared metadata*

Description

Function to convert labelled seurat object to fully prepared metadata

Usage

```
seurat_meta(seurat_object, ...)

## S3 method for class 'seurat'
seurat_meta(seurat_object, dr = "umap", ...)

## S3 method for class 'Seurat'
seurat_meta(seurat_object, dr = "umap", ...)
```

Arguments

`seurat_object`	seurat_object after tsne or umap projections and clustering
`...`	additional arguments
`dr`	dimension reduction method

Value

dataframe of metadata, including dimension reduction plotting info

Examples

```
## Not run:
seurat_meta(s_small)

## End(Not run)
```

`seurat_ref` *Function to convert labelled seurat object to avg expression matrix*

Description

Function to convert labelled seurat object to avg expression matrix

Usage

```
seurat_ref(seurat_object, ...)

## S3 method for class 'seurat'
seurat_ref(
  seurat_object,
  cluster_col = "classified",
  var_genes_only = FALSE,
  assay_name = NULL,
  method = "mean",
  subclusterpower = 0,
  if_log = TRUE,
  ...
)

## S3 method for class 'Seurat'
seurat_ref(
  seurat_object,
  cluster_col = "classified",
  var_genes_only = FALSE,
  assay_name = NULL,
  method = "mean",
  subclusterpower = 0,
  if_log = TRUE,
  ...
)
```

Arguments

`seurat_object`	seurat_object after tsne or umap projections and clustering
`...`	additional arguments
`cluster_col`	column name where classified cluster names are stored in seurat meta data, cannot be "rn"
`var_genes_only`	whether to keep only var_genes in the final matrix output, could also look up genes used for PCA

assay_name	any additional assay data, such as ADT, to include. If more than 1, pass a vector of names
method	whether to take mean (default) or median
subclusterpower	whether to get multiple averages per original cluster
if_log	input data is natural log, averaging will be done on unlogged data

Value

reference expression matrix, with genes as row names, and cell types as column names

Examples

```
ref <- seurat_ref(
    seurat_object = s_small,
    cluster_col = "res.1",
    var_genes_only = TRUE
)
ref[1:3, 1:3]
```

s_small   *Small clustered Seurat2 object*

Description

Small clustered Seurat2 object

Usage

```
s_small
```

Format

An object of class seurat of length 1.

Source

[pbmc_small] processed by seurat

See Also

Other data: cbmc_m, cbmc_ref, downrefs, object_loc_lookup, pbmc_markers_M3Drop, pbmc_markers, pbmc_matrix_small, pbmc_meta, pbmc_vargenes, s_small3, sce_small

s_small3 *Small clustered Seurat3 object*

Description

Small clustered Seurat3 object

Usage

```
s_small3
```

Format

An object of class Seurat of length 1.

Source

[pbmc_small] processed by Seurat

See Also

Other data: cbmc_m, cbmc_ref, downrefs, object_loc_lookup, pbmc_markers_M3Drop, pbmc_markers, pbmc_matrix_small, pbmc_meta, pbmc_vargenes, s_small, sce_small

vector_similarity *Compute similarity between two vectors*

Description

Compute the similarity score between two vectors using a customized scoring function Two vectors may be from either scRNA-seq or bulk RNA-seq data. The lengths of vec1 and vec2 must match, and must be arranged in the same order of genes. Both vectors should be provided to this function after pre-processing, feature selection and dimension reduction.

Usage

```
vector_similarity(vec1, vec2, compute_method, ...)
```

Arguments

vec1	test vector
vec2	reference vector
compute_method	method to run i.e. corr_coef
...	arguments to pass to compute_method function

Value

numeric value of desired correlation or distance measurement

write_meta *Function to write metadata to object*

Description

Function to write metadata to object

Usage

```
write_meta(object, ...)

## S3 method for class 'seurat'
write_meta(object, meta, ...)

## S3 method for class 'Seurat'
write_meta(object, meta, ...)

## S3 method for class 'SingleCellExperiment'
write_meta(object, meta, ...)
```

Arguments

`object`	object after tsne or umap projections and clustering
`...`	additional arguments
`meta`	new metadata dataframe

Value

object with newly inserted metadata columns

Examples

```
obj <- write_meta(
    object = s_small,
    meta = seurat_meta(s_small)
)
obj <- write_meta(
    object = s_small3,
    meta = seurat_meta(s_small3)
)
obj <- write_meta(
    object = sce_small,
    meta = object_data(sce_small, "meta.data")
)
```

Index