Package ‘scrattch.mapping’

March 14, 2023

Title Generalized mapping of annotations from shiny taxonomy to query data.

Version 0.1

Description ADD

License GPL-3

Depends feather,
anndata,
dplyr,
dendextend,
viridis,
feather,
tibble,
Matrix,
MatrixGenerics,
foreach,
pvclust,
scrattch.io,
scrattch.hicat,
scrattch.bigcat,
scrattch.vis,
mfishtools,
patchseqtools,
patchSeqQC,
Seurat,
cowplot,
umap,
arrow

Suggests knitr,
rmarkdown,
testthat,
future,
parallel,
doMC

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

R topics documented:

addDendrogramMarkers *Add marker genes to reference dendrogram for tree mapping*

Description

Add marker genes to reference dendrogram for tree mapping

Usage

```
addDendrogramMarkers(
  dend,
  norm.data,
  metadata,
  celltypeColumn = "cluster_label",
  subsample = 100,
  num.markers = 20,
 de.param = scrattch.hicat::de_param(low.th = 1, padj.th = 0.01, lfc.th = 1, q1.th =
  0.3, q2.th = NULL, q.diff.th = 0.7, de.score.th = 100, min.cells = 2, min.genes = 5),
  calculate.de.genes = TRUE,
  save.shiny.output = TRUE,
  mc.cores = 1,
  bs.num = 100,
  p = 0.7,
  low.th = 0.15,
  shinyFolder = paste0(getwd(), "/")
)
```

Arguments

`dend` A dendrogram in R format to which marker genes will be added, or a character string with a file location of "dend.RData"

`norm.data` A matrix of log normalized reference data, or character string with a file location of "data_t.feather". If a count matrix is provided, the data data will be log normalized. This should be the data matrix used to generate the dendrogram.

`metadata` Data frame of metadata with rows corresponding to cells/nuclei, and either row names or a column called "sample_id" corresponding to cell names. This matrix must include entries for all cells in norm.data. Could also be a file. Columns can be numeric, categorical, or factors.

`celltypeColumn` Column name correspond to the cell type names in the dendrogram (default = "cluster_label"). At least two cells per cell type in the dendrogram must be included.

`subsample` The number of cells to retain per cluster (default = 100)

`num.markers` The maximum number of markers to calculate per pairwise differential calculation per direction (default = 20)

`de.param` Differential expression (DE) parameters for genes and clusters used to define marker genes. By default the values are set to the 10x nuclei defaults from scrattch.hicat, except with min.cells=2 (see the function `de_param` in the scrattch.hicat for more details).

`calculate.de.genes`

Default=TRUE. If set to false, the function will search for a file called "de.genes.rda" to load precalculated de genes.

`save.shiny.output`

Should standard output files be generated and saved to the directory (default=TRUE). These are not required for tree mapping, but are required for building a patch-seq shiny instance. This is only tested in a UNIX environment. See notes.

`mc.cores` Number of cores to use for running this function to speed things up. Default = 1. Values>1 are only supported in an UNIX environment and require `foreach` and `doParallel` R libraries.

`bs.num, p, low.th`

Extra variables for the `map_dend_membership` function in scrattch.hicat. Defaults are set reasonably.

`shinyFolder` The location to save shiny output, if desired

NOTES

If save.shiny.output=TRUE, the following files will be generated: reference.rda, which includes a variable `reference` as follows: reference$cl.dat - These are the cluster means that are used for mapping comparisons reference$dend - This is the dendrogram with marker genes attached membership_information_reference.rda, which includes two variables `memb.ref` - matrix indicating how much confusion there is the mapping between each cell all of the nodes in the tree (including all cell types) when comparing clustering and mapping results with various subsamplings of the data `map.df.ref` - Result of tree mapping for each cell in the reference against the clustering tree, including various statistics and marker gene evidence. This is the same output that comes from tree mapping.#'

Value

An updated dendrogram variable that is the same as `dend` except with marker genes added to each node.

applyPatchseqQC *This function applies patchseqQC, given a taxonomy and query data*

Description

This function applies patchseqQC, given a taxonomy and query data

Usage

```
applyPatchseqQC(AIT.anndata, query.data, query.metadata, verbose = FALSE)
```

Arguments

AIT.anndata	A reference taxonomy object.
query.data	A logCPM normalized matrix to be annotated.
query.metadata	A data frame of metadata for the query data.
verbose	Should status be printed to the screen?

Value

A new query.metadata file with appended QC columns

buildMappingDirectory *Starting from an anndata object this function builds the minimum files required for patch-seq shiny*

Description

Starting from an anndata object this function builds the minimum files required for patch-seq shiny

Usage

```
buildMappingDirectory(
  AIT.anndata,
  mappingFolder,
  query.data,
  query.metadata,
  query.mapping = NULL,
  doPatchseqQC = TRUE,
  metadata_names = NULL,
  mc.cores = 1,
  bs.num = 100,
  p = 0.7,
  low.th = 0.15,
  min.confidence = 0.5
)
```

Arguments

`AIT.anndata` A reference taxonomy object.

`mappingFolder` The location to save output files for patch-seq (or other query data) results, e.g. "/allen/programs/celltypes/workgroups/rnaseqanalysis/shiny/star/human/human_patchseq_MTG_JA

`query.data` A logCPM normalized matrix to be annotated.

`query.metadata` A data frame of metadata for the query data.

`query.mapping` Mapping results from `taxonomy_mapping()` or other mapping functions (optional). If provided row names must match column names in query.data.

`doPatchseqQC` Boolean indicating whether patch-seq QC metrics should be calculated (default) or not.

`metadata_names` An optional named character vector where the vector NAMES correspond to columns in the metadata matrix and the vector VALUES correspond to how these metadata should be displayed in Shiny. This is used for writing the desc.feather file later.

`mc.cores` Number of cores to use for running this function to speed things up. Default = 1. Values>1 are only supported in an UNIX environment and require `foreach` and `doParallel` R libraries.

`bs.num, p, low.th`

Extra variables for the `map_dend_membership` function in scrattch.hicat. Defaults are set reasonably.

`min.confidence` Probability below which a cell cannot be assigned to a cell type (default 0.7). In other words, if no cell types have probabilties greater than resolution.index, then the assigned cluster will be an internal node of the dendrogram.

This function writes files to the mappingFolder directory for visualization with molgen-shiny tools — anno.feather - query metadata — data.feather - query data — dend.RData - dendrogram (copied from reference) — desc.feather - table indicating which anno columns to share — memb.feather - tree mapping of each query cell to each tree node (not just the best matching type like in treeMap) — tsne.feather - low dimensional coordinates for data — tsne_desc.feather - table indicating which low-D representations to share

`buildTaxonomy` *This function builds the minimum files required for Shiny*

Description

This function builds the minimum files required for Shiny

Usage

```
buildTaxonomy(
  counts,
  meta.data,
  feature.set,
  umap.coords,
  shinyFolder,
  cluster_colors = NULL,
  metadata_names = NULL,
  subsample = 2000,
  reorder.dendrogram = FALSE
)
```

Arguments

`counts` A count matrix in sparse format: dgCMatrix.

`meta.data` Meta.data corresponding to count matrix. Rownames must be equal to colnames of counts.

`feature.set` Set of feature used to calculate dendrogram. Typically highly variable and/or marker genes.

`umap.coords` Dimensionality reduction coordiant data.frame with 2 columns. Rownames must be equal to colnames of counts.

`shinyFolder` The location to save Shiny objects, e.g. "/allen/programs/celltypes/workgroups/rnaseqanalysis/shiny/

`cluster_colors` An optional named character vector where the values correspond to colors and the names correspond to celltypes in celltypeColumn. If this vector is incomplete, a warning is thrown and it is ignored.

`metadata_names` An optional named character vector where the vector NAMES correspond to columns in the metadata matrix and the vector VALUES correspond to how these metadata should be displayed in Shiny. This is used for writing the desc.feather file later.

`subsample` The number of cells to retain per cluster

`reorder.dendrogram`
Should dendogram attempt to match a preset order? (Default = FALSE). If TRUE, the dendrogram attempts to match the celltype factor order as closely as possible (if celltype is a character vector rather than a factor, this will sort clusters alphabetically, which is not ideal).

build_dend *Build dend (updated to specify dendextend version of "set")*

Description

Build dend (updated to specify dendextend version of "set")

Usage

```
build_dend(
  cl.dat,
  cl.cor = NULL,
  l.rank = NULL,
  l.color = NULL,
  nboot = 100,
  ncores = 1
)
```

Arguments

`cl.dat`

`cl.cor`

`l.rank`

`l.color`

`nboot`

`ncores`

Value

dendrogram and a couple of related things

---

build_train_list_on_taxonomy

Starting point for optimized tree mapping

Description

Starting point for optimized tree mapping

Usage

```
build_train_list_on_taxonomy(
  TaxFN = NA,
  Taxonomy,
  pre.train.list = NA,
  query.genes = NA,
  prefix = "",
  mapping.method = c("flat", "hierarchy"),
  prebuild = FALSE,
  newbuild = FALSE,
  mc.cores = 10,
  div_thr = 3,
  subsample_pct = 0.9,
  top.n.genes = 15,
  n.group.genes = 3000,
  rm.cl = c()
)
```

Arguments

`rm.cl`

Value

Mapping results

compactness_distance *Calculate a compactness score*

Description

This function calculates the compactness score, defined as the the average (Pearson) correlation-based distance between each cell and the assigned group centroid (median) using the variable genes. If a secondary group is provided (e.g., transgenic line, cortical layer, etc.), the function first sets gene expression values for each cell as expression values for the cluster median and then returns compactness per cell summarized by the secondary group.

Usage

```
compactness_distance(
  query.data,
  query.group,
  query.secondary = NULL,
  variable.features = rownames(query.data)
)
```

Arguments

`query.data` A logCPM normalized matrix of the data

`query.group` A group vector to calculate compactness distance over (e.g., cluster assignments)

`query.secondary` An optional secondary group vector for comparison with the primary group vector (e.g., cortical layer or transgenic line)

`variable.features` A precomputed set of variable features. If not provided, all genes are used.

Value

A vector of compactness scores for each cell

compare_heatmap *Compare and plot two sets of cluster assignments as a heatmap*

Description

This is a wrapper for the function `heatmap.3` in scrattch.hicat

Usage

```
compare_heatmap(
  cl,
  ref.cl,
  threshold = 0.2,
  cexLab = NULL,
  Rowv = NA,
  Colv = NA,
  ylab = NULL,
  xlab = NULL,
  main = NULL,
  margins = c(6, 6),
  scale = "none",
  trace = "none",
  dendrogram = "none",
  ...
)
```

Arguments

cl	A cluster factor object to compare to a reference
ref.cl	A cluster factor object for the reference clusters
threshold	Maximum value to show in heatmap. Lower values will highlight off-target expression more.
cexLab	Size of the label names to display on the screen. The function will attempt to guess this if not inputted, adjustments may be needed if not all labels are shown.
...	Other inputs to the function heatmap.3

Value

a list with output from heatmap.3, after displaying the heatmap to the screen.

compare_plot *Compare and plot two sets of cluster assignments*

Description

This is the subset of the compare_annotate function that does the plotting. It is identical to the scrattch.hicat implementation, but a bit more flexible on input formats.

Usage

```
compare_plot(cl, ref.cl)
```

Arguments

cl	A cluster factor object to compare to a reference
ref.cl	A cluster factor object for the reference clusters

Value

g A ggplot2 dot plot object for the comparison.

corrMap *Correlation based mapping*

Description

Correlation based mapping

Usage

```
corrMap(AIT.anndata, query.data)
```

Arguments

`AIT.anndata`	A reference taxonomy anndata object.
`query.data`	A logCPM normalized matrix to be annotated.

Value

Correlation mapping results as a data.frame.

get_cl_medians *Compute cluster medians for each row in a matrix*

Description

Compute cluster medians for each row in a matrix

Usage

```
get_cl_medians(mat, cl)
```

Arguments

`mat`	A gene (rows) x samples (columns) sparse matrix
`cl`	A cluster factor object

Value

a matrix of genes (rows) x clusters (columns) with medians for each cluster

loadTaxonomy *Read in a reference data set in Allen taxonomy format*

Description

Read in a reference data set in Allen taxonomy format

Usage

```
loadTaxonomy(
  refFolder,
  sample_id = "sample_id",
  hGenes = NULL,
  sub.sample = 1000,
  gene_id = "gene"
)
```

Arguments

refFolder	Directory containing the Shiny taxonomy.
sample_id	Field in reference taxonomy that defines the sample_id.
hGenes	User supplied variable gene vector. If not provided, then all genes are used.
sub.sample	Number of cells to keep per cluster.
gene_id	Field in counts.feather that defines the gene_id.

Value

Organized reference object ready for mapping against.

map_dend *Title*

Description

Title

Usage

```
map_dend(
  dend,
  cl.dat,
  map.dat,
  select.cells = colnames(map.dat),
  p = 0.8,
  low.th = 0.2,
  default.markers = NULL,
  seed = 42
)
```

Arguments

dend	
cl.dat	
map.dat	
select.cells	
p	
low.th	
default.markers	
seed	= random seed
cl	
dat	

Value

tree mapping to the dendrogram table (cells x nodes with values as probabilities)

map_dend_membership *Title*

Description

Title

Usage

```
map_dend_membership(
  dend,
  cl.dat,
  map.dat,
  map.cells,
  mc.cores = 10,
  bs.num = 100,
  seed = 42,
  ...
)
```

Arguments

dend	
cl.dat	
map.dat	
map.cells	
mc.cores	
bs.num	
seed	= random seed
...	
cl	
dat	

Value

membership table

prepareTaxonomy *Prepare taxonomy for optimized tree mapping*

Description

This function write ...

Usage

```
prepareTaxonomy(
  count,
  cl,
  cl.df,
  AIT.str,
  lognormal = NULL,

   taxonomy.dir = "/allen/programs/celltypes/workgroups/rnaseqanalysis/shiny/Taxonomies/"
)
```

Arguments

`cl`	assigned cluster
`cl.df`	cluster anno with hierarchy : cluster(cl)/subclass_label/neighborhood/root
`AIT.str`	taxonomy id
`taxonomy.dir`	
`counts`	countgene x cell

resolve_cl *Title*

Description

Title

Usage

```
resolve_cl(
  cl.g,
  cl.dat,
  markers,
  map.dat,
  select.cells,
  p = 0.7,
  low.th = 0.2,
  seed = 42
)
```

Arguments

`cl.g`

`markers`

`map.dat`

`select.cells`

`p`

`low.th`

`seed`
- random seed for reproducibility

`cl.med`

`dat`

Value

mapped.cl output

`run_mapping_on_taxonomy`

INFO – PLEASE ADD –

Description

INFO – PLEASE ADD –

Usage

```
run_mapping_on_taxonomy(
  query.dat,
  Taxonomy = "AIT12.0_mouse",
  prefix = "",
  TaxFN = NA,
  prebuild = FALSE,
  newbuild = FALSE,
  mapping.method = c("flat", "hierarchy"),
  iter = 100,
  mc.cores = 7,
  blocksize = 50000,
  dist.method = "cor",
  topk = 1,
  subsample_pct = 0.9,
  top.n.genes = 15,
  rm.clusters = NA,
  flag.serial = TRUE,
  flag.parallel.tmp = FALSE,
  flag.fuzzy = FALSE
)
```

Arguments

`cl.df`

Value

???

seuratMap *Seurat based mapping*

Description

Seurat based mapping

Usage

```
seuratMap(AIT.anndata, query.data, dims = 30, k.weight = 5)
```

Arguments

AIT.anndata	A reference taxonomy anndata object.
query.data	A logCPM normalized matrix to be annotated.

Value

Seurat mapping results as a data.frame.

taxonomy_mapping *Cell type annotation and initial QC*

Description

Perform initial mapping using three methods: Correlation-based, tree-based, and Seurat based, and will calculate some QC metrics.

Usage

```
taxonomy_mapping(
  AIT.anndata,
  query.data,
  corr.map = TRUE,
  tree.map = TRUE,
  seurat.map = TRUE,
  label.cols = c("cluster_label", "subclass_label", "class_label")
)
```

Arguments

`AIT.anndata`	A reference taxonomy object.
`query.data`	A logCPM normalized matrix to be annotated.
`corr.map`	Should correlation mapping be performed?
`tree.map`	Should tree mapping be performed?
`seurat.map`	Should seurat mapping be performed?
`label.cols`	Column names of annotations to map against. Note that this only works for metadata that represent clusters or groups of clusters (e.g., subclass, supertype, neighborhood, class)

Value

Mapping results from all methods.

`top_binary_genes` *Get top genes by beta (binary) score*

Description

Get top genes by beta (binary) score

Usage

```
top_binary_genes(data, cluster.names, gene.count = 2000)
```

Arguments

`data`	A count (or CPM or logCPM) matrix
`cluster.names`	A vector of cluster names in the reference taxonomy.
`gene.count`	The number of top genes to return (Default=2000)

Value

Boolean vector of cells to keep (TRUE) and cells to remove (FALSE)

treeMap *Tree based mapping*

Description

Tree based mapping

Usage

```
treeMap(AIT.anndata, query.data)
```

Arguments

`AIT.anndata`	A reference taxonomy anndata object.
`query.data`	A logCPM normalized matrix to be annotated.

Value

Tree mapping results as a data.frame.

writePatchseqQCmarkers *Save marker genes for patchSeqQC*

Description

This function write a file called `QC_markers.RData` that contains all the variables required for applying the patchseq QC algorithm `pathseqtools` (which is an more flexible version of the `patchSeqQC` algorithm). This is only used for patch-seq analysis.

Usage

```
writePatchseqQCmarkers(
  counts,
  metadata,
  subsample = 100,
  subclass.column = "subclass_label",
  class.column = "class_label",
  off.target.types = c("Glia", "glia", "non-neuronal", "Non-neuronal"),
  num.markers = 50,
  shinyFolder = paste0(getwd(), "/")
)
```

Arguments

`counts` A matrix of counts for the reference data, or character string with a file location of "counts.feather". If it appears cpm or logCPM matrix is provided, an warning will be thrown.

`metadata` Data frame of metadata with rows corresponding to cells/nuclei, and either row names or a column called "sample_id" corresponding to cell names. This matrix must include entries for all cells in norm.data. Could also be a file. Columns can be numeric, categorical, or factors and must include "subclass.column" and "class.column"

`subsample` The number of cells to retain per cluster (default = 100).

`subclass.column`
Column name corresponding to the moderate-resolution cell types used for the cell types of interest (default = "subclass_label").

`class.column` Column name corresponding to the low-resolution cell types used for the off-target cell types (default = "class_label").

`off.target.types`
A character vector of off-target (also known as ’contamination’) cell types. This must include at least one of the cell types found in "class.column" and/or "subclass.column" (both columns are checked)

`num.markers` The maximum number of markers to calculate per node per direction (default = 50)

`shinyFolder` = The location to save shiny output (default = current working directory).

Nothing is returned; however an R data object called "QC_markers.RData" is returned with the following variables markers, countsQC, cpmQC, classBr, subclassF, allMarkers,

Index