Package ‘celaref’

May 3, 2024

Title Single-cell RNAseq cell cluster labelling by reference

Version 1.22.0

Description After the clustering step of a single-cell RNAseq experiment, this package aims to suggest labels/cell types for the clusters, on the basis of similarity to a reference dataset. It requires a table of read counts per cell per gene, and a list of the cells belonging to each of the clusters, (for both test and reference data).

Depends R (>= 3.5.0), SummarizedExperiment

Imports MAST, ggplot2, Matrix, dplyr, magrittr, stats, utils, rlang, BiocGenerics, S4Vectors, readr, tibble, DelayedArray

Suggests limma, parallel, knitr, rmarkdown, ExperimentHub, testthat

biocViews SingleCell

VignetteBuilder knitr

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

git_url https://git.bioconductor.org/packages/celaref
git_branch RELEASE_3_19
git_last_commit 0f17788
git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-03

Author Sarah Williams [aut, cre]

Maintainer Sarah Williams <sarah.williams1@monash.edu>
contrast_each_group_to_the_rest

Contents

contrast_each_group_to_the_rest ........................................... 2
contrast_each_group_to_the_rest_for_norm_ma_with_limma .................. 4
contrast_the_group_to_the_rest ............................................. 6
contrast_the_group_to_the_rest_for_norm_ma_with_limma_for_microarray .... 7
convert_se_gene_ids .......................................................... 8
demo_cell_info_table ....................................................... 9
demo_counts_matrix .......................................................... 9
demo_gene_info_table ....................................................... 10
demo_gene_info_table ....................................................... 10
demo_microarray_expr ........................................................ 10
demo_microarray_sample_sheet .............................................. 11
demo_query_se ............................................................... 11
demo_ref_se ................................................................. 12
de_table.demo_query ........................................................ 12
de_table.demo_ref ........................................................... 13
find_within_match_differences ............................................... 13
get_counts_index ............................................................. 14
get_inner_or_outer_ci ........................................................ 14
get_limma_top_table_with_ci ............................................... 15
get_matched_stepped_mwtest_res_table .................................. 16
get_ranking_and_test_results .............................................. 16
get_rankstat_table ........................................................... 17
get_reciprocal_matches ...................................................... 18
get_stepped_pvals_str ........................................................ 19
get_the_up_genes_for_all_possible_groups .................................. 20
get_the_up_genes_for_group ................................................ 21
get_vs_random_pval ........................................................... 22
load_dataset_10Xdata ........................................................ 23
load_se_from_tables .......................................................... 24
make_ranking_violin_plot .................................................... 26
make_ref_similarity_names ................................................ 28
make_ref_similarity_names_for_group .................................... 31
run_pair_test_stats ........................................................... 32
subset_cells_by_group ...................................................... 33
subset_se_cells_for_group_test ........................................... 34
trim_small_groups_and_low_expression_genes ............................... 35

Index 37

contrast_each_group_to_the_rest

Description

Produces a table of within-experiment differential expression results (for either query or reference experiment), where each group (cluster) is compared to the rest of the cells.
contrast_each_group_to_the_rest

Usage

contrast_each_group_to_the_rest(dataset_se, dataset_name, 
    groups2test = NA, num_cores = 1, n.group = Inf, n.other = n.group * 5)

Arguments

dataset_se | Summarised experiment object containing count data. Also requires 'ID' and 'group' to be set within the cell information (see colData())
dataset_name | Short, meaningful name for this dataset/experiment.
groups2test | An optional character vector specifying specific groups to check. By default (set to NA), all groups will be tested.
num_cores | Number of cores to use to run MAST jobs in parallel. Ignored if parallel package not available. Set to 1 to avoid parallelisation. Default = 1
n.group | How many cells to keep for each group in groupwise comparisons. Default = Inf
n.other | How many cells to keep from everything not in the group. Default = n.group * 5

Details

Note that this function is slow, because it runs the differential expression. It only needs to be run once per dataset though (unless group labels change). Having package parallel installed is highly recommended.

If this function runs out of memory, consider specifying n.group and n.other to run on a subset of cells (taken from each group, and proportionally from the rest for each test). Alternatively use subset_cells_by_group to subset dataset_se for each group independently.

Both reference and query datasets should be processed with this function.

The tables produced by this function (usually named something like de_table.datasetname) contain summarised results of MAST results. Each group is compared versus cells in the group, versus not in the group, (i.e. always a 2-group contrast, other groups information is ignored). As per MAST recommendations, the proportion of genes seen in each cell is included in the model.

Value

A tibble the within-experiment de_table (differential expression table). This is a core summary of the individual experiment/dataset, which is used for the cross-dataset comparisons.

The table fields won’t necessarily match across datasets, as they include cell annotations information. Important columns (used in downstream analysis) are:

- **ID** | Gene identifier
- **ci_inner** | Inner (conservative) 95% confidence interval of log2 fold-change.
- **fdr** | Multiple hypothesis corrected p-value (using BH/FDR method)
- **group** | Cells from this group were compared to everything else
- **sig_up** | Significantly differentially expressed (fdr < 0.01), with a positive fold change?
rank  Rank position (within group), ranked by CI inner, highest to lowest.
rescaled_rank  Rank scaled 0(top most overrepresented genes in group) - 1(top most not-present genes)
dataset  Name of dataset/experiment

Examples

de_table.demo_query  <- contrast_each_group_to_the_rest(
  demo_query_se, "a_demo_query")
de_table.demo_ref  <- contrast_each_group_to_the_rest(
  demo_ref_se, "a_demo_ref", num_cores=2)

Description

This function loads and processes microarray data (from purified cell populations) that can be used as a reference.

Usage

contrast_each_group_to_the_rest_for_norm_ma_with_limma(norm_expression_table, sample_sheet_table, dataset_name, sample_name, group_name = "group", groups2test = NA, extra_factor_name = NA, pval_threshold = 0.01)

Arguments

norm_expression_table  A logged, normalised expression table. Any filtering (removal of low-expression probes/genes)
sample_sheet_table  Tab-separated text file of sample information. Columns must have names. Sample/microarray ids should be listed under sample_name column. The cell-type (or 'group') of each sample should be listed under a column group_name.
dataset_name  Short, meaningful name for this dataset/experiment.
sample_name  Name of sample_sheet_table with sample ID
group_name  Name of sample_sheet_table with group/cell-type. Default = "group"
groups2test  An optional character vector specifying specific groups to check. By default (set to NA), all groups will be tested.
extra_factor_name

Optionally, an extra cross-group factor (as column name in `sample_sheet_table`) to include in the model used by limma. E.g. An individual/mouse id. Refer limma docs. Default = NA

pval_threshold

For reporting only, a p-value threshold. Default = 0.01

Details

Sometimes there are microarray studies measuring purified cell populations that would be measured together in a single-cell sequencing experiment. E.g. comparing PBMC scRNA to FACs-sorted blood cell populations. This function will process microarray data with limma and format it for comparisons.

The microarray data used should consist of purified cell types from /emphone single study/experiment (due to batch effects). Ideally just those cell-types expected in the scRNAseq, but the method appears relatively robust to a few extra cell types.

Note that unlike the single-cell workflow there are no summarisedExperiment objects (they’re not really comparable) - this function reads data and generates a table of within-dataset differential expression contrasts in one step. I.e. equivalent to the output of `contrast_each_group_to_the_rest`.

Also, note that while downstream functions can accept the microarray-derived data as query datasets, it’s not really intended and assumptions might not hold (Generally, its known what got loaded onto a microarray!)

The (otherwise optional) ’limma’ package must be installed to use this function.

Value

A tibble, the within-experiment de_table (differential expression table)

See Also

`contrast_each_group_to_the_rest` is the funciton that makes comparable output on the scRNAseq data (dataset_se objects).

Limma Limma package for differential expression.

Other Data loading functions: `load_dataset_10Xdata`, `load_se_from_tables`

Examples

```r
contrast_each_group_to_the_rest_for_norm_ma_with_limma(
  norm_expression_table=demo_microarray_expr,
  sample_sheet_table=demo_microarray_sample_sheet,
  dataset_name="DemoSimMicroarrayRef",
  sample_name="cell_sample", group_name="group")
```

## Not run:

```r
contrast_each_group_to_the_rest_for_norm_ma_with_limma(
  norm_expression_table, sample_sheet_table=samples_table,
  dataset_name="Watkins2009PBMCs", extra_factor_name='description')
```

## End(Not run)
**Description**

Internal function to calculate differential expression within an experiment between a specified group and cells not in that group.

**Usage**

```r
contrast_the_group_to_the_rest(dataset_se, the_group, pvalue_threshold = 0.01, n.group = Inf, n.other = n.group * 5)
```

**Arguments**

- `dataset_se` Dataset summarisedExperiment object.
- `the_group` Group to test
- `pvalue_threshold` Default = 0.01
- `n.group` How many cells to keep for each group in groupwise comparisons. Default = Inf
- `n.other` How many cells to keep from everything not in the group. Default = `n.group` * 5

**Details**

This function should only be called by `contrast_each_group_to_the_rest` (which can be passed a single group name if desired). Else 'pofgenes' will not be defined.

MAST is supplied with log2(counts + 1.1), and zlm called with model '~ TvsR + pofgenes'. The p-values reported are from the hurdle model. FDR is with default fdr/BH method.

**Value**

A tibble, the within-experiment de_table (differential expression table), for the group specified.

**See Also**

`contrast_each_group_to_the_rest`
**Description**

Private function used by contrast_each_group_to_the_rest_for_norm_ma_with_limma

**Usage**

```r
contrast_the_group_to_the_rest_with_limma_for_microarray(norm_expression_table, sample_sheet_table, the_group, sample_name, extra_factor_name = NA, pval_threshold = 0.01)
```

**Arguments**

- `norm_expression_table`: A logged, normalised expression table. Any filtering (removal of low-expression probes/genes)
- `sample_sheet_table`: Tab-separated text file of sample information. Columns must have names. Sample/microarray ids should be listed under `sample_name` column. The cell-type (or 'group') of each sample should be listed under a column `group_name`.
- `the_group`: Which query group is being tested.
- `sample_name`: Name of `sample_sheet_table` with sample ID
- `extra_factor_name`: Optionally, an extra cross-group factor (as column name in `sample_sheet_table`) to include in the model used by limma. E.g. An individual/mouse id. Refer limma docs. Default = NA
- `pval_threshold`: For reporting only, a p-value threshold. Default = 0.01

**Value**

A tibble, the within-experiment de_table (differential expression table), for the group specified.

**See Also**

- `contrast_each_group_to_the_rest_for_norm_ma_with_limma` public calling function
- `Limma` Limma package for differential expression.
convert_se_gene_ids

Description

Change the gene IDs in in the supplied dataset_se object to some other id already present in the
gene info (as seen with rowData())

Usage

convert_se_gene_ids(dataset_se, new_id, eval_col, find_max = TRUE)

Arguments

dataset_se  Summarised experiment object containing count data. Also requires 'ID' and
'group' to be set within the cell information (see colData())

new_id  A column within the feature information (view colData(dataset_se)) of the
dataset_se, which will become the new ID column. Non-uniqueness of this
column is handled gracefully! Any NAs will be dropped.

eval_col  Which column to use to break ties of duplicate new_id. Must be a column within
the feature information (view colData(dataset_se)) of the dataset_se. Total
reads per gene feature is a good choice.

find_max  If false, this will choose the minimal eval_col instead of max. Default = TRUE

Value

A modified dataset_se - ID will now be new_id, and unique. It will have fewer genes if old ID to
new ID was not a 1:1 mapping. The selected genes will be according to the eval col max (or min).
should pick the alphabetical first on ties, but could change.

See Also

SummarizedExperiment For general doco on the SummarizedExperiment objects.
load_se_from_files For reading data from flat files (not 10X cellRanger output)

Examples

# The demo dataset doesn't have other names, so make some up
# (don't do this)
dataset_se <- demo_ref_se
rowData(dataset_se)$dummyname <- toupper(rowData(dataset_se)$ID)

# If not already present, define a column to evaluate,
# typically total reads/gene.
rowData(dataset_se)$total_count <- rowSums(assay(dataset_se))

dataset_se <- convert_se_gene_ids(dataset_se, new_id='dummyname', eval_col='total_count')
**demo_cell_info_table**  

---

**demo_cell_info_table**  

*Demo cell info table*

---

**Description**

Sample sheet table listing each cell, its assigned cluster/group, and any other information that might be interesting (replicate, individual e.t.c)

**Usage**

demo_cell_info_table

**Format**

An object of class `data.frame` with 515 rows and 4 columns.

**Value**

An example cell info table

---

**demo_counts_matrix**  

---

**demo_counts_matrix**  

*Demo count matrix*

---

**Description**

Counts matrix for a small, demo example datasets. Raw counts of reads per gene (row) per cell (column).

**Usage**

demo_counts_matrix

**Format**

An object of class `matrix` with 200 rows and 514 columns.

**Value**

An example counts matrix.
### demo_gene_info_table  
**Demo gene info table**

**Description**

Extra table of gene-level information for the demo example dataset. Can contain anything as long as there is a unique gene id.

**Usage**

demo_gene_info_table

**Format**

An object of class `data.frame` with 200 rows and 2 columns.

**Value**

An example table of genes.

---

### demo_microarray_expr  
**Demo microarray expression table**

**Description**

Microarray-style expression table for the demo example dataset. Rows are genes, columns are samples, as per counts matrix.

**Usage**

demo_microarray_expr

**Format**

An object of class `matrix` with 200 rows and 20 columns.

**Value**

An example table of (fake) microarray data.
**demo_microarray_sample_sheet**

*Demo microarray sample sheet table*

**Description**

Microarray sample sheet table for the demo example dataset. Contains array identifiers, their group and any other information that could be useful.

**Usage**

demo_microarray_sample_sheet

**Format**

An object of class `grouped_df` (inherits from `tbl_df, tbl, data.frame`) with 20 rows and 2 columns.

**Value**

An example microarray sample sheet

---

**demo_query_se**

*Demo query se (summarizedExperiment)*

**Description**

A summarisedExperiment object loaded from demo info tables, for a query set.

**Usage**

demo_query_se

**Format**

An object of class `SummarizedExperiment` with 200 rows and 485 columns.

**Value**

An example summarised experiment (for demo query dataset)
dea_table.demo_query

demo_ref_se  Demo reference se (summarizedExperiment)

Description
A summarisedExperiment object loaded from demo info tables, for a reference set.

Usage
demo_ref_se

Format
An object of class SummarizedExperiment with 200 rows and 515 columns.

Value
An example summarised experiment (for demo reference dataset)

dea_table.demo_query  Demo query de table

Description
Small example dataset that is the output of contrast_each_group_to_the_rest. It contains the results of each group compared to the rest of the sample (ie within sample differential expression)

Usage
de_table.demo_query

Format
An object of class data.frame with 800 rows and 13 columns.

Value
An example de_table from contrast_each_group_to_the_rest (for demo query dataset)
Demo ref de table

**Description**

Small example dataset that is the output of `contrast_each_group_to_the_rest`. It contains the results of each group compared to the rest of the sample (i.e., within-sample differential expression).

**Usage**

`de_table.demo_ref`

**Format**

An object of class `data.frame` with 800 rows and 13 columns.

**Value**

An example `de_table` from `contrast_each_group_to_the_rest` (for demo ref dataset)

---

find_within_match_differences

**Description**

Internal function to find if there are significant differences between the distributions, when there are multiple match groups.

**Usage**

```
find_within_match_differences(de_table.ref.marked, matches, the_test_group, the_test_dataset, the_ref_dataset, the_pval)
```

**Arguments**

- `de_table.ref.marked`
  - See `make_ref_similarity_names_for_group`
- `matches`
  - See `make_ref_similarity_names_for_group`
- `the_test_group`
  - See `make_ref_similarity_names_for_group`
- `the_test_dataset`
  - See `make_ref_similarity_names_for_group`
- `the_ref_dataset`
  - See `make_ref_similarity_names_for_group`
- `the_pval`
  - See `make_ref_similarity_names_for_group`
get_inner_or_outer_ci

Details
For use by make_ref_similarity_names_for_group

Value
String of within match differences

See Also
make_ref_similarity_names_for_group

get_counts_index

description
get_counts_index is an internal utility function to find out where the counts are (if anywhere.). Stops if there’s no assay called ’counts’, (unless there is only a single unnamed assay).

Usage
get_counts_index(n_assays, assay_names)

Arguments
n_assays How many assays are there? ie: length(assays(dataset_se))
assay_names What are the assays called? ie: names(assays(dataset_se))

Value
The index of an assay in assays called ’counts’, or, if there’s just the one unnamed assay - happily assume that that is counts.

get_inner_or_outer_ci

description
Given a fold-change, and high and low confidence interval (where lower < higher), pick the inner-most/most conservative one.

Usage
get_inner_or_outer_ci(fc, ci.hi, ci.lo, get_inner = TRUE)
get_limma_top_table_with_ci

Arguments

- fc: Fold-change
- ci.hi: Higher fold-change CI (numerically)
- ci.lo: smaller fold-change CI (numerically)
- get_inner: If TRUE, get the more conservative inner CI, else the bigger outside one.

Value

inner or outer CI from ci.hi or ci.lo

Description

Internal function that wraps limma topTable output but also adds upper and lower confidence intervals to the logFC. Calculated according to https://support.bioconductor.org/p/36108/

Usage

get_limma_top_table_with_ci(fit2, the_coef, ci = 0.95)

Arguments

- fit2: The fit2 object after calling eBayes as per standard limma workflow. Ie object that topTable gets called on.
- the_coef: Coefficient. As passed to topTable.
- ci: Confidence interval. Number between 0 and 1, default 0.95 (95%)

Value

Output of topTable, but with the (95 for the logFC.

See Also

contrast_the_group_to_the_rest_with_limma_for_microarray Calling function.
get_matched_stepped_mwtest_res_table

**Description**

Internal function to grab a table of the matched group(s).

**Usage**

```r
get_matched_stepped_mwtest_res_table(mwtest_res_table.this, the_pval)
```

**Arguments**

- `mwtest_res_table.this`: Combined output of `get_ranking_and_test_results`
- `the_pval`: Pvalue threshold

**Details**

For use by `make_ref_similarity_names_for_group`

**Value**

Stepped pvalues string

**See Also**

`make_ref_similarity_names_for_group`

---

get_ranking_and_test_results

**Description**

Internal function to get reference group similarity contrasts for an individual query group.

**Usage**

```r
get_ranking_and_test_results(de_table.ref.marked, the_test_group, the_test_dataset, the_ref_dataset, num_steps, pval = 0.01)
```
get_rankstat_table

Arguments

del_table.ref.marked
  see make_ref_similarity_names_using_marked
the_test_group
  The group to calculate the stats on.
the_test_dataset
  see make_ref_similarity_names_using_marked
the_ref_dataset
  see make_ref_similarity_names_using_marked
num_steps
  see make_ref_similarity_names_using_marked
pval
  see make_ref_similarity_names_using_marked

Details

For use by make_ref_similarity_names_using_marked, see that function for parameter details. This function just runs this for a single query group the_test_group

Value

Table of similarity contrast results/assigned names e.t.c for a single group. Used internally for populating mwtest_res_table tables.

See Also

make_ref_similarity_names_using_marked which calls this.

del_table.ref.marked  get_rankstat_table

Description

Summarise the comparison of the specified query group against in the comparison in del_table.ref.marked - number of 'top' genes and their median rank in each of the reference groups, with reference group rankings.

Usage

gget_rankstat_table(del_table.ref.marked, the_test_group)

Arguments

del_table.ref.marked
  The output of get_the_up_genes_for_all_possible_groups for the contrast of interest.
the_test_group
  Name of query group to test
Value

A tibble of query group name (test_group), number of 'top' genes (n), reference dataset group (group) with its ranking (grouprank) and the median (rescaled 0..1) ranking of 'top' genes (median_rank).

See Also

get_the_up_genes_for_all_possible_groups To prepare the de_table.ref.marked input.

Examples

# Make input
# de_table.demo_query <- contrast_each_group_to_the_rest(demo_query_se, "demo_query")
# de_table.demo_ref  <- contrast_each_group_to_the_rest(demo_ref_se, "demo_ref")

de_table.marked.query_vs_ref <- get_the_up_genes_for_all_possible_groups(
  de_table.demo_query,
  de_table.demo_ref)

get_rankstat_table(de_table.marked.query_vs_ref, "Group3")

Description

Internal function to run a binomial test of median test rank > 0.5 (random).

Usage

get_reciprocal_matches(mwtest_res_table.recip, de_table.recip.marked, the_pval)

Arguments

mwtest_res_table.recip
  Combined output of get_ranking_and_test_results for reciprocal test - ref vs query.

de_table.recip.marked
  Reciprocal ref vs query de_table.ref.marked

the_pval
  See make_ref_similarity_names_using_marked

Details

For use by make_ref_similarity_names_using_marked
get_stepped_pvals_str

Value

List of table of reciprocal matches tested from reference to query.

See Also

make_ref_similarity_names_using_marked

Description

Internal function to construct the string of stepped pvalues reported by make_ref_similarity_names_using_marked

Usage

get_stepped_pvals_str(mwtest_res_table.this)

Arguments

mwtest_res_table.this
  Combined output of get_ranking_and_test_results

Details

For use by make_ref_similarity_names_for_group

Value

Stepped pvalues string

See Also

make_ref_similarity_names_for_group
get_the_up_genes_for_all_possible_groups

Description

For the most overrepresented genes of each group in the test dataset, get their rankings in all the groups of the reference dataset.

Usage

get_the_up_genes_for_all_possible_groups(de_table.test, de_table.ref, rankmetric = "TOP100_LOWER_CI_GTE1", n = 100)

Arguments

de_table.test A differential expression table of the query experiment, as generated from contrast_each_group_to_the_rest
de_table.ref A differential expression table of the reference dataset, as generated from contrast_each_group_to_the_rest
rankmetric Specify ranking method used to pick the 'top' genes. The default 'TOP100_LOWER_CI_GTE1' picks genes from the top 100 overrepresented genes (ranked by inner 95 work best for distinct cell types (e.g. tissue sample.). 'TOP100_SIG' again picks from the top 100 ranked genes, but requires only statistical significance, 95 clusters (e.g. PBMCs).
n For tweaking maximum returned genes from different ranking methods. Will change the p-values! Suggest leaving as default unless you're keen.

Details

This is effectively a subset of the reference data, 'marked' with the 'top' genes that represent the groups in the query data. The distribution of the rescaled ranks of these marked genes in each reference data group indicate how similar they are to the query group.

This function is simply a convenient wrapper for get_the_up_genes_for_group that merges output for each group in the query into one table.

Value

de_table.marked This will alsmost be a subset of de_table.ref, with an added column test_group set to the query groups, and test_dataset set to test_dataset_name.

If nothing passes the rankmetric criteria, a warning is thrown and NA is returned. (This can be a genuine inability to pick out the representative 'up' genes, or due to some problem in the analysis)

See Also

get_the_up_genes_for_group Function for testing a single group.
**get_the_up_genes_for_group**

**Examples**

```r
de_table.marked.query_vs_ref <- get_the_up_genes_for_all_possible_groups(
  de_table.test=de_table.demo_query ,
  de_table.ref=de_table.demo_ref )
```

**Description**

For the most overrepresented genes of the specified group in the test dataset, get their rankings in all the groups of the reference dataset.

**Usage**

```r
get_the_up_genes_for_group(the_group, de_table.test, de_table.ref,
  rankmetric = "TOP100_LOWER_CI_GTE1", n = 100)
```

**Arguments**

- **the_group**: The group (from the test/query experiment) to examine.
- **de_table.test**: A differential expression table of the query experiment, as generated from `contrast_each_group_to_the_rest`.
- **de_table.ref**: A differential expression table of the reference dataset, as generated from `contrast_each_group_to_the_rest`.
- **rankmetric**: Specify ranking method used to pick the 'top' genes. The default 'TOP100_LOWER_CI_GTE1' picks genes from the top 100 overrepresented genes (ranked by inner 95 work best for distinct cell types (e.g. tissue sample.). 'TOP100_SIG' again picks from the top 100 ranked genes, but requires only statistical significance, 95 clusters (e.g. PBMCs).
- **n**: For tweaking maximum returned genes from different ranking methods. Will change the p-values! Suggest leaving as default unless you’re keen.

**Details**

This is effectively a subset of the reference data, 'marked' with the 'top' genes that represent the group of interest in the query data. The distribution of the rescaled ranks of these marked genes in each reference data group indicate how similar they are to the query group.

**Value**

*de_table.marked* This will be a subset of *de_table.ref*, with an added column *test_group* set to *the_group*. If nothing passes the rankmetric criteria, NA.
See Also

contrast_each_group_to_the_rest For prepraing the de_table.* tables. get_the_up_genes_for_all_possible_groups For running all query groups at once.

Examples

de_table.marked.Group3vsRef <- get_the_up_genes_for_group(
  the_group="Group3",
  de_table.test=de_table.demo_query,
  de_table.ref=de_table.demo_ref)

Description

Internal function to run a binomial test of median test rank > 0.5 (random).

Usage

get_vs_random_pval(de_table.ref.marked, the_group, the_test_group)

Arguments

detable.ref.marked
  see make_ref_similarity_names_for_group
the_group
  Reference group name
the_test_group
  Test group name #'

Details

For use by make_ref_similarity_names_for_group

Value

Pvalue result of a binomial test of each 'top gene' being greater than the theoretical random median rank of 0.5 (halfway).

See Also

make_ref_similarity_names_for_group
Description

Convenience function to create a SummarizedExperiment object (dataset_se) from a the output of 10X cell ranger pipeline run.

Usage

load_dataset_10Xdata(dataset_path, dataset_genome, clustering_set,
  gene_id_cols_10X = c("ensembl_ID", "GeneSymbol"),
  id_to_use = gene_id_cols_10X[1])

Arguments

dataset_path Path to the directory of 10X data, as generated by the cellRanger pipeline (versions 2.1.0 and 2.0.1). The directory should have subdirectories analysis, filtered_gene_bc_matrices and raw_gene_bc_matrices (only the first 2 are read).

dataset_genome The genome that the reads were aligned against, e.g. GRCh38. Check for this as a directory name under the filtered_gene_bc_matrices subdirectory if unsure.

clustering_set The 10X cellRanger pipeline produces several different cluster definitions per dataset. Specify which one to use e.g. kmeans_10_clusters Find them as directory names under analysis/clustering/

gene_id_cols_10X Vector of the names of the columns in the gene description file (filtered_gene_bc_matrices/GRCh38/genes.csv). The first element of this will become the ID. Default = c("ensembl_ID","GeneSymbol")

id_to_use Column from gene_id_cols_10X that defines the gene identifier to use as 'ID' in the returned SummarisedExperiment object. Many-to-one relationships between the assumed unique first element of gene_id_cols_10X and id_to_use will be handled gracefully by convert_se_gene_ids. Defaults to first element of gene_id_cols_10X

Details

This function makes a SummarizedExperiment object in a form that should work for celaref functions. Specifically, that means it will have an 'ID' field for genes (view with rowData(dataset_se)), and both 'cell_sample' and 'group' field for cells (view with colData(dataset_se)). See parameters for detail. Additionally, the counts will be an integer matrix (not a sparse matrix), and the group field (but not cell_sample or ID) will be a factor.

The clustering information can be read from whichever cluster is specified, usually there will be several choices.

This function is designed to work with output of version 2.0.1 of the cellRanger pipeline, may not work with others (will not work for 1.x).
load_se_from_tables

Value

A SummarisedExperiment object containing the count data, cell info and gene info.

See Also

- **SummarizedExperiment** For general doco on the SummarizedExperiment objects.
- **convert_se_gene_ids** describes method for converting IDs.

Other Data loading functions: **contrast_each_group_to_the_rest_for_norm_ma_with_limma, load_se_from_tables**

Examples

eample_10X_dir <- system.file("extdata", "sim_cr_dataset", package = "celaref")
dataset_se <- load_dataset_10Xdata(example_10X_dir, dataset_genome="GRCh38",
   clustering_set="kmeans_4_clusters", gene_id_cols_10X=c("gene"))

## Not run:
dataset_se <- load_dataset_10Xdata("~/path/to/data/10X_pbmc4k",
   dataset_genome="GRCh38",
   clustering_set="kmeans_7_clusters")

## End(Not run)

load_se_from_tables

Description

Create a SummarizedExperiment object (dataset_se) from a count matrix, cell information and optionally gene information.

**load_se_from_files** is a wrapper for **load_se_from_tables** that will read in tables from specified files.

Usage

load_se_from_tables(counts_matrix, cell_info_table, gene_info_table = NA,
   group_col_name = "group", cell_col_name = NA)

load_se_from_files(counts_file, cell_info_file, gene_info_file = NA,
   group_col_name = "group", cell_col_name = NA)
Arguments

counts_matrix  A tab-separated matrix of read counts for each gene (row) and each cell (column). Columns and rows should be named.

cell_info_table  Table of cell information. If there is a column labelled cell_sample, that will be used as the unique cell identifiers. If not, the first column is assumed to be cell identifiers, and will be copied to a new field labelled cell_sample. Similarly - the clusters of these cells should be listed in one column - which can be called 'group' (case-sensitive) or specified with group_col_name. Minimal data format: <cell_sample> <group>

gene_info_table  Optional table of gene information. If there is a column labelled ID, that will be used as the gene identifiers (they must be unique!). If not, the first column is assumed to be a gene identifier, and will be copied to a new field labelled ID. Must match all rownames in counts_matrix. If omitted, ID will be generated from the rownames of counts_matrix. Default=NA

group_col_name  Name of the column in cell_info_table containing the cluster/group that each cell belongs to. Case-sensitive. Default='group'

cell_col_name  Name of the column in cell_info_table containing a cell id. Ignored if cell_sample column is already present. If omitted, (and no cell_sample column) will use first column. Case-sensitive. Default=NA

counts_file  A tab-separated file of a matrix of read counts. As per counts_matrix. First column should be gene ID, and top row cell ids.

cell_info_file  Tab-separated text file of cell information, as per cell_info_table. Columns must have names.

gene_info_file  Optional tab-separated text file of gene information, as per gene_info_file. Columns must have names. Default=NA

Details

This function makes a SummarizedExperiment object in a form that should work for celaref functions. Specifically, that means it will have an 'ID' field for genes (view with rowData(dataset_se)), and both 'cell_sample' and 'group' field for cells (view with colData(dataset_se)). See parameters for details. Additionally, the counts will be an integer matrix (not a sparse matrix), and the group field (but not cell_sample or ID) will be a factor.

Note that data will be subsetted to cells present in both the counts matrix and cell info, this is handy for loading subsets of cells. However, if gene_info_file is defined, all genes must match exactly.

The load_se_from_files form of this function will run the same checks, but will read everything from files in one go. The load_se_from_tables form is perhaps more useful when the annotations need to be modified (e.g. programmatically adding a different gene identifier, renaming groups, removing unwanted samples).

Note that the SummarizedExperiment object can also be created without using these functions, it just needs the cell_sample, ID and group fields as described above. Since sometimes it might be easier to add these to an existing SummarizedExperiment from upstream analyses.
make_ranking_violin_plot

Value
A SummarisedExperiment object containing the count data, cell info and gene info.

Functions
• load_se_from_files: To read from files

See Also
SummarizedExperiment For general doco on the SummarizedExperiment objects.

Other Data loading functions: contrast_each_group_to_the_rest_for_norm_ma_with_limma,
load_dataset_10Xdata

Examples
# From data frames (or a matrix for counts):
demo_se <- load_se_from_tables(counts_matrix=demo_counts_matrix,
cell_info_table=demo_cell_info_table)
demo_se <- load_se_from_tables(counts_matrix=demo_counts_matrix,
cell_info_table=demo_cell_info_table,
gene_info_table=demo_gene_info_table)

# Or from data files:
counts_filepath <- system.file("extdata", "sim_query_counts.tab", package = "celaref")
cell_info_filepath <- system.file("extdata", "sim_query_cell_info.tab", package = "celaref")
gene_info_filepath <- system.file("extdata", "sim_query_gene_info.tab", package = "celaref")
demo_se <- load_se_from_files(counts_file=counts_filepath, cell_info_file=cell_info_filepath)
demo_se <- load_se_from_files(counts_file=counts_filepath, cell_info_file=cell_info_filepath,
gene_info_file=gene_info_filepath)

make_ranking_violin_plot

make_ranking_violin_plot

Description
Plot a panel of violin plots showing the distribution of the 'top' genes of each of query group, across
the reference dataset.

Usage
make_ranking_violin_plot(de_table.marked = NA, de_table.test = NA,
de_table.ref = NA, log10trans = FALSE, ...)

make_ranking_violin_plot
Arguments

- `de_table.marked`: The output of `get_the_up_genes_for_all_possible_groups` for the contrast of interest.
- `de_table.test`: A differential expression table of the query experiment, as generated from `contrast_each_group_to_the_rest`.
- `de_table.ref`: A differential expression table of the reference dataset, as generated from `contrast_each_group_to_the_rest`.
- `log10trans`: Plot on a log scale? Useful for distinguishing multiple similar, yet distinct cell type that bunch at top of plot. Default=FALSE.
- `...`: Further options to be passed to `get_the_up_genes_for_all_possible_groups`, e.g. `rankmetric`.

Details

In the plot output, each panel corresponds to a different group/cluster in the query experiment. The x-axis has the groups in the reference dataset. The y-axis is the rescaled rank of each ‘top’ gene from the query group, within each reference group.

Only the ‘top’ genes for each query group are plotted, forming the violin plots - each individual gene is shown as a tickmark. Some groups have few top genes, and so their uncertainty can be seen on this plot.

The thick black lines represent the median gene rescaled ranking for each query group / reference group combination. Having this fall above the dotted median threshold marker is a quick indication of potential similarity. A complete lack of similarity would have a median rank around 0.5. Median rankings much less than 0.5 are common though (an ‘anti-cell-groupA’ signature), because genes overrepresented in one group in an experiment, are likely to be relatively ‘underrepresented’ in the other groups. Taken to an extreme, if there are only two reference groups, they’ll be complete opposites.

Input can be either the precomputed `de_table.marked` object for the comparison, OR both `de_table.test` and `de_table.ref` differential expression results to compare from `contrast_each_group_to_the_rest`.

Value

A ggplot object.

See Also

`get_the_up_genes_for_all_possible_groups` To make the input data.

Examples

```r
# Make input
# de_table.demo_query <- contrast_each_group_to_the_rest(demo_query_se, "demo_query")
# de_table.demo_ref <- contrast_each_group_to_the_rest(demo_ref_se, "demo_ref")

# This:
make_ranking_violin_plot(de_table.test=de_table.demo_query,
                          de_table.ref=de_table.demo_ref )
```
# Is equivalent to this:
```r
de_table.marked.query_vs_ref <-
  get_the_up_genes_for_all_possible_groups( de_table.test=de_table.demo_query,
  de_table.ref=de_table.demo_ref)
make_ranking_violin_plot(de_table.marked.query_vs_ref)
```

## make_ref_similarity_names

### Description

Construct some sensible labels or the groups/clusters in the query dataset, based on similarity the reference dataset.

This is a more low level/customisable version of `make_ref_similarity_names`, (would usually use that instead). Suitable for rare cases to reuse an existing `de_table.ref.marked` object. Or use a `de_table.ref.marked` table with more than one dataset present (discoraged). Or to skip the reciprocal comparison step.

### Usage

```r
make_ref_similarity_names(de_table.test, de_table.ref, pval = 0.01,
  num_steps = 5, rankmetric = "TOP100_LOWER_CI_GTE1", n = 100)
make_ref_similarity_names_using_marked(de_table.ref.marked,
  de_table.recip.marked = NA, the_test_dataset = NA,
  the_ref_dataset = NA, pval = 0.01, num_steps = 5)
```

### Arguments

- **de_table.test**: A differential expression table of the query experiment, as generated from `contrast_each_group_to_the_rest`
- **de_table.ref**: A differential expression table of the reference dataset, as generated from `contrast_each_group_to_the_rest`
- **pval**: Differences between the rescaled ranking distribution of 'top' genes on different reference groups are tested with a Mann-Whitney U test. If they are significantly different, only the top group(s) are reported. It isn’t a simple cutoff threshold as it can change the number of similar groups reported. ie. A more stringent `pval` is more likely to decide that groups are similar - which would result in multiple group reporting, or no similarity at all. Unlikely that this parameter will ever need to change. Default = 0.01.
- **num_steps**: After ranking reference groups according to median 'top' gene ranking, how many adjacent pairs to test for differences. Set to 1 to only compare each group to the next, or NA to perform an all-vs-all comparison. Setting too low may means it is possible to miss groups with some similarity to the reported matches (similar_non_match column). Too high (or NA) with a large number of reference groups could be slow. Default = 5.
**make_ref_similarity_names**

**rankmetric**
Specify ranking method used to pick the 'top' genes. The default 'TOP100_LOWER_CI_GTE1' picks genes from the top 100 overrepresented genes (ranked by inner 95 work best for distinct cell types (e.g. tissue sample). 'TOP100_SIG' again picks from the top 100 ranked genes, but requires only statistical significance, 95 clusters (e.g. PBMCs).

**n**
For tweaking maximum returned genes from different ranking methods.

**de_table.ref.marked**
The output of `get_the_up_genes_for_all_possible_groups` for the contrast of interest.

**de_table.recip.marked**
Optional. The (reciprocal) output of `get_the_up_genes_for_all_possible_groups` with the test and reference datasets swapped. If omitted a reciprocal test will not be done. Default = NA.

**the_test_dataset**
Optional. A short meaningful name for the experiment. (Should match `test_dataset` column in `de_table.marked`). Only needed in a table of more than one dataset. Default = NA.

**the_ref_dataset**
Optional. A short meaningful name for the experiment. (Should match `dataset` column in `de_table.marked`). Only needed in a table of more than one dataset. Default = NA.

**Details**

This function aims to report a) the top most similar reference group, if there's a clear frontrunner, b) A list of multiple similar groups if they have similar similarity, or c) 'No similarity', if there is none.

Each group is named according to the following rules. Testing for significant (smaller) differences with a one-directional Mann-Whitney U test on their rescaled ranks:

1. The first (as ranked by median rescaled rank) reference group is significantly more similar than the next: Report *first only*.
2. When comparing differences between groups stepwise ranked by median rescaled rank - no group is significantly different to its neighbour: Report *no similarity*.
3. There's no significant differences in the stepwise comparisons of the first N reference groups - but there is a significant difference later on: Report *multiple group similarity*.

There are some further heuristic caveats:

1. The distribution of top genes in the last (or only) match group is tested versus a theoretical random distribution around 0.5 (as reported in `pval_vs_random` column). If the distribution is not significantly above random (It is possible in edge cases where there is a skewed dataset and no/low matches), *no similarity* is reported. The significant `pval` column is left intact.
2. The comparison is repeated reciprocally - reference groups vs the query groups. This helps sensitivity of heterogenous query groups - and investigating the reciprocal matches can be informative in these cases. If a query group doesn't 'match' a reference group, but the reference group does match that query group - it is reported in the group label in brackets. e.g. c1:th_lymphocytes(tc_lymocytes). Its even possible if there was no match (and `pval` = NA) e.g. emphc2:(tc_lymphocytes)
The similarity is formatted into a group label. Where there are multiple similar groups, they're listed from most to least similar by their median ranks.

For instance, a query dataset of clusters c1, c2, c3 and c4 against a cell-type labelled reference dataset might get names like: E.g.

- c1:macrophage
- c2:endothelial|mesodermal
- c3:no_similarity
- c4:mesodermal(endothelial)

Function `make_ref_similarity_names` is a convenience wrapper function for `make_ref_similarity_names_from_marked`. It accepts two 'de_table' outputs of function `contrast_each_group_to_the_rest` to compare and handles running `get_the_up_genes_for_all_possible_groups`. Sister function `make_ref_similarity_names_from_marked` may (rarely) be of use if the `de_table.marked` object has already been created, or if reciprocal tests are not wanted.

Value

A table of automagically-generated labels for each query group, given their similarity to reference groups.

The columns in this table:

- **test_group**: Query group e.g. "c1"
- **shortlab**: The cluster label described above e.g. "c1:macrophage"
- **pval**: If there is a similarity flagged, this is the P-value from a Mann-Whitney U test from the last 'matched' group to the adjacent 'non-matched' group. I.e. If only one label in shortlab, this will be the first of the stepped_pvals, if there are 2, it will be the second. If there is 'no_similarity' this will be NA (Because there is no confidence in what is the most appropriate of the all non-significant stepped pvalues.).
- **stepped_pvals**: P-values from Mann-Whitney U tests across adjacent pairs of reference groups ordered from most to least similar (ascending median rank). I.e. 1st-2nd most similar first, 2nd-3rd, 3rd-4th e.t.c. The last value will always be NA (no more reference group).
  
  e.g. refA:8.44e-10,refB:2.37e-06,refC:0.000818,refD:0.435,refE:0.245,refF:NA
- **pval_to_random**: P-value of test of median rank (of last matched reference group) < random, from binomial test on top gene ranks (being < 0.5).
- **matches**: List of all reference groups that 'match', as described, except it also includes (rare) examples where pval_to_random is not significant. "|") delimited.
- **reciprocal_matches**: List of all reference groups that flagged test group as a match when direction of comparison is reversed. (significant pval and pval_to_random). "|") delimited. Usually NA.
- **similar_non_match**: This column lists any reference groups outside of shortlab that are not significantly different to a reported match group. Limited by num_steps, and will never find anything if num_steps==1. "|") delimited. Usually NA.
- **similar_non_match_detail**: P-values for any details about similar_non_match results. These p-values will always be non-significant. E.g. "A > C (p=0.0214,n.s)"). "|") delimited. Usually NA.
- **differences_within**: This field lists any pairs of reference groups in shortlab that are significantly different. "|") delimited. Usually NA.
Functions

- `make_ref_similarity_names_for_group`: Internal function, called by `make_ref_similarity_names_using_marked` for each group.

Usage

```r
make_ref_similarity_names_for_group(the_test_group, mwtest_res_table, de_table.ref.marked, reciprocal_matches = NA, the_test_dataset, the_ref_dataset, the_pval)
```
run_pair_test_stats

Arguments

the_test_group  Query group to make name for
mwtest_res_table  Mann-whitney test results as constructed in make_ref_similarity_names_using_marked
de_table.ref.marked  The output of get_the_up_genes_for_all_possible_groups for the contrast of interest.
reciprocal_matches  Simplified table of reciprocal matches prepared within make_ref_similarity_names_using_marked. If omitted no reciprocal matching done. Default = NA.
the_test_dataset  A short meaningful name for the experiment. (Should match test_dataset column in de_table.marked)
the_ref_dataset  A short meaningful name for the experiment. (Should match dataset column in de_table.marked)
the_pval  pval as per make_ref_similarity_names_using_marked

Value

A tibble with just one group’s labelling information, as per make_ref_similarity_names_using_marked

See Also

make_ref_similarity_names_using_marked Only place that uses this function, details there.

Description

Internal function to compare the distribution of a query datasets 'top' genes between two different reference datasete groups with a Mann–Whitney U test. One directional test if groupA median < group B.

Usage

run_pair_test_stats(de_table.ref.marked, the_test_group, groupA, groupB, enforceAgtB = TRUE)
subset_cells_by_group

Arguments

de_table.ref.marked
   The output of get_the_up_genes_for_all_possible_groups for the contrast of interest.
the_test_group
   Name of the test group in query dataset.
groupA
   One of the reference group names
groupB
   Another of the reference group names
enforceAgtB
   Do a one tailed test of A `less` B (more similar)? Or two-tailed. Default = TRUE.

Details

For use by make_ref_similarity_names_using_marked

Value

A tibble of wilcox / man-whitneyU test results for this contrast.

See Also

make_ref_similarity_names_using_marked

Description

Utility function to randomly subset very large datasets (that use too much memory). Specify a maximum number of cells to keep per group and use the subsetted version to analysis.

Usage

subset_cells_by_group(dataset_se, n.group = 1000)

Arguments

dataset_se
   Summarised experiment object containing count data. Also requires `ID' and `group' to be set within the cell information.
n.group
   How many cells to keep for each group. Default = 1000
subset_se_cells_for_group_test

Details

The resulting differential expression table *de_table* will have reduced statistical power. But as long as enough cells are left to reasonably accurately calculate differential expression between groups this should be enough for celaref to work with.

Also, this function will lose proportionality of groups (there’ll be *n.groups* or less of each). Consider using the *n.group/n.other* parameters in *contrast_each_group_to_the_rest* or *contrast_the_group_to_the_rest* - which subsets non-group cells independantly for each group. That may be more appropirate for tissue type samples which would have similar compositions of cells.

So this function is intended for use when either; the proportionality isn’t relevant (e.g. FACs purified cell populations), or, the data is just too big to work with otherwise.

Cells are randomly sampled, so set the random seed (with *set.seed()*) for consistant results across runs.

Value

*dataset.se* A hopefully more managably subsetted version of the inputted *dataset.se*.

See Also

*contrast_each_group_to_the_rest* For alternative method of subsetting cells proportionally.

Examples

```r
dataset_se.30pergroup <- subset_cells_by_group(demo_query_se, n.group=30)
```

Description

This function for use by *contrast_each_group_to_the_rest* downsamples cells from a summarizedExperiment (*dataset.se*) - keeping *n.group* (or all if fewer) cells from the specified group, and *n.other* from the rest. This maintains the proportions of cells in the ’other’ part of the differential expression comparisons.

Usage

```r
subset_se_cells_for_group_test(dataset_se, the_group, n.group = Inf, n.other = n.group * 5)
```
trim_small_groups_and_low_expression_genes

Arguments

dataset_se Summarised experiment object containing count data. Also requires ‘ID’ and ‘group’ to be set within the cell information.

dataset_se The group being subsetted for

dataset_se How many cells to keep for each group. Default = Inf

dataset_se How many cells to keep from everything not in the group. Default = n.group * 5

Details

Cells are randomly sampled, so set the random seed (with set.seed()) for consistant results across runs.

Value

dataset_se A hopefully more managably subsetted version of the inputted dataset_se

See Also

Calling function contrast_each_group_to_the_rest

subset_cells_by_group Exported function for subsetting each group independantly upfront. (For when this approach is still unmanageable)

trim_small_groups_and_low_expression_genes

dataset_se A hopefully more managably subsetted version of the inputted dataset_se

Description

Filter and return a SummarizedExperiment object (dataset_se) by several metrics:

• Cells with at least min_lib_size total reads.
• Genes expressed in at least min_detected_by_min_samples cells, at a threshold of min_reads_in_sample per cell.
• Remove entire groups (clusters) of cells where there are fewer than min_group_membership cells in that group.

Usage

trim_small_groups_and_low_expression_genes(dataset_se, min_lib_size = 1000, min_group_membership = 5, min_reads_in_sample = 1, min_detected_by_min_samples = 5)
Arguments

- **dataset_se**: Summarised experiment object containing count data. Also requires 'ID' and 'group' to be set within the cell information (see `colData()`)
- **min_lib_size**: Minimum library size. Cells with fewer than this many reads removed. Default = 1000
- **min_group_membership**: Throw out groups/clusters with fewer than this many cells. May change with experiment size. Default = 5
- **min_reads_in_sample**: Require this many reads to consider a gene detected in a sample. Default = 1
- **min_detected_by_min_samples**: Keep genes detected in this many samples. May change with experiment size. Default = 5

Details

If it hasn't been done already, it is highly recommended to use this function to filter out genes with no/low total counts (especially in single cell data, there can be many) - without expression they are not useful and may reduce statistical power.

Likewise, very small groups (<5 cells) are unlikely to give useful results with this method. And cells with abnormally small library sizes may not be desirable.

Of course 'reasonable' thresholds for filtering cells/genes are subjective. Defaults are moderately sensible starting points.

Value

A filtered dataset_se, ready for use.

Examples

```r
demo_query_se.trimmed <- trim_small_groups_and_low_expression_genes(demo_query_se)
demo_query_se.trimmed2 <-
  trim_small_groups_and_low_expression_genes(demo_ref_se,
  min_group_membership = 10)
```
Index

* Data loading functions
  contrast_each_group_to_the_rest_for_norm_ma_with_limma, 4
  load_dataset_10Xdata, 23
  load_se_from_tables, 24

* Data-loading functions
  load_se_from_tables, 24

* datasets
  de_table.demo_query, 12
  de_table.demo_ref, 13
  demo_cell_info_table, 9
  demo_counts_matrix, 9
  demo_gene_info_table, 10
  demo_microarray_expr, 10
  demo_microarray_sample_sheet, 11
  demo_query_se, 11
  demo_ref_se, 12

  contrast_each_group_to_the_rest, 2, 5, 6,
    12, 13, 20–22, 27, 28, 31, 34, 35
  contrast_each_group_to_the_rest_for_norm_ma_with_limma,
    4, 7, 24, 26
  contrast_the_group_to_the_rest, 6
  contrast_the_group_to_the_rest_with_limma_for_microarray,
    7, 15
  convert_se_gene_ids, 8, 23, 24

  de_table.demo_query, 12
  de_table.demo_ref, 13
  demo_cell_info_table, 9
  demo_counts_matrix, 9
  demo_gene_info_table, 10
  demo_microarray_expr, 10
  demo_microarray_sample_sheet, 11
  demo_query_se, 11
  demo_ref_se, 12

  find_within_match_differences, 13

  get_counts_index, 14

  get_inner_or_outer_ci, 14
  get_top_table_with_ci, 15
  get_matched_stepped_mwtest_res_table, 16
  get_ranking_and_test_results, 16, 16, 18,
    19
  get_rankstat_table, 17
  get_reciprocal_matches, 18
  get_stepped_pvals_str, 19
  get_the_up_genes_for_all_possible_groups,
    17, 18, 20, 22, 27, 29–33
  get_the_up_genes_for_group, 20, 21
  get_vs_random_pval, 22

  load_dataset_10Xdata, 5, 23, 26
  load_se_from_files, 8
  load_se_from_files
    (load_se_from_tables), 24
  load_se_from_tables, 5, 24, 24

  make_ranking_violin_plot, 26
  make_ref_similarity_names, 28, 28
  make_ref_similarity_names_for_group,
    14, 16, 19, 22, 31
  make_ref_similarity_names_using_marked,
    17, 19, 32, 33
  make_ref_similarity_names_using_marked
    (make_ref_similarity_names), 28

  run_pair_test_stats, 32

  subset_cells_by_group, 33, 35
  subset_se_cells_for_group_test, 34

  trim_small_groups_and_low_expression_genes,
    35