Package ‘scITD’

March 23, 2022

Title Single-Cell Interpretable Tensor Decomposition

Version 1.0.2

Date 2022-03-23

Maintainer Jonathan Mitchel <jonathan.mitchel3@gmail.com>

Description Single-cell Interpretable Tensor Decomposition (scITD) employs the Tucker tensor decomposition to extract multicell-type gene expression patterns that vary across donors/individuals. This tool is geared for use with single-cell RNA-sequencing datasets consisting of many source donors. The method has a wide range of potential applications, including the study of inter-individual variation at the population-level, patient sub-grouping/stratification, and the analysis of sample-level batch effects. Each ``multicellular process'' that is extracted consists of (A) a multi cell type gene loadings matrix and (B) a corresponding donor scores vector indicating the level at which the corresponding loadings matrix is expressed in each donor. Additional methods are implemented to aid in selecting an appropriate number of factors and to evaluate stability of the decomposition. Additional tools are provided for downstream analysis, including integration of gene set enrichment analysis and ligand-receptor analysis. Tucker, L.R. (1966) <doi:10.1007/BF02289464>. Unkel, S., Hannachi, A., Trendafilov, N. T., & Jolliffe, I. T. (2011) <doi:10.1007/s13253-011-0055-9>. Zhou, G., & Cichocki, A. (2012) <doi:10.2478/v10175-012-0051-4>.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.0.0), Matrix

biocViews

Imports rTensor, ica, fgsea, circlize, reshape2, parallel, ComplexHeatmap, ggplot2, mgcv, utils, Rcpp, RColorBrewer, dplyr, edgeR, sva, stats, Rmisc, ggpubr, msigdbr, sccore, NMF

Suggests methods, knitr, rmarkdown, testthat, coda.base, grid, ssvd, simplifyEnrichment, WGCNA, cowplot, matrixStats, stringr, zoo, rlang, AnnotationDbi, GO.db, conos, pagoda2, betareg, slam, tm

RoxygenNote 7.1.2

LinkingTo Rcpp, RcppArmadillo, RcppProgress

NeedsCompilation yes

Author Jonathan Mitchel [cre, aut],
Evan Biederstedt [aut],
Peter Kharchenko [aut]

Repository CRAN

Date/Publication 2022-03-23 18:20:02 UTC

R topics documented:

apply_combat — *Apply ComBat batch correction to pseudobulk matrices. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Apply ComBat batch correction to pseudobulk matrices. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
apply_combat(container, batch_var)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`batch_var`	character A batch variable from metadata to remove

Value

The project container with the batc corrected pseudobulked matrices.

`calculate_fiber_fstats` *Calculate F-Statistics for the association between donor scores for each factor donor values of shuffled gene_ctype fibers*

Description

Calculate F-Statistics for the association between donor scores for each factor donor values of shuffled gene_ctype fibers

Usage

```
calculate_fiber_fstats(tensor_data, tucker_results, s_fibers)
```

Arguments

`tensor_data` list The tensor data including donor, gene, and cell type labels as well as the tensor array itself

`tucker_results` list The results from Tucker decomposition. Includes a scores matrix as the first element and the loadings tensor unfolded as the second element.

`s_fibers` list Gene and cell type indices for the randomly selected fibers

Value

A numeric vector of F-statistics for associations between all shuffled fibers and donor scores.

`check_rec_pres` *Helper function to check whether receptor is present in target cell type*

Description

Helper function to check whether receptor is present in target cell type

Usage

```
check_rec_pres(
  container,
  lig_ct_exp,
  rec_elements,
  target_ct,
  percentile_exp_rec
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`lig_ct_exp` numeric Scaled expression for a ligand in the source cell type

`rec_elements` character One or more components of a receptor complex

`target_ct` character The name of the target cell type

`percentile_exp_rec`
numeric The percentile of ligand expression above which all donors need to have at least 5 cells expressing the receptor.

Value

A logical indicating whether receptor is present or not.

`clean_data` *Clean data to remove genes only expressed in a few cells and donors with very few cells. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Clean data to remove genes only expressed in a few cells and donors with very few cells. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
clean_data(container, donor_min_cells = 5)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`donor_min_cells`
numeric Minimum threshold for number of cells per donor (default=5)

Value

The project container with cleaned counts matrices in each container$scMinimal_ctype$<ctype>$count_data.

colMeanVars *Calculates column mean and variance. Adapted from pagoda2. https://github.com/kharchenkolab/pagoda2/blob/main/src/misc2.cpp*

Description

Calculates column mean and variance. Adapted from pagoda2. https://github.com/kharchenkolab/pagoda2/blob/main/src/mis

Usage

```
colMeanVars(sY, rowSel, ncores = 1L)
```

Arguments

`sY`	sparse matrix Gene by cell matrix of counts
`rowSel`	numeric The selected rows (genes)
`ncores`	numeric The number of cores

Value

data.frame with columns of mean, variance, and number of observeatios for each gene across samples

Examples

```
library(Matrix)
donor_by_gene <- rbind(c(9,2,1,5), c(3,3,1,2))
donor_by_gene <- Matrix(donor_by_gene, sparse = TRUE)
result <- colMeanVars(donor_by_gene, rowSel = NULL, ncores=1)
```

compare_decompositions *Plot a pairwise comparison of factors from two separate decompositions*

Description

Plot a pairwise comparison of factors from two separate decompositions

Usage

```
compare_decompositions(
  tucker_res1,
  tucker_res2,
  decomp_names,
  meta_anno1 = NULL,
  meta_anno2 = NULL,
  use_text = TRUE
)
```

Arguments

`tucker_res1`	list The container$tucker_res from first decomposition
`tucker_res2`	list The container$tucker_res from first decomposition
`decomp_names`	character Names of the two decompositions that will go on the axes of the heatmap
`meta_anno1`	matrix The result of calling get_meta_associations() corresponding to the first decomposition, which is stored in container$meta_associations (default=NULL)
`meta_anno2`	matrix The result of calling get_meta_associations() corresponding to the second decomposition, which is stored in container$meta_associations (default=NULL)
`use_text`	logical If TRUE, then displays correlation coefficients in cells (default=TRUE)

Value

No return value, as the resulting plots are drawn.

Examples

```
test_container <- run_tucker_ica(test_container, ranks=c(2,4),
tucker_type='regular', rotation_type='hybrid')
tucker_res1 <- test_container$tucker_results
test_container <- run_tucker_ica(test_container, ranks=c(2,4),
tucker_type='regular', rotation_type='ica_dsc')
tucker_res2 <- test_container$tucker_results
compare_decompositions(tucker_res1,tucker_res2,c('hybrid_method','ica_method'))
```

compute_associations *Compute associations between donor proportions and factor scores*

Description

Compute associations between donor proportions and factor scores

Usage

```
compute_associations(donor_balances, donor_scores, stat_type)
```

Arguments

`donor_balances`	matrx The balances computed from donor cell type proportions
`donor_scores`	data.frame The donor scores matrix from tucker results
`stat_type`	character Either "fstat" to get F-Statistics, "adj_rsq" to get adjusted R-squared values, or "adj_pval" to get adjusted pvalues.

Value

A numeric vector of association statistics (one for each factor)

`compute_donor_props` *Get donor proportions of each cell type or subtype*

Description

Get donor proportions of each cell type or subtype

Usage

```
compute_donor_props(clusts, metadata)
```

Arguments

`clusts`	integer Cluster assignments for each cell with names as cell barcodes
`metadata`	data.frame The $metadata field for the given scMinimal

Value

A data.frame of cluster proportions for each donor.

`compute_LR_interact` *Compute and plot the LR interactions for one factor*

Description

Compute and plot the LR interactions for one factor

Usage

```
compute_LR_interact(
  container,
  lr_pairs,
  sig_thresh = 0.05,
  percentile_exp_rec = 0.75,
  add_ld_fact_sig = TRUE,
  ncores = container$experiment_params$ncores
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`lr_pairs` data.frame Data of ligand-receptor pairs. First column should be ligands and second column should be one or more receptors separated by an underscore such as receptor1_receptor2 in the case that multiple receptors are required for signaling.

`sig_thresh` numeric The p-value significance threshold to use for module- factor associations and ligand-factor associations (default=0.05)

`percentile_exp_rec`
numeric The percentile above which the top donors expressing the ligand all must be expressing the receptor (default=0.75)

`add_ld_fact_sig`
logical Set to TRUE to append a heatmap showing significance of associations between each ligand hit and each factor (default=TRUE)

`ncores` numeric The number of cores to use (default=container$experiment_params$ncores)

Value

The LR analysis results heatmap as ComplexHeatmap object. Adjusted p-values for all results are placed in container$lr_res.

`convert_gn` *Convert gene identifiers to gene symbols*

Description

Convert gene identifiers to gene symbols

Usage

```
convert_gn(container, genes)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`genes` character Vector of the gene identifiers to be converted to gene symbols

Value

A character vector of gene symbols.

count_word *count_word. From older version of simplifyEnrichment package.*

Description

count_word. From older version of simplifyEnrichment package.

Usage

```
count_word(term, exclude_words = NULL)
```

Arguments

`term`	A vector of description texts.
`exclude_words`	The words that should be excluded.

Value

A data frame with words and frequencies.

determine_ranks_tucker *Run rank determination by svd on the tensor unfolded along each mode*

Description

Run rank determination by svd on the tensor unfolded along each mode

Usage

```
determine_ranks_tucker(
  container,
  max_ranks_test,
  shuffle_level = "cells",
  shuffle_within = NULL,
  num_iter = 100,
  batch_var = NULL,
  norm_method = "trim",
  scale_factor = 10000,
  scale_var = TRUE,
  var_scale_power = 0.5,
  seed = container$experiment_params$rand_seed
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`max_ranks_test`	numeric Vector of length 2 specifying the maximum number of donor and gene ranks to test
`shuffle_level`	character Either "cells" to shuffle cell-donor linkages or "tensor" to shuffle values within the tensor (default="cells")
`shuffle_within`	character A metadata variable to shuffle cell-donor linkages within (default=NULL)
`num_iter`	numeric Number of null iterations (default=100)
`batch_var`	character A batch variable from metadata to remove. No batch correction applied if NULL. (default=NULL)
`norm_method`	character The normalization method to use on the pseudobulked count data. Set to 'regular' to do standard normalization of dividing by library size. Set to 'trim' to use edgeR trim-mean normalization, whereby counts are divided by library size times a normalization factor. (default='trim')
`scale_factor`	numeric The number that gets multiplied by fractional counts during normalization of the pseudobulked data (default=10000)
`scale_var`	logical TRUE to scale the gene expression variance across donors for each cell type. If FALSE then all genes are scaled to unit variance across donors for each cell type. (default=TRUE)
`var_scale_power`	numeric Exponent of normalized variance that is used for variance scaling. Variance for each gene is initially set to unit variance across donors (for a given cell type). Variance for each gene is then scaled by multiplying the unit scaled values by each gene's normalized variance (where the effect of the mean-variance dependence is taken into account) to the exponent specified here. If NULL, uses var_scale_power from container$experiment_params. (default=.5)
`seed`	numeric Seed passed to set.seed() (default=container$experiment_params$rand_seed)

Value

The project container with a cowplot figure of rank determination plots in container$plots$rank_determination_plot.

Examples

```
test_container <- determine_ranks_tucker(test_container, max_ranks_test=c(3,5),
shuffle_level='tensor', num_iter=4, norm_method='trim', scale_factor=10000,
scale_var=TRUE, var_scale_power=.5)
```

form_tensor *Form the pseudobulk tensor as preparation for running the tensor decomposition.*

Description

Form the pseudobulk tensor as preparation for running the tensor decomposition.

Usage

```
form_tensor(
  container,
  donor_min_cells = 5,
  norm_method = "trim",
  scale_factor = 10000,
  vargenes_method = "norm_var",
  vargenes_thresh = 500,
  batch_var = NULL,
  scale_var = TRUE,
  var_scale_power = 0.5,
  custom_genes = NULL,
  verbose = TRUE
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`donor_min_cells` numeric Minimum threshold for number of cells per donor (default=5)

`norm_method` character The normalization method to use on the pseudobulked count data. Set to 'regular' to do standard normalization of dividing by library size. Set to 'trim' to use edgeR trim-mean normalization, whereby counts are divided by library size times a normalization factor. (default='trim')

`scale_factor` numeric The number that gets multiplied by fractional counts during normalization of the pseudobulked data (default=10000)

`vargenes_method` character The method by which to select highly variable genes from each cell type. Set to 'anova' to select genes by anova. Set to 'norm_var' to select the top genes by normalized variance or 'norm_var_pvals' to select genes by significance of their overdispersion (default='norm_var')

`vargenes_thresh` numeric The threshold to use in variable gene selection. For 'anova' and 'norm_var_pvals' this should be a p-value threshold. For 'norm_var' this should be the number of most variably expressed genes to select from each cell type (default=500) `batch_var` character A batch variable from metadata to remove (default=NULL)

`scale_var` logical TRUE to scale the gene expression variance across donors for each cell type. If FALSE then all genes are scaled to unit variance across donors for each cell type. (default=TRUE)

`var_scale_power`
numeric Exponent of normalized variance that is used for variance scaling. Variance for each gene is initially set to unit variance across donors (for a given cell type). Variance for each gene is then scaled by multiplying the unit scaled values by each gene's normalized variance (where the effect of the mean-variance dependence is taken into account) to the exponent specified here. If NULL, uses var_scale_power from container$experiment_params. (default=.5)

`custom_genes` character A vector of genes to include in the tensor. Overrides the default gene selection if not NULL. (default=NULL)

`verbose` logical Set to TRUE to print out progress (default=TRUE)

Value

The project container with a list of tensor data added in the container$tensor_data slot.

Examples

```
test_container <- form_tensor(test_container, donor_min_cells=0,
norm_method='trim', scale_factor=10000, vargenes_method='norm_var', vargenes_thresh=500,
scale_var = TRUE, var_scale_power = 1.5)
```

---

get_all_lds_factor_plots

Generate loadings heatmaps for all factors

Description

Generate loadings heatmaps for all factors

Usage

```
get_all_lds_factor_plots(
  container,
  use_sig_only = FALSE,
  nonsig_to_zero = FALSE,
  annot = "none",
  pathways_list = NULL,
  sim_de_donor_group = NULL,
  sig_thresh = 0.05,
  display_genes = FALSE,
  gene_callouts = FALSE,
  callout_n_gene_per_ctype = 5,
  callout_ctypes = NULL,
  show_var_explained = TRUE,
  reset_other_factor_plots = TRUE
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`use_sig_only` logical If TRUE, includes only significant genes from jackstraw in the heatmap. If FALSE, includes all the variable genes. (default = FALSE)

`nonsig_to_zero` logical If TRUE, makes the loadings of all nonsignificant genes 0 (default=FALSE)

`annot` character If set to "pathways" then creates an adjacent heatmap showing which genes are in which pathways. If set to "sig_genes" then creates an adjacent heatmap showing which genes were significant from jackstraw. If set to "none" no adjacent heatmap is plotted. (default="none")

`pathways_list` list A list of sets of pathways for each factor. List index should be the number corresponding to the factor. (default=NULL)

`sim_de_donor_group`
numeric To plot the ground truth significant genes from a simulation next to the heatmap, put the number of the donor group that corresponds to the factor being plotted. Here it should be a vector corresponding to the factors. (default=NULL)

`sig_thresh` numeric Pvalue significance threshold to use. If use_sig_only is TRUE the threshold is used as a cutoff for genes to include. If annot is "sig_genes" this value is used in the gene significance colormap as a minimum threshold. (default=0.05)

`display_genes` logical If TRUE, displays the names of gene names (default=FALSE)

`gene_callouts` logical If TRUE, then adds gene callout annotations to the heatmap (default=FALSE)

`callout_n_gene_per_ctype`
numeric To use if gene_callouts is TRUE. Sets the number of largest magnitude significant genes from each cell type to include in gene callouts. (default=5)

`callout_ctypes` list To use if gene_callouts is TRUE. Specifies which cell types to get gene callouts for. Each entry of the list should be a character vector of ctypes for the respective factor. If NULL, then gets gene callouts for largest magnitude significant genes for all cell types. (default=NULL)

`show_var_explained`
logical If TRUE then shows an anottation with the explained variance for each cell type (default=TRUE)

`reset_other_factor_plots`
logical If TRUE then removes any existing loadings plots (default=TRUE)

Value

The project container with the list of all loadings heatmap plots placed in container$plots$all_lds_plots.

Examples

```
test_container <- get_all_lds_factor_plots(test_container)
```

---

get_callouts_annot *Get gene callout annotations for a loadings heatmap*

Description

Get gene callout annotations for a loadings heatmap

Usage

```
get_callouts_annot(
  container,
  tmp_casted_num,
  factor_select,
  sig_thresh,
  top_n_per_ctype = 5,
  ctypes = NULL
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`tmp_casted_num` matrix The gene by cell type loadings matrix

`factor_select` numeric The factor to investigate

`sig_thresh` numeric Pvalue cutoff for significant genes

`top_n_per_ctype` numeric The number of significant, largest magnitude genes from each cell type to generate callouts for (default=5)

`ctypes` character The cell types for which to get the top genes to make callouts for. If NULL then uses all cell types. (default=NULL)

Value

A HeatmapAnnotation object for the gene callouts.

`get_ctype_exp_var` *Get explained variance of the reconstructed data using one cell type from one factor*

Description

Get explained variance of the reconstructed data using one cell type from one factor

Usage

```
get_ctype_exp_var(container, factor_use, ctype)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`factor_use`	numeric The factor to get variance explained for
`ctype`	character The cell type to get variance explained for

Value

The explained variance numeric value for one cell type of one factor.

`get_ctype_prop_associations` *Compute and plot associations between donor factor scores and donor proportions of major cell types*

Description

Compute and plot associations between donor factor scores and donor proportions of major cell types

Usage

```
get_ctype_prop_associations(container, stat_type, n_col = 2)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`stat_type`	character Either "fstat" to get F-Statistics, "adj_rsq" to get adjusted R-squared values, or "adj_pval" to get adjusted pvalues.
`n_col`	numeric The number of columns to organize the plots into (default=2)

Value

The project container with a cowplot figure of results plots in container$plots$ctype_prop_factor_associations.

get_ctype_subc_prop_associations

Compute and plot associations between donor factor scores and donor proportions of cell subtypes

Description

Compute and plot associations between donor factor scores and donor proportions of cell subtypes

Usage

```
get_ctype_subc_prop_associations(
  container,
  ctype,
  res,
  n_col = 2,
  alt_name = NULL
)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
ctype	character The cell type to get results for
res	numeric The clustering resolution to retrieve
n_col	numeric The number of columns to organize the plots into (default=2)
alt_name	character Alternate name for the cell type used in clustering (default=NULL)

Value

The project container with a cowplot figure of results plots in container$plots$ctype_prop_factor_associations.

`get_ctype_vargenes` *Partition main gene by cell matrix into per cell type matrices with significantly variable genes only. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Partition main gene by cell matrix into per cell type matrices with significantly variable genes only. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
get_ctype_vargenes(
  container,
  method,
  thresh,
  ncores = container$experiment_params$ncores,
  seed = container$experiment_params$rand_seed
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`method`	character The method used to select significantly variable genes across donors within a cell type. Can be either "anova" to use basic anova with cells grouped by donor or "norm_var" to get the top overdispersed genes by normalized variance. Set to "norm_var_pvals" to use normalized variance p-values as calculated in pagoda2.
`thresh`	numeric A pvalue threshold to use for gene significance when method is set to "anova" or "empir". For the method "norm_var" thresh is the number of top overdispersed genes from each cell type to include.
`ncores`	numeric The number of cores to use (default=container$experiment_params$ncores)
`seed`	numeric Seed passed to set.seed() (default=container$experiment_params$rand_seed)

Value

The project container with pseudobulk matrices limted to the selected most variable genes.

get_donor_meta *Get metadata matrix of dimensions donors by variables (not per cell)*

Description

Get metadata matrix of dimensions donors by variables (not per cell)

Usage

```
get_donor_meta(container, additional_meta = NULL, only_analyzed = TRUE)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
additional_meta	character A vector of other variables to include (default=NULL)
only_analyzed	logical Set to TRUE to only include donors that were included in the formed tensor, otherwise set to FALSE (default=TRUE)

Value

The project container with metadata per donor (not per cell) in container$donor_metadata.

Examples

```
test_container <- get_donor_meta(test_container, additional_meta='lanes')
```

get_factor_exp_var *Get the explained variance of the reconstructed data using one factor*

Description

Get the explained variance of the reconstructed data using one factor

Usage

```
get_factor_exp_var(container, factor_use)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
factor_use	numeric The factor to investigate

Value

The explained variance numeric value for one factor.

`get_fstats_pvals` *Calculate adjusted p-values for gene_celltype fiber-donor score associations*

Description

Calculate adjusted p-values for gene_celltype fiber-donor score associations

Usage

```
get_fstats_pvals(fstats_real, fstats_shuffled)
```

Arguments

`fstats_real`	numeric A vector of F-Statistics for gene-cell type-factor combinations
`fstats_shuffled`	numeric A vector of null F-Statistics

Value

A vector of adjusted p-values for associations of the unshuffled fibers with factor donor scores.

`get_gene_modules` *Compute WGCNA gene modules for each cell type*

Description

Compute WGCNA gene modules for each cell type

Usage

```
get_gene_modules(container, sft_thresh)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`sft_thresh`	numeric A vector indicating the soft threshold to use for each cell type. Length should be the same as container$experiment_params$ctypes_use

Value

The project container with WGCNA gene co-expression modules added. The module eigengenes for each cell type are in container$module_eigengenes, and the module genes for each cell type are in container$module_genes.

get_gene_set_vectors *Get logical vectors indicating which genes are in which pathways*

Description

Get logical vectors indicating which genes are in which pathways

Usage

```
get_gene_set_vectors(container, gene_sets, tmp_casted_num)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
gene_sets	character Vector of gene sets to extract genes for
tmp_casted_num	matrix The gene by cell type loadings matrix

Value

A list of the logical vectors for each pathway.

get_indv_subtype_associations *Compute subtype proportion-factor association p-values for all subclusters of a given major cell type*

Description

Compute subtype proportion-factor association p-values for all subclusters of a given major cell type

Usage

```
get_indv_subtype_associations(container, donor_props, factor_select)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
donor_props	matrix Donor proportions of subtypes
factor_select	numeric The factor to get associations for

Value

A vector of association statistics each cell subtype against a selected factor.

get_intersecting_pathways

Extract the intersection of gene sets which are enriched in two or more cell types for a factor

Description

Extract the intersection of gene sets which are enriched in two or more cell types for a factor

Usage

```
get_intersecting_pathways(
  container,
  factor_select,
  these_ctypes_only,
  up_down,
  thresh = 0.05
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`factor_select`	numeric The factor to investigate
`these_ctypes_only`	character A vector of cell types for which to get gene sets that are enriched in all of these and not in any other cell types
`up_down`	character Set to "up" to get the gene sets for the positive loading genes. Set to "down" to get the gene sets for the negative loadings genes.
`thresh`	numeric Pvalue significance threshold for selecting enriched sets (default=0.05)

Value

A vector of the intersection of pathways that are significantly enriched in two or more cell types for a factor.

get_leading_edge_genes

Get the leading edge genes from GSEA results

Description

Get the leading edge genes from GSEA results

Usage

```
get_leading_edge_genes(container, factor_select, gsets, num_genes_per = 5)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`factor_select`	numeric The factor to get results for
`gsets`	character A vector of gene set names to get leading edge genes for.
`num_genes_per`	numeric The maximum number of leading edge genes to get for each gene set (default=5)

Value

A named character vector of gene sets, with leading edge genes as the names.

`get_lm_pvals` *Compute gene-factor associations using univariate linear models*

Description

Compute gene-factor associations using univariate linear models

Usage

```
get_lm_pvals(container, n.cores = container$experiment_params$ncores)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`n.cores`	Number of cores to use (default = container$experiment_params$ncores)

Value

The project container with a vector of adjusted p-values for the gene-factor associations in container$gene_score_associations.

Examples

```
test_container <- get_lm_pvals(test_container, n.cores=1)
```

get_max_correlations *Computes the max correlation between each factor of the decomposition done using the whole dataset to each factor computed using the subsampled/bootstrapped dataset*

Description

Computes the max correlation between each factor of the decomposition done using the whole dataset to each factor computed using the subsampled/bootstrapped dataset

Usage

```
get_max_correlations(res_full, res_sub, res_use)
```

Arguments

`res_full`	matrix Either the donor scores or loadings matrix from the original decomposition
`res_sub`	matrix Either the donor scores or loadings matrix from the new decomposition
`res_use`	character Can either be 'loadings' or 'dscores' and should correspond with the data matrix used

Value

a vector of the max correlations for each original factor

get_meta_associations *Get metadata associations with factor donor scores*

Description

Get metadata associations with factor donor scores

Usage

```
get_meta_associations(container, vars_test, stat_use = "rsq")
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`vars_test`	character The names of meta variables to get associations for
`stat_use`	character Set to either 'rsq' to get r-squared values or 'pval' to get adjusted pvalues (default='rsq')

Value

The project container with a matrix of metadata associations with each factor in container$meta_associations.

Examples

```
test_container <- get_meta_associations(test_container, vars_test='lanes', stat_use='pval')
```

get_min_sig_genes

Evaluate the minimum number for significant genes in any factor for a given number of factors extracted by the decomposition

Description

Evaluate the minimum number for significant genes in any factor for a given number of factors extracted by the decomposition

Usage

```
get_min_sig_genes(
  container,
  donor_rank_range,
  gene_ranks,
  use_lm = TRUE,
  tucker_type = "regular",
  rotation_type = "hybrid",
  n_fibers = 100,
  n_iter = 500,
  n.cores = container$experiment_params$ncores,
  thresh = 0.05
)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses. Should have
donor_rank_range	numeric Range of possible number of donor factors to use.
gene_ranks	numeric The number of gene ranks to use in the decomposition
use_lm	logical Set to true to use get_lm_pvals otherwise uses jackstraw (default=TRUE)
tucker_type	character Set to 'regular' to run regular tucker or to 'sparse' to run tucker with sparsity constraints (default='regular')
rotation_type	character Set to 'hybrid' to perform hybrid rotation on resulting donor factor matrix and loadings. Otherwise set to 'ica_lds' to perform ica rotation on loadings or ica_dsc to perform ica on donor scores. (default='hybrid')

`n_fibers` numeric The number of fibers the randomly shuffle in each jackstraw iteration (default=100)

`n_iter` numeric The number of jackstraw shuffling iterations to complete (default=500)

`n.cores` Number of cores to use in get_lm_pvals() (default = container$experiment_params$ncores)

`thresh` numeric Pvalue threshold for significant genes in calculating the number of significant genes identified per factor. (default=0.05)

Value

The project container with a plot of the minimum significant genes for each decomposition with varying number of donor factors located in container$plots$min_sig_genes.

Examples

```
test_container <- get_min_sig_genes(test_container, donor_rank_range=c(2:4),
gene_ranks=4, tucker_type='regular', rotation_type='hybrid', n.cores=1)
```

---

get_module_enr — *Identify gene sets that are enriched within specified gene co-regulatory modules. Uses a hypergeometric test for over-representation. Used in plot_multi_module_enr().*

Description

Identify gene sets that are enriched within specified gene co-regulatory modules. Uses a hypergeometric test for over-representation. Used in plot_multi_module_enr().

Usage

```
get_module_enr(container, ctype, mod_select, db_use = "GO", adjust_pval = TRUE)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`ctype` character The name of cell type for the cell type module to test

`mod_select` numeric The module number for the cell type module to test

`db_use` character The database of gene sets to use. Database options include "GO", "Reactome", "KEGG", "BioCarta", "Hallmark", "TF", and "immuno". More than one database can be used. (default="GO")

`adjust_pval` logical Set to TRUE to apply FDR correction (default=TRUE)

Value

A vector of p-values for the tested gene sets.

get_normalized_variance *Get normalized variance for each gene, taking into account mean-variance trend*

Description

Get normalized variance for each gene, taking into account mean-variance trend

Usage

```
get_normalized_variance(container)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

Value

The project container with vectors of normalized variances values in scMinimal objects for each cell type. Generally, this should be done through calling the form_tensor() wrapper function.

get_num_batch_ranks *Plot factor-batch associations for increasing number of donor factors*

Description

Plot factor-batch associations for increasing number of donor factors

Usage

```
get_num_batch_ranks(
  container,
  donor_ranks_test,
  gene_ranks,
  batch_var,
  thresh = 0.5,
  tucker_type = "regular",
  rotation_type = "hybrid"
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`donor_ranks_test` numeric The number of donor rank values to test

`gene_ranks` numeric The number of gene ranks to use throughout

`batch_var` character The name of the batch meta variable

`thresh` numeric The threshold r-squared cutoff for considering a factor to be a batch factor. Can be a vector of multiple values to get plots at varying thresholds. (default=0.5)

`tucker_type` character Set to 'regular' to run regular tucker or to 'sparse' to run tucker with sparsity constraints (default='regular')

`rotation_type` character Set to 'hybrid' to optimize loadings via our hybrid method (see paper for details). Set to 'ica_dsc' to perform ICA rotation on resulting donor factor matrix. Set to 'ica_lds' to optimize loadings by the ICA rotation. (default='hybrid')

Value

A ggpubr figure of ggplot objects showing batch-factor associations and placed in container$plots$num_batch_factors slot

Examples

```
test_container <- get_num_batch_ranks(test_container, donor_ranks_test=c(2:4),
gene_ranks=10, batch_var='lanes', thresh=0.5, tucker_type='regular', rotation_type='hybrid')
```

`get_one_factor` *Get the donor scores and loadings matrix for a single-factor*

Description

Get the donor scores and loadings matrix for a single-factor

Usage

```
get_one_factor(container, factor_select)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`factor_select` numeric The number corresponding to the factor to extract

Value

A list with the first element as the donor scores and the second element as the corresponding loadings matrix for one factor.

Examples

```
f1_res <- get_one_factor(test_container, factor_select=1)
```

get_pseudobulk *Collapse data from cell-level to donor-level via summing counts. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Collapse data from cell-level to donor-level via summing counts. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
get_pseudobulk(container, shuffle = FALSE, shuffle_within = NULL)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
shuffle	logical Set to TRUE to shuffle cell-donor linkages (default=FALSE)
shuffle_within	character A metadata variable to shuffle cell-donor linkages within (default=NULL)

Value

The project container with pseudobulked count matrices in container$scMinimal_ctype$<ctype>$pseudobulk slots for each cell type.

get_real_fstats *Get F-Statistics for the real (non-shuffled) gene_ctype fibers*

Description

Get F-Statistics for the real (non-shuffled) gene_ctype fibers

Usage

```
get_real_fstats(container, ncores)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`ncores`	numeric The number of cores to use

Value

A vector F-statistics for each gene_celltype-factor association of the unshuffled data.

`get_reconstruct_errors_svd` *Calculate reconstruction errors using svd approach*

Description

Calculate reconstruction errors using svd approach

Usage

```
get_reconstruct_errors_svd(tnsr, max_ranks_test, shuffle_tensor)
```

Arguments

`tnsr`	array A 3-dimensional array with dimensions of donors, genes, and cell types in that order
`max_ranks_test`	numeric Vector of length 3 with maximum number of ranks to test for donor, gene, and cell type modes in that order
`shuffle_tensor`	logical Set to TRUE to shuffle values within the tensor

Value

A list of reconstruction errors for each mode of the tensor.

`get_significance_vectors` *Get vectors indicating which genes are significant in which cell types for a factor of interest*

Description

Get vectors indicating which genes are significant in which cell types for a factor of interest

Usage

```
get_significance_vectors(container, factor_select, ctypes)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
factor_select	numeric The factor to query
ctypes	character The cell types used in all the analysis ordered as they appear in the loadings matrix

Value

A list of the adjusted p-values for expression of each gene in each cell type in association with a factor of interest.

get_subclusters *Perform leiden subclustering to get cell subtypes*

Description

Perform leiden subclustering to get cell subtypes

Usage

```
get_subclusters(
  container,
  ctype,
  resolution,
  min_cells_group = 50,
  small_clust_action = "merge"
)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
ctype	character The cell type to do subclustering for
resolution	numeric The leiden resolution to use
min_cells_group	numeric The minimum allowable cluster size (default=50)
small_clust_action	character Either ’remove’ to remove subclusters or ’merge’ to merge clusters below min_cells_group threshold to the nearest cluster above the size threshold (default=’merge’)

Value

A vector of cell subclusters.

get_subclust_de_hmaps *Get list of cell subtype differential expression heatmaps*

Description

Get list of cell subtype differential expression heatmaps

Usage

```
get_subclust_de_hmaps(container, all_ctypes, all_res)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
all_ctypes	character A vector of the cell types to include
all_res	numeric A vector of resolutions matching the all_ctypes parameter

Value

A list of cell subcluster DE marker gene heatmaps as grob objects.

get_subclust_enr_dotplot *Get scatter plot for association of a cell subtype proportion with scores for a factor*

Description

Get scatter plot for association of a cell subtype proportion with scores for a factor

Usage

```
get_subclust_enr_dotplot(
  container,
  ctype,
  res,
  subtype,
  factor_use,
  ctype_cur = ctype
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`ctype` character The cell type to plot

`res` numeric The subcluster resolution to use

`subtype` numeric The number corresponding with the subtype of the major cell type to plot

`factor_use` numeric The factor to plot

`ctype_cur` character The name of the major cell type used in the main analysis

Value

A ggplot object of each donor's cell subcluster proportions against donor scores for a selected factor.

`get_subclust_enr_fig` *Get a figure showing cell subtype proportion associations with each factor. Combines this plot with subtype UMAPs and differential expression heatmaps. Note that this function runs better if the number of cores in the conos object in container$embedding has n.cores set to a relatively small value < 10.*

Description

Get a figure showing cell subtype proportion associations with each factor. Combines this plot with subtype UMAPs and differential expression heatmaps. Note that this function runs better if the number of cores in the conos object in container$embedding has n.cores set to a relatively small value < 10.

Usage

```
get_subclust_enr_fig(container, all_ctypes, all_res)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`all_ctypes` character A vector of the cell types to include

`all_res` numeric A vector of resolutions matching the all_ctypes parameter

Value

A cowplot figure placed in the slot container$plots$subc_fig.

get_subclust_enr_hmap *Get heatmap of subtype proportion associations for each celltype/subtype and each factor*

Description

Get heatmap of subtype proportion associations for each celltype/subtype and each factor

Usage

```
get_subclust_enr_hmap(container, all_ctypes, all_res, all_factors)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
all_ctypes	character A vector of the cell types to include
all_res	numeric A vector of resolutions matching the all_ctypes parameter
all_factors	numerc A vector of the factors to compute associations for

Value

A ComplexHeatmap object in container$plots$subc_enr_hmap showing the univariate associations between cell subcluster proportions and each factor.

get_subclust_umap *Get a figure to display subclusterings at multiple resolutions*

Description

Get a figure to display subclusterings at multiple resolutions

Usage

```
get_subclust_umap(container, all_ctypes, all_res, n_col = 3)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
all_ctypes	character A vector of the cell types to include
all_res	numeric A vector of resolutions matching the all_ctypes parameter
n_col	numeric The number of columns to organize the figure into (default=3)

Value

The project container with a cowplot figure of all UMAP plots in container$plots$subc_umap_fig and the individual umap plots in container$plots$subc_umaps

---

get_subtype_prop_associations

Compute and plot associations between factor scores and cell subtype composition for various clustering resolution parameters

---

Description

Compute and plot associations between factor scores and cell subtype composition for various clustering resolution parameters

Usage

```
get_subtype_prop_associations(
  container,
  max_res,
  stat_type,
  integration_var = NULL,
  min_cells_group = 50,
  use_existing_subc = FALSE,
  alt_ct_names = NULL,
  n_col = 2
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`max_res` numeric The maximum clustering resolution to use. Minimum is 0.5.

`stat_type` character Either "fstat" to get F-Statistics, "adj_rsq" to get adjusted R-squared values, or "adj_pval" to get adjusted pvalues.

`integration_var` character The meta data variable to use for creating the joint embedding with Conos if not already provided in container$embedding (default=NULL)

`min_cells_group` numeric The minimum allowable size for cell subpopulations (default=50)

`use_existing_subc` logical Set to TRUE to use existing subcluster annotations (default=FALSE)

`alt_ct_names` character Cell type names used in clustering if different from those used in the main analysis. Should match the order of container$experiment_params$ctypes_use. (default=NULL)

`n_col` numeric The number of columns to organize the plots into (default=2)

Value

The project container with a cowplot figure of cell subtype proportion-factor association results plots in container$plots$subtype_prop_factor_associations.

---

get_sums *Calculates factor-stratified sums for each column. Adapted from pagoda2. https://github.com/kharchenkolab/pagoda2/blob/main/src/misc2.cpp*

Description

Calculates factor-stratified sums for each column. Adapted from pagoda2. https://github.com/kharchenkolab/pagoda2/blob/m

Usage

```
get_sums(sY, rowSel)
```

Arguments

`sY`	sparse matrix Gene by cell matrix of counts
`rowSel`	factor The donor that each cell is from

Value

matrix of summed counts per gene per sample

---

ht_clusters *Visualize the similarity matrix and the clustering. Adapted from simplifyEnrichment package. https://github.com/jokergoo/simplifyEnrichment/blob/master/R/ht_clusters.R*

Description

Visualize the similarity matrix and the clustering. Adapted from simplifyEnrichment package. https://github.com/jokergoo/simplifyEnrichment/blob/master/R/ht_clusters.R

Usage

```
ht_clusters(
  mat,
  cl,
  dend = NULL,
  col = c("white", "red"),
 draw_word_cloud = simplifyEnrichment:::is_GO_id(rownames(mat)[1]) || !is.null(term),
  term = NULL,
  min_term = 5,
  order_by_size = FALSE,
  exclude_words = character(0),
  max_words = 10,
  word_cloud_grob_param = list(),
  fontsize_range = c(4, 16),
  column_title = NULL,
  ht_list = NULL,
  use_raster = TRUE,
  ...
)
```

Arguments

`mat` A similarity matrix.

`cl` Cluster labels inferred from the similarity matrix, e.g. from ‘cluster_terms‘ or ‘binary_cut‘.

`dend` Used internally.

`col` A vector of colors that map from 0 to the 95^th percentile of the similarity values.

`draw_word_cloud` Whether to draw the word clouds.

`term` The full name or the description of the corresponding GO IDs.

`min_term` Minimal number of functional terms in a cluster. All the clusters with size less than “min_term“ are all merged into one separated cluster in the heatmap.

`order_by_size` Whether to reorder clusters by their sizes. The cluster that is merged from small clusters (size < “min_term“) is always put to the bottom of the heatmap.

`exclude_words` Words that are excluded in the word cloud.

`max_words` Maximal number of words visualized in the word cloud.

`word_cloud_grob_param` A list of graphic parameters passed to ‘word_cloud_grob‘.

`fontsize_range` The range of the font size. The value should be a numeric vector with length two. The minimal font size is mapped to word frequency value of 1 and the maximal font size is mapped to the maximal word frequency. The font size interlopation is linear.

`column_title` Column title for the heatmap.

`ht_list` A list of additional heatmaps added to the left of the similarity heatmap.

`use_raster`	Whether to write the heatmap as a raster image.
`...`	other parameters

Value

A list containing a ‘ComplexHeatmap::HeatmapList-class‘ object and GO term ordering.

`identify_sex_metadata` *Extract metadata for sex information if not provided already*

Description

Extract metadata for sex information if not provided already

Usage

```
identify_sex_metadata(container, y_gene = "RPS4Y1", x_gene = "XIST")
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`y_gene`	character Gene name to use for identifying male donors (default='RPS4Y1')
`x_gene`	character Gene name to use for identifying female donors (default='XIST')

Value

The project container with sex metadata added to the metadata.

`initialize_params` *Initialize parameters to be used throughout scITD in various functions*

Description

Initialize parameters to be used throughout scITD in various functions

Usage

```
initialize_params(ctypes_use, ncores = 4, rand_seed = 10)
```

Arguments

`ctypes_use`	character Names of the cell types to use for the analysis (default=NULL)
`ncores`	numeric Number of cores to use (default=4)
`rand_seed`	numeric Random seed to use (default=10)

Value

A list of the experiment parameters to use.

Examples

```
param_list <- initialize_params(ctypes_use = c("CD4+ T", "CD8+ T"),
ncores = 1, rand_seed = 10)
```

instantiate_scMinimal *Create an scMinimal object. Generally, this should be done through calling the make_new_container() wrapper function.*

Description

Create an scMinimal object. Generally, this should be done through calling the make_new_container() wrapper function.

Usage

```
instantiate_scMinimal(
  count_data,
  meta_data,
  metadata_cols = NULL,
  metadata_col_nm = NULL
)
```

Arguments

`count_data`	sparseMatrix Matrix of raw counts with genes as rows and cells as columns
`meta_data`	data.frame Metadata with cells as rows and variables as columns. Number of rows in metadata should equal number of columns in count matrix.
`metadata_cols`	character The names of the metadata columns to use (default=NULL)
`metadata_col_nm`	character New names for the selected metadata columns if wish to change their names. If NULL, then the preexisting column names are used. (default=NULL)

Value

An scMinimal object holding counts and metadata for a project.

Examples

```
scMinimal <- instantiate_scMinimal(count_data=test_container$scMinimal_full$count_data,
meta_data=test_container$scMinimal_full$metadata)
```

make_new_container *Create a container to store all data and results for the project. You must provide a params list as generated by initialize_params(). You also need to provide either a Seurat object or both a count_data matrix and a meta_data matrix.*

Description

Create a container to store all data and results for the project. You must provide a params list as generated by initialize_params(). You also need to provide either a Seurat object or both a count_data matrix and a meta_data matrix.

Usage

```
make_new_container(
  params,
  count_data = NULL,
  meta_data = NULL,
  seurat_obj = NULL,
  scMinimal = NULL,
  gn_convert = NULL,
  metadata_cols = NULL,
  metadata_col_nm = NULL,
  label_donor_sex = FALSE
)
```

Arguments

`params`	list A list of the experiment params to use as generated by initialize_params()
`count_data`	dgCMatrix Matrix of raw counts with genes as rows and cells as columns (default=NULL)
`meta_data`	data.frame Metadata with cells as rows and variables as columns. Number of rows in metadata should equal number of columns in count matrix (default=NULL)
`seurat_obj`	Seurat object that has been cleaned and includes the normalized, log-transformed counts. The meta.data should include a column with the header 'sex' and values of 'M' or 'F' if available. The metadata should also have a column with the header 'ctypes' with the corresponding names of the cell types as well as a column with header 'donors' that contains identifiers for each donor. (default=NULL)
`scMinimal`	environment A sub-container for the project typically consisting of gene expression data in its raw and processed forms as well as metadata (default=NULL)
`gn_convert`	data.frame Gene identifier -> gene name conversions table. Gene identifiers used in counts matrices should appear in the first column and the corresponding gene symbols should appear in the second column. Can remain NULL if the identifiers are already gene symbols. (default=NULL)

`metadata_cols`	character The names of the metadata columns to use (default=NULL)
`metadata_col_nm`	character New names for the selected metadata columns if wish to change their names. If NULL, then the preexisting column names are used. (default=NULL)
`label_donor_sex`	logical Set to TRUE to label donor sex in the meta data by using expressing of sex-associated genes (default=FALSE)

Value

A project container of class environment that stores sub-containers for each cell type as well as results and plots from all analyses.

`merge_small_clusts` *Merge small subclusters into larger ones*

Description

Merge small subclusters into larger ones

Usage

```
merge_small_clusts(con, clusts, min_cells_group)
```

Arguments

`con`	conos Object for the dataset with umap projection and groups as cell types
`clusts`	character The initially assigned subclusters by leiden clustering
`min_cells_group`	numeric The minimum allowable cluster size

Value

The subcluster labels with small clusters below the size threshold merged into the nearest larger cluster.

`nmf_unfolded` *Computes non-negative matrix factorization on the tensor unfolded along the donor dimension*

Description

Computes non-negative matrix factorization on the tensor unfolded along the donor dimension

Usage

```
nmf_unfolded(container, ranks)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`ranks`	numeric The number of factors to extract. Unlike with the Tucker decomposition, this should be a single number.

Value

The project container with results of the decomposition in container$tucker_results. The results object is a list with the donor scores matrix in the first element and the unfolded loadings matrix in the second element.

Examples

```
test_container <- nmf_unfolded(test_container, 2)
```

`normalize_counts` *Helper function to normalize and log-transform count data*

Description

Helper function to normalize and log-transform count data

Usage

```
normalize_counts(count_data, scale_factor = 10000)
```

Arguments

`count_data`	matrix or sparse matrix Gene by cell matrix of counts
`scale_factor`	numeric The number that gets multiplied by fractional counts during normalization of the pseudobulked data (default=10000)

Value

The normalized, log-transformed matrix.

normalize_pseudobulk *Normalize the pseudobulked counts matrices. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Normalize the pseudobulked counts matrices. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
normalize_pseudobulk(container, method = "trim", scale_factor = 10000)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`method` character The normalization method to use on the pseudobulked count data. Set to 'regular' to do standard normalization of dividing by library size. Set to 'trim' to use edgeR trim-mean normalization, whereby counts are divided by library size times a normalization factor. (default='trim')

`scale_factor` numeric The number that gets multiplied by fractional counts during normalization of the pseudobulked data (default=10000)

Value

The project container with normalized pseudobulk matrices in container$scMinimal_ctype$<ctype>$pseudobulk slots.

norm_var_helper *Calculates the normalized variance for each gene. This is adapted from pagoda2. https://github.com/kharchenkolab/pagoda2/blob/main/R/Pagoda2.R Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Calculates the normalized variance for each gene. This is adapted from pagoda2. https://github.com/kharchenkolab/pagoda2/l
Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
norm_var_helper(scMinimal)
```

Arguments

scMinimal	environment A sub-container for the project typically consisting of gene expression data in its raw and processed forms as well as metadata

Value

A list with the first element containing a vector of the normalized variance for each gene and the second element containing log-transformed adjusted p-values for the overdispersion of each gene.

---

parse_data_by_ctypes *Parse main counts matrix into per-celltype-matrices. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Parse main counts matrix into per-celltype-matrices. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
parse_data_by_ctypes(container)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

Value

The project container with separate scMinimal objects per cell type in the container$scMinimal_ctype slot

pca_unfolded *Computes singular-value decomposition on the tensor unfolded along the donor dimension*

Description

Computes singular-value decomposition on the tensor unfolded along the donor dimension

Usage

```
pca_unfolded(container, ranks)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
ranks	numeric The number of factors to extract. Unlike with the Tucker decomposition, this should be a single number.

Value

The project container with results of the decomposition in container$tucker_results. The results object is a list with the donor scores matrix in the first element and the unfolded loadings matrix in the second element.

Examples

```
test_container <- pca_unfolded(test_container, 2)
```

plotDEheatmap_conos *Plot a heatmap of differential genes. Code is adapted from Conos package. https://github.com/kharchenkolab/conos/blob/master/R/plot.R*

Description

Plot a heatmap of differential genes. Code is adapted from Conos package. https://github.com/kharchenkolab/conos/blob/mas

Usage

```
plotDEheatmap_conos(
  con,
  groups,
  container,
  de = NULL,
  min.auc = NULL,
  min.specificity = NULL,
  min.precision = NULL,
  n.genes.per.cluster = 10,
  additional.genes = NULL,
  exclude.genes = NULL,
  labeled.gene.subset = NULL,
  expression.quantile = 0.99,
  pal = (grDevices::colorRampPalette(c("dodgerblue1", "grey95", "indianred1")))(1024),
  ordering = "-AUC",
  column.metadata = NULL,
  show.gene.clusters = TRUE,
  remove.duplicates = TRUE,
  column.metadata.colors = NULL,
  show.cluster.legend = TRUE,
  show_heatmap_legend = FALSE,
  border = TRUE,
  return.details = FALSE,
  row.label.font.size = 10,
  order.clusters = FALSE,
  split = FALSE,
  split.gap = 0,
  cell.order = NULL,
  averaging.window = 0,
  ...
)
```

Arguments

`con` conos (or p2) object

`groups` groups in which the DE genes were determined (so that the cells can be ordered correctly)

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`de` differential expression result (list of data frames)

`min.auc` optional minimum AUC threshold

`min.specificity` optional minimum specificity threshold

`min.precision` optional minimum precision threshold

`n.genes.per.cluster` number of genes to show for each cluster

`additional.genes`
optional additional genes to include (the genes will be assigned to the closest cluster)

`exclude.genes` an optional list of genes to exclude from the heatmap

`labeled.gene.subset`
a subset of gene names to show (instead of all genes). Can be a vector of gene names, or a number of top genes (in each cluster) to show the names for.

`expression.quantile`
expression quantile to show (0.98 by default)

`pal` palette to use for the main heatmap

`ordering` order by which the top DE genes (to be shown) are determined (default "-AUC")

`column.metadata`
additional column metadata, passed either as a data.frame with rows named as cells, or as a list of named cell factors.

`show.gene.clusters`
whether to show gene cluster color codes

`remove.duplicates`
remove duplicated genes (leaving them in just one of the clusters)

`column.metadata.colors`
a list of color specifications for additional column metadata, specified according to the HeatmapMetadata format. Use "clusters" slot to specify cluster colors.

`show.cluster.legend`
whether to show the cluster legend

`show_heatmap_legend`
whether to show the expression heatmap legend

`border` show borders around the heatmap and annotations

`return.details` if TRUE will return a list containing the heatmap (ha), but also raw matrix (x), expression list (expl) and other info to produce the heatmap on your own.

`row.label.font.size`
font size for the row labels

`order.clusters` whether to re-order the clusters according to the similarity of the expression patterns (of the genes being shown)

`split` logical If TRUE splits the heatmap by cell type (default=FALSE)

`split.gap` numeric The distance to put in the gaps between split parts of the heatmap if split=TRUE (default=0)

`cell.order` explicitly supply cell order

`averaging.window`
optional window averaging between neighboring cells within each group (turned off by default) - useful when very large number of cells shown (requires zoo package)

`...` extra parameters are passed to pheatmap

Value

ComplexHeatmap::Heatmap object (see return.details param for other output)

plot_donor_matrix *Plot matrix of donor scores extracted from Tucker decomposition*

Description

Plot matrix of donor scores extracted from Tucker decomposition

Usage

```
plot_donor_matrix(
  container,
  meta_vars = NULL,
  cluster_by_meta = NULL,
  show_donor_ids = FALSE,
  add_meta_associations = NULL,
  show_var_explained = TRUE,
  donors_sel = NULL,
  h_w = NULL
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`meta_vars`	character Names of metadata variables to plot alongside the donor scores. Can include more than one variable. (default=NULL)
`cluster_by_meta`	character One metadata variable to cluster the heatmap by. If NULL, donor clustering is done using donor scores. (default=NULL)
`show_donor_ids`	logical Set to TRUE to show donor id as row name on the heamap (default=FALSE)
`add_meta_associations`	character Adds meta data associations with each factor as top annotation. These should be generated first with plot_meta_associations(). Set to 'pval' if used 'pval' in plot_meta_associations(), otherwise set to 'rsq'. If NULL, no annotation is added. (default=NULL)
`show_var_explained`	logical Set to TRUE to display the explained variance for each factor (default=TRUE)
`donors_sel`	character A vector of a subset of donors to include in the plot (default=NULL)
`h_w`	numeric Vector specifying height and width (defualt=NULL)

Value

The project container with a heatmap plot of donor scores in container$plots$donor_matrix.

Examples

```
test_container <- plot_donor_matrix(test_container, show_donor_ids = TRUE)
```

plot_donor_props *Plot donor celltype/subtype proportions against each factor*

Description

Plot donor celltype/subtype proportions against each factor

Usage

```
plot_donor_props(
  donor_props,
  donor_scores,
  significance,
  ctype_mapping = NULL,
  stat_type = "adj_pval",
  n_col = 2
)
```

Arguments

donor_props	data.frame Donor proportions as output from compute_donor_props()
donor_scores	data.frame Donor scores from tucker results
significance	numeric F-Statistics as output from compute_associations()
ctype_mapping	character The cell types corresponding with columns of donor_props (default=NULL)
stat_type	character Either "fstat" to get F-Statistics, "adj_rsq" to get adjusted R-squared values, or "adj_pval" to get adjusted pvalues (default='adj_pval')
n_col	numeric The number of columns to organize the plots into (default=2)

Value

A cowplot figure of ggplot objects for proportions of each cell type against donor factor scores for each factor.

plot_donor_sig_genes *Generate a gene by donor heatmap showing scaled expression of top loading genes for a given factor*

Description

Generate a gene by donor heatmap showing scaled expression of top loading genes for a given factor

Usage

```
plot_donor_sig_genes(
  container,
  factor_select,
  top_n_per_ctype,
  ctypes_use = NULL,
  show_donor_labels = FALSE,
  additional_meta = NULL,
  add_genes = NULL
)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
factor_select	numeric The factor to query
top_n_per_ctype	numeric Vector of the number of top genes from each cell type to plot
ctypes_use	character The cell types for which to get the top genes to make callouts for. If NULL then uses all cell types. (default=NULL)
show_donor_labels	logical Set to TRUE to display donor labels (default=FALSE)
additional_meta	character Another meta variable to plot (default=NULL)
add_genes	character Additional genes to plot for all ctypes (default=NULL)

Value

The project container with a heatmap plot in the slot container$plots$donor_sig_genes$<Factor#>. This heatmap shows scaled expression of top loading genes in each cell type for a selected factor.

Examples

```
test_container <- plot_donor_sig_genes(test_container, factor_select=1,
top_n_per_ctype=2)
```

plot_dscore_enr *Compute enrichment of donor metadata categorical variables at high/low factor scores*

Description

Compute enrichment of donor metadata categorical variables at high/low factor scores

Usage

```
plot_dscore_enr(container, factor_use, meta_var)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`factor_use` numeric The factor to test

`meta_var` character The name of the metadata variable to test

Value

A cowplot figure of enrichment plots.

Examples

```
fig <- plot_dscore_enr(test_container, factor_use=1, meta_var='lanes')
```

`plot_gsea_hmap` *Plot enriched gene sets from all cell types in a heatmap*

Description

Plot enriched gene sets from all cell types in a heatmap

Usage

```
plot_gsea_hmap(container, factor_select, thresh)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`factor_select` numeric The factor to plot

`thresh` numeric Pvalue threshold to use for including gene sets in the heatmap

Value

A stacked heatmap object from ComplexHeatmap.

plot_gsea_hmap_w_similarity

Plot already computed enriched gene sets to show semantic similarity between sets

Description

Plot already computed enriched gene sets to show semantic similarity between sets

Usage

```
plot_gsea_hmap_w_similarity(
  container,
  factor_select,
  direc,
  thresh,
  exclude_words = character(0)
)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
factor_select	numeric The factor to plot
direc	character Set to either 'up' or 'down' to use the appropriate sets
thresh	numeric Pvalue threshold to use for including gene sets in the heatmap
exclude_words	character Vector of words to exclude from word cloud (default=character(0))

Value

No value is returned. A heatmap showing enriched gene sets clustered by semantic similarity is drawn.

plot_gsea_sub

Look at enriched gene sets from a cluster of semantically similar gene sets. Uses the results from previous run of plot_gsea_hmap_w_similarity()

Description

Look at enriched gene sets from a cluster of semantically similar gene sets. Uses the results from previous run of plot_gsea_hmap_w_similarity()

Usage

```
plot_gsea_sub(container, clust_select, thresh = 0.05)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`clust_select`	numeric The cluster to plot gene sets from. On the previous semantic similarity plot, cluster numbering starts from the top as 1.
`thresh`	numeric Color threshold to use for showing significance (default=0.05)

Value

A heatmap plot from ComplexHeatmap showing one semantic similarity cluster of enriched gene sets with adjusted p-values for each cell type.

`plot_loadings_annot` *Plot the gene by celltype loadings for a factor*

Description

Plot the gene by celltype loadings for a factor

Usage

```
plot_loadings_annot(
  container,
  factor_select,
  use_sig_only = FALSE,
  nonsig_to_zero = FALSE,
  annot = "none",
  pathways = NULL,
  sim_de_donor_group = NULL,
  sig_thresh = 0.05,
  display_genes = FALSE,
  gene_callouts = FALSE,
  callout_n_gene_per_ctype = 5,
  callout_ctypes = NULL,
  specific_callouts = NULL,
  le_set_callouts = NULL,
  le_set_colormap = NULL,
  le_set_num_per = 5,
  show_le_legend = FALSE,
  show_xlab = TRUE,
  show_var_explained = TRUE,
  clust_method = "median",
  h_w = NULL,
  reset_other_factor_plots = FALSE,
  draw_plot = TRUE
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`factor_select` numeric The factor to plot

`use_sig_only` logical If TRUE, includes only significant genes from jackstraw in the heatmap. If FALSE, includes all the variable genes. (default = FALSE)

`nonsig_to_zero` logical If TRUE, makes the loadings of all nonsignificant genes 0 (default=FALSE)

`annot` character If set to "pathways" then creates an adjacent heatmap showing which genes are in which pathways. If set to "sig_genes" then creates an adjacent heatmap showing which genes were significant from jackstraw. If set to "none" no adjacent heatmap is plotted. (default="none")

`pathways` character Gene sets to plot if annot is set to "pathways" (default=NULL)

`sim_de_donor_group`
numeric To plot the ground truth significant genes from a simulation next to the heatmap, put the number of the donor group that corresponds to the factor being plotted (default=NULL)

`sig_thresh` numeric Pvalue significance threshold to use. If use_sig_only is TRUE the threshold is used as a cutoff for genes to include. If annot is "sig_genes" this value is used in the gene significance colormap as a minimum threshold. (default=0.05)

`display_genes` logical If TRUE, displays the names of gene names (default=FALSE)

`gene_callouts` logical If TRUE, then adds gene callout annotations to the heatmap (default=FALSE)

`callout_n_gene_per_ctype`
numeric To use if gene_callouts is TRUE. Sets the number of largest magnitude significant genes from each cell type to include in gene callouts. (default=5)

`callout_ctypes` character To use if gene_callouts is TRUE. Specifies which cell types to get gene callouts for. If NULL, then gets gene callouts for largest magnitude significant genes for all cell types. (default=NULL)

`specific_callouts`
character A vector of gene names to show callouts for (default=NULL)

`le_set_callouts`
character Pass a vector of gene set names to show leading edge genes for a select set of gene sets (default=NULL)

`le_set_colormap`
character A named vector with names as gene sets and values as colors. If NULL, then selects first n colors of Set3 color palette. (default=NULL)

`le_set_num_per` numeric The number of leading edge genes to show for each gene set (default=5)

`show_le_legend` logical Set to TRUE to show the color map legend for leading edge genes (default=FALSE) `show_xlab` logical If TRUE, displays the xlabel ’genes’ (default=TRUE)

`show_var_explained` logical If TRUE then shows an anotation with the explained variance for each cell type (default=TRUE)

`clust_method` character The hclust method to use for clustering rows (default=’median’)

`h_w` numeric Vector specifying height and width (defualt=NULL)

`reset_other_factor_plots` logical Set to TRUE to set all other loadings plots to NULL. Useful if run get_all_lds_factor_plots but then only want to show one or two plots. (default=FALSE)

`draw_plot` logical Set to TRUE to show the plot. Plot is stored regardless. (default=TRUE)

Value

The project container with a heatmap of loadings for one factor put in container$plots$all_lds_plots. The legend for the heatmap is put in container$plots$all_legends. Use draw(<hmap obj>,annotation_legend_list = <hmap legend obj>) to re-render the plot with legend.

Examples

```
test_container <- plot_loadings_annot(test_container, 1, display_genes=FALSE,
show_var_explained = TRUE)
```

---

plot_mod_and_lig *Plot trio of associations between ligand expression, module eigengenes, and factor scores*

Description

Plot trio of associations between ligand expression, module eigengenes, and factor scores

Usage

```
plot_mod_and_lig(container, factor_select, mod_ct, mod, lig_ct, lig)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`factor_select` numeric The factor to use

`mod_ct` character The name of the cell type for the corresponding module

`mod` numeric The number of the corresponding module

`lig_ct` character The name of the cell type where the ligand is expressed

`lig` character The name of the ligand to use

Value

A cowplot figure of ggplot objects for the three associations scatter plots.

`plot_multi_module_enr` *Generate gene set x ct_module heatmap showing co-expression module gene set enrichment results*

Description

Generate gene set x ct_module heatmap showing co-expression module gene set enrichment results

Usage

```
plot_multi_module_enr(
  container,
  ctypes,
  modules,
  sig_thresh = 0.05,
  db_use = "TF",
  max_plt_pval = 0.1,
  h_w = NULL
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`ctypes`	character A vector of cell type names corresponding to the module numbers in mod_select, specifying the modules to compute enrichment for
`modules`	numeric A vector of module numbers corresponding to the cell types in ctype, specifying the modules to compute enrichment for
`sig_thresh`	numeric P-value threshold for results to include. Only shows a given gene set if at least one module has a result lower than the threshold. (default=0.05)
`db_use`	character The database of gene sets to use. Database options include "GO", "Reactome", "KEGG", "BioCarta", "Hallmark", "TF", and "immuno". More than one database can be used. (default="GO")
`max_plt_pval`	max pvalue shown on plot, but not used to remove rows like sig_thresh (default=.1)
`h_w`	numeric Vector specifying height and width (defualt=NULL)

Value

A ComplexHeatmap object of enrichment results.

plot_rec_errors_bar_svd

Plot reconstruction errors as bar plot for svd method

Description

Plot reconstruction errors as bar plot for svd method

Usage

```
plot_rec_errors_bar_svd(real, shuffled, mode_to_show)
```

Arguments

real	list The real reconstruction errors
shuffled	list The reconstruction errors under null model
mode_to_show	numeric The mode to plot the results for

Value

A ggplot object showing the difference in reconstruction errors for successive factors.

plot_rec_errors_line_svd

Plot reconstruction errors as line plot for svd method

Description

Plot reconstruction errors as line plot for svd method

Usage

```
plot_rec_errors_line_svd(real, shuffled, mode_to_show)
```

Arguments

real	list The real reconstruction errors
shuffled	list The reconstruction errors under null model
mode_to_show	numeric The mode to plot the results for

Value

A ggplot object showing relative reconstruction errors.

plot_scores_by_meta *Plot dotplots for each factor to compare donor scores between metadata groups*

Description

Plot dotplots for each factor to compare donor scores between metadata groups

Usage

```
plot_scores_by_meta(container, meta_var)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`meta_var` character The meta data variable to compare groups for

Value

The project container with a figure of comparison plots (one for each factor) placed in container$plots$indv_meta_scores_asso

plot_select_sets *Plot enrichment results for hand picked gene sets*

Description

Plot enrichment results for hand picked gene sets

Usage

```
plot_select_sets(
  container,
  factors_all,
  sets_plot,
  color_sets = NULL,
  cl_rows = FALSE,
  h_w = NULL,
  myfontsize = 8
)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
factors_all	numeric Vector of one or more factor numbers to get plots for
sets_plot	character Vector of gene set names to show enrichment values for
color_sets	named character Values are colors corresponding to each set, with names as the gene set names (default=NULL)
cl_rows	logical Set to TRUE to cluster gene set results (default=FALSE)
h_w	numeric Vector specifying height and width (defualt=NULL)
myfontsize	numeric Gene set label fontsize (default=8)

Value

A list with a ComplexHeatmap object of select enriched gene sets as the first element and with a legend object as the second element.

plot_stability_results

Generate a plot for either the donor scores or loadings stability test

Description

Generate a plot for either the donor scores or loadings stability test

Usage

```
plot_stability_results(container, plt_data)
```

Arguments

container	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
plt_data	character Either 'lds' or 'dsc' and indicates which plot to make

Value

the plot

plot_subclust_associations *Plot association significances for varying clustering resolutions*

Description

Plot association significances for varying clustering resolutions

Usage

```
plot_subclust_associations(res, n_col = 2)
```

Arguments

res	data.frame Regression statistics for each subcluster analysis
n_col	numeric The number of columns to organize the plots into (default=2)

Value

A cowplot of ggplot objects showing statistics for regressions of proportions of each cell subtype (at varying clustering resolutions) against each factor.

prep_LR_interact *Prepare data for LR analysis and get soft thresholds to use for gene modules*

Description

Prepare data for LR analysis and get soft thresholds to use for gene modules

Usage

```
prep_LR_interact(
  container,
  lr_pairs,
  norm_method = "trim",
  scale_factor = 10000,
  var_scale_power = 0.5,
  batch_var = NULL
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`lr_pairs`	data.frame Data of ligand-receptor pairs. First column should be ligands and second column should be one or more receptors separated by an underscore such as receptor1_receptor2 in the case that multiple receptors are required for signaling.
`norm_method`	character The normalization method to use on the pseudobulked count data. Set to 'regular' to do standard normalization of dividing by library size. Set to 'trim' to use edgeR trim-mean normalization, whereby counts are divided by library size times a normalization factor. (default='trim')
`scale_factor`	numeric The number that gets multiplied by fractional counts during normalization of the pseudobulked data (default=10000)
`var_scale_power`	numeric Exponent of normalized variance that is used for variance scaling. Variance for each gene is initially set to unit variance across donors (for a given cell type). Variance for each gene is then scaled by multiplying the unit scaled values by each gene's normalized variance (where the effect of the mean-variance dependence is taken into account) to the exponent specified here. If NULL, uses var_scale_power from container$experiment_params. (default=.5)
`batch_var`	character A batch variable from metadata to remove (default=NULL)

Value

The project container with added container$scale_pb_extra slot that contains the tensor with additional ligands and receptors. Also has container$no_scale_pb_extra slot with pseudobulked, normalized data that is not scaled.

`project_new_data` *Project multicellular patterns to get scores on new data*

Description

Project multicellular patterns to get scores on new data

Usage

```
project_new_data(new_container, old_container)
```

Arguments

`new_container`	environment A project container with new data to project scores for. The form_tensor() function should be run.
`old_container`	environment The original project container that has the multicellular gene expression patterns already extracted. These patterns will be projected onto the new data.

Value

The new container environment object with projected scores in new_container$projected_scores. The factors will be ordered the same as the factors in old_container.

`reduce_dimensions` *Gets a conos object of the data, aligning datasets across a specified variable such as batch or donors. This can be run independently or through get_subtype_prop_associations().*

Description

Gets a conos object of the data, aligning datasets across a specified variable such as batch or donors. This can be run independently or through get_subtype_prop_associations().

Usage

```
reduce_dimensions(
  container,
  integration_var,
  ncores = container$experiment_params$ncores
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`integration_var` character The meta data variable to use for creating the joint embedding with Conos.

`ncores` numeric The number of cores to use (default=container$experiment_params$ncores)

Value

The project container with a conos object in container$embedding.

`reduce_to_vargenes` *Reduce each cell type's expression matrix to just the significantly variable genes. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Reduce each cell type's expression matrix to just the significantly variable genes. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
reduce_to_vargenes(container)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

Value

The project container with pseudobulked matrices reduced to only the most variable genes.

render_multi_plots *Create a figure of all loadings plots arranged*

Description

Create a figure of all loadings plots arranged

Usage

```
render_multi_plots(container, data_type, max_cols = 3)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`data_type`	character Can be either "loadings", "gsea", or "dgenes". This determines which list of heatmaps to organize into the figure.
`max_cols`	numeric The max number of columns to plot. Can only either be 2 or 3 since these are large plots. (default=3)

Value

The multi-plot figure.

Examples

```
test_container <- get_all_lds_factor_plots(test_container)
fig <- render_multi_plots(test_container, data_type='loadings')
```

reshape_loadings *Reshape loadings for a factor from linearized to matrix form*

Description

Reshape loadings for a factor from linearized to matrix form

Usage

```
reshape_loadings(ldngs_row, genes, ctypes)
```

Arguments

`ldngs_row`	numeric A vector of loadings values for one factor
`genes`	character The gene identifiers corresponding to each loading
`ctypes`	character The cell type corresponding to each loading

Value

A loadings matrix with dimensions of genes by cell types.

run_fgsea *Run fgsea for one cell type of one factor*

Description

Run fgsea for one cell type of one factor

Usage

```
run_fgsea(
  container,
  factor_select,
  ctype,
  db_use = "GO",
  signed = TRUE,
  min_gs_size = 15,
  max_gs_size = 500,
  ncores = container$experiment_params$ncores
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`factor_select`	numeric The factor of interest
`ctype`	character The cell type of interest
`db_use`	character The database of gene sets to use. Database options include "GO", "Reactome", "KEGG", "BioCarta", and "Hallmark". More than one database can be used. (default="GO")
`signed`	logical If TRUE, uses signed gsea. If FALSE, uses unsigned gsea. Currently only works with fgsea method. (default=TRUE)
`min_gs_size`	numeric Minimum gene set size (default=15)
`max_gs_size`	numeric Maximum gene set size (default=500)
`ncores`	numeric The number of cores to use (default=container$experiment_params$ncores)

Value

A data.frame of the fgsea results for enrichment of gene sets in a given cell type for a given factor. The results contain adjusted p-values, normalized enrichment scores, leading edge genes, and other information output by fgsea.

`run_gsea_one_factor` *Run gsea separately for all cell types of one specified factor and plot results*

Description

Run gsea separately for all cell types of one specified factor and plot results

Usage

```
run_gsea_one_factor(
  container,
  factor_select,
  method = "fgsea",
  thresh = 0.05,
  db_use = "GO",
  signed = TRUE,
  min_gs_size = 15,
  max_gs_size = 500,
  reset_other_factor_plots = FALSE,
  draw_plot = TRUE,
  ncores = container$experiment_params$ncores
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`factor_select` numeric The factor of interest

`method` character The method of gsea to use. Can either be "fgsea", "fgsea_special or "hypergeometric". (default="fgsea")

`thresh` numeric Pvalue significance threshold to use. Will include gene sets in resulting heatmap if pvalue is below this threshold for at least one cell type. (default=0.05)

`db_use` character The database of gene sets to use. Database options include "GO", "Reactome", "KEGG", and "BioCarta". More than one database can be used. (default="GO")

`signed` logical If TRUE, uses signed gsea. If FALSE, uses unsigned gsea. Currently only works with fgsea method (default=TRUE)

`min_gs_size` numeric Minimum gene set size (default=15)

`max_gs_size` numeric Maximum gene set size (default=500)

`reset_other_factor_plots`
logical Set to TRUE to set all other gsea plots to NULL (default=FALSE)

`draw_plot` logical Set to TRUE to show the plot. Plot is stored regardless. (default=TRUE)

`ncores` numeric The number of cores to use (default=container$experiment_params$ncores)

Value

A stacked heatmap plot of the gsea results in the slot container$plots$gsea$<Factor#>. The heatmaps show adjusted p-values for the enrichment of each gene set in each cell type for the selected factor. The top heatmap shows enriched gene sets among the positive loading genes and the bottom heatmap shows enriched gene sets among the negative loading genes for the factor.

Examples

```
test_container <- run_gsea_one_factor(test_container, factor_select=1,
method="fgsea", thresh=0.05, db_use="Hallmark", signed=TRUE)
```

`run_hypergeometric_gsea` *Compute enriched gene sets among significant genes in a cell type for a factor using hypergeometric test*

Description

Compute enriched gene sets among significant genes in a cell type for a factor using hypergeometric test

Usage

```
run_hypergeometric_gsea(
  container,
  factor_select,
  ctype,
  up_down,
  thresh = 0.05,
  min_gs_size = 15,
  max_gs_size = 500,
  db_use = "GO"
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`factor_select`	numeric The factor of interest
`ctype`	character The cell type of interest
`up_down`	character Either "up" to compute enrichment among the significant positive loading genes or "down" to compute enrichment among the significant negative loading genes.
`thresh`	numeric Pvalue significance threshold. Used as cutoff for calling genes as significant to use for enrichment tests. (default=0.05)
`min_gs_size`	numeric Minimum gene set size (default=15)
`max_gs_size`	numeric Maximum gene set size (default=500)
`db_use`	character The database of gene sets to use. Database options include "GO", "Reactome", "KEGG", and "BioCarta". More than one database can be used. (default="GO")

Value

A vector of adjusted p-values for enrichment of gene sets in the significant genes of a given cell type in a given factor.

`run_jackstraw` *Run jackstraw to get genes that are significantly associated with donor scores for factors extracted by Tucker decomposition*

Description

Run jackstraw to get genes that are significantly associated with donor scores for factors extracted by Tucker decomposition

Usage

```
run_jackstraw(
  container,
  ranks,
  n_fibers = 100,
  n_iter = 500,
  tucker_type = "regular",
  rotation_type = "hybrid",
  seed = container$experiment_params$rand_seed,
  ncores = container$experiment_params$ncores
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`ranks` numeric The number of donor ranks and gene ranks to decompose to using Tucker decomposition

`n_fibers` numeric The number of fibers the randomly shuffle in each iteration (default=100)

`n_iter` numeric The number of shuffling iterations to complete (default=500)

`tucker_type` character Set to 'regular' to run regular tucker or to 'sparse' to run tucker with sparsity constraints (default='regular')

`rotation_type` character Set to 'hybrid' to perform hybrid rotation on resulting donor factor matrix and loadings. Otherwise set to 'ica_lds' to perform ica rotation on loadings or ica_dsc to perform ica on donor scores. (default='hybrid')

`seed` numeric Seed passed to set.seed() (default=container$experiment_params$rand_seed)

`ncores` numeric The number of cores to use (default=container$experiment_params$ncores)

Value

The project container with a vector of adjusted pvalues in container$gene_score_associations.

Examples

```
test_container <- run_jackstraw(test_container, ranks=c(2,4), n_fibers=2, n_iter=10,
tucker_type='regular', rotation_type='hybrid', ncores=1)
```

`run_stability_analysis`

Test stability of a decomposition by subsampling or bootstrapping donors. Note that running this function will replace the decomposition in the project container with one resulting from the tucker parameters entered here.

Description

Test stability of a decomposition by subsampling or bootstrapping donors. Note that running this function will replace the decomposition in the project container with one resulting from the tucker parameters entered here.

Usage

```
run_stability_analysis(
  container,
  ranks,
  tucker_type = "regular",
  rotation_type = "hybrid",
  sparsity = sqrt(2),
  subset_type = "subset",
  sub_prop = 0.75,
  n_iterations = 100,
  ncores = container$experiment_params$ncores
)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`ranks`	numeric The number of donor, gene, and cell type ranks, respectively, to decompose to using Tucker decomposition.
`tucker_type`	character Set to 'regular' to run regular tucker or to 'sparse' to run tucker with sparsity constraints (default='regular')
`rotation_type`	character Set to 'hybrid' to optimize loadings via our hybrid method (see paper for details). Set to 'ica_dsc' to perform ICA rotation on resulting donor factor matrix. Set to 'ica_lds' to optimize loadings by the ICA rotation. (default='hybrid')
`sparsity`	numeric To use with sparse tucker. Higher indicates more sparse (default=sqrt(2))
`subset_type`	character Set to either 'subset' or 'bootstrap' (default='subset')
`sub_prop`	numeric The proportion of donors to keep when using subset_type='subset' (default=.75)
`n_iterations`	numeric The number of iterations to perform (default=100)
`ncores`	numeric The number of cores to use (default=container$experiment_params$ncores)

Value

The project container with the donor scores stability plot in container$plots$stability_plot_dsc and the loadings stability plot in container$plots$stability_plot_lds

Examples

```
test_container <- run_stability_analysis(test_container, ranks=c(2,4),
tucker_type='regular', rotation_type='hybrid', subset_type='subset',
sub_prop=0.75, n_iterations=5, ncores=1)
```

`run_tucker_ica` *Run the Tucker decomposition and rotate the factors*

Description

Run the Tucker decomposition and rotate the factors

Usage

```
run_tucker_ica(
  container,
  ranks,
  tucker_type = "regular",
  rotation_type = "hybrid",
  sparsity = sqrt(2)
)
```

Arguments

`container` environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

`ranks` numeric The number of donor factors and gene factors, respectively, to decompose the data into. Since we rearrange the standard output of the Tucker decomposition to be 'donor centric', the number of donor factors will also be the total number of main factors that can be used for downstream analysis. The number of gene factors will only impact the quality of the decomposition.

`tucker_type` character Set to 'regular' to run regular tucker or to 'sparse' to run tucker with sparsity constraints. The 'sparse' method is still under development, so we recommend using 'regular'. (default='regular')

`rotation_type` character Set to 'hybrid' to optimize loadings via our hybrid method (see paper for details). Set to 'ica_dsc' to perform ICA rotation on resulting donor factor matrix. Set to 'ica_lds' to optimize loadings by the ICA rotation. (default='hybrid')

`sparsity` numeric To use with sparse tucker. Higher indicates more sparse (default=sqrt(2))

Value

The project container with results of the decomposition in container$tucker_results. The results object is a list with the donor scores matrix in the first element and the unfolded loadings matrix in the second element.

Examples

```
test_container <- run_tucker_ica(test_container,ranks=c(2,4))
```

sample_fibers *Get a list of tensor fibers to shuffle*

Description

Get a list of tensor fibers to shuffle

Usage

```
sample_fibers(tensor_data, n_fibers)
```

Arguments

`tensor_data`	list The tensor data including donor, gene, and cell type labels as well as the tensor array itself
`n_fibers`	numeric The number of fibers to get

Value

A list of gene and cell type indices for the randomly selected fibers

scale_fontsize *Scale font size. From simplifyEnrichment package. https://github.com/jokergoo/simplifyEnrichment/blob/master/R/ht_clusters.R*

Description

Scale font size. From simplifyEnrichment package. https://github.com/jokergoo/simplifyEnrichment/blob/master/R/ht_cluste

Usage

```
scale_fontsize(x, rg = c(1, 30), fs = c(4, 16))
```

Arguments

`x`	A numeric vector.
`rg`	The range.
`fs`	Range of the font size.

Value

A numeric vector.

scale_variance *Scale variance across donors for each gene within each cell type. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Scale variance across donors for each gene within each cell type. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
scale_variance(container, var_scale_power)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`var_scale_power`	numeric Exponent of normalized variance that is used for variance scaling. Variance for each gene is initially set to unit variance across donors (for a given cell type). Variance for each gene is then scaled by multiplying the unit scaled values by each gene's normalized variance (where the effect of the mean-variance dependence is taken into account) to the exponent specified here. If NULL, uses var_scale_power from container$experiment_params.

Value

The project container with the variance altered for each gene within the pseudobulked matrices for each cell type.

seurat_to_scMinimal *Convert Seurat object to scMinimal object. Generally, this should be done through calling the make_new_container() wrapper function.*

Description

Convert Seurat object to scMinimal object. Generally, this should be done through calling the make_new_container() wrapper function.

Usage

```
seurat_to_scMinimal(seurat_obj, metadata_cols = NULL, metadata_col_nm = NULL)
```

Arguments

`seurat_obj` Seurat object that has been cleaned and includes the normalized, log-transformed counts. The meta.data should include a column with the header 'sex' and values of 'M' or 'F' if available. The metadata should also have a column with the header 'ctypes' with the corresponding names of the cell types as well as a column with header 'donors' that contains identifiers for each donor.

`metadata_cols` character The names of the metadata columns to use (default=NULL)

`metadata_col_nm` character New names for the selected metadata columns if wish to change their names. If NULL, then the preexisting column names are used. (default=NULL)

Value

An scMinimal object holding counts and metadata for a project.

`shuffle_fibers` *Shuffle elements within the selected fibers*

Description

Shuffle elements within the selected fibers

Usage

```
shuffle_fibers(tensor_data, s_fibers)
```

Arguments

`tensor_data` list The tensor data including donor, gene, and cell type labels as well as the tensor array itself

`s_fibers` list Gene and cell type indices for the randomly selected fibers

Value

The tensor_data object with the values for the selected fibers shuffled.

`stack_tensor` *Create the tensor object by stacking each pseudobulk cell type matrix. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Create the tensor object by stacking each pseudobulk cell type matrix. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
stack_tensor(container)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses

Value

The project container with the list of tensor data in container$tensor_data.

`stop_wrap` *Helper function from simplifyEnrichment package. https://github.com/jokergoo/simplifyEnrichment/blob/master/R/utils.R*

Description

Helper function from simplifyEnrichment package. https://github.com/jokergoo/simplifyEnrichment/blob/master/R/utils.R

Usage

```
stop_wrap(...)
```

Arguments

`...`	other parameters

Value

No value is returned.

subset_scMinimal *Subset an scMinimal object by specified genes, donors, cells, or cell types*

Description

Subset an scMinimal object by specified genes, donors, cells, or cell types

Usage

```
subset_scMinimal(
  scMinimal,
  ctypes_use = NULL,
  cells_use = NULL,
  donors_use = NULL,
  genes_use = NULL,
  in_place = TRUE
)
```

Arguments

scMinimal	environment A sub-container for the project typically consisting of gene expression data in its raw and processed forms as well as metadata
ctypes_use	character The cell types to keep (default=NULL)
cells_use	character Cell barcodes for the cells to keep (default=NULL)
donors_use	character The donors to keep (default=NULL)
genes_use	character The genes to keep (default=NULL)
in_place	logical If set to TRUE then replaces the input object with the new subsetted object (default=TRUE)

Value

A subsetted scMinimal object.

Examples

```
cell_names <- colnames(test_container$scMinimal_full$count_data)
cells_sub <- sample(cell_names,40)
scMinimal <- subset_scMinimal(test_container$scMinimal_full,
cells_use=cells_sub)
```

test_container *Data container for testing tensor formation steps*

Description

Data container for testing tensor formation steps

Usage

```
test_container
```

Format

An object of class `environment` of length 10.

tucker_ica_helper *Helper function for running the decomposition. Use the run_tucker_ica() wrapper function instead.*

Description

Helper function for running the decomposition. Use the run_tucker_ica() wrapper function instead.

Usage

```
tucker_ica_helper(
  tensor_data,
  ranks,
  tucker_type,
  rotation_type,
  sparsity,
  projection_container = NULL
)
```

Arguments

`tensor_data`	list The tensor data including donor, gene, and cell type labels as well as the tensor array itself
`ranks`	numeric The number of donor and gene factors respectively, to decompose to using Tucker decomposition.
`tucker_type`	character Set to 'regular' to run regular tucker or to 'sparse' to run tucker with sparsity constraints
`rotation_type`	character Set to 'hybrid' to optimize loadings via our hybrid method (see paper for details). Set to 'ica_dsc' to perform ICA rotation on resulting donor factor matrix. Set to 'ica_lds' to optimize loadings by the ICA rotation.

`sparsity` numeric Higher indicates more sparse

`projection_container` environment A project container to store projection data in. Currently only implemented for 'hybrid' and 'ica_dsc' rotations. (default=NULL)

Value

The list of results for tucker decomposition with donor scores matrix in first element and loadings matrix in second element.

`tucker_sparse` *Tucker Decomposition adapted from rTensor but with sparsity constraints added. This function is still being tested, so use with caution. https://github.com/jamesyili/rTensor/blob/master/R/rTensor_Decomp.R*

Description

Tucker Decomposition adapted from rTensor but with sparsity constraints added. This function is still being tested, so use with caution. https://github.com/jamesyili/rTensor/blob/master/R/rTensor_Decomp.R

Usage

```
tucker_sparse(tnsr, ranks = NULL, max_iter = 25, tol = 1e-05, sparsity = 5)
```

Arguments

`tnsr` Tensor with K modes.

`ranks` numeric Vector of the modes of the output core Tensor (default=NULL)

`max_iter` numeric Maximum number of iterations if error stays above tol (default=25)

`tol` numeric Relative Frobenius norm error tolerance (default=1e-5)

`sparsity` numeric Higher is more sparse (default=5)

Value

A list containing all the Tucker decomposition results components.

`update_params` *Update any of the experiment-wide parameters*

Description

Update any of the experiment-wide parameters

Usage

```
update_params(container, ctypes_use = NULL, ncores = NULL, rand_seed = NULL)
```

Arguments

`container`	environment Project container that stores sub-containers for each cell type as well as results and plots from all analyses
`ctypes_use`	character Names of the cell types to use for the analysis (default=NULL)
`ncores`	numeric Number of cores to use (default=NULL)
`rand_seed`	numeric Random seed to use (default=NULL)

Value

The project container with updated experiment parameters in container$experiment_params.

Examples

```
test_container <- update_params(test_container, ncores=1)
```

`vargenes_anova` *Compute significantly variable genes via anova. Generally, this should be done through calling the form_tensor() wrapper function.*

Description

Compute significantly variable genes via anova. Generally, this should be done through calling the form_tensor() wrapper function.

Usage

```
vargenes_anova(scMinimal, ncores)
```

Arguments

`scMinimal`	environment A sub-container for the project typically consisting of gene expression data in its raw and processed forms
`ncores`	numeric Number of cores to use

Value

A list of raw p-values for each gene.

Index