Package 'dominoSignal'

November 12, 2026

Title Cell Communication Analysis for Single Cell RNA Sequencing

Version 1.7.1

Description dominoSignal is a package developed to analyze cell signaling through ligand-receptor-transcription factor networks in scRNAseq data. It takes as input information transcriptomic data, requiring counts, z-scored counts, and cluster labels, as well as information on transcription factor activity (such as from SCENIC) and a database of ligand and receptor pairings (such as from CellPhoneDB). This package creates an object storing ligand-receptor-transcription factor linkages by cluster and provides several methods for exploring, summarizing, and visualizing the analysis.

BugReports https://github.com/FertigLab/dominoSignal/issues

Depends R(>= 4.2.0),

Imports biomaRt, ComplexHeatmap, circlize, ggpubr, grDevices, grid, igraph, Matrix, methods, plyr, reshape2, stats, utils, purrr, dplyr

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData false

biocViews SystemsBiology, SingleCell, Transcriptomics, Network, Sequencing

Suggests knitr, patchwork, rmarkdown, Seurat, testthat, formatR, BiocFileCache, SingleCellExperiment, withr, kableExtra

VignetteBuilder knitr

Language en-US

URL https://FertigLab.github.io/dominoSignal/

Config/roxygen2/version 8.1.1

Config/testthat/edition 3

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

git_branch devel

git_last_commit 80f2f89

git_last_commit_date 2026-10-15

Repository Bioconductor 3.24

Date/Publication 2026-11-12

Author Christopher Cherry [aut] (ORCID:
<https://orcid.org/0000-0002-5481-0055>),
Jacob T Mitchell [aut] (ORCID: <https://orcid.org/0000-0002-5370-9692>),
Sushma Nagaraj [aut] (ORCID: <https://orcid.org/0000-0001-5166-1309>),
Kavita Krishnan [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1345-0249>),
Dmitrijs Lvovs [aut],
Elana Fertig [ctb] (ORCID: <https://orcid.org/0000-0003-3204-342X>),
Jennifer Elisseeff [ctb] (ORCID:
<https://orcid.org/0000-0002-5066-1996>)

Maintainer Kavita Krishnan <kkrishnan@som.umaryland.edu>

Contents

add_rl_column . . . 3
avg_exp_for_complexes . . . 4
build_domino . . . 4
CellPhoneDB . . . 5
check_arg . . . 6
circos_ligand_receptor . . . 7
convert_genes . . . 8
conv_py_bools . . . 8
cor_heatmap . . . 9
cor_scatter . . . 10
count_linkage . . . 11
create_domino . . . 12
create_regulon_list_scenic . . . 14
create_rl_map_cellphonedb . . . 14
domino-class . . . 16
DominoObjects . . . 17
dom_clusters . . . 17
dom_correlations . . . 18
dom_counts . . . 19
dom_database . . . 19
dom_de . . . 20
dom_info . . . 21
dom_linkages . . . 21
dom_network_items . . . 22
dom_signaling . . . 23
dom_tf_activation . . . 24
dom_to_df . . . 24
dom_zscores . . . 25
do_norm . . . 26
feat_heatmap . . . 26
filter_by_subject_names . . . 28
gene_network . . . 28
get_ligand_expression . . . 29
get_resolved_ligands . . . 30
get_signaling_info . . . 31
ggplot_col_gen . . . 31
incoming_signaling_heatmap . . . 32

lc . . . 33
LinkageSummary . . . 34
linkage_summary-class . . . 34
mean_exp_by_cluster . . . 35
mean_ligand_expression . . . 35
obtain_circos_expression . . . 36
PBMC . . . 37
plot_differential_linkages . . . 37
print,domino-method . . . 38
print,linkage_summary-method . . . 39
read_if_char . . . 39
rename_clusters . . . 40
render_circos_ligand_receptor . . . 40
resolve_complexes . . . 41
resolve_names . . . 42
SCENIC . . . 42
show,domino-method . . . 43
show,linkage_summary-method . . . 43
signaling_heatmap . . . 44
signaling_network . . . 45
subset,linkage_summary-method . . . 47
summarize_linkages . . . 48
table_convert_genes . . . 49
test_differential_linkages . . . 50

Index **52**

`add_rl_column` *Adds a column to the RL signaling data frame.*

Description

This function adds a column to the internal rl 'map' used to map all receptor and receptor complexes to all ligand and ligand complexes.

Usage

```
add_rl_column(map, map_ref, conv, new_name)
```

Arguments

`map`	RL signaling data frame.
`map_ref`	Name of column to match new data to
`conv`	Data frame matching current data in map to new data.
`new_name`	Name of new column to be created in RL map

Value

An updated RL signaling data frame

Examples

```
data(CellPhoneDB)
lr_name <- data.frame("abbrev" = c("L", "R"), "full" = c("Ligand", "Receptor"))
rl_map_expanded <- add_rl_column(
    map = CellPhoneDB$rl_map_tiny, map_ref = "type_A",
    conv = lr_name, new_name = "type_A_full"
)
```

---

avg_exp_for_complexes *Get average expression for complexes*

Description

Get average expression for complexes

Usage

```
avg_exp_for_complexes(exp_mat, complexes_list)
```

Arguments

exp_mat	A matrix(or dataframe) of genes x clusters, values are z-scores averaged over the clusters
complexes_list	A list similar to dom@linkages$complexes

Value

A list containing average expression for any complexes

---

build_domino *Calculate a signaling network for a* domino() *object*

Description

This function calculates a signaling network. It requires a domino() object preprocessed from create_domino() and returns a domino() object prepared for plotting with the various plotting functions in this package.

Usage

```
build_domino(
  dom,
  max_tf_per_clust = 5,
  max_tf_pval = 0.01,
  max_rec_per_tf = 5,
  rec_tf_cor_threshold = 0.15,
  min_rec_percentage = 0.1
)
```

Arguments

dom	A domino() object from create_domino().
max_tf_per_clust	Maximum number of transcription factors called active in a cluster.
max_tf_pval	Maximum p-value from differential feature score test to call a transcription factor active in a cluster, serving as a significance threshold (used if object was created with TF selection method "clusters"). For selection with "variable" or "all", this argument is ignored, and TFs are ranked by max correlation with active receptors by cluster.
max_rec_per_tf	Maximum number of receptors to link to each transcription factor.
rec_tf_cor_threshold	Minimum Spearman correlation used to consider a receptor linked with a transcription factor. Increasing this will decrease the number of receptors linked to each transcription factor.
min_rec_percentage	Minimum proportion of cells in cluster expressing a receptor (0 to 1) for the receptor to be linked to transcription factors in that cluster. For receptor complexes, every component gene must exceed this threshold.

Value

A domino() object with a signaling network built

See Also

create_domino() to create a domino() object

Examples

```
data(DominoObjects)
dom <- DominoObjects$dom_tiny

# a relaxed example
pbmc_dom_built_tiny <- build_domino(
    dom = dom, max_tf_pval = .05, max_tf_per_clust = Inf,
    max_rec_per_tf = Inf, rec_tf_cor_threshold = .1, min_rec_percentage = 0.01
)
```

CellPhoneDB *CellPhoneDB subset*

Description

A list of four subsets of CellPhoneDB v4.0.0 data and resulting domino rl_map.

Usage

```
data("CellPhoneDB")
```

Format

A list of:

genes_tiny A subset of the CellPhoneDB `gene_input.csv` file.

proteins_tiny A subset of the CellPhoneDB `protein_input.csv` file.

complexes_tiny A subset of the CellPhoneDB `complex_input.csv` file.

interactions_tiny A subset of the CellPhoneDB `interaction_input.csv` file.

rl_map_tiny The subset formatted as a receptor-ligand map by `create_rl_map_cellphonedb()`.

Source

https://github.com/ventolab/cellphonedb-data/archive/refs/tags/v4.0.0.tar.gz; subset with `data-raw/tiny_data.R`.

---

check_arg *Check input arguments*

Description

Accepts an object and rules to check against; stops if requirements are not met

Usage

```
check_arg(
  arg,
  allow_class = NULL,
  allow_len = NULL,
  allow_range = NULL,
  allow_values = NULL,
  need_vars = NULL,
  need_colnames = FALSE,
  need_rownames = FALSE,
  need_names = FALSE
)
```

Arguments

`arg`	the argument to check
`allow_class`	vector of allowed classes
`allow_len`	vector of allowed lengths
`allow_range`	range of minimum and maximum values i.e. c(1, 5)
`allow_values`	vector of allowed values
`need_vars`	vector of required variables
`need_colnames`	logical for whether colnames are required
`need_rownames`	logical for whether rownames are required
`need_names`	logical for whether names are required

Value

Logical indicating whether the argument meets the requirements

circos_ligand_receptor *Plot expression of a receptor's ligands by other clusters as a chord plot*

Description

Creates a chord plot of expression of ligands that can activate a specified receptor where chord widths correspond to mean ligand expression by the cluster.

Usage

```
circos_ligand_receptor(
  dom,
  receptor,
  ligand_expression_threshold = 0.01,
  cell_idents = NULL,
  cell_colors = NULL
)
```

Arguments

dom	A domino() object that has undergone network building with build_domino()
receptor	Name of a receptor active in at least one cell type in the domino() object
ligand_expression_threshold	Minimum mean expression value of a ligand by a cell type for a chord to be rendered between the cell type and the receptor
cell_idents	Vector of clusters from cluster assignments in the domino() object to be included in the plot.
cell_colors	Named vector of color names or hex codes where names correspond to the plotted cell types and values to the colors.

Value

Renders a circos plot to the active graphics device

See Also

Other plotting: cor_scatter(), plot_differential_linkages()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
# basic usage
circos_ligand_receptor(dom, receptor = "CXCR3")
# specify colors
cols <- c("red", "orange", "green")
names(cols) <- dom_clusters(dom)
circos_ligand_receptor(dom, receptor = "CXCR3", cell_colors = cols)
```

convert_genes *Use biomaRt to convert genes*

Description

This function reads in a vector of genes and converts the genes to specified symbol type

Usage

```
convert_genes(
  genes,
  from = c("ENSMUSG", "ENSG", "MGI", "HGNC"),
  to = c("MGI", "HGNC"),
  host = "https://www.ensembl.org"
)
```

Arguments

genes	Vector of genes to convert.
from	Format of gene input (ENSMUSG, ENSG, MGI, or HGNC)
to	Format of gene output (MGI or HGNC)
host	Host to connect to. Defaults to https://www.ensembl.org following the useMart default, but can be changed to archived hosts if useMart fails to connect.

Value

A data frame with input genes as column 1 and converted genes as column 2

conv_py_bools *Change cases of True/False syntax from Python to TRUE/FALSE R syntax*

Description

Change cases of True/False syntax from Python to TRUE/FALSE R syntax

Usage

```
conv_py_bools(obj)
```

Arguments

obj	object that will be converted

Value

The converted object

cor_heatmap *Create a heatmap of correlation between receptors and transcription factors*

Description

Creates a heatmap of correlation values between receptors and transcription factors either with boolean threshold or with continuous values displayed

Usage

```
cor_heatmap(
  dom,
  bool = FALSE,
  bool_thresh = 0.15,
  title = TRUE,
  feats = NULL,
  recs = NULL,
  mark_connections = FALSE,
  ...
)
```

Arguments

dom	A domino() object with network built (build_domino())
bool	Boolean indicating whether the heatmap should be continuous or boolean. If boolean then bool_thresh will be used to determine how to define activity as positive or negative.
bool_thresh	Numeric indicating the threshold separating 'on' or 'off' for feature activity if making a boolean heatmap.
title	Either a string to use as the title or a boolean describing whether to include a title. In order to pass the 'main' parameter to ComplexHeatmap::Heatmap() you must set title to FALSE.
feats	Either a vector of features to include in the heatmap or 'all' for all features. If left NULL then the features selected for the signaling network will be shown.
recs	Either a vector of receptors to include in the heatmap or 'all' for all receptors. If left NULL then the receptors selected in the signaling network connected to the features plotted will be shown.
mark_connections	Boolean indicating whether to add an 'x' in cells where there is a connected receptor or TF. Default FALSE.
...	Other parameters to pass to ComplexHeatmap::Heatmap() . Note that to use the 'main' parameter of ComplexHeatmap::Heatmap() you must set title = FALSE and to use 'annCol' or 'annColors' ann_cols must be FALSE.

Value

A heatmap rendered to the active graphics device

See Also

Heatmap Plotting Functions: `feat_heatmap()`, `incoming_signaling_heatmap()`, `signaling_heatmap()`

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
# basic usage
cor_heatmap(dom, title = "PBMC R-TF Correlations")
# show correlations above a specific value
cor_heatmap(dom, bool = TRUE, bool_thresh = 0.1)
# identify combinations that are connected
cor_heatmap(dom, bool = FALSE, mark_connections = TRUE)
```

---

cor_scatter *Create a correlation plot between TF and receptor*

Description

Create a correlation plot between transcription factor activity scores and receptor expression

Usage

```
cor_scatter(dom, tf, rec, remove_rec_dropout = NULL, ...)
```

Arguments

dom	A `domino()` object with network built (`build_domino()` has been run)
tf	Target TF for plotting AUC score
rec	Target receptor for plotting expression
remove_rec_dropout	Whether to remove cells with zero expression for plot. Default (NULL) will use the value of `remove_rec_dropout` used with `create_domino()` when the `domino()` object was created. If the `domino()` object was created before that parameter was added, it will default to FALSE. To override the default, set this parameter to TRUE or FALSE
...	Other parameters to pass to `ggpubr::ggscatter()`.

Value

A ggplot scatter plot rendered in the active graphics device

See Also

Other plotting: `circos_ligand_receptor()`, `plot_differential_linkages()`

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
cor_scatter(dom, "FLI1","CXCR3")
```

count_linkage *Count occurrences of linkages across multiple domino results from a linkage summary*

Description

Count occurrences of linkages across multiple domino results from a linkage summary

Usage

```
count_linkage(
  link_summary,
  cluster,
  group.by = NULL,
  linkage = "rec_lig",
  subject_names = NULL
)
```

Arguments

link_summary	a linkage_summary() object
cluster	the name of the cell cluster being compared across multiple domino results
group.by	the name of the column in linkage_summary@subject_meta by which to group subjects for counting. If NULL, only total counts of linkages for linkages in the cluster across all subjects is given.
linkage	a stored linkage from the domino() object. Can compare any of 'tfs', 'rec', 'incoming_lig', 'tfs_rec', or 'rec_lig'
subject_names	a vector of subject_names from the linkage_summary to restrict counting to. If NULL, all subjects in link_summary are included.

Value

A data frame with columns for the unique linkage features and the counts of how many times the linkage occurred across the compared domino results. If group.by is used, counts of the linkages are also provided as columns named by the unique values of the group.by variable.

See Also

summarize_linkages() to create linkage_summary() objects, test_differential_linkages() to compare links across groups

Differential Signaling Workflow: plot_differential_linkages(), summarize_linkages(), test_differential_l

Examples

```
data(LinkageSummary)
count_linkage(
  link_summary = LinkageSummary$linkage_sum_tiny, cluster = "C1",
  group.by = "group", linkage = "rec")
count_linkage(
  link_summary = LinkageSummary$linkage_sum_tiny, cluster = "C1",
  group.by = "group", linkage = "tfs_rec", subject_names = c("P1", "P2", "P4", "P5"))
```

`create_domino` *Create a* `domino()` *object and prepare it for network construction*

Description

This function reads in a receptor ligand signaling database, cell level features of some kind (ie. output from pySCENIC), z-scored single cell data, and cluster id for single cell data, calculates a correlation matrix between receptors and other features (this is transcription factor module scores if using pySCENIC), and finds features enriched by cluster. It will return a `domino()` object prepared for `build_domino()`, which will calculate a signaling network.

Usage

```
create_domino(
  rl_map,
  features,
  counts,
  z_scores,
  clusters,
  tf_targets = NULL,
  verbose = TRUE,
  use_complexes = TRUE,
  rec_min_thresh = 0.025,
  remove_rec_dropout = FALSE,
  tf_selection_method = "clusters",
  tf_variance_quantile = 0.5
)
```

Arguments

`rl_map`	Data frame where each row describes a receptor-ligand interaction with required columns gene_A & gene_B including the gene names for the receptor and ligand and type_A & type_B annotating if genes A and B are a ligand (L) or receptor (R). Each interaction must pair one receptor with one ligand (rows that do not, such as ligand-ligand or missing types) are excluded with a warning.
`features`	Either a path to a csv containing cell level features of interest (ie. the auc matrix from pySCENIC) or named matrix with cells as columns and features as rows.
`counts`	Counts matrix for the data. Used to threshold receptors based on expression (`rec_min_thresh`), to calculate percentage of clusters with receptor expression, and to remove cells with zero receptor counts if `remove_rec_dropout` is TRUE. Dense matrices are converted to sparse `dgCMatrix` for storage.
`z_scores`	Matrix containing z-scored expression data for all cells with cells as columns and features as rows. Receptor genes that pass `rec_min_thresh` but are not present in the z_scores matrix are excluded from correlation calculations with a warning.
`clusters`	Named factor containing cell cluster with names as cells.
`tf_targets`	Optional. A list where names are transcription factors and the stored values are character vectors of genes in the transcription factor's regulon.
`verbose`	Boolean indicating whether or not to print progress during computation.

use_complexes Boolean indicating whether you wish to use receptor/ligand complexes in the receptor ligand signaling database. If FALSE, receptor/ligand pairs where either functions as a protein complex will not be considered when constructing the signaling network.

rec_min_thresh Minimum expression level of receptors by cell. Default is 0.025 or 2.5 percent of all cells in the data set. This is important when calculating correlation to connect receptors to transcription activity. If this threshold is too low then correlation calculations will proceed with very few cells with non-zero expression.

remove_rec_dropout
Whether to remove receptors with 0 expression counts when calculating correlations. This can reduce false positive correlation calculations when receptors have high dropout rates.

tf_selection_method
Selection of which method to target transcription factors. Options:

- "clusters": TFs are selected based on differential activity by cluster using Wilcoxon rank-sum test.
- "variable": TFs are selected based on coefficient of variation (also known as relative standard deviation or normalized root-mean-square) across all cells in dataset.
- "all": All TFs provided in the `features` matrix are included in downstream analysis.

tf_variance_quantile
Quantile of coefficient of variation used to threshold features to take if `tf_selection_method` is `variable`. Features with coefficient of variation ranked above quantile are kept (default of 0.5 keeps the most variable half of features and higher numbers keep fewer features). Ignored if tf_selection_method is not 'variable'.

Value

A `domino()` object

See Also

`create_rl_map_cellphonedb()` for creating receptor-ligand maps, `create_regulon_list_scenic()` for creating regulon lists from pySCENIC output, and `build_domino()` for building signaling networks from `domino()` objects created by this function.

Examples

```
data(CellPhoneDB)
data(SCENIC)
data(PBMC)

pbmc_dom_tiny <- create_domino(
 rl_map = CellPhoneDB$rl_map_tiny, features = SCENIC$auc_tiny,
 counts = PBMC$count_tiny, z_scores = PBMC$zscore_tiny,
 clusters = PBMC$clusters_tiny, tf_targets = SCENIC$regulon_list_tiny,
 use_complexes = TRUE, remove_rec_dropout = FALSE,
 verbose = FALSE
 )
```

`create_regulon_list_scenic` *Create a list of genes in regulons inferred by SCENIC*

Description

Generates a list of transcription factors and the genes targeted by the transcription factor as part of their regulon inferred by pySCENIC

Usage

```
create_regulon_list_scenic(regulons)
```

Arguments

`regulons`	Data frame or file path to the table of the output of the ctx function from pySCENIC

Value

A list where names are transcription factors and the stored values are character vectors of genes in the inferred regulons

Examples

```
data(SCENIC)
regulon_list_tiny <- create_regulon_list_scenic(regulons = SCENIC$regulons_tiny)
```

`create_rl_map_cellphonedb` *Create a receptor-ligand map from a CellPhoneDB signaling database*

Description

Generates a data frame of receptor-ligand interactions from a CellPhoneDB database annotating the genes encoding the interacting ligands and receptors to be queried in transcriptomic data.

Usage

```
create_rl_map_cellphonedb(
  genes,
  proteins,
  interactions,
  complexes = NULL,
  database_name = "CellPhoneDB",
  gene_conv = NULL,
  gene_conv_host = "https://www.ensembl.org",
  alternate_convert = FALSE,
  alternate_convert_table = NULL
)
```

Arguments

genes	data frame or file path to table of gene names in uniprot, hgnc_symbol, or ensembl format in CellPhoneDB database format
proteins	data frame or file path to table of protein features in CellPhoneDB format
interactions	data frame or file path to table of protein-protein interactions in CellPhoneDB format
complexes	optional: data frame or file path to table of protein complexes in CellPhoneDB format
database_name	name of the database being used, stored in output
gene_conv	character vector of length 2 formatted as (from, to) or (source, target) if gene conversion to orthologs is desired; options are ENSMUSG, ENSG, MGI, or HGNC (for from) and MGI or HGNC (for to). Identifiers in the gene_name column of genes (gene symbols, such as HGNC for CellPhoneDB human databases). Interactions with any gene lacking an ortholog are skipped.
gene_conv_host	host for conversion; default ensembl, could also use mirrors if desired
alternate_convert	boolean if you would like to use a non-ensembl method of conversion (must supply table; not recommended, use only if ensembl is down)
alternate_convert_table	a data frame with column names corresponding to gene symbol types (mm.ens, hs.ens, mgi, hgnc) and rows corresponding to the gene symbols themselves for use with alternate_convert = TRUE

Value

Data frame where each row describes a possible receptor-ligand interaction

See Also

create_domino() where output of this function can be used

Examples

```
data(CellPhoneDB)
rl_map_tiny <- create_rl_map_cellphonedb(genes = CellPhoneDB$genes_tiny,
 proteins = CellPhoneDB$proteins_tiny,
 interactions = CellPhoneDB$interactions_tiny,
 complexes = CellPhoneDB$complexes_tiny)

## Not run:
rl_map_tiny_conv <- create_rl_map_cellphonedb(genes = CellPhoneDB$genes_tiny,
  proteins = CellPhoneDB$proteins_tiny,
  interactions = CellPhoneDB$interactions_tiny,
  gene_conv = c("HGNC", "MGI"),
  gene_conv_host = "https://beta.ensembl.org")

## End(Not run)

# Using alternate conversion table instead of biomaRt
# (this simplified table is for illustration only, this is
# NOT the correct way to convert between HGNC and MGI symbols)
hs_genes <- unique(CellPhoneDB$genes_tiny$gene_name)
```

```
ortho_table <- data.frame(
  hgnc = hs_genes,
  mgi = paste0(substr(hs_genes, 1, 1), tolower(substr(hs_genes, 2, nchar(hs_genes)))))

rl_map_tiny_alt <- create_rl_map_cellphonedb(genes = CellPhoneDB$genes_tiny,
  proteins = CellPhoneDB$proteins_tiny,
  interactions = CellPhoneDB$interactions_tiny,
  complexes = CellPhoneDB$complexes_tiny, gene_conv = c("HGNC", "MGI"),
  alternate_convert = TRUE, alternate_convert_table = ortho_table)
```

`domino-class` *The domino class*

Description

The domino class contains all information necessary to calculate receptor-ligand signaling. It contains z-scored expression, cell cluster labels, feature values, and a referenced receptor-ligand database formatted as a receptor-ligand map. Calculated intermediate values are also stored.

Value

An instance of class `domino`

Slots

`db_info` List of data sets from receptor-ligand database

`counts` Raw count gene expression data

`z_scores` Matrix of z-scored expression data with cells as columns

`clusters` Named factor with cluster identity of each cell

`features` Matrix of features (TFs) to correlate receptor-ligand expression with. Cells are columns and features are rows.

`cor` Correlation matrix of receptor expression to features.

`linkages` List of lists containing info linking cluster->tf->rec->lig

`clust_de` Data frame containing differential expression results for features by cluster.

`misc` List of miscellaneous info pertaining to run parameters etc., including the dominoSignal versions used to create (`create_version`) and build (`build_version`) the object.

`cl_signaling_matrices` Incoming signaling matrix for each cluster

`signaling` Signaling matrix between all clusters.

DominoObjects *Example domino objects*

Description

A list of two domino objects built from the example data subsets (PBMC, SCENIC, and CellPhoneDB).

dom_tiny A `domino()` object created using `create_domino()`.

built_dom_tiny The same object after running `build_domino()`.

Usage

```
data("DominoObjects")
```

Format

An object of class `list` of length 2.

Source

Generated with `data-raw/tiny_data.R`.

dom_clusters *Access clusters*

Description

A function to pull cluster information from a `domino()` object

Usage

```
dom_clusters(dom, labels = FALSE)
```

Arguments

`dom`	a `domino()` object that has been created with `create_domino()`
`labels`	A boolean for whether to return the cluster labels for each cell or the clusters used for inferring communication

Value

A vector containing either the names of the clusters used or factors of the cluster label for each individual cell

See Also

Other accessors: `dom_correlations()`, `dom_counts()`, `dom_database()`, `dom_de()`, `dom_info()`, `dom_linkages()`, `dom_network_items()`, `dom_signaling()`, `dom_tf_activation()`, `dom_zscores()`

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
cluster_names <- dom_clusters(dom)
cell_cluster_label <- dom_clusters(dom, labels = TRUE)
```

`dom_correlations` *Access correlations*

Description

A function to pull receptor-transcription factor correlations from a `domino()` object

Usage

```
dom_correlations(dom, type = "rl")
```

Arguments

`dom`	a `domino()` object that has been created with `create_domino()`
`type`	either "rl" or "complex", to select between the receptor-ligand or complex correlation matrix

Value

A matrix containing the correlation values for each receptor (row) by transcription factor (column)

See Also

Other accessors: `dom_clusters()`, `dom_counts()`, `dom_database()`, `dom_de()`, `dom_info()`, `dom_linkages()`, `dom_network_items()`, `dom_signaling()`, `dom_tf_activation()`, `dom_zscores()`

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
cor_matrix <- dom_correlations(dom, "rl")
```

dom_counts *Access counts*

Description

A function to pull gene expression from a `domino()` object

Usage

```
dom_counts(dom)
```

Arguments

dom	a `domino()` object that has been created with `create_domino()`

Value

A matrix containing the gene expression values for each gene (row) by cell (column)

See Also

Other accessors: `dom_clusters()`, `dom_correlations()`, `dom_database()`, `dom_de()`, `dom_info()`, `dom_linkages()`, `dom_network_items()`, `dom_signaling()`, `dom_tf_activation()`, `dom_zscores()`

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
counts <- dom_counts(dom)
```

dom_database *Access database*

Description

A function to pull database information from a `domino()` object

Usage

```
dom_database(dom, name_only = TRUE)
```

Arguments

dom	a `domino()` object that has been created
name_only	a boolean for whether to return only the name of the database used or the entire database that is stored. Default TRUE.

Value

A vector of unique databases used in building the domino() object OR a data frame that includes the database information used in the domino() object creation

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_de(), dom_info(), dom_linkages(), dom_network_items(), dom_signaling(), dom_tf_activation(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
database_name <- dom_database(dom)
full_database <- dom_database(dom, name_only = FALSE)
```

---

dom_de *Access differential expression*

Description

A function to pull differential expression p-values from a domino() object

Usage

```
dom_de(dom)
```

Arguments

dom	a domino() object that has been created with create_domino()

Value

A matrix containing the p-values for differential expression of transcription factors (rows) in each cluster (columns)

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_database(), dom_info(), dom_linkages(), dom_network_items(), dom_signaling(), dom_tf_activation(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
de_mat <- dom_de(dom)
```

dom_info *Access build information*

Description

A function to pull the parameters used when running build_domino() from a domino() object

Usage

```
dom_info(dom)
```

Arguments

dom	a domino() object that has been created with create_domino()

Value

A list containing booleans for whether the object has been created and built, the parameters used in create_domino() and build_domino() to infer the signaling network, and the dominoSignal versions used to create and build the object (NULL if not recorded)

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_database(), dom_de(), dom_linkages(), dom_network_items(), dom_signaling(), dom_tf_activation(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
build_details <- dom_info(dom)
```

dom_linkages *Access linkages*

Description

A function to pull linkages from a domino() object

Usage

```
dom_linkages(
  dom,
  link_type = c("complexes", "receptor-ligand", "tf-target", "tf-receptor", "receptor",
    "incoming-ligand"),
  by_cluster = FALSE
)
```

Arguments

dom	a domino() object that has been created with create_domino()
link_type	one value (out of "complexes", "receptor-ligand", "tf-target", "tf-receptor", "receptor", "incoming-ligand") used to select the desired type of linkage. Note that "receptor" and "incoming-ligand" are only available when by_cluster is set to TRUE.
by_cluster	A boolean to indicate whether to return linkages overall or by cluster

Value

A list containing linkages between some combination of receptors, ligands, transcription factors, and clusters

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_database(), dom_de(), dom_info(), dom_network_items(), dom_signaling(), dom_tf_activation(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
complexes <- dom_linkages(dom, "complexes")
tf_rec_by_cluster <- dom_linkages(dom, "tf-receptor", TRUE)
```

---

dom_network_items *Access all features, receptors, or ligands present in a signaling network.*

Description

This function collates all of the features, receptors, or ligands found in a signaling network anywhere in a list of clusters. This can be useful for comparing signaling networks across two separate conditions. In order to run this build_domino() must be run on the object previously.

Usage

```
dom_network_items(dom, clusters = NULL, which_return = NULL)
```

Arguments

dom	a domino() object containing a signaling network (build_domino() was run)
clusters	vector indicating clusters to collate network items from. If left as NULL then all clusters will be included.
which_return	string indicating whether to collate "features", "receptors", or "ligands". If NULL then a list of all three will be returned.

Value

A vector containing all features, receptors, or ligands in the data set or a list containing all three.

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_database(), dom_de(), dom_info(), dom_linkages(), dom_signaling(), dom_tf_activation(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
monocyte_receptors <- dom_network_items(dom, "CD14_monocyte", "receptors")
all_tfs <- dom_network_items(dom, which_return = "features")
```

---

dom_signaling *Access signaling*

Description

A function to pull signaling matrices from a domino() object dom_signaling() and dom_signalling() are synonyms.

Usage

```
dom_signaling(dom, cluster = NULL)

dom_signalling(dom, cluster = NULL)
```

Arguments

dom	a domino() object that has been created with create_domino()
cluster	either NULL to indicate global signaling or a specific cluster for which a signaling matrix is desired

Value

A data frame containing the signaling score through each ligand (row) by each cluster (column) OR . a data frame containing the global summed signaling scores between receptors (rows) and ligands (columns) of each cluster

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_database(), dom_de(), dom_info(), dom_linkages(), dom_network_items(), dom_tf_activation(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
monocyte_signaling <- dom_signaling(dom, cluster = "CD14_monocyte")
```

dom_tf_activation *Access transcription factor activity scores*

Description

A function to pull transcription factor activity scores from a domino() object

Usage

```
dom_tf_activation(dom)
```

Arguments

dom	a domino() object that has been created with create_domino()

Value

A matrix containing the transcription factor activity scores for each TF (row) by cell (column)

See Also

Other accessors: dom_clusters(), dom_correlations(), dom_counts(), dom_database(), dom_de(), dom_info(), dom_linkages(), dom_network_items(), dom_signaling(), dom_zscores()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
tf_activation <- dom_tf_activation(dom)
```

dom_to_df *Turn domino object signaling information into a data frame*

Description

Constructs a data frame of ligand-receptor-TF signaling triplets from a built domino() object. Expression values are averaged across cells within each cluster (ligands from sending clusters, receptors and TFs from receiving clusters).

Usage

```
dom_to_df(dom, send_clusters = NULL, rec_clusters = NULL, exp_type = "counts")
```

Arguments

dom	A built domino() object (as output by build_domino())
send_clusters	Character/factor vector of cluster names for ligand signals. If NULL (default), uses all clusters in dom.
rec_clusters	Character/factor vector of cluster names for receptor/TF signals. If NULL (default), uses all clusters in dom.
exp_type	Either "counts" or "z_scores" for expression type

Value

Data frame with columns: ligand, receptor, transcription_factor, ligand_exp, rec_exp, tf_auc, sending_cl, receiving_cl. Each row is a ligand-receptor-TF triplet from a sender-to-receiver cluster pair with corresponding mean expression values. Empty data frame returned if no valid interactions found.

Examples

```
data("DominoObjects")
dom <- DominoObjects$built_dom_tiny
df <- dom_to_df(dom, exp_type = "z_scores")
head(df)
```

---

dom_zscores *Access z-scores*

Description

A function to pull z-scored expression from a `domino()` object

Usage

```
dom_zscores(dom)
```

Arguments

dom	a `domino()` object that has been created with `create_domino()`

Value

A matrix containing the z-scored gene expression values for each gene (row) by cell (column)

See Also

Other accessors: `dom_clusters()`, `dom_correlations()`, `dom_counts()`, `dom_database()`, `dom_de()`, `dom_info()`, `dom_linkages()`, `dom_network_items()`, `dom_signaling()`, `dom_tf_activation()`

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
zscores <- dom_zscores(dom)
```

do_norm *Normalize a matrix to its max value by row or column*

Description

Normalizes a matrix to its max value by row or column

Usage

```
do_norm(mat, dir)
```

Arguments

mat	Matrix to be normalized
dir	Direction to normalize the matrix (either "row" for row or "col" for column)

Value

A normalized matrix in the direction specified.

feat_heatmap *Create a heatmap of features organized by cluster*

Description

Creates a heatmap of transcription factor activity scores by cells grouped by cluster.

Usage

```
feat_heatmap(
  dom,
  feats = NULL,
  bool = FALSE,
  bool_thresh = 0.2,
  title = TRUE,
  norm = FALSE,
  cols = NULL,
  ann_cols = TRUE,
  min_thresh = NULL,
  max_thresh = NULL,
  ...
)
```

Arguments

dom	A domino() object with network built (build_domino())
feats	Either a vector of features to include in the heatmap or 'all' for all features. If left NULL then the features selected for the signaling network will be shown.
bool	Boolean indicating whether the heatmap should be continuous or boolean. If boolean then bool_thresh will be used to determine how to define activity as positive or negative.
bool_thresh	Numeric indicating the threshold separating 'on' or 'off' for feature activity if making a boolean heatmap.
title	Either a string to use as the title or a boolean describing whether to include a title. In order to pass the 'main' parameter to ComplexHeatmap::Heatmap() you must set title to FALSE.
norm	Boolean indicating whether or not to normalize the transcription factors to their max value.
cols	Named vector of colors to annotate cells by cluster color. Values are taken as colors and names as cluster. If left as NULL then default ggplot colors will be generated.
ann_cols	Boolean indicating whether to include cell cluster as a column annotation. Colors can be defined with cols. If FALSE then custom annotations can be passed to ComplexHeatmap::Heatmap().
min_thresh	Minimum threshold for color scaling if not a boolean heatmap
max_thresh	Maximum threshold for color scaling if not a boolean heatmap
...	Other parameters to pass to ComplexHeatmap::Heatmap(). Note that to use the 'main' parameter of ComplexHeatmap::Heatmap() you must set title = FALSE and to use 'annCol' or 'annColors' ann_cols must be FALSE.

Value

A heatmap rendered to the active graphics device

See Also

Heatmap Plotting Functions: cor_heatmap(), incoming_signaling_heatmap(), signaling_heatmap()

Examples

```
# basic usage
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
feat_heatmap(dom)
# using thresholds
feat_heatmap(
    dom,
    min_thresh = 0.1,
    max_thresh = 0.6,
    bool = FALSE
)
```

filter_by_subject_names *Restrict a linkage_summary object to a set of subject names*

Description

Restrict a linkage_summary object to a set of subject names

Usage

```
filter_by_subject_names(link_summary, subject_names)
```

Arguments

link_summary	a linkage_summary() object
subject_names	a vector of subject_names to restrict link_summary to, or NULL to leave link_summary unchanged

Value

link_summary, subset to the given subject_names, or unchanged if subject_names is NULL

gene_network *Create a gene association network*

Description

Create a gene association network for genes from a given cluster. The selected cluster acts as the receptor for the gene association network, so only ligands, receptors, and features associated with the receptor cluster will be included in the plot.

Usage

```
gene_network(
  dom,
  clust = NULL,
  OutgoingSignalingClust = NULL,
  class_cols = c(lig = "#FF685F", rec = "#47a7ff", feat = "#39C740"),
  cols = NULL,
  lig_scale = 1,
  layout = "grid",
  ...
)
```

Arguments

dom	A domino() object with network built (build_domino())
clust	Receptor cluster to create the gene association network for. A vector of clusters may be provided. This is a required parameter.
OutgoingSignalingClust	Vector of clusters to plot the outgoing signaling from
class_cols	Named vector of colors used to color classes of vertices. Values must be colors and names must be classes ('rec', 'lig', and 'feat' for receptors, ligands, and features).
cols	Named vector of colors for individual genes. Genes not included in this vector will be colored according to class_cols.
lig_scale	FALSE or a numeric value to scale the size of ligand vertices based on z-scored expression in the data set.
layout	Type of layout to use. Options are 'grid', 'random', 'sphere', 'circle', 'fr' for Fruchterman-Reingold force directed layout, and 'kk' for Kamada Kawai for directed layout.
...	Other parameters to pass to plot() with an igraph object. See igraph manual for options.

Value

An igraph plot rendered to the active graphics device. A list with the igraph object (graph) and the layout matrix (layout) is returned invisibly, or NULL if no signaling is found for the clusters.

See Also

Network Plotting Functions: signaling_network()

Examples

```
# basic usage
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
gene_network(
    dom,
    clust = "CD8_T_cell",
    OutgoingSignalingClust = "CD14_monocyte"
)
```

---

get_ligand_expression *Get ligand expression matrix for outgoing clusters*

Description

Get ligand expression matrix for outgoing clusters

Usage

```
get_ligand_expression(dom, send_clusters, lig_genes, complexes, exp_type)
```

Arguments

dom	A built domino() object (as output by build_domino())
send_clusters	Character or factor vector of cluster names for which to compute outgoing ligand signals
lig_genes	Character vector of ligand gene names; should intersect with expression matrix rownames
complexes	Named list where names are complex identifiers and values are character vectors of component genes. Used to aggregate expression for multi-gene complexes. Empty list is acceptable. The output of get_resolved_ligands() can be used here.
exp_type	Character of length 1: either "counts" or "z_scores" to specify expression type

Value

Data frame with a ligand column and then one column per send_cluster with mean expression values. Ligands not found are dropped with a message, complexes with missing components are dropped, and send_clusters with no cells are omitted.

---

get_resolved_ligands *Get ligands with resolved names*

Description

Get ligands with resolved names

Usage

```
get_resolved_ligands(dom)
```

Arguments

dom	A built domino() object (as output by build_domino())

Value

A named list with two elements:

lig_names	Character vector of unique ligand gene names or aliases
complex_names	Named list mapping complex names to component gene vectors

get_signaling_info *Get receptor-ligand signaling information*

Description

Get receptor-ligand signaling information

Usage

```
get_signaling_info(dom, rec_clusters, cl_ligands_sub, exp_type)
```

Arguments

dom	A built domino() object (as output by build_domino())
rec_clusters	Character or factor vector of cluster names for which to compute incoming receptor/TF signals
cl_ligands_sub	Data frame with columns 'ligand', 'cluster', 'mean_counts'; using reshape2::melt() on the output of get_ligand_expression() is a convenient way to get this
exp_type	Character of length 1: either "counts" or "z_scores" to specify expression type

Value

Data frame with columns: ligand, receptor, transcription_factor, ligand_exp, rec_exp, tf_auc, sending_cl, receiving_cl. Each row represents a ligand-receptor-TF triplet with receiver cluster, and corresponding expression values. Returns empty data frame if no interactions found.

ggplot_col_gen *Generate ggplot colors*

Description

Accepts a number of colors to generate and generates a ggplot color spectrum.

Usage

```
ggplot_col_gen(n)
```

Arguments

n	Number of colors to generate

Value

A vector of colors according to ggplot color generation.

`incoming_signaling_heatmap` *Create a cluster incoming signaling heatmap*

Description

Creates a heatmap of a cluster incoming signaling matrix. Each cluster has a list of ligands capable of activating its enriched transcription factors. The function creates a heatmap of cluster average expression for all of those ligands. A list of all cluster incoming signaling matrices can be found in the cl_signaling_matrices slot of a `domino()` object as an alternative to this plotting function. `incoming_signaling_heatmap()` and `incoming_signalling_heatmap()` are synonyms.

Usage

```
incoming_signaling_heatmap(
  dom,
  rec_clust,
  clusts = NULL,
  min_thresh = -Inf,
  max_thresh = Inf,
  scale = "none",
  normalize = "none",
  title = TRUE,
  ...
)

incoming_signalling_heatmap(
  dom,
  rec_clust,
  clusts = NULL,
  min_thresh = -Inf,
  max_thresh = Inf,
  scale = "none",
  normalize = "none",
  title = TRUE,
  ...
)
```

Arguments

`dom`	A `domino()` object with network built (`build_domino()`)
`rec_clust`	Which cluster to select as the receptor. Must match naming of clusters in the `domino()` object.
`clusts`	Vector of clusters to be included. If NULL then all clusters are used.
`min_thresh`	Minimum signaling threshold for plotting. Defaults to -Inf for no threshold.
`max_thresh`	Maximum signaling threshold for plotting. Defaults to Inf for no threshold.
`scale`	How to scale the values (after thresholding). Options are 'none', 'sqrt' for square root, or 'log' for log10(x+1).

normalize	Options to normalize the matrix. Accepted inputs are 'none' for no normalization, 'rec_norm' to normalize to the maximum value with each receptor cluster, or 'lig_norm' to normalize to the maximum value within each ligand cluster
title	Either a string to use as the title or a boolean describing whether to include a title. In order to pass the 'main' parameter to ComplexHeatmap::Heatmap() you must set title to FALSE.
...	Other parameters to pass to ComplexHeatmap::Heatmap(). Note that to use the 'column_title' parameter of ComplexHeatmap::Heatmap() you must set title = FALSE

Value

a Heatmap rendered to the active graphics device

See Also

Heatmap Plotting Functions: cor_heatmap(), feat_heatmap(), signaling_heatmap()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
# incoming signaling of the CD8  T cells
incoming_signaling_heatmap(dom, "CD8_T_cell")
```

---

lc *Pool items from list into vector*

Description

Helper function to convert from a nested series of lists to a single vector.

Usage

```
lc(list, list_names)
```

Arguments

list	List to pull items from
list_names	Names of items in list to pool

Value

A vector containing all items in the list by list_names

LinkageSummary *Example linkage summary*

Description

A list containing a mock linkages summary across six subjects in two groups, and the result of running `test_differential_linkages()` on it.

Usage

```
data("LinkageSummary")
```

Format

A list of:

linkage_sum_tiny A `linkage_summary()` object created with mock data.

linkage_diff_tiny Results from running `test_differential_linkages()` for receptors (`linkage = "rec"`) in cluster `"C1"`, grouped by `"group"`.

Source

Generated with `data-raw/tiny_data.R`.

linkage_summary-class *The domino linkage summary class*

Description

The linkage summary class contains linkages established in multiple domino objects through gene regulatory network inference and reference to receptor- ligand databases. A data frame summarizing meta features that describe the `domino()` objects compared in the linkage summary facilitates comparisons of established linkages and differential signaling interactions across categorical sample covariates.

Value

an instance of class `linkage_summary`

Slots

`subject_names` unique names for each domino result included in the summary

`subject_meta` data.frame with each row describing one subject and columns describing features of the subjects by which to draw comparisons of signaling networks. The first column holds the subject names, in the same order as `subject_names`.

`subject_linkages` nested list of linkages inferred for each subject, named by and in the same order as `subject_names`. Lists are stored in a hierarchical structure of subject-cluster-linkages, including transcription factors (tfs), links between transcription factors and receptors (tfs_rec), active receptors (rec), links between receptors and ligands(rec_lig), and incoming ligands (incoming_lig)

`mean_exp_by_cluster` *Get average expression for a set of genes over cluster(s)*

Description

Get average expression for a set of genes over cluster(s)

Usage

```
mean_exp_by_cluster(dom, clusts, genes)
```

Arguments

`dom`	A `domino()` object
`clusts`	Cluster(s) for which we want to get average expression
`genes`	The genes for which we want to get average expression

Value

A dataframe of genes x clusters, values are z-scores averaged over the clusters

`mean_ligand_expression` *Calculate mean ligand expression as a data frame for plotting in circos plot*

Description

Creates a data frame of mean ligand expression for use in plotting a circos plot of ligand expression and saving tables of mean expression.

Usage

```
mean_ligand_expression(x, ligands, cell_ident, cell_barcodes, destination)
```

Arguments

`x`	Gene by cell expression matrix
`ligands`	Character vector of ligand genes to be quantified
`cell_ident`	Vector of cell type (identity) names for which to calculate mean ligand gene expression
`cell_barcodes`	Vector of cell barcodes (colnames of x) belonging to cell_ident to calculate mean expression across
`destination`	Name of the receptor with which each ligand interacts

Value

A data frame of ligand expression targeting the specified receptor

Examples

```
data(DominoObjects)
counts <- dom_counts(DominoObjects$built_dom_tiny)
mean_exp <- mean_ligand_expression(counts,
    ligands = c("PTPRC", "FASLG"), cell_ident = "CD14_monocyte",
    cell_barcodes = colnames(counts), destination = "FAS"
)
```

---

obtain_circos_expression

Obtain Circos Expression

Description

Pull expression data from a domino() object and format for plotting as a receptor-oriented circos plot.

Usage

```
obtain_circos_expression(
  dom,
  receptor,
  ligands,
  ligand_expression_threshold = 0.01,
  cell_idents = NULL
)
```

Arguments

dom	A domino() object that has undergone network building with build_domino()
receptor	Name of a receptor active in at least one cell type in the domino() object
ligands	Character vector of ligands capable of interaction with the receptor
ligand_expression_threshold	Minimum mean expression value of a ligand by a cell type for a chord to be rendered between the cell type and the receptor
cell_idents	Vector of clusters from cluster assignments in the domino() object to be included in the plot.

Value

a data frame where each row describes plotting parameters of receptor-ligand interactions to pass to render_circos_ligand_receptor()

PBMC *PBMC scRNAseq data subset*

Description

A subset of the preprocessed 10x Genomics PBMC 3k scRNAseq data.

Usage

```
data("PBMC")
```

Format

A list of:

count_tiny Subset of the sparse counts matrix (`dgCMatrix`), genes x cells.

zscore_tiny Subset of z-scored expression matrix, genes x cells.

clusters_tiny Factor of cell-type labels named by cell barcode.

Source

https://doi.org/10.5281/zenodo.10124865 (file `pbmc3k_sce.rds`); formatted with `data-raw/tiny_data.R`.

`plot_differential_linkages` *Plot differential linkages among domino results ranked by a comparative statistic*

Description

Plot differential linkages among domino results ranked by a comparative statistic

Usage

```
plot_differential_linkages(
  differential_linkages,
  test_statistic,
  stat_range = c(0, 1),
  stat_ranking = c("ascending", "descending"),
  group_palette = NULL,
  gradient = c(minimum = "red", maximum = "gray90")
)
```

Arguments

differential_linkages	a data frame output from the test_differential_linkages() function
test_statistic	column name of differential_linkages where the test statistic used for ranking linkages is stored (p.value, p.adj, odds.ratio)
stat_range	a two value vector of the minimum and maximum values of test_statistic for plotting linkage features
stat_ranking	whether to rank the features by ascending or descending values of test_statistic
group_palette	a named vector of colors to use for each group being compared
gradient	a named two value vector of colors to use for the minimum and maximum values of test_statistic

Value

A heatmap-class object of features ranked by test_statistic annotated with the proportion of subjects that showed active linkage of the features.

See Also

Other plotting: circos_ligand_receptor(), cor_scatter()

Differential Signaling Workflow: count_linkage(), summarize_linkages(), test_differential_linkages()

Examples

```
data(LinkageSummary)
plot_differential_linkages(
    differential_linkages = LinkageSummary$linkage_diff_tiny,
    test_statistic = "p.value",
    stat_range = c(0, 1),
    stat_ranking = "ascending"
)
```

---

print,domino-method *Print domino object*

Description

Prints a summary of a domino() object

Usage

```
## S4 method for signature 'domino'
print(x, ...)
```

Arguments

x	A domino() object
...	Additional arguments to be passed to other methods

Value

A printed description of the number of cells and clusters in the domino() object

Examples

```
data(DominoObjects)
print(DominoObjects$built_dom_tiny)
```

---

print,linkage_summary-method *Print linkage summary object*

Description

Prints a description of a linkage summary object

Usage

```
## S4 method for signature 'linkage_summary'
print(x, ...)
```

Arguments

x	A linkage summary object
...	Additional arguments to be passed to other methods

Value

A printed description of the number of subjects, groups, and clusters in the linkage summary object

Examples

```
data(LinkageSummary)
print(LinkageSummary$linkage_sum_tiny)
```

---

read_if_char *Read in data if an object looks like path to it*

Description

Read in data if an object looks like path to it

Usage

```
read_if_char(obj)
```

Arguments

obj	object to read if not already object

Value

Object itself or data read in from path

`rename_clusters` *Renames clusters in a* `domino()` *object*

Description

This function renames the clusters used to build a `domino()` object

Usage

```
rename_clusters(dom, clust_conv, warning = FALSE)
```

Arguments

`dom`	a `domino()` object to rename clusters in
`clust_conv`	named vector of conversions from old to new clusters. Values are taken as new clusters IDs and names as old cluster IDs.
`warning`	logical. If TRUE, will warn if a cluster is not found in the conversion table. Default is FALSE.

Value

A `domino()` object with clusters renamed in all applicable slots.

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
new_clust <- c(
    "CD8_T_cell" = "CD8+ T Cells",
    "CD14_monocyte" = "CD14+ Monocytes", "B_cell" = "B Cells"
)
pbmc_dom_built_tiny <- rename_clusters(dom, new_clust)
```

`render_circos_ligand_receptor`

Render Circos Ligand Receptor Plot

Description

Renders a circos plot from the output of `obtain_circos_expression()` to the active graphics device

Usage

```
render_circos_ligand_receptor(
  signaling_df,
  receptor,
  cell_colors = NULL,
  ligand_expression_threshold = 0.01
)
```

Arguments

signaling_df	Data frame output from obtain_circos_expression()
receptor	Name of a receptor active in at least one cell type in the domino() object
cell_colors	Named vector of color names or hex codes where names correspond to the plotted clusters and the color values
ligand_expression_threshold	Minimum mean expression value of a ligand by a cell type for a chord to be rendered between the cell type and the receptor

Value

a circlize plot is rendered to the active graphics device

---

resolve_complexes *Convert between complex names and gene names*

Description

Convert between complex names and gene names

Usage

```
resolve_complexes(dom, genes)
```

Arguments

dom	A domino() object
genes	A vector of genes, some of which may be complexes

Value

A list where any complexes are mapped to a vector of component genes. The list names are set to the input gene names.

resolve_names *Convert between ligand names and gene names*

Description

Convert between ligand names and gene names

Usage

```
resolve_names(dom, genes)
```

Arguments

dom	A domino() object
genes	A vector of genes on which to resolve ligand and gene names

Value

A vector of names where ligand names have been replaced with gene names if applicable

SCENIC *SCENIC example subset*

Description

A subset of pySCENIC output for the PBMC example data.

Usage

```
data("SCENIC")
```

Format

A list of:

auc_tiny Matrix of AUC scores from pySCENIC for a subset of TFs and cells.

regulons_tiny Data frame subset of the pySCENIC regulon table.

regulon_list_tiny The pySCENIC regulons formatted as a named list by create_regulon_list_scenic().

Source

https://doi.org/10.5281/zenodo.10124865 (files auc_pbmc_3k.csv, regulons_pbmc_3k.csv); formatted with data-raw/tiny_data.R

show,domino-method *Show domino object information*

Description

Shows content overview of a domino() object

Usage

```
## S4 method for signature 'domino'
show(object)
```

Arguments

object	A domino() object

Value

A printed description of cell numbers and clusters in the object

Examples

```
data(DominoObjects)
show(DominoObjects$built_dom_tiny)
```

show,linkage_summary-method *Show linkage_summary object information*

Description

Shows content overview of linkage_summary object

Usage

```
## S4 method for signature 'linkage_summary'
show(object)
```

Arguments

object	A linkage_summary object

Value

A printed description of the number of subjects, groups, and clusters in the linkage summary object

Examples

```
data(LinkageSummary)
LinkageSummary$linkage_sum_tiny
show(LinkageSummary$linkage_sum_tiny)
```

signaling_heatmap *Create a network heatmap*

Description

Creates a heatmap of the signaling network. Alternatively, the network matrix can be accessed directly in the signaling slot of a `domino()` object using the `dom_signaling()` function. `signaling_heatmap()` and `signalling_heatmap()` are synonyms.

Usage

```
signaling_heatmap(
  dom,
  clusts = NULL,
  min_thresh = -Inf,
  max_thresh = Inf,
  scale = "none",
  normalize = "none",
  ...
)

signalling_heatmap(
  dom,
  clusts = NULL,
  min_thresh = -Inf,
  max_thresh = Inf,
  scale = "none",
  normalize = "none",
  ...
)
```

Arguments

dom	A `domino()` object with network built (`build_domino()`)
clusts	vector of clusters to be included. If NULL then all clusters are used.
min_thresh	minimum signaling threshold for plotting. Defaults to -Inf for no threshold.
max_thresh	maximum signaling threshold for plotting. Defaults to Inf for no threshold.
scale	how to scale the values (after thresholding). Options are 'none', 'sqrt' for square root, or 'log' for log10(x+1).
normalize	options to normalize the matrix. Normalization is done after thresholding and scaling. Accepted inputs are 'none' for no normalization, 'rec_norm' to normalize to the maximum value with each receptor cluster, or 'lig_norm' to normalize to the maximum value within each ligand cluster
...	other parameters to pass to `ComplexHeatmap::Heatmap()`

Value

A heatmap rendered to the active graphics device

See Also

Heatmap Plotting Functions: cor_heatmap(), feat_heatmap(), incoming_signaling_heatmap()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
# basic usage
signaling_heatmap(dom)
# scale
signaling_heatmap(dom, scale = "sqrt")
# normalize
signaling_heatmap(dom, normalize = "lig_norm")
```

---

signaling_network *Create a cluster to cluster signaling network diagram*

Description

Creates a network diagram of signaling between clusters. Nodes are clusters and directed edges indicate signaling from one cluster to another. Edges are colored based on the color scheme of the ligand expressing cluster signaling_network() and signalling_network() are synonyms.

Usage

```
signaling_network(
  dom,
  cols = NULL,
  edge_weight = 0.3,
  clusts = NULL,
  showOutgoingSignalingClusts = NULL,
  showIncomingSignalingClusts = NULL,
  min_thresh = -Inf,
  max_thresh = Inf,
  normalize = "none",
  scale = "sq",
  layout = "circle",
  scale_by = "rec_sig",
  vert_scale = 3,
  plot_title = NULL,
  ...
)

signalling_network(
  dom,
  cols = NULL,
  edge_weight = 0.3,
  clusts = NULL,
  showOutgoingSignalingClusts = NULL,
  showIncomingSignalingClusts = NULL,
```

```
  min_thresh = -Inf,
  max_thresh = Inf,
  normalize = "none",
  scale = "sq",
  layout = "circle",
  scale_by = "rec_sig",
  vert_scale = 3,
  plot_title = NULL,
  ...
)
```

Arguments

dom	A domino() object with network built (build_domino())
cols	named vector indicating the colors for clusters. Values are colors and names must match clusters in the domino() object. If left as NULL then ggplot colors are generated for the clusters
edge_weight	weight for determining thickness of edges on plot. Signaling values are multiplied by this value
clusts	vector of clusters to be included in the network plot
showOutgoingSignalingClusts	vector of clusters to plot the outgoing signaling from
showIncomingSignalingClusts	vector of clusters to plot the incoming signaling on
min_thresh	minimum signaling threshold. Values lower than the threshold will be set to the threshold. Defaults to -Inf for no threshold
max_thresh	maximum signaling threshold for plotting. Values higher than the threshold will be set to the threshold. Defaults to Inf for no threshold
normalize	options to normalize the signaling matrix. Accepted inputs are 'none' for no normalization, 'rec_norm' to normalize to the maximum value with each receptor cluster, or 'lig_norm' to normalize to the maximum value within each ligand cluster
scale	how to scale the values (after thresholding). Options are 'none', 'sqrt' for square root, 'log' for log10(x+1), or 'sq' for square
layout	type of layout to use. Options are 'random', 'sphere', 'circle', 'fr' for Fruchterman-Reingold force directed layout, and 'kk' for Kamada Kawai for directed layout
scale_by	how to size vertices. Options are 'lig_sig' for summed outgoing signaling, 'rec_sig' for summed incoming signaling, and 'none'. In the former two cases the values are scaled with asinh after summing all incoming or outgoing signaling. Vertices with no incoming/outgoing signaling due to other parameters are given a size of 0.
vert_scale	integer used to scale size of vertices with or without variable scaling from scale_by parameter.
plot_title	text for the plot's title.
...	other parameters to be passed to plot when used with an igraph object.

Value

An igraph plot rendered to the active graphics device

See Also

Network Plotting Functions: gene_network()

Examples

```
data(DominoObjects)
dom <- DominoObjects$built_dom_tiny
# basic usage
signaling_network(dom, edge_weight = 2)
# scaling, thresholds, layouts, selecting clusters
signaling_network(
    dom,
    showOutgoingSignalingClusts = "CD14_monocyte",
    scale = "none", normalize = "none", layout = "fr", scale_by = "none",
    vert_scale = 5, edge_weight = 2
)
```

---

subset,linkage_summary-method

Subset a linkage_summary object

Description

Subsets a linkage summary object by subject names or metadata

Usage

```
## S4 method for signature 'linkage_summary'
subset(x, subset)
```

Arguments

x	A linkage_summary object
subset	A logical expression referencing subject_names or columns of subject_meta

Value

A linkage_summary object containing only the subjects that match the specified criteria.

Examples

```
data(LinkageSummary)
links <- LinkageSummary$linkage_sum_tiny
subset(links, subset = !subject_names %in% c("P1", "P2"))
subset(links, subset = group == "G1")
subset(links, subset = subject_names %in% c("P1", "P2", "P4") | group == "G2")
```

summarize_linkages *Summarize linkages from multiple domino objects*

Description

Creates a linkage_summary() object storing the linkages learned in different domino() objects as nested lists to facilitate comparisons of networks learned by domino() across subject covariates.

Usage

```
summarize_linkages(domino_results, subject_meta, subject_names = NULL)

summarise_linkages(domino_results, subject_meta, subject_names = NULL)
```

Arguments

domino_results	list of analysis results with one domino() object per subject. Names from the list should match subject_names. Each object used must have a signaling network built with build_domino().
subject_meta	data frame that includes the subject features by which the objects could be grouped. The first column must be subject names. Rows are reordered to match order of subject_names.
subject_names	vector of subject names in domino_results. If NULL, defaults to first column of subject_meta.

Value

A linkage_summary() object consisting of nested lists of the active transcription factors, active receptors, and incoming ligands for each cluster across multiple domino results

See Also

Differential Signaling Workflow: count_linkage(), plot_differential_linkages(), test_differential_linkag

Examples

```
data(PBMC)
data(SCENIC)
data(CellPhoneDB)
data(DominoObjects)

# create alternative clustering by shuffling cluster assignments
clusters_tiny_alt <- setNames(
    PBMC$clusters_tiny[c(121:240, 1:120, 241:360)],
    names(PBMC$clusters_tiny)
)
clusters_tiny_alt <- as.factor(clusters_tiny_alt)

# build an alternative domino object
pbmc_dom_tiny_alt <- create_domino(
    rl_map = CellPhoneDB$rl_map_tiny,
    features = SCENIC$auc_tiny,
```

```
    counts = PBMC$count_tiny,
    z_scores = PBMC$zscore_tiny,
    clusters = clusters_tiny_alt,
    tf_targets = SCENIC$regulon_list_tiny,
    use_complexes = TRUE,
    remove_rec_dropout = FALSE,
    verbose = FALSE
)

pbmc_dom_built_tiny_alt <- build_domino(
    dom = pbmc_dom_tiny_alt,
    max_tf_pval = .05,
    max_tf_per_clust = Inf,
    max_rec_per_tf = Inf,
    rec_tf_cor_threshold = .1,
    min_rec_percentage = 0.01
)

# create a list of domino objects
dom_ls <- list(
    dom1 = DominoObjects$built_dom_tiny,
    dom2 = pbmc_dom_built_tiny_alt
)

# compare the linkages across the two domino objects
meta_df <- data.frame("ID" = c("dom1", "dom2"), "group" = c("A", "B"))
summarize_linkages(
    domino_results = dom_ls, subject_meta = meta_df,
    subject_names = meta_df$ID
)
```

`table_convert_genes` *Convert genes using a table*

Description

Takes a vector of gene inputs and a conversion table and returns a converted gene table

Usage

```
table_convert_genes(genes, from, to, conversion_table)
```

Arguments

`genes`	the genes to convert
`from`	gene symbol type of the input (one of ENSG, ENSMUSG, HGNC, MGI)
`to`	desired gene symbol type for the output (one of HGNC, MGI)
`conversion_table`	a data frame with column names corresponding to gene symbol types (mm.ens, hs.ens, mgi, hgnc) and rows corresponding to the gene symbols themselves

Value

A data frame of genes with original and corresponding converted symbols

`test_differential_linkages` *Statistical test for differential linkages across multiple domino results*

Description

Statistical test for differential linkages across multiple domino results

Usage

```
test_differential_linkages(
  link_summary,
  cluster,
  group.by,
  linkage = "rec_lig",
  subject_names = NULL
)
```

Arguments

`link_summary`	a `linkage_summary()` object
`cluster`	the name of the cell cluster being compared across multiple `domino()` results
`group.by`	the name of the column in `linkage_summary@subject_meta` by which to group subjects for counting.
`linkage`	a stored linkage from the `domino()` object. Can compare any of 'tfs', 'rec', 'incoming_lig', 'tfs_rec', or 'rec_lig'
`subject_names`	a vector of subject_names from the linkage_summary to restrict testing to. If NULL, all subjects in `link_summary` are included.

Value

A data frame of results from the test of the differential linkages. Rows correspond to each linkage tested. Columns correspond to:

- 'cluster' : the name of the cell cluster being compared
- 'linkage' : the type of linkage being compared
- 'group.by' : the grouping variable
- 'test_name' : the test used for comparison (Fisher's exact test)
- 'feature' : individual linkages compared
- 'test statistics' : an odds ratio (for 2 groups), p-value, and fdr-adjusted p-value.
- 'total_count' : total number of subjects where the linkage is active
- 'X_count' : number of subjects in each category of group.by (X) where the linkage is active
- 'total_n' : number of total subjects compared
- 'X_n' : total number of subjects in each category of group.by (X)

See Also

`summarize_linkages()` to create `linkage_summary()` objects

Differential Signaling Workflow: `count_linkage()`, `plot_differential_linkages()`, `summarize_linkages()`

Examples

```
data(LinkageSummary)
tiny_differential_linkage_c1 <- test_differential_linkages(
    link_summary = LinkageSummary$linkage_sum_tiny, cluster = "C1", group.by = "group",
    linkage = "rec", subject_names = c("P1", "P2", "P4", "P5")
)
```

Index

∗ **accessors**
dom_clusters, 17
dom_correlations, 18
dom_counts, 19
dom_database, 19
dom_de, 20
dom_info, 21
dom_linkages, 21
dom_network_items, 22
dom_signaling, 23
dom_tf_activation, 24
dom_zscores, 25
∗ **datasets**
CellPhoneDB, 5
DominoObjects, 17
LinkageSummary, 34
PBMC, 37
SCENIC, 42
∗ **differentials**
count_linkage, 11
plot_differential_linkages, 37
summarize_linkages, 48
test_differential_linkages, 50
∗ **heatmaps**
cor_heatmap, 9
feat_heatmap, 26
incoming_signaling_heatmap, 32
signaling_heatmap, 44
∗ **internal**
avg_exp_for_complexes, 4
check_arg, 6
conv_py_bools, 8
convert_genes, 8
do_norm, 26
filter_by_subject_names, 28
get_ligand_expression, 29
get_resolved_ligands, 30
get_signaling_info, 31
ggplot_col_gen, 31
lc, 33
mean_exp_by_cluster, 35
obtain_circos_expression, 36
read_if_char, 39
render_circos_ligand_receptor, 40
resolve_complexes, 41
resolve_names, 42
table_convert_genes, 49
∗ **networks**
gene_network, 28
signaling_network, 45
∗ **plotting**
circos_ligand_receptor, 7
cor_scatter, 10
plot_differential_linkages, 37

add_rl_column, 3
avg_exp_for_complexes, 4

build_domino, 4
build_domino(), *7*, *9, 10*, *12, 13*, *17*, *21, 22*, *24*, *27*, *29–32*, *44*, *46*, *48*

CellPhoneDB, 5, *17*
check_arg, 6
circos_ligand_receptor, 7
circos_ligand_receptor(), *10*, *38*
ComplexHeatmap::Heatmap(), *9*, *27*, *33*, *44*
conv_py_bools, 8
convert_genes, 8
cor_heatmap, 9
cor_heatmap(), *27*, *33*, *45*
cor_scatter, 10
cor_scatter(), *7*, *38*
count_linkage, 11
count_linkage(), *38*, *48*, *50*
create_domino, 12
create_domino(), *4, 5*, *10*, *15*, *17–25*
create_regulon_list_scenic, 14
create_regulon_list_scenic(), *13*, *42*
create_rl_map_cellphonedb, 14
create_rl_map_cellphonedb(), *6*, *13*

do_norm, 26
dom_clusters, 17
dom_clusters(), *18–25*
dom_correlations, 18
dom_correlations(), *17*, *19–25*

dom_counts, 19
dom_counts(), *17, 18*, *20–25*
dom_database, 19
dom_database(), *17–25*
dom_de, 20
dom_de(), *17–25*
dom_info, 21
dom_info(), *17–20*, *22–25*
dom_linkages, 21
dom_linkages(), *17–21*, *23–25*
dom_network_items, 22
dom_network_items(), *17–25*
dom_signaling, 23
dom_signaling(), *17–25*, *44*
dom_signalling *(*dom_signaling*)*, 23
dom_signalling(), *23*
dom_tf_activation, 24
dom_tf_activation(), *17–23*, *25*
dom_to_df, 24
dom_zscores, 25
dom_zscores(), *17–24*
domino *(*domino-class*)*, 16
domino(), *4, 5*, *7*, *9–13*, *17–25*, *27*, *29–32*, *34–36*, *38–44*, *46*, *48*, *50*
domino-class, 16
DominoObjects, 17

feat_heatmap, 26
feat_heatmap(), *10*, *33*, *45*
filter_by_subject_names, 28

gene_network, 28
gene_network(), *47*
get_ligand_expression, 29
get_ligand_expression(), *31*
get_resolved_ligands, 30
get_resolved_ligands(), *30*
get_signaling_info, 31
ggplot_col_gen, 31
ggpubr::ggscatter(), *10*

incoming_signaling_heatmap, 32
incoming_signaling_heatmap(), *10*, *27*, *32*, *45*
incoming_signalling_heatmap *(*incoming_signaling_heatmap*)*, 32
incoming_signalling_heatmap(), *32*

lc, 33
linkage_summary *(*linkage_summary-class*)*, 34
linkage_summary(), *11*, *28*, *34*, *48*, *50*
linkage_summary-class, 34
LinkageSummary, 34

mean_exp_by_cluster, 35
mean_ligand_expression, 35

obtain_circos_expression, 36
obtain_circos_expression(), *40, 41*

PBMC, *17*, 37
plot_differential_linkages, 37
plot_differential_linkages(), *7*, *10, 11*, *48*, *50*
print,domino-method, 38
print,linkage_summary-method, 39

read_if_char, 39
rename_clusters, 40
render_circos_ligand_receptor, 40
reshape2::melt(), *31*
resolve_complexes, 41
resolve_names, 42

SCENIC, *17*, 42
show,domino-method, 43
show,linkage_summary-method, 43
signaling_heatmap, 44
signaling_heatmap(), *10*, *27*, *33*, *44*
signaling_network, 45
signaling_network(), *29*, *45*
signalling_heatmap *(*signaling_heatmap*)*, 44
signalling_heatmap(), *44*
signalling_network *(*signaling_network*)*, 45
signalling_network(), *45*
subset,linkage_summary-method, 47
summarise_linkages *(*summarize_linkages*)*, 48
summarize_linkages, 48
summarize_linkages(), *11*, *38*, *50*

table_convert_genes, 49
test_differential_linkages, 50
test_differential_linkages(), *11*, *34*, *38*, *48*